GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Security Essentials: Network, Endpoint, and Cloud (Security 401)"
at http://www.giac.org/registration/gsec

USING SSL TO SECURE LDAP TRAFFIC TO MICROSOFT DOMAIN CONTROLLERS

GIAC (GSEC) Gold Certification

Author: Andrew Reid, andy@andyreid.co.uk
Advisor: Egan Hadsell

Accepted: September 18, 2011

Abstract

As Microsoft's Active Directory continues to gain momentum as a primary user authentication directory more application owners are requesting the use of Lightweight Directory Access Protocol (LDAP) for user authentication within their applications. By default Microsoft domain controllers do not provide a secure method for third party connections when using LDAP. This can create a false sense of security and the potential for loss of confidentiality. This paper will explain the configuration of LDAP over Secure Sockets Layer (SSL) to secure communication between application servers and Microsoft domain controllers.

© 2011 SANS Institute, Author retains full rights.

© 2011 The SANS Institute Author retains full rights.

1. Introduction

When deploying application servers there is often a choice to be made regarding the authentication of user credentials. In most cases this is to use an internal account database or an LDAP directory such as Microsoft Active Directory Domain Services.

If used and configured correctly LDAP can provide a secure way to authenticate users to a single authoritative data source simplifying the user logon experience and also reducing account overheads for account management.

Although relatively simple to configure and use, LDAP based authentication has issues. Unless an additional security protocol such as SSL or Transport Layer Security (TLS) is used to secure LDAP traffic is sent across the network unencrypted leaving it vulnerable if intercepted.

This paper outlines the issues when using insecure LDAP for authentication, details the configuration steps required within a domain infrastructure to allow the secure passing of authentication credentials and finally shows authentication requests being passed using secure LDAP.

Before proceeding with the configuration, generation of certificates and enabling secured LDAP section 2 provides a basic overview of the technologies, functionality and key terms discussed throughout this paper.

2. Public Key Cryptography

2.1 Introduction to X.500 Directory Services

An X.500 Directory is a store for objects; it is by design hierarchical which means objects can be stored under Organisational Units or Containers as shown in Figure 2.1 below.

Each object within the directory can have multiple attributes for example a user object will include attributes that define their mail account, address and userPrincipalName and userPassword shown in Figure 2.2.

© 2011 The SANS Institute Author retains full rights.

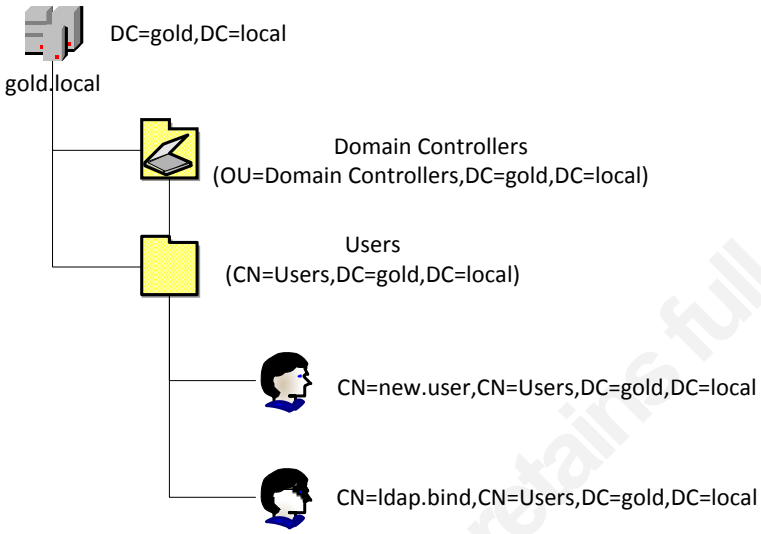


Figure 2.1 LDAP Directory Structure

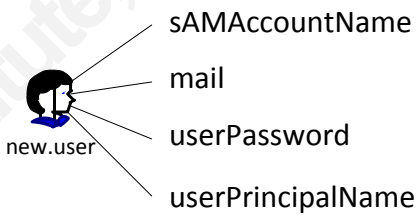


Figure 2.2 User Account Attributes

2.2 Introduction to LDAP

LDAP is an application protocol designed to provide access to an X.500 Directory. It enables read and write access to the directory through the use of simple messages such as Bind, Search, Add and Delete complete information regarding LDAP is documented in rfc2251 (Wahl, Howes, & Kille, 1997).

When using LDAP to authenticate users it is usual to define a base Distinguished Name (DN) or location where searches for objects start, in the case of Figure 2.1 if the

© 2011 The SANS Institute Author retains full rights.

© 2011 SANS Institute, Author retains full rights.

base DN was DC=gold, DC=local this would ensure all accounts within the directory can be found.

Also to authenticate to the directory a bind account is used, a DN for the bind user is supplied to the authenticating application. For the example shown in Figure 2.1 this would be CN=ldap.bind, CN=Users, DC=gold, DC=local where Common Name (CN) is the name of the object within the directory.

2.3 Introduction to Public/Private-Key Cryptography

For the purposes of this paper public/private-key cryptography is the underlying technology which allows the secure transfer of information between the application server and the domain controller. The reason for this security is these public and private keys are numbers which are mathematically linked in such a way that information encrypted with one key can only be decrypted with the other shown in Figure 2.3. For these reasons the private key should always be kept secure.

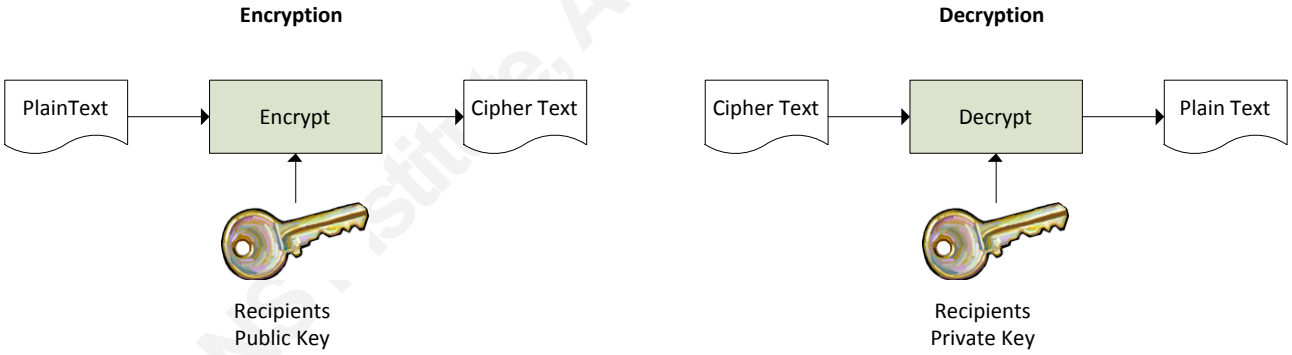


Figure 2.3 Encryption and decryption using public/private key cryptography

Before describing digital signatures, hashing needs to be understood. A hash algorithm takes variable length data and reduces it to a fixed length known as a message digest. This is done is such a way that even a small change in the original data causes large variation in the digest as shown in Figure 2.4 . In this way it is possible to validate if data has been altered by recalculating the hash and comparing it with the original hash value. Hashing algorithms include SHA1, SHA-256, SHA-384 and SHA-512.

Message	Algorithm	Message Digest or Hash
This is a test message	SHA1	be878249f2f30ad888b9bb632a5de67b967e4bc2
This was a test message	SHA1	21c121134344a2094cd2271395dc9d3ee82e87bc

Figure 2.4 Message Digests

© 2011 The SANS Institute Author retains full rights.

Now it is possible to confirm if a message has been altered we need one more step to ensure the file has not been altered and a new hash taken. The process used for this would be to take a digital signature. The following diagram in Figure 2.5 shows how a digital signature works.

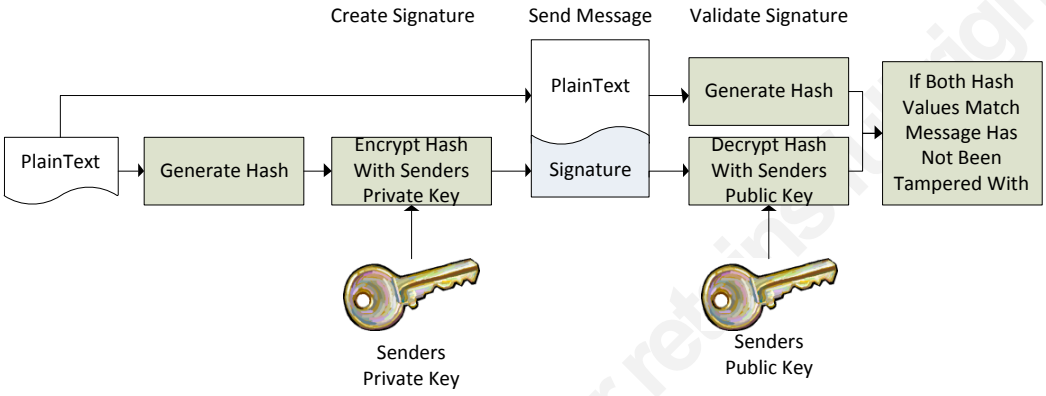


Figure 2.5 Digital signature generation and verification

More detailed information regarding Public/ Private Key cryptography can be found in Chapter 2 of "Understanding PKI: Concepts, Standards, and Deployment Considerations (2nd Edition)" (Adams, & Lloyd, 2003), for those who are interested in the mathematics behind cryptography "Handbook of Applied Cryptography" (Menezes, Van Oorschot, & Vanstone, 1997) covers this subject in greater depth.

2.4 Introduction to Certificates

A certificate, sometimes referred to as a digital certificate, is in essence a data structure containing a public key with additional metadata that defines what the public key can be used for, when it was issued and when is it valid until. This data is then signed by a trusted party to ensure no tampering can take place.

A simple breakdown of the makeup attributes is shown in Figure 2.6 below. Full details regarding certificate configuration can be found in rfc2459 (Housley, Ford, Polk, & Solo, 1999).

© 2011 The SANS Institute Author retains full rights.

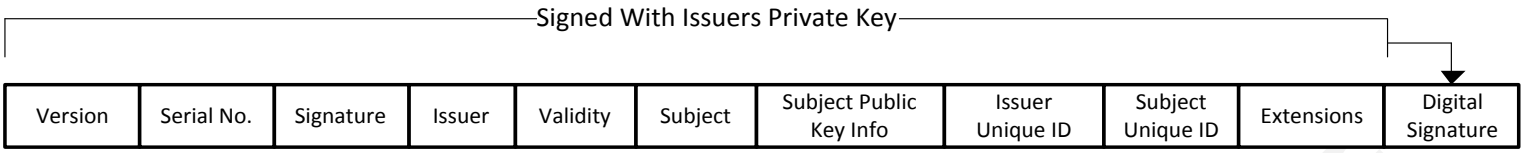


Figure 2.6 Certificate makeup

2.5 Introduction to Public Key Infrastructure

A Public Key Infrastructure (PKI) is a set of components that allow the creation, management, distribution, storage and revocation of public keys. An advantage of a PKI infrastructure is that all certificates issued can be trusted unless the certificate either appears on a revocation list; fails an online status check or if its validity period has expired.

Figures 2.7 and 2.8 below give an overview of certificate issuance and certificate validation. More detailed information about PKI can be found in "Understanding PKI: Concepts, Standards, and Deployment Considerations (2nd Edition)" (Adams, & Lloyd, 2003), Full technical details around components and operation of a PKI can be found within rfc2510 (Adams, & Farrell, 1999).

© 2011 SANS Institute, Author retains full rights.

© 2011 The SANS Institute Author retains full rights.

