One Identity Authentication Services 4.2.3

Smart Cards Administration Guide

Copyright 2020 One Identity LLC.

ALL RIGHTS RESERVED.

This guide contains proprietary information protected by copyright. The software described in this guide is furnished under a software license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of the applicable agreement. No part of this guide may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of One Identity LLC .

The information in this document is provided in connection with One Identity products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of One Identity LLC products. EXCEPT AS SET FORTH IN THE TERMS AND CONDITIONS AS SPECIFIED IN THE LICENSE AGREEMENT FOR THIS PRODUCT, ONE IDENTITY ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ONE IDENTITY BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ONE IDENTITY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. One Identity makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. One Identity does not make any commitment to update the information contained in this document.

If you have any questions regarding your potential use of this material, contact:

One Identity LLC.
Attn: LEGAL Dept
4 Polaris Way
Aliso Viejo, CA 92656

Refer to our Web site (http://www.OneIdentity.com) for regional and international office information.

Patents

One Identity is proud of our advanced technology. Patents and pending patents may apply to this product. For the most current information about applicable patents for this product, please visit our website at http://www.OneIdentity.com/legal/patents.aspx.

Trademarks

One Identity and the One Identity logo are trademarks and registered trademarks of One Identity LLC. in the U.S.A. and other countries. For a complete list of One Identity trademarks, please visit our website at www.OneIdentity.com/legal. All other trademarks are the property of their respective owners.

Legend

WARNING: A WARNING icon highlights a potential risk of bodily injury or property damage, for which industry-standard safety precautions are advised. This icon is often associated with electrical hazards related to hardware.

CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.

Authentication Services Smart Cards Administration Guide
Updated - August 2020
Version - 4.2.3

Contents

Privileged Access Suite for Unix

Unix security simplified

Privileged Access Suite for Unix solves the intrinsic security and administration issues of Unix-based systems (including Linux and macOS) while making satisfying compliance requirements easier. It unifies and consolidates identities, assigns individual accountability, and enables centralized reporting for user and administrator access to Unix. The Privileged Access Suite for Unix combines an Active Directory bridge and root delegation solutions under a unified console that grants organizations centralized visibility and streamlined administration of identities and access rights across their entire Unix environment.

Active Directory bridge

Achieve unified access control, authentication, authorization, and identity administration for Unix, Linux, and macOS systems by extending them into Active Directory (AD) and taking advantage of AD's inherent benefits. Patented technology allows non-Windows resources to become part of the AD trusted realm, and extends AD's security, compliance, and Kerberos-based authentication capabilities to Unix, Linux, and macOS. See www.oneidentity.com/products/authentication-services/ for more information about the Active Directory Bridge product.

Root delegation

The Privileged Access Suite for Unix offers two different approaches to delegating the Unix root account. The suite either *enhances* or *replaces* sudo, depending on your needs.

- By choosing to enhance sudo, you will keep everything you know and love about sudo while enhancing it with features like a central sudo policy server, centralized keystroke logs, a sudo event log, and compliance reports for who can do what with sudo.

  See www.oneidentity.com/products/privilege-manager-for-sudo/ for more information about enhancing sudo.
- By choosing to replace sudo, you will still be able to delegate the Unix root privilege based on centralized policy reporting on access rights, but with a more granular permission and the ability to log keystrokes on all activities from the time a user logs

in, not just the commands that are prefixed with "sudo." In addition, this option implements several additional security features like restricted shells, remote host command execution, and hardened binaries that remove the ability to escape out of commands and gain undetected elevated access.

See www.oneidentity.com/products/privilege-manager-for-unix/ for more information about replacing sudo.

Privileged Access Suite for Unix

Privileged Access Suite for Unix offers two editions: *Standard* edition and *Advanced* edition. Both editions include the Management Console for Unix, a common mangement console that provides a consolidated view and centralized point of management for local Unix users and groups; and Authentication Services, patented technology that allows organizations to extend the security and compliance of Active Directory to Unix, Linux, and macOS platforms and enterprise applications. In addition:

- The *Standard* edition licenses you for Privilege Manager for Sudo.
- The *Advanced* edition licenses you for Privilege Manager for Unix.

One Identity recommends that you follow these steps:

1. Install Authentication Services on one machine, so you can set up your Active Directory Forest.
2. Install Management Console for Unix, so you can perform all the other installation steps from the mangement console.
3. Add and profile hosts using the mangement console.
4. Configure the console to use Active Directory.
5. Deploy client software to remote hosts.

   Depending on which Privileged Access Suite for Unix edition you have purchased, deploy one of the following:

   - **Privilege Manager for Unix** software (that is, Privilege Manager Agent packages)

     -OR-

   - **Privilege Manager for Sudo** software (that is, Sudo Plugin packages)

About this guide

The *Authentication Services for Smart Cards Administration Guide* is intended for Windows, Unix, Linux, and macOS system administrators, network administrators, consultants, analysts, and any other IT professionals who will be installing and configuring Authentication Services for Smart Cards on the supported platforms. It describes the following:

- Basic Concepts
    - Supported platforms
    - Supported cards and readers
- Installation Prerequisites
- Installing Authentication Services for Smart Cards software
- Configuring the vendor's PKCS#11 driver
- Testing your configuration
- Enabling smart card login for selected services
- Troubleshooting Authentication Services for Smart Cards

NOTE: The term "Unix" is used informally throughout the Authentication Services documentation to denote any operating system that closely resembles the trademarked system, UNIX.

Introducing Authentication Services for Smart Cards

The Authentication Services for Smart Cards feature makes it possible for a user to insert a smart card in an Authentication Services-enabled workstation and authenticate to Active Directory. Authentication Services for Smart Cards functionality extends strong, two-factor authentication to both Windows and Unix using a single user repository.

Authentication Services for Smart Cards features and benefits

Deploying Authentication Services for Smart Cards provides the following features and benefits:

- Strong two-factor authentication for Unix.
- Smart Card login integrated with Active Directory.
- Integration with existing Unix applications.

Strong two-factor authentication for Unix

Authentication Services for Smart Cards strengthens Unix log on security by authenticating users using two-factor authentication. Two-factor authentication requires a user to combine something they have (a smart card or a security token) with something they know (a PIN or password). The security of a system is enhanced by using two-factor authentication, when compared to a traditional password system.

Smart Card login integrated with Active Directory

When Authentication Services for Smart Cards is used, all users are authenticated against Active Directory allowing user management from a single identity store. With Authentication Services for Smart Cards, you can use the same token to authenticate securely to both Windows and Unix workstations.

Integration with existing Unix applications

One Identity designed Authentication Services for Smart Cards to integrate with existing applications such as `Gnome`, `KDE`, `XDM`, and `su`.

Supported platforms

This release of Authentication Services for Smart Cards supports:

- Red Hat Enterprise AS, ES, and WS 5.0, 6.0, and 7.0 on x86 and x86-64 (AMD64, EMT64T)
- Apple macOS 10.12, 10.13. and 10.14 on x86_64

Supported cards and readers

One Identity designed Authentication Services for Smart Cards to support the PKCS #11 standard software interface and has tested it against OpenSC PKCS#11 library. This release of Authentication Services for Smart Cards supports all cards and readers that are supported by the RSA Security Inc. PKCS#11 Cryptographic Token Interface (Cryptoki).

Authentication Services for Smart Cards components

Authentication Services for Smart Cards has the following components:

- The Authentication Services for Smart Cards plugin.
- The `pam_vas_smartcard` PAM module.

- The `vastool smartcard` command line utility.
- Vendor PKCS#11 drivers.

Authentication Services for Smart Cards plugin

The Authentication Services for Smart Cards plugin is installed by the installer and provides the core smart card functionality.

The Authentication Services PAM module

PAM concepts

Pluggable Authentication Module (PAM) is an API that allows the system administrator to configure authentication mechanisms rather than hardcoding authentication mechanisms into the application. Administrators can customize an application's authentication system by making changes to `/etc/pam.conf` or an application-specific file in the `/etc/pam.d/` directory.

Authentication Services PAM modules are shared libraries that add support for a specific authentication mechanism. Unix platforms that support PAM normally have a PAM module called `pam_unix` for standard Unix authentication.

pam_vas_smartcard features

`pam_vas_smartcard` is an Authentication Services PAM module that supports login with a smart card. It provides many of the same features as the standard `pam_vas` module, including the ability to create home directories, perform UID conflict checking, and machine-based access control.

For information on configuring the `pam_vas_smartcard` module see the *pam_vas_smartcard man page*.

The vastool smartcard command line utility

`vastool` is a script-friendly command line utility that exposes a wide range of functionality to the Unix/Linux system administrator. Authentication Services for Smart Cards adds an additional `smartcard` command to allow configuration and troubleshooting of smart card-related issues. The following table lists some of the commands and functionality which you can access by running `vastool smartcard` command. For a complete list, see the *vastool man page*.

Table 1: vastool smartcard commands

Command	Function
configure	Configure smart card related settings such as the PKCS#11 driver and PAM.
info	Display information about smart cards and drivers.
test	Test smart card functionality.
trusted-certs	Manage the store of trusted certificates.
unconfigure	Remove smart card related settings.

Vendor PKCS#11 drivers

PKCS#11 is a standard software interface for accessing cryptographic functions on smart cards. Authentication Services for Smart Cards uses the vendor-provided PKCS#11 drivers to interface with the card.

