Kerberos White Paper

Executive Summary

This white paper provides a high-level description of the Kerberos protocol. The paper includes detailed information about important concepts and features of Kerberos authentication. The first section provides basic information about Kerberos authentication. Following this introduction to the protocol are several sections with details of how HP has implemented the Kerberos authentication protocol.

HP-UX supports the following Kerberos suite of products on the on the HP-UX 11.0, 11i v1, and 11i v2 operating systems:

- Pluggable Authentication Module Kerberos (PAM-Kerberos)
- Kerberos Client Software
- HP Kerberos Server
- Generic Security Service Application Programming Interface (GSS-API)
- Secure Internet Services (SIS)
- HP-UX Secure Shell (SSH)

The subsequent sections of this document discuss these in detail.
The paper concludes with a brief discussion of Kerberos protocol interoperability with other systems.

Problem Statement

The Internet is a vast place that connects millions of people from all corners of the globe to each other everyday. In such a network, information can be lost, stolen, corrupted, or misused. Another drawback of the internet is that it is difficult for individuals to confirm their identity to one another. Confidentiality is very important for some types of information, such as information related to banking and medical. It is therefore important that a user, who wants to access this kind of information online, be able to confirm that the user is who he/she claims to be. This process is called authentication. Kerberos plays a major role in authentication.
Traditionally, a process was set in place called *Authentication by Assertion.* Authentication by assertion works as follows:

When a user runs a program that accesses a network service, the program (called the client) asserts to the service that it is running on behalf of the user. This provides a very low level of security. Consider the example of Berkeley rlogin. If a user rlogins to an account under his own name, but on another machine, and if the user's .rhosts is set correctly, the rlogin program will assert the user's identity to the rlogin daemon on the remote machine, and the daemon does not require a password for login. This can become disastrous if an attacker is somehow able either to convince the rlogin program that he/she is the legitimate user, or to rewrite a mutant version of rlogin asserting that identity to the remote machine.

An alternative to this situation is to require a user to enter a password each time he/she accesses a network service. This is a very time-consuming process, and it is insecure when users access services on a remote machine. When a user is logged on to a remote machine and then logs in from there to another remote machine, the password travels unencrypted through the network.

Kerberos fixes these problems because it provides single-sign-on, which lets a user log in to a system and access multiple systems or applications without the need to enter the user name and password multiple times. In addition, Kerberos is designed so that entities have to authenticate themselves by

demonstrating possession of secret information. In this manner, Kerberos solves traditional problems involved with authentication.

Historical Evolution of Kerberos

The name Kerberos comes from Greek mythology; Cerberos was the three-headed dog that guarded the entrance to Hades. Kerberos is a network authentication protocol developed by MIT (Massachusetts Institute of Technology) as part of Project Athena, which started in 1983 when MIT decided to integrate network computers as part of its campus curriculum. The goals of Athena were the integration of a SSO (Single Sign-on), networked file systems, a unified graphical environment, and a naming convention service.

Kerberos has since evolved into a strategic security standard that provides secure authentication services to users, applications, and network devices, which eliminates the threats caused by passwords being stored or transmitted across the network. Additionally, Kerberos provides data integrity to ensure messages are not tampered with on the network and message privacy (encryption) to ensure messages are not visible to eavesdroppers on the network.

The Kerberos model is partly based on Needham and Schroeder's trusted third-party authentication protocol. Versions one through three never reached outside MIT, but version 4 was (and still is) quite popular, especially in the academic community. It is also used in commercial products like the AFS filesystem.

Why Kerberos?

The problem statement discussed the problems associated with traditional authentication methods and, in particular, how passwords are vulnerable because they travel unencrypted over the network. Password-based authentication is also inconvenient; users do not want to enter a password each time they access a network service.

Kerberos is designed to eliminate the need for users to demonstrate possession of private or secret information (the password). Instead, Kerberos disseminates this information. Kerberos Server lets entities authenticate themselves, without transmitting their passwords in clear text over the network.

Commonly used to secure particularly vulnerable network communications like ftp, telnet, and other widely used Internet protocols that normally transmit user IDs and passwords in clear text, Kerberos provides the "plumbing" for common authentication services. Its scalability means that Kerberos is ideal for large networks such as those used by governments, telecommunication networks, and major financial institutions.

