Symantec™ Product Authentication Service™ Administrator's Guide

Linux, Microsoft Windows, and UNIX

4.3

Symantec Product Authentication Service Administrator's Guide

Symantec Product Authentication Service Administrator's Guide
Doc Version: 1.6

Symantec Corporation
20330 Stevens Creek Blvd.
Cupertino, CA 95014
www.symantec.com

Printed in the United States of America.

Third-party legal notices

Third-party software may be recommended, distributed, embedded, or bundled with this Symantec product. Such third-party software is licensed separately by its copyright holder. All third-party copyrights associated with this product are listed in the accompanying release notes.

AIX is a registered trademark of IBM Corporation.

HP-UX is a registered trademark of Hewlett-Packard Development Company, L.P.

Linux is a registered trademark of Linus Torvalds.

Solaris is a trademark of Sun Microsystems, Inc.

Windows is a registered trademark of Microsoft Corporation.

Oracle is a registered trademark of Oracle Corporation.

Technical support

For technical assistance, visit http://support.veritas.com and select phone or email support. Use the Knowledge Base search feature to access resources such as TechNotes, product alerts, software downloads, hardware compatibility lists, and our customer email notification service.

Preface

Chapter 1 New Features in This Release

Chapter 2 Symantec Product Authentication Service: Introduction

Chapter 5 The Command Line Interface

Appendix A Debugging and Logging

Appendix B LDAP Plugin Details

Preface

This document is intended for those who will manage and maintain the Symantec Product Authentication Service. We assume that the readers of this document have a basic understanding of computer security systems in general, and the concepts related to them.

This preface covers the following topics:

What's In This Manual?

The table below lists and describes the chapters and appendices in this *Administrator's Guide*.:

Table 1-1 Chapters in This Manual

Chapter	Description
"New Features in This Release"	Features new in 4.3
"Symantec Product Authentication Service: Introduction"	Overview of components, architecture, principals and concepts necessary to an understanding of Symantec Product Authentication
"Symantec Product Authentication Service: Installation"	A cross reference to installation information for Symantec Product Authentication Service
"The Administration Console"	How to access and use the Administration Console
"The Command Line Interface"	How to access and use the command line interface
"Other Management Tasks"	Other aspects of management not dealt with in the Console or the command line interface
"Debugging and Logging"	Debugging and logging for Symantec Product Authentication Service
"LDAP Plugin Details"	Advanced information about the LDAP plugin. Most readers will find what they need in Chapter 3, "Symantec Product Authentication Service: Introduction".

Accessibility

Symantec products meet federal accessibility requirements for software as defined in Section 508 of the Rehabilitation Act:

- http://www.access-board.gov/508.htm

Keyboard shortcuts are available for all major graphical user interface (GUI) operations and menu items. Symantec products are compatible with operating system accessibility settings as well as a variety of assistive technologies. All manuals are also provided as accessible PDF files, and the online help is provided as HTML displayed in a compliant viewer.

Conventions

The following describes typographical and other conventions used in this guide.

Typographic Conventions

Table 1-2 Typographic Conventions

Typeface	Usage
`Bold fixed width`	Input. For example, type **`cd`** to change directories.
`Fixed width`	Paths, commands, filenames, or output. For example: The default installation directory is `/opt/VRTSxx`.
Italics	Book titles, new terms, or used for emphasis. For example: Do *not* ignore cautions.
`Sans serif (italics)`	Placeholder text or variables. For example: Replace *`filename`* with the name of your file.
Serif (no italics)	Graphical user interface (GUI) objects, such as fields, menu choices, etc. For example: Enter your password in the **Password** field.

Notes and Cautions

Note: This is a Note. Notes are used to call attention to information that makes using the product easier or helps in avoiding problems.

Caution: This is a Caution. Cautions are used to warn about situations that could cause data loss.

Key Combinations

Some keyboard command sequences use two or more keys at the same time. For example, holding down the **Ctrl** key while pressing another key. Keyboard command sequences are indicated by connecting the keys with a plus sign. For example:

Press **Ctrl+t**

Chapter 2

New Features in This Release

The following features are new to version 4.3 of Symantec Product Authentication Service:

- Support for the Generic Security Service authentication plugin.
- Remote Identity: The following CLIs now work with remote brokers and contain new parameters for that purpose.
  - `vssat setuptrust`
  - `vssat listpd`
  - `vssat createpd`
  - `vssat addprpl`
- Expanded functionality in remote management of Symantec Product Authentication Service, provided by new API subroutines
- Other new CLI commands:
  - `vssat showispbxexchflag`: Shows whether the PBX Exchange Installed attribute is set on the broker or not. (See “showispbxexchflag” on page 137.)
  - `vssat setispbxexchflag`: Sets the PBX Exchange Installed attribute to either enabled state or disabled state. (See “setispbxexchflag” on page 119.)
  - `vssat showcredinfo`: Displays the principal and domain information of a remotely provisioned identity on the target machine.
- CLIs for LDAP domain configuration.
- New CLI utilities
  - athealth: Used to to perform a quick scan on basic sanity of a particular AT installation. (See “athealth” on page 150.)

- atldapconf: Used for LDAP plug-in configuration.(See “atldapconf” on page 151.)

- Limited support is offered to the PAM plugin wherein, call back functionality is not supported for the PAM plugin.
- DefaultAuthSequence: To simplify the usage of PAM plugin, all the relevant plugins on a platform are brought under a common authentication type that is, unixpwd. The plugins are chained in a configurable order and are tried sequentially, until the authentication succeeds. unixpwd plugin is enhanced to iterate in a sequence of available plug-ins. The sequence of plugins to be tried under unixpwd domain type is specified in the broker configuration file. DefaultAuthSequence parameter is added under the Authentication Broker section. By default, DefaultAuthSequence parameter is set to ‘pam unixpwd nisplus nis’ on Unix platforms and to ‘nt ‘on Windows. During the broker startup, unixpwd plugin searches for DefaultAuthSequence parameter to identify the default authentication sequence. If it is absent, the default value is stored in the VRTSatlocal.conf file. You can manipulate DefaultAuthSequence parameter manually.
- Both the broker and the LDAP configuration related CLIs are enhanced to change and save the clear text bind password in obfuscated form. Additionally, the LDAP configuration related listldapdomain CLI when retrieving a LDAP domain, displays password in obfuscated form.
- Broker credential renewal enahncement enables the brokers to keep track of the validity period of their credentials and automatically start renewing the credentials one year before the expiry of the existing credentials.

Chapter 3

Symantec Product Authentication Service: Introduction

This section of the document provides an introduction to Symantec Product Authentication Service. It covers the following topics:

Background Terminology

A glossary is included as the last chapter in this document. The present chapter discusses, in greater depth than possible in a glossary, certain concepts necessary to a basic understanding of Symantec Product Authentication Service.

Authentication

Authentication is the validation of an identity. It works like this:

Who are you?

I'm Pat.

Prove it.

OK. Here are materials to prove who I am.

I'm convinced. Here's a certificate to indicate that I believe you're Pat. Combine it with your private key to make a valid product credential.

Authorization

Authorization is, basically, access control, that is, the determination of who is allowed to do what on what. It works like this:

I've got proof that I'm Pat. As Pat, I want to do X on YourProduct Resource Y.

I believe that you're Pat. Now let me consult the rules to see if you're allowed to do X on YourProduct Resource Y.

Security Policy

A security policy is a well-thought-out set of decisions regarding how your product should be used in a customer's environment, how your product could be misused, and what range of access rules your customers would like to see enforced by your product.

In order to set the range of allowable policy, you need to think about:

- Who/what should use your product at all.
- Who/what should be able to perform which tasks on which resources.

Secure Sockets Layer Protocol

Secure sockets layer (SSL) is a public key protocol originally created by Netscape and used for secure communications between clients and servers over the Web. For example, SSL protocol is often used for the transmission of credit cards and other sensitive data over the Internet.

The name refers to the fact that SSL runs in the network layer of a communications packet, above the TCP/IP layer and below the areas used for high-level application protocols such as HTTP.

With SSL, each encrypted transaction generates a session key, which may be either 40 bits or 128 bits. The longer the session key, the stronger the security.

When using an SSL connection, all communication between client and service is encrypted by the sender and decrypted by the receiver. The protocol can offer both client authentication and service authentication.

SSL and Symantec Product Authentication Service

SSL technology provides secured communication among the Symantec application client, Authentication Broker, and Symantec application service.

During the authentication process, the client uses the SSL layer for communicating with the Authentication Broker in order to request a product credential. Once the principal is authenticated, an SSL connection is established between the Symantec application client and the Symantec application service. The client and service interact over the SSL connection, sending and receiving messages as necessary until the client terminates the communication.

Nature and Purpose of Authentication

Symantec Product Authentication Service is responsible for:

- Validating identities, and therefore forming the basis for authorization and access control
- Protecting communications channels between Symantec application clients and Symantec application services through message integrity and confidentiality services

Since Symantec Product Authentication Service can use a variety of authentication mechanisms, it supports both existing and future authentication domains and technologies across multiple operating system environments.

Optionally, the security administrator can configure authentication to provide a single sign-on service for Symantec applications. In this case, users need log-on only once to a single Symantec application, and other applications can use the credentials acquired through the first logon.

Single Sign-On Authentication Through the SSPI Plugin

The Security Services Provider Interface (SSPI) from Windows provides a set of authentication and communication security services between applications running on Microsoft platforms.

Symantec Product Authentication Service works with the SSPI plugin to allow for unified login with Microsoft platforms. The user does not have to enter another password.

For example, assume that a user is already logged in to a Windows machine using the account/password of an NT domain. Symantec Product Authentication Service uses SSPI to acquire a credential. This SSPI connection is from the Symantec application client to the Authentication Broker. The communication from Symantec application client to Symantec application service is mutually authenticated SSL. Upon authentication at the Authentication Broker, Symantec Product Authentication Service acquires all the OS groups that the principal belongs to.

Authentication Principals

An authentication principal is an entity that has the ability to authenticate to Symantec Product Authentication Service with a unique identity. Typically an authentication principal represents a human user. However, it could also be a computer, a service, or a process such as a command line interface (CLI) whose identity is not related to any human being.

Most Symantec resource management applications are themselves authentication principals. That is, they will have identities distinct from the operating system login account identity under which they execute, and they will be authenticated based upon their identities. When we are talking about anything being authenticated, we will use the term authentication principal (as distinct from security principal).

Components: Overview

Symantec Product Authentication Service is a distributed application that is comprised of the following components:

- Authentication Broker (Server)
- Authentication Client (Runtime)
- Authentication plugins
- Administration UI (CLI and GUI)
- SDK used by products group (Java, C API's)

Description of Components and Participants

This section describes the components and participants in an Authentication setup.

- Application Host
  The machine on which a Symantec application is running.
- Authentication Broker
  Component that serves, one level beneath the Root Broker, as an intermediate registration authority and a certification authority. The Authentication Broker can authenticate clients, such as users or services, and grant them a product credential. It cannot, however, authenticate other brokers. That task must be performed by the Root Broker.
- Authentication Library
  Component that links with a Symantec application client and implements the program calls that request authentication. Conceptually, this library is separate from the part of authentication responsible for securing communications, but in practice, both components may be part of a single library.
- Authentication Mechanism
  The method by which authentication is conducted for principals in a specific name-space defined by a domain. For example, a Kerberos domain uses Kerberos tickets and password. In UNIX platforms, Kerberos domains are used through the GSS-API. An authentication mechanism encapsulates all the details of the authentication algorithm, including APIs, protocols, token formats, token contents semantics and database objects formats. Not all the ingredients are relevant in all mechanisms.
- Authentication Plugin
  A component used by the Authentication Broker to validate identities within a particular domain. An authentication plugin exists for each supported authentication mechanism. For example, one plugin can validate NIS identity and password combinations against an NIS database, while another uses a Kerberos ticket to authenticate the principal.
  Not all plugins need to exist in all platforms. For example, any given Symantec application client usually requires authentication against only one or, perhaps, two domains, thus requiring just one or two plugins (unless a single plugin, such as the password plugin mentioned above, can serve both domains). For instance, an Authentication Broker that runs on a UNIX machine won't be able to use a plugin that is Microsoft-specific.

- Communications Library
  A part of Symantec Product Authentication Service responsible for providing secure communication between a Symantec application client and a Symantec application service, using the Symantec credential acquired in a preceding authentication interaction. Conceptually, this library is separate from the part responsible for authenticating identities, but in practice, both components may be part of a single library.
- Resource Management Application
  A Symantec product whose resources are being protected by Symantec Product Authentication Service and Symantec Product Authorization Service.
- Root Broker
  The first Authentication Broker, which has a self-signed certificate. The Root Broker has a single private domain that holds only the names of brokers that shall be considered valid. The name of the Root Broker itself is stored as the fully qualified domain name.
- Symantec Application Service
  A program that is contacted by, and provides services to, a Symantec application client.
- Symantec Application Client
  A program that accesses a Symantec service or function provided by another program, called a Symantec application service. An example of a secured application client is the Symantec Volume Manager GUI. A Symantec application client uses authentication to validate the ID of the user of that client.
- Interface
  The component that allows the security administrator to manage various aspects of Symantec Product Authentication Service. There are two types of interface: CLI and Administration Console.
- SDK
  The software developers kit used to software groups internal to the company and made up of Java and C APIs.

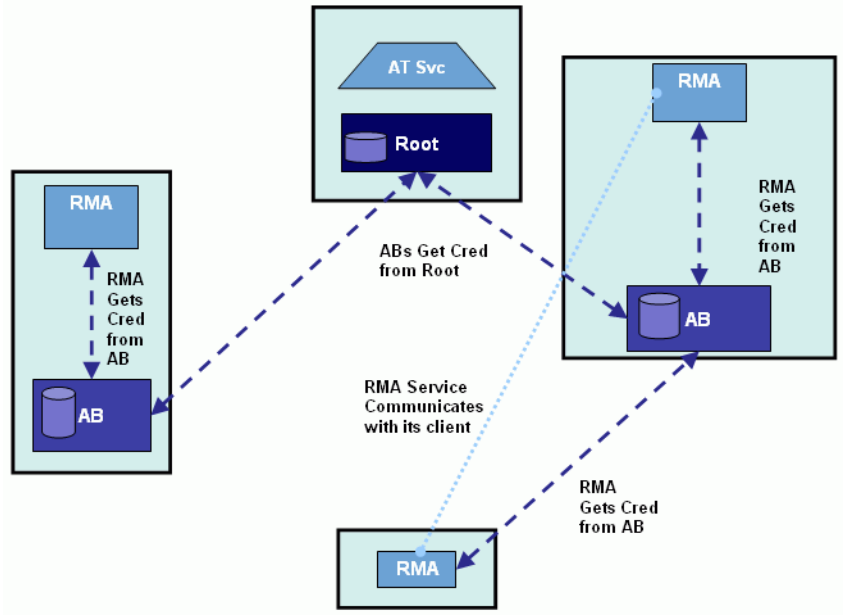
Diagrams of Participants and Architecture

The diagrams below indicates how these components can work together. In the first diagram (“Diagram of Participants and Architecture”):

- The Root Broker is shown authenticating the Authentication Brokers.

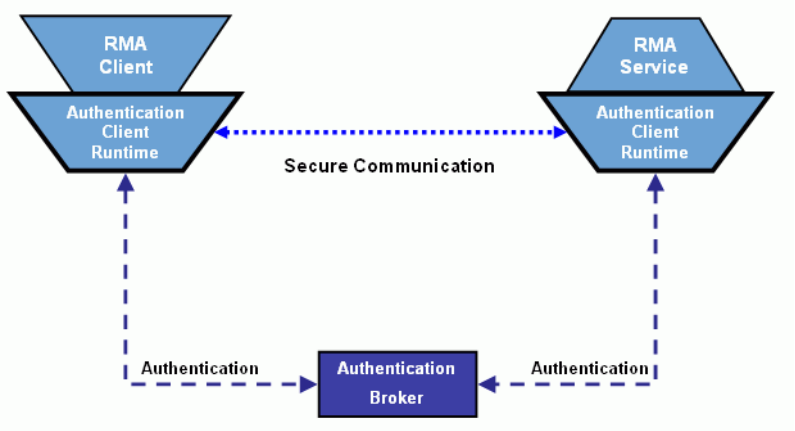
- The Authentication Brokers are shown authenticating resource management applications (RMA).
- A Symantec application service, which is running on the application host, is shown communicating with one of its clients:

Figure 3-1 Diagram of Participants and Architecture



The second diagram (“Resource Management Application and AT”) shows that both the RMA client machine and the RMA service machine must have an Authentication client installed.

Figure 3-2 Resource Management Application and AT



For deeper looks inside the Root Broker and Authentication Brokers, see diagrams in “Types of Brokers”.

Types of Brokers

To understand authentication, it is necessary to distinguish between the two types of brokers, Root Broker and Authentication Brokers.

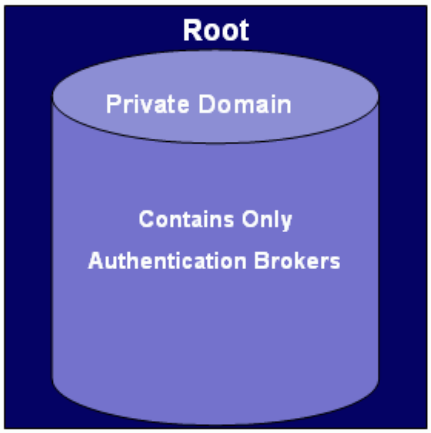
The Root Broker

When you install Symantec Product Authentication Service, the first broker established is called the *Root Broker*. It resides at the top level of the authentication hierarchy tree. The Root Broker is special enough that it will never be referred to simply as an “Authentication Broker”. Instead, it will always be distinguished as the *Root* or *Root Broker*.

The Root Broker is more trusted and has more power than Authentication Brokers.

- The Root Broker validates itself with self-signed credential
- It can authenticate other brokers
- It has a PDR with ONLY Authentication Brokers

Figure 3-3 Root Broker Host



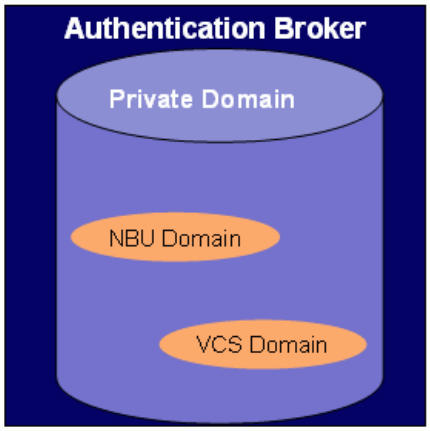
Authentication Brokers

There may be a number of Authentication Brokers besides the Root. Authentication Brokers reside one level below the Root Broker in the Authentication Broker tree. Each Authentication Broker serves as an intermediate registration authority and a certification authority that can authenticate clients, such as users or services.

Authentication brokers are comprised of two main sub-components:

- The registration authority, which determines whether the authentication principals are valid.
- The Certificate Authority, which issues a product credential to the requesting authentication principal based on the Registration Authority's validity check.

Figure 3-4 Sample Authentication Broker



Each Authentication Broker has a private domain repository which stores the services and users of any resource management applications choosing to use this broker. The figure above shows domains for NBU and VCS. Authentication Brokers will also have various plugins.

In some cases, an Authentication Broker may reside on a dedicated machine. Probably more often, an Authentication Broker will reside on an application host.

Summary: How Root Brokers Differ From ABs

The Root Broker differs from Authentication Brokers in the following ways:

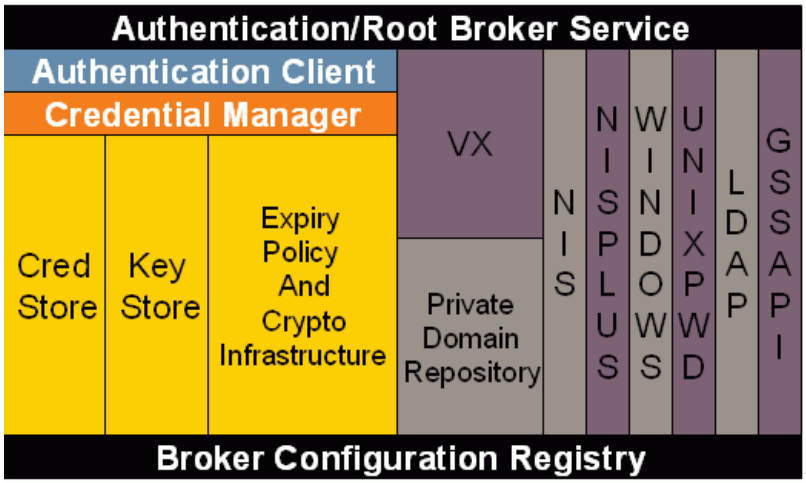
- The Root Broker has a self-signed certificate; Authentication Brokers have certificates signed by the Root.
- The Root Broker can authenticate other brokers; regular Authentication Brokers can authenticate clients and services, but not other brokers.
- The Root Broker has a special kind of private domain.

