Vintela Authentication from SCO Release 2.2

System Administration Guide

November 19, 2003

Table of Contents

Preface

Why VAS?

System administrators today must support heterogeneous platforms and applications for their users' business needs and requirements. When providing users with the best network accessibility and state-of-the art applications, system administrators are left with an integration and security nightmare.

Critical to the security of any network is the authentication and verification of user identities. By adopting Microsoft Active Directory some issues with authentication and identity management are solved. However, this introduces significant problems for the organization that additionally runs business critical applications on UNIX platforms and Linux. When system administrators are required to maintain multiple user authentication systems users are required to remember multiple passwords. System administrators might be clever enough to devise script-based password synchronization tools but this solution can become hard to support, maintain, and train additional staff to use.

Vintela Authentication from SCO (VAS) provides the solution for integrating UNIX and Linux systems with Active Directory. It supplies the discipline and controls necessary to ensure the security and integrity demanded in business environments.

VAS allows administrators to provide a secure environment where users have the same username and password for Windows, UNIX, and Linux logins without having to maintain password synchronizers or perform user administration tasks on multiple systems. VAS users can log in and authenticate to Active Directory from their UNIX servers and workstations the same way they do from Windows XP and Windows 2000/2003. VAS makes it possible to manage all users from within the standard Active Directory management environment.

Audience and Scope

This guide is intended for Windows, UNIX, and Linux system administrators who will be deploying VAS and are interested in the following:

- More detailed explanations of how the VAS client components work
- Detailed explanations of how VAS works in multiple domain environments
- Detailed explanations of how VAS works with Active Directory Sites
- How to migrate existing Unix users and groups into Active Directory

For information on basic installation and configuration of Active Directory and the VAS client, refer to the Installation and Configuration Guide.

Conventions Used in this Guide

The following notation conventions are used throughout this guide:

- Directories and filenames appear in monofont. For example, */etc/pam.conf*.
- Executable names are bolded. For example, **vascd**.
- Specific file and packaging formats will appear in bold. For example, the **RPM** package.
- Shell commands appear in monofont. For example,

  ```
  # vastool configure pam
  ```

  Within text, commands are bolded for readability. For example, using **vastool** you can create users, delete users, and list user information.
- Menu items and buttons appear in bold. For example, click **Next**.
- Selecting a menu item is indicated as follows: **Programs -> Administrative Tools -> VAS ->Active Directory Users and Computers**

Chapter 1

VAS Features and Benefits

Deploying VAS provides the following features and benefits:

True Integration with Active Directory. Active Directory users can authenticate to Unix resources and Active Directory groups can be used to provide access control to Unix resources. No password or account syncronization is used. All Unix authentication identity management features operate in real time with changes made by administrators on Windows domain controllers.

All Authentication Uses Kerberos. VAS uses Kerberos v5, which is the native authentication protocol for Windows 2000 and Windows 2003. The use of Kerberos eliminates the need to send passwords or password hashes over the network in plain text. All password change requests are performed using Kerberos, and enforce Windows password policies established by the domain administrator. Using Kerberos also eliminates the need for the distribution of SSL certificates to Unix clients and modifying Active Directory to use SSL for LDAP security. All VAS LDAP communication is secured using Kerberos. Finally, VAS maintains compatibility with MIT-style Kerberos implementations and can be used with Unix applications that link with 3rd party Kerberos libraries.

A Persistent Client Cache. VAS is a scalable product that uses a persistent client side storage to cache frequently accessed user account information. Intelligent caching algorithms allow VAS to limit the amount of network traffic it uses and simplifies the complexity of LDAP searches for Active Directory. This design also allows for hundreds of concurrent Unix processes to authenticate and resolve Unix account information (UID, GID, etc) without overloading the Active Directory server with search requests. The persistant

cache also allows VAS to be configured to continue working even when it loses contact with the Active Directory server. The option to use disconnected operation make VAS an ideal solution for mobile Unix/Linux users or in environments here dial-up networking is used.

Simple Deployment. VAS is packaged using the native package formats for each supported Unix platform. Native packaging makes it easy for Unix administrators to install and manage their VAS installations using existing software administration tools. VAS also provides easy-to-use command line utilities such as **vastool join** and **vastool timesync** to automate complex configuration tasks like PAM configuration and time syncronization. VAS product installation and configuration only takes a few minutes and does not require in-depth knowledge of Kerberos, LDAP, or Unix authentication and identity subsystems.

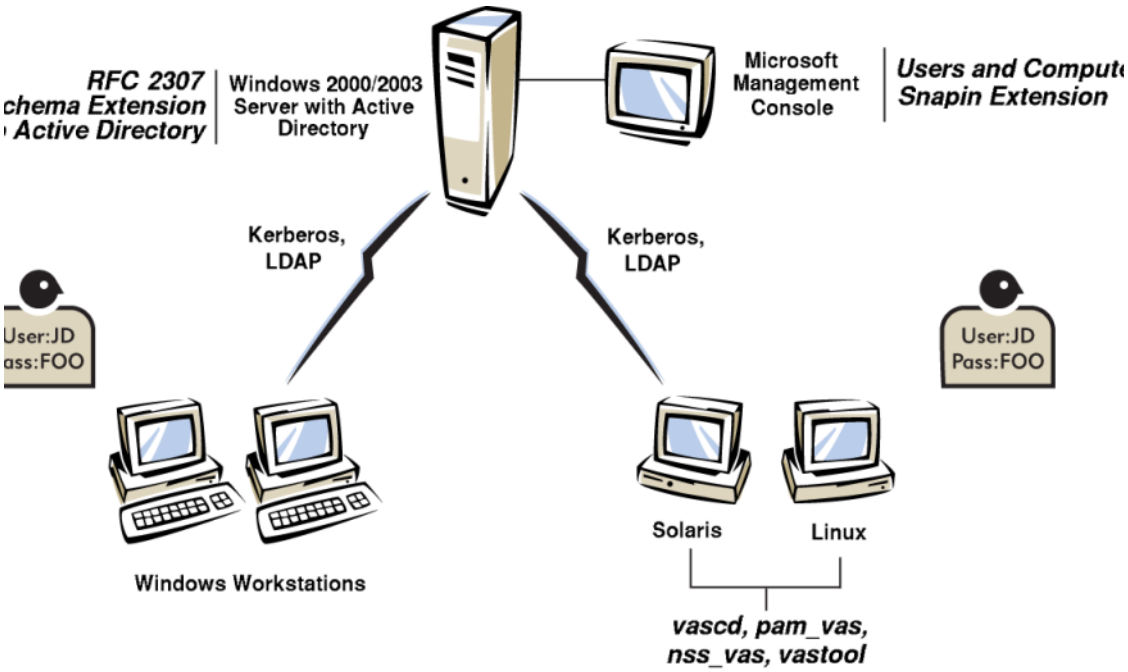
Integration with existing Unix utilites and applications. VAS has been designed to seamlessly integrate with the core Unix authentication subsystems (PAM and NSS) so that existing applications can take advantage of Active Directory integration without any modifications. For example, **Apache**, **OpenSSH**, **telnet**, and **ftp** all easily integrate with VAS and can authenticate Active Directory users immediately after the installation and configuration of VAS.

Chapter 2
VAS Components

This chapter provides a detailed explanation of each VAS software component. They are:

- The Active Directory Schema Extension. See page 4.
- The VAS Active Directory Users and Computers MMC Snapin Extension. See page 7.
- The **pam_vas** PAM module. See page 8.
- The **nss_vas** NSS module. See page 9.
- The **vascd** daemon. See “The vascd daemon” on page 11.
- The **vastool** command line utility. See page 12.
- The VAS Loadable Authentication Module for AIX. See page 13.

A graphical depiction of an Active Directory domain and the VAS components is shown in the figure below.



Active Directory Schema Extension

Active Directory Schema Concepts

The Active Directory schema defines what type of classes can be created and what attributes those classes can have in the directory. The base schema that is included when Active Directory is installed contains class and attribute definitions that are required by core Microsoft applications. However, to accomodate a wide range of applications, Microsoft Active Directory is designed to allow schema extensions so that additional class types and attributes can be stored in the directory.

Extending the Active Directory Schema is a procedure commonly performed when

installing directory enabled software that requires Active Directory to store types of information not provided for in the default Windows schema. In VAS's case, the default Active Directory schema does not contain any provisions for the storage of Unix user and group account information. In order for VAS to work, the Active Directory schema must be extended to allow storage of Unix and group identity information such as UID, GID, home directory, login shell, etc.

Supported Schema Extensions

In order to be as flexible as possible, VAS was designed to be compatible with multiple schemas. Supporting multiple schemas allows VAS to work in environments where the Active Directory schema might have already been extended for use with other products or experimental software.

VAS is fully compatible with the following schema extensions:

- The experimental RFC 2307 Schema recommendation
- The VAS Schema Extension
- The Microsoft SFU 3.0 Schema
- The Microsoft SFU 2.0 Schema

The experimental RFC 2307 Schema recommendation

RFC 2307 is a document that describes "An Approach for Using LDAP as a Network Information Service". The significance of RFC 2307 is that it introduces LDAP attributes that can be used to store Unix user and group account information. RFC 2307 is an experimental recommendation that has not yet been standardized by the IETF.

The VAS Schema Extension

The VAS Schema Extension follows the recommendations from RFC 2307 for the attributes used to store Unix user and group account information. The VAS Schema

Extension differs from RFC 2307 in that it does not include the *entire* experimental RFC 2307 schema extension.

VAS's use of the Kerberos protocol for authentication supercedes the RFC 2307 suggestion for the storage of "shadow password" information and crypted/hashed password attributes. Use of these attributes would duplicate functionality already accessable via the Kerberos protocol and would introduce significant security holes by making sensitive information accessible to anonymous or insecure users.

The VAS Schema Extension also omits RFC 2307's NIS Map specific attributes for storage of hosts, services, networks, etc. These attributes are unrelated to VAS's authentication and identity management solution. For more information see the NIS Migration section in the "Migration" chapter.

The Microsoft SFU 3.0 Schema

The SFU 3.0 schema provides attribute and class definitions to store Unix account information for users and groups. The VAS client will automatically detect this schema extension, and will be able to use the users and groups that have already been configured in Active Directory. You will not need to install any Windows components in this scenario or further extend the Active Directory schema.

However, the VAS Active Directory Users and Computers Snapin extension will not be able to manage SFU users and groups. This feature will be added in future versions of VAS.

It is possible to install the VAS schema alongside SFU 3.0. You will need to use the VAS Snapin extension to set the Unix attributes for all users and groups that were previously Unix enabled with the SFU Snapin extension.

The Microsoft SFU 2.0 Schema

The SFU 2.0 schema provides the same functionality as the SFU 3.0 schema, with some differences in attribute names. The VAS client can operate with SFU 2.0 the same way it does with SFU 3.0.

The VAS Schema Extension Utility

VAS provides a general use schema extension utility that allows the administrator to install the VAS schema extension on the Schema Master domain controller. The VAS Schema Extension Utility is located on the VAS installation media. For a complete description of how to extend the Active Directory schema, see the VAS Installation and Configuration Guide.

Global Catalog

The Windows Global Catalog is a database that contains a record of every object in the Active Directory forest. Each object in the Global Catalog only contains those attributes that have been enabled for Global Catalog storage. Windows requires that at least one domain controller in each domain act as a Global Catalog server. VAS supports the addition of Unix identity attributes to the global catalog. The use of Global Catalog servers allows replication of information across Active Directory forests to reduce search traffic across WAN links. For information on adding attributes to the Global Catalog, refer to your Active Directory documentation.

The VAS Active Directory Users and Computers Snapin Extension

The VAS Active Directory Users and Computers Snapin Extension is installed as part of the VAS.msi **MSI** file. The VAS Snapin extension extends the Users and Groups properties dialogs with property sheet tabs for managing user and group Unix account information. The VAS Snapin Extension can be installed on any administrative workstation that has the Active Directory Users and Computers Snapin installed.

The pam_vas PAM module

PAM Concepts

PAM stands for Pluggable Authentication Module, and is an API that allows the system administrator to configure authentication mechanisms rather than having authentication mechanisms hardcoded into the application. PAM is supported on Linux, HP-UX, AIX 5.2, and Solaris. Administrators can customize an application's authentication system by making changes to */etc/pam.conf* or an application specific file in the */etc/pam.d/* directory.

PAM modules are shared libraries that add support for a specific authentication mechanism. Unix platforms that support PAM normally have a PAM module called **pam_unix** for standard Unix authentication. **pam_vas** is the VAS PAM module that allows user logins to be authenticated against Windows Active Directory domains. **pam_vas** can be configured to provide the following features:

Kerberos-based User Authentication. Kerberos is the prefered authentication protocol in Windows 2000 and Windows 2003 domain environments. **pam_vas** allows PAM enabled applications to authentication users directly against the Windows domain and cache the appropriate Kerberos ticket credentials for use by other applications. **pam_vas** is designed to use Kerberos to enforce Windows account access controls and password policies.

Disconnected Authentication. **pam_vas** can be configured to allow Active Directory logins when Unix clients are disconnected from the network or when the Active Directory server is not available.

Automatic Home Directory Creation. Administrators can configure **pam_vas** to automatically create users' home directories if they do not exist at login time. The home directory is set up with the proper ownerships and permissions and is populated with the files from */etc/skel*.

UID Conflict Checking. It is possible to inadvertantly create UID conflicts between

Unix enabled Active Directory users and local system accounts stored in */etc/passwd*. UID conflicts create security holes where a local system user with the same UID as an Active Directory user could access that Active Directory user's files, and vice versa. **pam_vas** prevents this by not allowing Active Directory users to log in if they have a UID conflict *and* their UID is greater than 1000.

Machine Based Access Control. **pam_vas** allows administrators to selectively control which Active Directory users can access a Unix machine running VAS. Unix administrators can restrict access based on Active Directory group membership, Active Directory domain membership, and also for specific users.

Password Administration. **pam_vas** allows users to manage their Active Directory password. With **pam_vas**, password changes are immediately applicable to subsequent Unix and Windows logons. **pam_vas** enforces domain controller password policies so that password complexity and the frequency of forced password change can be configured by the Windows administrator.

For information on configuring the **pam_vas** module, see the **pam_vas** man page.

The nss_vas NSS module

NSS Concepts

Unix operating systems use various "databases" to obtain information about users, groups and so forth. Traditionally, these databases were stored as flat text files in the */etc* directory. For example, user information is stored in */etc/passwd* and group information is stored in */etc/group*. The Name Service Switch (NSS) is a subsystem built directly into the C runtime library (libc) and allows user and group identity information to be obtained from a variety of backend sources -- not just the text files in the */etc*

directory. System administrators can change the NSS configuration by modifying the */etc/nsswitch.conf* file.

NSS modules are shared libraries that add support for user and group identity information to be retrieved from a specific backend data source. **nss_vas** is the VAS NSS module that allows user and group identity information to be retrieved from Active Directory. In addition to allowing Unix identities to be managed from Active Directory, **nss_vas** provides the following important features:

Security. **nss_vas** differs from other LDAP NSS modules in that **nss_vas** never directly generates any LDAP traffic. The reason for this is that on Unix hosts, NSS modules can not be provided with any security context information that would allow authentication to the Active Directory server. Instead, NSS modules that generate LDAP traffic are limited to only being able to query "public" LDAP attributes or require that "guest" accounts to be created with a password that is stored in a world readable file on the Unix host.

For security reasons **nss_vas** only generates interprocess communication (IPC) requests to the **vascd** daemon (detailed later in this section). With **vascd**'s help, **nss_vas** can operate securely and efficiently.

Scalability. NSS modules are loaded into each process namespace separately. NSS modules that directly generate LDAP traffic will create a new LDAP connection for each new Unix process. Per-process LDAP connections and the amount of **nss_ldap** generated LDAP requests create a significant load for the Active Directory domain controllers.

To avoid per-process LDAP connections and excessive LDAP network traffic **nss_vas** uses interprocess communication (IPC) to **vascd** and cached data to handle NSS requests that would otherwise create scalability problems.

Disconnected Operation. Because of the frequency which which the NSS interfaces are called an NSS module cannot operate gracefully in a discon-

nected environment. Failure of an NSS module to operate efficiently can have a very serious negative affect on *all* processes running on the Unix host. If an NSS module blocks (waits for an LDAP connection to time out), the resulting problems can range from very sluggish operation to the complete failure of some Unix services.

With the help of **vascd**, **nss_vas** is able to quickly detect disconnected situations and continue to operate using the cached information.

The vascd daemon

vascd is the core Vintela Authentication Services client daemon. **vascd** must be running on Unix clients in order for VAS to operate correctly. When started, **vascd** uses Kerberos to authenticate to Windows Domain controllers using credentials that were established at the time that the computer was joined to the Domain. **vascd** then uses Kerberos encrypted LDAP sessions to query and cache Unix user and group information that is necessary for the scalable and secure operation of the **nss_vas** and **pam_vas** modules.

vascd provides several important features:

Secure Access Control. Because of the way that PAM and NSS subsystems operate, most LDAP-based Unix account management solutions require that anonymous or public access to Unix account properties be allowed. Vintela Authentication Services does not require a reduction in access control policy for Unix account properties. Since **vascd** authenticates as the Active Directory domain computer created when the Unix host was joined to the domain, there is no need to provide Anonymous access to any information used by the VAS client.

Secure LDAP without SSL. Once the Unix host is joined to the domain, **vascd** is able to authenticate as a domain computer and encrypt LDAP communications with domain controllers using a Kerberos session key.

The Vintela Authentication Services client never generates any plain-text LDAP traffic, and does not require the administrative overhead of SSL certificate distribution to Unix clients or running Active Directory with SSL enabled.