Kerberos Basics

Kerberos uses secret-key cryptography, which lets entities communicating over networks prove their identity to each other while preventing eavesdropping or replay attacks. It also provides data stream integrity (detection of modification) and secrecy (preventing unauthorized reading) using Data Encryption Standards such as DES, 3DES, and AES.

Kerberos is based on the concept of a trusted third party that performs secure verification of users and services. In the Kerberos protocol, this trusted third party is called the key distribution center (KDC).

Kerberos is used to verify that users and the network services they use are really who and what they claim to be. To accomplish this, a trusted Kerberos Server issues tickets to users. These tickets, which have a limited lifespan, are stored in a user's credential cache and can be used in place of the standard username-and-password authentication mechanism. The ticket can then be embedded in virtually any other network protocol, thereby letting the processes implementing that protocol to be sure about the identity of the principals involved.

How Kerberos Works

The Kerberos credential scheme embodies the SSO concept. Secure authentication is based on previously established initial credentials, which eliminates the need to re-key a password on multiple occasions.

A Kerberos server consists of the following elements:

- **Realm** - a user-defined administrative boundary.
- **Key Distribution Center (KDC)** - the heart of the Kerberos realm. It provides Kerberos authentication services by issuing encrypted tickets that require secret keys to decode.
- **Principal** - a unique name for a user or service stored in a KDC.
- **Tickets** - records that help a client authenticate to a server.

Under Kerberos, a client (generally either a user or a service) sends a request for a ticket to the KDC. The KDC creates a ticket-granting ticket (TGT) for the client, encrypts it using the KDC key, and sends the encrypted TGT back to the client. The client uses the TGT to obtain further service tickets, which provide the proof of the client's identity.

Users can also enable pre-authentication. When pre-authentication is enabled, a user must sign on to the KDC by providing knowledge of secret information. Once the identity of the user requesting for a ticket is confirmed, the KDC returns a set of initial credentials for the user, consisting of a ticket-granting-ticket (TGT) and a session key.

If a principal (user) needs to access any service located on a particular system, the KDC issues a service ticket for the specific service. A service ticket can be associated with one or more Kerberos-secured services on the same system. The service ticket is usually used by a client application on behalf of the user, to authenticate the user to the Kerberos-secured network service. The Kerberized client application automatically handles the transactions with the KDC. Service tickets and associated session keys are generally cached in the user's credentials cache file along with the user's TGT.

Authentication Process

The following steps describe how a client and a server authenticate each other using Kerberos. The step numbers match with the numbered arrows in Figure 1 below.