Authentication Services integrates with third-party drivers, such as OpenSC. Once the drivers are installed, Authentication Services references these drivers from the installed shared library; therefore, you need to know the name and location of this library when you configure Authentication Services for Smart Cards.

NOTE: Authentication Services for Smart Cards is derived from the RSA Security Inc. PKCS#11 Cryptographic Token Interface (Cryptoki).

3

Installing Authentication Services for Smart Cards

Before you install the smart card drivers and the Authentication Services software, you must first install the Authentication Services agent and join your Unix host to the Active Directory domain.

Refer to the *Authentication Services Installation Guide* for step-by-step instructions. See the Authentication Services - Technical Documentation page on the One Identity support site for this guide.

Installing vendor smart card drivers

When using Authentication Services for Smart Cards, you must install and configure vendor drivers for your cards and readers. For example, you must have a working PKCS#11 library. This is a shared library that implements the PKCS#11 Cryptographic Token Interface Standard. Consult your smart card vendor documentation for more details.

Logging in to Active Directory with your card

Authentication Services for Smart Cards requires that you:

- Enable smart card log on support for Active Directory.
- Initialize your card using vendor supplied software.
- Use your card to enroll for a smart card certificate with your Certificate Authority.

Ensure that you can use this card to log on to a Windows workstation before attempting to use it to log in with Authentication Services for Smart Cards.

Installing Authentication Services for Smart Cards software

Authentication Services for Smart Cards is bundled as a separate installation package on the Authentication Services Installation media.

To install Authentication Services for Smart Cards on a supported platform, run the Authentication Services installation script, as follows.

```
# ./install.sh vasclnt vassc
```

NOTE: If Authentication Services is already installed, you can omit the "vasclnt" argument.

Configuring Authentication Services for Smart Cards

You must configure Authentication Services for Smart Cards to work with your vendor's PKCS#11 library drivers.

Configuring the vendor’s PKCS#11 library

Authentication Services for Smart Cards interfaces with the smart card and the smart card reader using the vendor’s PKCS#11 driver. This is a shared library implementing a standard interface supported by most card vendors for accessing the cryptographic functions of smart cards and tokens.

NOTE: Authentication Services for Smart Cards is derived from the RSA Security Inc. PKCS#11 Cryptographic Token Interface (Cryptoki).

Authentication Services for Smart Cards requires that you configure Authentication Services with the location of your vendor's PKCS#11 driver. If the driver is not configured you will be unable to use some smart card functions and it displays an error similar to this:

```
vastool smartcard info card
ERROR: no PKCS#11 library specified in vas.conf
```

To configure Authentication Services you need to know the location of your vendor's PKCS#11 shared library on the file system. Consult your vendor documentation for this information.

NOTE: You can specify the location of the PKCS#11 using either the full path to the PKCS#11 shared library or a path relative to the appropriate pkcs11 library subdirectory under `/opt/quest` for your architecture. For example, `/opt/quest/lib/pkcs11` on x86 Linux systems. See Configuring the PKCS#11 library for 32-bit and 64-bit versions on page 19.

For Example:

The Gemalto 5.1 Drivers for Red HatLinux on x86 platforms are installed in `/usr/local/lib/libxltCk.so`.

Testing the PKCS#11 library for Authentication Services for Smart Cards compatibility (optional)

The `vastool` utility provides an option to test whether a PKCS#11 library is suitable for use with Authentication Services for Smart Cards.

To test the PKCS #11 library

1. Run the following command:

   ```
   vastool smartcard -l <library> test library
   ```

   where `<library>` is the path to the PKCS#11 library you want to test.

   For example, to test the Gemalto PKCS#11 drivers on a Red Hat x86 platform, run the following command:

   ```
   vastool smartcard -l \
   /usr/local/lib/libxltCk.so test library
   ```

   This displays the following output if the driver is correctly installed:

   ```
   Testing PKCS#11 library '/usr/local/lib/libxltCk.so':
   Checking PKCS#11 library may be dynamically loaded ... ok
   Checking PKCS#11 library contains necessary symbols ... ok
   Checking PKCS#11 function list can be obtained ... ok
   Checking PKCS#11 library version is compatible ... ok
   Checking PKCS#11 library can be initialized ... ok
   Checking PKCS#11 library can be finalized ... ok
   ```

Configuring the vendor's PKCS#11 library using VASTOOL

To configure the location of the PKCS#11 library using vastool

1. Log in and open a root shell.
2. Run the following command.

   ```
   vastool smartcard configure pkcs11 lib <library>
   ```

   where `<library>` is the path to the PKCS#11 library.

   For example:

   - To configure the CoolKey PKCS#11 library, run the following command:

     ```
     vastool smartcard configure pkcs11 lib /usr/lib/pkcs11/libcoolkeypk11.so
     ```

   - To configure the Gemalto 64-bit PKCS#11 library, run the following command:

     ```
     vastool smartcard configure pkcs11 lib /usr/local/lib64/libxltCk.so
     ```

   - To configure the ActivClient PKCS#11 library, run the following command:

     ```
     vastool smartcard configure pkcs11 lib
     /usr/local/ActivIdentity/ActivClient/lib/libacpkcs211.so
     ```

NOTE: You can configure the PKCS#11 Library using this procedure or by editing the `vas.conf` file.

Configuring the vendor's PKCS#11 library by editing the configuration file

You can manually configure the location of the vendor's PKCS#11 library by editing the setting in the `/etc/opt/quest/vas.conf` file.

To configure the PKCS#11 library by editing the vas.conf file

1. Log in and open a root shell.
2. Open the file `/etc/opt/quest/vas/vas.conf` in the editor of your choice.
3. Add the section:

   ```
   [pkcs11]
   pkcs11-lib = <library>
   ```

where <library> is the path to the vendor's PKCS#11 library.

Configuring the PKCS#11 library for 32-bit and 64-bit versions

When you install Authentication Services for Smart Cards on a 64-bit platform, you install both 64-bit and 32-bit versions of the libraries and Authentication Services PAM modules. If you want to use both architectures (for example to allow smart card login using a 32-bit application), you need both 32-bit and 64-bit PKCS#11 libraries.

To install both these libraries, follow the appropriate steps for your platform.

Configuring the card slot for your PKCS#11 library

If you have multiple readers, or your card reader supports multiple slots, your vendor's PKCS#11 library may require you to specify the card slot with which you will be using to log in. If you do not specify a slot, Authentication Services for Smart Cards will probe for the first available slot. Typically, you will not need to configure this option. For more details on which slot number to configure consult your vendor's PKCS#11 documentation.

If the slot is not specified correctly then some smart card functions may return an error, for example:

```
vastool smartcard info card
ERROR: smart card is not present in slot
```

Configuring the card slot using VASTOOL

To configure the location of the PKCS#11 library using vastool

1. Log in and open a root shell.
2. Run the command:

   ```
   vastool smartcard configure pkcs11 slot \
   <slot-id>
   ```

   where <slot-id> is the card slot.

NOTE: You can remove the PKCS#11 slot from the configuration by running the `vastool smartcard unconfigure pkcs11 slot` command.

Configuring the vendor's PKCS#11 slot by editing the configuration file

You can manually configure the location of the vendor's PKCS#11 card slot by editing the setting in the `/etc/opt/quest/vas.conf` file.

To configure the location of the PKCS#11 card slot in vas.conf

1. Log in and open a root shell.
2. Open the `/etc/opt/quest/vas/vas.conf` file in the editor of your choice.
3. Locate the `[pkcs11]` section (or add one if not present), and add the following:

   ```
   pkcs11-slot = <slot-id>
   ```

   where `<slot-id>` is the number of the slot you want to use to log in.

NOTE: Remember that specifying a slot id is optional. Authentication Services for Smart Cards will probe for an available slot if a slot id is not specified.

Configuring PAM applications for smart card login

To integrate Authentication Services for Smart Cards with existing applications you need to configure PAM. This section describes in detail how to configure the `pam_vas_smartcard` module for different scenarios, and gives recommendations for which options works well with some common login applications. The following topics are discussed:

- Security issues when configuring smart card login
- Usability issues when configuring smart card login
- Configuring PAM for smart card only login
- Configuring PAM for smart card and password login
- Configuring GDM
- Configuring KDM
- Configuring XDM
- Configuring Console Login
- Configuring Dtlogin

You can find background information on PAM and configuring Authentication Services PAM modules in the *Authentication Services Administration Guide*, which can be found on the Authentication Services - Technical Documentation page on the One Identity support site.

Security issues when configuring smart card login

One of the properties that makes smart card login more secure is that it requires the physical presence of a card or token to authenticate. To secure smart card login, you must limit card access to users who are physically present at the terminal and ensure that remote users cannot access cards.

You enable smart card login by configuring the Authentication Services PAM module `pam_vas_smartcard` for a given application. When the application requires authentication, it makes calls to this module which in turn communicates with the smart card and prompts the user for his PIN.

Because you can use PAM to authenticate both remote and local users, never configure smart card login for remote login applications such as `SSH`, `telnet`, or `ftp`. The `pam_vas_smartcard` module is unable to determine whether a login is from a local or remote user.

Therefore, if you enable `pam_vas_smartcard` on a remote login service, an attacker may be able to connect to these services and either attempt to guess the PIN of the locally inserted card, or cause denial of service by locking out the card after several attempts.

A further complication is that you can use some applications for both local and remote login (for example XDM or `/bin/login`).

For this reason it is not possible to enable the `pam_vas_smartcard` module for all applications, as you can with the normal Authentication Services PAM module. You must decide which services to enable using the `vastool smartcard configure pam` command and enable these one by one.

