Global Information Assurance Certification Paper

GIAC Certified Windows Security Administrator (GCWN)
Practical Assignment
Version 3.2, Option 2

Windows and UNIX Interoperability

Using Active Directory to Authenticate Your UNIX Users: Does it Work and How Secure is It?

By Martin Poulin

Submitted on July 23, 2003

Table of Contents

Abstract

One of the main benefits of Active Directory is that it allows central management of your user account and password policies. For companies that work in a heterogeneous UNIX / Windows environment, it might be worth considering using Active Directory to manage all user accounts on all systems, to reduce duplication of effort and enforce a single account policy across the enterprise.

The paper will outline the steps required to set up Active Directory as a means to authenticate UNIX users (focusing on HPUX but with related information for Linux and other UNIXes) as well as any security implications for each step. What are the risks & vulnerabilities associated with the process, how is the authentication traffic passed from the UNIX server to the DC, and can it be done reliably and securely?

1.0. Introduction

TestDom Industries, Inc. is a medium-sized manufacturing company with over 1000 employees worldwide, in 30 countries. As an ISO-registered company, compliance to standards is extremely important for all aspects of their business, including I.T.

Their user population runs standard Windows 2000 Professional desktops and their core business is handled through an ERP system running on HPUX, centralized in their Head Office data center. They recently finished implementing a global Active Directory forest, using the recommended single-forest, single domain model.

TestDom conducted a third-party security audit of their business systems, and found several issues surrounding their ERP system. Most of the issues identified were policy violations around UNIX accounts management. While TestDom has written policies regarding user account creation and deletion, as well as a password policy which defines password complexity and expiry, the policies were not enforced, and the overworked I.T. staff had been negligent in managing the UNIX accounts: not enforcing password complexity, allowing passwords to remain unchanged since creation, and not disabling or deleting user accounts for employees who have left the company.

In contrast, the Active Directory domain was found to have very few issues. TestDom's Active Directory domain was designed around their written policies, largely because with the use of Group Policy it was very straightforward to translate the written policies into practice.

TestDom determined that the UNIX password noncompliances could be best corrected by removing duplication of effort. Since the Active Directory database already contained a more current record of existing users, management decided to investigate using the Active Directory domain to authenticate the UNIX users for access to the ERP systems. Along with correcting the policy violations, this would give them the benefits of a single user database for the entire enterprise, such as reduced administration for I.T. staff, more compliance from users (i.e. no need to write down passwords), and the future possibility of adding stronger authentication methods such as tokens or smart cards into the ERP system.

TestDom's management team agreed to address the non-conformities by the end of the next fiscal quarter, giving the I.T. team just over 3 months to find, test, and deploy a solution. The intention of this paper is to do a security-focused examination of the process, looking in detail at the Windows authentication components and how they compare and interoperate with UNIX systems (and identify some caveats), as well as demonstrate how to ensure that all traffic is handled securely, end-to-end.

1.1. Test Environment

The test environment has been set up with the following hardware and software:

LAN Connection:

- one 4-port minihub
- no direct connection to the Internet

Windows Domain Controller:

- Pentium III workstation, 256MB Ram, 10GB HD
- Windows 2000 Server, SP3
- All patches and hotfixes to date as of the time of writing
- IP Address: 10.128.1.41
- Hostname: dc.testdom.local

HPUX Host:

- HP rp7450 (L-class) server. 1GB RAM, 36GB SCSI mirrored internal disks *(note: original testing was done on a 700-series workstation, which was unable to run IPSec. Configuration was moved to a more powerful L-class server for the IPSec testing)*
- HPUX version 11.0
- All patches to date as of the time of writing
- IP Address: 10.128.1.40
- Hostname: hpux.testdom.local

Active Directory Domain:

- Single-domain, single forest
- Domain name: testdom.local

2.0. Overview of Active Directory Authentication

To understand how to use Active Directory to authenticate UNIX users, we must first understand the components of Active Directory, and specifically how Active Directory authentication works. Then we'll be able to compare this to the standard UNIX methods of authentication, and see if there is a fit.

Active Directory is based upon several protocols, including an LDAP-based database for storing information about objects, a Kerberos V5-based authentication protocol, and the Windows 2000 security model which controls access to objects within Active Directory. For backwards-compatibility, Active Directory retains support for LAN Manager (LM), NTLM, and NTLMv2 authentication.

For more detailed information on Active Directory authentication, I strongly recommend reading the excellent SANS Reading Room article by Julio Silveira, "Auditing the Windows 2000 Authentication Process"[1] and "Windows 2000 Authentication: Under The Hood"[2], by Jan De Clercq, both of which served as excellent references for this section.

2.1. LM, NTLM and NTLMv2 Authentication

LM and NTLM authentication are the weakest forms of authentication in a Windows environment, and any solution that requires their use should not be considered. The main weaknesses in LM and NTLM derive from the use of the LAN Manager One-Way Function, which allows the password to be cracked in 7-character sections. There are many widely available tools such as L0phtCrack, which trivialize this process, making LM and NTLM unsuitable as a secure authentication mechanism.

NTLMv2 is much stronger than LM/NTLM, with a 128-bit keyspace, the ability to prevent LM challenges, and can be further secured using 128-bit encryption and session security. If possible, it is good security practice to use Group Policy to disable LM and NTLM authentication, and only allow NTLMv2 or Kerberos. In fact, TestDom has a written policy stating that no weak protocols are to be used for authentication, and they have enforced this policy in practice via the Default Domain Policy by setting:

> Computer Configuration | Windows Settings | Security Settings | Local Policies | Security Options | Network Security | LAN Manager Authentication level | "Send NTLMv2 Response only \ refuse LM & NTLM"

Also, to ensure that LAN Manager Hashes aren't stored in Active Directory, they enabled the following in their Default Domain policy:

Computer Configuration | Windows Settings | Security Settings | Local Policies | Security Options | Network Security | "Do not store LAN Manager Hash value on next password change"

This means that any solution they use for authenticating their users must support either NTLMv2 authentication or Kerberos. TestDom would prefer to use Kerberos if possible, since it is the default authentication mechanism in Windows 2000, and an Internet (rather than "de facto") standard. We'll examine Windows 2000 Kerberos Authentication a bit more closely in the next section.

2.2. Windows 2000 Kerberos Authentication

Windows 2000 Kerberos authentication is more flexible and secure than NTLMv2 in a number of ways: thanks to the Kerberos Ticketing system, servers don't need to contact a domain controller to authenticate Kerberos clients, whereas in an NTLM environment the PDC or BDC must be contacted each time a client authenticates to a server; Kerberos allows clients to verify the identity of the servers, where NTLM does not; and Kerberos has improved delegation of authentication over NTLM.

Microsoft's implementation of Kerberos is based on Kerberos version 5, but they have also extended the protocol to allow initial authentication to be handled by public key certificates instead of the standard shared secret keys. This extension, while a definite point of controversy in some circles, was designed to allow interactive Smart Card logons. It will be important to keep in mind when designing for interoperability with other operating systems.

The following is an abridged overview from Microsoft's documentation, of how Microsoft implements Kerberos in Windows 2000. More detailed information can be obtained from Microsoft's document entitled "Windows 2000 Kerberos Authentication"[3].

> **Key Distribution Center**
> ("Windows 2000 Kerberos Authentication", p. 19)[3]
>
> The Windows 2000 Key Distribution Center runs as a domain service in Active Directory, and runs on every Domain Controller in the domain. The account database is stored in AD and user information is retrieved from the Global Catalog. The KDC provides two services: The Authentication Service, which provides Ticket Granting Tickets; and the Ticket Granting Service, which issues tickets for services in the Active Directory domain.
>
> The Domain Controller's Local Security Authority (LSA) starts the KDC automatically. When the Active Directory domain is created, a special account called "krbtgt" is created. The KDC on all Domain Controllers within the domain use this account.

Account Database
("Windows 2000 Kerberos Authentication", pp. 19-20) [3]
All of the information about security principals is stored in Active Directory, as attributes of the account object. Since Active Directory already uses its own replication model, Microsoft does not use the Kerberos replication protocol. Rather, it replicates using the Active Directory multi-master replication protocol.

Account data is stored in the Directory System Agent (DSA). Directory service clients are never allowed to directly access the data store, so all requests for information must go through Active Directory Services Interfaces (ADSI).

Kerberos Policy
("Windows 2000 Kerberos Authentication", pp. 20-21) [3]

Kerberos Policy is set in the Default Domain Policy in Active Directory, and can only be modified by members of the "Domain Admins" group.

The following quote from Microsoft's documentation gives the following outline of the policies:

- ***Enforce user logon restrictions.*** *Determines whether the KDC validates every request for a session ticket against the user rights policy on the target computer. When this policy is enabled, the user requesting the session ticket must have the right either to Log on locally or to Access this computer from network. Validation of each request is optional because the extra step takes time and may slow network access to services. By default, this policy is enabled.*
- ***Maximum lifetime for service ticket.*** *Determines the maximum amount of time (in minutes) that a ticket granted for a service (that is, a session ticket) can be used to access the service. If the setting is zero minutes, the ticket never expires. Otherwise, the setting must be greater than ten minutes and less than the setting for Maximum lifetime for user ticket. By default, the setting is 600 minutes (ten hours).*
- ***Maximum lifetime for user ticket****. Determines the maximum amount of time (in hours) that a user's TGT can be used. When a user's TGT expires, a new one must be requested or the existing one must be renewed. By default, the setting is ten hours.*
- ***Maximum lifetime for user ticket renewal.***

Determines the longest period of time (in days) that a TGT can be used if it is repeatedly renewed. By default, the setting is seven days.

- ***Maximum tolerance for computer clock synchronization.*** *Determines the maximum difference (in minutes) that Kerberos will tolerate between the time on a client's clock and the time on a server's clock while still considering the two clocks synchronous. By default, the setting is five minutes.*
("Windows 2000 Kerberos Authentication", p. 21)[3]

This final policy brings up a very important point to remember when planning for interoperability. If we are to implement Kerberos successfully, we'll need to ensure that the clocks between our UNIX servers and the Windows servers are synchronized, either by configuring the UNIX host as an NTP source for the Windows Domain Controllers, or vice-versa.

Delegation of Authentication
("Windows 2000 Kerberos Authentication", p. 21)[3]

As mentioned before, Windows 2000 can use Kerberos to allow services to act as a client to access resources on other computers. For this to work properly, all computers involved must be running Windows 2000, they must be in an Active Directory domain, and both the client and service accounts must be marked for delegation.

Preauthentication
("Windows 2000 Kerberos Authentication", p. 22)[3]

Preauthentication is required by default on a Windows 2000 KDC; however it can be turned off for individuals if needed.

Kerberos Security Support Provider
("Windows 2000 Kerberos Authentication", pp. 22-23)[3]

An SSP is a library included with the Operating System to handle authentication. Windows 2000 includes an SSP for Kerberos as well as NTLM.

Credentials Cache
("Windows 2000 Kerberos Authentication", pp. 23-24)[3]

The Credentials Cache is an area of volatile memory where tickets and keys obtained from the KDC are stored. The cache is destroyed when a user logs off or shuts down the system, and the cache is never written to the virtual memory page file.

DNS Name Resolution
("Windows 2000 Kerberos Authentication", p. 24) [3]

Since the IP protocol is used for Kerberos messages, there needs to be a mechanism for resolving IP addresses for the KDC. Active Directory has built-in DNS services, and through the use of SRV resource records, Kerberos clients can locate the domain controllers and KDCs.

IP Transport
("Windows 2000 Kerberos Authentication", p. 24-25) [3]

The KDC in Active Directory listens on UDP port 88 of the domain controller, for compatibility with non-windows Kerberos clients. However, because Windows authorization data can be larger than the 1500 octets allowed by the UDP protocol, Kerberos messages are transmitted using TCP instead of UDP.

2.3. Microsoft's LDAP Implementation

Lightweight Directory Access Protocol, or LDAP, is defined in RFC 1777 (version 2) and RFC 2251 (version 3). Very simply defined, LDAP is a protocol used to read and write to a directory across a network.

Microsoft claims that they support LDAP versions 2 and 3, and that they are interoperable with any LDAP-compatible client. However, there is a catch; Active Directory will answer LDAP requests **in clear text**, but due to the nature of the Active Directory security model, only certain authorized users can access the entire directory. Full access to Active Directory is provided through Active Directory Services Interface, or ADSI. ADSI is Microsoft's proprietary interface into Active Directory, which does not itself use LDAP.
Therefore some work is needed before you can use LDAP to make queries from an LDAP client. This is another thing to keep in mind when planning for interoperability.

3.0. Different Methods of Authenticating Unix users

3.1. Traditional Password Files

Unlike Active Directory, where account and configuration information is stored in a replicated database, UNIX account and configuration information is traditionally stored in two text files: /etc/passwd and /etc/group. This can be a security risk in itself since the password file is world-readable, and it's possible for someone to obtain encrypted password hash from this file and crack them offline. Some methods, such as "shadow passwords", seek to address this risk by removing the password hashes from the password file and storing them in a more restricted file that is only readable by the root user, usually /etc/shadow.

3.2. NIS

There are other limitations to the traditional UNIX authentication scheme. Each host must have its own password file, so managing users across multiple hosts becomes quite tedious. Sun Microsystems' "Network Information Service", or NIS, was created to address this situation: NIS (and the newer NIS+) is a protocol for distributing user and system data between computers on a network. In NIS / NIS+, the user and group information resides on NIS servers, and NIS client systems retrieve the information across the network. Windows 2000 and Active Directory don't have built-in support for NIS, but with Microsoft Services for UNIX, that support can be added. As we'll see, this is *part* of the solution, but we won't actually be using an NIS server in Windows.

3.3. SAMBA and Winbind

Another solution for Windows and UNIX interoperability is Winbind. According to the SAMBA online documentation for version 3.0, Winbind is *"…a component of the Samba suite of programs as a solution to the unified logon problem. Winbind uses a UNIX implementation of Microsoft RPC calls, Pluggable Authentication Modules, and the Name Service Switch to allow Windows NT domain users to appear and operate as UNIX users on a UNIX machine*." ("Integrated Logon Support Using WinBind")[4] However, SAMBA and Winbind are more suited to a Windows NT 4.0 Domain than Active Directory, so we won't be using SAMBA or WinBind for our solution.