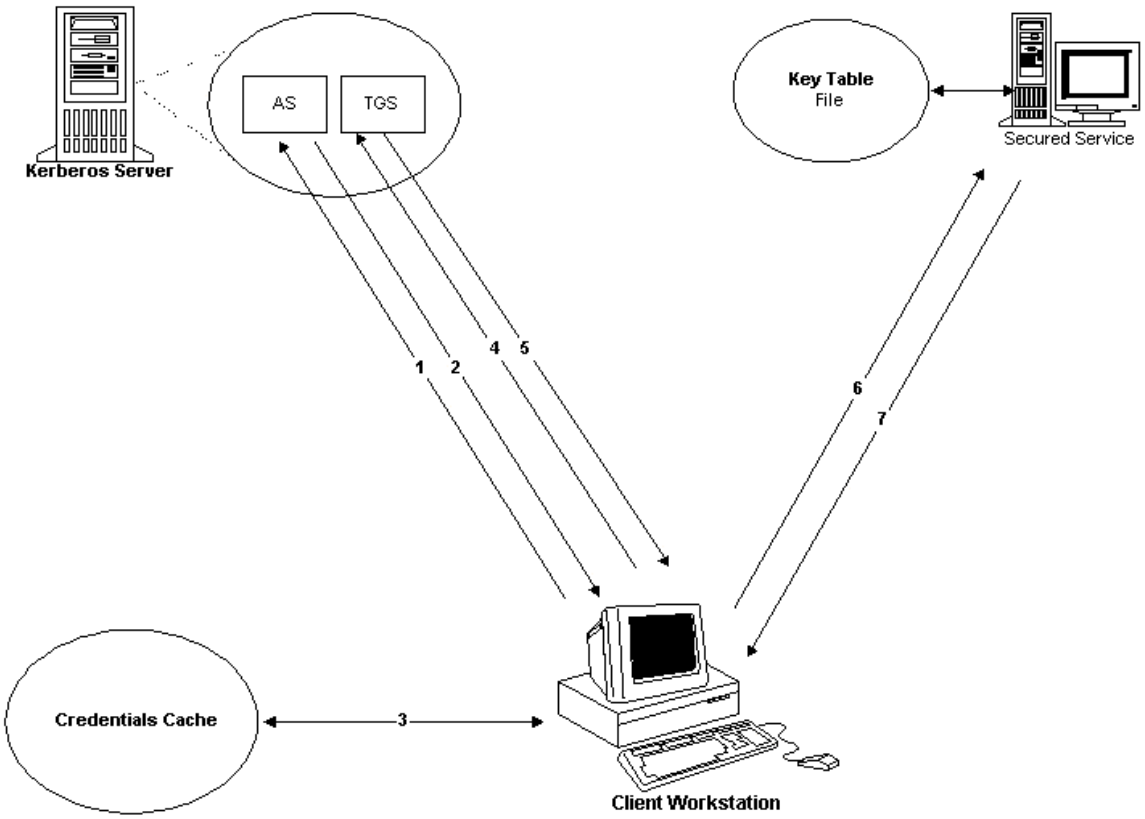

Figure 1: The Kerberos Authentication Protocol

Step 1. The user begins to use a Kerberized application by entering the user name and password. Optionally, the user can request for specific ticket flags and specify the key type to be used for constructing the secret key. The user can also accept the default, configured for the client.

The user sends the following information to the Authentication Service (AS) to obtain credentials:

- **Client**, **Server**, **T**, **N**; where
- **Client** indicates the user name, also referred to as the principal name
- **Server** indicates the Application Server
- **T** indicates the time stamp and
- **N** indicates nonce

Step 2. If the AS can decrypt the message successfully, it issues a temporary session key, which is encrypted with the user's secret key (a key derived from the user password, which is stored in the KDC), and a TGT encrypted with the TGS's secret key. The TGT contains the name of the user and a copy of the session key (a randomly generated temporary encryption key) to be used by the user and the Server for any subsequent communication.

Step 3. The user decrypts the session key. The TGT and the session key are stashed in the user's credential cache. The credentials are used to obtain tickets for each network service the principal wants to access.

This protocol exchange has two important features:

- The authentication scheme does not require that the password be sent across the network, either in encrypted form or in clear text.
- The client (or any other user) cannot view or modify the contents of the TGT.

Step 4. To obtain access to a secured network service such as `rlogin, rsh, rcp, ftp`, or `telnet`, the requesting client application uses the previously obtained TGT in a dialogue with the TGS to obtain a service ticket. The protocol is the same as used while obtaining the TGT, except that the messages contain the name of the server and a copy of the previously obtained TGT.

Step 5. The TGS returns a new service ticket that the application client can use to authenticate the service.

Step 6. The application client tries to authenticate to the service on the application server using the service ticket obtained from the TGS.

The secure application validates the service ticket using the server's service key present in the key tab file. Using this service key, the server decrypts the authenticator and verifies the identity of the user. It also verifies that the user's service ticket has not expired. If the user does not have a valid service ticket, then the server will return an appropriate error code to the client.

Step 7. (Optional) At the client's request, the application server can also return the time stamp the client sent encrypted in the session key. This ensures a mutual authentication between the client and the application server.

Kerberos Products on HP-UX

HP-UX supports the following Kerberos products. All HP-UX Kerberos products conform to the IETF specification for Kerberos Version 5 and are compliant with IETF RFC 1510.

- PAM Kerberos
- Kerberos Client Software
- HP Kerberos Server Version 3.1
- Generic Security Service Application Programming Interface (GSS-API)

PAM Kerberos (PAM-Kerberos)

The Kerberos implementation of PAM is based on RFC 86.0 of the Open Software Foundation. PAM allows multiple authentication technologies to co-exist on HP-UX.

The PAM framework allows options for account, session, password, and authentication management. PAM uses the Kerberos protocol for authentication management.