Scalability. Because of the way that PAM and NSS subsystems operate, most LDAP based Unix account management solutions generate excessive numbers of LDAP connections and LDAP search requests. This results in dramatically increased network traffic and load on the Windows Active Directory domain controllers. **vascd** establishes a single connection that is used to proxy all information requests for processes that call the NSS interfaces. At the same time, **vascd** is able to perform intelligent caching of frequently used information so that LDAP traffic is reduced to the absolute minimum.

Disconnected Operation. **vascd** maintains a persistent cache of frequently used information. This makes it possible for the system to continue to operate in environments where the network connection to the Active Directory server is unreliable or completely unavailable. Disconnected operation is particularly useful for dialup and laptop users that are frequently not connected to the network.

The vastool command line utility

vastool is a script-friendly command line utility that exposes a wide range of functionality to the Unix/Linux system administrator. The following table lists **vastool** commands and functionality:

attrs	List an Active Directory object's attributes
configure	Update configuration files to use the VAS components

create	Create a user, group, computer object, or service
delete	Delete a user, group, computer, service, or NIS Map
flush	Flush cached client daemon information
group	Modify group membership
isvas	Check to see if the specified user or group belongs to VAS
join	Join the host to the domain
kinit	Obtain tickets for principals and services
klist	List cached tickets
kdestroy	Destroy (erase) cached tickets
license	Install your VAS license
list	List users or groups and their attributes
nis-import	Load NIS Maps into the directory
nss	Call raw NSS function interfaces
passwd	Change your password or reset another user's password
realms	Update or display cached domain, site, and service information
search	Perform LDAP searches
service	Create/Modify service principals
timesync	syncronize the system clock with the domain
unconfigure	Update configuration files to not use the VAS components
unjoin	Remove the host from the domain
ypcat	Provides functionality similar to ypcat for imported NIS maps

The VAS Loadable Authentication Module for AIX

Most versions of Unix use the NSS and PAM subsystems for account information lookups and authentication. However, AIX uses an API called LAM (Loadable Authentication Module) that combines the the functionality of NSS and PAM. VAS provides an AIX LAM module, **VAS**, which combines the functionality of **pam_vas** and **nss_vas**.

AIX LAM Module Concepts

LAM modules are shared libraries that the system's **libc** uses based on the configurations in */usr/lib/security/methods.cfg* and */etc/security/user*. The *methods.cfg* file lists all of the available authentication modules. In */etc/security/user* different users can be configured to use different authentication modules. System-wide defaults can be set for the "default" user; including a list of LAM modules to try when authenticating users.

Each LAM module must provide methods for authenticating users, changing user passwords, and looking up user and group account information. AIX by default includes LAM modules for DCE and NIS. AIX has built in functionality for authenticating local and NIS users when using the **compat** setting in */etc/security/user*.

The VAS LAM module

The **VAS** LAM module is installed in */opt/vas/lib/security*. It provides the same functionality as the **pam_vas** PAM module for authentication. It also provides the same functionality as the **nss_vas** NSS module for user and group lookups. The VAS LAM module is added to the default user's authentication mechanism as part of joining VAS clients to the Active Directory domain.

Chapter 3

Unix Hosts in Active Directory

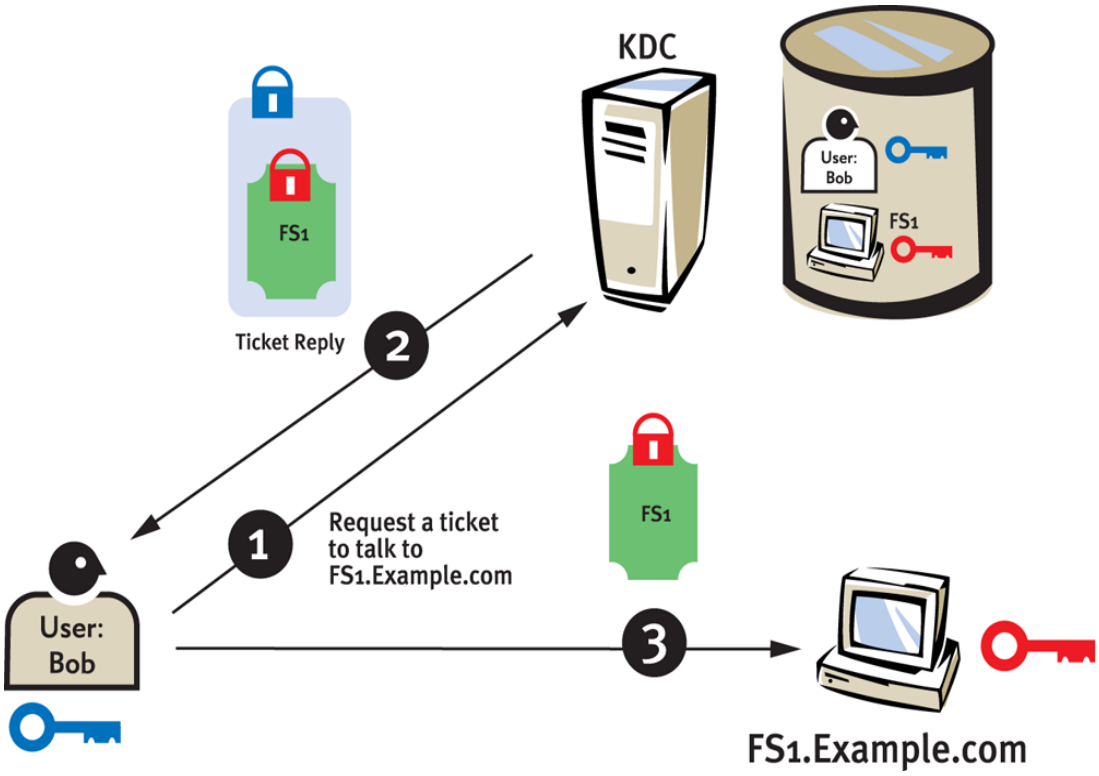
The VAS product allows Unix hosts to be intuitively represented in Active Directory as computer objects. This chapter addresses the topics that are relevant to the creation of Unix computer objects and the role Unix computer objects play in the VAS authentication and identity management solution:

Kerberos and Active Directory

Kerberos Concepts

The name Kerberos comes from Greek mythology; it is the three-headed dog that guarded the entrance to Hades. The Kerberos protocol originated at the Massachusetts Institute of Technology and was later formalized as a network security standard in RFC 1510.

The Kerberos network authentication service provides a means for "principals" to verify the identity of other principals. A "principal" is any entity that provides or consumes resources, (for example users, workstations, servers, and even individual service processes that run on network servers). Principals authenticate each other using tickets that are issued by the Kerberos Key Distribution Center (KDC). Tickets are encrypted using secret keys that are only known to the principals themselves and the KDC. Therefore, one principal can validate the authenticity of another if they can present a ticket encrypted by the correct secret key. An illustration of a simple ticket exchange is shown below.

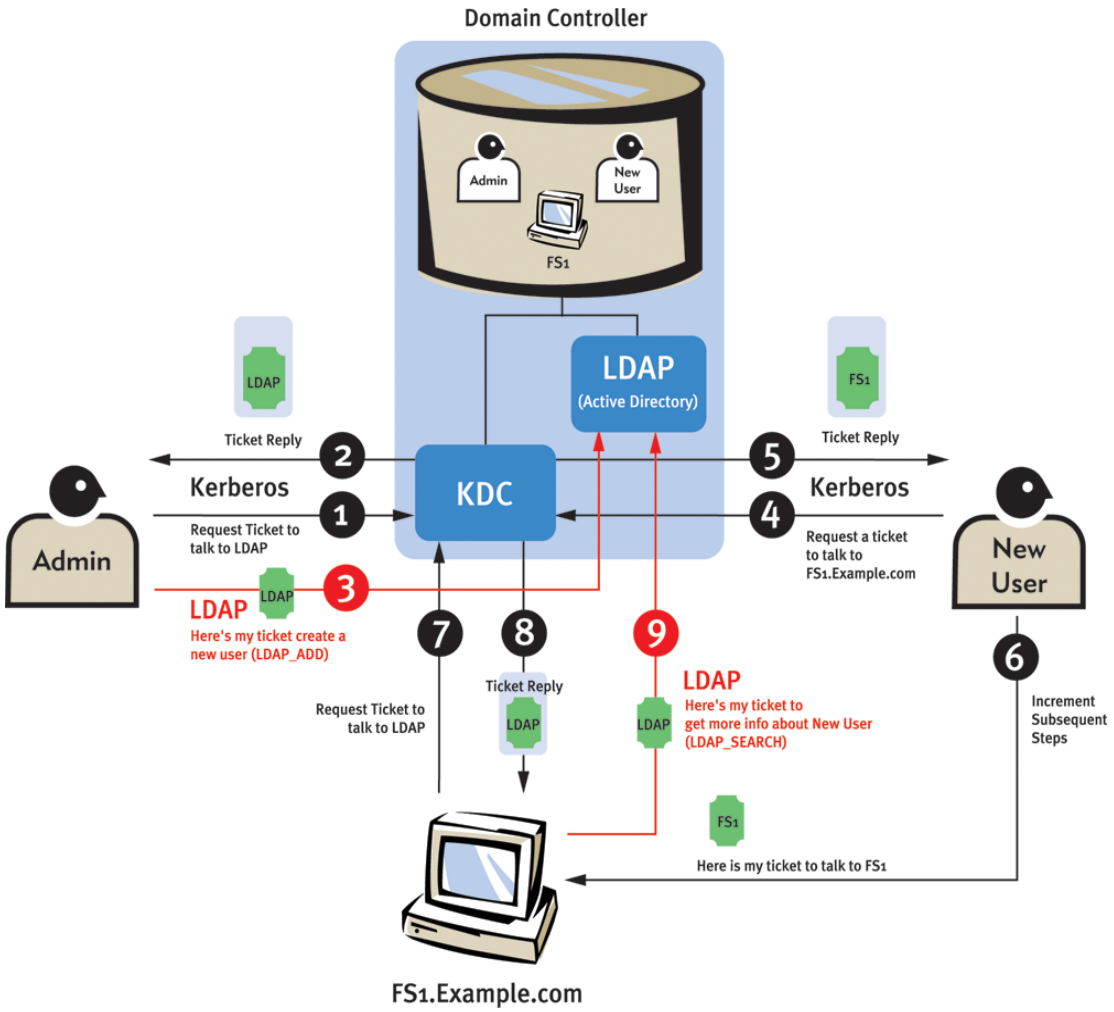


Kerberos and Windows

The Kerberos version 5 authentication protocol is the default network authentication protocol for Windows 2000, Windows XP and Windows 2003. The Windows Kerberos implementation is fully compatible with RFC 1510 and is much more refined than what is generally available from the MIT reference implementation. In fact, many users are not even aware that Kerberos provides nearly all of the security underpinnings for Windows domains.

There is a very close symbiotic relationship between Kerberos and Active Directory on a Windows Domain controller. On a domain controller, the KDC and Active Directory actually share some of same account information data. This is how a domain controller can act as both a KDC and an LDAP server. This means that user, computer, and service

accounts created in Active Directory can be used as Kerberos authentication principals. This also means that an authentication and identity management solution must use Kerberos and LDAP together as shown in the figure below.

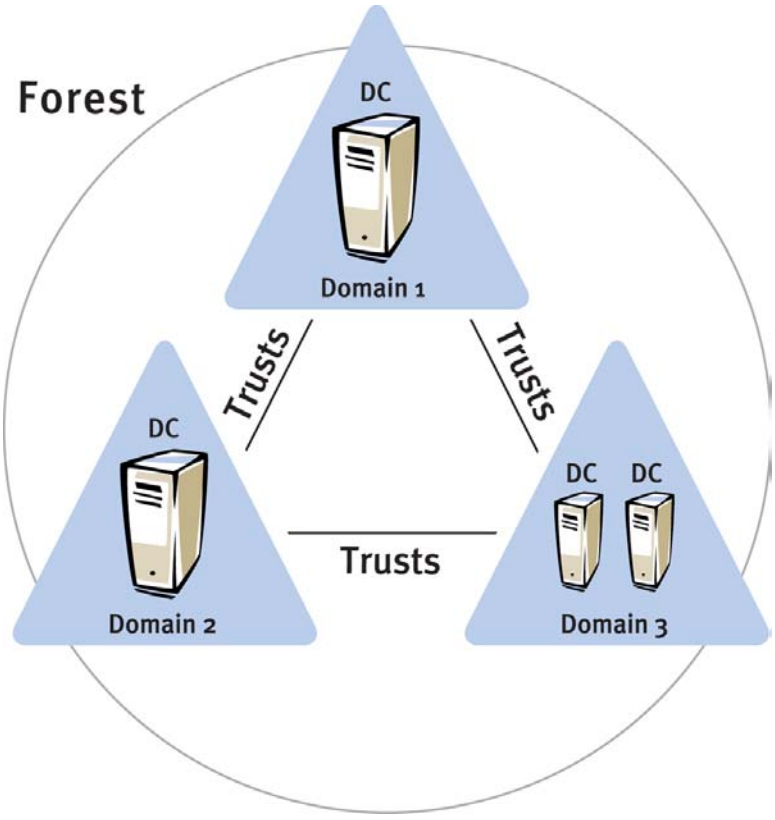


The purpose of VAS is to provide the software components that allow Unix hosts to authenticate user logins in the same way that Windows computers authenticate user logins. With VAS, the Unix machine is joined to the domain after which "Unix enabled" Active Directory users can login to the Unix host using their Active Directory username

and password. This is accomplished without disturbing any of the core Windows domain controller services. As far as the domain controller is concerned, a Unix host with VAS installed authenticates user logins the same way that a Windows computer does.

Multiple Domain Environments

Large organizations often deploy multiple domains with many domain controllers. Trusts are established between domains so that Kerberos tickets issued by one domain controller are honored by domain controllers of other domains. Multiple domains tied together by trust relationships are called a Forest as shown in the figure below.



Unix participation in multiple domain environments (forests) adds a layer of complexity that VAS fully supports. For example, if a user in one domain would like to logon to a Unix host in another domain, VAS is able to request an initial ticket from the appropriate domain controller in a domain where the user resides.

Active Directory Sites

Organizations with a large number of geographically separate sites have an alternative to deploying multiple domains -- Active Directory Sites. An Active Directory Site is a well-connected group of computers that share the same subnet. Often, these computers are all located at one physical location. Each Active Directory Site will have a domain controller(s) that should be used by all of the clients for authentication. This reduces the amount of WAN traffic between geographical locations, since all of the necessary authentication information in the directory is replicated between the domain controllers. All clients in a site then use the domain controller(s) at their site for authentication.

As with forests, adding Unix clients into a site adds more complexity that VAS automatically handles by detecting what domain controllers are for a given site and giving priority to those servers.

Joining the Active Directory Domain

The first and most important VAS configuration step is joining the Unix host to a Windows domain. As a member of the domain, the computer itself is able to communicate securely with Active Directory and act as a Kerberos authentication principal for validating user logons.

Though the **vastool join** command automates the entire process of joining the domain, understanding the details of each step of the join process will help the administrator to more quickly diagnose common configuration problems and recommend deployment specific-solutions.

The following is a list of the steps performed by the **vastool join** command:

1. Check for clock syncronization.
2. Discover Domains and Sites.
3. Create a computer object for the Unix host.
4. Create a key for the computer object.
5. Modify the system NSS configuration.
6. Modify the system authentication configuration (PAM).
7. Start **vascd**.
8. Load the VAS user and group cache.

Domain, Site, and Service Discovery

VAS discovers domains, domain controllers, sites, and services using DNS service record lookups and specialized LDAP queries. As domain controllers are added to the forest, the network DNS servers are updated with service records that allow VAS to locate the domain controllers through DNS queries. Once an LDAP service entry has been found, VAS can query the Active Directory to find additional information about the domains and domain controllers in the forest as well as the site to which each domain controller belongs.

When joining an Unix machine to the domain it is critical that the Active Directory DNS service entries are visible to the DNS name servers that are referenced in the Unix machine's resolver configuration (*/etc/resolv.conf*). If there is a DNS misconfiguration, the Unix machine will not be able to join the domain. The following error message is an indication of a DNS misconfiguration:

> ERROR: The servers for the example.com domain could not be detected. Please check your DNS configuration and ensure that your DNS server(s) have been properly configured for use with Active Directory.

The information about domains, domain controllers, sites, and services is cached so that it can be used later to allow VAS to contact the appropriate domain controllers for cross domain authentication, and when possible use domain controllers in the local site. The **vastool realms** command is used to inspect and maintain the domain controllers, domains, sites, and services cache. The vastool command **realms** refers to Kerberos realms which are equivalent to Windows Domains.

Time Synchronization

Kerberos is a time sensitive protocol that requires the system clock of a client machine to be roughly synchronized with the system clock of the Windows Domain Controller. Time synchronization between domain controllers and Windows clients is handled automatically. To ensure time synchronization Unix hosts may require additional configuration. If there is a significant difference between the system clock of the client machine and the Windows Domain Controller the following error message will be displayed:

```
Could not authenticate, error = Clock skew too great.
```

There are several ways to correct a clock skew on the Unix host. The easiest way is to use the **vastool timesync** command which will automatically synchronize the host's system clock to within 1 second of the Windows domain controller. The **vascd** daemon also contains a Simple Network Time Protocol (SNTP) implementation that will continue to keep the host's system clock roughly synchronized with the domain time.

There are situations where it is not advisable to use the **vastool timesync**. If the host is already configured to use Network Time Protocol (**ntpd**) to synchronize the system clock with corporate time service then the time *should* be synchronized already. If a clock skew occurs on a system running **ntpd** then there is either an error in the NTP configuration (*/etc/ntp.conf*) or the domain time itself has become unsynchronized.

Another situation that requires special attention is when the Unix host is running software that is time sensitive. Certain transactional databases and distributed systems react

badly if the system clock is changed abruptly. If the Unix host is running time sensitive software NTP should be used instead of the **vastool timesync** to synchronize time.

For more information on NTP consult your Unix OS documentation.