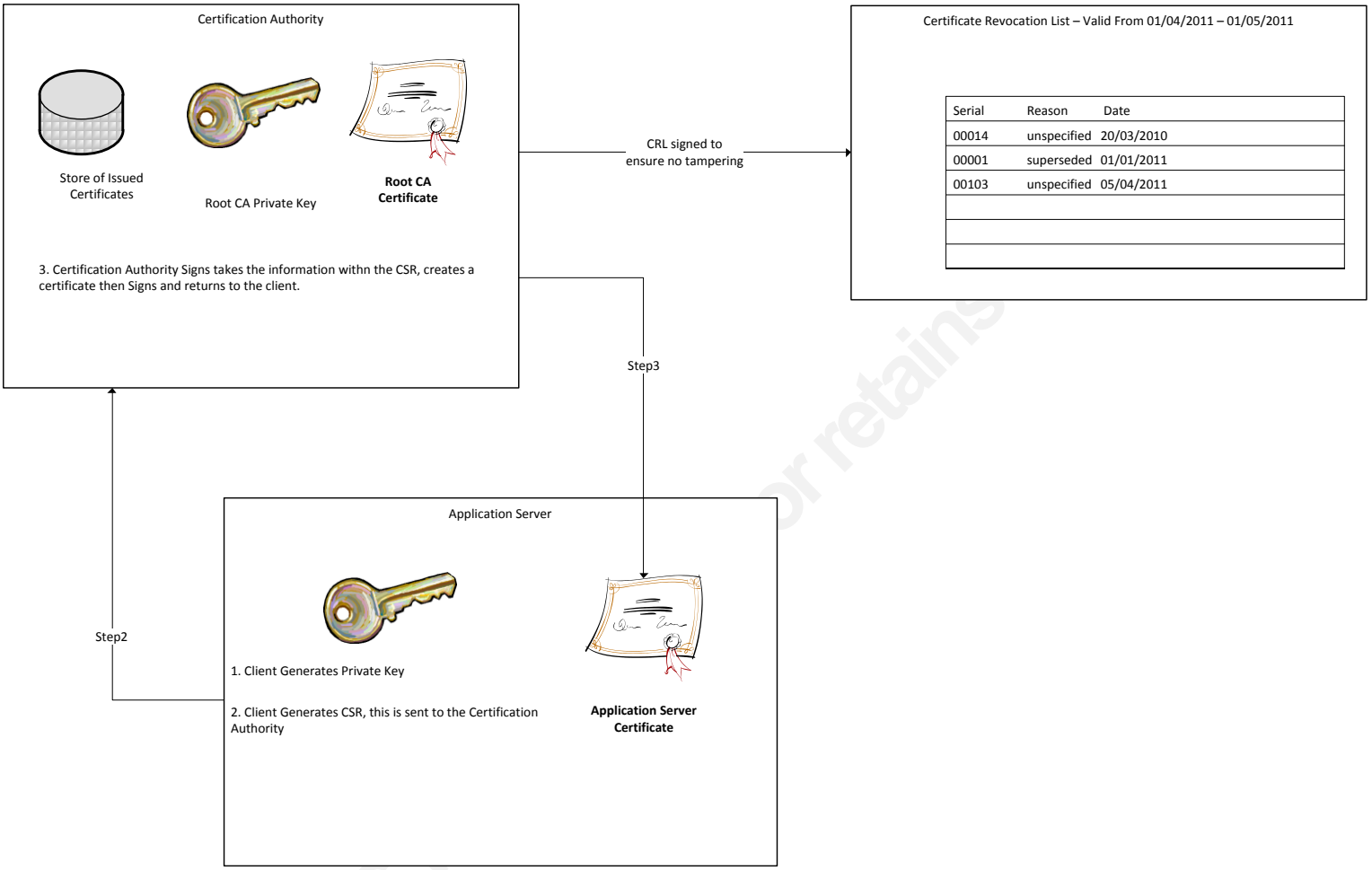


Figure 2.7 Certificate issuance

© 2011 The SANS Institute Author retains full rights.

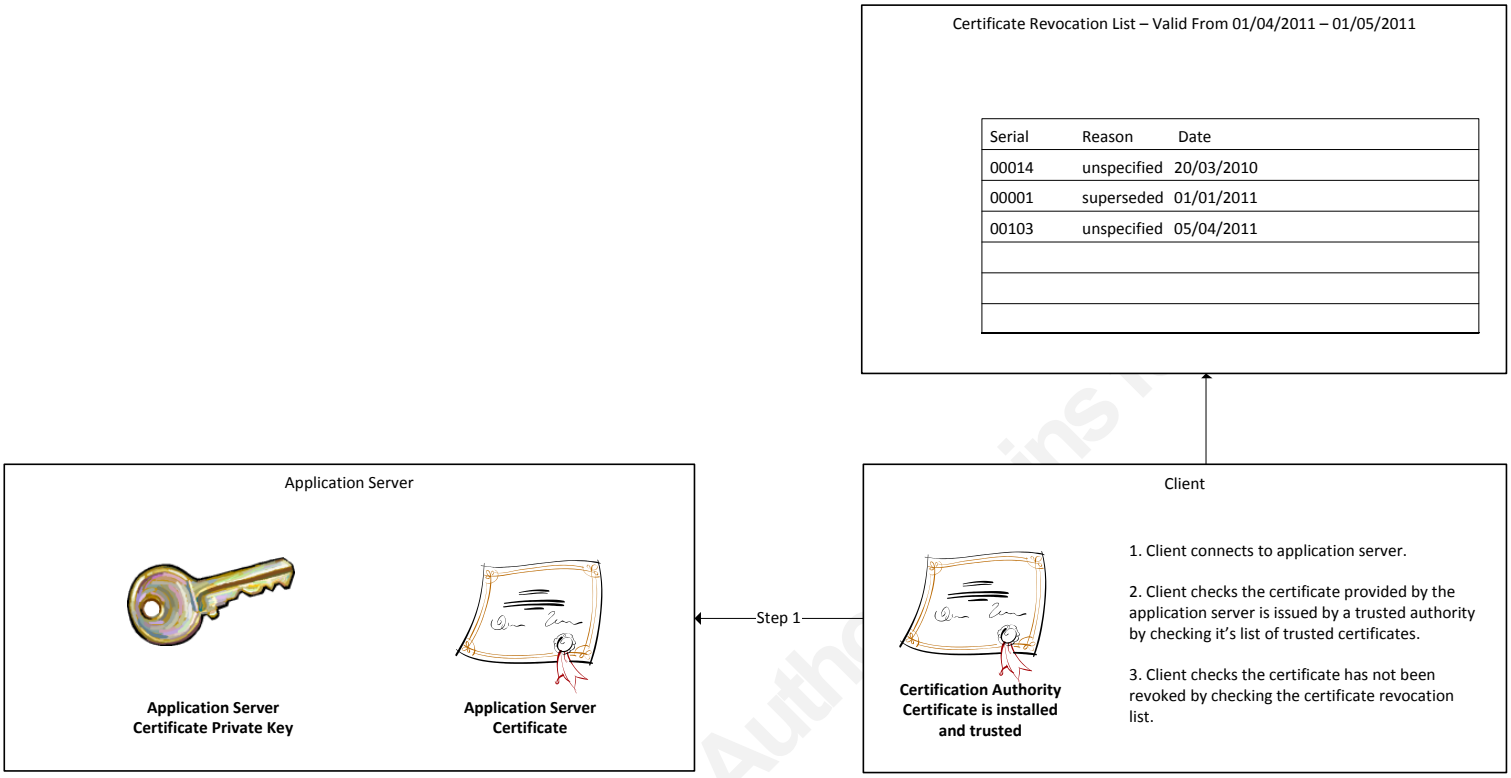


Figure 2.8 Certificate validation

3. Demonstrating Issues with LDAP

In this section we will cover an overview of the network infrastructure used within this paper, a demonstration of an authentication using LDAP showing its security risks, the generation and installation of a certificate to enable secure LDAP communication, a retest of the authentication once secure LDAP is enabled and finally configuration steps that allow LDAP to be audited and restricted.

3.1 Network Overview

The network used to demonstrate the concepts in this paper is shown below in Figure 3.1, this has a Microsoft Windows 2008 domain controller, the client machine is running Windows Server 2003 and the application server is running PGP Universal Version 3.0.

To generate the traffic for authentication the client machine has a copy of PGP Desktop version 10 which passes enrolment credentials to the application server; this in

© 2011 The SANS Institute Author retains full rights.

turn validates the user via LDAP calls between PGP Universal and the domain controller.

The user accounts and passwords used within this paper are as outlined in Figure 3.2.

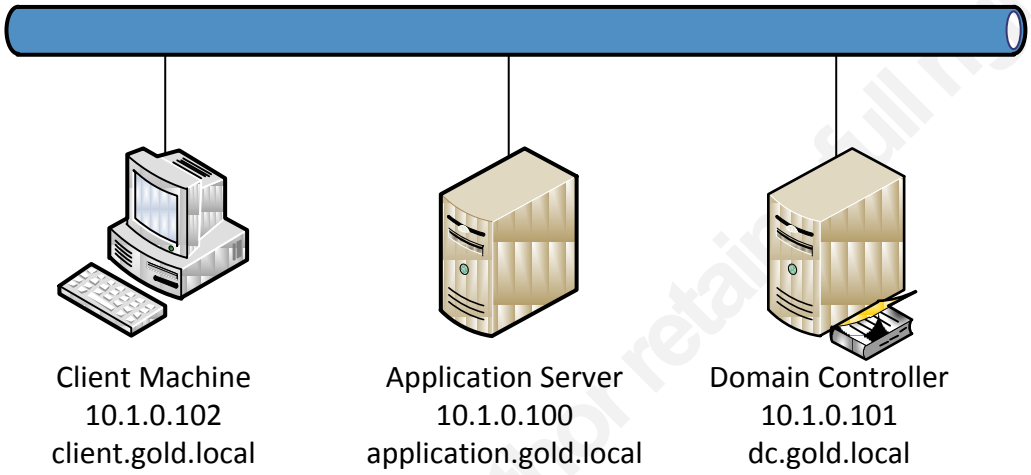


Figure 3.1 Network topology

User	Password	Function
ldap.bind	BindPassword1234	Allows the application server to authenticate to the domain controller
new.user	clientsPa55w0rd	The user account that will be authenticated by the application server

Figure 3.2 Credentials used for authentication

3.2 Authentication using LDAP

Given the setup shown in Figure 3.3 the authentication process is split into eight distinct stages. These stages are described below along with network traces for stages 2 and 3 to show the passing of authentication credentials.

© 2011 The SANS Institute Author retains full rights.

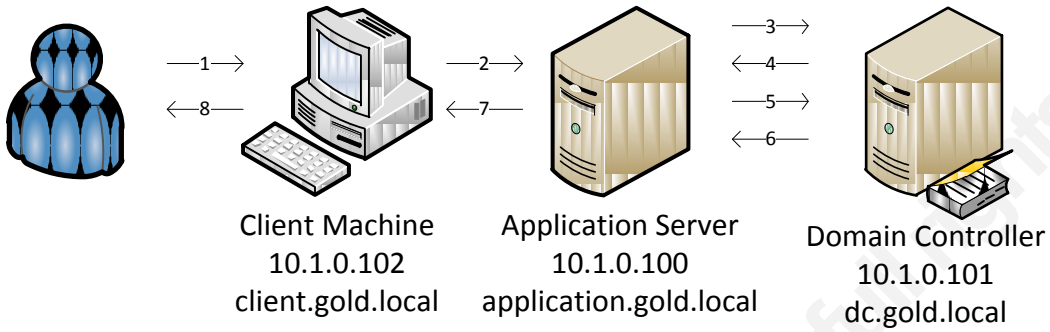


Figure 3.3 The authentication process

Stage 1 - User presents their credentials

As shown in Figure 3.4 the end user is prompted for their user credentials, these are then used by the application in the authentication process.