3.4. LDAP on UNIX

LDAP is a much more robust means of centralizing and distributing user and group information than NIS or traditional password files. Since Active Directory is based on LDAP, we should be able to use the LDAP services in Active Directory to provide the user and group information to the UNIX hosts. Most importantly for TestDom, LDAP is the vendor-recommended solution for authenticating

HPUX users against Active Directory. HP provides software called “LDAP-UX Client Services” for this task.

3.5. PAM

One very important component that comes into play for all methods of UNIX authentication is “PAM” – the Pluggable Authentication Modules. PAM simplifies the authentication process by acting as a service against which programs can authenticate. Individual programs can get PAM to do the authentication for them, and policies can be set via PAM configuration files. As we will see, PAM plays a very important role in securely authenticating UNIX users to Active Directory.

4.0 Microsoft Services for UNIX 3.0

4.1. SFU Overview

Microsoft Services for Unix (SFU) is a set of utilities based on the Interix subsystem to help integrate Windows into a UNIX environment. It provides services to integrate file sharing, remote command shells, scripting, network administration and authentication.

For our purposes, we are only interested in the SFU authentication mechanism. According to Microsoft's "Introduction to Services for Unix "[5] whitepaper:

> "*The primary objective of Services for UNIX is to provide integrated tools that bridge the gap between UNIX and Windows to both users and administrators. This allows the creation of enterprise networks where resources can be shared seamlessly. Access to resources is determined by enterprise policies and must accommodate the sharing of credentials, authorization, and authentication information from either the Windows or UNIX domain."*
> ("Introduction to Services for Unix", p. 1)[5]

So if Microsoft is to be believed, Services for UNIX is the solution to TestDom's problems. SFU offers two ways of managing UNIX authentication: password synchronization, and NIS server functionality via Active Directory.

4.2. Password Synchronization

Password synchronization gives the *appearance* of a single user database, as far as the user is concerned. However, there will still be multiple physical user databases (the Active Directory database and the UNIX password file) and special care needs to be taken to ensure that only the specified accounts are synchronized between UNIX and Windows. For these reasons, password synchronization does not appear to be the best solution.

4.3. Server for NIS

Server for NIS appears to be more useful in TestDom's situation. The user database is centralized in Active Directory so there are no potential conflicts between UNIX users and Windows users and, as we'll see, we can use the Schema extensions from Server for NIS in combination with LDAP and Kerberos to store and query UNIX attributes (as opposed to using the NIS protocol itself), and securely authenticate our users. For these reasons it seems to be the better choice.

4.4. SFU Vulnerabilities

Before we install anything, it's always a good idea to do a bit of research to find if there are any vulnerabilities. According to the CERT Coordination Center, at the time of this writing there are four known bulletins for SFU; only one of which affects SFU 3.0.

- **CERT VU#192995**, summarized here:
  http://www.kb.cert.org/vuls/id/AAMN-5CKTFF
  and here:
  http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms02-057.asp
  This is a minimal risk in our environment since it only affects applications which are created on the Sun Microsystems RPC library, using the SFU 3.0 Interix SDK. For our purposes, we will not be developing any applications using this SDK, nor will we be installing the Interix SDK. Nonetheless, if needed, patches are available from Microsoft at:
  http://www.microsoft.com/Downloads/Release.asp?ReleaseID=43447

- **CERT VU#581603**, a Denial of Service vulnerability, is discussed here:
  http://www.kb.cert.org/vuls/id/581603
  This only affects SFU 2.0. Since it is good practice to use the most current version of software when deploying a new installation, this vulnerability should not affect us. Still, there is a patch for this vulnerability if for any reason you plan to use SFU 2.0:
  http://www.microsoft.com/Downloads/Release.asp?ReleaseID=31592

- **CERT VU#994851** describes a memory leak in Telnet services for SFU 2.0. It is discussed in detail here:
  http://www.kb.cert.org/vuls/id/994851
  Again, since this also affects only SFU 2.0, and since we do not require (nor **want!**) Telnet services enabled, it is not a risk to our environment. Just in case, the patch is available here:
  http://www.microsoft.com/Downloads/Release.asp?ReleaseID=31595

- **CERT VU#952611**, which also depends on the Telnet service on SFU 2.0, is partly a vulnerability in Internet Explorer, and partly a Telnet client vulnerability. The details are here:
  http://www.kb.cert.org/vuls/id/JSHA-54X5ZA
  Like the previous vulnerability, this should not affect SFU 3.0, and there are several mitigating factors which make this a low risk vulnerability, which can be further mitigated by simply not installing the Telnet service. The patch can be found here, should it be required:
  http://www.microsoft.com/technet/security/bulletin/ms01-051.asp

So depending on what is installed, there are really only a few patches that we may need to apply in order to prevent any potential vulnerability. Combining this with the basic principle of "if you don't need it, don't install it", we should be able to install SFU with no remotely exploitable vulnerabilities. The rest of this section will explain step-by-step how to install SFU 3.0, and point out any potential pitfalls.

4.5. Installing Microsoft Windows Services for UNIX 3.0

Full details on installing and configuring SFU 3.0 can be found on the SFU 3.0 CD under the name "INSTALL.HTM" [6]. For our purposes, we'll want to install only the components needed for authenticating UNIX users, so this section will cover the specific steps needed to accomplish that.

The first thing to understand is that Server for NIS **must** be installed on a Domain Controller. In our test environment we have only a single Domain Controller (dc.testdom.com) but in your environment you may have multiple Domain Controllers. If that's the case, you'll need to decide which DC to install SFU on, based on several factors: location, conflicting services, availability, etc.

Be aware that installing SFU requires you to reboot your Domain Controller, so schedule the installation during a time that will not interfere with production operations. Also make sure you have a complete backup before making any changes to Active Directory, and that you test this installation in an isolated test environment before implementing it on a production system.

It's especially important to understand that Server for NIS will extend the Active Directory Schema. I will cover these extensions in detail later. **Before installing SFU or making any changes to the Schema, make sure you have a complete backup of your entire domain controller(s), including the "system state".**

4.5.1. Installing SFU from the Command Line

The entire installation can be run from the command line – all you need is the CD or a network share with the installation files. Since we are only installing one component – Server for NIS – we can use the following command line (you will need to modify the path and license key to suit your environment):

```
msiexec /I D:\SFUsetup.msi ADDLOCAL="NIS" PIDKEY="license key" SFUDIR="C:\SFU" /q
```

I'll break down the command to explain it further:

- **msiexec /I D:\SFUsetup.msi** – this tells the MSI installer to install SFU program from the D: drive. You can install from a network share by using the UNC path, e.g. \\server\share\SFUsetup.msi

- **ADDLOCAL=”NIS”** – this specifies that it should install only the “NIS” component
- **PIDKEY=”license key” –** unless you have a “Select” or “Enterprise” agreement with Microsoft, you’ll need to enter a valid license key for the SFU software.
- **SFUDIR=”C:\SFU” –** installs the SFU components to C:\SFU
- **/q –** runs the installation in “quiet mode”, allowing for unattended installations.

For more information on the command-line options, consult the installation notes located in “INSTALL.HTM” on the SFU CD.

4.5.2. Installing SFU using the GUI

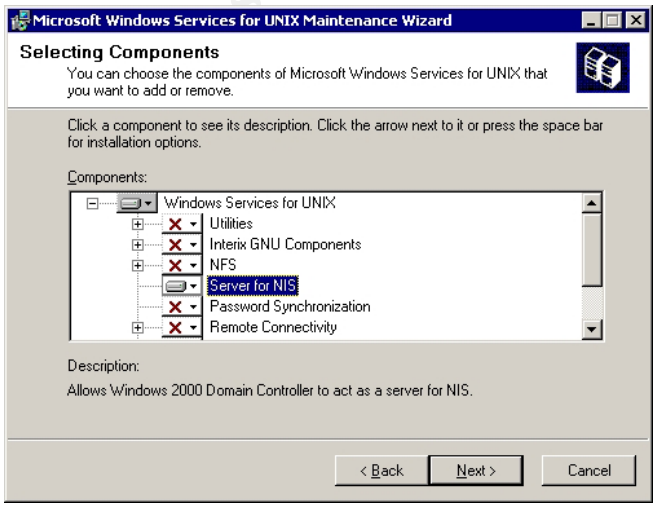
If you don’t like the command line, you can choose to install SFU using the GUI. This next section will go through the details of using the GUI to install SFU 3.0.

a) Start the installation by inserting the CD; otherwise you can start the installation by browsing to the SFUsetup.msi file and double-clicking it.

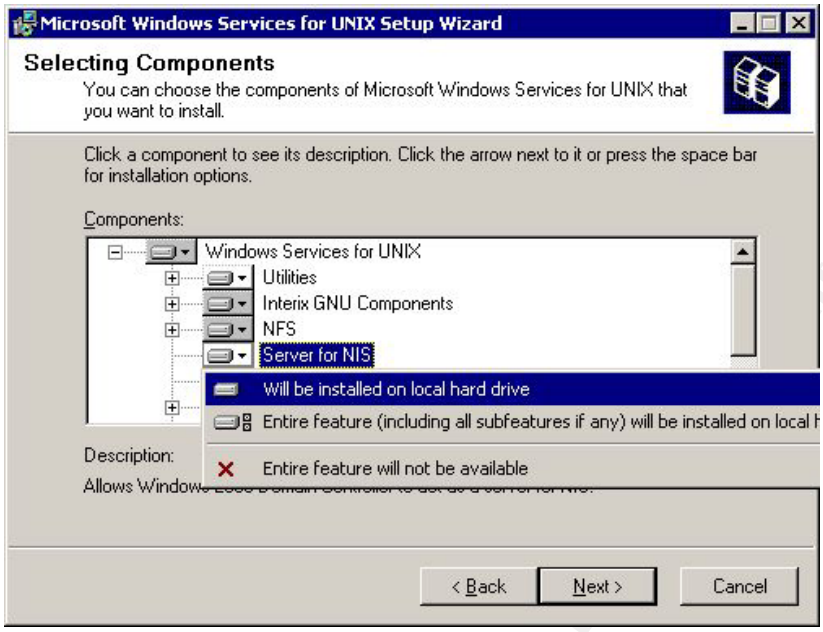
b) Fill in your user and organization information, and make sure to read the entire End User License Agreement before accepting, to ensure that there are no legal loopholes, which may impact your company.

c) When prompted for the Installation Options, be sure to select “Custom Installation”

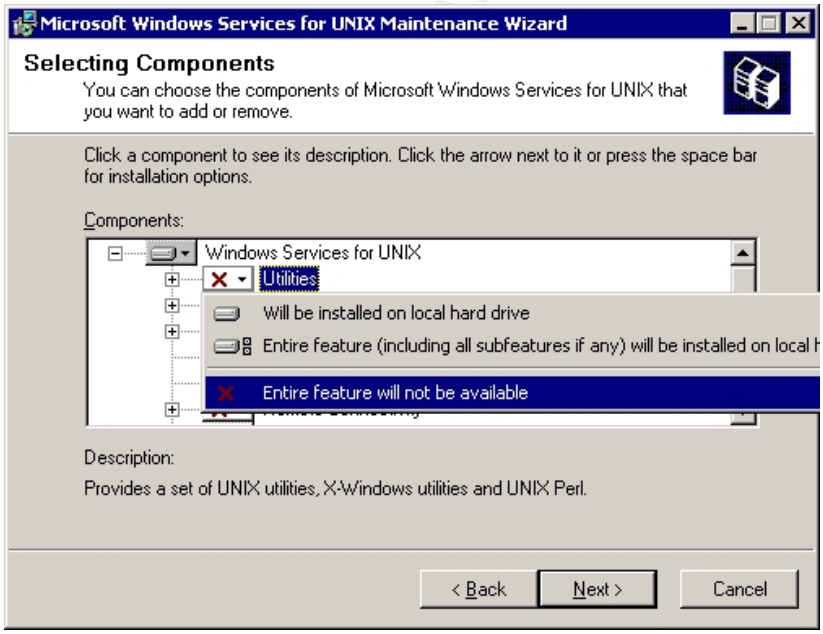
d) In the next window, you will be asked to select which components to install. In short, you want to install only the Server for NIS, as demonstrated in the following screenshot:



e) Select “Server for NIS” and choose “Will be installed on local hard drive.



f) For all other options, choose “Entire feature will not be available”



g) Once all of the options have been chosen, click “Next”.

h) You will be warned that the schema will be upgraded. Again, make sure you have a complete backup of your Active Directory Schema, since this change is irreversible. SFU will add POSIX attributes for all objects in Active Directory as well as security properties for these attributes.

i) Select a directory to install SFU (e.g. C:\SFU). Make sure to set appropriate permissions on this directory.

j) Wait for the installation to complete, and then reboot when prompted.

4.5.3. Extension of the Active Directory Schema

It's important to take a moment to understand exactly what we have done to Active Directory, by installing SFU. The following is an excerpt from the Microsoft TechNet article, "Server for NIS Overview" [7]

> *"While implementing NIS server functionality, Server for NIS does not store NIS maps as flat data or as plain text entries in Active Directory. It extends Active Directory schema to store UNIX attributes. Each map is created as a separate Active Directory class. It stores NIS map by associating each field of the map as a separate attribute. Entries in the NIS map are stored as objects of that class. Fields of NIS maps are stored as attributes of that object"*
>
> ("Server for NIS Overview", section "Extension of Active Directory Schema") [7]

I was able to find the following classes and attributes that SFU 3.0 adds to the Schema by using the Active Directory Schema tool. Notice that they are all identifiable by the string "msSFU30" at the beginning of the class or attribute name.