For more information on how to secure login to these applications for local users only, see the appropriate sections below.

NOTE: Using Authentication Services for Smart Cards, you cannot log in to a remote service using the local smart card with OpenSSH;, however, you can use Kerberos to log in to a remote service using Generic Security Services Application Program Interface (GSSAPI).

Usability issues with PAM applications

While PAM provides a mechanism for integrating custom authentication mechanisms, many applications are designed only to support username-based logins and password-based logins.

In general, most applications will work with Authentication Services for Smart Cards in the following way:

1. The application displays either a "Username: " or "Insert Card or enter username" prompt.
2. The user enters the username that is on their card. This may be their Unix login name, or their full UPN.

3. The application displays either a "PIN" or "Password" prompt.
4. The user enters his PIN.

Depending on how the `pam_vas_smartcard` module is configured, it is possible to either login using the smart card or a local user (such as root). It is also possible to configure PAM so that a user can log in with either a smart card or with a password.

Enabling smart card login for selected services

Once you have installed and configured Authentication Services correctly, you must enable smart card login. Authentication Services for Smart Cards provides a PAM module `pam_vas_smartcard.so` that allows integration of Authentication Services for Smart Cards with PAM-aware applications. For more information on options that you can use with the Authentication Services for Smart Cards module, see the *pam_vas_smartcard (8) man page*.

NOTE: Unlike Authentication Services password login, smart card login is not enabled for all PAM services by default. Because some services such as `SSH` and `telnet` use PAM to authenticate users over a network, enabling smart card login for these services is undesirable. Enabling would allow an attacker to attempt to brute force the card PIN or exceed the maximum login attempts for the card causing the card to be locked. For this reason only enable PAM for services which are used for local login (such as, `GDM`, `KDM`, and dtlogin). For more information, see Security issues when configuring smart card login on page 21.

Enabling smart card login

To enable smart card login

1. Log in and open a root shell.
2. Run the command:

   ```
   vastool smartcard configure pam <service>
   ```

   where `<service>` is the name of the service (such as, `gdm` or `kdm`) for which you want to enable smart card login.
3. Depending on the service you may need to restart to log in with a smart card.

Disabling smart card login

To disable smart card Login

1. Log in and open a root shell.
2. Run the command:

   ```
   vastool smartcard unconfigure pam <service>
   ```

   where `<service>` is the name of the service (such as, `gdm` or `kdm`) for which you want to enable smart card login.

Configuring applications for smart card and password login

When you install Authentication Services, most applications are configured to allow login to Active Directory with a password, or to a local user account.

To enable users to also log in with a smart card for a given service

1. Log in and open a root shell.
2. Run the command:

   ```
   vastool smartcard configure pam <service>
   ```

   where `<service>` is the name of the service to enable for smart card login.

This configures either the `/etc/pam.conf` file or `/etc/pam.d/<service>` file depending on your operating system and existing PAM configuration.

Example: Application configured for Redhat Enterprise Linux 5.0 login

After running the `vastool smartcard configure pam gdm` command, the GDM pam configuration on a Redhat Enterprise Linux 4.0 looks like this:

```
/etc/pam.d/gdm
#%PAM-1.0
auth required pam_env.so
auth [ignore=ignore success=done default=die] pam_vas_smartcard.so
create_homedir
auth required pam_stack.so service=system-auth
```

```
auth required pam_nologin.so
account [ignore=ignore success=done default=die] pam_vas_smartcard.so
account required pam_stack.so service=system-auth
password [ignore=ignore success=done default=die] pam_vas_smartcard.so
password required pam_stack.so service=system-auth
session required pam_vas_smartcard.so create_homedir
session required pam_stack.so service=system-auth
session optional pam_console.so
```

Note that when you joined the domain, it configures the `pam_stack.so` module for Authentication Services password login. You can see the configuration in the `/etc/pam.d/system-auth` file:

```
/etc/pam.d/system-auth
#%PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth required /lib/security/$ISA/pam_env.so
auth [ignore=ignore success=done default=die] pam_vas3.so create_homedir
auth sufficient /lib/security/$ISA/pam_unix.so likeauth nullok
auth required /lib/security/$ISA/pam_deny.so
account [ignore=ignore success=done default=die] pam_vas3.so
account required /lib/security/$ISA/pam_unix.so
account sufficient /lib/security/$ISA/pam_succeed_if.so uid < 100 quiet
account required /lib/security/$ISA/pam_permit.so
password [ignore=ignore success=done default=die] pam_vas3.so
password requisite /lib/security/$ISA/pam_cracklib.so retry=3
password sufficient /lib/security/$ISA/pam_unix.so nullok
use_authtok md5 shadow
password required /lib/security/$ISA/pam_deny.so
session required /lib/security/$ISA/pam_limits.so
session required pam_vas3.so create_homedir
session required /lib/security/$ISA/pam_unix.so
```

Configuring applications for smart card login

To configure an application to only allow smart card login you must first disable password-based login for that application. There are two ways to do this. You can either remove the `pam_vas3`-specific entries from the PAM configuration file or you can run the `vastool unconfigure pam` command.

The `vastool unconfigure pam` command disables Authentication Services password login for all applications because it removes all existing Authentication Services password (`pam_`

vas3) and Authentication Services for Smart Cards (pam_vas_smartcard) PAM modules from the configuration.

After you run the vastool unconfigure pam command, you can selectively enable Authentication Services password login for a service by running the vastool configure pam <service> command, as follows:

```
vastool smartcard configure pam gdm
vastool smartcard configure pam kde
vastool smartcard configure pam xdm
vastool smartcard configure pam login
vastool smartcard configure pam dtlogin etc.
```

NOTES:

This still allows you to log in as a local user account. To disable log in as a local user account, you must manually remove the pam_unix module.

You can enable the smartcard-only option for the pam_vas_smartcard module to display an error message if a Authentication Services user attempts to log in without a card present. See *Customizing PAM login prompts* in the *pam_vas_smartcard man page* for more information.

pam_vas_smartcard options

The pam_vas_smartcard module provides a number of options for configuring the behavior of the Authentication Services for Smart Cards. You can also use many of these options in the normal pam_vas3 module, as well. See the *pam_vas_smartcard man page* for more information about the available pam_vas_smartcard options.

Table 2: Smart Card-specific pam_vas_smartcard options

Option	Function
show-token-status	Display verbose information about smart card status when logging in.
smartcard-only	Enforce smart card logins for Authentication Services users. This displays an error if a Authentication Services user attempts to log in without a card inserted.
ignore-non-vas-user	Do not display an error message if a card is inserted which does not have a Unix-enabled user.
pin-required	Always prompt for a PIN, otherwise query the PKCS#11 driver to determine whether one is required first.
prompt-style	Display prompt information in a manner that may be more suitable for graphical PAM application.

Note that the `prompt-style` and `show-token-status` options are intended to modify the appearance of information presented by the PAM application, and may not display correctly with all PAM applications. One Identity recommends that you experiment with the `prompt-style` and `show-token-status` options to determine if these options are useful for a particular PAM application.

Configure Gnome Display Manager (GDM)

The Gnome Display Manager (GDM) is a PAM application providing graphical login. The following sections document how to configure and use GDM with smart card authentication.

Configuring GDM for smart card

To configure GDM for smart card

1. Run the following command:

```
vastool smartcard configure pam gdm
```

Typically, GDM initially displays an **Insert card:** prompt if you have specified the `smart card-only` option; otherwise it displays a **Insert card or enter username:** prompt. Once you have entered the username, it displays a **PIN:** prompt.

Note that you can select themes for GDM. The theme that you select may not display prompts or additional information from the `pam_vas_smartcard` module, or it may display both prompts and additional information.

You can modify how the prompts display in one of these ways:

- Specify new prompts using the `prompt-vassc-user` and `prompt-vassc-pin` options in the `[pam_vas]` section of `vas.conf`;
- Specify the `prompt-style=text` option of the `pam_vas_smartcard` module.

  This displays the prompts in that section of the GDM theme that display PAM messages, while not displaying anything in that section of the theme that display PAM prompts;
- Specify the `prompt-style=both` option of the `pam_vas_smartcard` module.

  This displays the prompts in that section of the GDM theme that display PAM messages, while displaying *Username:* and *PIN:* in that section of the theme that display PAM prompts.

Only choose the `prompt-style` and `show-token-status` options if the theme supports the display of PAM messages. It may be necessary to choose the `prompt-style=text` or `prompt-style=both` options if the theme does not support the display of PAM prompts, or if the prompts appear to be truncated.

If the theme displays PAM messages and you want token status messages to display, specify the `show-token-status=clear` option.

Note that some themes or versions of GDM may not display PAM messages correctly, or may fail to erase previous PAM messages. You must carefully consider the use of the `prompt-style` and `show-token-status=clear` options and only choose them if the overall display is suitable. One Identity recommends that you use a simple theme that displays PAM prompts without truncation.

Disable remote login

One Identity recommends that you disable remote login for GDM by disabling the X display manager control protocol (XDMCP). XDMCP is disabled by default; however, you can manually disable XDMCP.

Editing the GDM configuration file manually

To edit the GDM Configuration file manually

1. Open the GDM configuration file.

   This file is typically located at `/etc/X11/gdm/gdm.conf`, however "local" settings may take precedence. You can find the local settings in `/etc/X11/gdm/factory-gdm.conf` file.