Caution: The administrator of the Root Broker must put only Authentication Brokers in the Root Broker's private domain. The name of the Root Broker itself is stored there as the fully qualified domain name.

Deeper Look at Broker Architecture

The diagram below provides an even deeper look at Broker service architecture:

Figure 3-5 AB/Root Broker Service



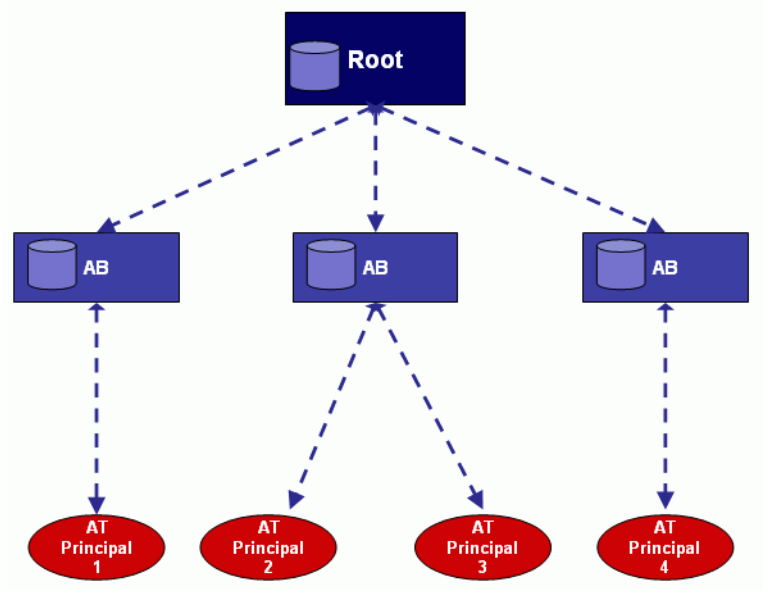
Authentication Broker Tree: Certificate Hierarchy

The Symantec Product Authentication Service implements a three level certificate hierarchy, which we call the *Authentication Broker Tree*. These are the three levels:

- Level one: The Root Broker
  In each Authentication Broker Tree, the Root Broker acts as the root certification authority and the registration authority for Authentication Brokers. The Root Broker has a single private domain that contains only Authentication Brokers.
- Level two: The Authentication Brokers
  Other Authentication Brokers may have 0 or more private domains whose members may be of various types -- users, services, command line interfaces. They are distinguished in the PDR by having attributes to each identity.
- Level three: The application clients and application services

The diagram below illustrates:

Figure 3-6 Diagram of Authentication Broker Tree



Ports Used by the Authentication Brokers

All Authentication Servers (Root Broker or Authentication Broker or Root + Authentication Broker) listen on port 2821, which is registered with IANA. This port number can be changed using the `vssat` command through the command line interface. (See "Using vssat" on page 87.) While connecting to an Authentication Broker or Root Broker or Root + Authentication Broker, you need to specify 2821 as the port number. The Administration Console and the administration commands of the command line interface default to 2821 when connecting to an Authentication Service.

The Authentication Broker can also be configured to receive requests from PBX service. In such configuration, it will also receive requests on the common PBX port 1556.

How Client Knows What Broker to Use

The security administrator can choose which Authentication Broker the client should contact for each domain from which it wants to authenticate. Domain-Broker mapping is the set of information telling which Authentication Broker to approach for each domain. For example:

- If I am a nis+ client, use Authentication Broker A.

- If I am a unixpwd client, use Authentication Broker B.

There are two ways in which a Symantec application client can obtain the identity and location of Authentication Brokers that are trusted by the Symantec application service:

- Either from the local configuration or
- From information provided from a Symantec service. For example, the Symantec Enterprise Architecture service could tell the client about a co-located or remote broker.

Credentials and Certificates

It is easy to confuse the two terms *certificate* and *credential*. They are not synonymous.

- A certificate is a type of electronic passport or ID card that vouches for the identity of its holder and binds the principal's name to his or her public key.
- A product credential (or “credential” for short) is an entitlement to be recognized as an authenticated principal. It does include a certificate, but it must also include the principal’s private key. Without both parts, it is not a credential.

Note: The names of API subroutines use the term "credential" in many circumstances where the term "certificate" would be more accurate. This occurs because the authentication library does not require the customer to understand the intricacies of PKI certificates in order to use the API effectively.

Product Credential

The product credential is an entitlement to be recognized as a valid identity. A product credential requires both (1) the principal’s private key and (2) a X.509v3 certificate with special extensions, produced and signed by the Authentication Broker or Root Broker, to bind the principal’s name to the public key. The product credential is a single sign-on credential valid for all Symantec applications that are enabled to use Symantec Product Authentication Service and that choose to participate in the Symantec single sign-on session.

Who Must Have Product Credentials

Any Symantec resource management application has two components, both of which must be authenticated with valid credentials:

- Symantec application client

A Symantec application client (*client* for short) is a program that accesses a Symantec service or function provided by another program.

- Symantec application service
  A Symantec application service (*service* for short) is the program that provides the requested services.

Both the client and the service act as clients when seeking authentication. The credential differs depending upon whether it is being granted to an application client or an application service. Generally credentials granted to a service have a longer life span than those granted to a client.

Extended Attributes of the Product Credential

Some extended attributes in the product credential are group membership of the principal and the IP address of the machine to which the credential is being granted. Using these extended attributes lets the communications library enforce certain communications security policies. For example, assume that the principal belongs to a group called Symantec Execs. The policy might specify that all communications to members of this group should be encrypted with a 128-bit encryption algorithm or better, unlike communications to members of other groups.

Who Signs Certificates

Certificates vouching for the authenticity of principals are signed by a hierarchy of certification authorities. The root certification authority (on the Root Broker) is the entity at the top of the hierarchy and is therefore the most trusted certification authority. Its certificate, vouching for itself, is self-signed and is called the root certificate.

The application program environment includes the set of root certificates that the authentication library will accept as valid signers of credentials. This information can be either pre-configured or obtained from the Symantec application service. In principal, this root certificate could be stored as part of the registry configuration. In practice, however, it is traditionally kept in a separate file, or set of files, following one of a small collection of standard formats:

- PKCS#12 as the “standard”
- JKS for Java JSSE use
- PEM in OpenSSL and for S/MIME implementations

How Long Credentials Are Valid

There is no single set life span for credentials.

When the administrator of the Authentication Broker creates an ID, he or she can indicate what kind of security principal the ID is for: for example, a service, a command line interface, a user. This information goes into the private domain, and it influences the longevity of the credential the entity will receive.

Note: This ID information is ***not*** put into the Root Broker's private domain; it belongs in the private domain for a ***non***-root Authentication Broker. In some situations, an administrator can choose to use an NT/NIS domain (non-private domain) instead of a private domain. If he or she does so, the Authentication Service may not have knowledge of whether the principal is a service or a user.

You probably want the product credentials issued to Symantec application services to have longer life spans than those issued to Symantec application clients, since a service must be up and running for prolonged periods of time.

The longevity of the credential determines its volatility in the machine that issued the credential. With Public-Key technology, associated with the X.509 certificate, there is a private-key that is kept only in the client. The credential and the private-key associated with it are stored on-disk and are kept obfuscated using a proprietary obfuscation mechanism internal to Symantec Product Authentication Service.

Special Types of Credentials

Two special kinds of credentials are:

- Proxy credential
  A special long-term credential with proxy rights. It is obtained during configuration of the web console, on behalf of the "webconsole" user. The proxy rights of this credential let the web console proxy on behalf of the actual user.
- Product web credential
  A special kind of credential that tells Symantec Product Authentication Service that there is no corresponding private key stored in the library. A product web credential must be used along with a proxy-capable credential of the web console.

Credential Life Cycle

Credentials have limited validity. Each credential has `notvalidBefore` and `notValidAfter` times. The period between these two times is the expiry period and is controlled by the Authentication Broker.

Expiry policy is configurable

The stages of the credential life cycle are:

- Beginning: Credential is added to store, through the Administration Console or through the CLI.
- Middle: Credential is used
- End: Credential expires or is destroyed, through the Administration Console or through the CLI.

In some cases, a credential is renewed and begins a new life cycle.

Bootstrapping Authentication

This section explains the bootstrapping process.

The steps performed to bootstrap an authentication sub-system are

1. The Root Broker is installed.
2. The Root Broker generates its own key pair and self-signed certificate.

Note: The process for key-pair generation is transparent.

---

3. A Symantec Product Authentication Service administrator configures the identities of Authentication Brokers in the Root Broker's authentication private domain repository.
4. When an Authentication Broker is installed, the Authentication Broker is configured to know the Root Broker's host name or IP address and port.
5. The Authentication Broker connects to the Root Broker and identifies itself with the password provided by the administrator.
6. The Root Broker verifies the identity of the Authentication Broker and generates the Authentication Broker certificate.
7. The Root Broker ships the certificate chain to the Authentication Broker.
8. The Authentication Broker then authenticates s and issues certificates. That is, the Authentication Broker acts as an intermediate certification authority.

Note: Security principals must trust the Root Broker's certificate, and the root certificate must be distributed to all clients of the Authentication Service.

How Root Certificates are Distributed

For a very high security level, we recommend manual distribution of the root certificate, for example on a floppy disk. In other cases, the root certificate is downloaded by the client/application by using the trust establishment API.

The security administrator can configure one of the following levels of security for distributing root certificates:

- High security: If a previously untrusted root is acquired from the peer (that is, if no certificate with the same signature exists in our trust store), the user will be prompted to verify the hash.
- Medium security: The first authentication Root Broker will be trusted without prompting. Any attempts to trust subsequent authentication Root Brokers will cause the user to be prompted for a hash verification before the certificate is added to the trusted store.
- Low security: The Authentication Broker certificate is always trusted without any prompting.

Note: Trust relationships can be inherited.

Steps Involved in Authenticating and Communicating

The steps we present for authenticating a client and establishing communication with a Symantec application service assume two stages:

- Getting the product credential
- Using the credential

Getting the Product Credential

Before the steps below are carried out, both the client and the service have generated a key pair locally. For each, this key pair includes both a private key and a public key.

Note: The client or service does not have to perform specific tasks to generate the key pair. The process is transparent.

The process works like this

1. The Symantec application client calls a function to initialize the authentication library for both the client and service. This operation creates an “instance” of the authentication library.
2. The Symantec application client makes programmatic calls to get a list of trusted Authentication Brokers and supported domains.
   See “Local Host Validation” for information on validating the principal on the local host without having to contact a remote Authentication Broker.
3. An SSL connection is established with the Authentication Broker.
   SSL is a public key protocol originally created by Netscape and used for secure communications between clients and servers over the Web. In the context of Symantec Product Authentication Service, Secure Sockets Layer technology provides secured communication between the Symantec application client, Authentication Broker, and Symantec application service.
   It is up to the users of the authentication client library to decide what to do if an Authentication Broker is unreachable. The returned error code from the authentication API will specify the actual error, and the user can either retry or handle the error in some other manner.
4. The user sends authentication materials to the Authentication Broker.
   These materials include the public key, identity, password, domain, and domain type. The exact nature of materials depends upon the type of authentication mechanism used. The type of mechanism is indicated by the domain in the input parameter. Some mechanisms require more than one round-trip communication with the broker to complete authentication.
   If the user does not provide information (i.e. if there is NULL input) the authentication library may re-use an existing product credential for the current instance or use an enterprise single sign-on credential to authenticate the user.
5. The Authentication Broker checks its authentication private domain repository to see whether the entity seeking authentication should be considered valid.
6. If the entity is deemed valid, the Authentication Broker uses its own private key to sign the public key of the entity striving to be authenticated. In doing so, it forms an X509 certificate.
7. The Authentication Broker sends the X509 certificate back to the client or service seeking to be validated.
8. On the client side, the signed certificate and the client’s own private key become the product credential.

9 The SSL connection with the Authentication Broker is closed.

Using the Credential to Establish a Secure Session

Once both a client and a service have product credentials, they use them to establish a secure session.

Types of Communications Security

To set the security level of connection between a Symantec application client and a Symantec application service, Symantec Product Authentication Service uses an explicit parameter passed by the application to the `vrtsAtSecConnAccept` function of the authentication communications API.

The three types of data communications security are:

- High Security (level 0): VRTSAT_COMM_VERIFY_ENCRYPT: Provides encrypted and signed data channel. There is mutual authentication in the handshake phase.
- Medium Security (level 1): VRTSAT_COMM_VERIFY_NONE: Provides encrypted and signed data channel. There is no mutual authentication in the handshake phase. This level should be used in a trusted environment only.
- Low Security (level 2): VERIFY_COMM_ENCRYPT_NONE: Provides a signed channel only. There is mutual authentication in the handshake phase. This level may be used on point-to-point links where snooping is not an issue.

Note: Peers should use the same communication level.

We recommend using medium security or higher. In all of the above, both peers must have a valid credential.

To establish a secure session

1 The entities use the communications API to initiate the session:

   a The client uses an API for initiating.

   b The service uses an API for responding.

2 Both peers pass their product credentials to the communications API.

3 The communications API exchanges the two entities' credentials in a handshake where each peer verifies that the other peer is valid.
   To do this, each checks that the other's credential is signed by an Authentication Broker and Root Broker that it trusts.

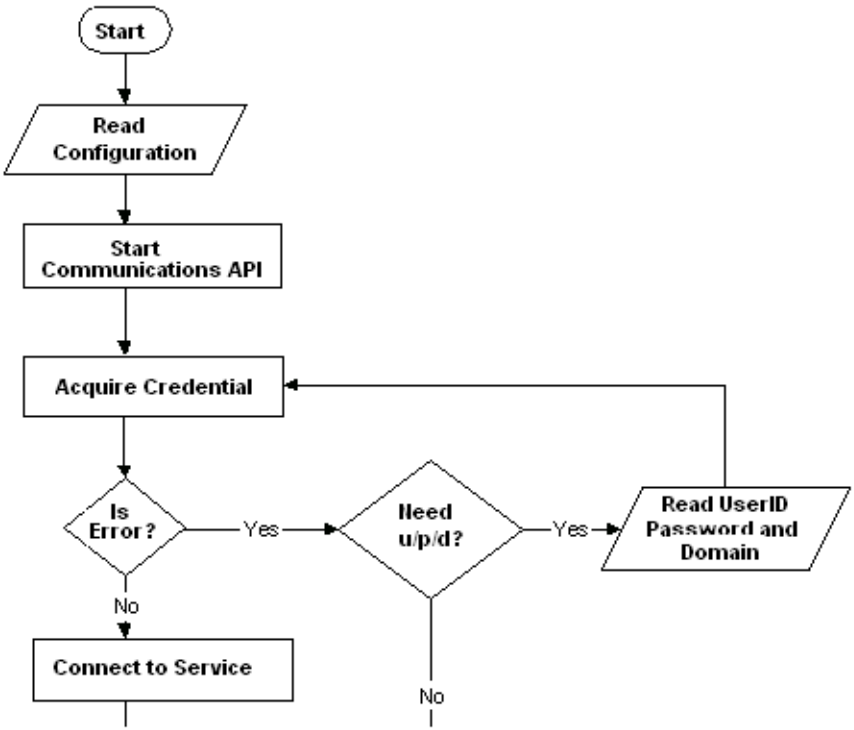
4 Once the handshake -- indicating mutual trust -- is completed, a secure session is formed, and the client and service are ready to exchange data.

5 The application client requests a secured connection with the Symantec application service.

Flow Diagram of the Process

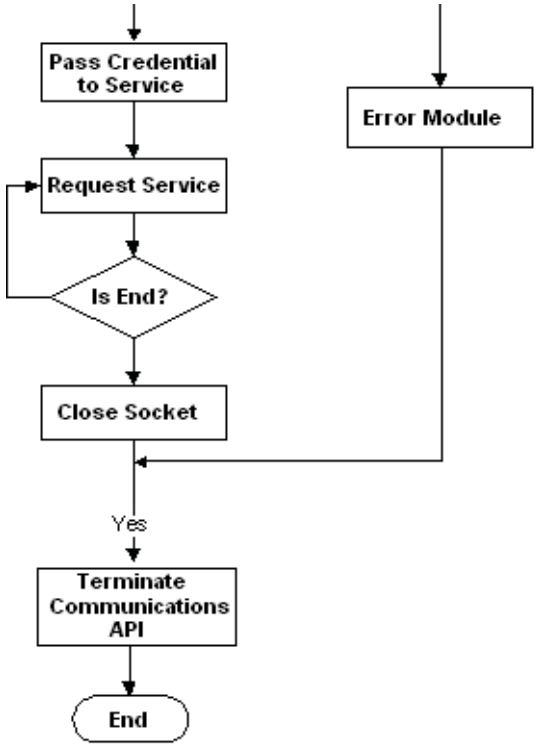
The two-part figure below depicts a program flow for a typical application client when using Symantec Product Authentication Service. Part 1 shows to the point of connecting to the service.

Figure 3-7 Flow Chart of Client Using Authentication



Part two of the flow chart starts at the point where the connection has been made to the service.

Figure 3-8 Flow Chart (continued) of Client Using Authentication



Single Sign-On Authentication

Single sign-on authentication lets you leverage existing single sign-on mechanisms, such as Kerberos or Microsoft Unified Logon.

To use single sign-on

1. The user authenticates itself to the enterprise single sign-on system.
2. When the user is validated, it obtains a single sign-on credential.
3. The user starts the Symantec application.
4. The Symantec application instructs the authentication library to use the single sign-on credential. If more than one single sign-on credential is available, the domain information in the authentication materials will indicate which credential to use.

5 The single sign-on credential is sent to the Authentication Broker.

6 The Authentication Broker accepts the single sign-on credential and produces a product credential to be used by all applications that are enabled to use Symantec Product Authentication Service and that are configured to leverage the enterprise single sign-on service.

To leverage enterprise single sign-on capabilities, the Symantec application client does not need to know what form of enterprise single sign-on system exists or what type of credential it uses. Such knowledge is the responsibility of the authentication library. An application program that is either configured or programmed to use the enterprise single sign-on service only needs to know that it should not prompt the user for a user-name and a password.

Local Host Validation

A resource management application (RMA) is a Symantec product whose resources are being protected by Symantec Product Authentication Service. The term *application host* refers to the machine on which a particular Symantec RMA has been installed.

Local Host validation is the process of validating the principal on the local host without having to contact a remote Authentication Broker. Local host validation is implemented in the client library itself.

Here is how it works:

1 The Symantec application client or service requests local host validation. The authentication client library then tries to verify the identity information passed in. The information can be passed in as the current identity's login context or as a tuple consisting of identity, domain, and domain type and password.

2 If the verification succeeds, then the client library generates a self- signed certificate.

3 This certificate can be passed by the client to the application service running on the same host.

4 The application service can use the `vrtsAtExtractInfo` API to get the details of the principal.

Note: A credential acquired through the local host validator can be used for communications security only if the entities are running at low security level.

Some uses of this feature involve clients that do not connect to other machines. In this case, the credential is not used for communications. Therefore, only the successful authentication is relevant.

Authentication When Using a Web Console

The authentication scenario is slightly different for web consoles, which look and behave much like desktop consoles, but which can be accessed through the internet. This is because the client, for web consoles, is a browser, and a browser cannot store and present a product credential.

Web consoles, therefore, make use of a special kind of credential called a product web credential, which works in conjunction with a proxy-capable credential.

For details on configuring to work with web consoles, see the *Installation Guide*.

Authentication Mechanisms

Each domain has either an implicit way for authenticating users (NIS+, authentication private domains) or an explicit authentication API, with corresponding protocols and algorithms.

An authentication mechanism is the way authentication is conducted for users in the name-space defined by a domain. An authentication mechanism encapsulates all the details of the authentication algorithm, including APIs, protocols, token formats, token contents semantics and database objects formats. Not all the ingredients are relevant in all mechanisms.

Domain Mapping

The Symantec Product Authentication Service associates a domain with a single authentication mechanism. For example:

- A Kerberos domain uses Kerberos tickets and passwords. In UNIX platforms, Kerberos domains are used through the GSS-API.
- An NT domain can also use Kerberos as the underlying mechanism, but the user name-space is distinct from that of the Kerberos domain used by UNIX. Moreover, in Windows 2000, Kerberos functionality is exercised through the SSPI interface, rather than the GSS-API used in UNIX.

Each of these domains map to different authentication plugins.

The converse is not true. That is, each authentication plugin does not have to map to a different domain. The Kerberos plugin, for example, might service a number of different domains in a single NT network.

Plugins

For each supported authentication mechanism, there is a plugin, a component used by the Authentication Broker to validate identities within a particular domain. Plugins map 1:1 to domain types.