- msSFU30Aliases (Attribute)
- msSFU30BootableDevice (Class)
- msSFU30BootFile (Attribute)
- msSFU30BootParameter (Attribute)
- msSFU30CryptMethod (Attribute)
- msSFU30DomainInfo (Class)
- msSFU30Domains (Attribute)
- msSFU30FieldSeparator (Attribute)
- msSFU30Gecos (Attribute)
- msSFU30GidNumber (Attribute)
- msSFU30HomeDirectory (Attribute)
- msSFU30Ieee802Device (Class)
- msSFU30IntraFieldSeparator (Attribute)
- msSFU30IpHost (Class)
- msSFU30IpHostNumber (Attribute)
- msSFU30IpNetmaskNumber (Attribute)
- msSFU30IpNetwork (Class)
- msSFU30IpNetworkNumber (Attribute)
- msSFU30IpProtocol (Class)
- msSFU30IpProtocolNumber (Attribute)
- msSFU30IpService (Class)
- msSFU30IpServicePort (Attribute)

- msSFU30IpServiceProtocol (Attribute)
- msSFU30IsValidContainer (Attribute)
- msSFU30KeyAttributes (Attribute)
- msSFU30KeyValues (Attribute)
- msSFU30LoginShell (Attribute)
- msSFU30MacAddress (Attribute)
- msSFU30MailAliases (Class)
- msSFU30MapFilter (Attribute)
- msSFU30MasterServerName (Attribute)
- msSFU30MaxGidNumber (Attribute)
- msSFU30MaxUidNumber (Attribute)
- msSFU30MemberNisNetgroup (Attribute)
- msSFU30MemberOfNisNetgroup (Attribute)
- msSFU30MemberUid (Attribute)
- msSFU30Name (Attribute)
- msSFU30NetGroupDetail (Attribute)
- msSFU30NetGroupHostAtDomain (Attribute)
- msSFU30NetGroupUserAtDomain (Attribute)
- msSFU30NetId (Class)
- msSFU30NetworkUser (Class)
- msSFU30NisDomain (Attribute)
- msSFU30NisMap (Class)
- msSFU30NISMapConfig (Class)
- msSFU30NisMapEntry (Attribute)
- msSFU30NisMapName (Attribute)
- msSFU30NisNetgroup (Class)
- msSFU30NisObject (Class)
- msSFU30NSMAPFieldPosition (Attribute)
- msSFU30OncRpc (Class)
- msSFU30OncRpcNumber (Attribute)
- msSFU30OrderNumber (Attribute)
- msSFU30Password (Attribute)
- msSFU30PosixAccount (Class)
- msSFU30PosixGroup (Class)
- msSFU30PosixMember (Attribute)
- msSFU30PosixMemberOf (Attribute)
- msSFU30ResultAttributes (Attribute)
- msSFU30SearchAttributes (Attribute)
- msSFU30SearchContainer (Attribute)
- msSFU30ShadowAccount (Class)
- msSFU30ShadowExpire (Attribute)
- msSFU30ShadowFlag (Attribute)
- msSFU30ShadowInactive (Attribute)
- msSFU30ShadowLastChange (Attribute)
- msSFU30ShadowMax (Attribute)
- msSFU30ShadowMin (Attribute)
- msSFU30ShadowWarning (Attribute)
- msSFU30Top (Class)
- msSFU30UidNumber (Attribute)
- msSFU30YpServers (Attribute)

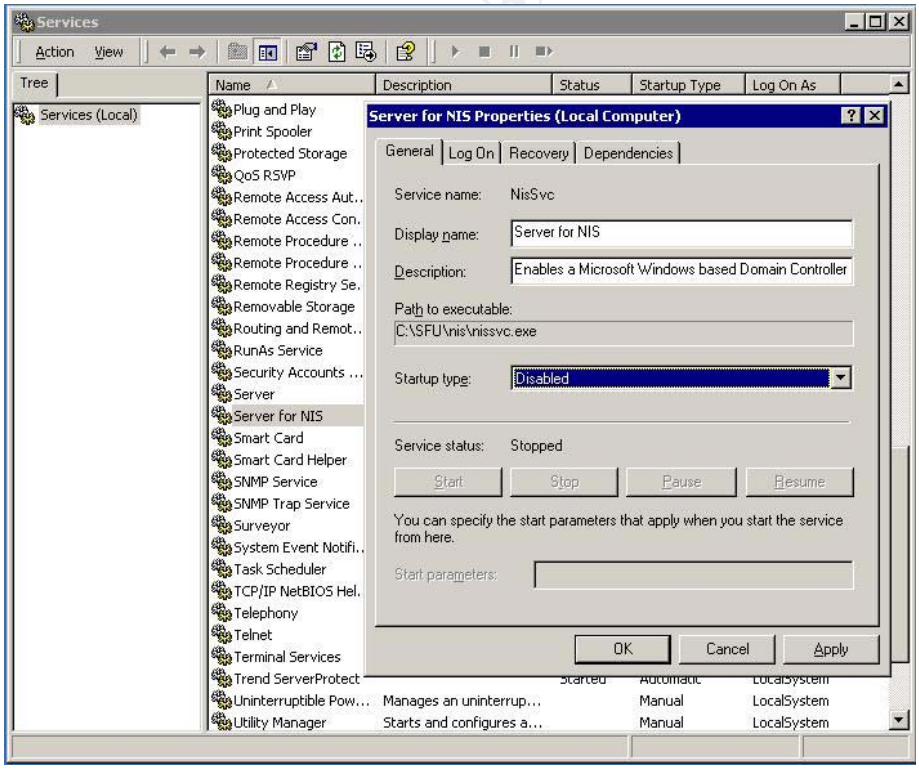
Now we have Server for NIS installed, and we can theoretically use it to act as an NIS server for our UNIX hosts. But is that what we really want? NIS is notoriously insecure (for more information on security issues with NIS, I recommend reading Eric Gallagher's "Password Security in NIS Systems" [8] in the SANS Reading Room), so it's not a good idea to open up these holes in our Active Directory. Taking into consideration security, standards, and the recommendations from our UNIX vendor (Hewlett Packard), it looks like the best approach will be to utilize the built-in LDAP and Kerberos functionality in Active Directory.

4.5.4. Disable Server for NIS

In fact, we can actually **disable** the Server for NIS service, and still authenticate our users using LDAP and Kerberos. Since we don't want to expose our Domain Controller to any unnecessary services, this is probably a good idea. At a command prompt, type the following:

```
net stop "Server for NIS"
```

You should see a message saying that the service stopped successfully. To make sure the service doesn't start again when the Domain Controller is rebooted, go into the Services control panel, and change the startup type to "Disabled". See the following screenshot:



Does this mean we wasted our time installing Server for NIS? Not at all. The SFU extensions to the Schema are very important in our quest to authenticate UNIX users – in particular things like UidNumber, GidNumber, HomeDirectory, LoginShell, etc. Sure, a talented programmer **could** use ADSI Edit to manually extend the Schema and add these classes and objects and attributes, but the work involved and the chance of mistakes make this unrealistic.

At this point, we have everything installed that we need on the Domain Controller. The next step is to configure Active Directory so it will be able to communicate securely with the UNIX server.

4.6. Configure Active Directory

4.6.1. Create the Proxy User

A special "proxy" account is needed to be able to read the extended POSIX attributes that SFU has installed in Active Directory. The account only needs read access to the POSIX attributes, so by following the principle of least privilege you should delegate only that access which is required.

If you are operating Active Directory in "mixed-mode" (i.e. "Permissions compatible with Pre-Windows 2000 servers) then any user can be the proxy user, since all users have read access to all attributes. This is not an ideal setup, and it's not generally recommended to operate in mixed mode for a long period of time.

If you are running in "native mode" then the proxy user must **either** be a member of the built-in "Pre-Windows 2000 Compatible Access" group, or you must delegate POSIX attribute read access to this user. This built-in group has the necessary permissions to read the extended attributes, but it also has read-access to all other attributes. For a restricted account like the proxy account, it is a much better solution to delegate only the required access.

We have configured our test environment in Native Mode, and created a proxy account called "uxproxy". This account is configured with a good strong pass phrase, and it is added to a special group we have created called "Restricted Accounts" (which as its name suggests, is a user group with limited permissions on the directory). The Primary Group is set to "Restricted Accounts" and we have removed the uxproxy account from "Domain Users". This ensures that we have complete control over the account's level of access.

Since this account has privileged read access to Active Directory, it's a good idea to enforce a password change policy on it, and add the password change to your internal procedures. Remember, if the account gets locked out, the UNIX users will not be able to authenticate!

NOTE: *To change the proxy user password in HPUX, use the* `ldap_proxy_config` *command, e.g.*

```
# cd /opt/ldapux/config
#./ldap_proxy_config -i
CN=uxproxy,CN=Users,DC=testdom,DC=local
<enter password when prompted>
```

You will also need to use the above command if you move the proxy user account from the default "Users" container – e.g. if you move it into an OU called "Untrusted Users", then the DN for the proxy user will change to "CN=uxproxy, CN=Untrusted Users, DC=testdom, DC=local".

4.6.2. Proxy User Delegation of Control

The following step-by-step will walk you through the delegation of control for the Proxy user. Hewlett Packard's document "Installing and Administering LDAP-UX Client Services with Microsoft Windows 2000 Active Directory" [9] (pp. 25-27) contains similar instructions. I have modified these instructions here to apply to Services for Unix 3.0:

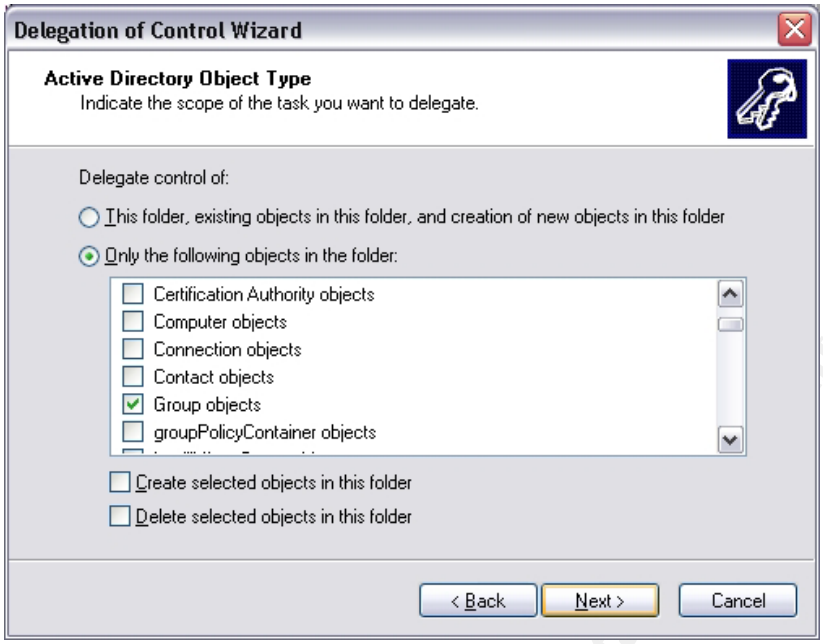
a) Open the Active Directory Users and Computers MMC Snap-in.

b) Right-click the container that holds the proxy user account, and select "Delegate Control". Click Next to begin the Delegation of Control Wizard.

c) Click "Add…" and add the name of the Proxy user (uxproxy) to the list. Click OK, and then click "Next".

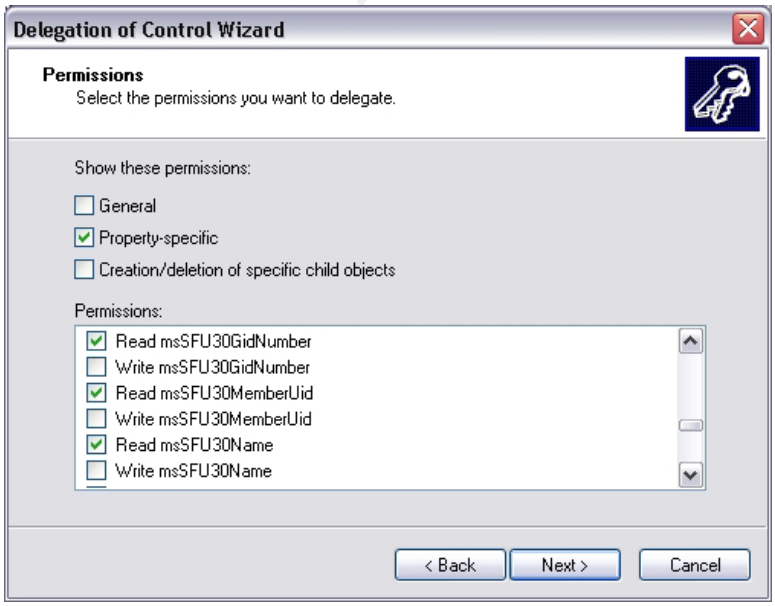
d) Select "Create a custom task to delegate", and click Next.

e) Under "Delegate control of:" select "Only the following objects in the folder". Select only "Group Objects", and click "Next".



f) In the Permissions window, choose “Property-specific” and select the following:
(note that not all properties appear in the screenshot)

- Read msSFU30GidNumber
- Read msSFU30memberUid
- Read msSFU30Name
- Read msSFU30Password



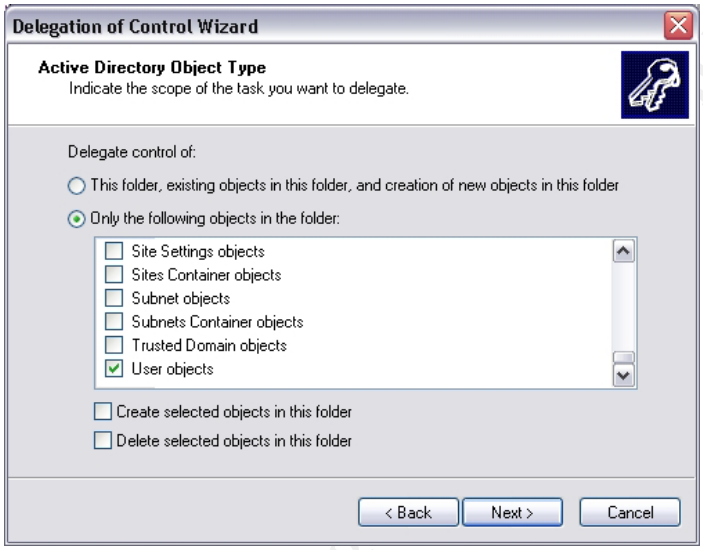
Click “Next”, and then “Finish”.

g) Once again, Right-click the container that holds the proxy user account, and select “Delegate Control”. Click Next to begin the Delegation of Control Wizard.

h) Click “Add…” and add the name of the Proxy user (uxproxy) to the list. Click OK, and then click “Next”.