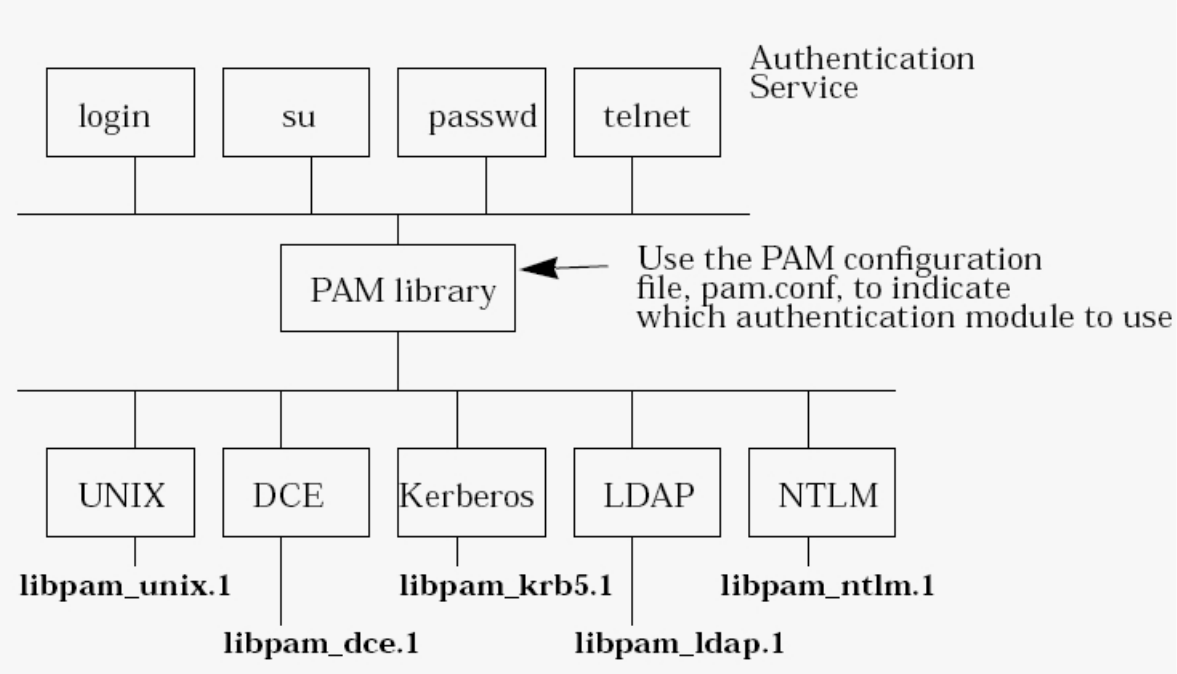

Figure 2: The PAM Library

Figure 2 shows the relationship between the PAM Kerberos library and various authentication modules that HP-UX provides. The PAM Kerberos library is one of the many authentication modules that PAM can invoke based on what is defined under the PAM configuration file: /etc/pam.conf. If PAM's authentication-management points to the shared, dynamically loadable PAM Kerberos library, PAM Kerberos is invoked for user authentication.

Kerberos Client (KRB5-Client) Software

In Kerberos, authentication takes place between clients and servers. So, in Kerberos terminology, a "Kerberos client" is any entity that gets a service ticket for a Kerberos service. A client is typically a user, but any principal can be a client (unless for some reason the administrator has explicitly forbidden a principal to be a client).

On HP-UX 11i onwards, the Kerberos utilities are part of the OS core. The Kerberos Client software consists of libraries, header files, manpages, and Kerberos utilities for implementing Kerberized client/server applications in either 32-bit or 64-bit development environment. The client libraries are based on MIT Kerberos V5 1.1.1. The HP-UX implementation of Kerberos utilities is compatible with the MIT reference implementation. The Kerberos Client libraries support encryption types such as DES, 3DES, and AES. There is a new Kerberos Client version 1.3.5.01 based on MIT Kerberos version 1.3.5, available as a Web release.

The Kerberos Client includes the following utilities:

- `kinit`, `klist`, and `kdestroy`: Manage credentials
- `kpasswd`: Change Kerberos passwords
- `ktutil`: Maintain the keytab file
- `kvno`: Display the Kerberos key version number of the principals

HP Kerberos Server Version 3.1

Kerberos Server is based on distributed client-server architecture. It ensures secure communication in a networked environment by leveraging individual trust relationships. It then distributes that trust across enterprise-wide, distributed client-server networks. It contains a GUI for configuration purposes.

