Implementation Guide for protecting

Linux/Unix/Solaris
Pam Modules

Trademarks

BlackShield ID, BlackShield ID SBE and BlackShield ID Pro are either registered trademarks or trademarks of CRYPTOCard Inc. All other trademarks and registered trademarks are the property of their owners.

Additional Information, Assistance, or Comments

CRYPTOCard's technical support specialists can provide assistance when planning and implementing CRYPTOCard in your network. In addition to aiding in the selection of the appropriate authentication products, CRYPTOCard can suggest deployment procedures that provide a smooth, simple transition from existing access control systems and a satisfying experience for network users. We can also help you leverage your existing network equipment and systems to maximize your return on investment.

CRYPTOCard works closely with channel partners to offer worldwide Technical Support services. If you purchased this product through a CRYPTOCard channel partner, please contact your partner directly for support needs.

To contact CRYPTOCard directly:

International Voice: +1-613-599-2441

North America Toll Free: 1-800-307-7042

support@cryptocard.com

For information about obtaining a support contract, see our Support Web page at http://www.cryptocard.com.

Related Documentation

Refer to the Support & Downloads section of the CRYPTOCard website for additional documentation and interoperability guides: http://www.cryptocard.com.

Publication History

Date	Changes	Version
Nov. 4, 2009	Document created	1.0

Table of Contents

Overview

PAM modules can be used in *Nix environments to provide an additional level of security within a given service or application which is PAM aware. This document contains procedures as well as general advice on augmenting current **authentication** mechanisms with strong two-factor one-time passwords.

Applicability

This integration guide is applicable to:

Summary	
Product Name	Linux/Unix/Solaris PAM Modules
Vendor Site	N/A
Supported Client Software	N/A
Authentication Method	RADIUS
Supported BlackShield ID Pro Agent Functionality	
BlackShield ID Authentication	RADIUS (PAP)
Authentication Mode	One-time password Challenge-response BlackShield ID Pro static password
New PIN Mode	User-changeable Alphanumeric 3-16 digit PIN User-changeable Numeric 3-16 digit PIN Server-changeable Alphanumeric 3-16 digit PIN Server-changeable Numeric 3-16 digit PIN

CRYPTOCard Server	
Authentication Server	BlackShield ID
Version	Professional Edition 2.3+

CRYPTOCard Agent	
Agent	N/A
Authentication Methods Supported	N/A
Version	N/A
Operating System 32-bit	Any PAM aware Linux/Unix/Solaris OS
Operating System 64-bit	Any PAM aware Linux/Unix/Solaris OS

Assumptions

BlackShield ID has been installed and configured and a “Test” user account can be selected in the Assignment Tab.

Operation

The RADIUS PAM module adds an additional level to authentication for any service or application which is PAM aware. The user would attempt to authenticate to the service or application, and would be prompted for their user name and password. Their password would be authenticated by the BlackShield ID server rather than the local passwd file.

Preparation and Prerequisites

1. You have downloaded the RADIUS PAM files from the FreeRadius website: http://freeradius.org/pam_radius_auth/
   * This URL is a 3rd website. It could change at any time.
2. You have extracted the contents of the tar.gz file to a directory of your choosing and that you have sufficient privileges to read and write to them
3. The Linux/Solaris username and the CRYPTOCard Token/username must be identical.
4. Even though CRYPTOCard authentication is being used, the CRYPTOCard user must have an account on the Linux/Solaris system in order for the users to connect. When a user logs on to a Unix system, the system reads the passwd file to find the user’s group, default shell and home directory. If this information does not exist, the user will fail to authenticate**.** This condition does not apply if NIS/NIS+/NFS or LDAP is being used.
5. An application must be PAM-aware in order to use a PAM module. The most common PAM configuration files are login, ftp, and sshd. Please consult the application’s documentation to determine if it is PAM-aware.