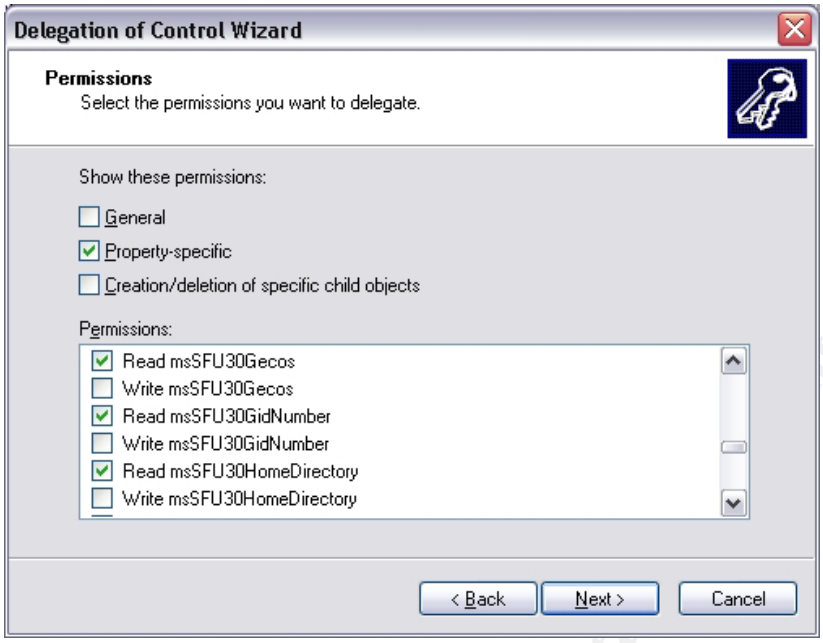
i) Select “Create a custom task to delegate, and click “Next”.

j) Under “Delegate control of:” select “Only the following objects in the folder”. Select only “User Objects”, and click “Next”.



k) In the Permissions window, choose “Property-specific” and select the following:
(again, not all properties appear in the screenshot)

- Read msSFU30Gecos
- Read msSFU30GidNumber
- Read msSFU30HomeDirectory
- Read msSFU30LoginShell
- Read msSFU30Name
- Read msSFU30Password
- Read msSFU30ShadowExpire
- Read msSFU30ShadowFlag
- Read msSFU30ShadowInactive
- Read msSFU30ShadowLastChange
- Read msSFU30ShadowMax
- Read msSFU30ShadowMin
- Read msSFU30ShadowWarning
- Read msSFU30UidNumber



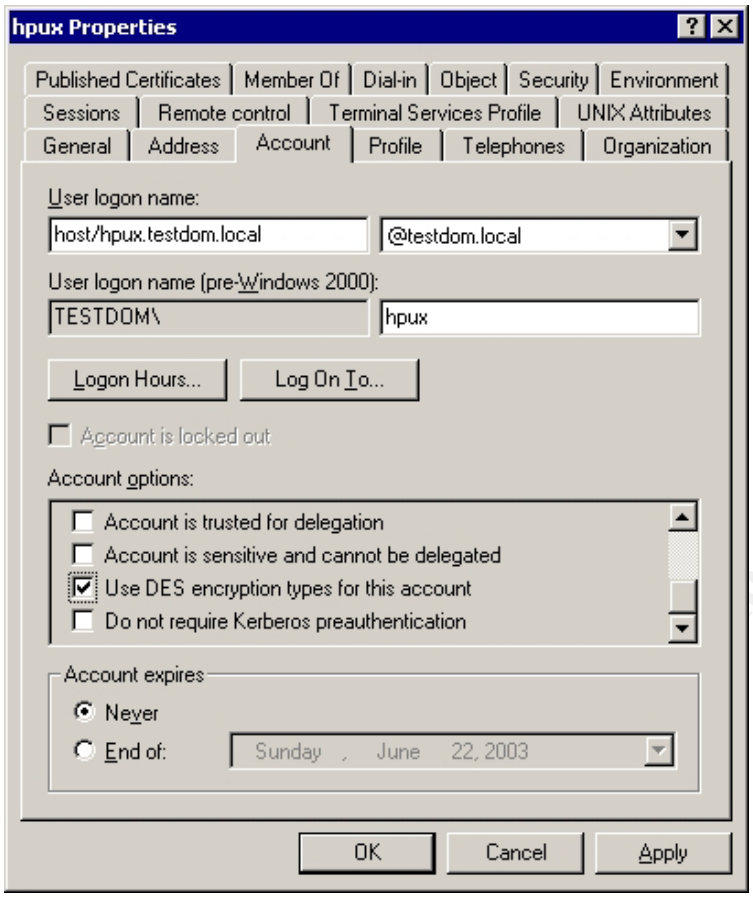
Click “Next”, and then “Finish”.

4.7. Add an Active Directory account for the UNIX Host

Now that we have a Proxy account set up which can query the POSIX attributes for the users, we will somehow need to get the UNIX host to communicate with the Windows 2000 Kerberos Services. To do this, we need to set up a user account for the UNIX host, and create a keytab file that will be imported into the UNIX host. Again, this user goes into our “Untrusted Users” container, and has very limited permissions.

In our test environment, we have created a user called “hpux” for this purpose.

For this account, it’s very important to make sure the “Use DES Encryption Types” is checked in the Account options, under the Account tab. The reason you need to select this is for interoperability with the HPUX Kerberos client, which is based on MIT Kerberos. By default, Windows accounts will use RC4-HMAC encryption for accounts, but setting “Use DES Encryption Types” will allow it to use “DES-CBC-CRC” for interoperability.



4.7.1. Create a Keytab file for the UNIX host

Once the account is created, create a keytab file for the Unix host using “ktpass”. Ktpass can be installed from the support tools on the Windows 2000 Server CD.

The following example is the ktpass syntax / output used for our test environment:

```
C:\>ktpass -princ host/hpux@testdom.local -mapuser hpux -pass this1s1nteresting! -out
unix.keytab
Successfully mapped host/hpux to hpux.
Key created.
Output keytab to unix.keytab:

Keytab version: 0x502
keysize 51 host/hpux@TESTDOM.LOCAL ptype 1 (KRB5_NT_PRINCIPAL) vno 1 etype 0x1
(DES-CBC-CRC) keylength 8 (0xda6d100d13ea8a2f)
Account has been set for DES-only encryption.
```

5.0. LDAP-UX Client Services

LDAP-UX Client Services is a tool used to store and query information from an LDAP directory. We need it on our HPUX machine so we can query Active Directory for user and account information. It can also be used for authentication, but be warned that usernames and passwords are sent in the clear when using LDAP-based authentication ("LDAP-UX Integration B.03.00 Release Notes, Fourth Edition", p. 26) [10]. More on that later…

5.1. Install Software and Patches

Install the following patches (and all related dependencies):

- PHCO_26089
- PHCO_25796
- PHNE_23003
- PHNE_23949

Install the following packages from the HP media:

- PAM Kerberos: J5849AA_B.11.00.12_HP-UX_B.11.00_32+64.depot (no reboot required)

- IPSEC: J4256AA_A.01.05_HP-UX_B.11.00_32+64.depot (requires a reboot)

Full installation instructions for IPSec can be found at:
http://docs.hp.com/hpux/pdf/J4255-90011.pdf

- LDAP-UX: J4269AA_B.03.10_HP-UX_B.11.00_32+64.depot (requires a reboot)

Installation instructions for LDAP-UX can be found at:
http://www.software.hp.com/cgi-bin/swdepot_parser.cgi/cgi/displayProductInfo.pl?productNumber=J4269AA&oper=install
Note: there is a known vulnerability in earlier versions of LDAP-UX integration services, so it is important to install at least version 03.10. See "Appendix A" for details on the vulnerability.

5.2. Configure LDAP-UX

Once installed, configure LDAP-UX:

a) Log on as root

b) Type: `cd /opt/ldapux/config`

c) Type: `/setup`
You will be asked if you would like to continue with the setup. Press <Enter> to choose "Yes".

d) You will be prompted for which Directory Server you want to connect to. Choose "2" for Windows 2000 Active Directory.

e) You will be prompted for the server host. Type the name of the Domain Controller where you have installed Microsoft Services for Unix 3.0 (in our case, "dc.testdom.local").

f) You will be asked to enter the Directory Server port number. Hit <Enter> to choose the default (389).

g) You will be asked for the Distinguished Name of the directory user allowed to check the Schema. This account should be a Domain Administrator. In our test case, we will use the Administrator account (CN=Administrator,CN=Users,DC=testdom,DC=local). Enter the DN, or hit <Enter> to choose the default.
If you're not sure what the DN is, you can find it in AD Users and Computers, under the "Object" tab.
You'll be prompted for the password.

h) You will be asked if you want to extend the schema. Choose "Yes".

i) When prompted, enter the DN of the Domain admin from step 7

(Note: If you get an error at this point saying "Can't extend schema on the Directory Server", you may need to modify the following registry key: HKLM\System\CurrentControlSet\Services\NTDS\Parameters\Schema Update Allowed – change the DWORD value to "1")

j) If the Schema was successfully extended, you will be prompted to enter the DN of a LDAP-UX profile entry. It's recommended to enter the example (cn=ldapuxprofile,CN=Configuration,DC=testdom,DC=com), otherwise ensure the DN is correct and all parent entries exist.

k) You'll be prompted again for the DN of the domain admin, to create the above profile. Enter the DN.

l) You'll be asked to provide any additional search hosts. If you have more than 1 DC on your LAN, you can enter them here.

m) You will be prompted for the base DN for LDAP user and group queries. Hit <Enter> to select the default.

n) When asked to accept the remaining defaults, hit <Enter> to choose "Yes".

o) You will be prompted to enter the proxy user DN information. In our case, the proxy user's DN is "CN=uxproxy,CN=Users,DC=testdom,DC=local". Enter the DN, and enter the password when prompted.

p) Answer, "Yes" when asked if ready to create profile entry. Press any key to continue.

q) You will be asked if you want to configure multi-domain support. In our case we only have a single domain in our forest, so we will choose "No".

r) Choose "Y" to restart the LDAP-UX daemon.

s) Enter the DN and password of the Unix Proxy user if prompted again.

t) Configure /etc/nsswitch.conf according to the example /etc/nsswitch.ldap

u) Copy /etc/krb5.conf.sample to /etc/krb5.conf and edit it to include your domain / Realm configuration. See the example below:

```
[libdefaults]
  default_realm = TESTDOM.LOCAL
  default_tkt_enctypes = DES-CBC-CRC
  default_tgs_enctypes = DES-CBC-CRC
  ccache_type = 2
#
#
[realms]
  TESTDOM.LOCAL = {
        kdc = dc.testdom.local:88
        kpasswd_server = dc.testdom.local:464
  }
#
#
[domain_realm]
  .testdom.local = TESTDOM.LOCAL
#
#
[logging]
  kdc = FILE:/var/log/krb5kdc.log
  admin_server = FILE:/var/log/kadmin.log
  default = FILE:/var/log/krb5lib.log
```

v) Run the ktutil utility to update the keytab file, using the following commands:

```
# ktutil
ktutil:  rkt /unix.keytab
ktutil:  l
```

You should see the following output:

```
slot KVNO Principal
---- ---- ---------------------------------------------------------
   1    1                 host/hpux@TESTDOM.LOCAL
```

Type the following:

```
ktutil:  wkt /etc/krb5.keytab
ktutil:  q
```

And verify that the keytab file has been updated:

```
# klist -e -k
```

If everything was successful, you should see the following:

```
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
---- ---------------------------------------------------------
   1 host/hpux.testdom.local@TESTDOM.LOCAL (DES cbc mode with
CRC-32)
```

w) Edit /etc/services as follows:

```
klogin       543/tcp           # Kerberos rlogin -kfall
kshell       544/tcp  krcmd    # Kerberos remote shell -kfall
ekshell      545/tcp  krcmd    # Kerberos encrypted remote shell -kfall
krbupdate    760/tcp  kreg     # Kerberos registration -kfall
kpasswd      761/tcp  kpwd     # Kerberos "passwd" -kfall
eklogin     2105/tcp           # Kerberos encrypted rlogin -kfall
#
kerberos5      88/udp      kdc             # Kerberos authentication
kerberos5      88/tcp      kdc             # Kerberos authentication
kerberos-adm   749/tcp     kerberos_adm    # Kerberos admin/changepw
kerberos-cpw   751/tcp     kerberos_master # Kerberos changepw
krb5_prop      754/tcp                     # Kerberos slave propagation
```

Now that all of the software has been installed and configured, we are almost ready to authenticate our UNIX users using Active Directory. In fact, the UNIX host can be configured to allow authentication over LDAP, although this is not what we want. We want to use Kerberos, and there's a good reason for this, which I'll demonstrate here by examining the traffic as it goes over the wire.