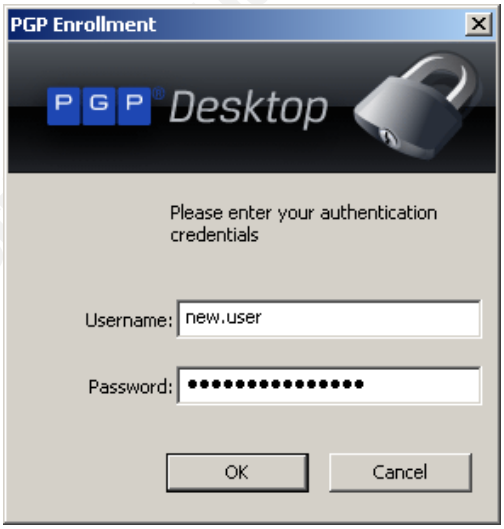


Figure 3.4 User enrolment screen

Stage 2 - Credentials are passed to the application server

Once the user credentials have been input they are passed to the application server, as you can see from the network trace below the application server uses TLS, which protects the user credentials when being sent over the network.

© 2011 The SANS Institute Author retains full rights.

The packet capture from the client in Figure 3.5 shows the initiation of secure http communication between the client workstation and the web server.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.1.0.102	10.1.0.100	TCP	62	murray > https [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
2	0.000711	10.1.0.100	10.1.0.102	TCP	62	https > murray [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM=1
3	0.000735	10.1.0.102	10.1.0.100	TCP	54	murray > https [ACK] Seq=1 Ack=1 Win=64240 Len=0
4	0.006109	10.1.0.102	10.1.0.100	TLSv1	130	Client Hello
5	0.006522	10.1.0.100	10.1.0.102	TCP	60	https > murray [ACK] Seq=1 Ack=77 Win=5840 Len=0
6	0.008278	10.1.0.100	10.1.0.102	TLSv1	709	Server Hello, Certificate, Server Hello Done
7	0.016412	10.1.0.102	10.1.0.100	TLSv1	193	Client Key Exchange
8	0.056607	10.1.0.100	10.1.0.102	TCP	60	https > murray [ACK] Seq=656 Ack=216 Win=6432 Len=0
9	0.056646	10.1.0.102	10.1.0.100	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
10	0.057032	10.1.0.100	10.1.0.102	TCP	60	https > murray [ACK] Seq=656 Ack=275 Win=6432 Len=0
11	0.057361	10.1.0.100	10.1.0.102	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
12	0.215504	10.1.0.102	10.1.0.100	TCP	54	murray > https [ACK] Seq=275 Ack=715 Win=63526 Len=0
13	2.215261	10.1.0.102	10.1.0.100	TLSv1	1115	Application Data
14	2.255706	10.1.0.100	10.1.0.102	TCP	60	https > murray [ACK] Seq=715 Ack=1336 Win=8488 Len=0
15	2.264236	10.1.0.100	10.1.0.102	TLSv1	992	Application Data, Application Data
16	2.265330	10.1.0.102	10.1.0.100	TLSv1	779	Application Data

Figure 3.5 – Packet capture of client to application server traffic

As HTTP is a TCP protocol the connection begins with a TCP three way handshake to establish the connection and guaranteed delivery of messages, these are shown in packets 1 to 3 of Figure 3.5.

While TCP provides guaranteed delivery of messages on its own it does not provide confidentiality. For this we need an additional protocol in the form of SSL or TLS. All versions of SSL and TLS share the same basic approach; they provide a secure channel between two communicating programs which arbitrary data can be sent (Rescorla, 2001). The agreement of ciphers and exchange of keys is completed in packets 4 to 12 shown below in Figure 3.6.

4	0.006109	10.1.0.102	10.1.0.100	TLSv1	130	Client Hello
5	0.006522	10.1.0.100	10.1.0.102	TCP	60	https > murray [ACK] Seq=1 Ack=77 Win=5840 Len=0
6	0.008278	10.1.0.100	10.1.0.102	TLSv1	709	Server Hello, Certificate, Server Hello Done
7	0.016412	10.1.0.102	10.1.0.100	TLSv1	193	Client Key Exchange
8	0.056607	10.1.0.100	10.1.0.102	TCP	60	https > murray [ACK] Seq=656 Ack=216 Win=6432 Len=0
9	0.056646	10.1.0.102	10.1.0.100	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
10	0.057032	10.1.0.100	10.1.0.102	TCP	60	https > murray [ACK] Seq=656 Ack=275 Win=6432 Len=0
11	0.057361	10.1.0.100	10.1.0.102	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
12	0.215504	10.1.0.102	10.1.0.100	TCP	54	murray > https [ACK] Seq=275 Ack=715 Win=63526 Len=0

Figure 3.6 Cipher agreement and key exchange

Once encryption ciphers are agreed on and keys exchanged communication between the server and client is sent securely. The packet in Figure 3.7 below shows the encrypted application data. Even if the data is intercepted it cannot be read.

© 2011 The SANS Institute Author retains full rights.

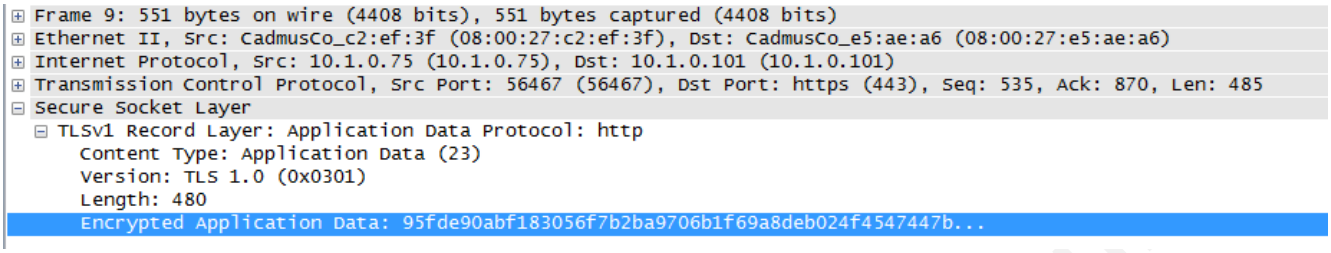


Figure 3.7 Encrypted application traffic

Stage 3 - Application server binds to the Directory Service.

Before the user can be authenticated the application binds to the directory service to locate the users DN within the directory. The connection is over unsecured LDAP so the full details of the bind account can be seen traversing the network in Packet 4 of Figure 3.8, this packet has been expanded in Figure 3.9 to display the text value of the password used to search for the user within the directory.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	10.1.0.100	10.1.0.101	TCP	48764 > ldap [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 WS=7
2	0.000206	10.1.0.101	10.1.0.100	TCP	ldap > 48764 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=8 SACK_PERM=1
3	0.000463	10.1.0.100	10.1.0.101	TCP	48764 > ldap [ACK] Seq=1 Ack=1 Win=5888 Len=0
4	0.000716	10.1.0.100	10.1.0.101	LDAP	bindRequest(1) "cn=ldap.bind,cn=users,dc=gold,dc=local" simple
5	0.003453	10.1.0.101	10.1.0.100	LDAP	bindResponse(1) success
6	0.003661	10.1.0.100	10.1.0.101	TCP	48764 > ldap [ACK] Seq=69 Ack=23 Win=5888 Len=0

Figure 3.8 Bind account details

© 2011 The SANS Institute Author retains full rights.

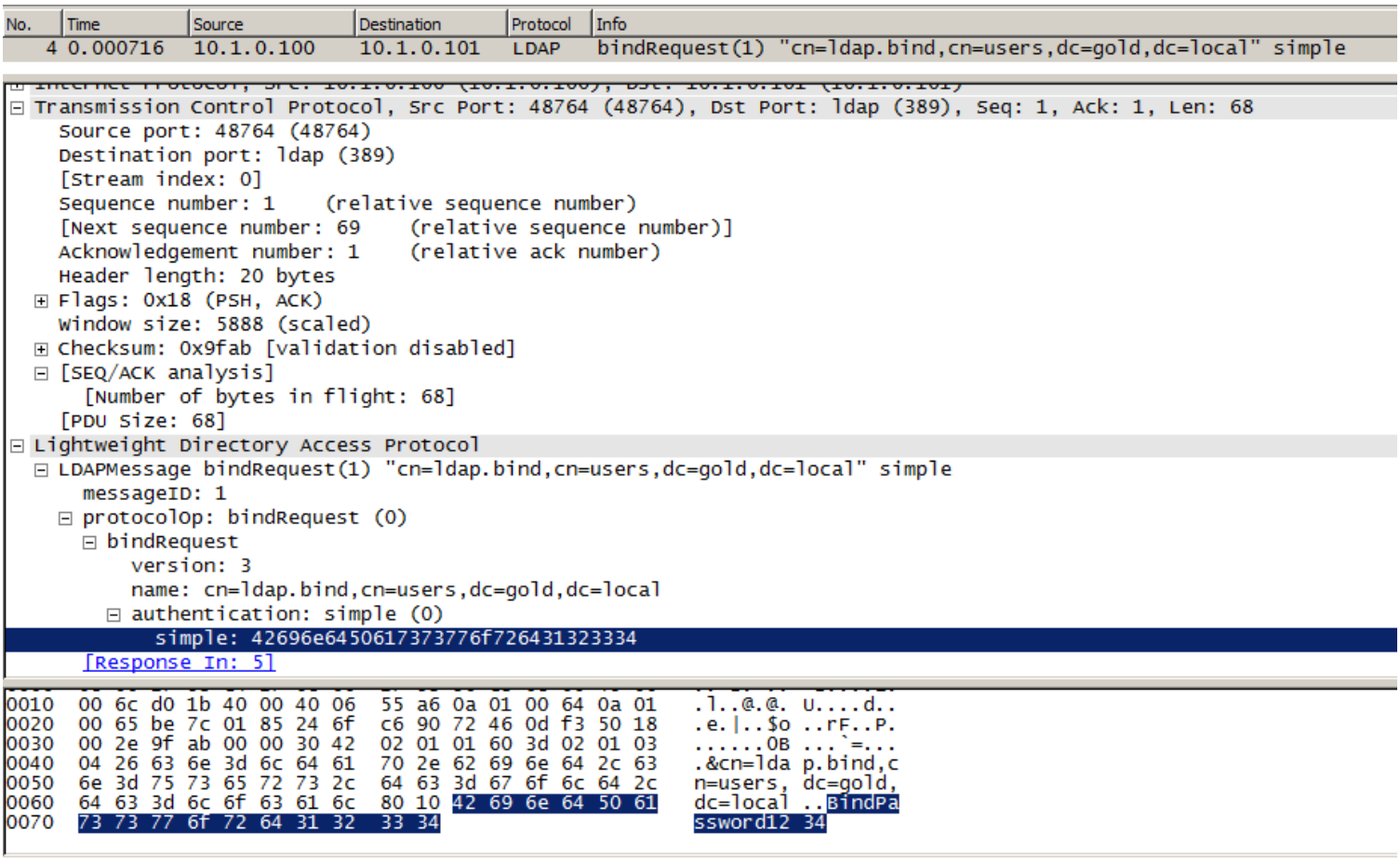


Figure 3.9 Bind account password

Stage 4 - Locate Users Account within the Directory

The application will then proceed to find the users complete distinguished name (DN) within the directory using the credentials entered in stage 1.