Installation

This section deals primarily with the installation of the FreeRADIUS PAM module. For more in-depth information on PAM configuration and a description of other modules please visit

http://www.us.kernel.org/pub/linux/libs/pam/

The Linux PAM modules are located in:	**/lib/security**
The Linux application-specific configuration files are in:	**/etc/pam.d**
The Solaris PAM modules are located in:	**/usr/lib/security**
The Solaris application-specific configuration files are in:	**/etc/pam.conf**

** Please review the Preparation and Prerequisites section before proceeding in the installation instructions.

You must download the following package before proceeding with this guide.
FreeRADIUS client download URL: http://freeradius.org/pam_radius_auth/

Compiling the PAM module

Login as root. Extract the package to a temporary location then browse to this temporary location. In order to compile the module for your system type:

```
make
```

On Linux copy pam_radius_auth.so to /lib/security as pam_radius_auth.so:

```
cp pam_radius_auth.so  /lib/security/pam_radius_auth.so
```

On Solaris copy pam_radius_auth.so to /usr/lib/security as pam_radius_auth.so.1

```
cp pam_radius_auth.so  /usr/lib/security/pam_radius_auth.so.1
```

(If you receive any errors while compiling, please see the **Troubleshooting** Section)

Server Configuration File (RADIUS)

When the FreeRADIUS PAM module is used it searches for a file called **server** in the **/etc/raddb** directory. This file contains the location of the RADIUS servers, the shared secret, and the order in which each RADIUS server will be checked. A generic server configuration file called **pam_radius_auth.conf** can be found in the FreeRADIUS module source directory which you extracted to a temporary location.

This file must be renamed and placed into the **/etc/raddb** directory.

Verify that an **/etc/raddb** directory exists. If it does not type:

```
mkdir /etc/raddbcd
```

Now, copy the generic server configuration file over to the **/etc/raddb** directory by going into the freeRADIUS PAM module source directory and typing:

```
mv pam_radius_auth.conf  /etc/raddb/server
```

Below is an example of the default server configuration file. Blank lines or lines beginning with # are considered as comments or simply ignored.

```
# pam_radius_auth configuration file.  Save as: /etc/raddb/server
# server[:port]       shared_secret         timeout (s)

127.0.0.1:1812        testing123            1
192.168.21.4:1812     testing123            3

# having localhost in your radius configuration is a Good Thing
# See the INSTALL file for pam.conf hints
```

The columns are as follows:

Server	[:port]	shared_secret	timeout (default 3 seconds)

The timeout field controls the time the module waits before deciding if the server has failed to respond. This setting is optional.

If multiple RADIUS Server lines exist, they are tried in order. If the server fails to respond, it is skipped and the next server is used.

A RADIUS port **must** be specified in the server file.

Check your BlackShield ID RADIUS Server port settings within Microsoft IAS, to verify you have entered the correct port within the /etc/raddb/server config file.

- Open Internet Authentication Service (IAS)
- Right click on Internet Authentication Service, and select Properties
- Click the Ports tab, and verify which authentication port(s) are used

Note: Official RADIUS port numbers 1812 or 1645.

Securing the RADIUS Server Configuration

Once the server file is completed, it MUST be secured in order to prevent tampering. The following procedure will secure the server file.

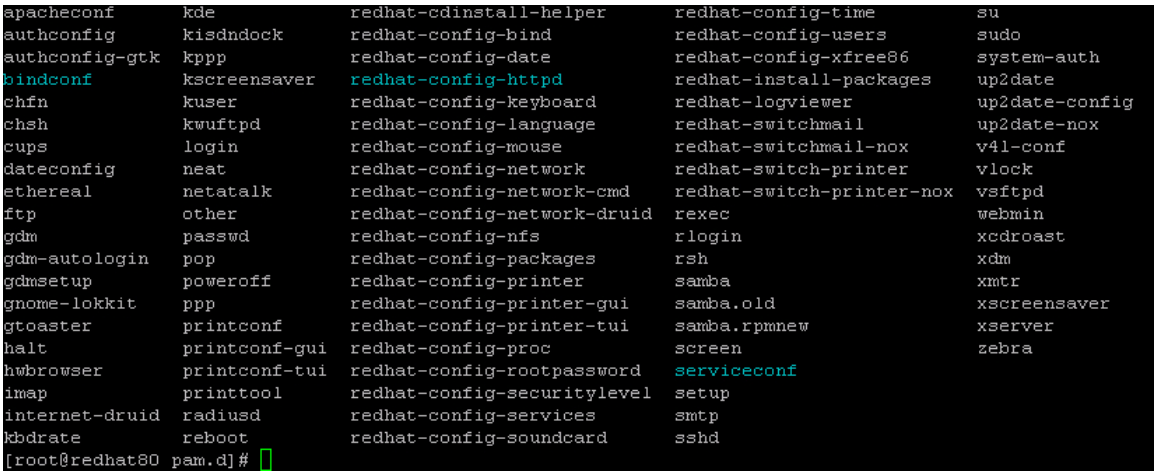
```
chown root /etc/raddb/

chmod go -rwx /etc/raddb

chmod go -rwx /etc/raddb/server
```