Computer Objects

Creating Computer Objects

The creation of a computer object in Active Directory is a very important step in joining the domain. The computer object identifies the Unix host in the domain and provides information used by the domain controller to issue Kerberos tickets when a user logs on to the host.

Computer Object Names and Hostnames

By default, the name of the computer object that is created when joining the domain is derived from the hostname of the Unix host, which in most cases is derived from the host's DNS name. There are some situations when the hostname may not match the DNS name or when the hostname changes depending on the IP address assigned by DCHP. In these cases the **-n** option should be used with **vastool join** to specify the name of the computer object.

If you are interested in running Kerberized services such as **sshd** , **telnetd**, or **ftpd**, it is highly recommended that the hostname, DNS name, and computer object name match. The reason is that the DNS name will be used by Kerberized client applications to determine the service name in a Kerberos ticket request. If the DNS name does not match information stored when the computer object was created, the domain controller will not be able to issue the service ticket.

Delegating Computer Creation Privileges

In order to create a computer object, a user must be an administrator or have been delegated permissions to create computer objects. When using the **vastool join** command the **-u** option must be used to specify a user with the appropriate permissions. By default, users in the Domain Admins group have permissions to create computer objects.

The built in *Administrator* account can not be used with the vastool **-u** option to join Unix hosts to the domain.

For more information on privilege delegation consult your Active Directory documentation.

Creating Computers in specific containers (OUs)

By default, computer objects are created in the *Computers* container of the Active Directory tree. To create the computer object in a different container, use the **-c** option when running **vastool join**.

Deleting Computer Objects

Removing a computer object from Active Directory using the Windows Active Directory Users and Computers utility will cause all subsequent Active Directory user logins to the Unix host to fail until the computer object is recreated using the **vastool join**.

The **vastool unjoin** command is the preferred method for removing Unix computer objects. Using **vastool unjoin** will cleanly delete the computer object in Active Directory, delete the cached user and group information, remove VAS configuration information in the Unix NSS and authentication configuration files, and stop the **vascd** daemon.

vastool unjoin must be run as root, and a user with the administrative privilege to delete the computer object must be specified with the **-u**.

Moving Computer Objects

When using the Windows Active Directory Users and Computers utility, Unix computer objects can be moved (drag-n-drop) from one container (OU) to another as long as the source and destination containers (OU's) are in the same domain. Unix Computer objects can not be moved across domain boundaries with Windows utilities such as **movetree** and **netdom**. Instead, the computer object should be removed using **vastool unjoin** and re-joined to the new domain.

VAS Keytab Files

A keytab file stores Kerberos keys for computer and service principals. The creation of a keytab is a complex process that involves setting the computer (or service) object password, generating the appropriate keys, and checking for key version number correlation. With VAS it is not necessary to manually export a keytab on the domain and import it on the Unix host. The VAS product performs the entire keytab generation process automatically when joining the domain or when creating service principals in Active Directory.

VAS keytab files are created in */etc/opt/vas* directory. Each keytab file is named according to the service that uses it. For example, the *host* principal keys are stored in the *VAS/host.keytab* file. VAS keytab files are stored using the "standard" MIT-style and may be used by third party applications.

If *host.keytab* is compromised by unauthorized root access on the Unix system, then the password for the associated computer object should be assumed to be compromised as well. You can reset the computer object's password at any time by running **vastool create host/** which generates a new random password and rebuilds the keytab file. Another option is to delete the computer object and recreate it.

Security Considerations

The default permissions for a computer object restrict the computer from accessing and modifying sensitive data in Active Directory. The schema extensions are carefully

designed to allow computers with default permissions to access only the Unix account data that is absolutely necessary for the normal operation of the **vascd** daemon. We recommend that administrators not modify the default permissions for the computer object to make them either more or less restrictive. Changing the computer object permissions could disrupt normal operation or create a security liability that might result in the compromise of sensitive data.

NSS Configuration

The NSS Configuration File

/etc/nsswitch.conf contains the configuration used by the Name Service Switch (NSS) subsystem to determine which NSS modules should be used to obtain information about users, groups, hosts, etc. Each line of the */etc/nsswitch.conf* represents a configuration setting for the particular database.

The following is a portion of a sample */etc/nsswitch.conf* file:

```
passwd:   files nis
group:    files nis
```

The order that modules are listed on each line of */etc/nsswitch.conf* is important. In the example above NSS would first use */etc/passwd* and */etc/group* to look up user and group information. If the desired user or group information exists in */etc/passwd* or */etc/group* then it will be used instead of identical information that may be stored in NIS. NIS will only be used to look up user or group information if the information can not be found in */etc/passwd* or */etc/group*.

Modifying the NSS configuration

During **vastool join** the *passwd* and *group* lines of */etc/nsswitch.conf* are automatically

modified to include the *vas* NSS module. The following is an example of what the *passwd* and *group* lines will look like after a Unix host has been joined to the domain.

```
passwd:   files vas nis
group:    files vas nis
```

After a Unix host has been joined to the domain, the Name Service Switch will first use */etc/passwd* and */etc/group* to look up user and group information. If the desired user or group information exists in */etc/passwd* or */etc/group* then it will be used, otherwise, the user and group information is obtained from Active Directory and lastly from NIS.

There are situations, such as troubleshooting NSS problems, when it is useful to change the NSS configuration so that VAS is not being used. This can be accomplished by editing the */etc/nsswitch.conf* directly or by running **vastool unconfigure nss**. To restore the configuration run **vastool configure nss**.

Restarting services when the NSS configuration changes

Service daemons should be restarted after changing the */etc/nsswitch.conf* file. A daemon restart is necessary so that the NSS instance for the daemon process will re-read the */etc/nsswitch.conf* file and load the appropriate NSS modules. Consult your Unix OS documentation for more information on restarting services.

Using nscd With VAS

nscd is a rudimentary caching daemon that can increase the efficency of the Name Service Switch. **nscd** caches results supplied by NSS modules. This cache is used instead of calling the NSS modules for a specified period of time. After a configurable timeout, the cached results are flushed and NSS again calls the NSS modules directly to load the cache.

VAS uses it's own caching mechanisms that are much more efficient at caching information retrieved from Active Directory. It is possible to use VAS and **nscd** together, but

doing so often produces unpredictable results. Therefore, the default behavior for **vastool join** and **vastool configure nss** is to modify */etc/nscd.conf* to disable **nscd** caching of *passwd* and *group* data.

To verify that **nscd** caching of *passwd* and *group* data is turned off, inspect the */etc/nscd.conf* file. Ensure that all *passwd* and *group* entries have been removed with comments except for the *enable-cache* entry which would be set to *no*.

```
   enable-cache            passwd          no
#  positive-time-to-live   passwd          600
#  negative-time-to-live   passwd          20
#  suggested-size          passwd          211
#  check-files             passwd          yes
   enable-cache            group           no
#  positive-time-to-live   group           3600
#  negative-time-to-live   group           60
#  suggested-size          group           211
#  check-files             group           yes
```

Instead of **nscd**, HP-UX uses a daemon called **pwgrd** whose configuration is stored in */etc/rc.config.d/pwgr* . As with **nscd**, VAS disables the **pwgrd** caching of user and group data when joining the domain or configuring NSS.

Advanced nss_vas Concepts

Case sensitivity of user and group names

In some environments the user and group names in Active Directory will be all upper case. Normally user and group names on Unix systems are lowercase. It is possible to have **nss_vas** explicitly change all user and group names to their lowercase versions. To enable the lowercase behavior, create a file owned by root at */etc/opt/vas/.vas_nss_lowercase*. If this file exists, then all user and group names will be lowercased. To disable this behavior, simply delete */etc/opt/vas/.vas_nss_lowercase*. You may need to restart the processes using **nss_vas** to apply the change.

Case sensitivity of User Logon Names

User logon names in Active Directory are case insensitive. For example, a user with a logon name of JOHND can logon to a Windows workstation with the username of johnd. Likewise, with VAS, the user logon name used when logging on to Unix hosts is also case insensitive.

User Logon Names for Cross Domain Login

When a user logs in to a VAS client that is in another domain, the user must log in with their full user logon name. This will be in the form of joe@example.com, where example.com is the domain that the user joe is in. This allows the VAS client to determine where to get joe's user account information and also determine what domain to authenticate joe against.

When doing cross domain logins with ssh, it is necessary to specify the user name with the **-l** option as follows:

```
ssh -l joe@example.com server.sub.example.com
```

The HP-UX telnet client does not accept standard user login names with the "@domain" component as login names. You will have to escape the '@' character in order to perform cross domain logins. For example, you would specify your whole user logon name as `joe\@example.com` in order for the HP-UX telnet daemon to accept you.

PAM Configuration

The PAM Configuration File

PAM stands for Pluggable Authentication Module and is an API that allows the system administrator to configure authentication mechanisms rather than having them hard-coded into applications. PAM is controlled by configuration settings in the *pam.conf* file

or by individual (service-specific) files in the */etc/pam.d* directory. The PAM configuration allows PAM modules to be "stacked" and combined to provide flexible and powerful authentication configurations.

The **pam_vas** module should be configured to be the first module in the stack. **pam_vas** returns an *IGNORE* result when the user being authenticated is not an Active Directory user. This gives other standard PAM modules in the stack (such as **pam_unix**) a chance to authenticate the user.

When joining the domain, **vastool** automatically modifies the PAM configuration to add *auth*, *account*, *session*,and *password* entries for **pam_vas.so** as part of the stack for all PAM services. On Linux the entries are made with new explicit *[value=action]* control-flags. On Solaris, HPUX, and Solaris the entries are made as *sufficient*.

To verify that the PAM configuration has been modified correctly to use VAS, inspect the *pam.conf* file or one of the (service specific) PAM configuration files in the */etc/pam.d* directory. Check to ensure that a an entry for **pam_vas.so** exists as the first entry in the stack for each service that should be using VAS to authenticate users.

Refer to the system administration documentation for your operating system concerning PAM before performing any customizations beyond the changes made by the **vastool configure pam** commands described in in this section.

Using vastool to modify PAM configuration

The **vastool configure pam** command is particularly useful when installing new software on a Unix host that has already been joined to the domain. When joining the domain, **vastool** only configures PAM for *existing* services. Therefore, after installing new service software, you must run **vastool configure pam** to add the correct **pam_vas.so** entries for the newly installed service. The following is an example of how **vastool** can be used to modify the PAM configuration for a newly installed service (**sshd**):

```
# /opt/vas/bin/vastool configure pam sshd
```

There are situations, such as when troubleshooting problems, when it is useful to change the PAM configuration so that VAS is not being used. This can be accomplished

by editing the PAM configuration files directly and commenting out the **pam_vas.so** entries. An easier method is to simply run the **vastool unconfigure pam**. The following command removes the **pam_vas.so** entries from the **sshd** service.

```
# /opt/vas/bin/vastool unconfigure pam sshd
```

If you omit the service name to unconfigure, **vastool unconfigure pam** will remove **pam_vas.so** entries from *all* services as shown below:

```
# /opt/vas/bin/vastool unconfigure pam
```

Restarting services for PAM configuration changes

Service daemons should be restarted after any modification to their PAM configuration. A daemon restart is necessary so that the PAM library calls will re-read the new PAM configuration. Service daemons can be restarted individually or all at once by cycling **init** run-levels.

Advanced pam_vas Concepts

Kerberos Ticket Caches

The **pam_vas** module uses the Kerberos protocol to authenticate users against Active Directory. The Kerberos protocol allows users to obtain a Ticket Granting Ticket or TGT that can then be used to obtain other tickets to authenticate to services. Once the TGT has been obtained it can be used as a single sign on mechanism that does not make users repeatedly enter in their password.

By default, when a user establishes a login session via a service configured to use the **pam_vas** module, they will have a ticket cache stored in their home directory with the name *.krb5cc*. In situations where the **pam_vas** module cannot securely create the credentials cache file in a user's home directory, it uses a default location of */tmp/krb5cc_xxx* where *xxx* is the user's UID. The **KRB5CCNAME** environment variable is also set to allow other Kerberos-aware applications to locate the ticket cache created by

the **pam_vas** module.

The tickets in the ticket cache do not contain the users password, but they could be used in a brute force attack to determine passwords, much like an attacker could use the password hashes in /etc/shadow if those were publicly available. To protect the ticket cache, the *$HOME/.krb5cc* file is created with permissions of **0600** (readable and writable only by the file owner) and owned by the user. This file should not be moved or modified by the user. To further reduce the possiblity of brute force attacks, the ticket cache is removed by the **pam_vas** module when the user terminates the logon session.

User Home Directory Creation

By default **pam_vas** creates users' home directories if they do not exist. The home directories are created with the appropriate permissions of **0700** (readable, writable, and executable only by the owner of the directory) and owned by the user. The files in */etc/skel* are also copied into the new home directory. **pam_vas** can only create home directories on local filesystems.

Automatic creation of home directories can be disabled by removing the **create_homedir** option from the *auth* entries in the PAM configuration for the desired logon service. Disabling automatic creation of home directories may be useful in environments where home directories are stored on network file servers.

To disable automatic home directory creation, modify the *auth* line to remove the **create_homedir** option:

```
auth [ignore=ignore success=done default=die] \
   /opt/vas/lib/security/pam_vas.so get_tgt create_homedir
```

The modified entry should look like the following:

```
auth [ignore=ignore success=done default=die] \
    /opt/vas/lib/security/pam_vas.so get_tgt
```

Using pam_vas For Authentication Only

pam_vas can be optimized for use with services that do not provide shell login functionality. Examples of non-shell services would be **ftp**, **imap**, and **pop**. In situations where **pam_vas** is being used with non-shell services you should consider removing the **get_tgt** option from the *auth* entry for pam_vas. This reduces the amount of Kerberos network traffic that has to be performed, and does not reduce functionality because most non-shell logon services will not need to use the TGT for subsequent authentications anyway.

To optimize a non-shell service, modify the *auth* line to remove both **create_homedir** and **get_tgt** options:

```
auth [ignore=ignore success=done default=die] \
   /opt/vas/lib/security/pam_vas.so get_tgt create_homedir
```

The modified entry should look like the following:

```
auth [ignore=ignore success=done default=die] \
    /opt/vas/lib/security/pam_vas.so
```

Disconnected Authentication

pam_vas allows users to continue to use their Active Directory passwords to authenticate even when the network connection to the Active Directory server is disrupted. Users must login at least once while in the connected state in order to be able to log in when the system is disconnected.

During a normal connected authentication operation, **pam_vas** caches a salted SHA-1 hash of the user's password. When the system is disconnected, **pam_vas** allows logins if the supplied password matches the cached password hash for the given user. User password hashes are cached in a secure file in */var/state/vas/authcache/authcache.vdb*, which has permissions of 0600 (readable and writable only by the owner of the file) and is owned by root. Though SHA-1 is a strong hash algorithm it is important to protect the *authcache.vdb* file in the same way as you would protect */etc/shadow*.

On some systems and services it may be preferable to disable caching of usernames and password hashes. To do this, add the **no_disconnected** option to the *auth* entries for **pam_vas** for all services that should not cache users' passwords.

pam_vas and Password Management

Windows allows the configuration of a password policy to force users to change passwords often, to select strong passwords, or to force password change at the next login after resetting the password. The **pam_vas** module also supports the full range of Windows password policy features.

When users log in and their password is expired, **pam_vas** prompts for the old password, and a new password. However, not all services use the PAM interface correctly to support interactive password change prompts. For example, KDE's **kdm** does not process the PAM requests from **pam_vas** correctly, but GNOME's **gdm** and the system **login** prompt do.

Debugging PAM Problems

If you experience problems authenticating Active Directory users using the **pam_vas** module, it is often helpful to enable verbose debug logging with the **debug** option. The **debug** option can be appended to any **pam_vas** entry. The debug output is logged to the authentication system log through syslog. Check your */etc/syslog.conf* to determine where authentication messages are logged. On RedHat Linux, authentication messages are logged to */var/log/secure* by default.

On Solaris, there is no default syslog setup for auth messages. If you do not see verbose **pam_vas** messages in the system log you may need to enable syslog for auth messages, by adding the following lines to */etc/syslog.conf*, and then restarting **syslogd** (make sure that you use *tabs* as a column separator):

```
auth.alert      /dev/console
auth.crit       root
auth.debug      /var/log/auth
```

AIX Configuration

AIX supports the standard Unix functions for user and group information lookups. However, it does not support NSS in the same way that most other Unix versions do. On AIX there is no */etc/nsswitch.conf* file or support for NSS modules. AIX has its own system and API's for authentication and user account information lookup, and its own configuration files.

AIX Configuration Files

AIX has two main files that are modified when joining the machine to the Windows domain. The first is */usr/lib/security/methods.cfg*. which lists the available authentication modules on the system. **vastool join** automatically adds an entry for the VAS authentication module.

Once the VAS authentication module has been added to the list of available modules, the VAS authentication module must be added to the list of modules used when authenticating users found in */etc/security/user*. In */etc/security/user* AIX allows you to configure the authentication mechanisms for specific users or a default for all other users. **vastool join** automatically configures the "default" user to use both the "compat" authentication module and the VAS authentication module.

The */etc/security/user* file is very complicated. Refer to your AIX documentation for information on the file format before changing any of the defaults that are set by the **vastool join** command.

Account Information Functions

The VAS AIX module provides the standard functions for looking up user account information and group information. The VAS AIX module works in the same way that **nss_vas** does. When requests for information come, IPC messages are sent to **vascd**. **vascd** updates the accounts cache. When the cache has been updated, the VAS AIX module reads account information directly from the cache.

Case sensitivity of User and Group Names

The VAS AIX module supports the same option as **nss_vas** to explicitly change all group and user names to lower case. This option is turned on by creating a file at */etc/opt/vas/.vas_nss_lowercase* that is owned by the root user.