For example, one plugin can validate NIS identity and password combinations against an NIS database, while another uses a Kerberos ticket to authenticate the principal. Some types of plugins are:

- nis – NIS v2 (formerly called YP – Yellow Pages)
- nisplus – NIS v3
- ldap – OpenLDAP and iPlanet implementations
- GSS -- Generic Security Service Application Program Interface (GSS-API)
- pam – Pluggable Authentication Mechanism (for UNIX)
- nt – Windows NT domains and Active Directory (SSPI)
- vx – Symantec Private Domain

Not all plugins need to exist in all platforms. Any given Symantec application client usually requires authentication against only one or, perhaps, two domains, thus requiring just one or two plugins (unless a single plugin, such as the password plugin mentioned above, can serve both domains). For instance, an Authentication Broker that runs on a UNIX machine won't be able to use an authentication plugin that is Microsoft-specific.

The Symantec LDAP Plugin

Symantec Product Authentication Service supports LDAP Authentication by using a plugin module that is designed to support RFC 2307 by default. The LDAP Authentication plugin module is called `authldap` and is a shared library, which ships with the Authentication Broker.

LDAP Deployment Recommendations

Our recommendations for deploying the LDAP Authentication plugin are as follows:

- Use the schema specified in RFC 2307 when storing NIS data in LDAP directories.

- Configure to communicate to a single LDAP directory server per domain at any given time.
- Deploy with Secure Socket Layer (SSL) enabled.

Explanation of LDAP Recommendations

Symantec Product Authentication Service supports LDAP Authentication by using a plugin module. The LDAP Authentication plugin can be configured to communicate to a single LDAP directory server per domain at any given time.

Note: Because data, such as user name and password, are unprotected during transmission, we strongly recommend users to deploy the LDAP Authentication plugin with Secure Socket Layer (SSL) enabled.

Who Can Enable the LDAP Plugin Authentication

Organizations who manage their user and network data in Lightweight Directory Access Protocol (LDAP) directories, can utilize Symantec Product Authentication Service's LDAP Authentication plugin to perform user authentication.

Note: The LDAP plugin is primarily intended for UNIX users. It has not been tested with Windows Active Directory.

Basic Steps for Enabling LDAP

Use the following procedure to enable the LDAP plugin.

To enable the LDAP plugin

1. Shut down the Symantec Product Authentication Service server.
2. Use the addldapdomain CLI command to add the LDAP domains. For example:

   ```
   vssat addldapdomain --domainname MYADDOMAIN --server_url ldap://
   my_ad_host.mydomain.myenterprise.com -u
   cn=users,dc=mydomain,dc=myenterprise,dc=com --group_base_dn
   dc=users,dc=mydomain,dc=myenterprise,dc=com --schema_type msad
   ```

3. Restart the Symantec Product Authentication Service server. You do not need to reboot the machine.

CLI commands for LDAP configuration

CLIs exist for adding an LDAP domain, removing an LDAP Domain, and listing the LDAP domains. For details, see "addldapdomain," "listldapdomains," and "removeldapdomain."

The GSS-API Authentication Plugin

The Generic Security Service Application Program Interface (GSS-API) authentication plugin supports a maximum number of different GSS-API mechanisms with minimal configurations required. All configuration occurs at the Authentication Broker. No Client-side configuration is required.

For more information about GSS-API, please see RFC 2743 and RFC 2744.

Configuring the GSS-API Plugin

Sections for the GSS-API plugin are added to `VRTSatlocal.conf`. The main `gssapi` section lets users configure the service name (`ServiceName`) and indicate whether to include the exported principal name in the product credential (`ExportGSSName`).

The `ServiceName` is the name of the service that the GSS-API server is providing. It is the target name used in `gss_init_sec_context()`.

The `ExportGSSName` flag, if set to a non-zero value, tells the Authentication Broker to include the exported principal name in the product credential. The exported principal name can be extracted from the product credential using `vrtsAtGetExportedName()`.

A typical GSSAPI configuration section would looks like this.

```
[Security\Authentication\Authentication
Broker\AtPlugins\gssapi]
"PluginSharedLibFileName"="authgssapi.dll"
"IsEnabled"=dword:00000000
"ServiceName"="host"
"ExportGSSName"=dword:00000001
[Security\Authentication\Authentication
Broker\AtPlugins\gssapi\CA]
"WebExpiryInterval"=dword:00007080
"ExpiryInterval"=dword:00015180
"UserExpiryInterval"=dword:00015180
"ServiceExpiryInterval"=dword:01e13380
[Security\Authentication\Authentication
Broker\AtPlugins\gssapi\DomainInfos]
[Security\Authentication\Authentication
Broker\AtPlugins\gssapi\DomainInfos\gssapi]
```

As is true for other plugins, the "CA" section is configurable as well.

The pluggable authentication mechanism (PAM)

Pluggable Authentication Mechanism (PAM) is a framework that lets you add and remove authentication mechanisms for the integrated services and applications without having to change the services themselves. With the PAM plugin, you can use your existing PAM authentication modules to authenticate clients through Symantec Product Authentication Service and acquire product credentials.

PAM is available on UNIX platforms only. Windows does not support it.

The PAM plugin is a dynamic link library that gets loaded on demand with a default PAM service name of “login”. You can enable or disable the plugin or change its service name through a CLI or through the GUI.

To change the plugin name, run the following:

```
vssat  updateplugin --pluginname pam --attribute ServiceName
--value "NewName"
```

To enable the plugin, run the following:

```
vssat updateplugin --pluginname pam --attribute DoNotLoad
--value 0 --type int
```

To disable the plugin, run the following:

```
vssat updateplugin --pluginname pam --attribute DoNotLoad
--value 1 --type int
```

To enable the plugin with a new name, disable the plugin first, and then change the name and enable again.

If a CLIs accepts a domain type, it accepts pam as a supported domain type.

Note: For this release, limited support is offered to the PAM plugin wherein, call back functionality is not supported for the PAM plugin.

Chapter 4

Symantec Product Authentication Service: Installation

See the Symantec Product Authentication Service *Installation Guide* for information on installing and configuring. Also see the README.txt files and the product Release Notes.

Chapter 5

The Administration Console

This chapter begins with a general description of the Administration Console and then explains how to access it and how to use it to carry out various management tasks.

Sections include:

Preparing to Run the Administration Console

Caution: Consult the Readme for up-to-date instructions pertaining to the specific build of Symantec Product Authentication Service you have installed.

The Administration Console does not have a separate installable. It is installed as part of the basic installation process.

Location of Binaries

On a Windows Platform:

- Authentication: All binaries and script files can be found at `<authentication install directory>`\bin

On a Solaris Platform

- Authentication
  - libAtWrapper.so, AtWrapper.jar, vssatgui.jar, VxHelpViewer.jar and VxHelpViewerl10n.jar can be found at <*authentication install directory*>/lib.
  - runvssatgui.sh can be found at <*authentication install directory*>/bin

Prerequisites

To run the Administration Console, you must meet the following prerequisites:

- For AIX, make sure Java 1.3.x is installed on your system and is pointed to in your PATH statement.
- For systems other than AIX, make sure Java 1.4.2 or above is installed on your system and is pointed to in your PATH statement. To download JDK/ JRE visit the following sites:
  - For SUN, Linux, Windows: the Java web site
  - For HP-UX: the Hewlett Packard web site

Dependencies

The Administration Console can run in any of three modes.

Authentication Plus Authorization Mode

If you want to run the Administration Console in full authentication plus Authorization mode, you must install both Symantec Product Authentication Service and Symantec Product Authorization Service. The console will recognize whether Authorization is installed on the system and will display the screens and functions accordingly, as either active or inactive..

Authentication-Only Mode

If you want to run the console in Authentication-Only mode, you only need to install Symantec Product Authentication Service.

Client-Only Mode: Credentials Administration Only

If you only want to view and use the Credentials area of the Administration Console, you only need to install the authentication client.

Understanding Console Security

The Administration Console has two parts. You can think of them as two separate modes:

- Authentication only
- Authentication plus authorization

Installing the authorization client enables use of the authorization part of the console.

Since it is not appropriate for all users to be able to work with the Administrative Console, certain security measures have been established.

Authentication Console Security

To administer the Symantec Product Authentication Service, you must login to the actual machine where the Authentication Service is installed and perform the administration tasks from that machine. That is, in the current release you cannot administer the Authentication Service remotely. Furthermore, you must login as administrator (on Windows) or root (on UNIX) to perform administration of the broker.

Authorization Console Security

See the Symantec Product Authorization Service *Administrator's Guide*.

Starting the Administration Console

You can use either the CLI or the Administration Console to manage Symantec Product Authentication Service.

To use the console

1. Log on with administrative rights (on Windows) or as root (on UNIX) on the actual machine where the Authentication Service is installed.
   You will perform the administration tasks from that machine. In the current release you cannot administer the Authentication Service remotely.
2. Start the Administration Console as follows:
   - On Windows:
     - Modify your path statement, if necessary, so that it points to the `<authentication install directory>\bin` directory for authentication and the `<authorization install`

directory>\bin directory for authorization (if you have installed it).

- Run the Administration Console in either of the two following ways:
  From the **Start** menu or
  Run runvssatgui.bat from the <*authentication install directory*>\bin directory.

- On UNIX:
  - Modify your path statement, if necessary, so that it points to the <*authentication install directory*>/bin directory for authentication and the <*authorization install directory*>/bin directory for authorization (if you have installed it).
  - Run <authentication install directory>/bin/runvssatgui.sh

3 When the Console is started, set up a trust relationship with the broker that the Authorization Service trusts.

Note: This is a one time activity and need not be done every time you use the Administration Console.

---

a Select **Credentials** from the quick access panel, and select the Trust Relationships tab.

b Click **Establish Trust** on the toolbar and complete the dialog.
- Broker: the name of the machine that holds the Authorization Service
- Port: Default is 2821

c Click **OK**.
The message Established Trust with Broker should be displayed.

If You Have Trouble Authenticating

If you receive an error message, check the following:

- Make sure the service is running.
- Make sure that you have set up trust with the authentication broker.
- Make sure that you are logging in as the local administrator.

- If you are on a UNIX platform, make sure you entered the domain name. For UNIX, this is a required field.

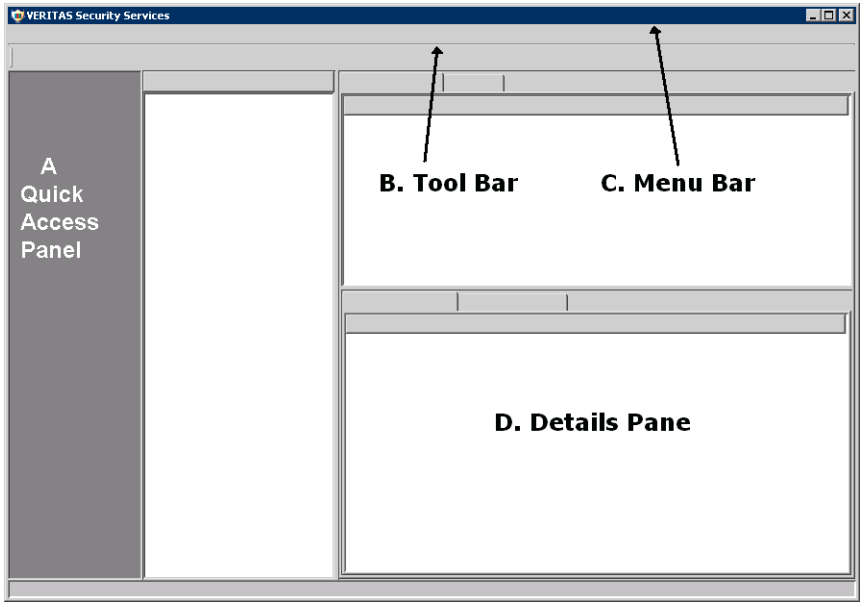
Appearance of the Administration Console

A Symantec application client will get its Administration Console from the host on which the Symantec application service is installed. Therefore, the appearance of the Administration Console will vary, depending upon what components are installed on the host.

- If only the client is installed, the only portion of the console that will be available to the client is the portion used for managing credentials.
- If the authentication service is installed, but authorization is not, all authentication functions will be available, but authorization screens and functions will be inactive.
- If both authentication and authorization are installed, all functions will be available.
- Appearance may also depend upon which resource management application you are using.

In general, the screens will resemble the one below. The number of tab screens on the right will vary, as will the number of panes in the right window:

Figure 5-9 Administration Console



The major areas of the Administration Console are:

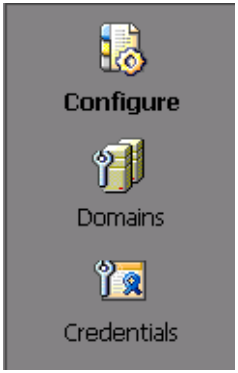
- Quick Access Panel (A): An area containing buttons for different categories of operation. Selecting different options in the quick access panel will cause different tab screens and windows to be displayed for you to work with.
- Tool Bar (B): Offers a shortcut way to take action on the commonly used operations of the console by clicking the toolbar buttons with the mouse.
- Menu Bar (C): Used for accessing the popup menus for the various operations that can be performed through the console.
- Details Pane (D): Displays information and functions corresponding to the option that was selected in the quick access panel.

Quick Access Panel

The quick access panel contains different options, depending upon whether you are working in Authentication-Only mode or in Authentication plus Authorization mode.

In Authentication-Only mode, the Quick Access Panel looks like this:

Figure 5-10 Quick Access Panel in Authentication-Only Mode



Selecting an option in the quick access panel will cause appropriate tab screens and windows to be displayed for you to work with.

Tool Bar

The toolbar lets you click buttons to perform a number of functions, which differ depending upon the selection you have made from the Quick Access Panel.

Menu Bar

The menu bar is used for accessing the popup menus for the various operations that can be performed through the Administration Console. Inactive menus are greyed out.

Details Pane

The details pane displays information and functions corresponding to the option that was selected in the quick access panel. The details pane may be comprised of one or more tab screens and one or more screens.

Using the Administration Console

This section of the documentation presents examples of ways to use the Administration Console.

Seeing What you Have

After you install Symantec Product Authentication Service, you may want to begin by examining certain basic information about your installation. For example, you can:

- Show location of private domain repository
- Show existing brokers
- Show existing credentials
- Show domain information
  - List private domains
  - Show information about a particular private domain
  - List existing principals in a private domain
  - Show information about a specific principal
  - Show domains supported by a plugin
  - Show the expiry intervals and available domains for a plugin

Show Location of Private Domain Repository

An authentication private domain repository (PDR) is a store of one or more authentication private domains. The Authentication Broker loads this repository, and principals are checked against it in order to be validated.

To show the location of the private domain repository

1 Select **Configure** from the Quick Access Panel.

2 Select either the Authentication Broker or the Root Broker tab.

CLI Equivalent

vssat showpdr

Show Existing Brokers

To show the existing brokers

1 Select **Credentials** from the Quick Access Panel.

2 Select the Domain-Broker Mapping tab.

CLI Equivalent

vssat showbrokers

Working with Security Levels

The security administrator can configure one of the following levels of security for distributing root certificates:

- High security (2): If a previously untrusted root is acquired from the peer (that is, if no certificate with the same signature exists in our trust store), the user will be prompted to verify the hash.

Note: The Root Hash is the thumbprint of the Root Broker's credential, it takes the form of a binary file and uniquely identifies a Root Broker. The Root Hash is used for establishing trust relationships. It can be found in `/opt/VRTSat/bin` for UNIX and `<InstallDir>\Authentication\bin` on Windows.

- Medium security (1): The first authentication broker will be trusted without prompting. Any attempts to trust subsequent authentication brokers will cause the user to be prompted for a hash verification before the certificate is added to the trusted store.
- Low security (0): The authentication broker certificate is always trusted without any prompting.

Show the Security Level (CLI Only)

At present, it is not possible to perform this function through the Administration Console.

The basic CLI command for showing the security level is as follows:

vssat showsecuritylevel

The display will indicate whether the security level is low (0), medium (1), or high (2).

Set the Security Level

At present, in the Administration Console, you can set a security level only when establishing a new trust relationship.

To set the security level for a new trust

1. Select **Credentials** from the Quick Access Panel.
2. Select the Trust Relationship tab.
3. Open the Authentication menu and select **Trust Relation** --> **Establish Trust**.
4. Provide the information requested in the dialog box and click **OK**:

- **Broker**: Specifies the broker to be contacted for downloading the root certificate. This can be either a Root Broker or an Authentication Broker location.
  - Root broker: The selfsigned root certificate is downloaded. If the root trusts other roots (i.e., has additional roots in its trust store), these are also downloaded.
  - Authentication broker: The root certificate in the chain (the root which signed the Authentication Broker certificate) is downloaded. If the Authentication Broker trusts other roots (i.e., has additional roots in its trust store), these are also downloaded.
- **Port Number**: The port of the broker that is to be contacted.
- **Security Type**: In this dialog, you can select either high, medium, or low as security level.
  - Low: The downloaded root certificates are added to local Trust store without verification.
  - High: The downloaded root certificates are added to local Trust store after hash verification. A list of hashes can be supplied in root_trust_handlinginfo structure. A trust establishment attempt in high security level would fail in the absence of any input hashes. If multiple roots are downloaded, hash verification is required for each.

    If the trust establishment attempt is made in high level and no matching hash is found, the handle to the root certificate is returned as root_credential out param. Applications can choose to provide an option of manual hash comparison or certificate information display on errors such as above.
  - Medium: The first trust establishment attempt proceeds in low security level. All subsequent ones proceed in high level.
- **Trust Type**: If you select **High** or **Medium**, with trust type as **Normal Trust**, the Administration Console will display the certificate details along with the hash. You will be asked whether to trust that broker. If you answer **Yes**, the root certificate of that broker will be added.

CLI Equivalent

vssat setsecuritylevel

Working with Credentials

A product credential is an entitlement to be recognized as a valid identity. A product credential requires both (1) the principal's private key and (2) a X.509v3 certificate with special extensions, produced and signed by the authentication

broker or root broker, to bind the principal's name to the public key. The product credential provides single-sign-on capability for all Symantec applications that use the Symantec Product Authentication Service and that choose to participate in the Symantec single sign-on session.

In working with credentials, you might want to perform the following tasks:

- Show where credentials are stored
- Set the directory where credentials are stored
- See trust relationships
- Show credentials that exist
- Request a certificate
- Delete a credential
- See where trust information is stored
- Establish a trust
- Remove a trust

Show Where Credentials are Stored

To see the directory where credentials are stored

1. Select **Credentials** from the Quick Access Panel.
2. Select the General tab.

CLI Equivalent

vssat showcredstore

Set Directory Where Credentials are Stored

To set the directory where credentials are stored

1. Select **Credentials** from the Quick Access Panel.
2. Select the General tab.
3. Click **Edit** to browse for the directory you want to select, and click **Choose**.

CLI Equivalent

vssat setcredstore

Show Credentials that Exist

To show existing credentials

1. Select **Credentials** from the Quick Access Panel.
2. Select the Personal tab.
3. Highlight the principal for which you want to see the credential, and either double-click or click **View Certificate**.

Note: You can also view certificates from the Trust Relationship tab.

CLI Equivalent

vssat showcred

Request a Certificate

To request that a certificate be added to the local store

1. Select **Credentials** from the Quick Access Panel.
2. Select the Life Cycle Management tab.
3. Enter the required information into the fields, and click **Submit**.
   If the user name includes a space, enclose the entire name in quotation marks.

CLI Equivalent

vssat authenticate

Delete a Credential

To delete a credential from the local store

1. Select **Credentials** from the Quick Access Panel.
2. Select the Life Cycle Management tab, and expand the **Destroy Credentials** area.
3. Provide the principal name, domain name, and domain type.
4. Indicate the broker that issued the credential.
5. Click **Delete Now**.

CLI Equivalent

vssat deletecred

Set up a Trust Relationship

To set up a trust relationship

1. Select **Credentials** from the Quick Access Panel.
2. Select the Trust Relationship tab.
3. Open the Authentication menu and select **Trust Relation** --> **Establish Trust**.
4. Provide the name and port number of the broker.
5. Indicate the level of security to use for this trust.

CLI Equivalent

vssat setuptrust

Remove a Trust Relationship

To remove a trust

1. Select **Credentials** from the Quick Access Panel.
2. Select the Trust Relationship tab.
3. Highlight the relationship to be deleted.
4. Open the Authentication menu and select **Trust Relation** --> **Remove Trust**.
5. Confirm the deletion by clicking **OK**.

CLI Equivalent

vssat `removeldapdomain`

Working with Domain-Broker Mapping

Domain-Broker mapping provides the set of information telling which authentication broker should be approached, for each domain, when attempting to authenticate.

Domain-Broker maps can be set up as something shared by all users on the machine or for a particular user. If you need to create a domain broker map for all users on the machine that will use Authentication, then use the global scope to create the mapping. This is helpful when a security-aware administrator wants to create a map for all users on that machine. The global maps can be overridden by the local maps, which are for the logged in user only.

In working with domain-broker mappings, you may want to perform the following tasks:

- See existing broker mappings

- Add a broker mapping
- Delete a broker mapping

Show Domain-Broker Mapping

To see the domain-broker mappings

1. Select **Credentials** from the Quick Access Panel.
2. Select the Domain-Broker mapping tab.

The list will indicate which broker is to be approached for authentication by principals in a particular domain of a particular type. Scope will be listed as either local or global. "Local" means that it pertains only to the current user; "global" means that it is system wide.