Configuring application-specific configuration files

The last step in setting up the FreeRADIUS PAM module is to configure the PAM-aware application you would like to implement. All the applications listed in the **/etc/pam.d** directory or the pam.conf file are PAM-aware. CRYPTOCard only offers support for the PAM aware application listed in the Linux and Solaris PAM Configuration examples section listed below. In theory, the CRYPTOCard module will work for most applications in the **pam.d** directory.



(To see a complete listing of PAM module types, control flags and arguments please see the **Troubleshooting** section).

Linux - PAM Configuration Examples

The following examples are for several PAM aware applications. PAM aware applications (su, halt, reboot etc…) that require **root** authentication should not use CRYPTOCard authentication. CRYPTOCard authentication should only be used for End Users or administrators with root privileges.

CRYPTOCard only supports the configuration examples outlined in this section.

Login / Telnet

The LOGIN PAM file affects telnet and local console login sessions. The following LOGIN PAM configuration file enables CRYPTOCard authentication.

#%PAM-1.0		
auth	**required**	**/lib/security/pam_radius_auth.so**
auth	required	/lib/security/pam_nologin.so
account	required	/lib/security/pam_stack.so service=system-auth
account	required	/lib/security/pam_permit.so
password	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_stack.so service=system-auth
session	optional	/lib/security/pam_console.so

Telnet and console login sessions support challenge response. To enable challenge response, make the following configuration changes to the LOGIN PAM file.

#%PAM-1.0		
auth	**required**	**/lib/security/pam_radius_auth.so skip_password**
auth	required	/lib/security/pam_nologin.so
account	required	/lib/security/pam_stack.so service=system-auth
account	required	/lib/security/pam_permit.so
password	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_stack.so service=system-auth
session	optional	/lib/security/pam_console.so

The following is an example of a Telnet session using challenge response.

```
Red Hat Linux release 8.0 (Psyche)
Kernel 2.4.18-14 on an i686
login: redhat8
Challenge: 70806383
Enter Response: 517-2311
Last login: Wed Apr  9 15:20:26 from 192.168.10.117
[redhat8@redhat80 redhat8]$
```

FTP / VSFTPD

The FreeRADIUS PAM module should only be used to enforce strong user authentication for real user accounts. Using token based passwords for anonymous access is not supported because it would impede the user from connecting anonymously.

FTP		
#%PAM-1.0		
auth	required	**/lib/security/pam_listfile.so item=user sense=deny file=/etc/ftpusers onerr=succeed**
auth	required	/lib/security/pam_radius_auth.so
auth	required	/lib/security/pam_shells.so
account	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_stack.so service=system-auth

VSFTPD		
#%PAM-1.0		
auth	**required**	**/lib/security/pam_listfile.so item=user sense=deny file=/etc/vsftpd.ftpusers onerr=succeed**
auth	required	/lib/security/pam_radius_auth.so
auth	required	/lib/security/pam_shells.so
account	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_stack.so service=system-auth

Note: The FTP protocol does not support challenge response.

SSHD (OpenSSH)

For security reasons and compatibility with the FreeRADIUS PAM module you must have at least SSH2 version 2.4 for F-Secure or SSH2 version 2.9 for OpenSSH.

Note: CRYPTOCard will only provide support for versions of OpenSSH/OpenSSL included with RedHat or any updates provided by RedHat.

#%PAM-1.0		
auth	**required**	**/lib/security/pam_radius_auth.so**
auth	required	/lib/security/pam_nologin.so
account	required	/lib/security/pam_stack.so service=system-auth
password	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_limits.so
session	optional	/lib/security/pam_console.so

The SSH protocol supports challenge response. To enable challenge response, make the following configuration changes to the SSHD PAM configuration file.