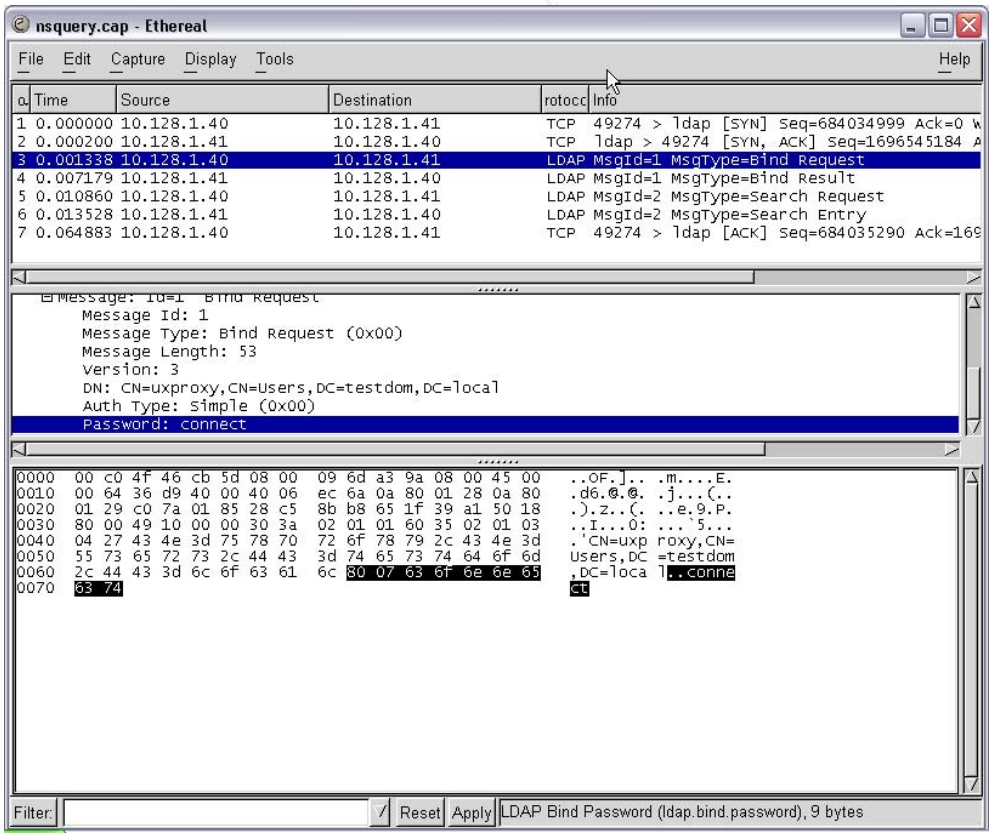
5.3. Examining LDAP Traffic

Assuming that the /etc/nsswitch.conf file has been configured as outlined above, the UNIX host should query LDAP for account information. As a simple test, we'll try to query the name service for "bob", an account which exists only in Active Directory and not in /etc/passwd:

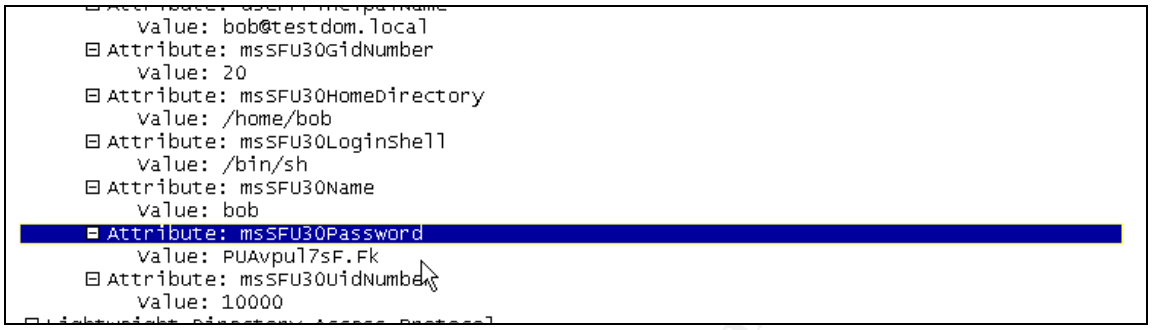
```
# nsquery passwd bob

Using "files ldap" for the passwd policy.

Searching /etc/passwd for bob
bob was NOTFOUND

Switch configuration: Allows fallback

Searching ldap for bob
User name: bob
User Id: 10002
Group Id: 125
Gecos:
Home Directory: /home/bob
Shell: /sbin/sh

Switch configuration: Terminates Search
```

So it appears that LDAP searches are working. What this tells us is that the host is configured properly, and the UNIX proxy account “uxproxy” is authenticating properly to the Domain Controller since it is able to retrieve POSIX attributes from the Directory.

However, it will be interesting to know exactly how this information is passing across the wire. Let’s take a look at a sniffer capture to see:



Zooming in on the capture data, it appears that information is being sent across the network in clear text! In fact, the password that we're seeing in this screenshot belongs to the UNIX proxy account. If we were to attempt to log in as a user at this point, all of the user's POSIX attributes would be sent across the wire in the clear as well, including what appears to be the password hash, as seen in this capture from Ethereal:



Remember the warning at the beginning of the section on LDAP-UX? All traffic between the LDAP client and the Domain Controller, **including authentication traffic**, is sent **in the clear.** This is not something we desire, nor is it something we can ignore. Let's see how Kerberos can help solve this problem.

6.0. Using Kerberos for Authentication

6.1. Configure PAM for Kerberos

The first thing we need to do is ensure that PAM is configured to use Kerberos for authentication. Notice the "auth sufficient" lines. This means that PAM still allows UNIX authentication, which allows us to keep a local "root" account on the UNX host in case the Kerberos realm is not available for any reason. Otherwise, we could lose root access on the server!

```
# the format for an entry is
# <service>     <module_type> <control> <module path> <options>
#
# see pam.conf(4) for more details
#
# Authentication management
#
login    auth sufficient     /usr/lib/security/libpam_unix.1
login    auth required       /usr/lib/security/libpam_krb5.1 try_first_pass
su       auth sufficient     /usr/lib/security/libpam_unix.1
su       auth required       /usr/lib/security/libpam_krb5.1 try_first_pass
dtlogin  auth sufficient     /usr/lib/security/libpam_unix.1
dtlogin  auth required       /usr/lib/security/libpam_krb5.1 try_first_pass
dtaction auth sufficient     /usr/lib/security/libpam_unix.1
dtaction auth required       /usr/lib/security/libpam_krb5.1 try_first_pass
OTHER    auth sufficient     /usr/lib/security/libpam_unix.1
OTHER    auth required       /usr/lib/security/libpam_krb5.1 try_first_pass
```

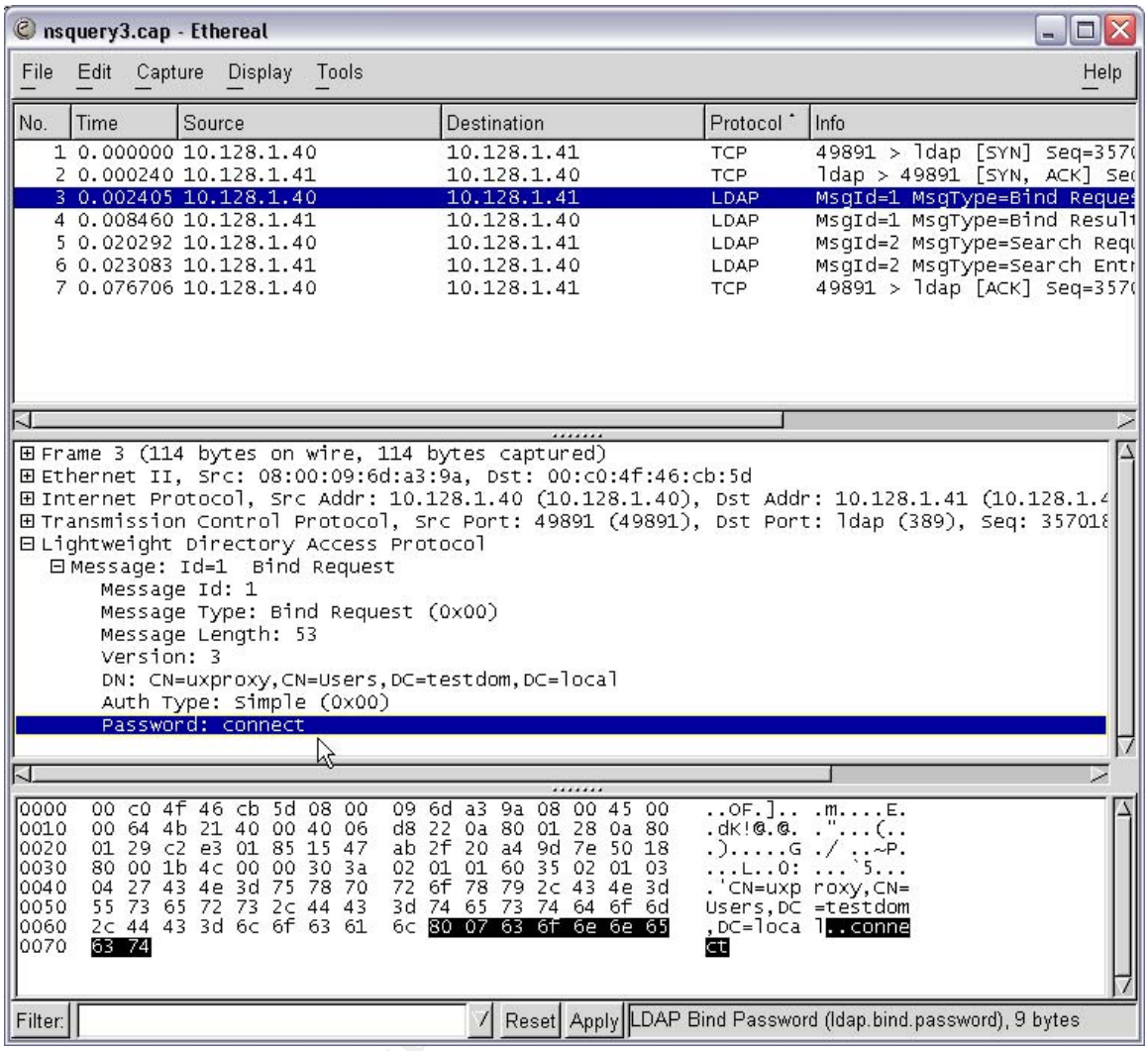
Here, we've configured PAM to try authenticating with the UNIX password database first. If this succeeds, the user is allowed in. If not, PAM will try Kerberos. If Kerberos fails, then PAM reports that authentication failed and the user is denied. More granular control can be achieved by configuring individual users via /etc/pam_user.conf. For more information on advanced configuration of PAM, I would strongly recommend reading the pam.conf man page [11].

6.2. How does PAM / Kerberos Change the Traffic?

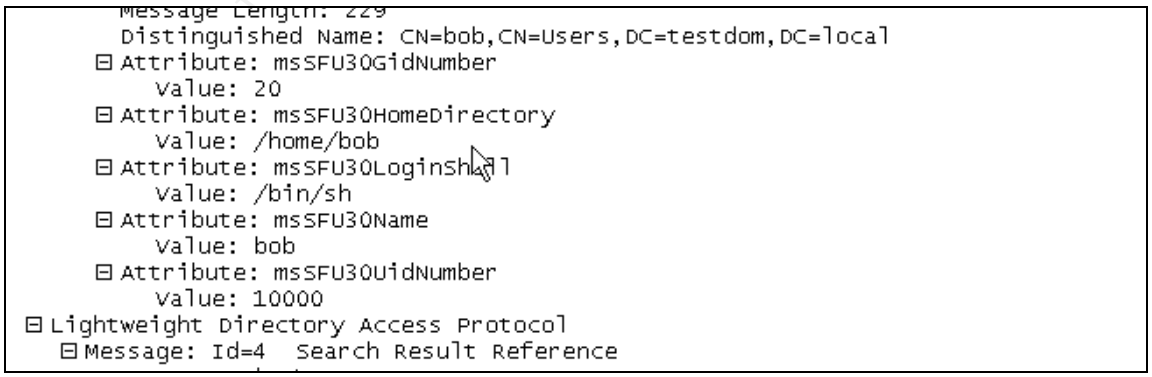
Now we can examine the traffic again to see whether we have solved the problem. First, flush the LDAP cache by typing the following command:

```
/opt/ldapux/bin/ldapclientd -f
```

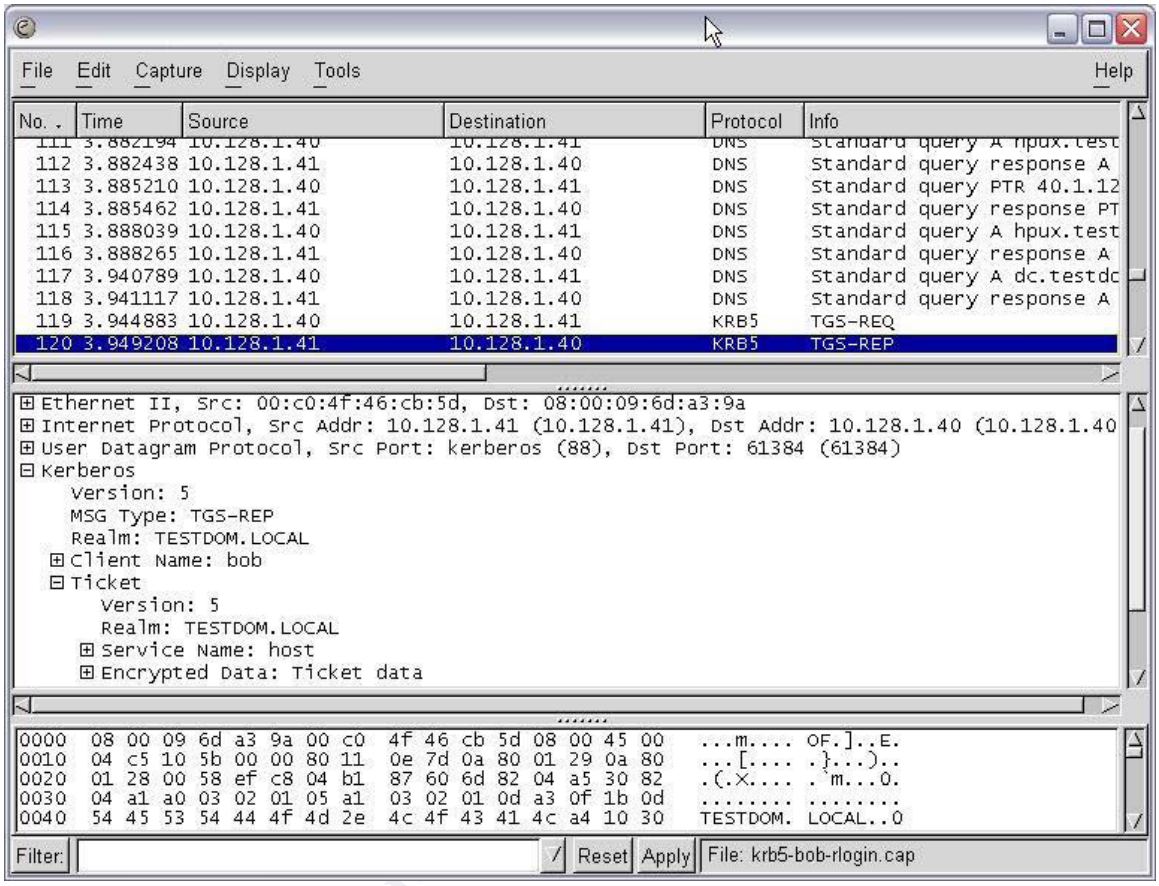
then do another nsquery and capture the traffic:



A new nsquery turns up the same problem! It appears that the proxy user authentication is not handled by Kerberos. What about the user's POSIX attributes and password hash? Does that information pass in the clear as well? We can test this by logging in as an AD user and examining the traffic:



In fact, it appears that *some* of the POSIX attributes (home directory, UID, GID) are passed in the clear. But the password is not – in fact it seems that once LDAP locates the account information, Kerberos takes over and actually manages the authentication piece, as seen in the following sniffer capture:



So now we're left with one final thing, and that is to find a way to encrypt the LDAP portion. That's what we'll cover in the next section.