CLI Equivalent

vssat showallbrokerdomains

Add a Domain-Broker Mapping

To add a domain-broker mapping

1. Select **Credentials** from the Quick Access Panel.
2. Select the Domain-Broker mapping tab.
3. Open the Authentication menu and select **Domain Broker Mapping** --> **Add Mapping**.
4. In the dialog box, indicate the broker that is to be approached for authentication -- that is, the broker to which the domain is to be mapped.
5. Provide the name and type of the domain that is to use this broker.
6. Indicate whether you want to add this mapping only for the local user or to the global registry.
   Global scope is helpful when a security-aware administrator wants to create a map for all users on a particular machine. The global maps can be overridden by the local maps, which are for the logged in user only.
7. When you have completed the dialog, click **Add**.

CLI Equivalent

vssat addbrokerdomain

Delete a Domain-Broker Mapping

To delete a domain-broker mapping

1. Select **Credentials** from the Quick Access Panel.
2. Select the Domain-Broker mapping tab.
3. Highlight the mapping to be deleted.
4. Open the Authentication menu and select **Domain Broker Mapping** --> **Delete Mapping**.
5. Confirm the deletion by clicking **OK**.

CLI Equivalent

vssat deletebrokerdomain

Working with Private Domains

A private domain, also called an authentication private domain, is a specialized authentication domain used to hold identities and password hashes for authentication principals unique to, and managed by, Symantec products for which customers do not want to reuse an existing identity in another domain. Authentication private domains can be used to hold identities of Symantec point products, such as SAN Point Control and Volume Manager.

In working with private domains, you may want to perform the following tasks:

- List private domains
- Create or delete a private domain
- Show information about a specific private domain
- Set certain attributes of private domains
- Add or delete a principal in a domain
- Update information about a principal in a private domain

List Private Domains

To see a list of private domains

1. Select **Domains** from the Quick Access Panel.
2. Select the Private Domains tab.

Private domains are listed, along with each domain's default expiry policy and the expiry policies for users and for services.

CLI Equivalent

vssat listldapdomains

Create a Private Domain

To create a private domain

1. Select **Domains** from the Quick Access Panel.
2. Select the Private Domains tab.
3. Open the Authentication menu and select **Domains** --> **Create Domain**.
4. Type the name of the new domain and its expiry interval, and click **Create Domain**.

CLI Equivalent

vssat createpd

Show Information about a Particular Private Domain

To show information about a particular private domain

1. Select **Domains** from the Quick Access Panel.
2. Select the Private Domains tab.
3. Highlight the domain about which you want to see information.

The upper pane will display information about the expiry policies. The lower pane will list principals in the highlighted domain.

CLI Equivalents

To show domain and expiry:

vssat showpd

To see information about the principals in the domain:

vssat listpdprincipals

Set Private Domain Attributes

At present, the only attribute you can set is the expiry time.

To set private domain attributes

1. Select **Domains** from the Quick Access Panel.
2. Select the Private Domains tab.

Private domains are listed, along with each domain's default expiry policy and the expiry policies for users and for services.

3 Highlight the domain whose expiry interval you want to change.

4 Open the Authentication menu and select **Domains** --> **Update Domain**.

5 Enter the new expiry interval in seconds, and click **Update Domain**.

The basic CLI command for setting certain attributes of a specific private domain is as follows:

vssat setpdr

Delete a Private Domain

To delete a particular private domain

1 Select **Domains** from the Quick Access Panel.

2 Select the Private Domains tab.

3 Highlight the domain to be deleted.

4 Open the Authentication menu and select **Domains** --> **Delete Domain**.

5 Confirm the deletion by clicking **OK**.

CLI Equivalent

vssat deletepd

List Existing Principals in a Private Domain

To see a list of existing principals

1 Select **Domains** from the Quick Access Panel.

2 Select the Private Domains tab.

3 Highlight a domain.

The lower pane will display information about the principals in the highlighted domain.

CLI Equivalent

vssat listpdprincipals

Add a Principal in a Private Domain

To add a principal in a private domain

1 Select **Domains** from the Quick Access Panel.

2 Select the Private Domains tab.

3 Highlight the domain to which you want to add the principal.

4 Open the Authentication menu and select **Principal** --> **Add Principal**.

5 Provide a name and password for the new principal.

6 In the Principal Type field, indicate which expiry interval will be used: the one established for users, the one established for services, or the default interval.

7 Select **Can Proxy** if you want this principal to be able to act as a proxy for another principal.
This case is specifically useful for web server credentials where the web server proxies to its backend services for the end user who is using web browser.

8 Select **Can Accept Proxy** if you want this principal to be able to accept proxies.
This case is specifically useful for the back end services of a web-server. The web server, before handing out the product web credential of end user, checks if the receiving peer can accept a proxy.

9 Click **Add Principal**.

CLI Equivalent
vssat addprpl

Show Information about a Specific Principal

To see information about a specific existing principal

1 Select **Domains** from the Quick Access Panel.

2 Select the Private Domains tab.

3 Highlight the domain in which the principal resides.

The lower pane will display information about the principals.

CLI Equivalent
vssat showprpl

Update a Principal in a Private Domain

To update a principal in a private domain

1 Select **Domains** from the Quick Access Panel.

2 Select the Private Domains tab.

3 Highlight the domain in which the principal resides, and select the principal.

4 Highlight the principal to be updated.

5 Open the Authentication menu and select **Principal** --> **Update Principal**.

6 If you wish, change the Principal Type, which indicates which expiry interval will be used: the one established for users, the one established for services, or the default interval.

7 Select **Can Proxy** if you want this principal to be able to act as a proxy for another principal.
This case is specifically useful for web server credentials where the web server proxies to its backend services for the end user who is using web browser.

8 Select **Can Accept Proxy** if you want this principal to be able to accept proxies.
This case is specifically useful for the back end services of a web-server. The web server, before handing out the product web credential of end user, checks if the receiving peer can accept a proxy.

9 Click **Save**.

CLI Equivalent

vssat updateplugin

Change the Password for a Principal

To change the password for a principal

1 Select **Domains** from the Quick Access Panel.

2 Select the Private Domains tab.

3 Highlight the domain in which the principal resides.

4 Highlight the principal whose password you want to change.

5 Open the Authentication menu and select **Principal** --> **Change Password**.

6 Type the old password.

7 Type and retype the new password, and click **Change Password**.

CLI Equivalent

vssat changepasswd

Delete a Principal

To delete a principal from a private domain

1. Select **Domains** from the Quick Access Panel.
2. Select the Private Domains tab.
3. Highlight the domain from which you want to delete the principal.
4. Highlight the principal to be deleted.
5. Open the Authentication menu and select **Principal** --> **Delete Principal**.
6. Confirm the deletion by clicking **Yes**.

Required Steps after Deleting Principal

When you delete a principal, we recommend doing one of the following, as well as deleting the principal:

- Either delete all the ACLs that include the principal who has been deleted
- Or put the principal into the "disabled principals" list

You might prefer to put the principal into the "disabled principals" list if, for example, the user is away temporarily and does not want his or her identity to be used for anything in their absence. When the user returns, all the permissions can be easily restored by removing the principal from the "disabled principals" list.

If the user has left the company, then removing all ACLs that include this principal makes sense.

CLI Equivalent

vssat deleteprpl

Working with Plugins

An authentication plugin is a component used by the authentication broker to validate identities within a particular domain. An authentication plugin exists for each supported authentication mechanism. For example, one plugin can validate NIS identity and password combinations against an NIS database, while another uses a Kerberos ticket to authenticate the principal.

In working with plugins, you might want to perform the following tasks:

- Show domains supported by a plugin
- Show existing expiry intervals for a plugin
- Show basic information about a plugin

Note: One important task you may want to perform related to plugins is to enable the LDAP plugin for Symantec Product Authentication Service. This task is not performed through the GUI. See "Basic Steps for Enabling LDAP" on page 41. For a deeper look at this special plugin, see Appendix B, "LDAP Plugin Details" on page 167.

Show Domains Supported by a Plugin

To see a list of domains supported by a plugin

1. Select **Domains** from the Quick Access Panel.
2. Select the Plugins tab.

CLI Equivalent

vssat showdomains

Show the Expiry Intervals and Available Domains for a Plugin

To see expiry intervals and available domains for a specific plugin

1. Select **Domains** from the Quick Access Panel.
2. Select the Plugins tab.
3. Highlight the plugin.

The default, user, and service intervals will be displayed in the top pane.Available domains for the plugin will be displayed in the bottom pane.

CLI Equivalent

vssat showexpiryintervals

Working with Principals

An authentication principal is a user, computer, or process such as a command line interface (CLI) or service that has the ability to authenticate to Symantec Product Authentication Service with a unique identity.

Since every principal must be a member of a domain, the tasks related to principals are a subset of those related to domains. Therefore, see "Working with Private Domains" for details.

In working with principals, you might want to perform the following tasks:

- List Existing Principals in a Private Domain
- Add a Principal in a Private Domain

- Show Information about a Specific Principal
- Update a Principal in a Private Domain
- Change the Password for a Principal
- Delete a Principal

Console Objects: Symantec Product Authentication Service

Here follows the set of Administration Console objects for Symantec Product Authentication Service.

Configure Area

Many configuration decisions are made at the time of installation. If you select the **Configure** option in the Quick Access Panel, you can view many of these options and change some of them.

The **Configure** area will display some or all of the following tab screens, depending upon your configuration:

- Authentication Broker
- Root Authentication Broker
- Authorization Service

Authentication Broker Tab

Use this screen to see the identity of the machine on which the Authentication Broker is installed, as well as location of the authentication private domain repository.

Authentication Broker

The Authentication Broker is the component that serves, one level beneath the Root Broker, as an intermediate registration authority and a certification authority. The Authentication Broker can authenticate clients, such as users or services, and grant them a certificate that will become part of the product credential. An Authentication Broker cannot, however, authenticate other brokers. That task must be performed by the Root Broker.

Authentication Private Domain

An authentication Private Domain is a specialized authentication domain used to hold identities and password hashes for authentication principals unique to,

and managed by, Symantec products for which customers do not want to reuse an existing identity in another domain. Authentication private domains can be used to hold identities of Symantec point products, such as SAN Point Control and Volume Manager.

Authentication Private Domain Repository (PDR)

An authentication private domain repository is a store of one or more authentication private domains. The Authentication Broker loads this repository, and principals are checked against it in order to be validated.

Root Broker Tab

The Root Broker is the first, most trusted, Authentication Broker. As such, it has a self-signed certificate. The Root Broker has a single private domain that holds only the names of brokers that shall be considered valid. The name of the Root Broker itself is stored as the fully qualified domain name.

Configuration tasks for the Root Broker are carried out at installation. On the host where the Root Broker is installed, you can use the Root Broker tab screen to see the identity of the machine on which the Root Broker is installed, as well as location of the root's private domain repository.

The **Hash** field displays the cryptographic hash (or thumbprint) of the root certificate. This hash can be used in setting up a trust relationship to verify the root certificate. This hash is ***not*** secret and can be posted on trusted internal infrastructure (like Intranet or Directory) from where it can be easily downloaded or distributed in order to help root certificate verification.

Note: Modifications of the hash are not allowed through the Administration Console.

Domains Area

An authentication domain defines a set of identities for authentication principals. In addition, it can provide authorization information related to the principals in the set, such as group membership. Some examples are names in the NIS+, NTML, or Active Directory domains.

Some authentication domains are public. A special type of authentication domain is the authentication private domain, which is specific to Symantec.

When you select the **Domains** option from the Quick Access Panel, the following tab screens become available:

- Private Domain

- Plugins

Private Domain Tab

Use this screen to perform any of the following management tasks:

- See details about the principals belonging to private domains
- Create or delete private domains
- Create or remove principals of private domains
- Reset passwords

Highlight a private domain in the upper pane to see details about its principals in the lower pane.

To create, delete, or update a domain, open the Authentication menu and select **Domains** and the desired option.

To add, delete, update, or change the password for a principal in a domain, open the Authentication menu and select **Principals** and the desired option.

Nature of Private Domains

An authentication private domain is a specialized domain used to hold identities and password hashes for authentication principals that are unique to, and managed by, Symantec products for which customers do not want to reuse an existing identity in another domain.

The Root Broker has a single private domain that holds only the names of brokers that shall be considered valid. The name of the Root Broker itself is stored there as the fully qualified domain name.

Other Authentication Brokers may have 0 or more private domains whose members may be of various types -- users, services, command line interfaces, etc. -- although their types are not distinguished in the private domains.

Authentication private domains can be used to hold identities of Symantec point products, such as SAN Point Control and Volume Manager.

Create Domain Dialog

Use this dialog to add an authentication private domain to a broker. Indicate the domain name, and provide an expiry interval in seconds. Then click **Create Domain**.

An authentication private domain is a specialized domain used to hold identities and password hashes for authentication principals that are unique to, and managed by, Symantec products for which customers do not want to reuse an existing identity in another domain.

The Root Broker has a single private domain that holds only the names of brokers -- including itself -- that shall be considered valid. Therefore, you cannot add a domain to the Root Broker.

Other Authentication Brokers may have 0 or more private domains whose members may be of various types -- users, services, command line interfaces, etc. -- although their types are not distinguished in the private domains.

Authentication private domains can be used to hold identities of Symantec point products, such as SAN Point Control and Volume Manager.

Delete Domain Dialog

Click **Yes** on this dialog to confirm that you want to delete the highlighted item.

Update Domain Dialog

Use this dialog to modify the expiry interval for a domain.

The expiration interval is expressed as the number of seconds. A zero means the next level of expiry policy is used, i.e., the domain expiry policy is used. If that also is zero, then plugin-wide the expiry policy is used. If that also is zero, then software specifies some defaults.

Add Principal Dialog

Use this dialog to add a principal to a selected authentication private domain.

To add a principal

1. Make sure that you have selected the domain to which you want to add the principal. If you have not, click **Cancel** and highlight the right domain.
2. Provide a name and password for the new principal.
3. In the Principal Type field, indicate which expiry interval will be used: the one established for users, the one established for services, or the default interval.
4. Select **Can Proxy** if you want this principal to be able to act as a proxy for another principal.

   This case is specifically useful for web server credentials where the web server proxies to its backend services for the end user who is using web browser.
5. Select **Can Accept Proxy** if you want this principal to be able to accept proxies.

This case is specifically useful for the back end services of a web-server. The web server, before handing out the product web credential of end user, checks if the receiving peer can accept a proxy.

6 Click **Add Principal**.

Delete Principal Dialog

Click **Yes** on this dialog to confirm that you want to delete the highlighted item.

Change Password Dialog

Use this dialog to change the password for the highlighted principal.

To reset the password

1 Type the old password.

2 Type the new password.

3 Confirm by retyping.

4 Click **Change Password**.

Update Principal Dialog

Use this dialog to modify information about a principal.

Expiration policy may be set for users, for services, or for the default value. Select from the dropdown list,

The expiration interval is expressed as the number of seconds. A zero means the next level of expiry policy is used, i.e., the domain expiry policy is used. If that also is zero, then plugin-wide the expiry policy is used. If that also is zero, then software specifies some defaults.

If you select **Can Proxy,** this principal will be able to act as a proxy for another principal. This case is specifically useful for web server credentials where the web server proxies to its backend services for the end user who is using web browser.

If you select **Can Accept Proxy,** this principal will be able to accept proxies. This case is specifically useful for the back end services of a web-server. The web server, before handing out the product web credential of end user, checks if the receiving peer can accept a proxy.

Plugins Tab

An authentication mechanism is the way authentication is conducted for users in the name-space defined by a domain. An authentication mechanism

encapsulates all the details of the authentication algorithm, including APIs, protocols, token formats, token contents semantics and database objects formats. Not all the ingredients are relevant in all mechanisms.

For each supported authentication mechanism, there is a plugin, a component used by the Authentication Broker to validate identities within a particular domain. For example, one plugin can validate NIS identity and password combinations against an NIS database, while another uses a Kerberos ticket to authenticate the principal.

Use the Plugins tab screen to see details about existing plugins or to specify expiry policy.

The Authentication Service will install all the supported plugins for the particular OS that you are using. You cannot add or delete a plugin in this release of Symantec Product Authentication Service

Expiry Intervals

For plugins, expiry policies are set on a per-plugin basis, rather than on a per-principal basis. The expiry policy will tell the broker (whether Root Broker or Authentication Broker) how much expiry time it should set on a credential that it generates.

Policy may be set for users, for services, or for the default value. Select an interval from the dropdown list, or select **User defined** and set an interval in hours, days or months.

If an authenticated principal is a service (or CLI), the service principal expiry interval is applied. If the authenticated principal is a user, then the user expiry interval is applied. If it cannot be determined whether the principal is a service or a user, the default expiry interval is used.

Expiry Interval Dialog

Use this dialog to change an expiry interval for the selected plugin.

Expiry policies, set on a per-plugin basis, tell the broker how much expiry time it should set on a credential that it generates.

Indicate whether you want to set policy for users, for services, or for the default value, and enter an interval in seconds. Then click **Change Expiry**.

Credentials Area

When you select the **Credentials** option from the Quick Access Panel, you can manage the credentials manager.

The Credential Manager Defined

The credential manager is a component that manages credentials acquired during Authentication. The credentials that are acquired are cached in a local credential store.

Tip: This means that once an application service or client has authenticated with its Authentication Broker, they are no longer dependent upon that Broker in order to be on line. They can run even if the Broker is not running.

Available Tabs in the Credentials Area

In the Credentials area, the following tab screens are available:

- General
- Trust Relationships
- Life Cycle Management
- Domain-Broker Mapping
- Personal

Note: The credential manager handles this issue because, behind the scenes, an API subroutine acquires the credential when given user information made up of Identity, DomainName, DomainType, and Password. With this API, the credential manager figures out which broker to approach, based on the domain information.

Credentials: General Tab

A product credential (called simply *credential* in the Administration Console) is an entitlement to be recognized as a valid identity. A product credential requires both of the following:

- The principal's private key
- A X.509v3 certificate with special extensions, produced and signed by the Authentication Broker or root broker, to bind the principal's name to the public key.

The product credential provides single-sign-on capability for all Symantec applications that use the Symantec Product Authentication Service and that choose to participate in the Symantec single sign-on session.

Use the General tab to see where the Root stores credentials.

Credentials: Trust Relationship Tab

A certification authority is a trusted third party responsible for issuing, managing, and revoking certificates that vouch for the identities of the certificate holders. Certificates vouching for the authenticity of principals are signed by a hierarchy of certification authorities.

- The root certification authority (on the Root Broker) is the entity at the top of the hierarchy and is therefore the most trusted certification authority. Its certificate, vouching for itself, is self-signed and is called the root certificate.
- On the next level of the trust hierarchy is the Authentication Broker. It authenticates security principals and generates certificates signed by the Authentication Broker. That is, the Authentication Broker acts as an intermediate certification authority, saying, in effect, "The Root Broker trusts me. And I trust you."

Use the trust relationships tab screen to see information about trust relationships that have been established:

- The **Issued To User...** area displays the trusted entity in the format `user[domaintype:domainname]`, for example
  `broker[vx:root@something.something.com]`
- The **Issued By Broker...** area displays the name of the issuing entity, using the same format as used for the trusted entity:
- The **Expiration Date** area shows when the certificate expires.

View Certificate Dialog

Use this dialog to see details about the selected certificate.

A product credential (called simply *credential* in the Administration Console) is an entitlement to be recognized as a valid identity. A product credential requires both of the following:

- The principal's private key
- A X.509v3 certificate with special extensions, produced and signed by the Authentication Broker or Root Broker, to bind the principal's name to the public key.

The product credential provides single-sign-on capability for all Symantec applications that use the Symantec Product Authentication Service and that choose to participate in the Symantec single sign-on session.

Establish Trust Dialog

A certification authority is a trusted third party responsible for issuing, managing, and revoking certificates that vouch for the identities of the

certificate holders. Certificates vouching for the authenticity of principals are signed by a hierarchy of certification authorities.

The root certification authority (on the Root Broker) is the entity at the top of the hierarchy and is therefore the most trusted certification authority. Its certificate, vouching for itself, is self-signed and is called the root certificate. On the next level of the trust hierarchy is the Authentication Broker. It authenticates security principals and generates certificates signed by the Authentication Broker. That is, the Authentication Broker acts as an intermediate certification authority, saying, in effect, “The Root Broker trusts me. And I trust you.”

Broker: Specifies the broker to be contacted for downloading the root certificate. This can be either a Root Broker or an Authentication Broker location.

- Root broker: The self-signed root certificate is downloaded. If the root trusts other roots (i.e., has additional roots in its trust store), these are also downloaded.
- Authentication broker: The root certificate in the chain (the root which signed the Authentication Broker certificate) is downloaded. If the Authentication Broker trusts other roots (i.e., has additional roots in its trust store), these are also downloaded.