Users can choose to configure the Kerberos Server v3.1 with a native (C-Tree) backend database or with an LDAP backend database.

Introduction to LDAP

Lightweight Directory Access Protocol (LDAP) is an Internet protocol that email programs use to look up contact information on a server. LDAP was designed at the University of Michigan to adapt a complex enterprise directory system (called X.500) to the modern Internet. X.500 is too complex to support on desktops and over the Internet, so LDAP was created to provide the same service. LDAP has broader applications such as looking up services and devices on the Internet (and intranets).

LDAP-enabled directories are becoming the defacto corporate standard to reduce user management cost. LDAP gained a lot of popularity with the explosive growth of the Internet and World Wide Web. LDAP-based directory servers are used to store the enterprise user and service information as well as the customer relationship information for e-commerce applications.

Kerberos Server on HP-UX with Native Back End

If you choose to use Kerberos Server with a native C-Tree back end, Kerberos Server maintains complete information for all the principals with their keys in a database on the machine on which the Kerberos server is configured. The native C-Tree database is used as the default backend database on the Kerberos Server v3.1.

Kerberos Server on HP-UX with LDAP Back End

Kerberos Server can also be configured with LDAP as the back end. If you choose to use LDAP, user information is stored in the LDAP directory in a centralized location. HP-UX users can log in to the system by accessing the user information from the LDAP directory with the help of LDAP-UX Integration product.

Benefits of an LDAP Back End

As the number of different networks and applications has grown, the number of specialized directories of information has also grown, resulting in islands of information that are difficult to maintain. LDAP, an open industry standard, has evolved to meet these needs by providing access to a common directory infrastructure. LDAP defines a standard method for accessing and updating information in a single directory.

By integrating the Kerberos principals with the corresponding users in an LDAP directory, you can create a single point of user and group management. This simplifies account administration by allowing user administration to be performed from a single location.

Implementing this solution involves the following steps:

- Modify the configuration files on the Kerberos Server
- Extend the LDAP directory schema

Integrating the Kerberos Principal into the LDAP Directory

A directory contains entries which are organized in a tree structure called the Directory Information Tree (DIT). Entries are arranged within the DIT based on their Distinguished Names (DN). DN is a unique name that unambiguously identifies a single entry. DNs are made up of a sequence of relative distinguished names (RDNs). Each RDN in a DN corresponds to a branch in the DIT leading from the root of the DIT to the directory entry. A DN is composed of a sequence of RDNs separated by commas, such as `cn=alex, ou=Sales, o=bambi.com`.

Figure 3 shows how a Kerberos principal is integrated in to the LDAP directory.