7.0. Encrypting the LDAP Traffic

There are two solutions for LDAP encryption built in to Active Directory: Secure Sockets Layer (SSL) and IPSec.

7.1. LDAP over SSL

SSL seems to be the simpler of the two options – in theory all you need to do is configure the Domain Controller to communicate over SSL, and then configure the LDAP client to use SSL, and the session can be encrypted. The good news is that enabling SSL on the Domain Controller is a fairly straightforward task.

The bad news is that at the time of writing, HP does not support LDAP over SSL in their LDAP-UX product. HP has promised SSL support in the next release of LDAP-UX, but it doesn't appear that we'll be able to use it in our scenario.

However, since other vendors' implementations of PAM-LDAP may support LDAP over SSL (most do), it is useful to include some information on how to configure the Domain Controller to support it. When properly configured, the Domain Controller will be listening on TCP port 636 for SSL-LDAP requests.

7.1.1. Configure the DC for SSL

The steps are as follows:

> *a) Install an Enterprise Certificate Authority on a Windows 2000 Domain Controller. This automatically installs a certificate on a server.*
> *b) Open the Default Domain Controller Policy using the Group Policy Editor.*
> *c) Under **Computer Configuration**, click **Windows Settings**.*
> *d) Click **Security Settings**, and then click **Public Key Policies**.*
> *e) Click **Automatic Certificate Request Settings**.*
> *f) Use the wizard to add a policy for Domain Controllers.*
>
> *When you complete these steps, all Domain Controllers automatically request a certificate and can support LDAP using SSL port 636.*
>
> (Microsoft Corporation – KB Article 24078) [12]

Once this is done, you can see that the Domain Controller is listening on port 636 using the "netstat –an" command:

```
C:\Documents and Settings\Administrator>netstat -an

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:53             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:88             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:111            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:389            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:464            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:593            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:636            0.0.0.0:0              LISTENING
  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

So your SSL-enabled LDAP clients should now be able to communicate securely with the Domain Controller.

However, we will need to look at IPSec to secure the traffic between our HPUX box and the Domain Controller, since SSL is not yet supported in LDAP-UX.

7.2. IPSec

7.2.1. Overview of IPSec

Before we get too deep into configuring IPSec, we should take a moment to understand what exactly IPSec is and what it does. Configuring IPSec policies can be a complicated task, especially when working with very different platforms, and a single mis-configuration can cause unexpected outages, not to mention hours of frustration trying to troubleshoot. By understanding the components and terminology, we'll be able to make the right decisions when it comes to configuring IPSec on both Windows and UNIX.

IPSec is defined as a network-layer suite of protocols (as opposed to a single protocol) for authenticating and encrypting network traffic. By working at the network layer, it becomes (theoretically at least) easier to implement IPSec transparently to applications and users.

The main protocols in IPSec are the Internet Key Exchange (IKE), Authentication Header (AH), and the Encapsulating Security Payload (ESP).

IKE
IKE is defined in RFC 2409 as a protocol to "negotiate, and provide authenticated keying material for, security associations in a protected manner" ("RFC2409: The Internet Key Exchange (IKE)") [13]. In other words, IKE is where the two hosts decide how to communicate securely. These negotiations are carried out using the

Internet Security Association and Key Management Protocol (ISAKMP).

The IKE negotiation creates a "Security Association" or SA. An SA, according to the RFC, is "a set of policy and key(s) used to protect information" ("RFC2409: The Internet Key Exchange (IKE)") [13.] So IKE negotiates *how* to secure each phase and the SA contains all of the information: cipher, hash, authentication method, and the keys.

IKE negotiation happens in two phases. The RFC defines the first phase, also known as "Main Mode" or "Aggressive Mode" as "where the two ISAKMP peers establish a secure, authenticated channel with which to communicate" ("RFC2409: The Internet Key Exchange (IKE)") [13]. The second phase or "Quick Mode" is "where Security Associations are negotiated on behalf of services such as IPSec" ("RFC2409: The Internet Key Exchange (IKE)") [13].

AH
The Authentication Header, defined in RFC 2402, is used to provide authentication and protection against replay attacks ("RFC2402: IP Authentication Header") [14]. AH works in two modes: Transport Mode and Tunnel Mode; for our purposes we will be concerned with Transport Mode.

ESP
Encapsulating Security Payload is defined in RFC 2406. The RFC specifies that "ESP is used to provide confidentiality, data origin authentication, connectionless integrity, an anti-replay service (a form of partial sequence integrity), and limited traffic flow confidentiality" ("RFC2406: IP Encapsulating Security Payload") [15]. ESP is the core protocol of IPSec, and like AH can operate in Transport Mode and Tunnel Mode. (Fossen, "IPSec, RRAS & VPNs") [16]

Now we have a basic understanding of the concepts behind IPSec, we can examine how they are implemented within Windows, and compare this to HP's implementation in HPUX. We'll need to ensure that, when setting up the IPSec policy on the Domain Controller, it is compatible with the UNIX host.

7.2.2. Configure the DC for IPSec

One important note before we get started: the IPSec/9000 product on HPUX supports only commercial certificates such as Verisign or Baltimore, or preshared keys. There is no documentation on whether it supports certificates from an internal Windows 2000 CA, and there is no apparent way within any of the IPSec/9000 utilities to import or request a certificate from a private CA, so the assumption at this point is that it does not support them. Windows 2000 IPSec supports both commercial certificates and preshared keys. Therefore, in the test environment I will use preshared keys since it is impractical to purchase a commercial certificate for testing purposes.

It is very important to remember that in Windows, the Preshared Keys are stored in **cleartext** in both the registry and the IPSec Policy console. The key is also stored in cleartext in the HPUX IPSec GUI management tool. However, this cleartext key is only used for the initial authentication, not encryption, so it can't be used to decrypt IPSec communications. (Fossen, "IPSec, RRAS & VPNs", p. 67) [16] Nonetheless, if you decide to use Preshared Keys for interoperability, you must make absolutely certain that your Domain Controller is protected from intruders. Otherwise, I strongly recommend using a commercial PKI solution for a production system.

In Windows, you can configure IPSec policies per-machine, or through Group Policy. For this example, we'll configure IPSec using the "Default Domain Controllers Policy".

7.2.2.1. Create the Policy

a) Open the Default Domain Controllers Policy MMC snap-in.

b) Drill down to "Computer Configuration" : "Windows Settings" : "Security Settings" : "IP Security Policies on Active Directory"

c) Right-click the "IP Security Policies on Active Directory" container, and choose "Create IP Security Policy"

d) Click Next

e) In the Name: field, type "HPUX IPSec Policy". Enter a description and click "Next".

f) Ensure "Activate the Default response rule" is checked, and click "Next".

g) In the "Initial authentication method" window, select "Use this string to protect the key exchange (preshared key) and enter the preshared key. Make a note of this key; you will need it on the HPUX host later. Click "Next".
(again, it's at this point where you would configure use of a commercial CA certificate rather than a preshared key, in a production environment).

h) Make sure "Edit Properties" is checked, and click "Finish."

What we've done this far is create a new policy that will use a Preshared key for the IKE Phase 1 negotiation.

Why not just use one of the default IPSec Policy templates such as "Secure Server"? Aside from the possibility of something unexpected happening (e.g., see Microsoft article Q232817) as a result of applying these policies to a Domain

Controller, in this case it makes more sense to create this policy from scratch, so we know *exactly* what is going into the policy before we apply it – in accordance with SANS best practices (Fossen, "IPSec, RRAS & VPNs", p. 62) [16]

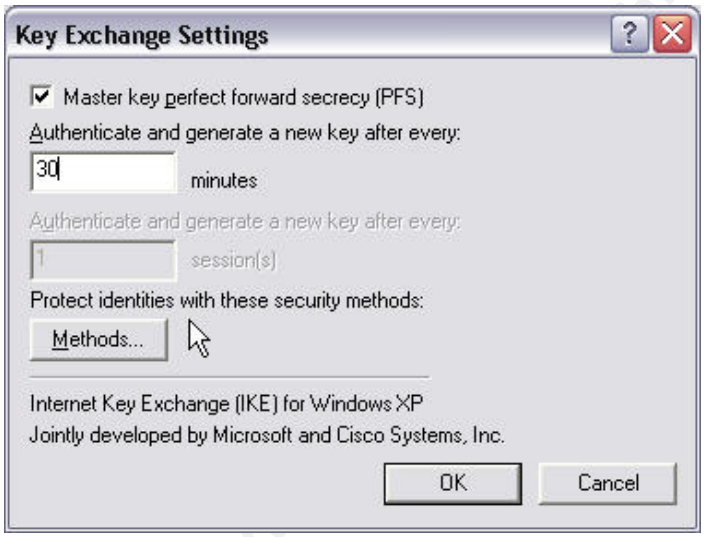
7.2.2.2. Set up IKE Phase I

a) In the "HPUX IPSec Policy Properties" window, click on the "General" tab.

b) Click "Advanced"

c) Check "Master Key Perfect Forward Secrecy". This will ensure that all keys are generated randomly, and is supported by IPSec/9000

d) Change the "Minutes" field to 30, as per SANS best practices. (Fossen, "IPSec, RRAS & VPNs", pp. 50-51) [16]



e) Click on "Methods…"

f) Change the Security Method Preference order to set "IKE – 3DES – MD5 – Medium (2)" as the preferred method (i.e. "move up" until it is at the top.) Highlight all of the other methods and click "Remove".
This will ensure that we are using 3DES encryption for the IKE exchange, MD5 for integrity, and a 1024-bit prime number for the Diffie-Hellman group - all of which is supported by IPSec/9000. Again, this is in accordance with SANS best practices (Fossen, "IPSec, RRAS & VPNs", pp. 50-51) [16]

g) Click OK and OK again.

7.2.2.3. Configure Phase II

We have now configured the IKE Phase I negotiation settings. Next we'll work on the Phase II negotiation:

a) Click on the "Rules" tab

b) Click "Add…" then click Next

c) Select "This rule does not specify a tunnel" and click Next

d) Select "All network connections" and click Next

e) Select "Use this string to protect the key exchange (preshared key)" and type in the preshared key. Click next.
(Once again, in a production environment, this is where you would configure it to use a commercial CA certificate, rather than a preshared key)

f) We don't want to encrypt all traffic, either IP or ICMP. We only want to encrypt traffic between HPUX and the Domain Controller, so we will want to create our own "Filter". Click "Add"

g) Change the name to "HPUX Selector". ("Selector" is a more appropriate name than "Filter" since we aren't filtering out traffic as in a firewall rule;

we are selecting which traffic to apply IPSec security to). Enter a detailed description, and click “Add”.

h) The “IP Filter Wizard” appears. Click Next.

i) Under “Source Address”, choose “A Specific IP Address”, and enter the IP address of the HPUX host.

j) Ensure that “Mirrored” is checked. “Click Next

k) Under “Destination Address”, choose “My IP Address”. Click Next

l) In “Select a Protocol Type”, choose “TCP” and click Next.

m) We could choose to encrypt only TCP port 389 (LDAP) but why not encrypt everything? It simplifies things and has the benefit of added security. Select “From Any Port” and “To Any Port”, click “Next” and then click Finish. Click Close.

n) You should now see the “HPUX Selector” listed in the “IP Filter Lists”. Make sure to select it (click the radio button) and click Next.

o) Check the “Use Add Wizard,” and click “Add”

p) The “Filter Action Wizard” appears. Click Next.

q) In the Name field, type “HPUX Filter Action”, and type a description. Click Next.

r) Choose “Negotiate Security” and click Next

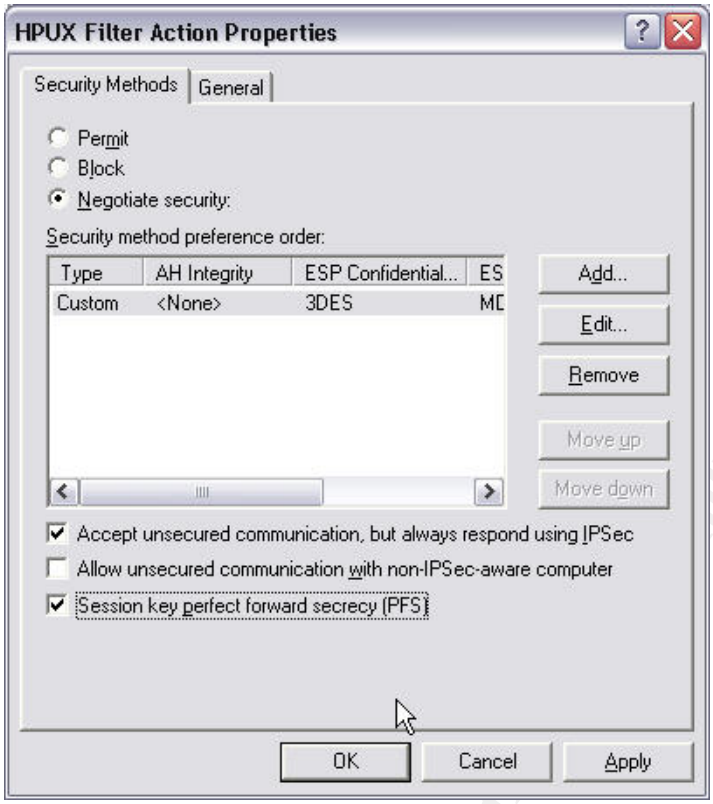
s) Choose “Do not communicate with computers that do not support IPSec” and click Next.

t) Choose “Custom” and click “Settings”.

u) Under “Integrity Algorithm”, choose “MD5” and for “Encryption Algorithm” choose 3DES. Select “Generate a new key every 900 seconds”

v) Click “OK” and “Next”, then ensure “Edit Properties” is selected, and click “Finish”

w) Ensure “Accept Unsecured communication, but always respond with IPSec” and “Session Key Perfect Forward Secrecy (PFS)” are both selected, and click “OK”.



x) Select the radio button for "HPUX Filter Action"

y) Click "Next", then Finish and then Close.

We have now completed setting up the Phase II rules that will be negotiated for encrypting the IPSec traffic and managing the keys, again following the SANS best practices for Phase 2 SAs (Fossen, "IPSec, RRAS & VPNs", p. 63) [16]

7.2.3. Assign the IPSec Policy

The last thing we need to do is assign the policy. In the "IP Security Policies on Active Directory" container, right-click the "HPUX IPSec Policy" and choose "Assign". This will force communications between the DC and the HPUX host to use IPSec (once IPSec is installed and configured on the HPUX host).

To sum up what has been done, we have created an IPSec policy to require IPSec for all communications between the HPUX host and the Domain Controller, using a Preshared Secret for the IKE authentication; set IKE to use 3DES Encryption to encrypt the identity of the peers, MD5 for Integrity, and a "medium" (i.e. 1024-bit prime) Diffie-Hellman group (A.K.A. Diffie-Hellman Group 2); ESP is set to use MD5 for Integrity and 3DES for Encryption.

Of course this means we'll next need to configure IPSec on the HPUX host to match the policy we set on the DC. The next steps will outline how to do this.

7.3. Configure IPSec on the HPUX Host

(Note: As I discovered in my testing, older HP hardware such as the 700-series workstations won't run the IPSec software properly, even though it will install without warning. Unexpected behavior such as password errors and daemons not starting are symptomatic of older CPUs which are too slow to handle the processing required by the IPSec/9000 product.)

7.3.1. Install and Configure IPSec/9000

a) Install the IPSec/9000 product (J4256AA) if not already installed.

b) run "ipsec_admin –newpasswd" to set the password. The password must be at least 15 characters and spaces are allowed, so it's a good idea to use a pass phrase that's easy to remember, yet still complex.

7.3.2. Create an IPSec Policy

Create a policy using the "ipsec_mgr" utility. This is a GUI program that requires an X server either on the HPUX machine, or on a remote terminal / workstation (e.g. Hummingbird Exceed on Windows or a Linux workstation running an X desktop).

a) run "ipsec_mgr"

b) Select the "IPSec Policies" tab

c) Click "Create"

d) type a name for the policy, e.g. "DC-Policy"

e) Un-check the "Exclusive" box, if checked

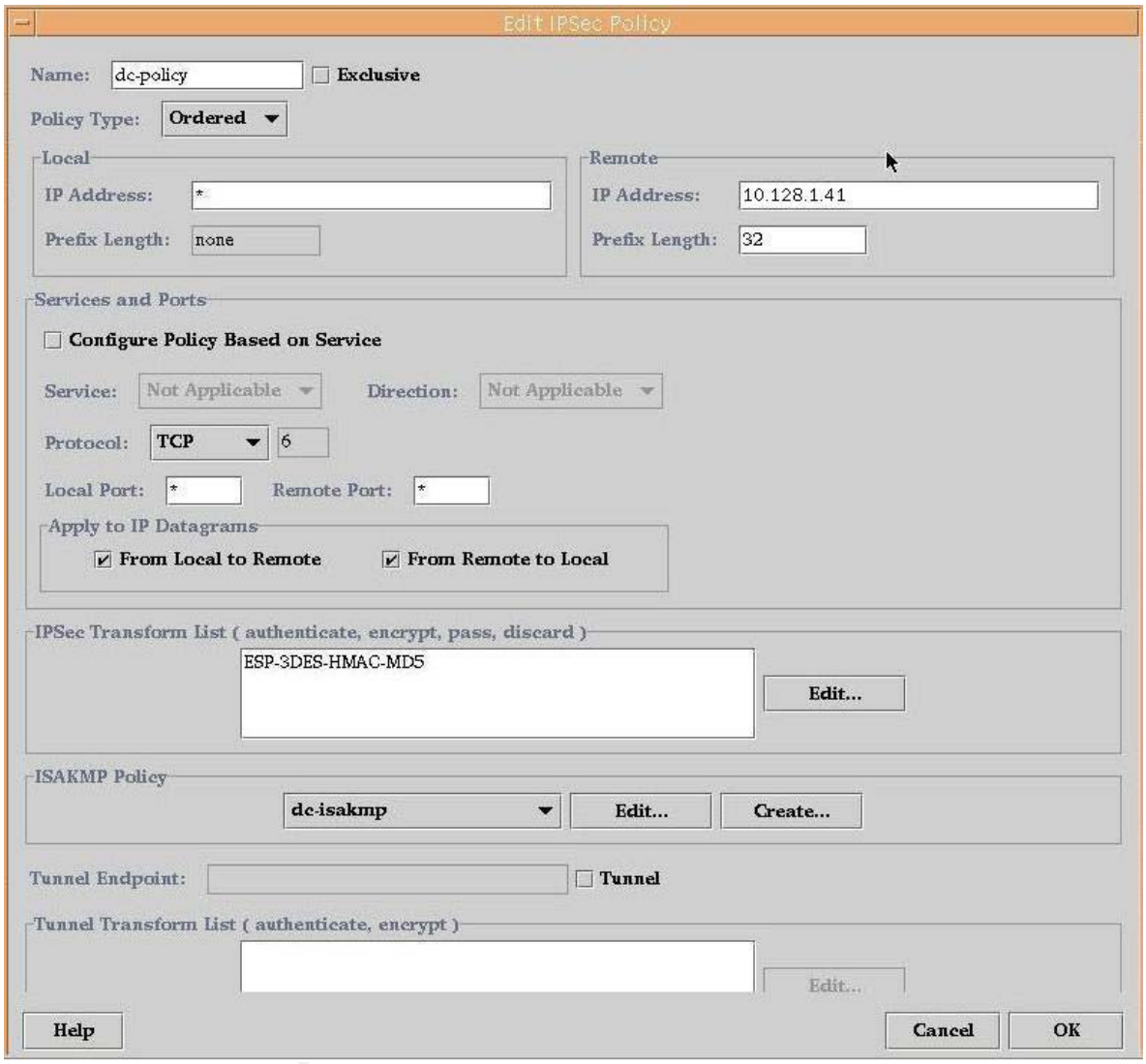
f) for "Policy Type", choose "Ordered"

g) in "Local", choose "*" and in Remote, enter the IP Address of the Domain Controller.

h) Under "Services and Ports", for Protocol, choose TCP

i) For "Apply to IP Datagrams", choose "From local to remote" and "From remote to local"

j) Under IPSec Transform List, click “Edit” and choose “ESP-3DES-HMAC-MD5”



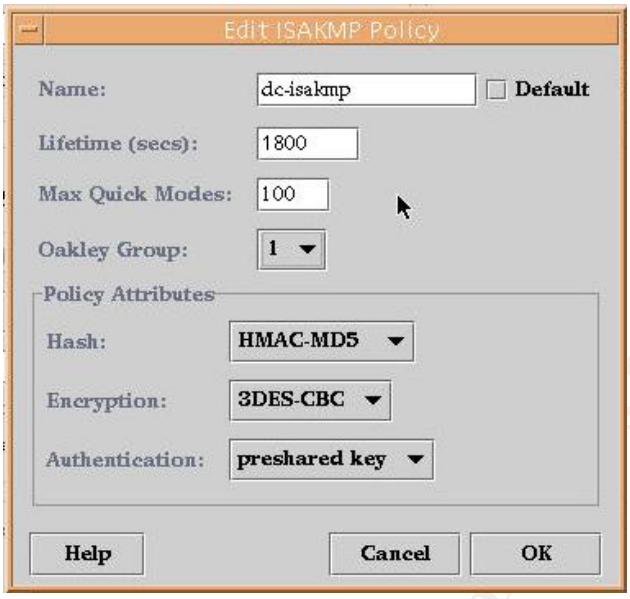
7.3.3. Create an ISAKMP Policy

a) Under ISAKMP Policy, click “Create”

b) Fill in the fields as follows, and click OK:

- Name = DC-ISAKMP
- Lifetime = 1800 seconds (i.e. 30 minutes),
- Max Quick Modes = 100
- Oakley Group = 1
- Hash = HMAC-MD5
- Encryption=3DES-CBC

- Authentication=preshared key (once again, in a production environment you would choose to use a certificate rather than a preshared key)



7.3.4. Configure the Preshared Key

a) Click the “Preshared Keys” Tab

b) Click Create

c) Type the IP of the Domain Controller, and type the same preshared key that you entered in the policy on the Domain Controller.

d) Click OK, and click Exit

e) When asked to save changes to the policy file, click OK

7.3.5. Configure IPSec to start at Bootup

a) Click the “Options…” menu at the top of the IPSec Manager window

b) Select “Boot-up options”

c) Click “Enable IPSec at bootup”

d) In Policy Filename, type /var/adm/ipsec/policies.txt

e) Click OK

7.3.6. Start and Test IPSec

a) Start IPSec with "ipsec_admin –start"
- if everything is configured properly, you should see a series of successful "IPSEC_ADMIN" messages as the daemons and IPSec kernel are started, e.g:

```
# ipsec_admin -start
IPSEC_ADMIN: Please enter the IPSec password: ******************
IPSEC_ADMIN: Starting up the secauditd program.
IPSEC_ADMIN: ALERT-Starting up IPSec/9000.
IPSEC_ADMIN: Starting up the ikmpd program.
IPSEC_ADMIN: The ikmpd program successfully started up.
IPSEC_ADMIN: Starting up the secpolicyd program.
IPSEC_ADMIN: Starting up the IPSec kernel.
IPSEC_ADMIN: IPSec kernel successfully started up.
IPSEC_ADMIN: Security Association Data Base successfully flushed.
IPSEC_ADMIN: IKE MM SAs successfully flushed.
```

b) Check status with "ipsec_admin –status'
- if everything is ok, you should see a report similar to the following:

```
----------------- IPSec Status Report -----------------
secauditd program: Running and responding
secpolicyd program: Running and responding
ikmpd program: Running and responding
IPSec kernel: Up
IPSec Audit level: Error
IPSec Audit file: /var/adm/ipsec/auditWed-Jul--9-15-32-27-2003.log
Max Audit file size: 100 KBytes
IPSec Policy file: /var/adm/ipsec/policies.txt
Level 4 tracing: None
-------------- End of IPSec Status Report -------------
```

Test that the policy is working, using the "ipsec_policy" command, e.g.:

```
ipsec_policy -da 10.128.1.41 -p TCP
```

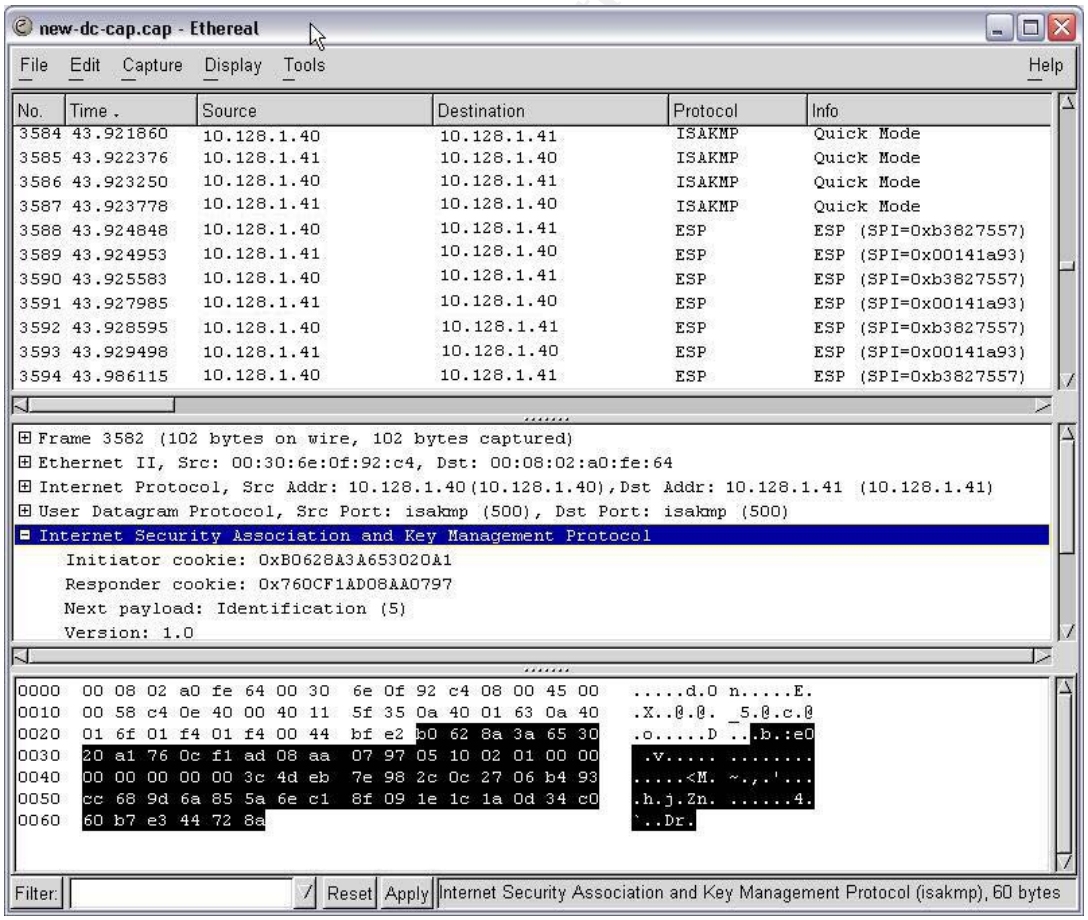
The above command will test how the policy will be applied against the IP address of the Domain Controller, using the TCP protocol. It should respond with something like:

```
------------------------ Ordered Policy Rule -------------------------
Rule ID: dc-policy
Cookie: 1    State: Ready
Src IP Addr: *  Src Port number: *
Dst IP Addr: 10.128.1.41  Prefix: 32   Dst Port number: *
Network Protocol: TCP    Direction: outbound
Filter: Secure
Shared SA: Yes
Number of SA(s) Needed: 1
Number of SA(s) Created: 1
Kernel Requests Queued: 0
```

```
-- SA Number 1 --
   Security Association Type: ESP
   Encryption Algorithm: 3DES-CBC
   Authentication Algorithm: HMAC-MD5
   SPI (hex): 40EE387C
   SPI updated: ISAKMP