Enabling debug messages

It is possible to enable debugging messages for the account information lookups by creating the */etc/opt/vas/.vasaix_nss_debug* file which must be owned by the root user. When this file exists, then "auth" messages will be logged to **syslogd**.

Authentication Functions

By default, AIX applications use the `authenticate` function to authenticate users. The VAS AIX module implements this function to authenticate users against the Windows domain controller using Kerberos. The VAS `authenticate` function supports all the features of **pam_vas**. The only limitation is that the options cannot be configured or turned off the way they can in **pam_vas**.

AIX 5.1 and 5.2 each support the PAM API. However, none of applications included in the base OS installation use PAM for authentication. **pam_vas** is included so that applications that have been PAM enabled can use its functionality. You must explicitly configure PAM by using the **vastool configure pam** command, since PAM is not configured as part of **vastool join** on AIX.

The AIX Loadable Authentication Module does not provide an interface for cleaning up user login sessions. This means that the VAS AIX module cannot delete the user's Kerberos ticket cache when they logout. We recommend that you use the **vastool kdestroy** command to cleanup user ticket caches. You can run **vastool kdestroy** from a user's .bash_logout script, if they are using the **bash** shell. Consult your OS documentation for inforation on other logout scripts for the shells your users use.

Enabling Debugging

It is possible to enable debugging messages for the account information lookups by creating the */etc/opt/vas/.vasaix_auth_debug* file which must be owned by the root user. When this file exists, then "auth" messages will be logged to **syslogd** similar to the way that "auth" messages are logged to **syslogd** by **pam_vas** when the **debug** option is used.

The vascd Daemon

The **vascd** daemon (or process) must be started on Unix workstations in order for VAS to operate correctly. **vascd** plays an important role in providing many of the security, scalability, and stability features of VAS. **vascd** acts as a proxy for requests for information that is stored in Active Directory. **vascd** provides the requested information either from a cache, or by contacting the Active Directory server. When obtaining information from Active Directory, **vascd** authenticates as the computer using the credentials that were established at the time that the computer object was created in the Active Directory domain.

The vascd Data Cache

In order to minimize network traffic and the load on the Active Directory server, **vascd** aggressively caches data that is retrieved from Active Directory so that subsequent requests can be satisfied from the cache without having to generate LDAP traffic. The cache is only updated when it is determined that a change has been made in the directory or when a new user logs in. Nevertheless, because of the way that the NSS subsystem is called it is not uncommon for hundreds of requests for user and group information to be generated in a matter of seconds. Therefore, in order to further reduce the load on the network and on the directory, **vascd** enforces a "blackout period" during which all NSS-initiated requests are resolved from the cache.

By default, the "blackout period" is set to a value of 10 minutes. This means that changes to Unix Account information may take up to 10 minutes (by default), to become visible through the Name Service Switch to VAS clients. In other words, there

are two events that will force **vascd** to query Active Directory for new information:

1. A user logs in

2. The "blackout period" expires

Until one of the events listed above, logged-in users and existing processes will not see newly added or modified users. For environments where changes must take affect faster, change the "blackout period" by modifying or adding the **update-interval** setting to */etc/opt/vas/vas.conf*. The following change to */etc/opt/vas/vas.conf* specifies the length of the "blackout period" to be 300 seconds:

```
[vascd]
update-interval = 300
```

As a general rule, decreasing the **update-interval** results in additional network traffic (depending on the number of Unix hosts and their use). In small installations (less than 100 hosts or less than 100 users) the blackout period can be safely reduced. In larger installations it is recommended that the blackout period be left at the default value or increased to 30 minutes or 1 hour. Regardless of the blackout period, the administrator can force vascd to update the cache immediately by signaling **vascd** with SIGHUP, using the vascd init script to restart **vascd**, or by executing a **vastool flush**. Note that whenever calling **vastool flush**, the entire user and group cache will be reloaded which can generate significant amounts of LDAP traffic, so it should be used sparingly.

Disconnected Mode

When **vascd** is unable to contact the KDC or the Active Directory server, it reverts to disconnected mode. While in disconnected mode *all* NSS and PAM requests are resolved from the cache. Disconnected mode can be entered for one or more of the following reasons:

- The computer object has been deleted. If the host's computer object has been deleted then **vascd** can no longer communicate with Active Directory because it can not authenticate. The solution to this problem is to re-create the computer object, then restart **vascd**.

- The */etc/opt/vas/host.keytab* file is missing or invalid. If */etc/opt/vas/host.keytab* is deleted or becomes corrupt then **vascd** is no longer able to communicate with Active Directory because it does not have credentials to authenticate as the computer. The solution to this problem is to delete then re-create the computer and restart **vascd**.

- The computer is physically disconnected from the network or the network is down.

- The Active Directory server is down.

Since most Unix account information requests are correctly handled by the cache, it is often difficult to tell if **vascd** is in disconnected mode. One sure way to determine the state of **vascd** is to look at the system log file. **vascd** makes a short log entry each time the connection mode changes.

Finally, the **vascd** disconnected mode is not intended to solve all problems related to completely disconnected situations. For example, **vascd** does not have any control over the ability of the system to continue to resolve DNS names or to continue accessing network file systems in an abrupt and completely disconnected situation.

Chapter 4

Unix Users and Groups

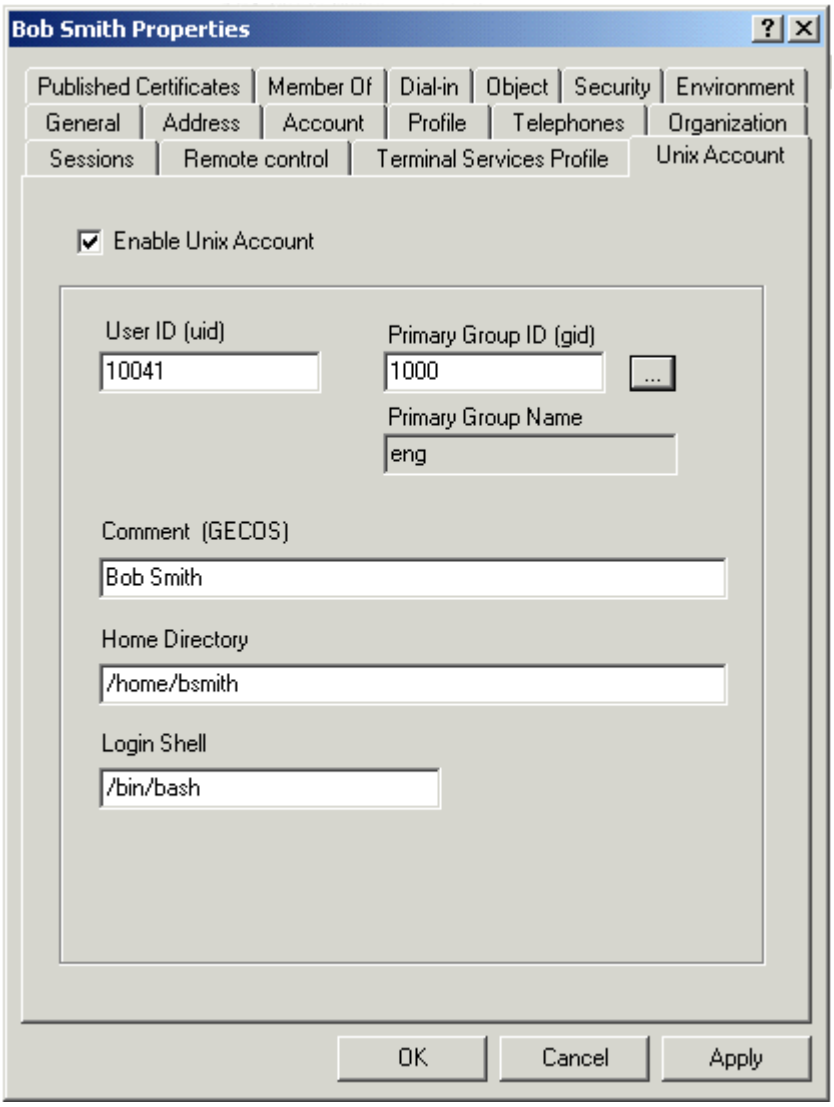
With VAS you can manage Unix user accounts, passwords, and Unix groups from Active Directory. Information in this chapter addresses the following concepts:

- Managing Unix User Accounts. See page 41.
- Managing Unix Group Accounts. See page 47.
- UID and GID Management. See page 52.
- Password Management. See page 53.
- Workstation Access Control. See page 56.

Managing Unix User Accounts

Using the VAS Snapin Extension

After installing the VAS Active Directory Users and Computers Snapin extension, a `Unix Account` tab appears in the properties dialog of all Active Directory users as shown in the following figure.



If the `Unix Account` tab does not appear in the User Properties dialog, review the installation steps outlined in the *Vintela Authentication Services Installation and Configuration Guide* to ensure that the VAS Active Directory Users and Computers Snapin extension has been installed.

The following is a list of the different fields on the User Unix Account tab.

`Enable Unix Account` This checkbox is used to enable and disable Unix and Linux accounts. If this checkbox is enabled, then the user will be able to login to Unix machines that have been joined to the domain. Disabling a Unix or Linux account causes the `Login Shell` to be set to */bin/false* and prohibits the user from logging in to Unix and Linux machines.

`User ID` This field is used to set the numeric Unix User ID or **UID**. It should be set to a numeric value between 1000 and the maximum UID for your Unix platform. When the Unix account is enabled for the first time, a default value is generated that is one greater than the highest UID previously used by users in the same Active Directory organizational unit (OU). If it is the first Unix account to be enabled in the organizational unit then the default value will be 1000. For more information on avoiding UID conflicts see "UID and GID Management" on page 52.

`Primary Group ID` This field is the Unix User Group ID or user **GID** that is used when determining the group ownership of files that are created by the user. The **Browse** button allows the administrator to look through and select the Primary Group ID from the list of Unix enabled groups. However, it is not required that the Primary Group ID be set to a value from in the browse list.

`Primary Group Name` If the Primary Group ID is set to the GID value of a Unix enabled group account in Active Directory, then the Primary Group Name field displays the group name of the Active Directory Group. If the Primary Group ID is set to a value not associated with a Unix enabled Active Directory group, then Primary Group Name displays the text `Unknown Group`.

`Comment (GECOS)` This is a free form field that is usually used to record the user's full name and other information (such as phone number and office location). The only restriction is that this field must not contain a colon (:) character. The default value is the user's full name.

`Home Directory` This is the user's Unix home directory. If the home directory does

not exist when the user logs in to a machine for the first time, it can be created by the **pam_vas** module. However, since a great deal of Unix user profile information (for example, desktop environment, shell profiles, and application specific settings) are saved in a user's home directory it is common to store home directories on a network file system such as NFS.

Storing home directories on a distributed file system is especially desirable in environments where Unix users log in to multiple workstations. Automount utilities can then be configured so that the user's same home directory information is available in the same directory location across multiple Unix machines.

`Login Shell` This is the shell that is executed when the user logs in using a terminal-based login. The default value for `Login Shell` is */bin/bash* . `Login Shell` must specify a program or script that exists on all Unix systems the user will be logging in to. Since login shells reside in different directories on different Unix operating systems it may be necessary to use symlinks to ensure that login shell locations are uniform across multiple Unix systems.

Managing User Accounts from the Unix Command Line

Using **vastool** you can create users, delete users, and list user information from scripts or the Unix command line. To create a user, use the **vastool create** command. For example, the following command would create the user *bsmith* in Active Directory :

```
$ vastool create bsmith
```

The command above creates a user in Active Directory that does not have its Unix Account enabled. To create a user that does have its Unix account enabled, you need to pass in a string formatted like a line from */etc/passwd* as an argument to the **-i**. You would do this as follows:

```
$ vastool create -i "bsmith:x:1003:1000:Bob:/home/bsmith:/bin/
```

```
bash" \
          bsmith
```

By default all users that are created with **vastool create** are created in the `Users` Organizational Unit (OU). To create a user in a different OU, use the **-c** command line option. For example, the following command would create a Unix enabled user *bsmith* in the *OU=sales,DC=example,DC=com* OU.

```
$ vastool create -i "bsmith:x:1003:1000:Bob:/home/bsmith:/bin/
bash" \
          -c "OU=sales,DC=example,DC=com" bsmith
```

To delete a user, use **vastool delete**. For example, the following command would delete the *bsmith* user.

```
$ vastool delete bsmith
```

To list users, use **vastool list users**. For example, the following command would list all the users with Unix accounts enabled:

```
$ vastool list users
jdoe:VAS:1000:1000:John Doe:/home/jdoe:/bin/bash
djones:VAS:1001:1000:Dave Jones:/home/djones:/bin/bash
molsen:VAS:1002:1000:Mary Olsen:/home/molsen/bin/bash
bsmith:VAS:1003:1000:Bob Smith:/home/bsmith:/bin/bash
```

The **vastool list users** command above does not directly contact Active Directory. Instead it uses the information from the **vascd** cache.

To bypass the cache and contact Active Directory directly, run the **vastool list** command with the **-l** option as follows:

```
$ vastool list -l users
jdoe:VAS:1000:1000:John Doe:/home/jdoe:/bin/bash
djones:VAS:1001:1000:Dave Jones:/home/djones:/bin/bash
molsen:VAS:1002:1000:Mary Olsen:/home/molsen/bin/bash
bsmith:VAS:1003:1000:Bob Smith:/home/bsmith:/bin/bash
```

Using **vastool list users** with the **-l** option is not efficient when used repeatedly from scripts because it generates LDAP traffic at every call. When scripting it is preferable to use **vastool list users** without the **-l** option and rely on **vascd** to keep the cache up to date.

When troubleshooting problems it is often useful to compare the output of **vastool list users** and **vastool list -l users**. Differences between the output of these two commands indicates that **vascd** is having problems keeping the cache up to date. If this is the case, the system administrator should check the system log for **vascd** error messages and flush the cache using the **vastool flush** command.

For more information about the **vastool list** command see the **vastool** man page in the appendix.

Moving Users in Active Directory

Moving Users to New Organizational Units

Use the `move` action or drag-n-drop functionality of the Active Directory Users and Computers Snapin to move existing users from one organizational unit (OU) to another. As long as Unix enabled users are being moved between OU's in the same domain, no additional configuration (other than the move action) is necessary.

Moving Users to new Domains

Unix enabled users can be moved from one domain to another using Windows admin utilities such as **movetree** and **netdom**. However, after moving the user object the user will not be able to logon to Unix machines until the user's `User logon name` (or *userPrincipalName*) is modified to correspond to the new domain the user belongs to after the move.

To change the `User logon name`, open Active Directory Users and Computers. Open the new user's properties page and select the `Account` tab. Ensure that the suffix of the `User logon name` matches the new domain. If necessary change the user-name domain component (for example, the part beginning with an @) by selecting the appropriate value from the drop-down list. After changing the User logon name, you will need to reset the user's password as well.

Disabling Unix User Acccounts

When a user's Unix account is disabled (by unchecking the `Enable Unix Account` checkbox), the user's `login shell` is set to */bin/false*. This prohibits the user from logging in via interactive login utilities. In technical terms, the check for whether or not a Unix account is disabled is determined in a PAM *session* call. This means that non-interactive PAM enabled applications that do not use the PAM session interface would still be able to authenticate the user. An example of one such application is the POP3 daemon (**pop3d**) commonly distributed on Linux. Since **pop3d** is not concerned with giving the mail user a login shell there is no need to call the PAM session interface.

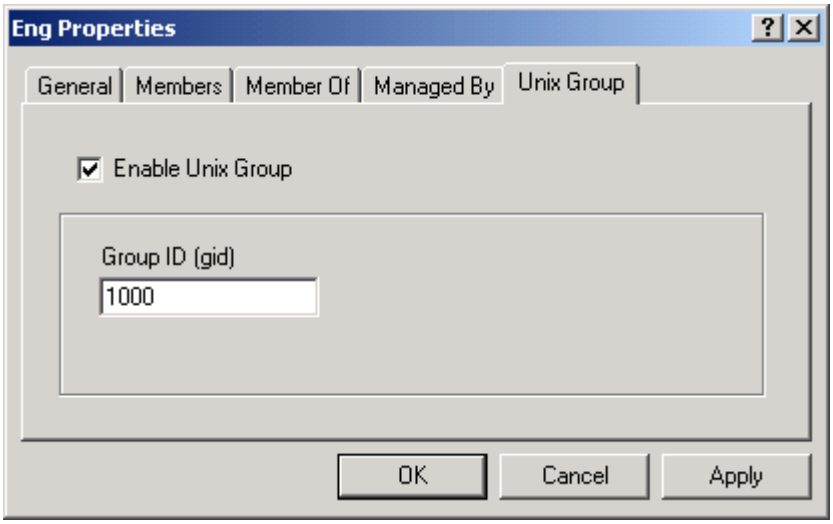
Therefore, in the case of **pop3d** (and other non-shell services) a user would be allowed to authenticate even if their Unix Account was disabled. To completely disable an account for interactive and non-interactive authentication, select the `Account is Disabled` setting from the `Account` tab or right click on the user and select `Disable Account` in the Active Directory Users and Computers snapin.

Another important behavior of a user with a disabled Unix Account is that all of the Unix account information remains valid even though the user is not able to log in on Unix or Linux hosts. Retaining Unix account information for disabled Unix accounts is necessary in order for the ownership of files and directories to be properly displayed (such as when using the **ls -l** command). If the user is removed (by removing the Active Directory user object) files owned bye the deleted user will be "orphaned" and will be listed as being owned by a numeric UID.

Managing Unix Group Accounts

Using the VAS Snapin Extension

After installing the VAS Active Directory Users and Computers Snapin extension, a `Unix Group` tab appears in the properties dialog of all Active Directory groups as shown in the following figure.



If the `Unix Group` tab does not appear in the Group Properties dialog, review the installation steps outlined in the *Vintela Authentication Services Installation and Configuration Guide* to ensure that the VAS Active Directory Users and Computers Snapin extension has been installed.