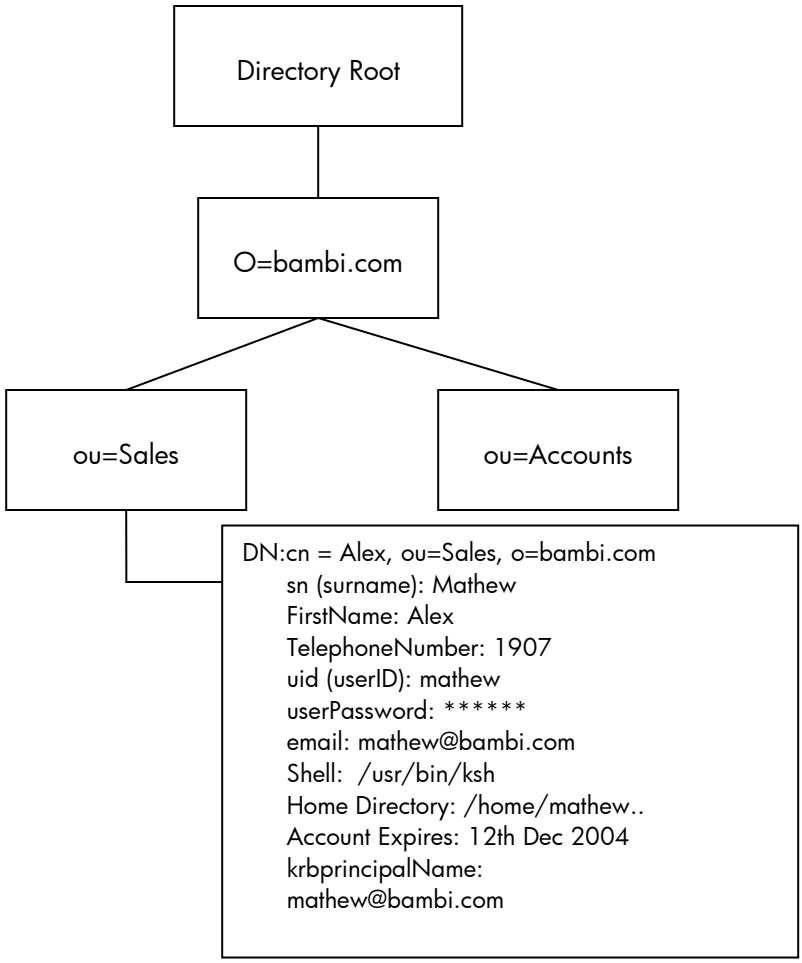

Figure 3: Integrating a Kerberos Principal in to the LDAP Directory

Figure 3 illustrates data related to the user Alex Mathew, who is located in the LDAP directory at `cn=Alex, ou=Sales, o=BAMBI.COM`. With both the POSIX account and LDAP information integrated, you can associate data like Alex's UNIX identity, his Kerberos identity, and any other attributes related to Alex within a single LDAP directory entry. In this case, different authentication mechanisms can share common data like account expiration date, password expire times, and failed authentication counts.

Generic Security Service Application Programming Interface (GSS-API)

GSS-API is an interface that provides security services to applications using peer-to-peer communication.

Using GSS-API routines, applications can perform the following operations:

- Enable an application to authenticate another application's user.
- Enable an application to delegate access rights to another application.
- Apply security services, such as confidentiality and integrity, on a per-message basis

GSS-API supports a secure connection between two communicating applications. The application that establishes the secure connection is called the *context initiator*. The application that accepts the secure connection is called the *context acceptor.*

GSS-API provides a standard programming interface that is authentication mechanism independent. GSS-API enables programmers to design applications and its associated protocols that can use

different authentication technologies, including Kerberos. HP recommends using GSS-API in application programs wherever possible.

Secure Internet Services (SIS)

HP-UX provides built-in support in a secure environment for Secure Kerberized Internet services such as `ftp, rcp, rlogin, telnet`, and `remsh`. Kerberized applications can have their behavior modified in the /etc/krb5.conf file. These applications use the Kerberos ticket instead of a password to authenticate the user to the remote machine.

ftp

To use the Kerberos `ftp` program, run it as you would normally run `ftp`. When you enter `ftp` <hostname>, you will still be prompted for your username. Press Enter. You will be logged in automatically. You should not be prompted for a password when trying to connect. If you are, **do not type your password.** Any password you type will not be encrypted and will go over the network in clear text.

rcp

You can use Kerberized `rcp` to transfer files securely between systems using Kerberos authentication. Kerberized `rcp` does not prompt for passwords. You must already have a valid TGT before using `rcp`.

rlogin/rsh

The Kerberos `rlogin` and `rsh` clients behave almost the same way as their non-Kerberized equivalents. Because of this, it is recommended that—if they must be included in the network—files such as /etc/hosts.equiv and .rhosts in the root user's directory be removed. The Kerberized versions have the added benefit of using Kerberos protocol for authentication. They can also use Kerberos for password encryption.

telnet

The Kerberos telnet client has many command line arguments that control its behavior; refer to the `manpage, telnet(1)` for complete information. The telnet client uses a session key even after the service ticket, from which it was derived, has expired. This means that the telnet session remains active even after the ticket originally used to gain access is no longer valid. This is insecure in a strict environment; however, the tradeoff between ease-of-use and strict security tends to lean in favor of ease-of-use in this situation. HP recommends that the telnet connection be re-initialized periodically by disconnecting and reconnecting with a new ticket. The overall lifetime of a ticket is defined by the KDC, normally defined to be eight hours.

Common Internet File System (CIFS)

The CIFS Client supports the Kerberos authentication mechanism. To authenticate, CIFS uses the standard procedures of RFC 2478 (GSS-API), which allow a client or server to call for authentication independently of the final choice of authentication method. Use of Kerberos in the CIFS environment provides significant security improvements over the older NT LanManager (NTLM) protocol traditionally used by CIFS Clients and Servers.