#%PAM-1.0		
auth	**required**	**/lib/security/pam_radius_auth.so skip_password**
auth	required	/lib/security/pam_nologin.so
account	required	/lib/security/pam_stack.so service=system-auth
password	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_limits.so
session	optional	/lib/security/pam_console.so

For challenge response the following changes must also be made to the sshd_config file:

```
PasswordAuthentication no
PermitEmptyPasswords no
ChallengeResponseAuthentication yes
PAMAuthenticationViaKbdInt yes
UsePrivilegeSeparation no
```

Note: UsePrivilegeSeparation was introduced to address a challenge response vulnerability in the SSHD daemon. Older versions of the OpenSSH sshd_config file will not include this setting.

KDE / GDM / XDM (Graphical Desktop Logon)

BlackShield ID authentication can be enabled for users who logon to KDE or Gnome. In theory, any Desktop manager is supported, as they will most likely use XDM, GDM, or KDE as their logon manager. The following changes to either the XDM, GDM or KDE PAM configuration file enables CRYPTOCard authentication.

#%PAM-1.0		
auth	**required**	**/lib/security/pam_radius_auth.so**
auth	required	/lib/security/pam_nologin.so
account	required	/lib/security/pam_stack.so service=system-auth
password	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_stack.so service=system-auth
session	optional	/lib/security/pam_console.so

CRYPTOCard does not support using challenge response with graphical logon.

Note: To globally enable a graphical logon on startup edit the /etc/ inittab. Change the "**id:3:initdefault:**" entry to "**id:5:initdefault:**".

XSCREENSAVER

If CRYPTOCard authentication is being enforce for KDE, XDM or GDM CRYPTOCard authentication for xscreensaver should also be enabled. The following configuration will enforce CRYPTOCard authentication.

#%PAM-1.0		
auth	**required**	**/lib/security/pam_radius_auth.so**

POP / POPS / IMAP / SMTP

The POP, POPS, IMAP and SMTP daemon can be configured to perform CRYPTOCard authentication each time a user performs a send and/or receive request. The following conditions must be taking into account before implementing CRYPTOCard authentication:

- The end user can no longer use the "Save Password" option in their email client.
- Every time the email client checks for new email, the end user will be prompted to enter a one-time password. If possible, increase the mail retrieval setting to 30 minutes or higher (POP/POPS/IMAP only).
- Every time the email client sends an email, the end user will be prompted to enter a one-time password (SMTP only).
- Challenge response is not supported.

The following changes to the POP/IMAP PAM configuration file will enable CRYPTOCard authentication.

#%PAM-1.0		
auth	**required**	**/lib/security/pam_radius_auth.so**
account	required	/lib/security/pam_stack.so service=system-auth

The following changes to the SMTP PAM configuration file will enable CRYPTOCard authentication.

#%PAM-1.0		
auth	**required**	**/lib/security/pam_radius_auth.so**
account	required	/lib/security/pam_stack.so service=system-auth

PPP

This section assumes you already have Linux setup as a dialup server. For information on how to setup a Linux dialup server, please read the PPP-HOWTO. Challenge response for PPP is not supported. In order to get PAM and PPPD up and running you must make the following changes to the PPP PAM configuration file.

#%PAM-1.0		
auth	required	/lib/security/pam_nologin.so
auth	required	/lib/security/pam_radius_auth.so
account	required	/lib/security/pam_stack.so service=system-auth
session	required	/lib/security/pam_stack.so service=system-auth

In the **/etc/ppp/options** file you must at least have

```
+pap
-chap
lock
asyncmap 0
crtscts
modem
debug
```

kdebug 7
login

Do not include the “auth” argument in the options file. In the **/etc/mgetty+sendmail/login.config** file make the following adjustments:

```
/AutoPPP/ -      a_ppp /usr/sbin/pppd file /etc/ppp/options
*     -     -    /bin/login @
#*    -     -    /usr/sbin/pppd @
```

PAM module types, control flags and arguments

(This information was gathered from the PAM Administrators Guide. It can be found online at http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam.html or in the Solaris Man Pages)

A general configuration line has the following form:

module-type	control-flag	module-path	arguments
auth	required	/lib/security/pam_nologin.so	try_first_pass

Module-type: One of (currently) four types of module. The four types are as follows:

Auth: This module type provides two aspects of authenticating the user. Firstly, it establishes that the user is who they claim to be, by instructing the application to prompt the user for a password or other means of identification. Secondly, the module can grant group membership (independently of the /etc/groups file discussed above) or other privileges through its credential granting properties.