------------------------ Ordered ISAKMP Rule -------------------------
Rule ID: dc-isakmp
Rule Type: Ordered    Cookie: 9162
Remote IP Address: 10.128.1.41   Prefix: 32
Group Type: 1    Authentication Method: Pre-shared Keys
Authentication Algorithm: HMAC-MD5    Encryption Algorithm: 3DES-CBC
Number of Quick Modes: 100    Lifetime (seconds): 1800
```

c) Test using an AD account, and capture the traffic with a sniffer to prove IPSec is working.
- test with nsquery, e.g. “nsquery passwd bob”, to see if the HPUX host is able to talk to the LDAP services on the domain controller
- test logging on as an AD user, using “su”, to ensure that authentication / Kerberos is working over IPSec

The sniffer capture should show something like the following:



As the sniffer output above shows, all TCP traffic between the UNIX host and the Domain Controller is now encrypted. You can see the ISAKMP key exchange negotiations, and the ESP headers. The most important thing to see is that there are no more clear text passwords being passed across the network in the back-end authentication mechanism, so it would appear that our work here is finally done.

Now that we have configured a working IPSec policy between the Domain Controllers and the HPUX machine, we can use this as a template to create other IPSec policies. For example, we could now create a Default Domain policy to require IPSec for all clients when communicating with the HPUX Server, thereby ensuring that *everything* to and from the HPUX machine is encrypted. We've proven that it can work; all that is required is further configuration, testing and deployment.

8.0. Conclusion

We have come a long way in our quest to authenticate our UNIX users against Active Directory. As we've learned, even though Microsoft advertises Active Directory as being standards-based, in reality there is quite a bit of work involved to achieve the true interoperability required. But once the work is done, maintaining the single user database and setting policies on the accounts is easily accomplished, and the groundwork has been laid for the ability to improve the overall security of the environment.

Vendor support for this type of interoperability has proven very important, since the configuration of the tools can be quite complex and access to documentation and support is essential. HP provides excellent support for Windows / UNIX interoperability with tools such as LDAP-UX, PAM-Kerberos, and IPSec/9000. Microsoft also provides tools for interoperability, specifically Server for NIS as part of Microsoft Services for Unix 3.0, and the built-in LDAP, Kerberos, and IPSec features of Windows 2000.

Microsoft's support for IPSec is excellent, and is possibly the one area where they haven't yet "embraced and extended" the standard to suit their own needs. However, it did take a good deal of trial-and-error to get IPSec working between Windows and UNIX. Most of this difficulty was simply due to the complexity of IPSec itself and the multiple possible options in configuring it. It's very true that one small configuration difference will render IPSec unusable, and it can take a great deal of time and troubleshooting skills to find and correct the mistake. Planning and testing is a necessity to help avoid this, and I would recommend sticking to SANS best practices wherever possible.

While our test environment was built using preshared keys for IPSec authentication, it is more secure and therefore recommended to use digital certificates. Since HP's IPSec/9000 doesn't seem to support a private CA, the only option would be to use certificates from a commercial CA.

Putting all of the pieces together can be a bit of a challenge, and as we've seen, it's very important to verify that the network traffic you are generating is truly secured, so that there are no surprises (such as passwords being transmitted in the clear). Remember some of the lessons learned from both careful reading of the documentation and our sniffer tests – don't assume that just because the steps have been followed that everything is working the way you *want* it to work.

A test lab is imperative before implementing a solution like this on production systems, since many of the steps involved can have irreversible effects and could potentially cause denial of service. As with anything, you'll want to make sure that this fits into your own environment. Additional complications such as multiple domains and trusts, internal politics, or unsupported configurations may make this solution unsuitable for some environments.

Ultimately, it took several weeks of testing and careful planning before TestDom was able to start using Active Directory authentication on their production UNIX systems. In the end, however, they were able to show compliance to the outstanding issues raised in their ISO audit, with a consistent user accounts and password policy across all systems. This, combined with their findings of a reduction in overall administrative effort and helpdesk requests, as well as the potential to improve their overall network security using IPSec policies has made the task a worthwhile effort.

List of References

1. Silveira, Julio – "Auditing the Windows 2000 Authentication Process"– SANS InfoSec Reading Room, 2001
URL: http://www.sans.org/rr/win2000/audit_w2k.php

2. De Clercq, Jan – "Windows 2000 Authentication: Under The Hood" – Compaq Computer Corporation, 1999
URL: http://activeanswers.compaq.com/aa_downloads/6/100/225/1/42407.doc

3. "Windows 2000 Kerberos Authentication" – Microsoft Corporation, 1999
URL: http://www.microsoft.com/windows2000/techinfo/howitworks/security/kerberos.asp

4. Potter, Tridgell, Mummaneni, Vernooij, Terpstra – "Integrated Logon Support Using WinBind" – Chapter 15, "SAMBA Project Documentation", 21 April, 2003
URL: http://www.samba.org/samba/devel/docs/html/winbind.html

5. "Introduction to Windows Services for Unix 3.0" – Microsoft Corporation, 2002
URL: http://www.microsoft.com/windows/sfu/docs/sfuwp.doc

6. "Installing Microsoft Windows Services for UNIX 3.0" – Microsoft Corporation
(from Microsoft Windows Services for UNIX 3.0 installation cd – INSTALL.htm)

7. "Server for NIS overview" – Microsoft Corporation, 2003
URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windows2000serv/deploy/sfu/servnis.asp

8. Gallagher, Eric – "Password Security in NIS Systems" – SANS Infosec Reading Room – July 19, 2001
URL: http://www.sans.org/rr/papers/6/107.pdf

9. "Installing and Administering LDAP-UX Client Services with Microsoft Windows 2000 Active Directory" – Hewlett Packard Company, 2002
URL: http://docs.hp.com/hpux/pdf/J4269-90017.pdf

10. "LDAP-UX Integration B.03.00 Release Notes, Fourth Edition" – Hewlett Packard Company, 2002
URL: http://docs.hp.com/hpux/pdf/J4269-90018.pdf

11. pam.conf (4) man page – Hewlett Packard Company, October 1997

12. Microsoft Knowledge Base Article 247078 – "How to Enable Secure Socket Layer (SSL) Communication Over LDAP" – Microsoft Corporation, 2003
URL: http://support.microsoft.com/default.aspx?scid=kb;en-us;247078

13. Harkins, D. & Carrel, D. – "RFC2409: The Internet Key Exchange (IKE)" – The Internet Society, 1998
URL: http://www.ietf.org/rfc/rfc2409.txt?number=2409

14. Kent, S. & Atkinson, R. – "RFC2402: IP Authentication Header" – The Internet Society, 1998
URL: http://www.ietf.org/rfc/rfc2402.txt?number=2402

15. Kent, S. & Atkinson, R. – "RFC2406: IP Encapsulating Security Payload" – The Internet Society, 1998
URL: http://www.ietf.org/rfc/rfc2406.txt?number=2406

16. Fossen, Jason – "IPSec, RRAS & VPNs" – SANS Institute, 2002

17. "Windows 2000 Kerberos Interoperability"- Microsoft Corporation, 2000
URL: http://www.microsoft.com/windows2000/techinfo/howitworks/security/kerbint.asp

18. CERT Bulletin – "Microsoft Corporation Information for VU#192995" – CERT Coordination Center, May 29, 2003
URL: http://www.kb.cert.org/vuls/id/AAMN-5CKTFF

19. CERT Bulletin – "Microsoft Services for UNIX Network File System (NFS) server is vulnerable to denial of service via memory leak" – CERT Coordination Center, Sept 4, 2001
URL: http://www.kb.cert.org/vuls/id/581603

20. CERT Bulletin – "Microsoft Services for UNIX Telnet server is vulnerable to denial of service via memory leak" – CERT Coordination Center, August 7, 2001
URL: http://www.kb.cert.org/vuls/id/994851

21. CERT Bulletin – "Microsoft Information for VU#952611" – CERT Coordination Center, August 7, 2001
URL: http://www.kb.cert.org/vuls/id/JSHA-54X5ZA

22. "iPlanet Straight Talk: LDAP vs Active Directory" – LDAPGuru
URL: http://www.ldapguru.org/modules/news/article.php?storyid=15

23. "Active Directory Overview" – Microsoft Corporation, 1999
URL: http://www.microsoft.com/windows2000/server/evaluation/features/dirlist.asp#heading9

24. Fossen, Jason – "Windows 2000/XP Active Directory" – SANS Institute, 2003

25. "Active Directory Architecture" – Microsoft Corporation, 2003
URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/ad/windows2000/deploy/projplan/adarch.asp

26. "Step-by-Step Guide to Kerberos 5 (krb5 1.0) Interoperability" – Microsoft Corporation, 2000
URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windows2000serv/howto/kerbstep.asp

27. "Introduction to OpenLDAP Directory Services" – OpenLDAP Foundation, 2003
URL: http://www.openldap.org/doc/admin21/intro.html

28. "Windows Services for UNIX 3.0" – Microsoft Corporation, 2003
URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windows2000serv/deploy/sfu/sfu.asp

29. "Installing LDAP-UX Client Services with Microsoft Windows 2000 Active Directory"
URL: http://docs.hp.com/hpux/pdf/J4269-90017.pdf

30. "Installing and Administering LDAP-UX Client Services" – Hewlett Packard Company, 2002
URL: http://docs.hp.com/hpux/pdf/J4269-90016.pdf

31. "Installing and Administering LDAP-UX Client Services with Microsoft Windows 2000 Active Directory" – Hewlett Packard Company, 2002
URL: http://docs.hp.com/hpux/pdf/J4269-90017.pdf

32. Isler, Don – "A Basic Step-by-Step Summary of Kerberos v5.1 Setup on HPUX Platform" – Hewlett Packard Company, 2003
URL: http://www1.itrc.hp.com/service/cki/docDisplay.do?docLocale=en_US&docId=200000068724106

33. "Installing and Administering IPSec/9000" – Hewlett Packard Company, 2001
URL: http://docs.hp.com/hpux/pdf/J4255-90011.pdf

34. "Step-by-Step Guide to Internet Protocol Security (IPSec)" – Microsoft Corporation, 2003
URL: http://www.microsoft.com/windows2000/techinfo/planning/security/ipsecsteps.asp

35. Yeong, Howes & Kille – "RFC1777: Lightweight Directory Access Protocol (LDAP)" – The Internet Society, 1995
URL: http://www.ietf.org/rfc/rfc1777.txt?number=1777

36. Wahl, Howes & Kille – "RFC2251: Lightweight Directory Access Protocol (LDAP) v. 3" – The Internet Society, 1997
URL: http://www.ietf.org/rfc/rfc2251.txt?number=2251

37. HP ITRC Document Id HPSBUX0209-221 – "Security Vulnerability in LDAP-UX Integration" – Hewlett Packard Company, 2002
URL: http://www2.itrc.hp.com/service/cki/search.do?category=c0&mode=text&searchString=ldap-ux&searchCrit=allwords&docType=Security

Appendix A

```
 ---------------------------------------------------------------
 Source: HEWLETT-PACKARD COMPANY
 SECURITY BULLETIN: HPSBUX0209-221
 Originally issued: 30 Sep 2002
 SSRT2346  Security Vulnerability in LDAP-UX Integration
 ---------------------------------------------------------------

NOTICE: There are no restrictions for distribution of this
Bulletin provided that it remains complete and intact.

The information in the following Security Bulletin should be
acted upon as soon as possible.  Hewlett-Packard Company will
not be liable for any consequences to any customer resulting
from customer's failure to fully implement instructions in this
Security Bulletin as soon as possible.

 ---------------------------------------------------------------
PROBLEM: pam_authz vulnerabilities in LDAP-UX Integration product

IMPACT:  Potential for increased privilege.

PLATFORM: HP9000 Series 700/800 running HP-UX releases 11.00
          and 11.11.

SOLUTION: Update to B.03.01 version

MANUAL ACTIONS: Yes - Update
                HP-UX 11.00 and 11.11, update to
                LDAP-UX version B.03.01.

AVAILABILITY: A new version is available on http://software.hp.com.

 ---------------------------------------------------------------
 A. Background
    The LDAP-UX Integration product (J4269AA) versions B.02.00 and
    B.03.00 contain defects which can cause r-commands to execute
    under the wrong user id.

 B. Recommended solution
    Update to the LDAP-UX Integration B.03.01 version which is
    available on http://software.hp.com, under the Internet and
    Security solutions.

 C. To subscribe to automatically receive future NEW HP Security
    Bulletins from the HP IT Resource Center via electronic
    mail, do the following:

    Use your browser to get to the HP IT Resource Center page
    at:

       http://itrc.hp.com

    Use the 'Login' tab at the left side of the screen to login
    using your ID and password.  Use your existing login or the
    "Register" button at the left to create a login, in order to
    gain access to many areas of the ITRC.  Remember to save the
    User ID assigned to you, and your password.

    In the left most frame select "Maintenance and Support".

    Under the "Notifications" section (near the bottom of
    the page), select "Support Information Digests".

    To -subscribe- to future HP Security Bulletins or other
    Technical Digests, click the check box (in the left column)
    for the appropriate digest and then click the "Update
    Subscriptions" button at the bottom of the page.
```

or

To -review- bulletins already released, select the link (in the middle column) for the appropriate digest.

To -gain access- to the Security Patch Matrix, select the link for "The Security Bulletins Archive". (near the bottom of the page) Once in the archive the third link is to the current Security Patch Matrix. Updated daily, this matrix categorizes security patches by platform/OS release, and by bulletin topic. Security Patch Check completely automates the process of reviewing the patch matrix for 11.XX systems.

For information on the Security Patch Check tool, see: http://www.software.hp.com/cgi-bin/swdepot_parser.cgi/cgi/displayProductInfo.pl?productNumber=B6834AA

The security patch matrix is also available via anonymous ftp:

ftp://ftp.itrc.hp.com/export/patches/hp-ux_patch_matrix/

On the "Support Information Digest Main" page: click on the "HP Security Bulletin Archive".

D. To report new security vulnerabilities, send email to

security-alert@hp.com

Please encrypt any exploit information using the security-alert PGP key, available from your local key server, or by sending a message with a -subject- (not body) of 'get key' (no quotes) to security-alert@hp.com.

--