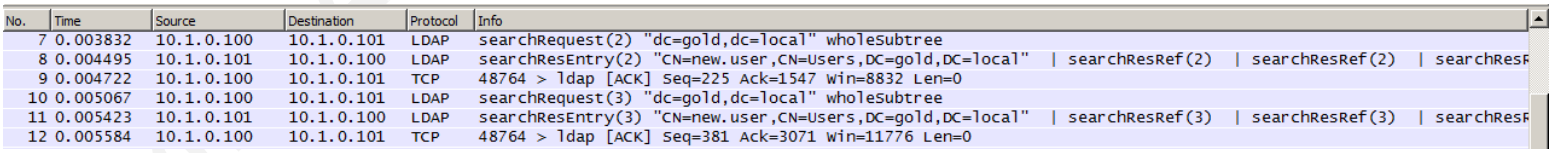


Figure 3.10 Locating the user to be authenticated

Packet 7 in Figure 3.10 shows the search which is sent to locate the full DN of the user, the base object is set to the top of the domain 'dc=gold, dc=local' and the scope setting of 'wholeSubtree' ensures everything below the baseObject will be searched while the filter shown in Figure 3.11 will find any account where userPrincipalName, sAMAccountName, mail or proxyAddresses is set to new.user.

Andrew Reid;andy@andyreid.co.uk

© 2011 The SANS Institute Author retains full rights.

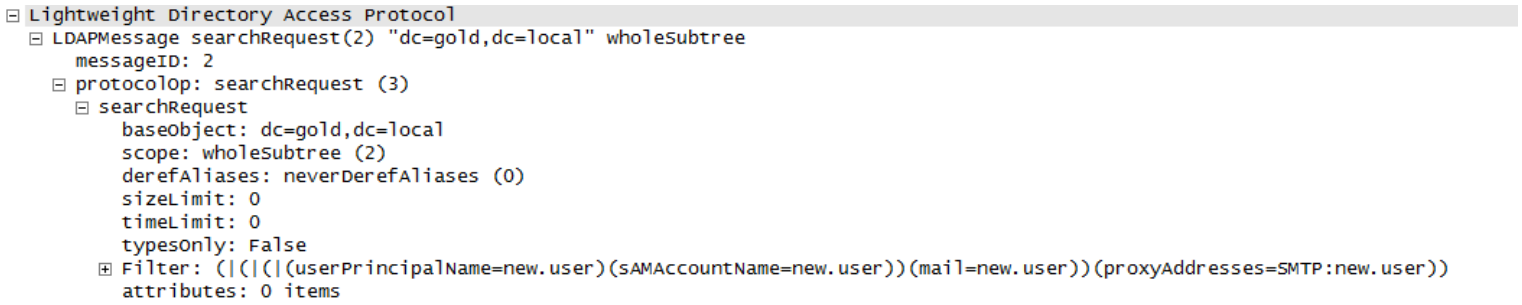


Figure 3.11 LDAP search filter

Figure 3.12 shows the result of the query returned in packet 8 and includes the DN of the user and also its assigned attributes.

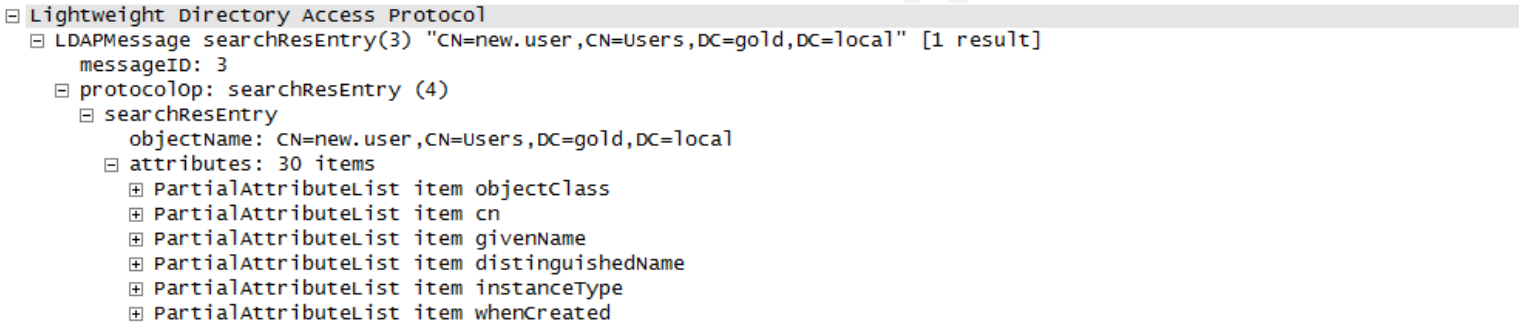
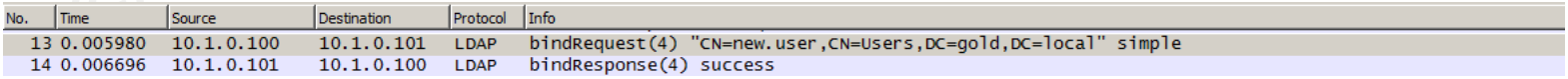


Figure 3.12 Response to search query

Stage 5 - Authenticate by binding

The application then sends a bind request to the directory service using the DN of the user as found in stage 4 and the password that was supplied in stage 1. Again as LDAP is used these credentials are passed in the clear this is displayed in packet 13 and shown in Figure 3.13 and Figure 3.14.



No.	Time	Source	Destination	Protocol	Info
13	0.005980	10.1.0.100	10.1.0.101	LDAP	bindRequest(4) "CN=new.user,CN=Users,DC=gold,DC=local" simple
14	0.006696	10.1.0.101	10.1.0.100	LDAP	bindResponse(4) success

Figure 3.13 Bind request and response

© 2011 The SANS Institute Author retains full rights.

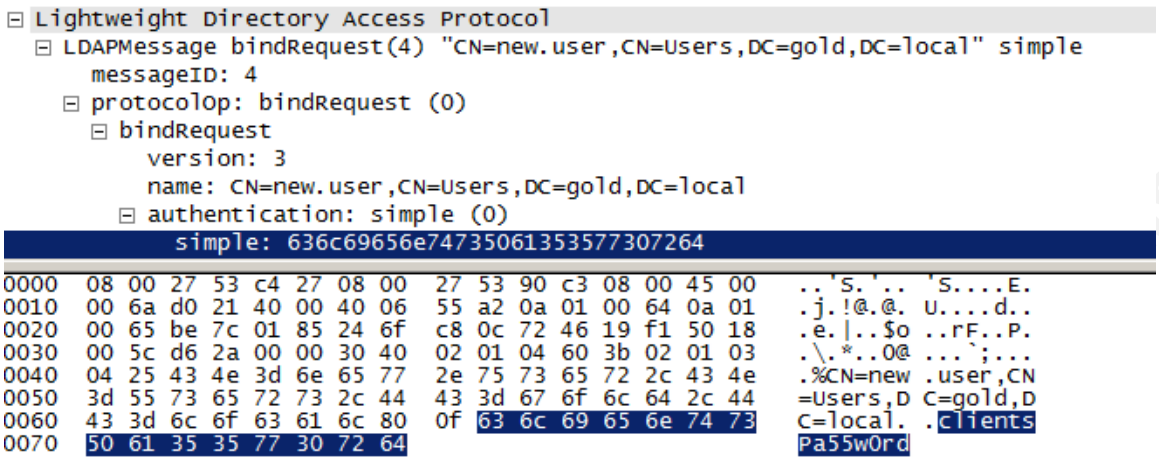


Figure 3.14 User account credentials

Stage 6 - Domain controller respond to application server

Packet 14 of Figure 3.13 is the response from the directory service showing the authentication was successful. At this point the application server knows if the user has been successfully authenticated.

Stage 7 - Application server send output to client application

The application will then process the success or failure message from the domain controller and pass this on to the client application. This traffic is also encrypted to keep it secure.

Stage 8 Application responds to user

The final step in the authentication process is to inform the end user; in this case once the user is enrolled the main application screen is shown displaying the user's keys as in Figure 3.15.

© 2011 The SANS Institute Author retains full rights.

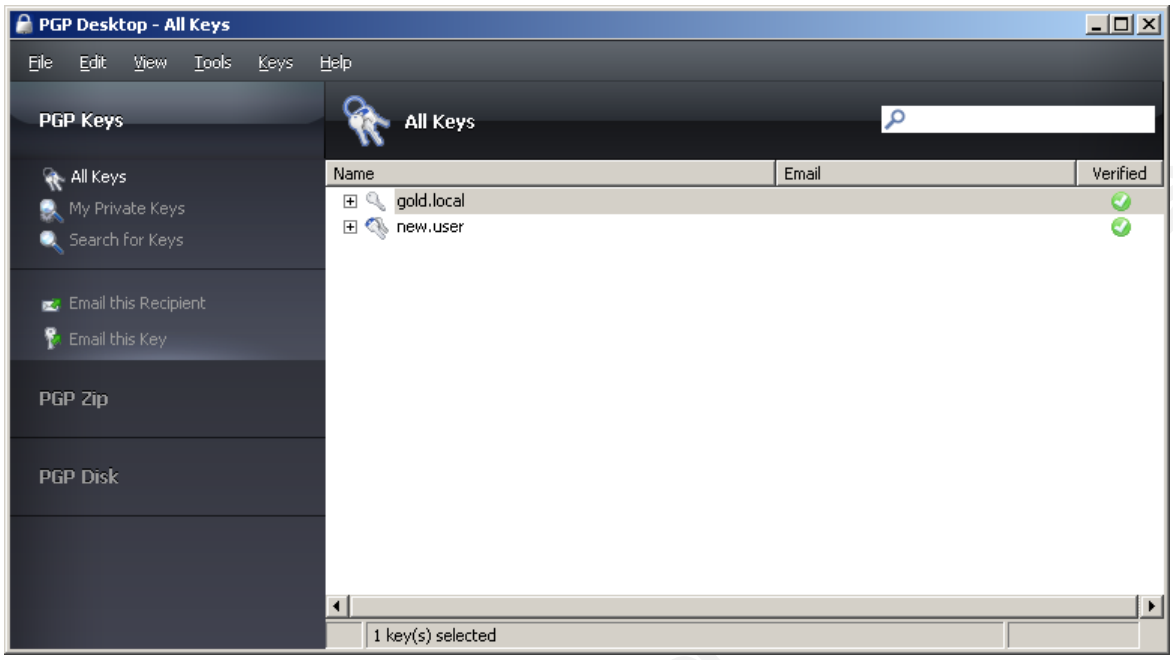


Figure 3.15 PGP Desktop showing user keys

In the event of an authentication failure a "Your credentials were not accepted." error message is displayed and prompts the user to try again as shown in Figure 3.16.

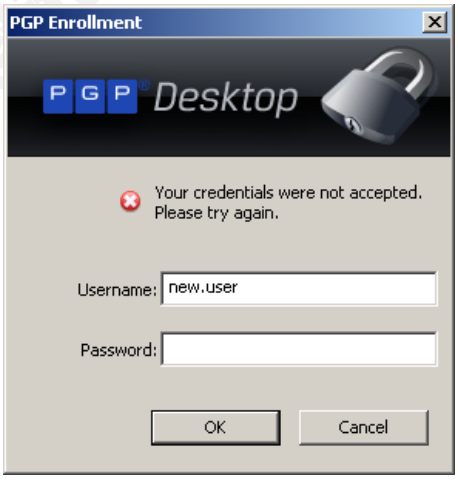


Figure 3.16 User enrolment failure

As can be seen from the network traces outlined above, the use of LDAP alone for authentication results in credentials transferred between the application server and the directory server being unencrypted. If intercepted this could result the compromise of

© 2011 The SANS Institute Author retains full rights.

these accounts.

4. Configuring Secure LDAP

The use secure LDAP on a domain controller requires a certificate to be generated and installed on each domain controller that will be used for authentication. This can either be generated as a self-signed certificate or a certificate signing request can be generated and passed to a third party for signing.

Microsoft Certificate Services can also be used to generate, sign and install certificates on domain controllers, but this introduces the complexity of running a full PKI solution which is beyond the scope of this paper.