Account: This module performs non-authentication based account management. It is typically used to restrict/permit access to a service based on the time of day, currently available system resources (maximum number of users) or perhaps the location of the applicant user---`root' login only on the console.

Session: *Primarily, this module is associated with doing things that need to be done for the user before/after they can be given service. Such things include the logging of information concerning the opening/closing of some data exchange with a user, mounting directories, etc.*

Password: *This last module type is required for updating the authentication token associated with the user. Typically, there is one module for each `challenge/response' based authentication (auth) module-type.*

Control flags: The control-flag is used to indicate how the Linux-PAM library will react to the success or failure of the module it is associated with. Since modules can be stacked (modules of the same type execute in series, one after another), the control-flags determine the relative importance of each module.

Required: This indicates that the success of the module is required for the module-type facility to succeed. Failure of this module will not be apparent to the user until all of the remaining modules (of the same module-type) have been executed.

Requisite: Like required, however, in the case that such a module returns a failure, control is directly returned to the application. The return value is that associated with the first required or requisite module to fail. Note this flag can be used to protect against the possibility of a user getting the opportunity to enter a password over an unsafe medium. It is conceivable that such behavior might inform an attacker of valid accounts on a system. This possibility should be weighed against the insignificant concerns of exposing a sensitive password in a hostile environment.

Sufficient: The success of this module is deemed `sufficient' to satisfy the PAM library that this module-type has succeeded in its purpose. In the event that no previous required module has, failed, no more `stacked' modules of this type are invoked. (Note, in this case subsequent required modules are not invoked.). A failure of this module is not deemed as fatal to satisfying the application that this module-type has succeeded.

Optional: As its name suggests, this control-flag marks the module as not being critical to the success or failure of the user's application for service. In general, Linux-PAM ignores such a module when determining if the module stack will succeed or fail. However, in the absence of any definite successes or failures of previous or subsequent stacked modules this module will determine the nature of the response to the application. One example of this latter case is when the other modules return something like PAM IGNORE.

Arguments: Not all of these options are relevant for all uses of the module.

use_first_pass: Instead of prompting the user for a password, retrieve the password from the previous authentication module. If the password does not exist, return failure. If the password exists, try it, returning success/failure as appropriate.

try_first_pass: Instead of prompting the user for a password, retrieve the password from the previous authentication module. If the password exists, try it, and return success if it passes. If there was no previous password, or the previous password fails authentication, prompt the user with "Enter RADIUS password: ", and ask for another password. Try this password, and return success/failure as appropriate. This is the default for authentication.

skip_passwd: *Do not prompt for a password, even if there was none retrieved from the previous layer. Send the previous one (if it exists), or else send a NULL password. If this*

fails, exit. If an Access-Challenge is returned, display the challenge message, and ask the user for the response. Return success/failure as appropriate. The password sent to the next authentication module will NOT be the response to the challenge. If a password from a previous authentication module exists, it is passed on. Otherwise, no password is sent to the next module.

conf=foo: Set the configuration filename to 'foo'. Default is **/etc/raddb/server**

client_id=bar: Send a NAS-Identifier RADIUS attribute with string 'bar'. If the client_id is not specified, the PAM_SERVICE type is used instead. ('login', 'su', 'passwd', etc.) This feature may be disabled by using 'client_id='. i.e. A blank client ID.

use_authtok: Force the use of a previously entered password. This is needed for pluggable password strength checking i.e. try cracklib to be sure it's secure, then go update the RADIUS server.

accounting_bug: When used, the accounting response vector is NOT validated. This option will probably only be necessary on old (i.e. Livingston 1.16) servers.