`Enable Unix Group` This checkbox is used to enable and disable a Unix Group. Enabling a Unix group allows a Unix GID to be assigned to the group so that it can be used for access control on Unix. Disabling a Unix group deletes the GID attribute which prohibits users who are members of the group from accessing Unix files based on Unix group permissions.

`Group ID` This field is used to set the numeric Group ID or GID for the Unix Group. It should be set to a numeric value between 1000 and the maximum group ID for your Unix platform. When the Unix Group is enabled for an Active Directory group, a default value is generated that is one greater than the highest GID previously being used by Unix Groups in the same Active Directory organizational unit (OU). If it is the first group to be be "Unix-enabled" in the OU unit then the default value will be 1000.

Managing Active Directory Groups from the Unix Command Line

Using **vastool** you can create groups, delete groups, and list group information from scripts or the the Unix command line. To create a user, use the **vastool create** command. For example, the following command would create the *sales* group:

```
$ vastool create -g sales
```

The command above creates a group in Active Directory that is not "Unix-enabled". To create a group that is "Unix-enabled", you need to pass in a string formatted like a line from */etc/group* as an argument to the **-i** as follows:

```
$ vastool create -i "sales:x:1003:" -g sales
```

By default all groups that are created with **vastool create** are created in the `Users` Organizational Unit (OU). To create a group in a different OU, use the **-c** command line option. For example, the following command would create a unix enabled group *sales* in the *OU=sales,DC=example,DC=com* OU.

```
$ vastool create -i "sales:x:1003"-c "OU=sales,DC=example,DC=com" -g sales
```

To delete a group, use **vastool delete** with the **-g** option. For example, the following command would delete the *sales* group.

```
$ vastool delete -g sales
```

To list groups, use **vastool list groups**. For example, the following command would list all the groups with Unix accounts enabled:

```
$ vastool list groups
eng@example.com:VAS:1001:jdoe@example.com,djones@example.com
it@example.com:VAS:1002:molsen@example.com
sales@example.com:VAS:1003:bsmith@example.com
```

The **vastool list groups** command above does not directly contact Active Directory. Instead it uses the information from the **vascd** cache.

To bypass the cache and contact Active Directory directly, run the **vastool list** com-

mand with the **-l** option as follows:

```
$ vastool list -l groups
eng@example.com:VAS:1001:jdoe@example.com,djones@example.com
it@example.com:VAS:1002:molsen@example.com
sales@example.com:VAS:1003:bsmith@example.com
```

Using **vastool list groups** with the **-l** option is not efficient when used repeatedly from scripts because when called repeatedly it may generate too much LDAP traffic. When scripting it is preferable to use **vastool list groups** without the **-l** option and rely on **vascd** to keep the cache up to date.

When troubleshooting problems it is often useful to compare the output of **vastool list groups** and **vastool list -l groups**. Differences between the output of these two commands indicates that **vascd** is having problems keeping the cache up to date. If this is the case, the system administrator should check the system log for **vascd** error messages and flush the cache using the **vastool flush** command.

Moving Groups in Active Directory

Moving Groups to new Organizational Units

Use the `move` action or drag-n-drop functionality of the Active Directory Users and Computers Snapin to move existing groups from one organizational unit (OU) to another OU within the same domain. "Unix-enabled" groups can not be moved across domain boundaries with Windows utilities such as **movetree** and **netdom**. Instead, the group should be removed and recreated in the new domain.

Active Directory Group Types and Scopes

There are two types of groups in Active Directory, distribution groups and security groups. Security groups can be used to set permissions and provide access control. VAS only supports "Unix enabling" security groups. Distribution groups are not supported by VAS because they cannot be used for security purposes.

An Active Directory security group can have one of the following scopes:

Domain Local	The group is only visible within its local domain. Domain local groups can include other groups and users from any domain (when Active Directory is in native mode), but are only usable in the local domain and sub-domains.
Global	The group is visible throughout the Active Directory forest but its membership is restricted to users and groups within the same domain or a parent domain in the same tree.
Universal	The group is visible throughout the Active Directory forest and may contain users, other universal groups, and global groups from any domain. Universal groups are useful when a group needs to be used in multiple domains.

As a general rule, the VAS client will only use "Unix enabled" security groups that are reside in the same domain as the VAS client. The exception is primary GID of a user that logs in to a Unix host that is joined to a domain that is different from the domain where the user resides. In this case, VAS performs additional group discovery to ensure that the name primary group of the user can be resolved on the Unix host. Primary groups that come from other domains will be named differently than the groups that reside in the same domain as the Unix computer. When displayed the primary groups from other domains names will consist of the group name appended with "@" then the domain name.

Primary Unix groups from other domains should *not* used to provide access control to Unix resources because they will only be available if the cross domain users log in to the Unix client otherwise. Windows domain local groups from the domain where the Unix computer object resides will always be visible to the Unix host, and can contain users from other from other domains. Groups with global and universal scope can be alsoc be used by VAS, but only Unix hosts they'll only be seen by Unix hosts that are in the same domain as the group. Windows domain local groups that are in the same domain as the Unix computer object are the best choice use in controlling access to Unix files and resources.

Nested Group Support

Active Directory allows nested groups, or groups that contain other groups. Though this feature will be available in a future release of the VAS, the current release of VAS will only recognize members of the group that are not nested within other groups.

UID and GID Management

When using VAS to manage users from Active Directory it is important to keep track of Unix user UIDs and Unix group GIDs. For maximum flexibility, the `Unix Account` tab for users and groups in the Active Directory Users and Computers Snapin gives a warning but does allow duplicate UIDs and GIDs to be assigned within a single Active Directory tree. Therefore, without careful planning, it is easy to lose track of which UIDs and GIDs have been used.

UID/GID Partitions

A simple way to avoid UID/GID conflicts is to divide up the UID/GID namespace according to an administrative model that makes sense to the network administrator. This strategy works especially well if organizational units (OUs) are used to divide users and groups into logical, separately managed, containers.

If a range of UIDs for GIDs is administratively allocated per OU, the network administrator can avoid conflicts simply by using the default UID/GID recommendations that are displayed when an account is "Unix-enabled" for the first time. The exception is when the first user or group in an OU is "Unix-enabled". In this case the administrator will need to manually set the UID and GID values to the first value in the allocated UID/GID range for the new OU. After this, all new users that are "Unix-enabled" will suggest non-conflicting default values based on existing "Unix-enabled" users in that OU. For more information about creating users see “UID and GID Management” on page 52.

Avoiding Duplications with Local Accounts

When creating local user accounts in >*/etc/passwd* and */etc/group* it is important to be very careful not to duplicate UIDs or GIDs of "Unix-enabled" Active Directory users and groups. The administrator needs to be especially careful when creating users using the **useradd** or **adduser** commands. The reason is that unless the **-u** option is used to explicitly specify the UID, the default behavior of **useradd** is to use NSS calls to determine the next highest UID. If the **nss_vas** module is configured in */etc/nsswitch.conf* as it should be for a machine joined to the AD domain, the result will be the creation of a new local user in */etc/passwd* with a UID that is one higher than highest UID in Active Directory.

For security reasons, the **pam_vas** module does not allow Active Directory users to log in who share a UID with a system account. The **pam_vas** module also does not allow Active Directory users to log in if they have a UID conflict with other users in Active Directory. The one exception is that **pam_vas** does not perform UID conflict checks for Active Directory users with UIDs under 1000.

Password Management

Windows Domain Password Policies

Windows 2000 and Windows 2003 allow password policies to be applied to domains and domain controllers via group policy. Windows domain password policies allow administrators to enforce strong password policies for Windows users including settings for minimum password length, password complexity, password change frequency, and password history. VAS enforces all the Windows password policies for Unix logins. See the Microsoft Windows server documentation for more information about setting Windows password policies.

Changing Passwords

One of the primary features of VAS is that the same user name and password can be used to logon to Windows and Unix. In conjunction with this feature it is also possible for a user to perform a password change from Windows and Unix so that subsequent logins to either platform will accept the new password.

Changing a user's Active Directory password from Unix using VAS can be accomplished using the **vastool passwd** command, or existing PAM enabled password utilities that work with the **pam_vas** PAM module.

Changing passwords with vastool passwd

The **vastool passwd** command can be used by a user to perform a password change or to set another user's password. For example the following command would prompt the current user to change their password:

```
$ vastool passwd
```

The caller will be prompted to enter their current password, the new password, and a verification of the new password.

If **vastool passwd** is run with the **-u** vastool option then it will perform a password change for the user specified with the **-u** option. For example, the following command will let the current user change bsmith's password (if they know bsmith's current password):

```
$ vastool -u bsmith passwd
```

When using **vastool passwd** from scripts it is often advantageous to use the vastool **-s** option which allows the password prompts to be satisfied with input from **stdin**. For example a web page could be written to enable users to change their passwords using **vastool passwd**. The web page would need to get the user's old password and the new password, and then call **vastool passwd** as follows:

```
$ vastool -u $USERNAME -s passwd
```

The caller must write the user's current password, the new password, and a verification

of the new password to the stdin of the **vastool passwd** process.

If a user is specified on the **vastool passwd** command line then the password is *reset* (not changed) for that user. This requires administrative priviledges. For example the following command would reset the password for the *bsmith* user. *unixadmin* must be a user with administrative priviledges:

```
$ vastool -u unixadmin passwd bsmith
```

For more information on using the **vastool passwd**, see the **vastool** man page in the appendix.

Changing passwords with system utilities and pam_vas

On PAM enabled systems, password changes generate calls to the *password* interface of the PAM modules configured for use with the application or utility that is changing the password. The most commonly used utility is **/bin/passwd**. If **/bin/passwd** is configured to use **pam_vas**, then the password change will be made on the domain controller for "Unix-enabled" Active Directory users.

On some systems such as HP-UX and Solaris, the **/bin/passwd** command does not properly leverage the PAM abstraction and contains mechanism specific code that is keyed by settings in */etc/nsswitch.conf*. If this is the case, you may see the following output:

```
passwd:  Changing password for mattp
Supported configurations for passwd management are as follows:
    passwd: files
    passwd: files ldap
    passwd: files nis
    passwd: files nisplus
    passwd: compat
    passwd: compat AND
    passwd_compat: ldap OR
    passwd_compat: nisplus
Please check your /etc/nsswitch.conf file
Permission denied
```

If you see the above output, you must use the **vastool passwd** command to change pass-

words of "Unix-enabled" Active Directory users as described in the previous section. To change passwords of local users listed in */etc/passwd* , use the **passwd -r files** command to instruct **/bin/passwd** to change local passwords.

Note that on HP-UX the **passwd -r files** does not work if the passwd entry in */etc/nsswitch.conf* has more than two entries.

Password Expiration

Windows allows the configuration of a password policy to force users to change passwords often, to select strong passwords, or to force password change at the next login after resetting the password. The **pam_vas** module supports the full range of Windows password policy features.

When users log in and their password is expired, **pam_vas** prompts for the old password, and a new password using the "PAM conversation" mechanism. However, not all applications use the PAM interface correctly to support the interactive password change prompts that are provided through the "PAM conversation" mechanism. For example, Linux KDE's **kdm** does not process the PAM requests from **pam_vas** correctly, but GNOME's **gdm** and the system **login** applications do.

Workstation Access Control

Administrators commonly want to restrict access to sensitive machines on the network to only selected users and groups. This is especially true of Unix machines which are used to deploy business critical applications.

VAS provides a mechanism for specifying the Active Directory users, groups and domains that can authenticate to Unix hosts. The VAS authentication modules (**pam_vas** and the VAS AIX module), consult two configuration files in the */etc/opt/vas/* directory named *users.allow* and *users.deny*. If either of these these files exist then the VAS authentication components will use them to determine which users should be allowed to successfully authenticate.

For a detailed description of *users.allow* and *users.deny*, see the **pam_vas** man page in the appendix.

Chapter 5

Migration

VAS provides several features that allow you to migrate user, group, and other configuration information stored in flat text files to Active Directory, which then serves as the central management point for both Windows workstations and Unix hosts.

Information in this section addresses the following concepts:

- Importing Users and Groups. See page 57.
- Migration from NIS. See page 59.
- Migration from MIT Kerberos. See page 62.

Importing Users and Groups

For environments that have pre-existing */etc/passwd* files and */etc/group* files either locally or through NIS, the **vastool load** command provides a way to import these users and groups into Active Directory.

The **vastool load** command can read any file that follows the */etc/passwd* or */etc/group* format, or it can read from it's standard in.

vastool load does not check for UID/GID conflicts before creating the user and group objects. Therefore, before importing a passwd file or a group file, make sure that the import will not cause UID or GID conflicts with accounts that are already in Active Directory. It is also important to make sure that the file does not contain local system accounts that would be inappropriate for storage in Active Directory.

There may be situations where Active Directory accounts already exist for the Unix users and groups and you simply want to import their Unix identity information (such as UID and GID). In this situation you will want to use the **-e** option. Using the **-e** option will cause **vastool load** to set the Unix attributes for existing users and groups that are being imported. For more information on using the **-e** option see the **vastool** MAN page.

Also, note that only users with the appropriate administrative rights in Active Directory can create users and groups. So if your current login session is not as an administrative user, use the **vastool -u** option to specify the administrative user name.

To import users from a file run the following command:

```
$ vastool -u unixadmin load -f /tmp/my_passwd_file.txt users
```

The above command creates all of the users from the */tmp/my_passwd_file.txt* in the default *Users* container in Active Directory. Use the **-c** option to specify a container in which to create users. The following is an example of importing users into the *sales* OU:

```
$ vastool -u unixadmin load -f /tmp/my_passwd_file.txt \
          -c  OU=sales,DC=example,DC=com  users
```

One limitation of importing users is the inability for VAS to preserve the passwords of imported users. VAS does not have access to users' plain text passwords, and Active Directory cannot accept the encrypted password hashes that are stored in *passwd* files. Therefore, the **vastool load** command generates a new random password for each user which is saved to a file the administrator can use to notify users of their login password. Another option is to instruct **vastool load** to set a common default password for all newly imported user accounts with the **-p** password option.

By default imported users are forced to change their passwords when they first login.

To import groups from a file run the following command:

```
$ vastool -u unixadmin load -f /tmp/my_group_file.txt groups
```

As with **vastool load users**, groups are created in the default *Users* container. Likewise, the **-c** option can be used to loading groups into a specific container as follows:

```
$ vastool -u unixadmin load -f /tmp/my_group_file.txt \
          -c  OU=sales,DC=example,DC=com  groups
```

Group membership lists are stored in the Active Directory Group object's *members* attribute. The values stored in *members* must be distinguished names (DNs). In order to create the group object with its existing members, **vastool** must look up each users DN, meaning that all members of the group must already exist in Active Directory. When importing users and groups, you should always import the users first, then the groups.

Migration from NIS

NIS Architecture

The Network Information System (NIS) was developed to provide a system for managing and distributing information such as user accounts, passwords, and other network related information to Unix hosts attached to the network. The main purpose of NIS was to provide centralized control for the standard Unix configuration files normally found in the */etc* directory.

NIS uses a client/server architecture, where NIS servers distribute NIS information to clients on the network. Multiple NIS servers can be used; normally a "master" NIS server drives a number of "slave" NIS servers. NIS uses a flat namespace segmented by domains. Each NIS server acts as a server for one domain and NIS clients can access the information that is in their NIS domain.

Each NIS server has a copy of the NIS information database. This database is generated on the master server from a set of text files that contain the plain text configuration information that should be distributed to NIS clients. Changes to the master database are replicated to the NIS slave servers that are members of the NIS domain. These changes are ultimately used by NIS clients that are members of the NIS domain.

NIS Maps

NIS information is divided into separate databases called NIS Maps. Each NIS Map normally represents one of the plain text files on the master NIS server. For example, there will be a NIS Map for the *passwd* file which holds user account information; there will also be a NIS Map for the *group* file which holds all of the groups and group membership lists.

A NIS Map is basically a two columned table with key/value pairs. Since each type of NIS Map can have different kinds of keys, there are normally multiple NIS Maps for each text file on the master server. For example, the *passwd* NIS Map has a *passwd.byname* map and a *passwd.byuid* map. In *passwd.byname* the username is the key, and in *passwd.byuid* the user UID is the key.

Applications and users can access NIS Map data by using the **ypcat** and **ypmatch** commands. **ypcat** will print out a NIS Map's contents while **ypmatch** can be used to search a NIS Map for key values. Applications can also get NIS Map information by directly using NIS RPC's to directly query the NIS server.

VAS support for NIS

When migrating Unix user and group management to Active Directory it is usually desirable to migrate NIS information to Active Directory. VAS provides several features that allow Unix hosts to continue to access NIS information even though that information is being stored in Active Directory.

The VAS NIS Map Schema Extension

The default Active Directory Schema does not provide any attribute or class definitions for storing NIS Map data. This means that a schema extension must be installed to allow the storage of NIS Map information.

The VAS NIS Schema extension consists of a new class definition for a **nisMap** class. Each **nisMap** class has two NIS related attributes that are added by the VAS schema extension, **nisMapData** and **nisMapFormat**. Each of these attributes are free form strings. **nisMapData** stores the NIS Map file contents, and **nisMapFormat** describes how to parse the map. The **nisMapFormat** attribute is not used in VAS 2.2, but will be

in VAS 2.3. The name of a **nisMap** object is stored in it's **CN** attribute.

Migrating NIS maps to Active Directory

Current NIS Map files can be migrated to Active Directory using the **vastool nis-import** command, which will create a **nisMap** object and store it's map data in plain text in the **nisMapData** attribute. For complete instructions and examples of using **vastool nis-import** see the **vastool** man page.

The NIS *passwd* and *group* maps cannot be migrated using **vastool nis-import** because Unix user and group information is stored in the user and group objects themselves -- not in **nisMap** objects. Use **vastool load** to import users and groups into Active Directory.