The following steps detail the generation of a public/private key pair. The steps also show the subsequent signing of the public key by a certificate authority or via self-signing.

4.1 Generating the signing request

To generate the key pair and the certificate signing request (CSR) the certreq command and request.inf file from Appendix A shown below is used with the subject line updated to contain the name of the domain controller and the relevant organisational information shown in Figure 4.1.

```
Subject =
"E=andy@andyreid.co.uk,CN=dc.gold.local,O=andyreid.co.uk,OU=Servers,L=Crewe,S=Cheshire,C=GB"
; replace with the FQDN of the DC
```

Figure 4.1 Updated certificate subject information

When requesting a certificate from a third party certification authority additional fields may be required. These have been included in the Figure 4.1 above and are decoded as in Figure 4.2 below.

Field	Value
E	Email Address
CN	Common name of the certificate
O	Organisation
L	Locality Name (e.g. City)

© 2011 The SANS Institute Author retains full rights.

ST	State, or Province
C	Two letter ISO county code
OU	Organisational Unit

Figure 4.2 List of valid fields for certreq

Once saved the CERTREQ command can be executed to generate the public/private key pair and certificate signing request, which is detailed in Figure 4.3 below. The resulting CSR is shown in Figure 4.4.

```
CERTREQ -new request.inf request.csr
```

Figure 4.3 Public/Private key and CSR generation

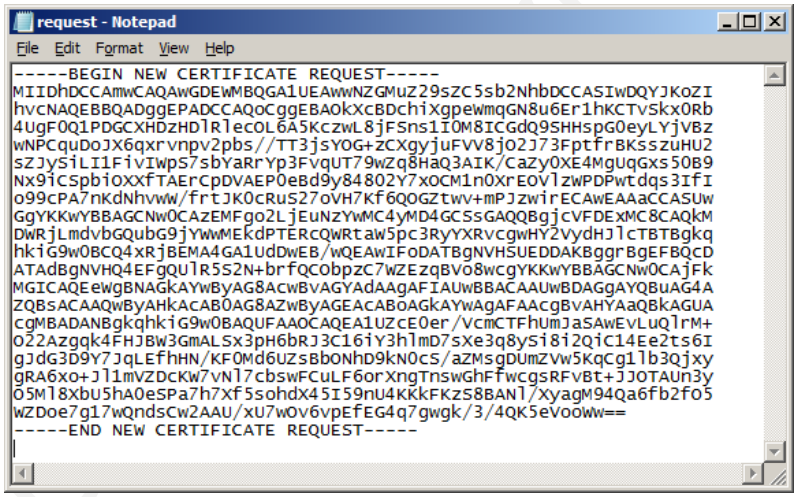


Figure 4.4 Certificate signing request

4.2 Creating a third party signed certificate

The CSR generated in 4.4 can be sent to a third party to sign. Once the certificate is returned it is combined with the private key in the certificate store using the command detailed in Figure 4.5, where certificate.cer is the name of the certificate.

```
CERTREQ -accept certificate.cer
```

Figure 4.5 Certreq command to combine signed certificate and private key

4.3 Creating a self-signed certificate

If the certificate is to be used on a development or test system a self-signed

© 2011 The SANS Institute Author retains full rights.

certificate can be used. This type of certificate is quick to generate and install but lacks some of the controls of using a third party certification authority. Certificate authorities provide certificate revocation lists and increase the likelihood that the root and intermediary certificates are already published to machines within the environment.

To generate the certificate the makecert command is used. This comes as part of the Windows Software Development Kit (SDK) which is an additional download and can be found at the following URL http://www.microsoft.com/downloads/en/details.aspx?FamilyId=F26B1AA4-741A-433A-9BE5-FA919850BDBF&displaylang=en

Once the SDK is installed the certificate can be generated with the command detailed in Figure 4.6. This should be entered as one continuous command. Details of the various options are given Figure 4.7 below.

```
makecert -a sha1 -eku 1.3.6.1.5.5.7.3.1 -sky exchange -sr localmachine -ss My -pe -r -n
"CN=dc.gold.local" -len 2048 -m 12 CERT.CER
```

Figure 4.6 Generation of certificate using Makecert

Option	Description
-a	Signature algorithm, sha1 is used
-eku	Enhanced key usage attribute, in this case 1.3.6.1.5.5.7.3.1 – Server Authentication
-sky	Subjects Key Type - Exchange
-sr	Certificate store location (either local machine or current user)
-ss	Subject Store, My (Personal Folder)
-pe	Mark the private key as exportable
-r	Creates a self-signed certificate
-n	Name of the certificate
-len	Key length
-m	Duration of the certificate in months
CERT.CER	Output file for the public certificate

Figure 4.7 Details of options used to generate certificate with Makecert

© 2011 The SANS Institute Author retains full rights.

4.4 Installing the certificate

In both examples shown above the certificate will be installed into the local machines store, this is sufficient for the active directory service to pick up the certificate. With Windows Server 2008 Microsoft recommends using the NT Directory Service (NTDS) Personal Store. This can be completed by exporting the certificate from the machine's store and importing it into the NTDS store as shown in the steps below.

Step 1 – **Exporting the certificate from the machine's store**

Launch the Microsoft Management Console (MMC), (adding the certificate snap in under the context of the computer store) then expand the Personal store then right click the certificate to be exported then select 'All Tasks' then 'Export' as shown in Figure 4.8. Ensure when the certificate is exported the private key is also exported with the certificate (when exporting you will be prompted for a password to protect the private key). This will be saved as a Personal Information Exchange (PFX) file which and be imported in the next step.

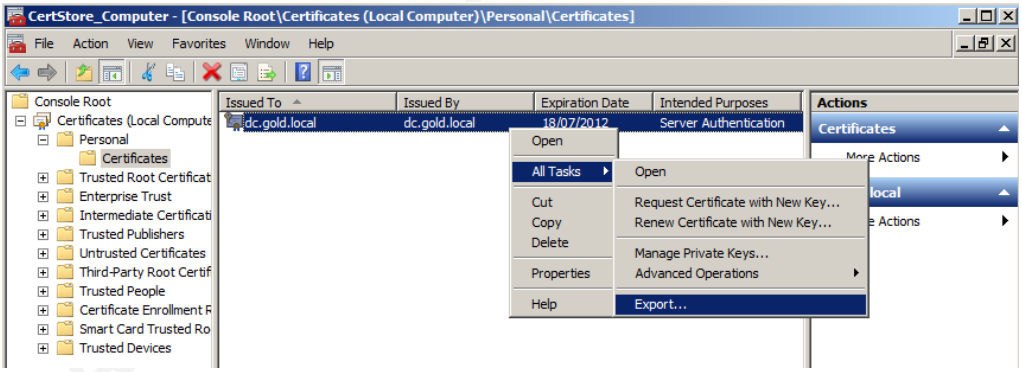


Figure 4.8 Exporting a certificate from the local machine's store

Step 2 – Importing the certificate into the NTDS store

To import the certificate into the NTDS Personal Store from the MMC add the certificate snap in but this time select manage certificates for service accounts. When prompted for the service account to manage, select 'Active Directory Domain Services'. Right click the 'NTDS\Personal' folder and select 'All Tasks' then 'Import' as is shown in Figure 4.9.

© 2011 The SANS Institute Author retains full rights.

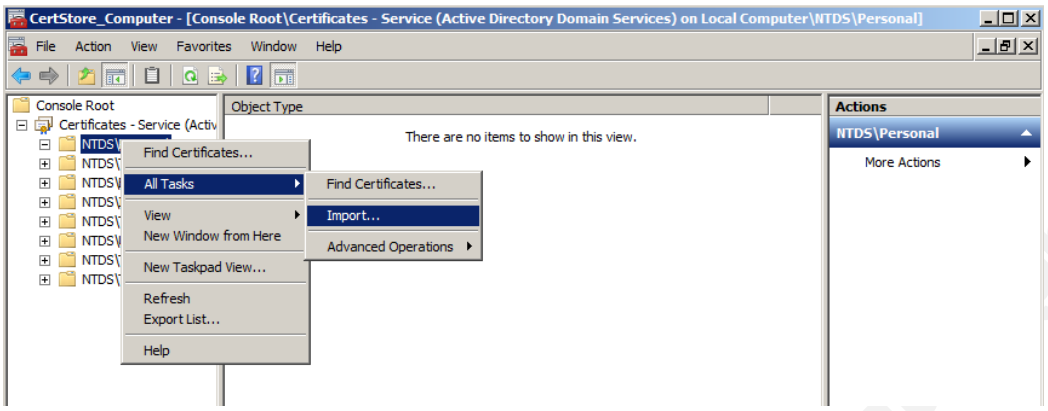


Figure 4.9 Importing a certificate to the NTDS personal store

Set the File type to Personal Information Exchange (.pfx / .p12) then browse to the previously exported PFX file and select 'Import'. When prompted to enter the password for the private key there is a check box 'Mark this key as exportable'. if you need to export the certificate later this check box will need to be selected. Not doing so will mean the private key associated with the certificate cannot be moved from this server.

4.5 Installing the certificate hierarchy

To allow the server to use the certificate it must be trusted by the local machine, this is achieved by the signing certificates being installed into the relevant trusted certificate store. This can be tested by double clicking the certificate from within the certificate management console. Figure 4.10 below shows a self-signed certificate that is not trusted by the server.

© 2011 The SANS Institute Author retains full rights.

Figure 4.10 An untrusted certificate

To allow the server to trust this certificate the public certificate needs to be installed into the 'Trusted Root Certificate store' on the local computer. To do this launch the MMC and add the Certificate snap in selecting manage the Local Computer store. Expand the 'Trusted Root Certification Authorities' folder then right click Certificates then select 'All Tasks' then 'Import' this is shown in Figure 4.11.

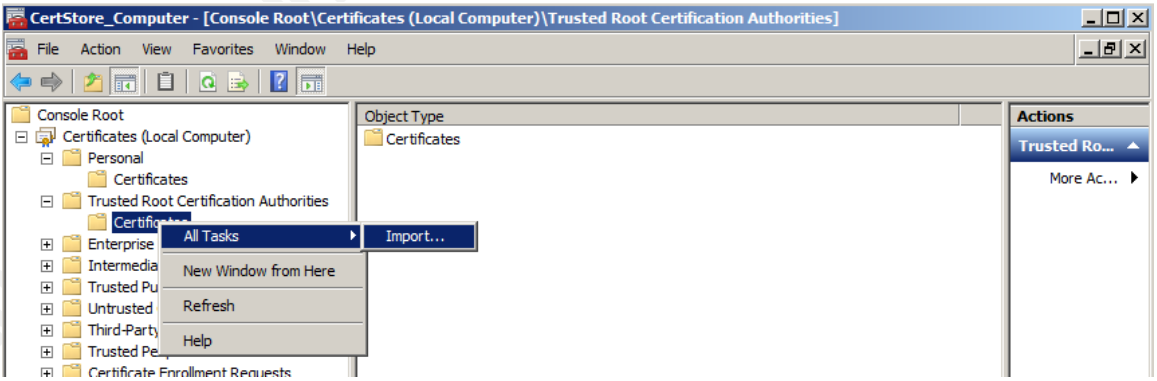


Figure 4.11 Importing a Trusted Root Certificate

Browse to the location of the certificate (CER) file that was either generated when Makecert was run, or supplied by the third party certification authority and import this into the 'Trusted Root Certification Authorities' store. If Intermediary certificates have

© 2011 The SANS Institute Author retains full rights.

also been supplied store these in the 'Intermediate Certification Authority' stores using the same process.

5. Testing the Secure LDAP Connection

To test the certificate has been loaded and the secure LDAP connection is functioning as required, launch LDP.EXE. This can be found in c:\windows\system32 on Windows 2008 systems or installed as part of the Windows 2003 Support Tools.