2. Look for the `[XDMCP]` section and verify that the `Enable` property is either not present, commented out, or is set to `false`, as follows:

   ```
   [XDMCP]
   Enable=false.
   ```

   NOTE: Whether modifying the GDM configuration manually or by using the graphical user interface, you must restart GDM.

Editing the GDM configuration file with the graphical application

GDM includes a graphical application that you can use to configure GDM. The following steps document how to disable remote login with this application:

To disable remote login

1. Run `/usr/bin/gdmsetup`.
2. Click the **XDMCP** tab.
3. Verify that the **Enabled XDMCP** is not selected.

NOTE: Whether modifying the GDM configuration manually or by using `/usr/bin/gdmsetup`, you must restart GDM.

Using GDM with a smart card

To perform smart card login by means of Gnome Display Manager (GDM)

1. Insert your smart card.
2. Enter your username or UPN at the **Username:** prompt, if required.

   NOTE: GDM permits a null entry. An unspecified username allows the `pam_vas_smartcard` module to obtain the username from the smart card itself.
3. Enter your PIN at the **Password:** prompt.
4. Click the **Login** button.

Configure K Display Manager (KDM)

The K Display Manager (KDM) is a PAM application providing graphical login. The following sections document how to configure and use KDM with smart card authentication.

Configuring KDM for smart card

To configure KDM for smart card

1. Run the following command:

   ```
   vastool smartcard configure pam kde
   ```

Unlike GDM, KDM presents both a **Username:** and a **Password:** prompt simultaneously to the user. You can not change these prompts. The `prompt-vassc-user` and `prompt-vassc-pin` options in the `[pam_vas]` section of `vas.conf` have no effect.

Note that KDM displays additional information from the Authentication Services PAM module in a pop-up window, which only disappears when the user clicks **OK**. Thus, the `prompt-style` and `show-token-status` options are not recommended for KDM.

Disabling remote login

To disable remote login

1. Open the KDM configuration file for editing.

   Typically this file is located at `/etc/X11/xdm/xdm-config` or `/usr/share/config/kdm/kdmrc` on Redhat.
2. Look for the `[XDMCP]` section and verify that the `Enable` property is either not present, commented out, or is set to `false`, like this:

   `[XDMCP] Enable=false.`

Using KDM with a smart card

To perform smart card login by means of K Display Manager (KDM)

1. Insert your smart card.
2. Enter your username or UPN at the **Username:** prompt.
3. Enter your PIN at the **Password:** prompt.
4. Click the **Login** button.

KDM calls the `pam_vas_smartcard` module to perform the authentication.

NOTE: KDM displays the **Username:** and **Password:** prompts regardless of the presence or absence of the smart card in the reader. In addition, KDM does not allow you to enter an empty username. While GDM permits an unspecified username, KDM requires one. Failure to provide a username results in a "login failed" message.

Configure X Display Manager (XDM)

The X Display Manager (XDM) is a PAM application providing graphical login. The following sections document how to configure XDM with smart card authentication.

Configure XDM for smart card

To configure XDM for smart card

1. Run the following command:

```
vastool smartcard configure pam xdm
```

XDM is similar to KDM. It displays a **Login:** and a **Password:** prompt, neither of which you can modify. Thus the `prompt-vassc-user` and `prompt-vassc-pin` options in the `[pam_vas]` section of `vas.conf` have no effect.

XDM does not display any additional information from the Authentication Services PAM module. Thus, the `prompt-style` and `show-token-status` options also have no effect under XDM.

Disabling remote login

One Identity recommends that you disable remote login for XDM by disabling the X display manager control protocol (XDMCP).

NOTE: XDMCP is disabled by default.

To manually disable XDMCP

1. Open the XDM configuration file for editing.

   This file is typically located at `/etc/X11/xdm/xdm-config`.
2. Verify that the `DisplayManager.requestPort` property is set to 0, like this:

   ```
   DisplayManager.requestPort: 0
   ```

Configure console login

The `/usr/bin/login` program is a PAM application for performing login to the system. Typically `/usr/bin/login` is called by the getty program for login to the console. The following sections document how to configure and use console login with smart card authentication.

Configuring console login for smart card

To configure console login for smart card

1. Run the following command:

   ```
   vastool smartcard configure pam login
   ```

NOTE: The login program always displays a **login:** prompt, which you cannot modify. Similarly, the getty program always displays a **login:** prompt, and passes the value it receives to the login program. Thus, the `prompt-vassc-user` option in the `[pam_vas]` section of `vas.conf` has no effect for the login program. However, the **PIN:** prompt may be changed by specifying a value for the `prompt-vassc-user` option in the `[pam_vas]` section of `vas.conf`.

A typical smart card-enabled console login looks similar to the following:

```
penguin.vintela.com login: matlock
PIN: ********
```

The login program can display additional information on standard output. Specify the `prompt-style` option of the `pam_vas_smartcard` module for additional prompting. However, it only displays additional prompting information for PIN prompts, as in the following example:

```
penguin.vintela.com login: matlock
Enter PIN for matlock@vintela.com
PIN: ********
```

Note that you can also specify the `show-token-status` option of the `pam_vas_smartcard` module if you want status information. For example:

```
Penguin.vintela.com login: matlock
Inspecting smart card …
PIN: ********
Authenticating …
```

Disabling remote login

Some remote login programs (such as, `ftp` or `telnet`) also use login the program. For this reason One Identity recommends that you disable remote login services if you have smart card login enabled for the console. Consult the administrator's guide for your operating system for further details on disabling `ftp` or `telnet`.

Using console login with a smart card

To perform smart card login by means of the console

1. Insert your smart card.

   The getty program prompts for a login.
2. Enter your username or UPN at the **Username:** prompt.

   You must enter the username or UPN that is on the smart card.
3. Enter your PIN at the **Password:** prompt.
4. Click the **Login** button.

Configuring certificates and CRLs

Because Authentication Services for Smart Cards uses Public Key cryptography, it must also obtain and manage Certificates and CRLs. This section includes background information on Public Key Infrastructure components, describes how these are used in Authentication Services for Smart Cards, and demonstrates how to manage certificates and CRLs for use when authenticating to Active Directory.

Concepts

Public Key Infrastructure

Public key cryptography makes communication on the Internet possible by providing means for a large number of people to exchange secure messages without having to first

agree on a shared key. However, to be able to securely encrypt a message, or verify the digital signature on a document, you must be sure that you are using the correct public key for the intended recipient. If you were to encrypt or verify a message with the wrong public key, then you may inadvertently send out information that can be decrypted by an attacker, or you may rely on the integrity of a document that has been modified in transit.

Both confidentiality and integrity rely on authenticating the sender and recipient to be sure that information goes to, or comes from the person intended. Despite the fact that public keys were created to solve the problem of key distribution, it seems to have replaced one key distribution problem with another.

One method of authenticating public keys is to have these keys (and information about the entity they belong to) digitally signed by a third party. This signed information is called a digital certificate (or commonly just a certificate) and the third party is known as a Certification Authority (CA).

While it is still necessary to obtain and authenticate the public key of the certification authority by some secure means, the problem of obtaining the public keys of all the people you wish to communicate with has been reduced to a problem of just obtaining a single key for a CA that has issued certificates for these people. CAs can also issue certificates to other CAs, so that it is possible to produce a chain of certificates (called a certification path) from a root CA (sometimes called a trust anchor) to the person or end entity with whom we wish to communicate.

The collection of algorithms, protocols, policies and procedures which enable us to build a scalable infrastructure for communicating using public key cryptography and certificates is known as a Public Key Infrastructure (PKI). PKIs are in everyday use around the world to provide security for banks, online merchants, traders and consumers alike.

Certificates and CRLs

A certificate authenticates public keys by binding the key with information associated with its user. A certificate also contains other information such as when the key is valid, what the key may be used for, and whether the key may be used to sign other certificates.

A Certificate Revocation List (CRL) is issued by a CA at regular intervals and lists certificates that have been invalidated or revoked before their expiry date.

When verifying whether a certificate is valid, a user must check both the certificate signature, and the current valid CRL to ensure that the certificate is not listed on it.

Reasons for revocation can include:

- Compromise of the private key
- Loss of the key or token
- Invalid or out of date information in the certificate

An example might be when a user leaves a company.

How Authentication Services for Smart Cards uses certificates and CRLs

Authentication Services for Smart Cards uses Public Key cryptography to authenticate users to Active Directory. It uses keys and certificates stored on the smart card to perform a version of the Kerberos authentication protocol called PKINIT. When Active Directory has authenticated the user, it in turn authenticates itself back to Authentication Services for Smart Cards.

This mutual authentication is critical to the security of the PKINIT protocol, and requires that Authentication Services for Smart Cards verify the certificate presented by Active Directory as part of this exchange against one or more trusted certificates and CRLs.

Trusted certificates used by Authentication Services are stored in the `/var/opt/quest/vas/certs` directory.

Map certificate to user (implicit and explicit)

Mapping certificates to users can be done implicitly or explicitly. Authentication Services supports mapping one cert to one user or mapping multiple certs to one user. Mapping one cert to multiple users is not supported.

Implicit mapping

The user's `subjectAlternativeName` (SAN) on the certificate typically matches the user's `userPrincipalName` (UPN) or Kerberos v5 principal name. The match is implicit and no additional mapping effort is needed. In the following example, the user `fred` has certificate where the SAN matches the user's UPN in Active Directory. The match is implicit and the user is authenticated.