In VAS 2.2, the only way to manage **nisMap** objects is to maintain a a copy of the original text file that was used to import the map with **vastool nis-import**. When making changes, edit the original text file and use the **vastool nis-import** with the **-o** option to re-import and overwrite the existing **nisMap** object with the new data.

VAS 2.3 will include an MMC Snapin extension to enable the management of **nisMap** data without having to reimport the NIS Map.

Using the VAS ypcat replacement

In order for applications to access the NIS Maps stored in Active Directory, they will have to use the VAS */opt/vas/bin/ypcat* replacement script. This script calls the **vastool ypcat** command, which returns the NIS Map data that was previously imported into Active Directory.

In order to reduce LDAP traffic, **vastool ypcat** actually returns information that was previously cached by **vascd**. The cache is updated according to the **vascd update-interval** setting. See the **vascd** MAN page for more information about the **update-interval** setting.

You may need to rename the system **ypcat** utility and make a symlink to the VAS **ypcat** utility in its place in order for all applications to take advantage of the VAS NIS functionality.

NIS Map Search Location

In order to emulate the behavior of NIS domains, it is possible to use different sets of NIS Maps for different sets of computers within the same Active Directory domain. A VAS client will only look for NIS Maps in the container where its computer account object was created. This allows administrators to have different sets of NIS Maps stored in different organizational units (OU's). The computer objects can then be created using **vastool join** with the **-c** option to specify a container the computer account should be created in.

VAS 2.3 will include an option to specify a search base for the NIS Maps, so that administrators can have further control over the NIS Map configuration.

NIS Support Limitations

There are some limitations the of the NIS support included in the VAS 2.2 product. Some of these limitations are related to limitations of NIS itself, others are limitations of the VAS 2.2 product that will be addressed by NIS support enhancements being developed for the VAS 2.3 product.

Many applications do not use **ypcat** to access NIS Map information, but instead use NIS RPC's over the network to obtain data directly from the NIS server. Until VAS 2.3 is released, these applications will not be able to access the VAS NIS Map data stored in Active Directory.

Migration from MIT Kerberos

VAS provides the same Active Directory Kerberos inter-operability as MIT Kerberos without the complexity that goes along with it. Setting up Kerberos inter-operation with Active Directory is as simple as installing VAS on a Unix computer and joining the computer to the Active Directory domain. With VAS there is no need to create "computer service accounts", export keytabs on the domain controller, manage keytabs on the Unix host, or manipulate Kerberos configuration files.

Keytabs and the krb5.conf

Because VAS utilizes Kerberos v.5 and because its configuration and keytab files (*/etc/opt/vas/vas.conf* and */etc/opt/vas/host.keytab*, respectively) are compatible with structure and functionality used by MIT Kerberos, migrating from MIT Kerberos to VAS is very simple.

Assuming VAS is already installed, complete the following:

1. If necessary, rename the existing MIT */etc/krb5.conf* file.

   ```
   # mv /etc/krb5.conf /etc/krb5.conf.orig
   ```

2. Create a symbolic link pointing from */etc/krb5.conf* to */etc/opt/vas/vas.conf* as follows:

   ```
   # ln -s /etc/opt/vas/vas.conf /etc/krb5.conf
   ```

3. Store cached Active Directory domain or Kerberos realm information in *vas.conf* by executing the following command:

   ```
   # /opt/vas/bin/vastool realms cache toconf
   ```

   Because VAS utilizes DNS SRV records to discover domains and domain controllers, the */etc/opt/vas/vas.conf* file does not normally contain any specific realm configuration information. When using VAS with MIT Kerberos, the */etc/opt/vas/vas.conf* file must contain explicit realm configuration for each Active Directory domain. This is accomplished with the **vastool realms cache toconf** command as shown above.

If Active Directory servers are added to or removed from your network and the DNS SRV records change, you may need to re-run **vastool realms cache toconf** as shown above.

User and Group migration

Because MIT Kerberos is an authentication-only solution, most MIT Kerberos deploy-

ments still require that Unix account information be stored in */etc/passwd* and */etc/group*, or in NIS. If this is the case, users can be migrated to Active Directory using instructions provided in “Unix Users and Groups” on page 41.

Appendix A

pam_vas Manual Page

pam_vas

A PAM module for authenticating Active Directory users on Linux/UNIX computers via the Kerberos protocol.

```
auth <control-flags> /opt/vas/lib/security/pam_vas.so
debug traceget_tgtservice=servicename helper_timeout=sec-
onds create_home dirno_access_check no_uid conflict_check
no_disconnected no_force_update get_nonvas_pass
realm_prompt check_pwd LastSet

acct <control-flags> /opt/vas/lib/security/pam_vas.so
debug

password <control-flags> /opt/vas/lib/security/pam_vas.so
debug helper_timeout=seconds

session <control-flags> /opt/vas/lib/security/pam_vas.so
debug
```

Description

pam_vas is a PAM module that uses the Kerberos protocol to authenticate users against Active Directory. It must be used in conjunction with the `vascd` daemon and the NSS module `nss_vas`, and enables Unix services to authenticate users whose credentials are stored in Active Directory- which acts as the Kerberos Key Distribution Center (KDC).

PAM enabled services are configured with four types of entries, auth, acct, password,

and session. pam_vas can be used with all of these. The following is a list of each type of PAM configuration entry and what pam_vas can do for that type of PAM call:

auth	pam_vas will handle the authentication of a user name and password via the Kerberos protocol against Active Directory. pam_vas can also create a user's home directory if it does not exist, check for system UID conflicts, set up a user's Kerberos ticket cache, and check for computer access restrictions for the given user.
acct	pam_vas will check for user account restrictions set in Active Directory.
password	pam_vas will change a user's Active Directory password.
session	pam_vas will initialize a user's login session.

`vastool configure pam` will configure the system's PAM configuration. Do not be change the defaults created by `vastool` unless you really know what you are doing.

Control Flags

pam_vas has some special features which allow you to take advantage of the extended syntax for control flags in newer versions of PAM that are found in recent Linux distributions (this extended syntax is not supported in current versions of Solaris). pam_vas will ignore any calls for users that are not Active Directory users, i.e. system accounts. This allows you to use the following syntax:

[ignore=ignore success=done default=die]

"ignore=ignore" allows the PAM library to call other PAM modules in the stack if the username is not recognized by pam_vas as an Active Directory user. "success=done" instructs the PAM library to complete the authentication process if pam_vas is successful. "default=die" instruct the PAM library to error out if pam_vas returns an error condition. This control flag syntax is the default configuration on systems that support the /etc/pam.d/* PAM configuration. On platforms that use `/etc/pam.conf`, the "suffi-

cient" control flag is used by default.

If the PAM library on your operating system (for example- Solaris 8 and 9), does not support this extended syntax, you will need to use the "sufficient" control flag. Unfortunately, this control flag will cause the PAM library to continue down the PAM stack when pam_vas fails which may result in multiple prompts to the user.

For example, if the PAM library does not support the extended control flag syntax, it is not possible to distinguish the difference between a local user login, and an Active Directory login with an incorrect password. The result is that if an Active Directory user mis-types their password, the PAM library will continue down the PAM stack and additionally allow the other PAM modules (such as pam_unix) to prompt the user for a password.

Please note that PAM configuration file syntax can be quite complex, so make sure you know what you are doing before changing the default control flags that are configured by `vastool`.

Auth Arguments

These are the arguments that you can append on to the auth entries for pam_vas. By default, pam_vas auth entries are configured with the `get_tgt` and `create_homedir` arguments.

`debug`	Log debug messages to syslog. These will normally go to the secure system logs. This is useful for debugging authentication problems.
`trace`	Log debug messages to syslog that track the call stack of pam_vas. These will normally go to the secure system logs. This is useful for debugging authentication problems.
`get_tgt`	Obtains a Kerberos Ticket Granting Ticket (TGT) using the user's supplied credentials, and then uses that TGT to obtain a service ticket for the computer object in Active Directory. A Kerberos TGT is a ticket that is obtained with the user's password, and can then be used to obtain other tickets without having to reuse the user's password. In other words, the use of `get_tgt` saves the necessary TGT credentials for subsequent use of to ker-

berized "sign on" utilities.

Removing `get_tgt` option will cause pam_vas to obtain a service ticket directly using the user's password, without obtaining the TGT. The ticket is not stored anywhere on the filesystem, and is destroyed when the PAM session is ended. This results in less network traffic and load on the KDC. It is therefore recommended that you remove the `get_tgt` option when using pam_vas with a service that does not establish interactive login sessions, such as a web server or an IMAP server.

`service={service principal name}` By default pam_vas will try to obtain a service ticket for your computer principal name. The computer principal name will be generated from the computer's hostname, or from the `host_principal_override` option in the `[libdefaults]` section in `/etc/opt/vas/vas.conf`. In the default case, the computer object for the local machine is the Service Principal used by pam_vas. The `service` allows the administrator to change the service name to be something other than the "host" principal.

`helper_timeout=`*`seconds`* `pam_vas` uses an external program called `vasauth_helper` to perform the Kerberos operations. There is a default timeout of 10 seconds, after which the external program will return. If using pam_vas in an environment where the network connection to the Active Directory KDC is very slow, or there is large number of backup domain controllers that are often used, then you can extend the timeout.

If the helper_timeout expires before `vasauth_helper` completes, then `pam_vas` will perform disconnected authentication.

`create_homedir` Create the user's home directory if it does not exist. This will also copy the `/etc/skel` contents into the new home directory and setup the appropriate permissions (i.e. 0600).

`no_access_check` Disables the workstation access control checks that pam_vas performs. These access checks are based on `/etc/opt/vas/`

users.allow and /etc/opt/vas/users.deny. See the Computer Access Control section portion of this man page for more details.

no_uidconflict_check Disables the UID conflict checking. UID checking will not allow any Active Directory user to login to the system if they have a UID conflict with another user. This does not affect local accounts. UID checking is performed for security reasons to prevent users from accidentally gaining access to files and resources they should not be able to.

UID checking is *not* performed for UID's under 1000. This allows administrators to setup accounts in Active Directory that could be used for root access.

no_disconnected Disables disconnected authentication. Setting this option will also disable the caching of the user passwords hashes.

realm_prompt Adds the user's realm name to the password prompt. Note that that this is a potential security hole since it shows that a valid account name has been entered.

no_force_update In order for users to be able to login immediately after being created, pam_vas will ask vascd to ignore it's blackout period and update the cache for the user immediately. To disable this behavior, add no_force_upate to the pam_vas options. For services that perform many authentications repeatedly, adding this option will greatly reduce the amount of LDAP traffic and the number of LDAP searches the VAS client will perform.

get_nonvas_pass On platforms that do not support the extended syntax for PAM, this option is necessary to get the system PAM modules underneath pam_vas such as pam_unix to reuse the password pam_vas obtained. This also prevents Active Directory from being prompted twice when they enter an incorrect password.

check_pwdLastSet Checks the user's pwdLastSet attribute after a successful authentication to ensure that their password has not expired. If

pwdLastSet is set to 0, then the user is required to change their password before they can login. This provides a workaround for situations where Active Directory will continue to give Kerberos tickets to users' whose passwords have expired.

Acct Arguments

These are the arguments that can be append on to the acct entries for pam_vas. By default, pam_vas acct entries are not configured with any arguments.

`debug` Log debug messages to syslog. These will normally go to the secure system logs.

`trace` Log debug messages to syslog that show the call stack of pam_vas. These messages will normally go to the secure system logs.

Password Arguments

These are the arguments that can be append on to the password entries for pam_vas. By default, pam_vas password entries are not configured with any arguments.

`debug` Log debug messages to syslog. These will normally go to the secure system logs.

`trace` Log debug messages to syslog that show the call stack of pam_vas. These messages will normally go to the secure system logs.

`helper_timeout=`*`seconds`* pam_vas uses an external program called `vasauth_helper` to perform the Kerberos password change. There is a default timeout of 10 seconds, after which the external program will return. If using `pam_vas` in an environment where the network connection to the Active Directory KDC is very slow or if there are many backup domain controllers and it is common for primary domain controller to be unavailable, then you should extend the timeout.

Session Arguments

These are the arguments that can be append on to the session entries for pam_vas. By default, pam_vas session entries are not configured with any arguments.

`debug`	Log debug messages to syslog. These will normally go to the secure system logs.

Computer Access Control

It is possible to have fine grained control over which Active Directory users can login using pam_vas. This is especially useful when pam_vas is used on sensitive servers where shell access should only be granted to a few users. Note that computer access control functionality does not work with local user accounts, only with Active Directory user accounts.

By default, each time a user is successfully authenticated, pam_vas will check to see if access should be allowed by using information recorded in `/etc/opt/vas/users.allow` and `/etc/opt/vas/users.deny`.

Lines starting with '#' are comments. Valid entries are user principal names, groups, and realm names. User principal names will have the form of user@realm and realm names will have the form of @realm. Any entry that does not have the '@' character will be interpreted as a group name. A user will be granted access according to the following rules:

1. If the `/etc/opt/vas/users.allow` file contains the users's UPN.

2. If the `/etc/opt/vas/users.allow` file contains a group the user is a member of and the `/etc/opt/vas/users.deny` file does not contain the user's UPN.

3. If the `/etc/opt/vas/users.allow` file contains the realm of the user, and the `/etc/opt/vas/users.deny` file does not contain the user's UPN or a group the user belongs to.

4. If the `/etc/opt/vas/users.allow` file is empty or does not exist, and the `/etc/opt/vas/users.deny` file does not deny the user in anyway- explicitly, by group membership, or by the realm of the user.

5. If the `/etc/opt/vas/users.allow` file and the `/etc/opt/vas/users.deny` file do not exist.

In all other cases, the user is denied access. Note that an empty `/etc/opt/vas/users.allow` file will be treated the same as if the `/etc/opt/vas/users.allow` file did not exist- the same applies to the `/etc/opt/vas/users.deny` file. Once the `/etc/opt/vas/users.allow` file has some entries, only users who qualify according to those entries are allowed access. There is no need to explicitly deny everyone in `/etc/opt/vas/users.deny`.

In all cases, the users.allow file takes precedence over the users.deny file. For example, if a user is group allowed but group denied, the user is granted access. Also, if a user is explicitly allowed, but explicitly denied by having the user's UPN in both files, then the user is granted access.

Note that in determining whether a given user is a member of a listed group, both the explicit group membership list of the given group and the user's primary group id are used. So if a user is not explicitly listed in a group's membership list, but a user's primary group ID is the same as the listed group's, then the user is considered a member of that group.

Since it possible to put groups into `/etc/opt/vas/users.allow` and `/etc/opt/vas/users.deny`, you can set each file's contents once and then manage who has access to that Unix client through Active Directory by managing the group membership lists of the groups used in the files.

It is possible to turn off this access check in `pam_vas` through the `no_access_check`. This may be useful with applications such as imapd, pop3d, and Apache that don't grant shell access to users but still need to authenticate them. This allows administrators to set up Unix clients that have restricted access to shell services, but still run business applications that can authenticate all Active Directory users.

The following is an example of a `/etc/opt/vas/users.allow` file that grants access to the kyle and jason users and to the unixAdmins group:

```
kyle@example.com
```

```
jason@example.com

unixAdmins
```

The following example shows a `/etc/opt/vas/users.deny` file that is configured to deny access to the brad user -- this user belongs to unixAdmins group, but is in trouble and has had his access taken away.

```
brad@example.com
```

Note that the `/etc/opt/vas/users.deny` file will not be used as often as `/etc/opt/vas/users.allow` in most cases. The `/etc/opt/vas/users.deny` file is provided to allow maximum flexibility to administrators.

Disconnected Authentication

By default, pam_vas supports disconnected authentication. Each time a user successfully logs in against the Active Directory server, their password is stored as an SHA-1 hash in a secure cache file that is only readable by the root user. A timestamp is also stored with the hashed password, and cached passwords are good for 30 days.

This disconnected authentication allows services to continue to authenticate users if the network connection to the Active Directory server goes down, or if a laptop is used without plugging into the network. This behavior can be disabled by using the `no_disconnected` flag- in this case passwords will not be cached so no disconnected authentication can take place.

Appendix B

vascd Manual Page

vascd

The client daemon for use with `pam_vas` and `nss_vas`

`vascd -h -v -l -d -p` *`pidfile`* `-o` *`user`* `-f` *`logfile`* `-c` *`update-interval`* `-n` *`principal`* `-t` *`timesync-inteval`* `-r` *`realms cache-sync-interval`*

Description

vascd is a daemon that must be started on UNIX/Linux workstations in order for pam_vas and nss_vas to operate correctly. When started, vascd authenticates to Active Directory using credentials that were established at the time that the computer object was created in the Active Directory domain (see vastool (1) for details). vascd then uses this secure connection to Active Directory to proxy and cache user and group account information for other processes.

The use of vascd allows several important features:

Security - Due to the way that PAM and NSS subsystems operate, most LDAP based UNIX account management solutions require that anonymous or public access be allowed to UNIX account attributes. Since vascd authenticates as an Active Directory domain computer, vascd can access UNIX account information that is protected by Active Directory access control restrictions.

Scalability - Due to the way that the PAM and NSS subsystems operate, most LDAP based UNIX account management solutions generate excessive numbers of LDAP connections and LDAP search requests. This results in dramatically increased network traffic and load on the LDAP server. vascd establishes a single connection to Active

Directory for the computer it is running on, and proxies all of the NSS requests through that single connection. At the same time, vascd is able to perform intelligent caching of frequently used information so that LDAP traffic is reduced to the absolute minimum.

Disconnected Operation - Since vascd maintains a persistant cache of frequently used information, it is possible for the entire authentication system to continue to operate in environments where the network connection to Active Directory is unreliable or completely unavailable.

vascd Options

These are the options you can pass to `vascd` when starting the daemon.