When LDP has launched select Connection then Connect then change the Server name to the domain controller Fully Qualified Domain Name (FQDN) then update the port to 636 and finally tick the SSL check box, this is shown in Figure 5.1.

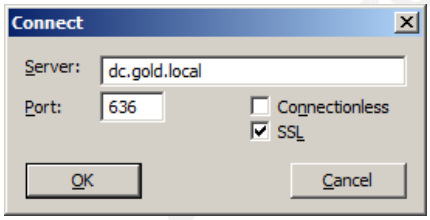


Figure 5.1 Configuring LDP to use secure LDAP

Select OK to initiate the connection. If everything has been successful a screen similar to the one shown in Figure 5.2 below will be displayed, otherwise check the installation of each of the certificates, ensuring the correct certificates are installed into the trust stores and the domain controller certificate includes its private key.

© 2011 The SANS Institute Author retains full rights.

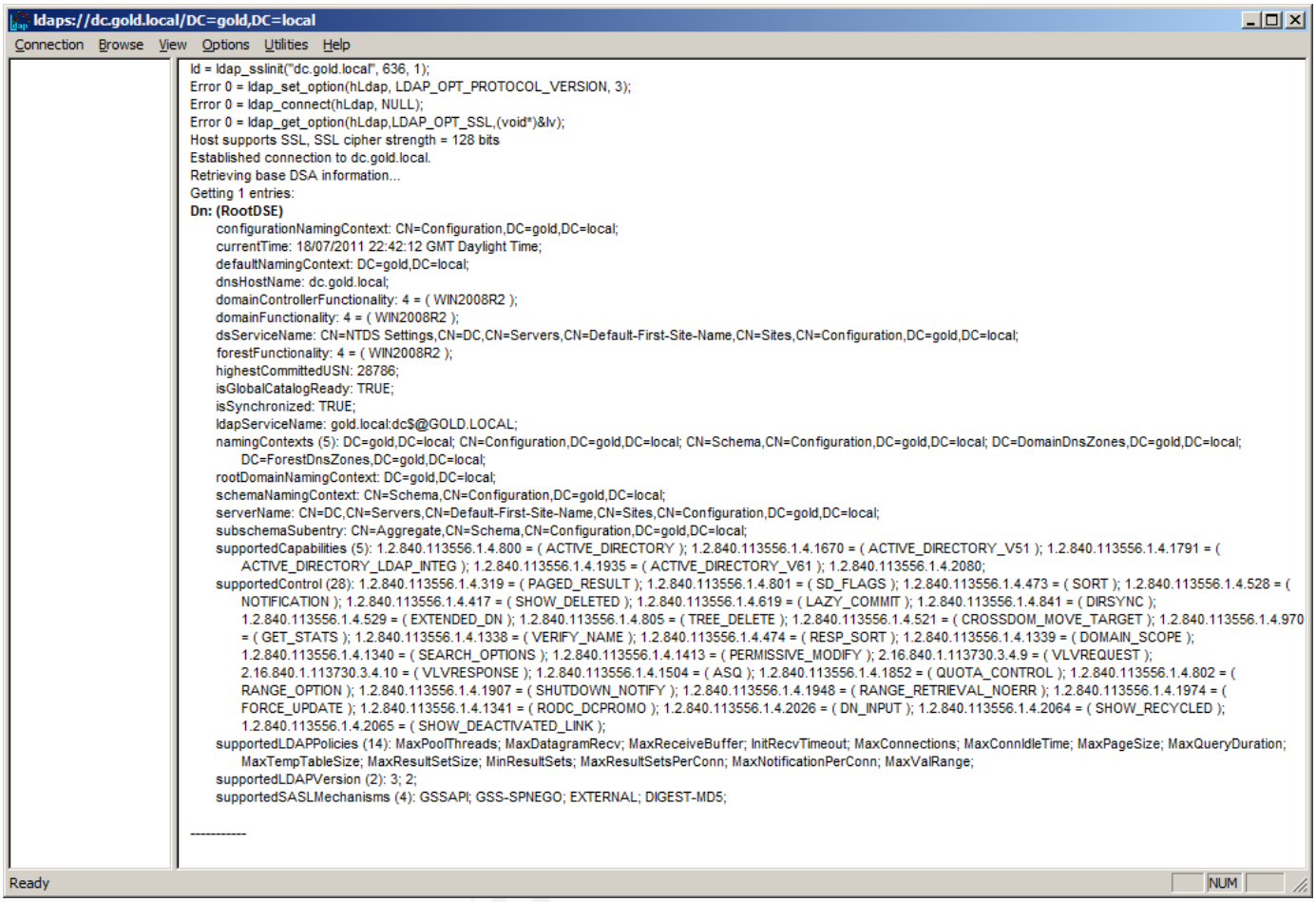


Figure 5.2 LDP connecting using Secure LDAP

5.1 Authentication Using Secure LDAP

Once the certificate has been successfully tested and the application server reconfigured to use secure LDAP on port 636, the authentication process can be retested.

To test the configuration of secure LDAP has been successful the steps outlined in section 4 of this document have been re-run and network traces for stages 3 and 5 as detailed in Figure 5.3 below are shown below to confirm credentials are no longer passed unencrypted.

© 2011 The SANS Institute Author retains full rights.

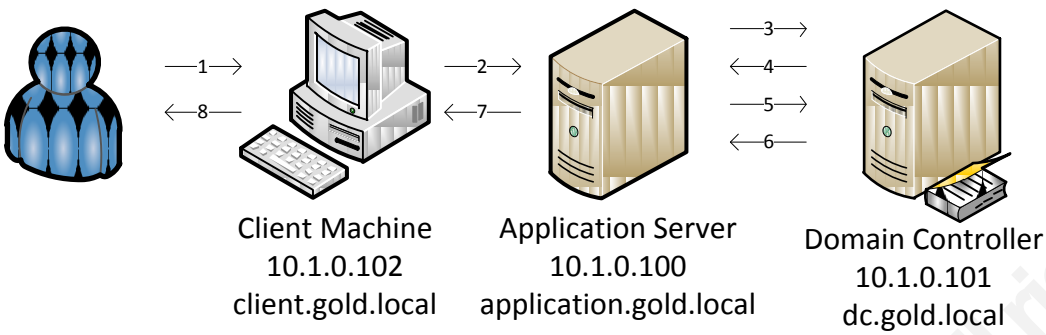


Figure 5.3 The authentication process

Stage 3 - Application server binds to the Directory Service.

The network trace in Figure 5.4 below shows secure LDAP is now being used rather than LDAP and a similar process of 'Client Hello' and 'Server Hello'. Then exchange messages take place before application data is transmitted.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	10.1.0.100	10.1.0.101	TCP	51774 > ldaps [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 WS=7
2	0.000367	10.1.0.101	10.1.0.100	TCP	ldaps > 51774 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=8 SACK_PERM=1
3	0.000690	10.1.0.100	10.1.0.101	TCP	51774 > ldaps [ACK] Seq=1 Ack=1 Win=5888 Len=0
4	0.005809	10.1.0.100	10.1.0.101	SSLv2	Client Hello
5	0.006343	10.1.0.101	10.1.0.100	TLSv1	Server Hello, Certificate, Certificate Request, Server Hello Done
6	0.006853	10.1.0.100	10.1.0.101	TCP	51774 > ldaps [ACK] Seq=122 Ack=1382 Win=8832 Len=0
7	0.009488	10.1.0.100	10.1.0.101	TLSv1	Certificate, Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
8	0.015309	10.1.0.101	10.1.0.100	TLSv1	Change Cipher Spec, Encrypted Handshake Message
9	0.016114	10.1.0.100	10.1.0.101	TLSv1	Application Data, Application Data
10	0.086793	10.1.0.101	10.1.0.100	TLSv1	Application Data

Figure 5.4 Network trace showing negotiation and secure data transfer

Looking at an expanded view Packet 5 in Figure 5.5 where the Domain Controller certificate is sent to the application server, it can be seen this is the same as the certificate that was installed onto the Domain Controller shown in Figure 5.6.

© 2011 The SANS Institute Author retains full rights.

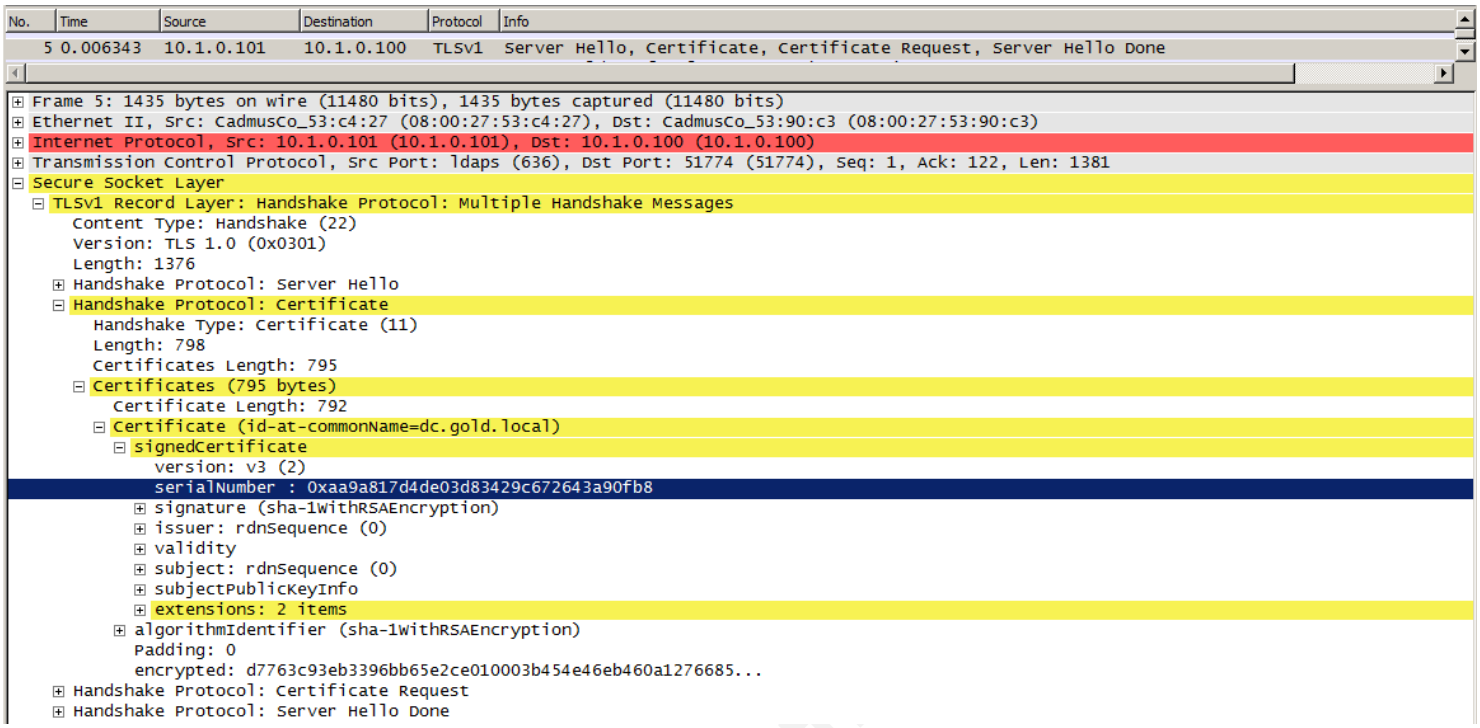


Figure 5.5 Domain Controller sends its certificate to the application server

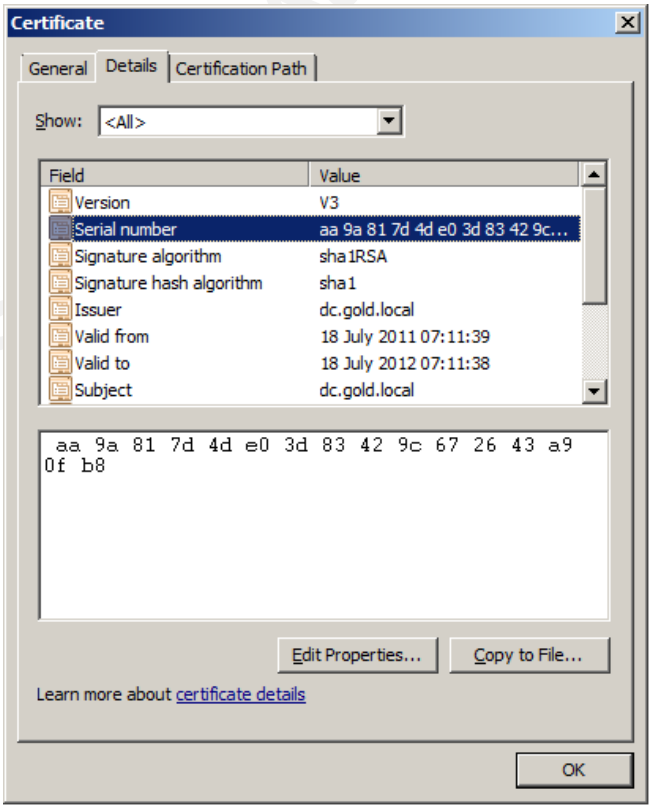


Figure 5.6 Certficate installed on the Domain Controller

Andrew Reid;andy@andyreid.co.uk

© 2011 The SANS Institute Author retains full rights.