AD Attribute:

```
$ /opt/quest/bin/vastool -u host/ attrs fred userPrincipalName
```

```
userPrincipalName: fred@example.com
```

Smart Card attribute:

```
UPN: fred@example.com
```

Explict mapping

If the user's SAN on the certificate does not match the user's UPN or Kerberos v5 principal name, you can explicitly map a certificate to a user using any one of several certificate attributes, including:

- Subject and Issuer fields
  `altSecurityIdentities: X509:<I>DC=local,DC=dod,CN=SpatDoD Root CA<S>CN=gman`
- Subject DN
  `altSecurityIdentities: X509:<S>CN=gman`
- Subject Key Identifier
  `altSecurityIdentities: X509:<SKI>ddde2ca4b86db8a908b95c6cbcc8bb1ac7a09a41`
- Issuer, and Serial Number
  `altSecurityIdentities: X509:<I>DC=local,DC=dod,CN=SpatDoD Root CA<SR>32000000000003bde810`
- SHA1 Hash
  `altSecurityIdentities: X509:<SHA1-PUKEY>ed913fa41377dbfb8eac2bc6fcae71ecd4a974fd`
- RFC822 name
  `altSecurityIdentities: X509:<RFC822>efedman@fedid.gov`

For details see: HowTo: Map a user to a certificate via all the methods available in the altSecurityIdentities attribute.

To explicitly map a certificate to a user

1. If necessary, disable UPN mapping behavior. For details, HowTo: Disable UPN mapping for SmartCard logon and How to disable the Subject Alternative Name for UPN mapping.
2. View the user's `altSecurityIdentites` attributes to see the certificates mapped to a user then map to one of the attributes.

You can define explicit mapping in Active Directory by importing a cert and mapping it to a user object. For details, see Defining the Mapping in Active Directory, Enable Explicit Mapping section.

Alternatively, you can look at the certificates mapped to a user, as shown in the following example.

Example: View the certificates mapped to a user

The user's `altSecurityIdentites` attributes shows the certificates mapped to a user.

For example, the following command line displayed the list of certificates mapped to the user `fred`. As we will see later, the `subject key identifier` (in bold below) can be used to map.

```
$ /opt/quest/bin/vastool -u host/ attrs fred altSecurityIdentities
```

```
altSecurityIdentities: X509:<SKI>983678656A4CF815843B4491BDD7B71784F2256E
```

```
altSecurityIdentities: X509:<SKI>45C700F77ACA71414AAA2D6C9FF9D93825B4F54A
```

```
altSecurityIdentities: X509:<SHA1-PUKEY>D628703850294D437A301ADFFD0D48FE2842F424
```

```
altSecurityIdentities: X509:<I>C=US,S=Example State,L=Example Location,O=Example
Organization,OU=Example OU,CN=User Bob,E=user.bob@example.com<SR>78563412
```

```
altSecurityIdentities: X509:<S>C=US,L=Example,O=Example Organization,OU=Example
OU,CN=subject-dn-mapping
```

```
altSecurityIdentities: X509:<SKI>05A7DAD104DA8FEDB8B9200F415D1719F483BF9F
```

```
altSecurityIdentities: X509:<RFC822>certmapping@smartcard.com
```

```
altSecurityIdentities:  X509:<I>C=US,S=Example  State,L=Example  Location,O=Example
Organization,OU=Example  OU,CN=User  Bob,E=user.bob@example.com<S>CN=subject-issuer-
mapping
```

As shown in the example below, the smart card attributes do not correspond to a user in Active Directory (AD). However, the subject key identifier (bolded below) can be used to map to the `altSecurityIdentites` (bolded in the above example).

Smart card attributes:

```
subjectAlternativeName: skimapping@smartcard.com
```

```
Subject = CN=subject-key-identifier-mapping
```

```
Issuer = emailAddress=bob@example.com,CN=User Bob,OU=Example OU,O=Example
Organization,L=Example Location,S=Example State,C=US
```

```
Serial Number = 16
```

```
Not valid before = 2019-12-02 23:54:26
```

```
Not valid after = 2020-12-01 23:54:26
```

```
Signature algorithm = md5WithRSAEncryption
```

```
Key algorithm = rsaEncryption
```

```
Key usage = keyEncipherment, digitalSignature
```

```
Extended key usage = Microsoft Smartcardlogin, 1.3.6.1.5.2.3.4, TLS Web Client
Authentication
```

```
Subject key identifier = 05A7DAD104DA8FEDB8B9200F415D1719F483BF9F
```

```
SHA1 hash = b009a0019b0a31125dcd0e80238ef7a263578a36
```

At this point, once the smart card is inserted into a reader, Authentication Services will:

1. Find the subject key identifier on the certificate.
2. Locate the user in Active Directory that the subject key identifier is mapped to.
3. Authenticate fred@example.com.

Bootstrapping trusted certificates

By default Authentication Services for Smart Cards is configured to automatically retrieve trusted certificates and CRLs from Active Directory. It is possible to do this securely because Authentication Services sets up a secure communication channel at join time using the symmetric host key that it uses to join itself to the domain.

Active Directory stores trusted certificates for smart card login in the `NtAuthCertificates` container which is located by the LDAP distinguished name.

```
CN=NtAuthCertificates,CN=Public Key Services,CN=Configuration,
DC=<domain>,DC=<domain>,…
```

By default, any certificates placed in this location in Active Directory are automatically distributed to both Windows and Authentication Services for Smart Cards clients.

Authentication Services for Smart Cards places these trusted certificates in the `NtAuth` subdirectory of the `/var/opt/quest/vas/certs` directory.

NOTE: You should not place any additional certificates in this subdirectory as they may be deleted from time to time. You may however place additional trusted certificates directly in the `/var/opt/quest/vas/certs` directory.

Automatic CRL retrieval

By default Authentication Services for Smart Cards retrieves any CRLs that are required to verify the certificates presented by Active Directory and automatically updates these as they expire and new certificates are issued. To be able to retrieve CRLs, the certificates to which they correspond must contain a CRL distribution points extension that contains an LDAP URI from which to download the CRL.

CRLs are stored in the `/var/opt/quest/vas/crls` directory.

Options for controlling certificate and CRL processing

Authentication Services provides a number of `vas.conf` options for configuring bootstrapping behavior.

Table 3: Options for configuring bootstrapping behavior

Option	Function
`auto-crl-download`	Whether to automatically download CRLs as needed.
`auto-crl-removal`	Whether to remove out-of-date CRLs from the cache automatically.
`bootstrap-trusted-certificate`	Whether trusted certificates should be automatically retrieved from Active Directory.
`trusted-certs-update-interval`	How often trusted certs and CRL should be updated (default 8 hours).
`auto-crl-download-bind-type`	How to bind to the LDAP directory when retrieving CRLs.

Managing Certificates and CRLs

Update certificates manually

By default certificates and CRLs are updated if the `trusted-certs-update-interval` has expired, and then only during the login process. You can request an update of the trusted certificates directory manually by using the `vastool smartcard trusted-certs` command, as follows:

```
vastool smartcard trusted-certs update
```

NOTE: You can schedule an update during off hours using a `cron` job.

Force an update of certificates

You can manually update the trusted certificates outside the configured period. For example, to retrieve a recently added trusted certificate, use the `-f` option with the `vastool smartcard trusted-certs` command, as follows:

```
vastool smartcard trusted-certs update -f
```

This command removes the existing certificates from the `NtAuth` subdirectory and retrieves all the current trusted certificates from Active Directory.

Disable bootstrap and manage certificates and CRLs manually

You can disable certificate bootstrapping and CRL downloading and distribute these items to Authentication Services clients by other means, such as Group Policy.

To disable bootstrap and manage certificates and CRLs manually

1. Set the `auto-crl-download`, `auto-crl-removal` and `bootstrap-trusted-certs` options to false in the `[pkinit]` section of the `/etc/opt/quest/vas/vas.conf` files, as follows:

   ```
   [pkinit]
   auto-crl-download = false
   auto-crl-removal = false
   bootstrap-trusted-certs = false
   ```