Secure Shell

HP-UX Secure Shell offers transparent encrypted security for HP-UX `11.0, 11i v1, 11i v1.6`, and `11i v2`. It is a Kerberized application that can use Kerberos as an authentication method. The

client/server architecture supports the SSH-1 and SSH-2 protocols and provides secured remote login, file transfer, and remote command execution.

HP-UX Secure Shell uses hashing to ensure data integrity and provides secure tunneling features, port forwarding, and an SSH agent to maintain private keys on the client. HP-UX Secure Shell supports the following authentication methods:

- Kerberos 5/GSS-API
- Password
- Public key
- Host-based

HP supports HP-UX Secure Shell at no additional cost to customers with HP-UX support agreements. HP-UX Secure Shell is a fully tested HP product. The following technologies are tested with HP-UX Secure Shell:

- Kerberos 5/GSS-API
- IPv6
- Trusted Systems
- TCP Wrappers
- PAM (PAM_UNIX, PAM_Kerberos, PAM_LDAP)

Compatibility/Interoperability

Because of its widespread acceptance and implementation in other operating systems, including Windows 2000 ®, Windows 2003, Solaris, and Linux, the Kerberos authentication protocol can interoperate in a heterogeneous environment allowing users on machines running one operating system to securely authenticate themselves on hosts of a different operating system.

Summary

Adding Kerberos to a network can increase the overall security available to the users and administrators of that network. Remote sessions can be securely authenticated and encrypted. In addition, Kerberos allows the user and service principal's database to be managed securely from any machine that supports the Kerberos protocol. All HP-Kerberos products are interoperable with other RFC 1510 compliant Kerberos implementations, and with MS Windows 2000 Active Directory services.

References

This section includes the URLs that customers can use if they would like to learn more about Kerberos and URLs that link to information about support, services, and accessibility.

1. Kerberos Manuals on the HP documentation website:
http://www.docs.hp.com/hpux/internet/index.html#Kerberos

2. The ITRC Website at http://itrc.hp.com where there is a wealth of information available for all HP-UX products, regarding maintenance, support, training and education.

Glossary

Following is a list of terms used throughout this document:

Name	Definition
Authentication	Verification of the claimed identity of a principal.
Authenticator	A record containing information that can be shown to have been recently generated using the session key known only by the client and server.
Authorization	The process of determining whether a client can use a service, which objects the client is allowed to access, and the type of access allowed for each.
Client	A process that makes use of a network service on behalf of a user. Note that in some cases, a server can itself be a client of some other server (e.g., a print server can be a client of a file server).
Credentials	A TGT plus the session key sent by the AS in response to an authentication request. This is stored in the user's credential cache, thus eliminating the need to re-key passwords multiple times during a session.
KDC	Key Distribution Center, a network service that supplies tickets and temporary session keys; or an instance of that service or the host on which it runs. The KDC services both initial ticket and ticket-granting ticket requests. The initial ticket portion is sometimes referred to as the Authentication Server (or service). The ticket-granting ticket portion is sometimes referred to as the ticket-granting server (or service).
Kerberos	A network authentication protocol. It is designed to provide strong authentication for client/server applications by using secret-key cryptography. A free implementation of this protocol is available from the Massachusetts Institute of Technology.
LDAP	Lightweight Directory Access Protocol (LDAP), an Internet protocol that email programs use to look up contact information from a server. LDAP was designed at the University of Michigan to adapt a complex enterprise directory system (called X.500) to the modern Internet. X.500 is too complex to support on desktops and over the Internet, so LDAP was created to provide this service. LDAP has broader applications, such as looking up services and devices on the Internet and intranets.

Principal	A unique name for a user or service stored in a KDC
Secret key	An encryption key shared by a principal and the KDC, distributed outside the bounds of the system, with a long lifetime. In the case of a user's principal, the secret key is derived from a password.
Service	A resource provided to network clients; often provided by more than one server (for example, remote file service).
Session key	A temporary encryption key used between two principals, with a lifetime limited to the duration of a single login session.
Ticket	A record that helps a client authenticate itself to a server; it contains the client's identity and other information all sealed using the server's secret key.
TGT	An initial ticket issued by the AS which is used to request additional tickets from the TGS for access to network services.

XXXX-XXXXEN, 03/2005