Packet 9 shown in Figure 5.7 below shows encrypted traffic is being send between the application server and the Domain Controller.

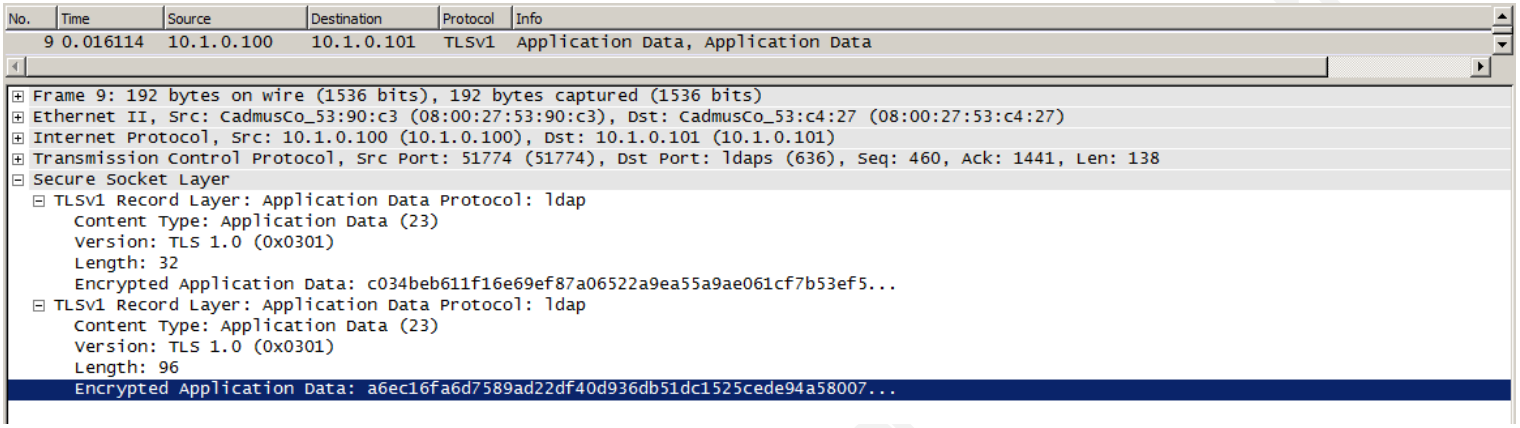


Figure 5.7 Encryptred traffic between application server and Domain Controller

Stage 5 - Authenticate by binding

As seen from the earlier network LDAP network traces the 'authenticate by binding' step was the second connection between the application server and the domain controller. This is shown from packets 29 onwards in Figure 5.8 below with encrypted application data being in packet 37 shown in Figure 5.9 below.

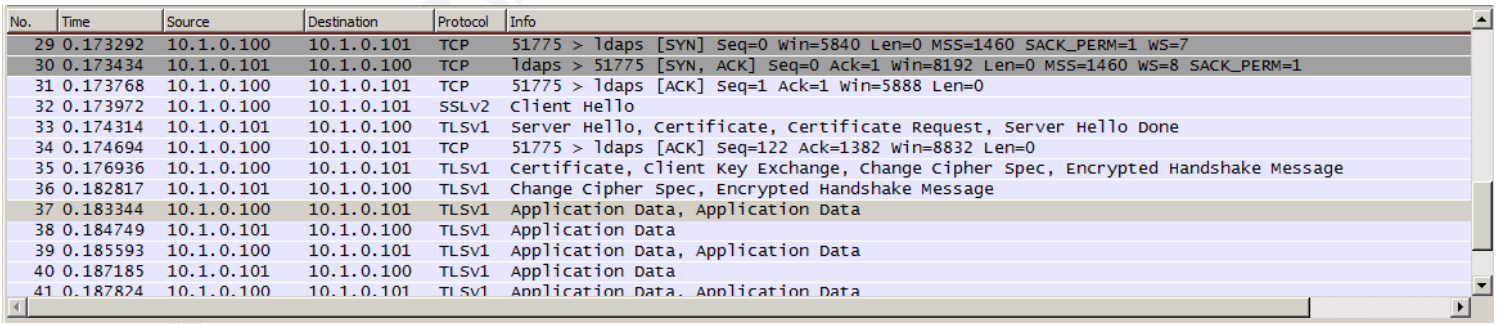


Figure 5.8 Authenticate by binding with user credentials

© 2011 The SANS Institute Author retains full rights.

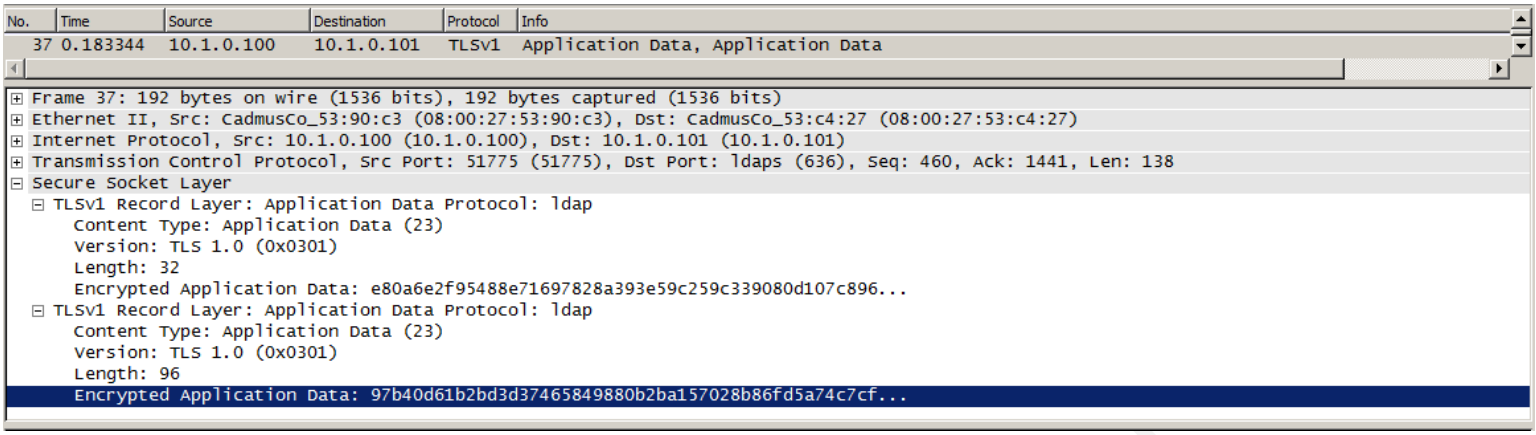


Figure 5.9 Encryption is now being used for the transfer of data

6. Enable enhanced LDAP Auditing and Restriction

Within Windows server 2008 options exist to audit insecure LDAP connections and also to disable Simple Authentication and Security Layer (SASL) LDAP binds that do not request signing (integrity verification) and LDAP simple binds that are performed on a clear text (non-SSL/TLS-encrypted) connection.

To help identify insecure client connections a summary event ID 2887 is logged every 24 hours. Additional logging can be configured by amending the Active Directory Diagnostic Logging. This causes an Event ID 2889 to be logged whenever an insecure connection is detected and provides details of the calling IP address and user ID as shown in Figure 6.1 below.

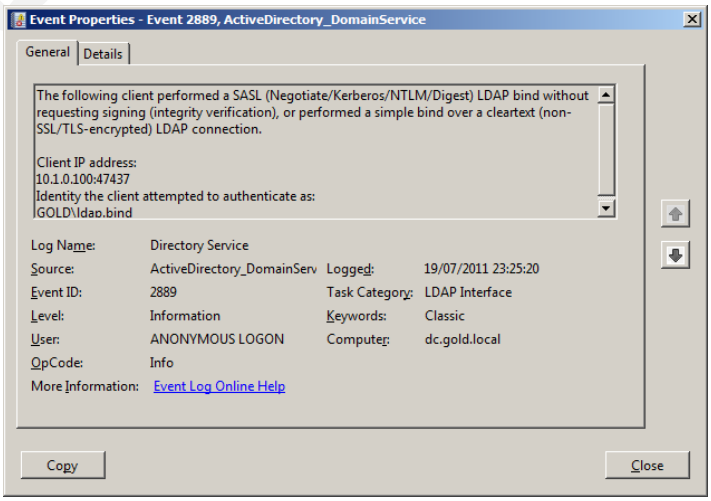


Figure 6.1 Windows event showing an LDAP Connection over an insecure channel

Andrew Reid;andy@andyreid.co.uk

© 2011 The SANS Institute Author retains full rights.

To enable this setting launch the registry editor and browse to the 'HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTDS\Diagnostics' registry sub key then amend the 'LDAP Interface Events' setting to 2 as shown in Figure 6.2, this will enable basic logging. Full details regarding Active Directory Diagnostic Logging can be found at the on Microsoft's TechNet site (Microsoft, 2011a).

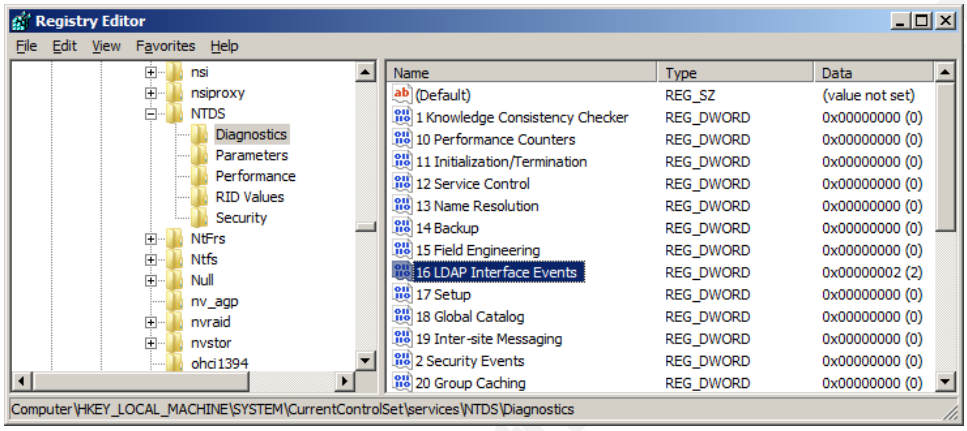


Figure 6.2 Enabling LDAP Interface Events

To configuration steps to disable insecure LDAP binds can be completed by amending the group policy, editing the domain controller's local policy or editing the domain controller's registry directly. Steps detailing the various ways of configuring this setting this can be found within the following Microsoft knowledge base article 'How to enable LDAP signing in Windows Server 2008' (Microsoft, 2011b). The steps shown below detail the process to enable the LDAP signing setting within the Default Domain Controllers Group Policy Object.