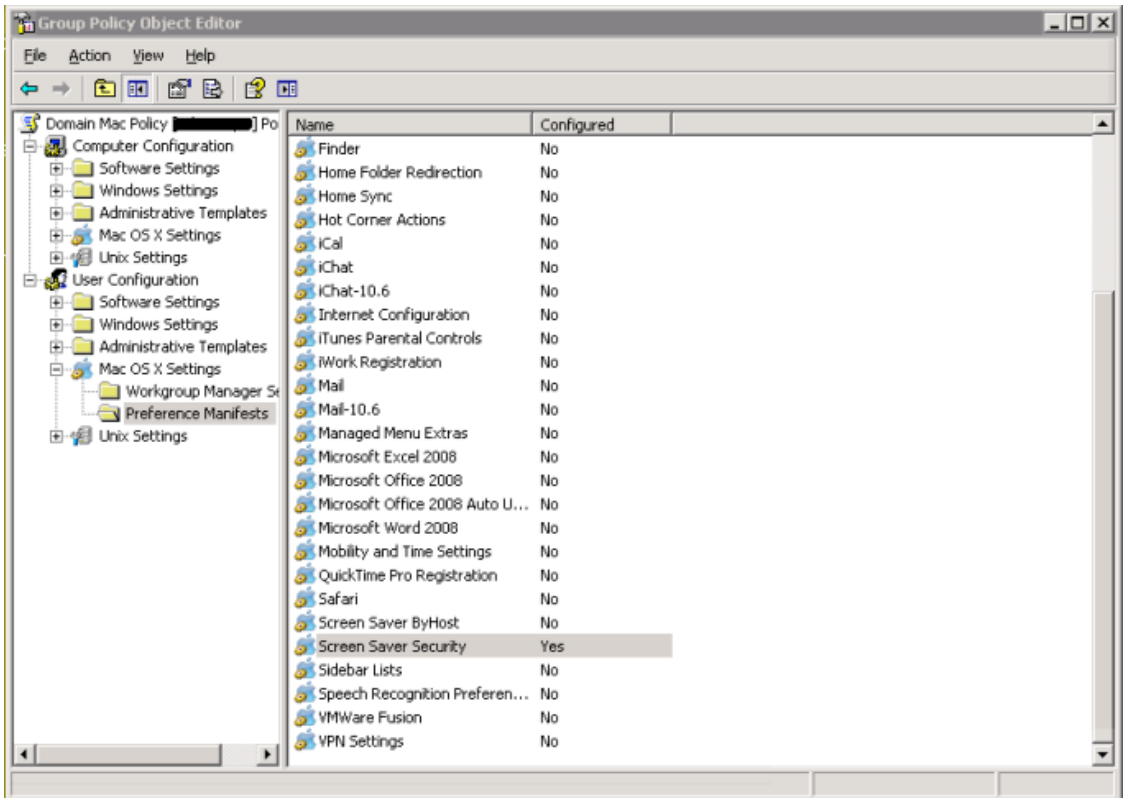
2. Place the trusted certificates in the `/var/opt/quest/vas/certs` directory.
3. Place CRLs in the `/var/opt/quest/vas/crls` directory.

Locking the screen saver upon card removal (macOS)

The ability to lock the screen saver when a token is removed is a feature and function of the macOS screen saver.

To enforce this setting using Mac OS X Group Policy

1. Navigate to **User Configuration | Mac OS X Settings | Preference Manifests | Screen Saver Security** from the Windows administrative machine that has the Authentication Services components installed.



2. Set the following settings to ensure that the screen saver becomes locked and stays locked once the smart card is removed:
   - **Screensaver Require Password Delay**: 0
   - **Require Password For Screensaver Unlock**: 1

Testing Authentication Services for Smart Cards

After you install and configure Authentication Services for Smart Cards to work with your vendor's PKCS#11 library drivers, you will want to validate your installation.

Testing general configuration and login using smart card

This procedure tests the Authentication Services for Smart Cards installation. It ensures that the library is installed correctly, the card has been initialized, there is a valid user certificate installed, and the card can be used to log into Active Directory.

To test the Authentication Services for Smart Cards installation

1. Attach a supported reader.
2. Insert the initialized card.
3. Run the following command.

   ```
   vastool smartcard test all
   ```

   If the card is configured correctly, it displays output similar to the following:

   ```
   Config:
   -------
   Checking that a PKCS#11 library is specified ... ok
   (Specifying PKCS#11 slot is optional)
   Library:
   --------
   Testing PKCS#11 library '/usr/local/lib/libxltCk.so':
   Checking PKCS#11 library may be dynamically loaded ... ok
   Checking PKCS#11 library contains necessary symbols ... ok
   Checking PKCS#11 function list can be obtained ... ok
   ```

```
Checking PKCS#11 library version is compatible ... ok
Checking PKCS#11 library can be initialized ... ok
Checking PKCS#11 library can be finalized ... ok
Card:
-----
Getting mechanisms ... ok
Checking for required mechanisms ... ok
Testing that card contains a user ... ok
User:
-----
Testing user j.doe@example.com
Testing if PIN is required ... ok
Enter PIN for j.doe@example.com: ****
Performing login to card ... ok
Generating signature ... ok
Verifying signature ... ok
Login:
-----
Testing user j.doe@example.com
Testing if PIN is required ... ok
Enter PIN for j.doe@example.com:
Performing login to card ... ok
Creating ID for client with UPN 'j.doe@example.com' ... ok
Establish initial credentials using PKCS#11 ... ok
```

Testing the configuration

The `vastool smartcard test` command provides a number of tests to determine whether you have correctly set up your environment and initialized your cards. While this step is optional, One Identity strongly recommends that you test your configuration before you enable Authentication Services for Smart Cards for a specific login service.

Some of the available tests require that you insert a card.

NOTE: See the *vastool man page* for more details about the different options available for the `vastool smartcard test` subcommand.

Test the PKCS#11 library

To test that the PKCS#11 library is configured correctly

1. Run the `vastool smartcard test library` command.

   For example, to test the currently configured library, enter:

```
vastool smartcard test library
```

If it is configured correctly, it returns output similar to:

```
Testing PKCS#11 library '/usr/local/lib/libxltCk.so':
Checking PKCS#11 library may be dynamically loaded ... ok
Checking PKCS#11 library contains necessary symbols ... ok
Checking PKCS#11 function list can be obtained ... ok
Checking PKCS#11 library version is compatible ... ok
Checking PKCS#11 library can be initialized ... ok
Checking PKCS#11 library can be finalized ... ok
```

To test a library other than the currently configured one

1. Specify an argument to `vastool smartcard test library`.

   For example:

   ```
   # vastool smartcard test library \
   /usr/local/lib/libxltCk.so
   ```

   If the library could not be loaded, or does not export a PKCS#11 interface, then `vastool smartcard test library` displays an error message, similar to the following:

   ```
   # vastool smartcard test library
   /usr/local/lib/libpkcs11broken.so
   Testing PKCS#11 library '/usr/local/lib/libpkcs11broken.so':
   Checking PKCS#11 library may be dynamically loaded ... ok
   Checking PKCS#11 library contains necessary symbols ... failed
   ERROR: PKCS#11 library does not contain symbol 'C_GetFunctionList'
   ```

Test the smart card is initialized correctly

To test that a smart card has been correctly initialized

1. Insert the smart card into the reader.
2. Run `vastool smartcard test card`. For example:

   ```
   # vastool smartcard test card
   Getting mechanisms ... ok
   Checking for required mechanisms ... ok
   Testing that card contains a user ... ok
   ```

This test displays a warning if the card is not recognized, or has not been correctly initialized.

Test the smart card user

To test that a card has been initialized with an appropriate user

1. Run the `vastool smartcard test user` command, as follows:

   ```
   # vastool smartcard test user
   Testing user user@vas.example
   Testing certificate validity ... ok
   Testing if PIN is required ... ok
   Enter PIN for user@vas.example: xxxxxxxx
   Performing login to card ... ok
   Generating signature ... ok
   Verifying signature ... ok
   ```

   This tests whether a valid user is on the card, and whether you are able to log into the card and use its cryptographic functions. If your card requires a PIN, enter the password at the prompt.

   The `vastool smartcard test card` function generates output similar to the following:

   ```
   CKM_RSA_X_509 CKM_MD2_RSA_PKCS CKM_MD5_RSA_PKCS CKM_SHA1_RSA_PKCS
   CKM_DES_KEY_GEN CKM_DES_ECB CKM_DES_CBC CKM_DES_CBC_PAD CKM_DES2_KEY_GEN
   CKM_DES3_KEY_GEN CKM_DES3_ECB CKM_DES3_CBC CKM_DES3_CBC_PAD CKM_MD2 CKM_MD5
   CKM_SHA_1
   Checking that CKM_RSA_PKCS mechanism is supported ... ok
   Checking info for CKM_RSA_PKCS mechanism ... ok
   Checking CKM_RSA_PKCS mechanism supports signing ... ok
   Checking CKM_RSA_PKCS mechanism supports decryption ... ok
   Testing that card contains a user ... ok
   ```

Test user log in

NOTE: This command requires that you are joined to a domain.

To test whether it is possible to log in using the inserted card

1. Run the `vastool smartcard test login` command.

   For example:

   ```
   # vastool smartcard test login
   Testing user user@vas.example
   Testing certificate validity ... ok
   ```

```
Testing if PIN is required ... ok
Enter PIN for user@vas.example:
Performing login to card ... ok
Creating ID for client with UPN 'user@vas.example' ... ok
Establish initial credentials using PKCS#11 ... ok
```

This command uses the inserted card to perform a log in to Active Directory. It displays a warning if the user is not Unix enabled, and displays an error if the log in fails. This command is useful when troubleshooting Authentication Services for Smart Cards log in problems.

Appendix A

Troubleshooting

To help you troubleshoot your Authentication Services for Smart Cards installation, One Identity recommends the following resolutions to some of the common problems you might encounter.

Steps to diagnose problems

Authentication Services for Smart Cards provides a number of tools and options to diagnose problems.

1. Check the smart card reader
2. Check the PKCS#11 library
3. Check the card
4. Check login
5. Enable debugging for smart card login with PAM
6. Enable debugging for the Authentication Services daemon
7. Enable debugging for the PKCS#11 library

Check the smart card reader

To troubleshoot problems with the card reader, first ensure that the reader is connected to the Unix workstation correctly, and that it is detected by the system.

To ensure that the reader is connected correctly

1. Run the following command:

```
/sbin/lsusb
```

This displays output showing that the card reader is attached to one of the USB ports. For example:

```
Bus 003 Device 001: ID 0000:0000
Bus 002 Device 002: ID 04e6:511c SCM Microsystems, Inc.
Bus 002 Device 001: ID 0000:0000
Bus 001 Device 001: ID 0000:0000
```

This shows a Reflex v3 USB reader connected to the workstation.