Port Number: The port of the broker that is to be contacted.

Security Type: In this dialog, you can select either high, medium, or low as security level.

- Low: The downloaded root certificates are added to local Trust store without verification.
- Medium: The first trust establishment attempt proceeds in low security level. All subsequent ones proceed in high level.
- High: The downloaded root certificates are added to local Trust store after hash verification. A list of hashes can be supplied in root_trust_handlinginfo structure. A trust establishment attempt in high security level would fail in absence of any input hashes. If multiple roots are downloaded, hash verification is required for each.
  If the trust establishment attempt is made in high level and no matching hash is found, the handle to the root certificate is returned as root_credential out parameter. Applications can choose to provide an option of manual hash comparison or certificate information display on errors such as above.

Trust Type: If you select **High** or **Medium**, with trust type as **Normal Trust**, the Administration Console will display the certificate details along with the hash. You will be asked whether to trust that broker. If you answer **Yes**, the root certificate of that broker will be added.

Remove Trust Dialog

Click **Yes** on this dialog to confirm that you want to delete the highlighted item.

Credentials: Life Cycle Management Tab

A product credential (called simply *credential* in the Administration Console) is an entitlement to be recognized as a valid identity. A product credential requires both of the following:

- The principal's private key
- A X.509v3 certificate with special extensions, produced and signed by the Authentication Broker or Root Broker, to bind the principal's name to the public key.

The product credential provides single-sign-on capability for all Symantec applications that use the Symantec Product Authentication Service and that choose to participate in the Symantec single sign-on session.

A product credential can be seen as having three stages of life:

- Beginning: Credential is added to store
- Middle: Credential is used
- End: Credential expires or is destroyed

On this tab screen, expand either **Manual Request for Credential** or **Destroy Credentials** to request a new credential or to destroy an existing credential.

Requesting a Credential

Expand the **Manual Request for Credential** area. Complete the requested information and click **Submit**.

- **Name:** The name for which the credential is being requested. This will be the currently logged-in user if no user is provided. If the user name includes a space, enclose the entire name in quotations marks.
- **Password:** The password of the principal for which the credential is being requested. If none is provided, the password for the currently logged-in user will be used.
- **Domain Name:** Optional, the domain name of the principal for which a credential is being requested.
- **Domain Type:** The domain type of the principal for which a credential is being requested.
- Click **Broker** and complete the dialog if you want to select the broker who should issue the certificate.

Destroying a Credential

Expand the **Destroy Credentials** area. Complete the requested information and either click **Delete Now**.

- **Name:** The name for which the credential is being destroyed. This will be the currently logged-in user if no user is provided.
- **Domain Name:** Optional, the domain name of the principal for which a credential is being destroyed.
- **Domain Type:** The domain type of the principal for which a credential is being destroyed.

Issued By Broker: The broker who issued the certificate that is to be destroyed.

To destroy credentials that have expired, click the **Destroy Expired Credential** area and click **Delete Now**.

Broker Selection Dialog

Use this dialog provide the name and port number for the broker from which you are requesting a credential.

Credentials: Domain-Broker Mapping Tab

The security administrator can choose which Authentication Broker the client should contact for each domain from which it wants to authenticate. For example:

- If I am a nis+ client, use Authentication Broker A.
- If I am a unixpwd client, use Authentication Broker B.

Domain-Broker mapping is the set of information telling which Authentication Broker to approach, for each domain, when attempting to authenticate.

Information on Domain-Broker mapping is stored in either the global configuration or the local configuration. If you need to create a domain broker map for all users on the machine that will use Authentication, then use the global scope to create the mapping. This is helpful when a security-aware administrator wants to create a map for all users on that machine. The global maps can be overridden by the local maps, which are for the logged in user only.

When an Authentication Broker needs to be contacted for a domain, first local entries are searched, and then global.

Add Mapping Dialog

Use this dialog to create another mapping -- the set of information telling which Authentication Broker to approach when attempting to authenticate. The

information will be stored in either the global configuration or the local configuration.

When an Authentication Broker needs to be contacted for a domain, first local entries are searched, and then global.

Delete Mapping Dialog

Click **Yes** on this dialog to confirm that you want to delete the highlighted item.

Credentials: Personal Tab

Use the personals tab screen to see information about credentials:

- The **Issued To User...** area displays the trusted entity in the format `user[domaintype:domainname]`, for example
  `broker[vx:root@something.something.com]`
- The **Issued By Broker...** area displays the name of the issuing entity, using the same format as used for the trusted entity:
- The **Expiration Date** area shows when the certificate expires.

Chapter 6

The Command Line Interface

Symantec Product Authentication Service provides a command line interface (CLI) for performing a number of management tasks.

The present chapter covers the following topics:

Purpose of the CLI

The command line interface lets you perform tasks by typing commands, rather than by using a graphical user interface (GUI). Many functions can be performed through either interface; others are specific to one of the two.

Management Capabilities of the CLI

Management capabilities of the CLI are of three types:

- Broker Management: vxatd (For details, see “Using vxatd” on page 83.)
- Managing authentication: (For details, see “Using vssat” on page 87.)

Accessing the CLI

To access the CLI (command line interface) for authentication

1. Install the Authentication Service.
2. Navigate to the following directory (if you have installed in the default location):

   Windows: `C:\program files\veritas\security\authentication\bin`

   UNIX: C:`/opt/VRTSat/bin`

Command Usage

The convention used for the command line interface (CLI) is presented below. For the purposes of the examples, we use the `vssat` command.

Explanation of Abbreviations

In all commands, the abbreviation *AT* refers to Authentication; the abbreviation *AZ* refers to Authorization.

Commands With No Arguments

There are some instances when no arguments are required. Typically this is the case for options which show global information and don’t need qualifiers for filtering.

```
vssat commandoption
```

Commands With Optional or Required Arguments

If an argument is optional, it is placed within brackets **[ ]** as follows:

```
vssat commandoption [--optionalargument <data>]
```

If an argument is required, it uses the syntax below:

```
vssat commandoption --mandatoryargument <data>
```

Commands With Mutually Exclusive Arguments

If two possibilities are mutually exclusive, they are separated by a bar |, as below:

```
vssat commandoption --mutuallyexclusiveargument
        <choice1|choice2|choice3|>
```

If there is a situation in which each mutually-exclusive choice requires different data, the syntax below is used:

```
vssaz commandoption --mandatoryargument
        {Choice1,<Data1>|Choice2,<Data2a>,<Data2b>
        |Choice3,
        <Data3a>,<Data3b>|Choice4,<Data4>}
```

Using vxatd

Used with a number of arguments, the `vxatd` command lets you manage the broker in any of the following modes:

- Root mode
- Root + AB (Authentication Broker) mode
- Authentication Broker (AB) mode

To configure the broker, use the `vxatd` command, as described below.

Syntax to Use for Starting the Broker

Options for `vxatd` are entered as tokens followed by information pertaining to the option itself. The first time you run `vxatd`, use the following syntax:

Start in Root Broker Mode

Use the following syntax to start in Root mode:

```
vxatd -r [-d]
```

Start in Authentication Broker Mode

Use the following syntax to start in Authentication Broker mode:

```
vxatd -a -n <broker identity> -p <password> -x <domain type> -y
<domain name> -q <root broker name> -z <root broker port> -h
<hash file name>
```

Start in Root + AB Mode

Use the following syntax to start in Root plus Authentication Broker mode:

```
vxatd -a -r
```

After the first invocation, the command would read the appropriate configuration from the registry and pick up the existing key pairs and certificates. Therefore, after the first invocation, no arguments are mandatory.

Common Options

`-r`

Start the broker in root mode. This would be the first broker in a deployment.

Root key material is generated. A self-signed certificate is generated. Root private domain repository is initialized. Hash file is generated. The broker starts (Option “a” combined with option “r” would start the broker in Root + AB mode.)

`-a`

Start the authentication broker by authenticating with the root. Authentication broker key material is generated.

Authentication request is sent to the root. On successful authentication the authentication broker credential is installed in the store. Authentication broker private domain repository is initialized.The broker starts listening on the designated port.

Prerequisites: Authentication Broker’s authentication principal identity is added to the Root Broker’s private domain. Network connectivity to the root.

`-a -r`

Start the broker in root + authentication broker mode.

Root key material is generated. A Self Signed credential is generated. Root private domain repository is initialized. Hash file is generated. Authentication broker key material is generated. Authentication broker credential signed by the root is generated. Authentication broker private domain repository is initialized. The broker starts listening on the designated port.

`-d`

Start the broker in debug mode.

Additional Arguments for AB Start Up in AB Only Mode

The following arguments are required for starting up the authentication broker -- not the root broker -- the first time.

`-h <hash file name>`

Hash file name. This option forces the AB authentication process to happen in the High security level. The Root certificate would be added to the local store only if the hash from the file matches the downloaded certificate's hash. When configuring in AB Only mode, the root hash file argument is mandatory:

Prerequisites: Hash file transferred from the root to the authentication broker.

The Authentication Broker that was installed in AB Only mode can be started in high security mode only. The root certificate downloaded in the trust phase of broker installation is compared to the hash in the file. If there is a mismatch, the broker service terminates.

The root hash file is generated on the root broker host. Before using `vxatd` with the `-h` argument, this file must be transferred manually from the root to the Authentication Broker.

`-n <broker identity>`

Broker identity as configured in the root private domain.

`-o`

Key/credential generation mode for the broker. Keys/credentials are generated and stored in the appropriate directory structure. The broker invocation state (AB, Root, Root+AB) is stored. The broker starts listening and accepting requests only when invoked without this flag. Keys in the store are protected with user context. This provides another level of security (in form of programmatic work) for a hacker, beyond the file system security. For now the protection has no machine dependency.

`-p <password>`

Broker password as configured in the root private domain.

`-q <root broker name>`

Root broker name host name or IP.

`-x <domain type>`

Domain type for the authentication broker identity. This is "vx" (private domain).

`-y <domain name>`

Domain name for the authentication broker identity. This is the root private domain name in which the authentication broker identity is configured.

`-z <root broker port>`

Root broker port.

Windows Specific Arguments

`-u`
: Remove the program from the service manager.

`-k`
: Stop the service.

`-t`
: Set start up type for the existing service.

- Requests new credentials, making the broker part of specified domain
- Resets broker mode and identity information in the registry

Examples

Example 1

Syntax for starting vxatd on a child node if, for example on MachineA you have started `vxatd` as follows: `vxatd -r -a` . Use the following technique to start it on MachineB so that MachineB is a trusting child of MachineA.

Assume that you have Root + AB on MachineA.

Start an authentication broker on Machine B by following these steps

1. Adding an authentication principal in the private domain of Root (MachineA) for Authentication Broker on MachineB.
2. Running `vxatd` as follows:

```
vxatd -a -n<broker identity> -p <password> -x <domain type> -y
<domain name> -q <root broker name> -z <root broker port> -h
<root hash file name if working in high security level>
```

Note: If `-h` is not given, then the product is installed in low security mode, in which there is no authentication.

Example 2

For migrating root+AB to domain `root@somename.mycompany.com` (assuming that vx is a private domain), use the following syntax:

```
vxatd -j -n brokeridentity -p password -x vx -y
root@durga.veritas.com -h hashfile -qdurga -z2821
```

The result is an Authentication Broker in the `root@somename.mycompany.com` domain.

Using vssat

The convention used for the command line interface (CLI) is described below.

Optional arguments are placed within brackets **[ ]**. If an argument must have one, but only one, of a set of enumerated values, the data choices are separated by a "**|**". When the command arguments themselves are mutually exclusive, the arguments are grouped using **{** and **}**.

```
vssat SampleOption --Arg <data> --ArgWithChoiceOfData
<DataChoice1|DataChoice2|DataChoice3>
{ArgChoice1,<data>|ArgChoiceB,<data1>,<data2>| ArgChoice3,
<data1>,<data2>} [--OptionalArg <data>]
```

addbrokerdomain

Name

addbrokerdomain

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat addbrokerdomain --broker <host:port> --domain <type:name>
[--global]
```

Description

Use this command to add a mapping of a domain to a broker. Such a mapping indicates which broker should be approached when trying to authenticate to a particular domain. The broker to be added should be up and able to be pinged. If you delete an authentication principal, do one of the following to prevent unauthorized access:

- Either put the ID in the Disabled Principals List (and/or)
- Delete the ACLs associated with this identity, and take out the membership of the identity from the groups that this identity belongs to.

Arguments

This command allows the following arguments:

`--broker <host:port>`
: The host and port of the broker. The registered port for Authentication is 2821. If the broker has been configured with another port number, consult your security administrator for information.

`--domain <type:name>`
: Domain information of the domain for which broker name needs to be configured.

`--global`

Indication that the entry should be added to the global registry. If not specified, the entry is added to the local registry. Local registry settings only influence the current logged on OS principal, whereas global registry settings may apply to all OS principals of that host.

Examples

This command can be used as shown below:

```
vssat addbrokerdomain --broker MyHost:2821 --domain
nt:NewBrokerDomain
```

addldapdomain

Name

`addldapdomain`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat addldapdomain --domainname <domain name> --server_url
<server URL> --user_base_dn <user base DN> --group_base_dn
<group base DN> [--server_trusted_ca_file <trusted CA file
name>] [--schema_type <rfc2307 | msad>] | [--user_object_class
<user object class> --user_attribute <user attribute> --
user_gid_attribute <user GID attribute> --group_object_class
<group object class> --group_attribute <group attribute> --
group_gid_attribute <group GID attribute>] [--auth_type <FLAT |
BOB | FLAT SKIPNESTED | BOB SKIPNESTED>] [--admin_user <admin
user DN>] [--admin_user_password <admin user password>] [--
search_scope <SUB | ONE | BASE>]
```

Description

Use this command to add an LDAP domain to the authentication broker.

Users not familiar with how LDAP operates must work with their LDAP administrators to determine the following information:

- What type of LDAP directory the enterprise uses (i.e. Microsoft Active Directory, OpenLDAP, iPlanet, etc). The type of LDAP directory dictates the type of scheme to use.
- The URL to the LDAP directory. For example, “ldap://my_ldap_host.mydomain.myenterprise.com:389”, “ldaps://my_ssl_ldap_host.mydomain.myenterperise.com”, etc.

Note: An LDAP URL must start with “ldap://” for non-SSL, or “ldaps://” for SSL-enabled LDAP directory.

- The distinguished name (DN) of the users container. Normally, the users container is in one of the naming contexts. For most LDAP directories, you can use the ldapsearch utility, provided by the directory vendor, to find out the naming contexts. For example:

  ldapsearch --group_object_class -h <my host> `--server_url` base --auth_type "" namingContexts

  For Microsoft Active Directory, the users container resembles this example:

  cn=users,dc=<domain name>,dc=<enterprise name>,dc=com

- The distinguished name (DN) of the groups container. Normally, the groups container is in one of the naming contexts. For most LDAP directories, you can use the ldapsearch utility, provided by the directory vendor, to find out the naming contexts. For example:

  ldapsearch --group_object_class -h <my host> `--server_url` base --auth_type "" namingContexts

  For Microsoft Active Directory, the groups container looks like this example:

  cn=users,dc=<domain name>,dc=<enterprise name>,dc=com

- The schema to facilitate users and groups.

  If the enterprise has migrated their NIS data to the LDAP directory according to Request For Comments 2307, it must use the RFC 2307 schema. RFC 2307 uses the “posixAccount” objectclass to facilitate user objects. It uses the “posixGroup” objectclass to facilitate group objects.

  If the enterprise uses Microsoft Active Directory, it must use the Microsoft Active Directory schema. In this schema, the “user” objectclass facilitates both user and group objects.

  If the enterprise uses neither RFC 2307 nor Microsoft Active Directory, it must determine the following:

  - The LDAP objectclass to facilitate user objects.
  - The LDAP objectclass to facilitate group objects.
  - The user attribute in the user objectclass to facilitate user name/ID. We use the following rules to construct the DN to the user entry.

    <user attribute>=<user name>,<user container DN>

    For example, if the user attribute is configured to “cn” and users container DN is configured to “dc=mydomain,dc=myenterprise, dc=com” and the user name for the authenticate call is “jdoe”, the LDAP DN for “jdoe” is:

    cn=jdoe,dc=mydomain,dc=myenterprise,dc=com

  - The group identifier (GID) attribute in the user objectclass to identify the groups the given user belongs to.

- The group attribute in the group objectclass to facilitate group name. We use the following rules to construct the DN to the group entry.

  <group attribute>=<group name>,<group container DN>

  For example, if the group attribute is configured to “cn” and groups container DN is configured to “dc=mydomain,dc=myenterprise, dc=com” and the group name is “adm”, the LDAP DN for “adm” is:

  cn=adm,dc=mydomain,dc=myenterprise,dc=com
- The group ID attribute in the group objectclass to facilitate group ID for the given group.

Note: It is not mandatory to restart the broker after adding the LDAP domain. However, the ‘vssat authenticate’ command will not work for the newly added domain without the broker parameter until the broker is restarted.

Required arguments

`--domain DomainType:DomainName`

A symbolic name that uniquely identifies an LDAP domain.

`--server_url Server URL`

The URL of the LDAP directory server for the given domain. The LDAP server URL must start with either “ldap://” or “ldaps://”. Starting with “ldaps://” indicates that the given LDAP server requires SSL connection. (i.e. ldap://my-server.myorg.com:443”) If the LDAP server URL starts with “ldaps://”, the user must also specify --server_trusted_ca_file.

`--user_base_dn User Base DN`

The LDAP-distinguished name for the user container. For example, ou=user,dc=mydomain,dc=myenterprise,dc=com.

`--group_base_dn Group Base DN`

The LDAP-distinguished name for the group container. For example, ou=group,dc=mydomain,dc=myenterprise,dc=com.

`--auth_type <FLAT | BOB>`

This attribute is a string that dictates the type of LDAP authentication mechanism to be used for the given domain. "AuthType" can be either "FLAT" or "BOB". "FLAT" means to use the existing one-level bind while "BOB" indicates Bind-Search(Obtain)-Bind. In "BOB" authentication mode, AT uses a proxy account to bind with the Active Directory, and then searches for the distinguished name before authenticating (bind) the user.

For example,"AuthType"="BOB".

`--admin_user <admin user DN>`

This attribute is a string that contains the DN of the admin user or any user which have search permission to the user container, or user

subtree as specified by "UserBaseDN". If the user container is searchable by anyone, including an anonymous user. This attribute can be configured to an empty string. For example, "AdminUser"=""

`--admin_user_password <admin user password>`

This attribute is a string that contains the bind password of the user that is specified in AdminUser. If AdminUser is an empty string, this attribute must also be an empty string. For example, adminUserPassword""=""

`--search_scope <SUB | ONE | BASE>`

This attribute is a string that indicates the search scope. "SearchScope" can be either "SUB", "BASE", or "ONE". For example, "SearchScope"="SUB"

Optional arguments

`--server_trusted_ca_file Trusted CA file Name`

The complete path to the name of the file that contains the trusted CA certificates in PEM format. You must use this parameter if the given LDAP server URL starts with “ldaps://” (indicating the need for an SSL connection). However, if the given LDAP server URL starts with “ldap://”, this parameter must be omitted.

`--schema_type Schema Type`

Specify which type of LDAP schema to use.

Note: If you do use --schema_type, you must omit the following parameters: --schema_type--schema_type, --user_attribute, -i, -o. These values are set automatically, based upon the schema type you chose.

If you do not use --schema_type, neither the rfc2307 nor the msad parameters are set automatically, and you must therefore provide the values yourself.

Two default schema types are currently supported.

- rfc2307: the schema that is specified in RFC 2307
- msad: Microsoft Active Directory schema.

With RFC2307, the following schema is used.

User Object Class: posixAccount
User Attribute: uid
User GID Attribute: gidNumber
Group Object Class: posixGroup
Group Attribute: cn
Group GID Attribute: gidNumber

With Microsoft Active Directory, the following schema is used.

User Object Class: user
User Attribute: cn
User GID Attribute: memberOf
Group Object Class: group
Group Attribute: cn
Group GID Attribute: cn

Note: For msad schema if you select auth type as "BOB", the User Attribute is set to"sAMAccountName".

`--user_object_class User Object Class`

Specify the LDAP object class for the user object. (i.e. posixAccount). This parameter is required if it is absent, but you must not use --schema_type.

`--user_attribute User Attribute`

Specify the user attribute within the user object class, using the following syntax:

<user attribute>=<prplname>,<user base DN>

For example, the LDAP DN for jdoe is as follows:

`"cn=jdoe,dc=mydomain,dc=myenterprise,dc=com"`

where:

- The *<user attribute>* is "cn"
- The *<prplname>* is "jdoe"
- The *<user base DN>* is "dc=mydomain,dc=myenterprise,dc=com"

Do not use this attribute if you use `--schema_type`.

`--user_gid_attribute User Group Id Attribute`

Specify the attribute within the user object class to retrieve the groups the user belongs to.

Do not use this attribute if you use `--schema_type`.