`-h`	Shows the `vascd` usage.
`-v`	Print the `vascd` version and exit.
`-l`	Print the `vascd` licensing information and exit.
`-d`	Causes the daemon to not detach from the controlling tty, and prints debugging messages to stdout. This is useful for debugging purposes.
`-p pidfile`	Specify an alternative pid file. Default is `/var/state/vas/vascd/.vascd.pid`
`-o user`	Run vascd as the specified user. Default is daemon.
`-f logfile`	Specify an alternative log file. Default uses syslog if the system supports syslog. Otherwise, vascd will log to `/var/log/vascd` by default.
`-c update-interval`	Specify the "blackout" period in seconds. Default is 600 seconds (10 minutes). This setting can also be set in `vas.conf` file in the "[vascd]" section. An example is: [vascd] update-interval = <seconds> In order to minimize network traffic and the load on the directory

server, vascd aggressively caches data that is retrieved from the directory server so that subsequent requests can be satisfied from the cache with out having to contact the directory server. The cache is only updated when it is determined that a change has been made in the directory. Nevertheless, due to the way that the NSS subsystem is called it is not uncommon for hundreds of requests to be generated in a matter of seconds. Therefore, in order to further reduce the load on the network and on the directory, vascd enforces a "blackout period" during which all requests will be resolved from the cache.

By default the blackout period is set to 10 minutes. What this means is that the addition of new users and changes to UNIX account information may take as long as 10 minutes to become visible on the Linux/Unix workstation due to the blackout period. For environments where changes must take affect quicker, it is possible to change the "blackout period" using the `-c` command line option when starting vascd. Using `-c` to decrease the "blackout period" will result in a system where hosts are more responsive to changes made in Active Directory-- at the expense of increased network traffic and load on the Active Directory server.

The amount of additional network traffic depends on the number of Linux/UNIX hosts and their use. In small installations (less than 100 hosts or less than 100 users) the "blackout period" could be safely reduced. In larger installations it is recommended that the "blackout period" be left at the default value or increased to 30 minutes or 1 hour. Regardless of the "blackout period", the administrator can force vascd to update the cache immediately by signaling vascd with SIGHUP or by executing `vastool flush`, which will cause vascd to reload all of it's information at the next NSS request..

`-n principal` The principal name that vascd will authenticate as. There must be a valid keytab entry in the vas.keytab file for the principal. By default `vascd` will authenticate as the host principal created during `vastool join`.

`-t timesync-interval` vascd will operate as a time syncronization agent if, on start up, vascd detects that no other process has bound the NTP port (123). If the NTP port is not bound, vascd will issue SNTP requests to the host that was configured with the `vastool join` or `vastool configure realm`. The `-t` option allows the system administrator to specify the frequency (in hours) that time syncronization occurs. The default is every 12 hours. If the timesync-interval is set to 0 vascd will not operate as a timesync agent.

If a specialized NTP daemon is bind to syncronize time it is crucial that this daemon be started *before* vascd. This way the NTP port will be bound when vascd starts. vascd will not operate as a timesync agent and there will be no conflicts. If, for some reason, vascd must be started before the NTP daemon, then the `-t` option should be set to zero to disable vascd timesync and avoid what would otherwise be time syncronization thrashing between the specialized NTP daemon and vascd.

This option can also be specified in `/etc/opt/vas/vas.conf` in the "[vascd]" section. An example is:

[vascd] timesync-interval = <hours>

`-r realmscache-sync-interval` `vascd` will rebuild the realms cache periodically. This cache is used to reduce the amount of DNS traffic that is needed in order to discover all of the services in the Active Directory Domains- `vascd` will rebuild this cache periodically to get any updates that have been made. Setting this option to 0 will disable the realms cache syncing. By default this is done every 24 hours.

This option can also be specified in `/etc/opt/vas/vas.conf` in the "[vascd]" section. An example is:

[vascd] realmscache-sync-interval = <minutes>

Appendix C

vastool Manual Page

vastool

A command line administration tool for use with pam_vas, vascd, and nss_vas.

`vastool -h -v -u` *`username`* `-w` *`password`* `-s` *`vastool_command vastool_command arguments`*

Description

`vastool` is a command line program that allows you to configure `vascd`, `pam_vas`, and `nss_vas`; access information stored in Active Directory; and to store information in Active Directory. `vastool` is located at `/opt/vas/bin/vastool`. It has been designed to be script-friendly and to allow administrators to easily manage users, groups, and other information stored in Active Directory from UNIX/Linux workstations.

In order to run `vastool`, you must specify the options for `vastool`, a command to run, and the options for that specific command. The following is a list of supported vastool commands and a brief description of each command's purpose. A more detailed explanation of each command will follow later.

`attrs`	List an Active Directory object's attributes.
`configure`	Update PAM, NSS, and other configuration files.
`create`	Create a user, group, or computer object in Active Directory.
`delete`	Delete users, groups, computer objects, or NIS Map objects in Active Directory.

`flush`	Flush the `vascd` cache.
`group`	Add or remove users from Active Directory groups.
`join`	Configure the system to use `pam_vas` for authentication, `nss_vas` for NSS information for users and groups, adds a computer object to Active Directory, and starts `vascd`.
`kinit`	Perform kinit functions - obtains Kerberos ticket(s) for service(s).
`klist`	Perform klist functions - lists the Kerberos tickets stored in the calling user's credentials cache.
`kdestroy`	Perform kdestroy functions- destroys all existing tickets in the calling user's credentials cache.
`license`	Install a license for the installation.
`list`	List users and groups in Active Directory, along with their Unix account information.
`load`	Import users and groups into Active Directory from a file that follows the format of `/etc/passwd` or `/etc/group`.
`nis-import`	Load NIS Maps into Active Directory.
`passwd`	Change your password or reset another user's password in Active Directory.
`realms`	Detect the realms (domains) on your network and the servers providing LDAP and Kerberos services for those realms.
`timesync`	Query and syncronize system time with Active Directory or other specified time server.
`unconfigure`	Update PAM, NSS, and other configuration files to not use the VAS components.

`unjoin`	Configure the system to not use the VAS client for authentication and for NSS, and then removes the computer object from Active Directory.
`ypcat`	Provides functionality similar to ypcat- prints out the contents of NIS Map objects stored in Active Directory

`vastool` Options

These are the options you can pass to `vastool`. They must be specified before the command name.

`-h [command]`	If no command is specified, it shows the `vastool` usage and a list of available commands. If a command is specified, it shows the usage for that vastool command.
`-v`	Print out the `vastool` version and exit.
`-u principal`	Sets the principal name to authenticate as when the vastool command needs to access Active Directory. If the caller has root access, "host/" can be specified and vastool will authenticate as the computer object that `vastool` is running on. If `-u` is not used, then `vastool` will authenticate as the calling user, and will attempt to reuse Kerberos tickets from the user's credentials cache. If `-u` is specified, then no existing credentials cache will be used, and new tickets obtained will not be saved to disk.
`-w password`	This option allows you pass in a password on the command line. Please note that this is a security hole in a production environment, and should never be used, except in testing environments.
`-s`	This option will cause `vastool` to not prompt for any initial passwords, but instead read them from stdin. This allows you to use some vastool commands in a non-interactive mode.

Vastool Command Synopsis

The following is a detailed description of all the available `vastool` commands. Their usage descriptions, a detailed explanation of their purpose, how they work, and examples are included for each command.

vastool attrs

`vastool attrs` can be used to view attributes stored on objects in Active Directory. These attributes are the LDAP attributes on the objects in Active Directory.

`vastool` *`vastool_options`* `attrs -u -s -g -d` *`objectname attribute Name`*

The objectname parameter will be interpreted differently according to the flag used. The following list explains how each flag works.

`-u`	Interprets objectname as a user name. This is the default behavior if no flag is specified.
`-s`	Interprets objectname as a service principal name.
`-g`	Interprets objectname as a group name.
`-d`	Interprets objectname as an LDAP distinguished name. This allows you to view the attributes on any object in Active Directory.

`-u` will cause `vastool` to interpret the name as a user name. This is also the default behavior if no flags are specified. However, if the user name starts with "host/", then the name will be interpreted as a service principal name. `-s` will interpret the name as a Kerberos service principal name. `-g` will interpret the name as a group name.

Following is an example of getting the home directory for the user john, getting the last time the computers container was modified, and getting a list of the members of the eng group.

```
vastool attrs john unixHomeDirectory

vastool attrs -d "CN=computers,DC=example,DC=com" whenChanged

vastool attrs -g eng member
```

vastool configure

`vastool configure` can be used to modify your system's PAM, NSS, and your Kerberos realm configuration. This command must be run as root.

```
vastool vastool_options configure realm realm_name [servers...] | extra-realm realm_name [servers...] | computer-name name | samba [smb.conf path] [workgroup] | nss | pam [service...]
```

Note that most of these command are for advanced usage only. The `vastool join` will automatically perform these steps for you.

Configuring the realm will modify `/etc/opt/vas/vas.conf` to use the given realm_name as your default realm. If a list of server names is passed in, these servers will stored as the servers for the given realm. In Active Directory terms, the realm will be the domain name of the domain this computer will be a member of. `vastool configure extra-realm` can also be used to configure other domains if you need to support multiple servers in your Active Directory tree. This will add information for these realms, but it will not make the new realm the default realm.

Configuring samba will integrate your local samba configuration with the VAS configuration. You will be prompted for the location of your smb.conf file and your workgroup/domain (these can optionally be specified as arguments on the command line), then the smb.conf will be updated to work with Active Directory and VAS. Samba's password for authenticating as the computer will be synchronized with the VAS computer password. This allows your Samba server to run with domain security alongside VAS on the same machine. *Note:* only samba version 2.2.8 is supported at this time.

If you ever reset the computer password, then you will need to run `vastool configure samba` to synchronize the computer password between Samba and VAS again, otherwise Samba will no longer be able to authenticate as the computer.

Configuring NSS will modify the passwd and group entries in the `/etc/nsswitch.conf` to include an entry for `vas`, (the nss_vas NSS module), after the `files` NSS module. You can manually edit `/etc/nsswitch.conf` to put `vas` in front of `files`. This change will cause any username that has both a local account and a VAS account to be resolved to the VAS account. By default, local accounts will override Active Directory accounts. At this time, no other NSS subsystems besides passwd and group are supported.

Configuring PAM will modify either `/etc/pam.conf`, or the files located in `/etc/pam.d/` to use the `pam_vas` PAM module. If no service names are specified, then all existing services, including the default "other", will be configured to use pam_vas. For more information on configuring and customizing pam_vas, see pam_vas (5).

If you are configuring a computer whose hostname will not always match up with the name of the computer object in Active Directory that represents your computer, you must use `vastool configure computer-name` to specify the name of the computer as it appears in Active Directory. Otherwise, your computer will not be able to communicate with Active Directory. When using `vastool join` you can specify this name with the `-n` option.

Following is an example configuring the example.com realm, configuring the example.com realm and using a special name for the host computer, configuring NSS, configuring all PAM enabled services, and configuring the login, telnet, and ssh services to use VAS.

```
vastool configure realm example.com

vastool configure extra-realm sub.example.com server.sub.exam-
ple.com

vastool configure computer-name mycomputer

vastool configure nss

vastool configure pam

vastool configure pam login telnet sshd
```

If you are configuring a Solaris 8 or 9 system, you will need to use the `-g` option so that vastool will configure the pam.conf file to better handle authentication. The VAS mod-

ule will prompt all users for their passwords instead other modules. This allows VAS to do better checking for authentication and improve security. Using this option causes vastool to modify the pam.conf file by commenting out, altering, adding, or removing pam options for other modules in the file besides those for the VAS line entries.

```
vastool configure pam -g

vastool configure pam -g login telnet sshd
```

vastool create

`vastool create` can be used to create a user, group, or computer object in Active Directory. This command must be run as root when creating a computer object.

`vastool vastool_options create -g -c` *`container`* `-p` *`password`* `-i` *`info`* `-x -e` *`name`*

`vastool create` will interpret the specified name, and then create different types of objects according to the format of the name. To create a computer object, the name must be formatted as "host/FQDN" where FQDN is the fully qualified domain name of the computer. Note that computer object creation is normally handled by `vastool join`- the create computer option is provided for advanced users. To create a computer object for the local computer and use it's hostname, just specify "host/". If the name does not start with "host/", the name will be interpreted as a user name if `-g` is not specified. The `-g` option will cause the name to be interpreted as a group name, and a group object will be created.

Note that all user, group, and computer object creation can only be done in the Active Directory domain your computer is a member of.

The new user, group, or computer object will be located in the default users or computer containers in Active Directory. You can create the new object anywhere in the Active Directory tree by using the `-c` option to specify the distinguished name (DN) of a container to create the object in.

When creating a user or a group, the new user or group will not be Unix enabled by default, unless you use the `-i` option. By specifying an information string with `-i`, you can specify the information for the user's/group's Unix account. This string should be formatted as an entry in `/etc/passwd` or `/etc/group`. When creating a user, you

can specify that user's new password with the `-p` option. Unless the `-x` option was specified, the newly created user will also be forced to change their password during their first login.

It is possible to Unix enable an existing user or group. To do this, use the `-e` option. You must specify an information string with the `-i` when using `-e`. Note that this will not create the user or group- it will only add the Unix/Linux attributes to an existing user or group. It will also override those attributes if they already exist for the user or group, so use caution when using `-e`.

`vastool create` must be run as root when creating a computer object for the computer that vastool is running on. Part of the computer creation process is setting the computer object's password so that vascd can authenticate to Active Directory. This key is stored in a secure file that can only be accessed by root at `/etc/opt/vas/host.keytab`. `vastool create` does not need to be run as root when creating users or groups.

Also, when creating a computer object without using just "host/" as the name, you will need to also run `vastool configure computer-name {name}`. This is just the first name part of the FQDN that should be passed in the "host/FQDN" string.

The user you authenticate to Active Directory as must have the appropriate administrative privileges in order to create the new user, group, or computer object. Computer object creation can be delegated to other users besides Administrators. To accomplish this, the Active Directory administrator must initially create the computer object in Active Directory using `vastool create` or `vastool join`. Then, the administrator can give another user rights to reset that computer object's password. This will allow that user to reinstall VAS without the administrator. In this situation, vastool will recognize that the computer object for this computer already exists, and will just try to reset the computer's password.

Following are two examples of user creation, two examples of group creation, and two examples of computer creation.

```
vastool -u admin create -c "OU=Engineering,DC=example,DC=com"
jdoe

vastool -u admin create -i "jdoe:x:1001:1000:John Doe:/home/
jdoe:/bin/bash" jdoe
```

```
vastool -u admin create -g marketing

vastool -u admin create -g -i "marketing:x:1005:john,mary" mar-
keting

vastool -u admin create host/

vastool -u admin create -c "OU=Engineering,DC=example,DC=com"
host/
```

vastool delete

`vastool delete` can be used to delete users, groups, computer objects, and NIS Map objects in Active Directory. This command must be run as root when deleting the computer object

`vastoolvastool_optionsdelete-g-n-dname...`

You must have the appropriate administrative rights to delete objects in Active Directory. The names passed in will be interpreted according to the options used. The `-g` option will interpret the names as group names and the Active Directory objects for those groups will be deleted. `-n` will interpret the names as NIS Map objects. `-d` will interpret the names as LDAP Distinguished Names.

If no options are specified, then the names will be interpreted as user names, unless the names start with "host/", then the names will be interpreted as computer names. To delete the computer object for the computer `vastool delete` is running on, use "host/" as the computer name. You must have root access to delete the computer object for the local computer, since the key for the computer is removed from `/etc/opt/vas/vas.keytab`.

Following is one example of group deletion, one example of user deletion, one example of NIS Map deletion, two examples of computer deletion, and an example of deleting an ldap object.

```
vastool delete -g eng

vastool delete jsmith

vastool delete -n hosts
```

```
vastool -u admin delete host/

vastool delete host/server.example.com

vastool delete -d "CN=Foo,DC=example,DC=com"
```

vastool flush

`vastool flush` can be used to clear the vascd cache. This command must be run as root.

```
vastool vastool_options flush-raccounts | auth | keytab |
users | groups
```

Flushing the accounts cache will remove all cached user, group and NIS Map information. This will force vascd to do complete lookups the next time it receives any NSS IPC requests. Flushing the auth cache will remove all cached user passwords. These are stored as SHA1 hashes in a secure file that is only accessible by root. Flushing the keytab will delete the VAS keytab file. Flushing the users cache will delete all cached user account information, and flushing the groups cache will delete all cached group information.

The users and groups cache will be regenerated after being flushed, unless the `-r` option is specified.

If you do not specify an argument to `vastool flush`, then the accounts and auth arguments will be implied, and all user/group account information, NIS Map information, and cached passwords will be deleted.

On systems that do not support PAM and NSS, then as part of flushing the users and groups, they will also be unmerged from the `/etc/passwd` and `/etc/group` files as well. If the caches are regenerated, then the users and groups will be merged back in.

vastool group

`vastool group` can be used to modify group membership lists.

```
vastool vastool_options group group_name add | delname...
```

You must have enough administrative privileges to modify the group object in Active

Directory. Using the `add` option will add the listed users to the specified group. The `del` option will remove the specified users. Note that these changes will only appear on the Linux/UNIX systems if the group and users used in the command have been Unix-enabled. The changes will occur in Active Directory regardless of whether or not the users and groups have been Unix-enabled.

Please note that if the specified members do not already exist in Active Directory, then those names will not be added or removed from the group membership list.

Following is an example of adding the jsmith user to the eng group, and removing the jsmith user from the eng group.

```
vastool group eng add jsmith

vastool group eng del jsmith
```

vastool join

`vastool join` is a convenient command that wraps all of the necessary steps to configure the VAS client on a computer into one. It configures your realm, creates a computer object in Active Directory, and configures PAM and NSS. This command must be run as root.