NOTE: Some readers require that you insert a card before the USB driver detects it.

Consult your vendors troubleshooting guide for more details on determining whether the reader is connected.

Check the PKCS#11 library

Authentication Services for Smart Cards requires that you install a PKCS#11 driver to access cryptographic functions on the smart card.

To determine which PKCS#11 library is installed

1. Run the `vastool smartcard info library` command, as follows:

```
# vastool smartcard info library
Library: /usr/local/lib/libxltCk.so
PKCS#11 version : 2.1
PKCS#11 manufacturer : Gemalto
PKCS#11 library description: Gemalto PKCS #11 Module
PKCS#11 library version : 5.2
```

To determine whether the driver is working correctly

1. Run the `vastool smartcard test library` command.

   For example:

```
# vastool smartcard test library
Testing PKCS#11 library '/usr/local/lib/libxltCk.so':
Checking PKCS#11 library may be dynamically loaded ... ok
Checking PKCS#11 library contains necessary symbols ... ok
Checking PKCS#11 function list can be obtained ... ok
Checking PKCS#11 library version is compatible ... ok
Checking PKCS#11 library can be initialized ... ok
Checking PKCS#11 library can be finalized ... ok
```

Check the card

To obtain information about the smart card you are attempting to use for log in

1. Run the `vastool smartcard info card` command, as follows:

```
# vastool smartcard info card
label : MS interop NS card
manufacturerID: Gemalto
model : Access eg 32K v2
serial number : 0001162CFF021982
flags : { CKF_RNG CKF_LOGIN_REQUIRED CKF_USER_PIN_INITIALIZED
CKF_DUAL_CRYPTO_OPERATIONS}
Number of mechanisms on card: 18
CKM_RSA_PKCS_KEY_PAIR_GEN
CKM_RSA_PKCS
CKM_RSA_X_509
CKM_MD2_RSA_PKCS
CKM_MD5_RSA_PKCS
CKM_SHA1_RSA_PKCS
CKM_DES_KEY_GEN
CKM_DES_ECB
CKM_DES_CBC
CKM_DES_CBC_PAD
CKM_DES2_KEY_GEN
CKM_DES3_KEY_GEN
CKM_DES3_ECB
CKM_DES3_CBC
CKM_DES3_CBC_PAD
CKM_MD2
CKM_MD5
CKM_SHA_1
```

   This displays information about the type of card inserted and the supported cryptographic operations.

To determine whether a particular card can be used with Authentication Services for Smart Cards

1. Run the `vastool smartcard test card` command, as follows:

```
# vastool smartcard test card
```

```
Getting mechanisms ... ok
Checking for required mechanisms ... ok
Testing that card contains a user ... ok
```

Check login

To log in with a given smart card it must contain a certificate that contains the User Principal Name (UPN) of the user with which that the card can be used to log in.

To determine the user on a given card

1. Run the `vastool smartcard info user` command, as follows:

   ```
   # vastool smartcard info user
   UPN: sc-1-a@a.vas
   subject = /DC=vas/DC=a/CN=Users/CN=Smartcard 1. A
   issuer = /DC=vas/DC=a/CN=ca-root-a
   ```

   This displays information from the user certificate on the card.

   ```
   serialNumber = 5907991B000100000016
   notBefore = Oct 3 04:53:34 2006 GMT
   notAfter = Oct 3 04:53:34 2007 GMT
   signatureAlgorithm = sha1WithRSAEncryption
   keyAlgorithm = rsaEncryption
   ```

To determine whether this user is suitable for logging on to Active Directory

1. Run the `vastool smartcard test user` command, as follows:

   ```
   # vastool smartcard test user
   Testing user sc-1-a@a.vas
   Testing certificate validity ... ok
   Testing if PIN is required ... ok
   Enter PIN for sc-1-a@a.vas:
   Performing login to card ... ok
   Generating signature ... ok
   Verifying signature ... ok
   ```

   This retrieves the user information, tests whether the user on the card is user-enabled, and tests that the certificate can verify digital signatures generated by the card.

To simulate a full log on with Active Directory

1. Run the `vastool smartcard test login` command, as follows:

```
# vastool smartcard test login
Testing user sc-1-a@a.vas
Testing certificate validity ... ok
Testing if PIN is required ... ok
Enter PIN for sc-1-a@a.vas:
Performing login to card ... ok
Creating ID for client with UPN 'sc-1-a@a.vas' ... ok
Establish initial credentials using PKCS#11 ... ok
Enabling debug for vastool commands
```

To enable additional debugging information

1. Run `vastool` with the `-d` option, as follows:

   ```
   # vastool -d 4 smartcard test login
   ```

   You can set the debug level from 1-6 for increasing levels of verbosity. Level 4 is generally sufficient for most smart card debugging.

Enable debugging for smart card login with PAM

The `pam_vas_smartcard` module supports an additional debug option that enables syslog to capture debugging information. This option is the same as the debug option supported by the `pam_vas3` module. See *Enabling diagnostic logging* in the *Authentication Services Administration Guide* for more information on how to configure syslog for this option.

Enable debugging for the Authentication Services daemon

To enable additional debugging for the Authentication Services daemon

1. Run the `debug-level` option in `vas.conf`, as follows:

   ```
   [vasd]
   debug-level=4
   ```

See *Enabling diagnostic logging* in the *Authentication Services Administration Guide* for more information on debugging `vasd`.

Enable debugging for the PKCS#11 library

If a failure occurs when testing your cards, it is valuable to have as much debug information as possible. Some PKCS #11 libraries may provide a way to collect additional debugging information. For example, the following procedure explains how to enable debugging for the PKCS#11 library using OpenSC. For more information on OpenSC, see OpenSC Manual Pages: Section 5.

To enable debugging for the PKCS#11 library

1. Navigate to `/usr/etc/opensc.conf`.
2. Edit the `opencs.conf`, adding the following configuration options to the `opensc-pkcs11` application block:
    - `debug = <num>;`

      where `<num>` indicates the amount of debug information to be included. A greater value means more debugging information is included. Default: 0.

      The OPENSC_DEBUG environment variable overwrites this setting.
    - `debug_file = <filename>;`

      where `<filename>` is the name of the file to which the debug information will be written. Default: `stderr`.

      Special values, `stdout` and `stderr` are recognized.

Troubleshooting vastool errors

The following sections describe symptoms and possible causes that you might encounter when using the `vastool smartcard` commands.

For information on other `vastool` commands, see the *Authentication Services Administration Guide*, which can be found on the Authentication Services - Technical Documentation page on the One Identity support site.

vastool ERROR: no PKCS#11 library specified in vas.conf

You encounter this error when you have not configured a PKCS#11 library.

To configure the PKCS#11 library

1. Run the `vastool smartcard configure pkcs11 lib` command.

vastool ERROR: Could not get symbol 'C_ GetFunctionList'

You encounter this error when you have configured an invalid PKCS#11 library.

To determine that the PKCS#11 library is installed and configured

1. Run the `vastool smartcard info config` command.

To test whether the configured library is valid

1. Run the `vastool smartcard test library` command.

vastool ERROR: invalid ELF header

You encounter this error when the configured PKCS#11 library is not a valid shared object.

To determine that the PKCS#11 library is installed and configured

1. Run the `vastool smartcard info config` command.

To test whether the configured library is valid

1. Run the `vastool smartcard test library` command.

vastool ERROR: cannot open shared object file

You encounter this error when the configured PKCS#11 library does not exist.

To determine that the PKCS#11 library is installed and configured

1. Run the `vastool smartcard info config` command.

To test whether the configured library is valid

1. Run the `vastool smartcard test library` command.

vastool ERROR: smart card is not present in slot

You encounter this error when the card reader is not correctly installed.

For more information, see Check the smart card reader on page 45.

vastool WARNING: "Smart card user X is not unix enabled" issue

Symptom:

A warning displays, similar to the following:

```
WARNING: Smartcard user "vas-user@altsuffix.vas" is not unix enabled.
You will not be able to log in with this card using VAS.
```

Diagnosis:

You will get a warning message that says, "`Smartcard user is not unix enabled.`" because Authentication Services cannot find that user in its cache. Authentication Services 4.x is different from previous versions in that it interprets names in user principal name format as the Active Directory Kerberos principal name, which is actually <sAMAccountName>@<KerberosRealm>. If you have configured your smart cards with the user principal name from Active Directory, but the suffix of the user principal name on your smart card does not match the name of the Kerberos realm for your Active Directory domain, then you are using an alternative user principal name suffix. In other words, your Active Directory domain is COMPANY.COM, but the user principal on your smart card is vas-user@ALTSUFFIX.VAS.

Solution:

Configure **vas.conf** to use user principal name as the logon attribute. This can be done by any of the following methods:

1. Authentication Services Configuration Group Policy Setting:
    a. Open **QAS Configuration** in the Group Policy editor.
    b. Type **username-attr-name** in the search field and click the **Search** button.
    c. Set the value to **userPrincipalName**.
    d. Click **OK** to close the dialog.
    e. Apply Group Policy on the Authentication Services client by running the `vgptool apply` command.
2. Manually edit the `vas.conf`.
    a. Open the `vas.conf` file on the Authentication Services client.
    b. In the `[vasd]` section, set "`username-attr-name = userPrincipalName`".
    c. Save the `vas.conf` file.
    d. Run the `vastool flush` command to repopulate user information.