Linux - Example and description of an application configuration file

Login\Telnet		
#%PAM-1.0		
auth	required	/lib/security/pam_securetty.so
auth	**sufficient**	**/lib/security/pam_radius_auth.so**
auth	required	/lib/security/pam_pwdb.so try_first_pass shadow nullok
auth	required	/lib/security/pam_nologin.so
account	required	/lib/security/pam_pwdb.so
password	required	/lib/security/pam_cracklib.so
password	required	/lib/security/pam_pwdb.so nullok use_authtok md5 shadow
session	required	/lib/security/pam_pwdb.so
session	optional	/lib/security/pam_console.so

The **first line** allows root to log in from certain areas. All other users are ignored by it. (By default root cannot telnet or ftp into a system).

The **second line** asks the user for their CRYPTOCard password. It then checks with the RADIUS server, if this passes, the user is given a token. Since this is flagged as sufficient, if the user's password works, PAM skips down to the 5th line, if not PAM moved down to the next line.

The **third line** takes the password that was supplied in line two and runs it's checks on it. If it passes, then it gives it is ok. If the password does not pass (or in the case of root, one wasn't asked for) then the module asks the user for a password. It then runs this new password through it's tests.

The **fourth line** checks to see if the nologin file exists. If it does, then only root is allowed to login. This is for letting root do maintenance without having to remain in single user mode.

The **fifth line** checks the status of the users account. It might do anything from warn them that their password is about to expire, not let them in if their account has expires, or simply be silent and let the user in.

The **sixth line** does a password check and tells the user if their password isn't very good.

The **seventh line** updates any password authentication associated with that user.

The **eighth line** simply logs the username and service-type to syslog.

The **ninth line** authorizes any console programs. As you can see this is optional and won't stop anything, though it does send out a warning if it doesn't pass.

If you encounter a problem that cannot be solved using the tips above, contact support@cryptocard.com or call us at (800) 307-7042 or +1-613-599-2441 Monday through Friday 8:30 am to 5:00 pm EST.

Troubleshooting

Compiling the modules returns an error.

While compiling if you receive a make error, you will have to edit the Makefile to remove the GNU make directives 'ifeq', 'else', etc. You may want to consider getting a more recent version of GNUmake.

The IAS server and/or BlackShield ID Manger Snapshot does not see the requests

Check the **/etc/raddb/server** file. Make sure that the ip address; port and secret are set up the same as the port and secret between CRYPTOAdmin and you RADIUS Server.

Accessing a locked out Linux system

If the system has been configured to use mixed mode authentication and the RADIUS servers become inaccessible the linux/unix server can only be accessed from single user mode.

Enabling Single mode on a RedHat system.

RedHat can be configured to use one of two boot managers; **Lilo** and **Grub**.

Lilo

At the Lilo prompt, you can hit the **<Tab>** key to show the list of possible choices. If Lilo is not configured to be interactive, press and hold the **<Alt>** or **<Shift>** key before the "**Lilo**'' message appears. Type a name from the list followed by **single** or **–s**.

linux single or **linux –s**

Make the necessary modifications to the system.

Grub

If you have a GRUB password configured, type **p** and enter the password. Select the version of the kernel that you wish to boot and type **e** for edit. You will be presenting with a list of items in the configuration file for the title you just selected. Select the line that starts with **kernel** and type **e** to edit the line.

kernel /boot/vmlinuz-2.4.18-14 ro root=LABEL=/

Go to the end of the line and type **single** as a separate word. Press the Enter Key to exit edit mode.

kernel /boot/vmlinuz-2.4.18-14 ro root=LABEL=/ single

Back at the GRUB screen, type **b** to boot into single user mode.

Make the necessary modifications to the system.

Failed Logon attempts

Symptom:	Login Failed
Indication:	11/19/2008 12:36:49 PM Henry Authentication Failure 312191514 192.168.2.1 Invalid OTP
Possible Causes:	The One Time Password provided for the user is incorrect.
Solution:	• Resync the token from the BlackShield ID manager • Ensure the user is using the correct

Symptom:	Login Failed
Indication:	11/19/2008 12:47:24 PM Henry Authentication Failure 312191514 192.168.2.1 Invalid PIN
Possible Causes:	The PIN provided for the user is incorrect.
Solution:	Within the BlackShield ID manager reset the PIN from the '**Secured Users'** tab.