`--group_object_class Group Object Class`

Specify the LDAP object class for the group object. (i.e. posixGroup).

Do not use this attribute if you use `--schema_type`.

`--group_attribute Group Attribute`

Specify the group attribute within the group object class, using the following syntax:

<group attribute>=<group>,<group base DN>

For example, the LDAP DN for adm is as follows:

`"cn=adm,dc=mydomain,dc=myenterprise,dc=com"`

where:

- The *<group attribute>* is "cn"
- The *<group>* is "adm"
- The *<group base DN>* is "dc=mydomain,dc=myenterprise,dc=com"

Do not use this attribute if you use `--schema_type`.

`--group_gid_attribute Group GID Attribute`

Specify the attribute within the group object class to retrieve the group Do not use this attribute if you use `--schema_type`.

Example 1

```
vssat addldapdomain --domainname MYADDOMAIN --server_url ldap://
my_ad_host.mydomain.myenterprise.com -u
cn=users,dc=mydomain,dc=myenterprise,dc=com --group_base_dn
dc=users,dc=mydomain,dc=myenterprise,dc=com --schema_type msad
```

Example 2

```
vssat addldapdomain --domainname MYENTERPRISE --server_url
ldap://my_openldap_host.myenterpise.com -u
dc=people,dc=myenterprise,dc=com --group_base_dn
dc=group,dc=myenterprise,dc=com --schema_type rfc2307
```

Example 3

```
vssat addldapdomain --domainname TESTDOMAIN --server_url ldap://
myldapserver.myenterprise.com -u
ou=users,ou=engineering,dc=myenterprise.com --group_base_dn
ou=groups,ou=engineering,dc=myenterprise.com --schema_type
inetOrgPerson --user_attribute uid --user_gid_attribute gid --
group_object_class MyDomainGroups --group_attribute cn --
group_gid_attribute gid
```

Example 4

```
vssat addldapdomain --domainname TEST --server_url ldaps://
my_openldap_host.
myenterpise.com:443 --server_trusted_ca_file /user/local/
openssl/trusted_cas.pem
-u dc=people,dc=myenterprise,dc=com --group_base_dn
dc=group,dc=myenterprise,dc=com --schema_type rfc2307
```

addprpl

Name

addprpl

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat addprpl --pdrtype <root|ab|cluster> --domain <name> --
prplname <prpl name> [--password <password>] [--credexpiry
<expiry period (sec)>] [--prpltype <default|user|service>] [--
can_proxy] [--can_accept_proxy] [--is_broker_admin] [--
is_domain_admin] [--broker <host:port> --domain_admin_prplname
<domain admin identity> [--domain_admin_domain <type:name>]]
```

Description

Use this command to create authentication principals in the domain. While creating principals, input should be principal name, password, type, etc.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`
: Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--domain <name>`
: The name of the domain in which the principal is to be created.

`--is_broker_admin`
: Gives the broker administrator privilege to the principal being created.

`--is_domain_admin`
: Gives the domain administrator privilege to the principal being created.

`--prplname <prpl name>`
: Name of the principal you want to create.

`--password <password>`
: The password that will be used for the new principal.

`--credexpiry <expiry period (sec)>`
: The expiration interval, expressed as the number of seconds. A zero means the next level of expiry policy is used, i.e., the domain expiry policy is used. If that also is zero, then plugin-wide the expiry policy is used. If that also is zero, then software specifies some defaults.

`--prpltype <default|user|service>`
: The type of principal to be created -- whether a user or a service.

`--can_proxy`
: If specified, this principal can act as a proxy for another principal. This case is specifically useful for web server credentials where the web server proxies to its backend services for the end user who is using web browser.

`--can_accept_proxy`
: If specified, gives rights for the entity to accept proxies. This case is specifically useful for the back end services of a web-server. The web server, before handing out the product web credential of end user, checks if the receiving peer has been cleared to accept the product web credential or whether it can accept the proxy.

The following parameters are required only to create the principal in a remote broker. The remote broker must be running with PBX support. Caller must authenticate to the remote broker using either a broker administrator identity or a domain administrator identity of the remote domain and pass that information using the following parameters:

`--broker <host:port>`
: Remote broker host name and port number.

`--domain_admin_prplname <domain admin identity>`
: Name of the principal who is a broker administrator or a domain administrator of the target domain on the remote broker.

`--domain_admin_domain <type:name>`
: Domain name and type of the remote domain admin identity. If this parameter is not specified, then the admin principal is assumed to be in the same domain where the new principal is being created.

Example 1

```
vssat addprpl --pdrtype ab --domain broker@MyHost --prplname
TomSawyer --password LetTomIn --expinterval 24000 --prpltype
user
```

Example 2

When this operation is performed on a remote broker, the user must execute `setuptrust` and `authenticate` prior to calling `addprpl`:

```
vssat setuptrust --broker root_broker.my_domain.com:2821 --
securitylevel high --hashfile /tmp/root_hash vssat setuptrust --
broker root_broker.my_domain.com:2821 --securitylevel high --
hash 668a754f8201f10955db6326cb076e4086c10477

vssat authenticate --domain vx:dom1@remote_broker.my_domain.com
--broker remote_broker.my_domain.com:1556 --prplname admin --
password secret

vssat addprpl --pdrtype ab --domain
dom1@remote_broker.my_domain.com --password my_pass --prpltype
service --is_broker_admin --is_domain_admin --broker
remote_broker.my_domain.com:1556 --domain_admin_prplname admin -
-domain_admin_domain vx:dom1@remote_broker.my_domain.com
```

authenticate

Name

`authenticate`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat authenticate --domain <type:name> [--prplname <prpl name>
[--password <password>]] [--broker <host:port>]
```

Description

Use this command to obtain a credential for an authentication principal from an Authentication Broker. The authentication principal's name, domain, domain type, and password are used by the Authentication Broker specified by the broker information to validate the identity and issue the credential.

Arguments

This command allows the following arguments:

`--domain <type:name>`
: The name and type of the domain that holds the principal. The private domain names need not be fully qualified ones. The given broker name, without "@<fully qualified broker name>" will also be accepted.

`--prplname <prpl name>`
: The name of the principal that is to be authenticated. This argument is optional if you are using "nt" as domaintype and want to use SSPI. Similarly, it is optional when you use "localhost" domain type. For other domain types, it is not optional and must be provided.

`--password <password>`
: The password of the principal that is to be authenticated. This argument is optional if you are using "nt" as domaintype and want to use SSPI. Similarly, it is optional when you use "localhost" domain type. For other domain types, it is not optional and must be provided.

--broker <*host:port*>

The host and port of the broker. If a domain-broker mapping is already present, providing the brokerinfo is not mandatory. The registered port for Authentication is 2821. If the broker has been configured with another port number, consult your security administrator for information.

Example 1

```
vssat authenticate --domain vx:broker@MyHost --prplname
TomSawyer --password LetTomIn --broker MyHost:2821
```

Example 2

```
vssat authenticate --domain vx:broker --prplname TomSawyer --
password LetTomIn --broker MyHost:2821
```

changepasswd

Name

changepasswd

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat changepasswd --pdrtype <root|ab|cluster> --domain <name> --prplname <prpl name> [--currentpasswd <oldpasswd>] [--newpasswd <newpasswd>] [--repeatednewpasswd <repnewpasswd>]
```

Description

Use this command to change a password for a principal. Password is optionally provided on the command line. If not specified on command line, it is prompted for in non-echo mode.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`
: Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--domain <name>`
: The name of the primary domain.

`--prplname <prpl name>`
: The name of the principal whose password is to be changed.

`--currentpasswd <oldpasswd>`
: The old password that you are changing.

`--newpasswd <newpasswd>`
: The new password you are setting up.

`--repeatednewpasswd <repnewpasswd>`
: The new password typed again as confirmation.

Examples

This command can be used as shown below:

```
vssat changepasswd --pdrtype ab --domain broker@MyHost --prplname TomSawyer --oldpasswd LetTomIn --newpasswd PleaseLetTomIn --repeatednewpasswd PleaseLetTomIn
```

createpd

Name

`createpd`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat createpd --pdrtype <ab|cluster> --domain <name> [--
credexpiry <expiry period (sec)>] [--domain_admin_password
<domain admin password>] [--broker <host:port> --
broker_admin_prplname <broker admin identity> --
broker_admin_domain <type:name>]
```

Description

Use this command to create a private domain in the repository by taking a unique name and other attributes like expiry time of credentials issued for this domain's principals.

Arguments

This command allows the following arguments:

`--pdrtype <ab|cluster>`

Type of private domain repository -- whether Authentication Broker or Cluster. Root Broker is not an option because you cannot create or delete domains in the root private domain repository. The root private domain repository has only one domain, where all Authentication Broker's identities are stored.

`--domain <name>`

The name of the domain to be created.

The domain name cannot be more than 63 characters.

`--credexpiry <expiry period (sec)>`

The expiry period expressed in seconds.

`--domain_admin_password`

Domain administrator password for the domain being created.

The following parameters are required only to create the private domains on a remote broker. The remote broker must be running with PBX support. Caller must authenticate to the remote broker using a broker administrator identity and pass that information using the following parameters:

`--broker <host:port>`
Remote broker host name and port number.

`--broker_admin_prplname <broker admin identity>`
Name of the principal who is a broker administrator on the remote broker.

`--broker_admin_domain <type:name>`
Domain name and type of the remote broker admin identity.

Example 1

```
vssat createpd --pdrtype ab --domain broker@MyHost --credexpiry
24000
```

Example 2

When this operation is performed on a remote broker, the user must execute `setuptrust` prior to calling `createpd`:

```
vssat authenticate --domain
vx:broker@remote_broker.my_domain.com --broker
remote_broker:1556 --prplname admin --password secret

vssat createpd --pdrtype ab --domain new_domain --
domain_admin_password secret --broker
remote_broker.my_domain.com:1556 --broker_admin_prplname admin -
-broker_admin_domain vx:broker@remote_broker.my_domain.com
```

deletebrokerdomain

Name

`deletebrokerdomain`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat deletebrokerdomain --broker <host:port> --domain
<type:name> [--global]
```

Description

Use this command to delete a mapping of a domain to a broker. Such a mapping indicates which broker should be approached when trying to authenticate to a particular domain. There is an option to specify whether this entry should be deleted from local registry or global registry.

Arguments

This command allows the following arguments:

`--broker <host:port>`
: The host and port of the broker. The registered port for Authentication is 2821. If the broker has been configured with another port number, consult your security administrator for information.

`--domain <type:name>`
: The name of the domain to be deleted.

`--global`
: Indication that the entry should be deleted from the global registry. Local registry settings only influence the current logged on OS principal, whereas global registry settings may apply to all OS principals of that host.

Examples

To delete the mapping of nt:NewBrokerDomain to MyHost:2821 from the local registry:

```
vssat deletebrokerdomain --broker MyHost:2821 --domain
nt:NewBrokerDomain
```

To delete from the global configuration:

```
vssat deletebrokerdomain --broker MyHost:2821 --domain
nt:NewBrokerDomain --global
```

deletecred

Name

`deletecred`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat deletecred --domain <type:name> [--prplname <prpl name> [-
-broker <name:port>]]
```

Description

Use this command to delete a credential from a store, given user information like name and domain details. Whatever details were given to request a credential can be used to delete the same.

Arguments

This command allows the following arguments:

`--domain <type:name>`

The name of the domain that holds the principal whose credential is to be deleted.

`--prplname <prpl name>`

Name of the principal whose credential you want to delete.

--broker <*name:port*>

The host and port of the broker. Though port is specified here, it is ignored in the processing of this command. If broker is specified, then only the credential from a specific broker is deleted.

There can be two different credentials for the same authentication principal from two different authentication brokers.

Examples

This command can be used as shown below:

```
vssat deletecred --domain nt:NewDomainName --prplname TomSawyer
--broker MyHost:2821
```

deleteexpiredcreds

Name

`deleteexpiredcreds`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat deleteexpiredcreds
```

Description

Use this command to delete expired credentials from a store.

Arguments

None

deletepd

Name

`deletepd`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat deletepd --pdrtype <ab|cluster> --domain <name>
```

Description

Use this command to delete a private domain from the authentication private domain repository. Deleting a domain deletes the principals in the domain, along with the domain itself.

Arguments

This command allows the following arguments:

`--pdrtype <ab|cluster>`

Type of private domain repository -- whether Authentication Broker or Cluster. Root Broker is not an option because you cannot create or delete domains in the root private domain repository. The root private domain repository has only one domain, where all Authentication Broker's identities are stored.

`--domain <name>`

The name of the domain that is to be deleted.

Examples

This command can be used as shown below:

```
vssat deletepd --pdrtype ab --domain nt:NewBrokerDomain
```

deleteprpl

Name

`deleteprpl`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat deleteprpl --pdrtype <root|ab|cluster> --domain <name> --
prplname <prpl name>[--silent]
```

Description

Use this command to delete a principal from a private domain. The command will fail if the principal and domain do not exist.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`
: Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster.

`--domain <name>`
: Name of the domain in which the principal resides.

`--prplname <prpl name>`
: Name of the principal to be deleted.

--silent
: Disables confirm messages.

Examples

This command can be used as shown below:

```
vssat deleteprpl --pdrtype ab --domain broker@MyHost --prplname
TomSawyer
```

listldapdomains

Name

`listldapdomains`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat listldapdomains
```

Description

Use this command to list all the LDAP domains in the authentication broker. This command needs no additional parameters. Output is similar to the following:

```
listldapdomains
----------------------
----------------------
Found:  2
Domain Name:           VSS
Server URL:            ldap://your_ldap_server.com
SSL Enabled:           No
User Base DN:          <distinguish name of your user container>
User Object Class:     posixAccount
User Attribute:        uid
User GID Attribute:    gidNumber
Group Base DN:         <distinguish name of your group container>
Group Object Class:    posixGroup
Group Attribute:       cn
Group GID Attribute:   gidNumber
...
```

Arguments

None

listpd

Name

`listpd`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat listpd --pdrtype <root|ab|cluster> [--broker <host:port> -
-broker_admin_prplname <broker admin identity> --
broker_admin_domain <type:name>]
```

Description

Use this command to list the domains inside the private domain repository of the local broker or a remote broker. To list the domains from a remote broker, one has to authenticate with the remote broker using a broker admin identity of the remote broker.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`

Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

The following parameters are required only to list the private domains on a remote broker. The remote broker must be running with PBX support. Caller must authenticate to the remote broker using a broker administrator identity and pass that information using the following parameters:

`--broker <host:port>`

Remote broker host name and port number.

`--broker_admin_prplname <broker admin identity>`

Name of the principal who is a broker administrator on the remote broker.

`--broker_admin_domain <type:name>`

Domain name and type of the remote broker admin identity.

Example 1

```
vssat listpd --pdrtype ab
```

Example 2

When this operation is performed on a remote broker, the user must execute `setuptrust` and `authenticate` prior to calling `listpd`:

```
vssat setuptrust --broker remote_broker.my_domain.com:1556 --
securitylevelhigh
```

```
vssat authenticate --domain
vx:broker@remote_broker.my_domain.com --prplname
remote_broker_admin --password secret --broker
remote_broker.my_domain.com:1556

vssat listpd --pdrtype ab --broker
remote_broker.my_domain.com:1556 --broker_admin_prplname
remote_broker_admin --broker_admin_domain
vx:broker@remote_broker.my_domain.com
```

listpdprincipals

Name

`listpdprincipals`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat listpdprincipals --pdrtype <root|ab|cluster> --domain
<name>
```

Description

Use this command to list all the principals in the private domain.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`

Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--domain <name>`

Name of the private domain whose principals you want to list.

Examples

This command can be used as shown below:

```
vssat listpdprincipals --pdrtype ab --domain broker@MyHost
```

removeldapdomain

Name

`removeldapdomain`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat removeldapdomain -d <domain to be removed>
```

Description

Use this command to remove an LDAP domain from the authentication broker.

Arguments

`-d,--domain DomainName`

A symbolic name that uniquely identifies an LDAP domain.

Example

```
vssat addldapdomain -d MYADDOMAIN
```

removetrust

Name

`removetrust`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat removetrust --broker <brokerinfo like host>
```

Description

Use this command to delete the root certificate that comes from the mentioned broker.

Arguments

`--broker`

The name of the broker that issued the root certificate being deleted.

Examples

This command can be used as shown below:

```
vssat removetrust --broker MyHost
```

renewcredential

Name

`renewcredential`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat renewcredential --domain <type:name> --prplname <prpl
name> --broker <host:port>
```

Description

Use this command to renew the credential of a given principal, when you provide a domain and broker.

Arguments

This command allows the following arguments:

`--domain <type:name>`

The name of the domain that holds the credential to be renewed.

`--prplname <prpl name>`

Name of the principal whose credential is to be renewed.

--broker <*host:port*>

The host and port of the broker. The registered port for Authentication is 2821. If the broker has been configured with another port number, consult your security administrator for information.

Examples

This command can be used as shown below:

```
vssat renewcredential --domain nt:NewBrokerDomain --prplname
JohnDoe --broker MyHost:2821
```

resetpasswd

Name

`resetpasswd`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat resetpasswd --pdrtype <root|ab|cluster> --domain <name> --
prplname <prpl name> [--newpasswd <newpasswd>] [--
repeatednewpasswd <repnewpasswd>]
```

Description

The administrator uses this command to reset a password when the authentication principal forgets the password. The command does not require that you type the old password.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`
: Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--domain <name>`
: The name of the primary domain.

`--prplname <prpl name>`
: The name of the principal whose password is to be changed.

`--newpasswd <newpasswd>`
: The new password you are setting up.

`--repeatednewpasswd <repnewpasswd>`
: The new password typed again as confirmation.

Examples

This command can be used as shown below:

```
vssat resetpasswd --pdrtype ab --domain broker@MyHost --prplname
TomSawyer --newpasswd PleaseLetTomIn --repeatednewpasswd
PleaseLetTomIn
```

setcredstore

Name

`setcredstore`

Synopsis

Shut down the Authentication Broker before executing this command. Use the following syntax, without line breaks:

```
vssat setcredstore --storetype <file|memory|registry> --
storefile <file if file type> [--enableobfuscation]
```

Description

Use this command to set credential store details. These details contain the store type (in memory, on file, in Windows registry etc.) If it is on file, the location of the file can be specified and seen.

Arguments

This command allows the following arguments:

`--storetype <file|memory|registry>`
: Information about the type of credential store for which you want to specify details. The store may be in memory, in files, or in the registry.

`--storefile <file if file type>`
: The path where the files reside, if you have chosen **File**.

`--enableobfuscation`
: Indication that obfuscation is to be enabled.

Examples

This command can be used as shown below:

```
vssat setcredstore --storetype file --storefile "C:\Program
Files\VERITAS\Security\Authentication\System Profile\Certstore"
```

Notes

For this release, only **File** is an acceptable choice.

setexpiryintervals

Name

`setexpiryintervals`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat setexpiryintervals --pluginname <plugin name> --prpltype
<default,user,service,webcredential> --credexpiry <expiry
period>
```

Description

Use this command to set any of three levels of credential expiry intervals: generic, user principal, and service (or CLI) principal expiry levels. These intervals are to be set at the plugin level. To go one level up, i.e., from principal to domain to plugin, you must set the expiry to zero at that level. For example, assume the expiry for the principal is 1000. If the administrator wants to remove the principal expiry and make Authentication issue a certificate based on domain expiry, the administrator must set principal expiry to zero. Similarly, from domain to plugin wide.

Arguments

This command allows the following arguments:

`--plugin <plugin name>`
: The name of the plugin where the credential expiry interval is to be set. If plugin names are not changed from the installed state, then its value can be “vx”, “nt”, “nis”, “nisplus”, “unixpwd”.

`--prplptype <default,user,service,product web credential>`
: The type of expiry to be set. For OS domains or public domains only the default expiry policy will be used, since Symantec Product Authentication Service cannot differentiate between a user account and a service account. Therefore, setting user or service expiry policies for native domains may not have any effect on the actual credential expiry.

`--credexpiry <expiry period (sec)>`
: The expiry period expressed in seconds.

Examples

This command can be used as shown below:

```
vssat setexpiryintervals --pluginname vx --prpltype user --
credexpiry 36000
```

setispbxexchflag

Name

`vssat setispbxexchflag`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat setispbxexchflag [--enable|--disable]
```

Description

Use this command to set the PBX Exchange Installed attribute to either enabled state or disabled state (1 or 0) based on `--enable` option or `--disable` option respectively. Based on the PBX Exchange Installed attribute setting, a broker either starts does not start the PBX related services. If set, it starts them; and if not set, it does not. PBX Related services include PBX based authentication support, and remote administration.

Arguments

This command allows the following arguments:

`--enable`
: Starts the PBX related services.

`--disable`
: Does not start the PBX related services.