3. Edit the `vas.conf` with `vastool`.
    a. Run the following command:

    ```
    vastool configure vas vasd username-attr-name userPrincipalName
    ```

    b. Run the `vastool flush` command to repopulate user information.

Troubleshooting PAM or "vastool smartcard test login" errors

The following sections describe symptoms and possible causes that you might encounter when trying to log in with the `pam_vas_smartcard` module or using the `vastool smartcard test login` command.

NOTE: Not all PAM applications display the error messages described in this section. You may need to enable debug, or use `vastool smartcard test login` to display these messages. For more information, see Enable debugging for smart card login with PAM on page 49.

Login fails when the network connectivity is down

You encounter a login failure with a "`KDC is unreachable`" or "`KRB5_KDC_UNREACH`" error message when the network connectivity between the client and Active Directory is down, or there is a configuration problem.

Enabling debug or using `vastool smartcard test login` with `-d 6` help you determine if this is a connectivity or DNS issue.

Login fails when the system's internal clock is not synchronized

You encounter a login failure with a message that says, "`Your system's internal clock is not synchronized with your authentication server`" or "`KRB5KRB_AP_ERR_SKEW`" when your system clock needs to be synchronized with Active Directory.

To synchronize your system clock with Active Directory

1. Run the following command as root: `vastool timesync`.

Login fails when the user account is disabled

You encounter a login failure with a message that says, "The authentication server policy does not allow you to log in at this time.", "KRB5KDC_ERR_POLICY"or "KRB5KDC_ERR_CLIENT_REVOKED" when a user's account has been restricted, locked out, or expired. This message is also displayed when a user, whose account is marked "Smart card required for login", attempts to log in with a password.

Check the user's account settings in Active Directory. For more information, see Check login on page 48.

Login fails when the user's certificate is not authorized

You encounter a login failure with a message that says, "Your certificate cannot be verified by the authentication server" or "KRB5_KDC_ERROR_CANT_VERIFY_CERTIFICATE" when either Authentication Services for Smart Cards was unable to automatically bootstrap the trusted certificates; or, the CA certificate that was used to issue that certificate is not in NtAuthCertificatescontainer in Active Directory. Generally, this error occurs when Active Directory is verifying the user's certificate, or when Authentication Services for Smart Cards is verifying the KDC certificate returned by Active Directory.

For more information, see Bootstrapping trusted certificates on page 36.

Troubleshooting "KDC has no support for padata type" issue

Symptom:

An error displays, similar to the following:

```
KRB5KDC_ERR_PADATA_TYPE_NOSUPP (-1765328368): KDC has no support for padata type
```

Diagnosis:

This error occurs if the domain controller does not have a Domain Controller Authentication Certificate.

Solution:

1. From the Certificates console open the Certificate Request wizard.
2. Select **Domain Controller Authentication**.

3. Click **Enroll**.

Troubleshooting "Cannot contact any KDC for requested realm" issue

Symptom:

An error displays, similar to the following:

```
ERROR: VAS_ERR_KRB5: Failed to obtain credentials. Client: vas-user@ALTSUFFIX.VAS,
Service: krbtgt/ALTSUFFIX.VAS@ALTSUFFIX.VAS, Server: (null)
   Caused by:
   KRB5_KDC_UNREACH (-1765328228): Cannot contact any KDC for requested realm
Reason: unable to reach any KDC in realm ALTSUFFIX.VAS
```

Diagnosis:

You will get an error message that says, "`Cannot contact any KDC for requested realm`" because Authentication Services cannot obtain a Kerberos ticket for the user principal name encoded on the smart card.

This will occur when Authentication Services is unable to communicate with a domain controller. Run the `vastool info servers` command and try to ping your domain controllers to ensure that your network is properly configured and Authentication Services has found a domain controller to use for communication with Active Directory.

If the problem persists, you may have a problem with your user principal name suffix. This occurs when the suffix of the user principal name on your smart card does not match the name of the Kerberos realm for your Active Directory domain. In other words, your Active Directory domain is COMPANY.COM, but the user principal on your smart card is vas-user@ALTSUFFIX.VAS. This means you are using an alternative user principal name suffix.

Solution:

Configure **vas.conf** to use user principal name as the logon attribute. This can be done by any of the following methods:

1. Authentication Services Configuration Group Policy Setting:
    a. Open **QAS Configuration** in the Group Policy editor.
    b. Type **username-attr-name** in the search field and click the **Search** button.
    c. Set the value to **userPrincipalName**.
    d. Click **OK** to close the dialog.
    e. Apply Group Policy on the Authentication Services client by running the `vgptool apply` command.

2. Manually edit the `vas.conf`.
    a. Open the `vas.conf` file on the Authentication Services client.
    b. In the `[vasd]` section, set "`username-attr-name = userPrincipalName`".
    c. Save the `vas.conf` file.
    d. Run the `vastool flush` command to repopulate user information.
3. Edit the `vas.conf` with `vastool`.
    a. Run the following command:

        ```
        vastool configure vas vasd username-attr-name userPrincipalName
        ```

    b. Run the `vastool flush` command to repopulate user information.

Troubleshooting log errors

The following section describes symptoms and possible causes of log error messages when attempting to log in or perform other Authentication Services for Smart Cards functions.

Log shows "clock skew problems"

Log shows "server policy does not allow them on" or "account is expired"

Log shows "Failed authentication attempt: cannot verify certificate"

Log shows "clock skew problems"

You will get a log error message that says, "`clock skew problems`" when you encounter a login failure because your system clock was out of sync with Active Directory.

To synchronize your system clock with Active Directory

1. Run the following command as root: `vastool timesync`.

Log shows "server policy does not allow them on" or "account is expired"

You will get log error messages that say, "`server policy does not allow them on`" or "`account is expired`" when a user's account has been restricted, locked out, or expired; or when a user, whose account is marked **Smart card required for login**, attempts to log in with a password.

Check the user's account settings in Active Directory. For more information, see Check login on page 48.

Log shows "Failed authentication attempt: cannot verify certificate"

You will get a log error message that says, "`Failed authentication attempt: cannot verify certificate`" when Active Directory is verifying the user's certificate, or when Authentication Services for Smart Cards is verifying the KDC certificate returned by Active Directory. The most likely causes are either that the CA certificate that was used to issue that certificate is not in the `NtAuthCertificates` container in Active Directory, or Authentication Services for Smart Cards was unable to automatically bootstrap the trusted certificates.

Check the user's account settings in Active Directory. For more information, see Check login on page 48.

See also Bootstrapping trusted certificates on page 36.

Troubleshooting "`Client not trusted`" issue

Symptom:

An error displays, similar to the following:

```
ERROR: could not establish initial credentials
ERROR: VAS_ERR_KRB5: at ticket.c:72 in ticket_generate_good_error
Failed to obtain credentials. Client: vas-user@SC.VAS, Service:
krbtgt/SC.VAS@SC.VAS
Caused by:
KRB5_KDC_ERR_CLIENT_NOT_TRUSTED (-1765328322): Client not
trusted
```

Diagnosis:

You will get an error message that says, "`Client not trusted`" if Active Directory cannot determine the validity of the client certificate supplied by the smart card, or the validity of any certificate that issued the client certificate.

This may occur for a number of reasons:

1. The Certification Authority service (CA) is not running on the domain controller.

   Active Directory passes the certificate to the CA for verification. If the CA is not running, the certificate cannot be verified and is therefore not trusted.

2. The CA is running, but the Certificate Revocation List (CRL) contained in the certificate is out of date and a new CRL cannot be obtained.

Typically, the CRL is obtained by means of LDAP calls to an external revocation server, and if this server is unreachable or cannot supply a new CRL, the CA cannot check the revocation status of the certificate and the client is therefore not trusted.

Solution:

1. Confirm that the Certification Authority service is running on the domain controller. Navigate to **Start | Administrative Tools | Certification Authority**, and verify that there is a small green tick attached to the "CA" property in the left-hand side of the panel. If there is a small red dot for this property, right click on **CA**, and select **All Tasks | Start Service**.
2. Confirm that the revocation server is reachable:
    a. Attach a smart card reader to the Windows machine for which login is required.
    b. Insert the smart card into the reader.
    c. Open up a command prompt, and run the command `certutil –SCInfo`. This command attempts to verify the client certificate on the smart card, including CRL checks.
    d. Check the output for the following:

```
The revocation function was unable to check revocation
because the revocation server was offline. 0x80092013
(-2146885613)
------------------------------------
Revocation check skipped -- server offline
```

Troubleshooting certificate lookups

Symptom:

Certificate lookups fail.

Diagnosis:

This failure occurs because the default IPC timeout of 5 seconds is insufficient to handle some referrals.

Solution:

Set a sufficient value for the `vascache-ipc-timeout` property in `vas.conf`, as follows:

```
[libvas]
vascache-ipc-timeout = 10
```

About us

One Identity solutions eliminate the complexities and time-consuming processes often required to govern identities, manage privileged accounts and control access. Our solutions enhance business agility while addressing your IAM challenges with on-premises, cloud and hybrid environments.

Contacting us

For sales and other inquiries, such as licensing, support, and renewals, visit https://www.oneidentity.com/company/contact-us.aspx.

Technical support resources

Technical support is available to One Identity customers with a valid maintenance contract and customers who have trial versions. You can access the Support Portal at https://support.oneidentity.com/.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to videos at www.YouTube.com/OneIdentity
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product

Index

G

I

K

L

N

P

R

S

T

U

V

X