`vastool vastool_options join -c` *`container`* `-n` *`computer_name`* `-f realm_name servers...`

`vastool join` will internally call `vastool configure realm` to configure the realm, `vastool create` to create a computer object in Active Directory for the computer, `vastool configure pam` to configure the PAM subsystem, and `vastool configure nss` to configure the NSS subsystem. The `vascd` client daemon will then be started. For more information on each of these steps, see their respective sections in this document.

The `-c` option will allow you to specify a container where your new computer object will be created. If that is not specified, then the computer object will be created in the default computers container. The `-n` option allows you to specify a different name for the computer object than what vastool would generate from your hostname. If `-n` is used, then a special parameter will be set in `/etc/opt/vas/vas.conf` to denote to all the VAS components which name should be used for the computer object. You can

also set this name by running `vastool configure computer-name`. The computer name specified with the `-n` option should not be in the "host/FQDN" form- it should just be a name which is an alternative to the system's hostname. It also must not contain any '.' characters- this name is not the FQDN.

The `-n` option implies that the `-f` is set, and it will override the settings for any existing computer objects with the same name. `-n` is most useful when an administrator sets up delegated computer creation.

Following the options, you must specify your Kerberos realm, which will be the same as your Active Directory domain. The services for this domain will be automatically detected through DNS and LDAP lookups. If you do not intend to use DNS, or it is not configured, you must specify a list of servers for your domain after the realm name.

If a computer object already exists in the directory for the computer name you are trying to use, an error will be reported. To override the existing computer object, use the `-f` option. In this case, the computer object's authentication key will be reset. Any other systems authenticating as that computer object will no longer be able to authenticate after the authentication key is reset.

Following is an example of `vastool join` using all of the defaults, then an example of joining a computer with a name other than it's hostname into a non default container in an environment where DNS is not properly configured.

```
vastool -u admin join example.com server.example.com

vastool -u admin join -c "OU=Testlab,DC=example,DC=com" -n
test_server example.com server.example.com
```

vastool kinit

`vastool kinit` can be used to obtain Kerberos tickets.

```
vastool vastool_options kinit service
```

If no arguments are specified, then the Kerberos TGT is obtained if it is not in the user's ticket cache. If services are specified, then those tickets will be obtained. If the `-u` vastool option was not specified, then these tickets will be stored in the user's ticket cache.

`vastool kinit` can be used to debug problems with Kerberos authentication. For example, to test if `vascd` can authenticate to Active Directory, you would run as root:

```
vastool -u host/ kinit
```

Using the vastool -s option, you can use the vastool kinit command as an authentication API from scripts and other programs which do not use PAM or the VAS API. This can be done by running:

```
vastool -u jdoe -s kinit
```

and then writing jdoe's password to stdin of the vastool process. The exit code of the vastool process will be 0 on a successful authentication, and 1 if authentication failed.

If you see any error messages, then vascd could not authenticate to Active Directory.

vastool klist

`vastool klist` can be used to list all of the tickets currently in the calling user's Kerberos ticket cache.

```
vastool klist
```

The tickets in the user's ticket cache are printed to stdout. They will show the name of the service each ticket is for, the time the ticket was issued, and the time the ticket will expire. The ticket cache will be stored as a file owned by the user with permissions of 0600 at $HOME/.krb5cc or in /tmp/krb5cc_{user's uid}.

vastool kdestroy

`vastool kdestroy` will destroy all of the tickets that are in the calling user's Kerberos ticket cache.

```
vastool vastool_options kdestroy
```

A user's Kerberos ticket cache is a file owned by the user with permissions of 0600 that will be either at $HOME/.krb5cc or in /tmp/krb5cc_{user's uid}. Normally, the user's Kerberos TGT is stored there along with any other tickets that have been obtained. These tickets can all be cleared with `vastool kdestroy`.

vastool license

`vastool license` will install a license key and print out the current license information. This command must be run as root when installing a license key.

`vastool vastool_options license -q -s -f` *`file serial_number key`*

`vastool license` can be used to look up the installed license information with the `-q`. It will report how many users the installed license is good for.

To install a license, you must omit the `-q` and supply the serial number and key. This can be done by supplying the serial number key as command line arguments, by using the `-s` to pipe them into the process's stdin, or by placing the serial number and key in a text file on the same line, where the serial number is first, with a space, and then the license key.

After installing a license, the users cache will be flushed to ensure that correct number of users is loaded into the cache.

Following is an example of installing a license key, and an example of querying the current license information.

```
vastool license -q

vastool license NOTA-A-REAL-SERIAL-NUMBER PUT-YOUR-KEY-HERE
```

vastool list

`vastool list` can be used to list the users and groups that are stored in Active Directory.

`vastool vastool_options list -a -l -f -c users | user {username} | groups | group {groupname}`

By default, `vastool list` will not directly use LDAP, but will use vascd to lookup the group and user accounts. This allows `vastool list` to take advantage of vascd's information cache. The `-l` option will force `vastool list` to directly use LDAP and bypass the vascd cache.

The `-a` option will list all the users or groups in Active Directory, not just the Unix enabled ones. This will automatically also set the `-l` option and force `vastool list` to directly contact Active Directory over LDAP. When `-a` is used, only the name attributes will be listed. Not using `-a` will list all of the attributes for Unix enabled groups and users.

The `-f` option will force `vascd` to update it's cache immediately, without respecting the ldap blackout period. The `-c` will not send an update IPC to `vascd`, but just read directly from the cache. This may result in old data that is incorrect.

`vastool list user` and `vastool list group` will only list the user/group specified. The `-l` still applies and will force LDAP lookups. The `-a` will search all users and groups, not just the unix enabled ones.

Following are examples of listing the Unix enabled groups that `vascd` knows about, the Unix enabled users `vascd` knows about, and listing the Unix enabled users obtained directly from Active Directory.

```
vastool list groups

vastool list users

vastool list -l users
```

vastool load

`vastool load` can be used to import existing users and groups.

`vastool vastool_options load -f` *`file`* `-c` *`container`* `-p` *`password`* `-x -e -r users | groups`

`vastool load` will read from a file if the `-f` option is specified, otherwise it will read from stdin. The input must follow the format of `/etc/passwd` if loading users. If loading groups the input must be formatted like `/etc/group`. You can load the users or groups into any Active Directory container using the `-c` option. Otherwise, they will be created in the default users container.

Please note that existing passwords cannot be imported into Active Directory. If `-p` is used to specify a password when loading users, all of the new users will have their pass-

words set to the specified password. Otherwise, a random password made up of alpha-numeric characters will be generated for each user. These generated passwords will be stored in a file the administrator can use to notify the new user's what their password is. Unless the `-x` option was specified, the newly created users will be forced to change their password during their first login. Passwords cannot be set for groups, and the `-p` option is ignored when loading groups.

Any errors will be logged to stderr and a log file whose location will be printed out at the end of the `vastool load` operation. It is very important to ensure that the UID's and GID's specified for the users and groups being imported do not conflict with existing users and groups already in Active Directory. The import process does not check for conflicts before creating the new users and groups.

When importing groups that have members, those members need to be created first. For example, for the following group entry: `group1:x:1400:user1,user2,user3`, you should first import user1, user2, and user3. Otherwise, when the group object is created in Active Directory none of the members will be stored in the group. This is due to the fact that group membership lists in Active Directory are stored as lists of distinguished names, and those DN's cannot be looked up if the group members do not already exist.

Once the load is complete, the vascd user or group cache will automatically be updated to get the newly create users or groups. This can be disabled with the `-r` option.

There may be situations where you already have Active Directory accounts for the Unix/Linux users and groups you are importing. In this situation you will want to use the `-e` option. This will cause vastool to set the Unix/Linux attributes for existing users and groups that are being imported. Note that this will not create users or groups; they must already exist in Active Directory. An error will be reported for each user or group that is the list being imported that does not already have an account in Active Directory.

The following is an example of importing a file of users into a specific Active Directory container, and setting all of their default passwords to "change.me".

```
vastool load -f /tmp/newusers.txt -p change.me -c OU=eng,DC=exam-
ple,DC=com users
```

Note that after migrating Unix users and groups into Active Directory, you will need to remove those user accounts from the local `/etc/passwd` and `/etc/shadow` files,

and remove the group accounts from `/etc/group` on every Unix machine where they were previously.

vastool nis-import

`vastool nis-import` can be used to create NIS Map objects in Active Directory.

`vastool vastool_options nis-import -o -f` *`filename`* `-c` *`container`* `NIS Map Name`

A NIS Map normally represents some standard configuration file such as `/etc/hosts`. `vastool nis-import` can read in a file specified with the `-f` option, or if `-f` is not specified, then the NIS Map contents will be read from stdin. The `-c` option allows you to specify the Active Directory container to create the NIS Map object in. If `-c` is not specified, the NIS Map object will be created in the default computers container. The container that the NIS Map is created in is significant since only the computers in that same container will see the NIS Map. This behavior allows you to emulate the behavior of NIS domains by enabling you to distribute maps of the same name to different sets of computers.

If the NIS Map already exists in Active Directory, you can overwrite it's contents using the `-o` option.

The NIS Map Name argument is the name of the NIS Map object. This should normally be the name of the file you are importing. For example, if creating a NIS Map object for `/etc/hosts`, you would use "hosts" as the name.

The following is a list of the currently supported NIS Maps files with the map names that use those files:

`hosts`	hosts.byaddr, hosts.byname
`ethers`	ethers.byaddr, ethers.byname
`aliases`	mail.aliases
`protocols`	protocols.byname, protocols.bynumber
`services`	services.byname, services.byservice, service.byservicename

`rpc`	rpc.byname, rpc.bynumber
`netgroup`	only netgroup is supported

Additional types of NIS Maps can be created and loaded into Active Directory- you are not limited to the supported types of NIS Maps listed above. Any type of text file can be distributed using this mechanism. In order to use a custom NIS Map type, you must first create and import the NIS Map into Active Directory using `vastool nis-import`. You must also create a parser script that can parse the NIS Map contents. This parser script should be installed under `/opt/vas/libexec/nismaps` with the same name as the map name that is passed to `/opt/vas/bin/ypcat`. The parsing script can be any script or program- the only requirement is to handle the following command line format: [-k] filename. If -k is specified then the parser should output the nismap keys. The filename will be a path to a file that contains the NIS Map contents that should be parsed. For more examples of how to write a parser script, see the nismap parser scripts that are installed under `/opt/vas/libexec/nismaps`.

Please note that you can not import a NIS Map for the passwd or group maps. Users and groups must be imported into Active Directory with `vastool load`. `vastool ypcat` will use the cached information for users and groups from `vascd` when ypcat'ing the passwd or group NIS Maps.

The following is an example of importing a NIS Map for the hosts file.

```
vastool nis-import -f /home/jsmith/nisMaps/hosts hosts
```

vastool passwd

`vastool passwd` can be used to change your password, or to reset another user's password if you have enough administrative privileges.

```
vastool vastool_options passwd -c user_name
```

On some platforms, such as United Linux based Linux distributions and Solaris 8/9, the system passwd change utility does not work correctly when the `vas` NSS module is listed in `/etc/nsswitch.conf`. VAS users will not be able to change their passwords using the system `passwd` command. `vastool passwd` can be used by VAS users as a workaround.

If no user name is specified, then the calling user's (or the user specified with the `-u` vastool option) password will be changed. If a user name is specified on the command line after the vastool passwd command, then `vastool passwd` will attempt to set the specified user's password. To set (or reset) passwords you must have administrative rights. `vastool passwd` changes passwords in Active Directory using the Kerberos change password protocol.

Note that these two modes of password changing are fundamentally different. When changing a password, you must authenticate as the user whose password is being changed. When setting another user's password, you must authenticate as an administrative user that has privileges to reset that user's password. Changing passwords requires knowledge of the user's current password; setting passwords does not.

The following is an example of the calling user changing their own password:

```
$ vastool passwd
```

The following is an example of the calling user changing the bsmith user's password (Note that the calling user must know bsmith's current password):

```
$ vastool -u bsmith passwd
```

The following example shows the calling user resetting the bsmith user's password. Note that the calling user does not need to know bsmith's password, but must authenticate as the calling user's identity, and must have administrative rights to set bsmith's password.

```
$ vastool passwd bsmith
```

It is possible to use `vastool passwd` in a non-interactive mode by using the `-s` vastool option. This will cause all password prompts to be satisified by reading from stdin. This allows web applications, scripts, and other applications to use vastool passwd for users. The application must get the necessary information from the user and then supply that information to the stdin of the `vastool passwd` command.

When using the vastool `-s` option while changing a user's password, the caller must write the user's current password to stdin followed by the '\n' character, then write the new password followed by the '\n' character, then write the new password again followed by the '\n' character.

The following example shows the correct options to change the bsmith user's password and supply the necessary information to the stdin of the process.

```
$ vastool -u bsmith -s passwd
```

The process should then write bsmith's current password, the new password, and the new password again to the stdin of the command.

The `-c` option allows the root user to set a user's cached password. This is useful mainly for debugging purposes. It does *NOT* change the user's password in Active Directory, only in the local cache.

One important note is that when using the `pam_vas` PAM module with disconnected authentication enabled, then `vastool passwd` will not be able to sync up the user credential cache with the new password for the user since it will not have root access on the system. On systems where the `passwd` utility correctly works, the user's changed password will be synced up correctly in the user credential cache.

vastool realms

`vastool realms` can be used to query the network for the Active Directory domains and also will detect the domain controllers on the network. This information can be stored in the realms cache- to do this you must be root.

```
vastool vastool_options realms { find {root | srvs |
domains} [realm] | site {names | srvs | local | subnet} |
cache {list | update | update-realm [realm] } | flush
[realm] | toconf}
```

`vastool realms find` will detect various settings, services, and domains on your network. `realms find root` will detect the forest root. `realms find domains` will detect all of the domains in the entire forest. `realms find srvs` will find the services for a given domain.

`vastool site names` will detect all of the configured Active Directory sites in your forest. `vastool site srvs` will list all of the domain controllers and which site they are in. `vastool site local` will detect what site the local VAS client should belong to. `vastool site subnet` will determine the subnet of your client.

`vastool realms cache list` will print out the service and domain information that is stored in the realms cache. This cache is used to decrease the amount of DNS traffic that must be performed by the VAS client. `vastool realms cache update` will detect all of the domains and services in the entire forest and store that information in the realms cache. `cache update-realm` will detect the services and update the cache for only the given realm.

`vastool flush` will clear all of the entries from the realms cache, and reload it. If a realm name is passed, then only the entries for the given realm will be reloaded.

When using VAS with MIT Kerberos compatible applications, you will need to symlink `/etc/krb5.conf` to `/etc/opt/vas/vas.conf`. Before doing that though, you need to make sure that all of the realm information in the VAS realms cache is stored in `/etc/opt/vas/vas.conf`. This can be done with `vastool realms toconf`. This will make a realms entry for each realm VAS knows about in the [realms] section. This way the MIT-compatable applications will be able to work by reusing the VAS configuration information.

vastool timesync

`vastool timesync` can be used to query the time on the Active Directory server and syncronized the system clock with the Active Directory server. In order to set the system clock this command must be run as root.

```
vastool vastool_options timesync [timeserver] [-q]
```

Running `vastool timesync` without specifying a timesever will automatically use the Active Directory server that was configured using the `vastool join` or the `vastool configure realm`commands.

Use the `-q` option to query the server's time with out setting the system clock.

vastool unconfigure

`vastool unconfigure` can be used to remove the VAS configuration from the NSS and PAM subsystems. This command must be run as root.

```
vastool vastool_options unconfigure nss | pam [service...]
```

Running `vastool unconfigure nss` will remove the vas settings in `/etc/nsswitch.conf`. Running `vastool unconfigure pam` without specifying any service names will remove the VAS configuration from all PAM enabled services. If service names are specified, only those specified services will have their VAS configuration removed.

Running `vastool unconfigure` with no arguments will unconfigure both the NSS configuration and all PAM enabled services from using VAS.

The following are examples of removing the VAS configuration from `/etc/nsswitch.conf`, disabling VAS authentication from the ssh server, and disabling VAS authentication for all PAM enabled services.

```
vastool unconfigure nss

vastool unconfigure pam sshd

vastool unconfigure pam
```

vastool unjoin

`vastool unjoin` is a convenient command that wraps all of the steps to remove your computer object from Active Directory and to remove the VAS configuration from the NSS and PAM subsystems in one step. This command must be run as root.

`vastool vastool_options unjoin -n` *`computer_name`*

`vastool unjoin` will first remove the NSS and PAM configurations with `vastool unconfigure`. It will then prompt you for your administrative password in order to delete the computer object for your machine in Active Directory. As part of this process, `vascd` will be stopped.

If you used the `-n` option in the `vastool join` process, then you need to specify that same name that you used in the join in the `vastool unjoin` process.

The following are examples of unjoining the machine where the computer object is named after the system hostname, and unjoining the machine when the computer object name does not match the hostname.

```
vastool -u admin unjoin
```

```
vastool -u admin unjoin -n computer_name
```

vastool ypcat

`vastool ypcat` can be used to view NIS Map objects that are stored in Active Directory. It does not use the NIS protocol, but uses the VAS client daemon, `vascd`, to access the NIS Map objects over Kerberos/LDAP. It is functionally equivalent to the standard ypcat.

```
vastool vastool_options ypcat -k -x NIS Map Name
```

The `-k` option will print out the NIS Map keys. Without `-k`, only the values are printed out. The `-x` option will print out the NIS Map nickname translation table that `vastool ypcat` uses. These NIS Map objects must be imported into Active Directory using `vastool nis-import`.

You do not need to authenticate to Active Directory when using `vastool ypcat`. `vastool ypcat` will send an IPC to vascd who will get the NIS Map contents. `vascd` will be able to use it's LDAP cache, making the ypcat process very efficient.

The following are examples of viewing the map nickname translation table, viewing the hosts NIS Map with it's keys.

```
vastool ypcat -x

vastool ypcat -k hosts
```