Examples

This command can be used as shown below:

```
vssat setispbxexchflag --enable
```

setpd

Name

setpd

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat setpd --pdrtype <root|ab|cluster> --domain <name> --
credexpiry <expiry period (sec)>
```

Description

Use this command to set the attributes of the private domains. Currently the only attribute you can set in this way is the expiry time.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`
: Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--domain <name>`
: The name of the domain whose attributes are to be set.

`--credexpiry <expiry period (sec)>`
: The expiry period expressed in seconds.

Examples

This command can be used as shown below:

```
vssat setpd --pdrtype ab --domain Broker@MyHost --credexpiry
4200
```

setpdr

Name

setpdr

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat setpdr --pdrtype <root|ab|cluster> --pdrfile <fqfn of pdr
file>
```

Description

Use this command to change the default location of the private domain repository. When the pdrfile is changed, current configuration is not immediately saved in the new pdrfile. Upon restart, the new pdrfile is loaded.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`
: Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--pdrfile`
: The name of the file that will serve as the private domain repository.

Examples

This command can be used as shown below:

```
vssat setpdr --pdrtype root C:\program
files\veritas\security\authentication\bin\RBAuthPDR
```

setsecuritylevel

Name

`setsecuritylevel`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat setsecuritylevel --level <low|medium|high>
```

Description

Use this command to set the security level.

Arguments

This command allows the following arguments:

`--level <low|medium|high>`

The level of security that you wish to set.

Examples

This command can be used as shown below:

```
vssat setsecuritylevel --level low
```

Notes

The security administrator can configure one of the following levels of security for distributing root certificates:

- High security (2): If a previously entrusted root is acquired from the peer (that is, if no certificate with the same signature exists in our trust store), the user will be prompted to verify the hash.
- Medium security (1): The first authentication broker will be trusted without prompting. Any attempts to trust subsequent authentication brokers will cause the user to be prompted for a hash verification before the certificate is added to the trusted store.
- Low security (0): The authentication broker certificate is always trusted without any prompting.

setuptrust

Name

`setuptrust`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat setuptrust --broker <host:port> -- securitylevel
<low|medium|high> [--hashfile <filename> | --hash <root hash in
hex>]
```

Description

Use this command to contact the broker to be trusted, obtain its certificate or details over the wire, and add to the trust repository if the furnished details are trustworthy.

Arguments

This command allows the following arguments:

`--broker <host:port>`
: The host and port of the broker to be trusted. The registered port for Authentication is 2821. If the broker has been configured with another port number, consult your security administrator for information.

`--hash <root hash in hex>`
: Root hash in hexadecimal format. Trust will be set up in high security mode. Setup trust will fail if the supplied root hash does not verify.

`--hashfile <filename>`
: A binary file containing the root hash. Trust will be set up in high security mode. Setup trust will fail if the supplied root hash does not verify.

`--securitylevel <low|medium|high>`
: The level of security that you wish to set. (See “How Root Certificates are Distributed” on page 33.)Select

Example 1

```
vssat setuptrust --broker MyHost:2821 --securitylevel high
```

Example 2

```
vssat setuptrust --broker root_broker.my_domain.com:2821 --
securitylevel high --hashfile /tmp/root_hash
```

Example 3

```
vssat setuptrust --broker root_broker.my_domain.com:2821 --
securitylevel high --hash
668a754f8201f10955db6326cb076e4086c10477
```

Notes

The security administrator can configure one of the following levels of security for distributing root certificates:

- High security (2): If a previously untrusted root is acquired from the peer (that is, if no certificate with the same signature exists in our trust store), the user will be prompted to verify the hash.
- Medium security (1): The first authentication broker will be trusted without prompting. Any attempts to trust subsequent authentication brokers will cause the user to be prompted for a hash verification before the certificate is added to the trusted store.
- Low security (0): The authentication broker certificate is always trusted without any prompting.

showallbrokerdomains

Name

`showallbrokerdomains`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showallbrokerdomains [--global]
```

Description

Use this command to display all the mapping of domain to broker. Results show broker name, broker port, domain name, and domain type. They indicate that for someone trying to authenticate to the given domain of the given type, they would approach the given broker name at the given port number. The Global option indicates whether this mapping is for all principals or just the current OS logged on principal.

Arguments

This command allows the following arguments:

`--global`

Indication that the command should query all the domains and their brokers from the global configuration registry.

Examples

This command can be used as shown below:

```
vssat showallbrokerdomains --global
```

showalltrustedcreds

Name

`showalltrustedcreds`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showalltrustedcreds
```

Description

Use this command to display a list of all trusted credentials -- that is, root certificates.

Arguments

None

Examples

This command can be used as shown below:

```
vssat showalltrustedcreds
```

showbackuplist

Name

`showbackuplist`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showbackuplist [-f <file name>]
```

Description

Use this command for the following purposes:

- To list critical files and directories to back up
- To take the backup of the displayed list of files

Warning: The `vssat showbackuplist` command does not itself restore files.

Each directory or file displays on a separate line.

The following record types exist:

The B| record	Indicates a file to backup. If the next line begins with an R| tag, the R| tag indicates the name to use when restoring the file. If there is no R| line, then the restore name is the same as the backup name.
The K| record	Specifies a registry key to backup

The format of the output is as follows:

B | FileOrDirToBeBackedup

R | RestoreAboveFileOrDirToFileOrDir

K| RegistryKey

Arguments

This command allows the following arguments:

`-f,--filename FileName`

File name into which to write the output. If you do not specify `-f`, the output is displayed onto the standard output.

Examples

To display on screen the list of directories and files that need to be backed up, run the following command:

```
vssat showbackuplist
```

To get the same listing in a file named `list.txt`, run the following command.

```
vssat showbackuplist -f list.txt
```

showbrokerhash

Name
`showbrokerhash`

Synopsis
For this command, use the following syntax, without line breaks:

```
vssat showbrokerhash
```

Description
Use this command to display the Root Broker hash. This root broker hash is published by the Root Broker administrator for his/her users so that they can do setup trust using this information. Publishing is done using their company's accepted security related information dissemination tools.

Arguments
None

Examples
This command can be used as shown below:

```
vssat showbrokerhash
```

showbrokermode

Name

`showbrokermode`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showbrokermode
```

Description

This command is applicable only on a machine where an authentication broker is installed and can be run only by an Administratror or root. Use this command to display the current mode the broker is running in. It returns the broker mode as an integer value ranging from 0 to 3:

- 0 - Broker not configured yet.
- 1 - Running as Authentication Broker only.
- 2 - Running as Root broker only.
- 3 - Running as both Authentication and Root broker.

Arguments

None

Examples

This command can be used as shown below:

```
vssat showbrokermode
```

showbrokers

Name

`showbrokers`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showbrokers --domain <type:name>
```

Description

Use this command to display the brokers for a particular domain.

Arguments

This command allows the following arguments:

`--domain <type:name>`

The domain for which the brokers are to be displayed.

Examples

This command can be used as shown below:

```
vssat showbrokers --domain nt:NetBrokerDomain
```

showcred

Name

showcred

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showcred [--domain <type:name> [--prplname <prpl name> [--
broker <name:port>]]]
```

Description

Use this command to display the credential available in the local repository, given authentication principal information like name and domain name, type. If used without any options, all credentials in the certificate credential store, for the user invoking the command, will be returned. Password is not required because the user has access to credential manager directory and this access is sufficient. If broker information is not provided, then all the credentials that belong to the authentication principal are shown.

It is possible to have multiple credentials for the same authentication principal from different authentication brokers.

Arguments

This command allows the following arguments:

`--domain <type:name>`
: The name of the domain that holds the principal whose credential is to be shown.

`--prplname <prpl name>`
: Name of the principal whose credential you want to display.

--broker <*host:port*>
: The host and port of the broker. The port must be specified, but it is ignored for this command.

Examples

This command can be used as shown below:

```
vssat showcred --domain vx:broker@MyHost --prplname admin
```

showcredinfo

Name

showcredinfo

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showcredinfo --tag <identity tag> [--en]
```

Description

Use this command to display the principal and domain information of a remotely provisioned identity on the target machine.

Arguments

This command allows the following arguments:

`--tag <identity tag>`

Unqualified identity tag. When a unique identity is provisioned onto a large number of machines, the complete principal name will be the tag@fully_qualified_host_name.

`--en`

Displays the identity information in english.

Examples

This command can be used as shown below:

```
vssat showcredinfo --tag NBU_AGENT
```

Shows

```
Principal Name : NBU_AGENT@my_host.my_domain.com
Domain Type    : vx
Domain Name  : broker@my_broker.my_domain.com
Broker Name    : my_broker.my_domain.com
Broker Port      : 1556
```

showcredstore

Name
showcredstore

Synopsis
For this command, use the following syntax, without line breaks:

```
vssat showcredstore
```

Description
Use this command to display credential store details. These details contain the store type (in memory, on file, in Windows registry etc.) If it is on file, the location of the file can be specified and seen.

Arguments
None

Examples
This command can be used as shown below:

```
vssat showcredstore
```

showdomains

Name

`showdomains`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showdomains --pluginname <plugin name>
```

Description

Use this command to query a plugin for the domains it supports.

Arguments

This command allows the following arguments:

`--pluginname <plugin name>`

Name of the plugin whose supported domains you wish to see.

Examples

This command can be used as shown below:

```
vssat showdomains --pluginname vx
```

showexpiryintervals

Name

`showexpiryintervals`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showexpiryintervals --pluginname <plugin name>
```

Description

Use this command to display the intervals of credential expiry that have been set. The CLI can set four levels of credential expiry intervals: generic, user, web, and service principal expiry intervals. These intervals are to be set at the plugin level. For private domain, there is a generic expiry interval support.

Arguments

This command allows the following arguments:

`--pluginname <plugin name>`

Name of the plugin whose credential expiry levels you want to see.

Examples

This command can be used as shown below:

```
vssat showexpiryintervals --pluginname vx
```

showglobalplugininfo

Name
`showglobalplugininfo`

Synopsis
For this command, use the following syntax, without line breaks:

```
vssat showglobalplugininfo
```

Description
Use this command to show the credential expiry policies for all plugins. The order in which credential expiry policy is applied is: (1) individual principal expiry policy, (2) domain expiry policy, (3) plugin expiry policy, (4) global, all plugins expiry policy.

Arguments
None

Examples
This command can be used as shown below:

```
vssat showglobalplugininfo
```

showispbxexchflag

Name

`vssat showispbxexchflag`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showispbxexchflag
```

Description

Use this command to show whether the PBX Exchange Installed attribute is set on the broker or not. Based on the PBX Exchange Installed attribute setting, a broker either does or does not start the PBX related services. If set, it starts them; and if not set, it does not. PBX Related services include PBX based authentication support, and remote administration. Output of this command is 0 if the flag is not set and 1 if it's set.

Arguments

None

Examples

This command can be used as shown below:

```
vssat showispbxexchflag
```

showpd

Name

showpd

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showpd --pdrtype <root|ab|cluster> --domain <name>
```

Description

Use this command to display the attributes of the private domains. Currently the only attribute that will be displayed is the expiry interval.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`

Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--domain <name>`

The name of the domain whose attributes you want to see.

Examples

This command can be used as shown below:

```
vssat showpd --pdrtype ab --domain broker@MyHost
```

showpdr

Name

`vssat showpdr`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showpdr --pdrtype <root|ab|cluster>
```

Description

Use this command to see the locations of the private domain repositories.

Arguments

`--pdrtype <root|ab|cluster>`

Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

showplugininfo

Name

`showplugininfo`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showplugininfo --pluginname <name of the plugin>
```

Description

Use this command to show all the details about a plugin. Details include expiry policies, etc.

Arguments

This command allows the following arguments:

`--pluginname <name of the plugin>`

The name of the plugin (such as vx, nis, nisplug, unixpwd, nt) for which you want to see details.

Examples

This command can be used as shown below:

```
vssat showplugininfo --pluginname vx
```

showprpl

Name

showprpl

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showprpl --pdrtype <root|ab|cluster> --domain <name> --
prplname <prpl name>
```

Description

Displays attributes -- such as principal type and expiry policy -- of a principal within a domain.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`
: Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--domain <name>`
: The name of the domain in which the principal resides.

`--prplname <prpl name>`
: The name of the principal whose attributes you want to see.

Examples

This command can be used as shown below:

```
vssat showprpl --pdrtype ab --domain broker@MyHost --prplname
TomSawyer
```

showsecuritylevel

Name
`showsecuritylevel`

Synopsis
For this command, use the following syntax, without line breaks:

```
vssat showsecuritylevel
```

Description
Use this command to display the security level.

Arguments
None

Examples
This command can be used as shown below:

```
vssat showsecuritylevel
```

showsystemtrustdir

Name

`showsystemtrustdir`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showsystemtrustdir
```

Description

Use this command to see if the systems-wide trust information is being used and to show the corresponding directory. Results show the system default trust directory list as a colon separated string. This directory is a platform-specific one that OpenSSL supports. The command will display whatever OpenSSL picks plus the directory value stored in SSL_CERT_DIR environment variable. All the root certificates in those directories are trusted roots.

The `vssat showsystsemtrustdir` may appear to return a non-existent directory. The explanation for this behavior stems from the fact that the default trust directory set up by SSL is:

```
/usr/local/ssl/certs:/var/VRTSat/.VRTSat/profile/systruststore
```

X509_get_default_cert_dir returns it. This directory need not be present. If it is present and has self-signed certificates, then those will be trusted. If it is there, it is used; else it is not used.

Examples

This command can be used as shown below:

```
vssat showsystemtrustdir
```

showversion

Name

`showversion`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat showversion
```

Description

Use this command to display the version of the Symantec Product Authentication Service command line interface.

Examples

This command can be used as shown below:

```
vssat showversion
```

updateplugin

Name

updateplugin

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat updateplugin --pluginname <plugin name> --attribute
<attribute name> --value <attribute value> --type {int|string}
```

Description

Use this command to update the plugin information. This command works with all the plugins and can be used to enable/disable the plugin or to update any of its attributes.

Arguments

This command allows the following arguments:

`--pluginname <plugin name>`
: Name of the plugin to be updated.

`--attribute <attribute name>`
: Name of the attribute to be changed.

`--value <attribute value>`
: New value of the attribute.

`--type {int|string}`
: Type of attribute. It can be either a string or an integer.

Example

This command can be used as follows:

- To change the default PAM service name to 'SampleAuth'

  ```
  vssat updateplugin --pluginname pam --attribute ServiceName
  --value sampleauth --type string
  ```

- To turn off the PAM plugin

  ```
  vssat updateplugin --pluginname pam --attribute IsEnabled
  --value 0 --type int
  ```

- To enable a plugin, run the following:

  ```
  vssat updateplugin --pluginname <plugin name> --attribute
  DoNotLoad --value 0 --type int
  ```

- To disable the plugin, run the following:

  ```
  vssat updateplugin --pluginname <plugin name> --attribute
  DoNotLoad --value 1 --type int
  ```

updateprpl

Name

`updateprpl`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat updateprpl --pdrtype <root|ab|cluster> --domain <name> --
prplname <prpl name> --prpltype <default|user|service> --
credexpiry <expiry period (sec)> [--can_proxy] [--
can_accept_proxy]
```

Description

Use this command to update the attributes of the principal. These attributes are the ones that were taken at creation time.

Arguments

This command allows the following arguments:

`--pdrtype <root|ab|cluster>`
: Type of private domain repository -- whether Root Broker, Authentication Broker, or Cluster

`--domain <domain name>`
: Name of the domain in which the principal resides.

`--prplname <prplname>`
: Name of the principal whose attributes are to be updated.

`--prpltype <default|user|service>`
: If this is different, then this principal is updated.

`--credexpiry <expiry period (sec)>`
: The expiry period expressed in seconds. To unset the expiry time for a principal, changing it back to the default expiry policy for the domain, set it to 0. If this is different, the principal will be updated.

`--can_proxy`
: If specified, this principal can act as a proxy for another principal. This case is specifically useful for web server credentials where the web server proxies to its backend services for the end user who is using web browser. If this is different, the principal will be updated.

`--can_accept_proxy`
: If specified, gives rights for the entity to accept proxies. This case is specifically useful for the back end services of a web-server. The web server, before handing out the product web credential of end user, checks if the receiving peer has been cleared to accept the product web credential or whether it can accept the proxy.

Examples

This command can be used as shown below:

```
vssat updateprpl --pdrtype ab --domain broker@MyHost --prplname
TomSawyer --prpltype user --credexpiry 0
```

validategroup

Name

`validategroup`

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat validategroup --groupname <name> --domain <type:name> --
broker <host:port>
```

Description

Use this command to check the validity of a given group when you provide the name of the domain and broker.

Arguments

This command allows the following arguments:

`--groupname <name>`

The name of the group to be validated.

`--domain <type:name>`

The name of the domain that holds the group to be validated.

--broker <*host:port*>

The host and port of the broker. The registered port for Authentication is 2821. If the broker has been configured with another port number, consult your security administrator for information.

Examples

This command can be used as shown below:

```
vssat validategroup --groupname MyGroup --domain
nt:NewBrokerDomain --broker MyHost:2821
```

validateprpl

Name

validateprpl

Synopsis

For this command, use the following syntax, without line breaks:

```
vssat validateprpl --domain <type:name> --prplname <prpl name> --broker <host:port>
```

Description

Use this command to check the validity of a given principal when you provide the name of the domain and broker.

Arguments

This command allows the following arguments:

`--domain <type:name>`
: The name of the domain that holds the principal to be validated.

`--prplname <prpl name>`
: Name of the principal to be validated.

`--broker <host:port>`
: The host and port of the broker. The registered port for Authentication is 2821. If the broker has been configured with another port number, consult your security administrator for information.

Examples

This command can be used as shown below:

```
vssat validateprpl --domain nt:NewBrokerDomain --prplname JohnDoe --broker MyHost:2821
```

Utilities

athealth

Name

athealth

Synopsis

athealth -i*<install dir>* -d *<data dir>* [-l *<log level>*] [-g]

Description

Health check utility is used to perform a quick scan on basic sanity of a particular AT installation. The utility checks if AT has been installed properly. Additionally, this utility is used to check the basic parameters if an error is encountered with the AT setup.The utility can be used to on a client as well as broker installation.

Required Arguments

-l *<log level>*

Log level can be a value between 0 (no logging) to 4 (max logging).

-g

Creates output file "athealthconf.out" that contains diagnostic info gathered from run of the utility.

Optional Arguments

-i *<install dir>*

Install directory is the directory of installation that is to be checked using the athealth utility.

Typically this is the parent directory of directory that contains vrtsat_t.dll or libvrtsat_t.so.

-d *<data dir>*

Data directory is the directory having AT configuration and credential files.

atldapconf

Name

atldapconf

Description

The LDAP configuration tool is a CLI program that facilitates configuring LDAP plugin for the Authentication broker. Use this command to connect to the enterprise LDAP server and detect the default parameters for searching the users and groups.

To call the LDAP configuration tool run the atldapconf command. The tool uses following CLIs:

- -d, discover
- -c, createatcli
- -x, atconfigure

-d, discover

Name

discover

Synopsis

Use the following syntax, without line breaks:

```
atldapconf -d -s <ldap server name> [- p <ldap server port>]-u
<search_user> [-g <search group>] [-f <attribute_list_file>] [-m
<admin_username>][-w <admin_password>][-l <loglevel>]
```

Description

Use this command to connect to the LDAP server. This command searches the attributes of the user and the group. It creates a attribute list file that contains the valid values for all the attributes in an descending order of priority. You can change the order of priority.

The discover command also retrieves the valid values for the LDAP attributes, which have multiple values such as, ObjectClass. Other attributes of LDAP directory are configurable.

Further, you can also search the commonly used attributes that exist on the server and put all the valid attributes in the same attributes list file. The commonly used attributes differ for different LDAP implementations. These values are pre-defined in separate lists for each LDAP implementation. The predefined values are defined in a header file. For example, the list for user gid attributes looks like, - {"gidNumber", "memberOf","gid" }

Required Arguments

`-s <ldap server name>`
Name of the LDAP server. On Windows platforms, if the machine is logged onto the network, then this parameter is optional.

`-u <search_user>`
Used to find out the base search paths for users

`-g <search group>`
Used to find out the base search paths for group.

Optional Arguments

`-f <attribute_list_file>`
Name of the attribute list file. The default file name is "AttributeList.txt".

`- p <ldap server port>`
The port of the LDAP server. The default value is 389. To bind to the server, the command uses the username and password. If these options are not provided, the commands prompts the user to provide a username and password. Currently, only simple authentication is supported, which takes the user name and password in clear text.

`-m <admin_username>`
User name of the connecting user. This is required to make the initial connection to the ldap server when the anonymous searches are disabled.

`-w <admin_password>`
Password of the connecting user. This is required to make the initial connection to the ldap server when the anonymous searches are disabled.

`-l <loglevel>`
Generates a log file named "atldapconf.debug". The loglevel determines the amount information that goes into the log. The value of loglevel ranges from 0 to 4.

Examples

```
atldapconf -d -s sample.server.com -g SAMPLE-DIST-LIST
```

-c, createatcli

Name

`createatcli`

Synopsis

Use the following syntax, without line breaks:

```
atldapconf -c -d <domainname> [-i <attribute_list_file>] [-o
<at_cli_file>] [-a <FLAT|BOB>] [-s <BASE|ONE|SUB>] [-l
<loglevel>]
```

Description

Use this command to take the attribute list generated by the discover command as input. The command parses the attributes list file and selects the attribute with the highest priority and creates a CLI file complete with vssat addldapdomain.

Required Arguments

`-d <domainname>`
: The domain name.

Optional Arguments

`-i <attribute_list_file>`
: The name of attribute list file. The default file name is "AttributeList.txt".

`-o <at_cli_file>`
: The name of the AT CLI file. The default file name is "CLI.txt".

`-a <FLAT|BOB>`
: The type of authentication. The default authentication type is FLAT.

`-s <BASE|ONE|SUB>`
: The scope of search. The default scope type is SUB.

`-l <loglevel>`
: Generates a log file named "atldapconf.debug". The loglevel determines the amount information that goes into the log. The value of loglevel ranges from 0 to 4.

Examples

```
atldapconf -c -d domainname
```

-x, atconfigure

Name

`atconfigure`

Synopsis

Use the following syntax, without line breaks:

```
atldapconf -x [-f <at_cli_file>] [-p <at_install_path>] [-o
<broker_port>] [-l <loglevel>] [-v verify]
```

Description

Use this command to read and execute the AT CLI that was generated by the -c, createatcli command, and add the domain to AT.

Optional arguments

`-f <at_cli_file>`
: The name of the AT CLI file. Default file name is "CLI.txt".

`-p <at_install_path>`
: The install path. It checks in the present working directory and default locations of installation for older versions of AT.

`-o <broker_port>`
: The broker port. Default port is 2821.

`-l <loglevel>`
: Generates a log file named "atldapconf.debug". The loglevel determines the amount information that goes into the log. The value of loglevel ranges from 0 to 4.

`-v verify`
: Verifies the newly added domain, after adding it. If required, the command prompts the user for a user name and password to check if the user can be authenticated.

Note: The broker service needs to be up and running for using the -v option.

Examples

atldapconf -x -l 4 -v

Chapter 7

Other Management Tasks

Other aspects of managing Symantec Product Authentication Service are not represented in the Administration Console. We offer a brief discussion of them here, with cross references to more detailed discussion, where appropriate.

The chapter includes the following topics:

Optional Authentication Client Configurations

Optionally, the administrator can configure the Authentication client by specifying outbound port ranges or by specifying the interface to which the library should be bound.

Specifying Outbound Port Ranges for Authentication Client

The Authentication client supports port ranges for outbound ports. The port ranges can be specified in the registry on Windows and in `/etc/vx/vss/VRTSat.conf` on UNIX.

Specifying Outbound Port Range for Windows

On Windows, `HKEY_LOCAL_MACHINE\Software\VERITAS\Security\Authentication\Client` can have two keys - PortRangeMin and PortRangeMax.

- PortRangeMin specifies the starting port number.
- If PortRangeMax is not specified, PortRangeMax is defaulted to PortRangeMin + 1000.

Specifying Outbound Port Range for UNIX

On UNIX, the section is `Security\Authentication\Client`. The key name and semantics are identical to those in Windows.

Specifying an Interface for Authentication Client

On machines which have multiple interfaces, the VRTSat client library can be configured to bind to a specific interface. This interface can be specified in the following registry location:

Specifying a Client Interface for Windows

On Windows, use the following setting to specify a client interface:

HKEY_LOCAL_MACHINE\Software\VERITAS\Secuity\Authentication\Client,
UseInterface = "IP Address"

Specifying a Client Interface for UNIX

On UNIX, use the following setting to specify a client interface:

In the /etc/vx/vss/VRTSat.conf file,
in Security\Authentication\Client section,

UseInterface = "IP address"

Here IP address is the interface address.

Managing the Security of Root Certificates

Any two entities that try to setup an SSL communication link need to trust the root certificates that the entities certificates chain to. Unless the root certificates are distributed successfully, SSL communication links cannot be established.

Symantec Product Authentication Service identifies three security levels to solve this problem:

- High: In high security level, the root certificate is distributed in-band, but the user is prompted for acceptance before establishing the connection.
- Medium: In the medium security level, the certificate is only accepted the first time. The next time a new root is encountered, the user is prompted.
- Low: In the low-level security level, a new root certificate is always accepted automatically.

Note: For the Root Broker and Authentication Broker, security is set at **high**. For the client, the security level can be set through the Credential Manager: General screen.

Managing Symantec Application Clients and Services

A resource management application is a Symantec product whose resources are being protected by Symantec Product Authentication Service.

- A Symantec application client is a program that accesses a service or function provided by another program, called a Symantec application service.
- A Symantec application service is a program that is contacted by, and provides services to, a Symantec application client. The service with which the client must communicate is installed on a host.

Certain management tasks are specific to a particular Symantec resource management application. For example, a NetBackup administrator may set the backup schedules, etc. For information on this type of management, consult documentation for the specific resource management application.

Generally, management of a Symantec application client is not the concern of the Authentication security administrator, except for establishing which Authentication Broker the client should use.

Managing Specific Symantec Application Clients

Note: A Symantec application client will get its Administration Console from the host on which the Symantec application service is installed. The only portion of the Administration Console that will be available to the client is the portion used for managing credentials.

Preparing a New Application to Use Authentication

When a new Symantec resource management application is to be installed, before the application can use the Symantec Product Authentication Service, you must add the services of the new remote management application into the Authentication Broker's private domain.

Broker Credential Renewal

Brokers keep track of the validity period of their credentials and automatically start renewing the credentials one year before the expiry of the existing credentials.

Broker credential renewal can be done automatically as well manually .

Automatic Broker Credential Renewal Process

3 **Broker startup:** During startup, the broker checks if the automatic broker credential renewal is on. Automatic renewal is on by default. Consuming products can turn it off using the "AutomaticCredentialRenew" parameter. For more information, see "AutomaticCredentialRenew." If this parameter is set to off, the broker does not attempt to renew the credential by itself. Administrator will have to do this manually.

4 **Renewal threshold period:** If the automatic renewal is on, then broker checks the renewal threshold period. It is one year before the expiry of the existing credential. For example if the validity is for eight years, then seven years from the time broker is commissioned.

5 **Renewing broker credentials:** If the renewal threshold period is reached, then it will start renewing the broker credentials. First, the broker will take a

snapshot of its certificate store. This will allow us to restore to the previous state if something goes wrong.

a If the current broker mode is Root or Root+AB, then the Root credential is renewed first.

   A new Root credential is generated out of the existing Root key pair and deposited in the credential store. The new Root credential has a new validity period. This new Root credential over writes the existing Root credential in both regular certificate store and the trusted store. For more information see, "Renewing Root Broker Credentials."

   If the broker mode is Root+AB, then a new AB credential is generated using the existing AB key pair. The new AB credential has a new validity period.

   For more information see, "Renewing Root+AB Credentials."

b If the current broker mode is AB only, then the broker renews its own credential with its remote Root broker.

   Note: The AT package on the AB machine needs to be upgraded for this.

   The default AB's credentials renewal threshold is set to a week less than the Root's credentials. Thus, the AB's credential are renewed one week after the Root's credentials are renewed.

   The new AB credentials has a new validity period and overwrites the older credentials.

   For more information, see"Renewing AB Credentials."

Incase, the renewal fails, the broker still come up with its existing credential and retries the operation a day later. The broker keeps on trying until the renewal is successful.

Broker credential renewal can also be done manually using the -w option. For more information see, "Manual Broker Credential Renewal Option."

Renewing Root Broker Credentials

This automatic credential renewal is triggered upon reaching the renewal threshold.

Alternately, administrator can also renew the Root credentials manually. For more information, see "Renew Root Broker Credentials."

Renewing AB Credentials

The AT broker supports AB credentials of the remote Root brokers. However, the AB needs to re-establish trust with the remote Root broker if it is running in Root only mode. This is because, the broker running in Root only mode uses the Root credential for accepting the incoming connections and when it renews the

Root credential, clients have to re-establish trust with that Root broker. On the other hand, AB is not required to re-establish trust if the remote Root broker is running in Root+AB mode. This is because, the remote Root+AB broker uses its AB credential to accept the connections and the certificate chain is completed using the old Root credential from the client's trusted store.

If products do not want to upgrade their ABs, then they can manually acquire a new credential for each of the AB that is under the newly renewed Root broker.

To acquire new credentials manually

1. Perform a setuptrust against the renewed Root broker to download the new Root credentials.
2. Run the "`vxatd -a -n <broker identity> -p <password> -x <domain type> -y <domain name> -q <root broker name> -z <root broker port> -h <hash file name>`" command to acquire new AB credentials. The new AB credentials are available with its new validity.

Renewing Root+AB Credentials

To renew credentials of Root+AB first renew Root broker credentials and then AB as explained above.

AT Client and Broker Credential Renewal

The existing credentials issued by the AB continue to function even after renewing the AB credentials. Clients can continue to acquire new credentials and renew the existing credentials until the older Root credential expires. Clients would continue to operate with the older Root credential in their trusted store. Thus, peer credentials that are signed by both old and new ABs will be accepted. Similarly, the old credentials issued by the old (pre-renew) broker will be accepted by the peers that have established trust with the new (renewed) Root broker.

The new credentials issued by the renewed AB and Root brokers can have expiry date up to 20 years. The actual expiry date depends on the type of credential and the expiry intervals.

To download the new Root credentials, AT clients are required to re-establish trust with their broker once before the old Root credential expires. AT clients can re-establish trust within a year (Root credential renewal threshold period) after the Root/AB renewed their credentials.

To perform the trust establishment in high security mode, the clients need to receive the hash of the new Root credential out of band. If the AT CLI is used, then it will prompt to verify the incoming Root credential.

Broker Renewal Configuration Parameters

The following parameters are added to the broker configuration to support automatic broker credential renewal:

AutomaticCredentialRenew

Configures the broker to automatically renew its credentials towards the end of the validity period. This parameter applies to both Root and AB.

Section: [Security\Authentication\Authentication Broker]

Key: AutomaticCredentialRenew

Type: Integer

Allowed Values: 0/1

Default: 1

RootRenewThreshold

Specifies when the automatic Root broker credential renewal should happen. The broker starts credential renewal when the remaining validity period falls below this limit.

Section: [Security\Authentication\Authentication Broker]

Key: RootRenewThreshold

Type: Integer

Unit: days

Allowed Values: 1 to 20*365

Default: 365

ABRenewThreshold

Specifies when the automatic AB credential renewal should happen. The broker starts credential renewal when the remaining validity period falls below this limit.

Section: [Security\Authentication\Authentication Broker]

Key: ABRenewThreshold

Type: Integer

Unit: days

Allowed Values: 1 to 20*365

Default: 360

ABCredExpiry

Specifies the credential expiry limit for the new AB credentials issued by a Root broker.

Section: [Security\Authentication\Authentication Broker

Key: ABCredExpiry

Type: Integer

Unit: seconds

Default Value: 20*365*24*3600

RBCredExpiry

Specifies the credential expiry limit for the new Root credentials.

Section: [Security\Authentication\Authentication Broker

Key: RBCredExpiry

Type: Integer

Unit: seconds

Default Value: 20*365*24*3600

Manual Broker Credential Renewal Option

To manually renew the broker credentials, use the -w option added to vxatd. This command takes a back up the existing credential store.

Shutdown the broker process before running the command and start it manually afterwards.

Renew Root Broker Credentials

Run the following command to renew the Root broker credentials:

```
# vxatd -o -r -w
```

Renew Authentication Broker (AB) Credentials

Run the following command to renew the Authentication broker credentials:

```
# vxatd -o -a -w
```

Renew Root+AB Credentials

```
# vxatd -o -a -r -w
```

You can also renew the Root+AB broker in two steps. First, renew the root credential only and then renew the AB credential as explained above.

Configuring Broker Validity

Products can configure the broker credential validity up to 20 years. Only the renewed credentials can be configured for such validity.

Appendix A

Debugging and Logging

This chapter includes the following topics:

Purpose

The Symantec Product Authentication Service is capable of producing detailed debug information at run time. This information may be requested by the development team as an aid in tracking down bugs.

Enabling Service Debugging

You can use the command `vxatd -d` to enable Symantec Product Authentication Service debugging and logging.

Enabling Client Side Debugging

Client side debugging/logging can be turned on as follows:

On UNIXx

In Bourne or Korn shell,

```
$ AtClientDebugLog=4:<log file>
$ export AtClientDebugLog
```

In C shell,

```
$ setenv AtClientDebugLog 4:<log file>
```

On Windows

```
set AtClientDebugLog=4:<log file>
```

Two more environment variables generate debug output from the shared library loader module. They can be turned on as shown below:

On UNIX

In Bourne or Korn shell,

```
$ VRTSat_API_DEBUG_LEVEL=4
$ VRTSat_API_DEBUG_FILE=<log file>
$ export VRTSat_API_DEBUG_LEVEL
$ export VRTSat_API_DEBUG_FILE
```

In C Shell,

```
$ setenv VRTSat_API_DEBUG_LEVEL=4
$ setenv VRTSat_API_DEBUG_FILE=<log file>
```

On Windows

```
set VRTSat_API_DEBUG_LEVEL=4
set VRTSat_API_DEBUG_FILE=<log file>
```

Choosing Log Level

To indicate the log level, `vxatd -d` options takes an optional parameter. The log levels are:

- 0 - No logging
- 1 - Log errors
- 2 - Log warnings
- 3 - Log debug messages
- 4. - Log Info messages

For example:

`vxatd -d4` (sets the log level to “log info messages”)

`vxatd -d1` (sets log level to errors)

In non-interactive mode, DebugLevel is picked up from the "DebugLevel" values in the localconfig. The default DebugLevel is 1 (error).

A log level specified from the command line will override the log level specified in the local config. This allows administrators to run `vxatd` in debug mode for some time and then run it in error log mode later.

The AtClientDebugLog variable will have no effect on the server logging.

Location of Log Files

In non-interactive mode (daemon or NT service), messages are logged to `vxatd.log`. The log file name is configurable. It is picked up from "DebugLogFileName" entry in the localconfig. Defaults for this value are:

- Windows: `vxatd.log` in the current directory
- UNIX: `/var/VRTSat/vxatd.log`

Appendix B

LDAP Plugin Details

Organizations who manage their user and network data in Lightweight Directory Access Protocol (LDAP) directories, can utilize Authentication Broker's LDAP Authentication plugin to perform user authentication.

This section describes the storage of LDAP data and explains how the plugin works.

Check the Release Notes for possible further configuration information.

For information on enabling LDAP, see "Basic Steps for Enabling LDAP" on page 41 of the Symantec Product Authentication Service *Administrator's Guide*.

Schema

Enterprises who migrated NIS data to LDAP directories are expected to follow the schema specification described in RFC 2307. However, RFC 2307 is merely a guideline, not an absolute requirement. Some domain authentication implementations, such as Microsoft Active Directory, do not follow RFC 2307.

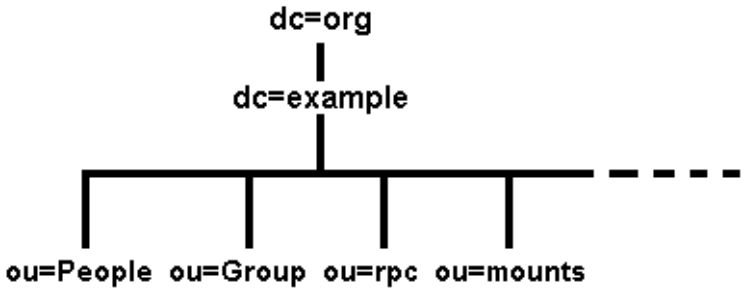
The LDAP Authentication plugin is designed to support RFC 2307 by default, but it is also flexible enough to accommodate other deployment scenarios by making attributes, base distinguished names (DNs), and object classes configurable.

Storing NIS Data in an LDAP Directory

RFC 2307 specified a schema for storing NIS data in an LDAP directory. It provided attributes and object classes for all common databases such as passwd, group, hosts, shadow, services, netgroup, protocol, ethers etc. However, it does not mandate how a tree is to be organized.

A common practice is to use a tree structure like this:

A Common Tree Structure



Here, domain-component naming is used to match the domain name `example.org`. Separate subtrees are provided for each type of information. Data from the passwd file goes into `ou=People` rather than `ou=passwd` because it is assumed that each person has one primary account and that other information will be added to the same entry (phone number, -mail address etc.).

User Password Data

The following is an example of how a typical account entry looks:

```
dn: uid=jdoe,ou=People,dc=example,dc=org
uid: jdoe
cn: John Doe
```

```
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
shadowLastChange: 11296
shadowMax: 99999
shadowWarning: 7
loginShell: /bin/csh
uidNumber: 500
gidNumber: 500
homeDirectory: /home/jdoe
gecos: John Doe
userPassword: {crypt}$Aab1$uQSw.ohy$XuiRSC...
```

User passwords should be encrypted before storing into LDAP directories. For more information on how passwords are encrypted, please read the man page about the crypt command on your UNIX shell (i.e. man crypt).

Group Data

RFC 2307 defines group membership using the `memberUid` attribute in a posixGroup object. Because `memberUid` is a multi-valued attribute, the group map can be searched very efficiently. The example group entry would look like this in the directory:

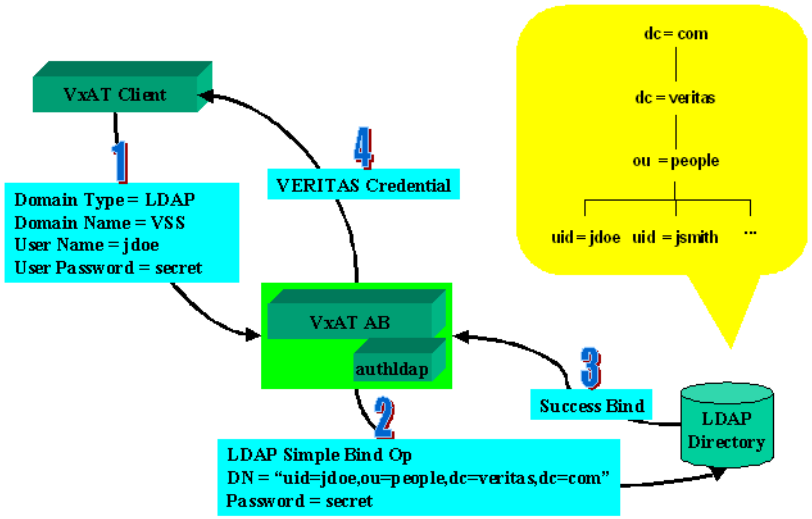
```
dn: cn=ldapbods,ou=Group,dc=example,dc=org
objectClass: posixGroup
objectClass: top
cn: ldapbods
userPassword: {crypt}x
gidNumber: 389
memberUid: tim
memberUid: steve
memberUid: colin
memberUid: damy
memberUid: Andrew
```

How authldap Works

The LDAP Authentication plugin module (`authldap`) is a shared library, which ships with the Authentication Broker.

Figure 1: LDAP Authentication illustrates the authentication process. In this example, we have an enterprise user, John Doe, who belongs to the veritas.com domain. John Doe had carefully chosen “secret” as his password. Domain name, VSS, is configured to map to LDAP directory tree, “ou=people,dc=veritas,dc=com”. With domain name to LDAP tree mapping, we can precisely construct an LDAP DN for the user by using domain name and user name.

Figure B-11 LDAP Authentication



Interpretation of Figure Showing LDAP Authentication

The figure can be interpreted as follows:

1. The Authentication Client performs user authentication with the Authentication Broker (AB). For this example:
   - Domain type = LDAP
   - Domain name = VSS
   - User name = jdoe
   - User password = secret
2. Authentication Broker does the following:
   a. Constructs the corresponding LDAP DN for John Doe based on the given user name and domain name.
   b. Establishes an SSL session with the LDAP directory if necessary.
   c. Performs an LDAP bind operation with the LDAP directory.
3. A successful LDAP bind operation indicates the user is authenticated to the LDAP directory. Authentication retrieves group information from LDAP directory.
4. Authentication issues product credential to John Doe.

Glossary

Administration Console

A graphical interface used to administerAuthentication. For example, the administrator uses it to indicate the location of the different components, trust relationships, plugins, private Symantec domains, etc.

Application Service

A program that is contacted by, and provides services to, an application client.

CLI

Command line interface.

Object

An entity, whether visible and tangible or not, that can be manipulated by a process or program.

Plaintext

The unencrypted input to an encryption process.

Resource Management Application

A Symantec product whose resources are being protected by Symantec Product Authentication Service.

Security Policy

A well-thought-out set of decisions regarding how your product should be used in a customer's environment, how your product could be misused, and what range of access rules your customers would like to see enforced by your product.

Subject

A thread executing on behalf of (i.e., with the permissions of) an authentication principal. Those permissions would have been granted by an administrator explicitly to the security principal which includes that authentication principal.

Index

D

E

G

H

L

M

N

O

P

Q

R

S

T

V

W