Step 1 – Edit the Default Domain Controllers Policy

From the Group Policy Management Console browse to the Domain Controllers OU, the right click and edit the Default Domain Controllers Policy shown in Figure 6.3.

© 2011 The SANS Institute Author retains full rights.

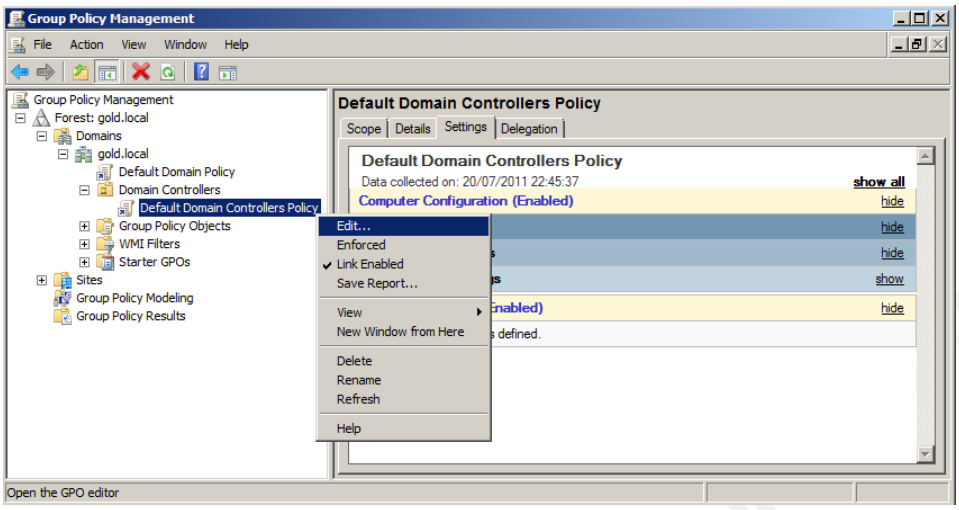


Figure 6.3 Editing the Default Domain Controllers Policy

Step 2 – Enable LDAP server signing requirements.

Within the Group Policy Management Editor browse to the 'Computer Configuration\Policies\Windows Settings\Security Settings\Local Policies\Security Options' registry key as shown in Figure 6.4. Double click on Domain controller: LDAP server signing requirements and change this setting from 'None' to 'Require signing' this is shown in Figure 6.5.

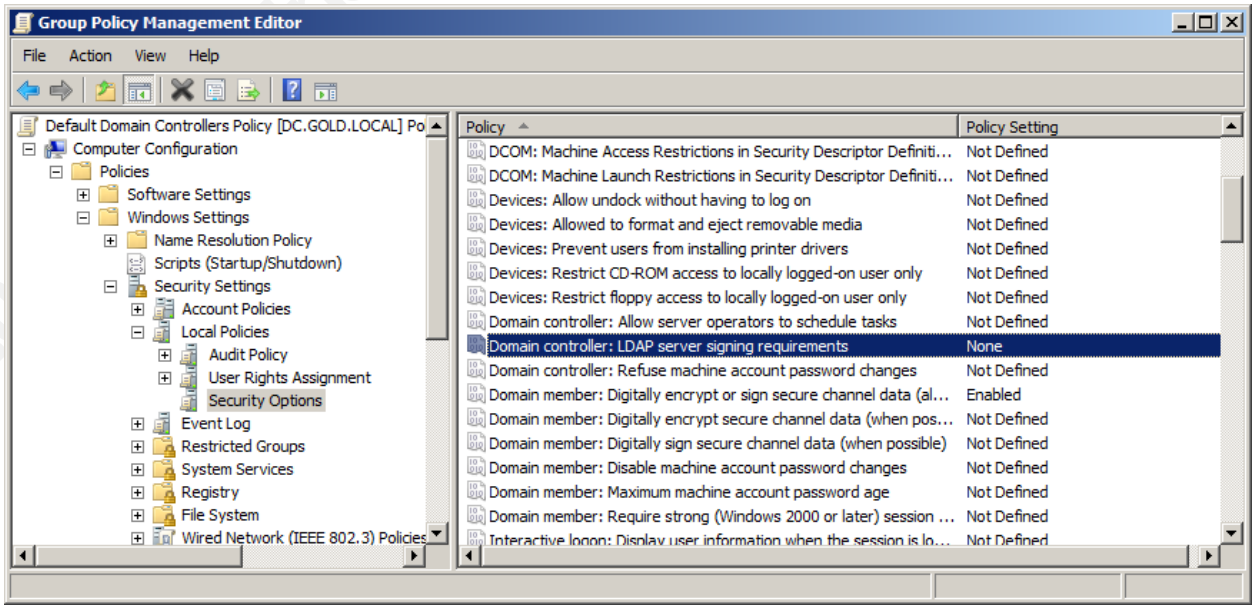


Figure 6.4 Locate the Security Options registry key

© 2011 The SANS Institute Author retains full rights.

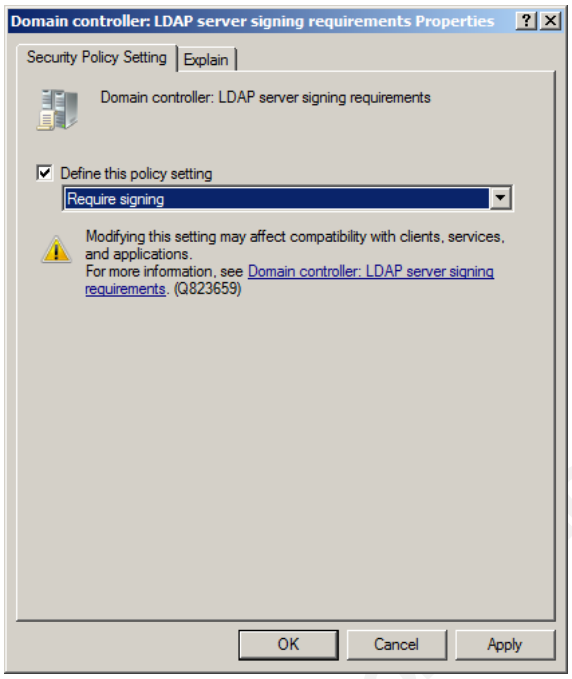


Figure 6.5 Enable the LDAP Require signing option

A warning will be displayed that this may affect compatibility with clients, services and applications. To continue with the change select 'Yes'. Now when any insecure connections are attempted the Domain Controller will respond with a bind response message that it requires strong authorisation and will terminate the connection shown in Figure 6.6.

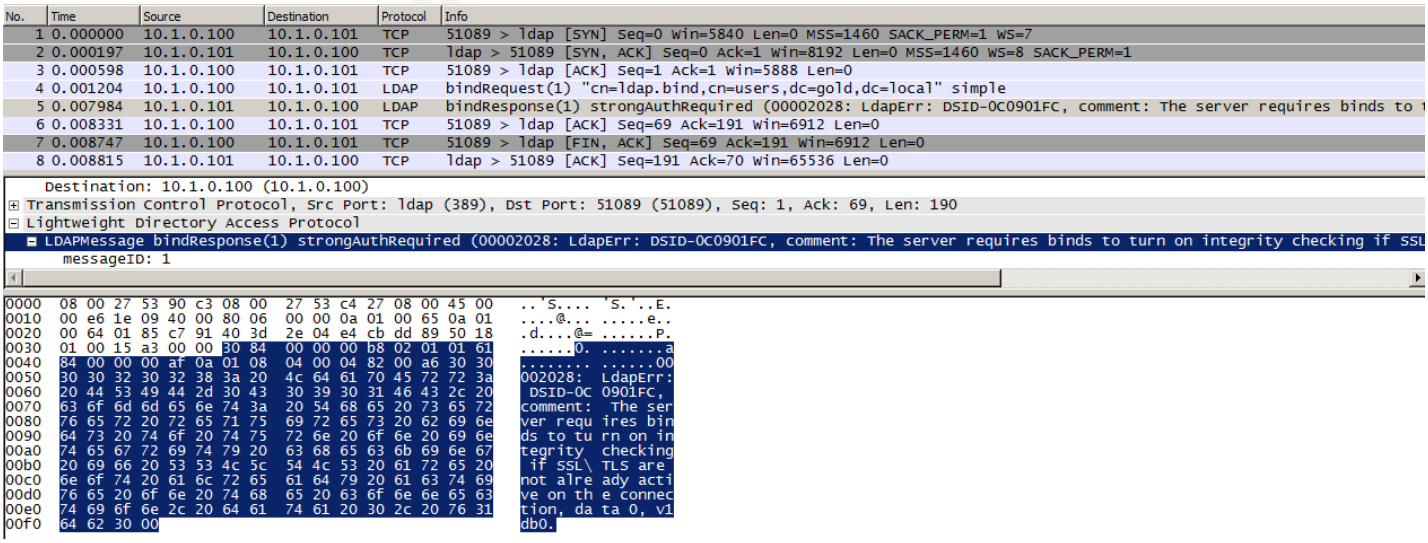


Figure 6.6 The Require Signing setting causes insecure connections to be rejected

Andrew Reid;andy@andyreid.co.uk

© 2011 The SANS Institute Author retains full rights.

7. Conclusion

It has been shown that the default configuration of a Microsoft Domain Controller does not allow secure connectivity using the LDAP protocol. This can put user credentials at risk if intercepted as they cross the network. Firstly through the use of network traces it was shown that when using LDAP all traffic including service account and user credentials were passed over the network in clear text.

Secondly using the steps outlined in this paper, a certificate was generated and installed to enable secure communication on a Domain Controller. After configuring the application server to use secure LDAP, network traces were re-run showing the application server and Domain Controller negotiating and using a secure connection for the transfer of authentication traffic and other data.

Finally configuration options were shown that would allow real time logging of insecure connections to take place, and if required turn off the ability of the Domain Controller to accept insecure connections.

This information allows Domain Administrators to understand the issues of insecure LDAP traffic and how to configure their infrastructure to reduce this risk as well as to detect and advise when insecure connections are made to their infrastructure.

Andrew Reid;andy@andyreid.co.uk

© 2011 The SANS Institute Author retains full rights.

8. References

Adams, C., & Farrell, S. (1999, March). *RFC2510 - Internet X.509 Public Key Infrastructure Certificate Management Protocols*. Retrieved from http://www.ietf.org/rfc/rfc2510.txt

Adams, C, & Lloyd, S. (2003). *Understanding pki: concepts, standards, and deployment considerations.* Addison-Wesley Professional.

Housley, R., Ford, W., Polk, W., & Solo, D. (1999, January). *RFC2459 - Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. Retrieved from http://www.ietf.org/rfc/rfc2459.txt

Menezes, A. Van Oorschot, P. & Vanstone, S. (1997*). Handbook of applied cryptography.* CRC.

Microsoft. (2011a). *Active Directory Diagnostic Logging*. Retrieved August 17, 2011, from http://technet.microsoft.com/en-gb/library/cc961809.aspx

Microsoft. (2011b). *How to enable LDAP signing in Windows Server*. Retrieved July 20, 2011 from http://support.microsoft.com/kb/935834

Rescorla, Eric. *SSL and TLS: designing and building secure systems*. Addison-Wesley Professional, 2001. Print.

Wahl, M., Howes, T., & Kille, S. (1997, December*).RFC2251 - Lightweight Directory Access Protocol (v3)*. Retrieved from http://www.ietf.org/rfc/rfc2251.txt

© 2011 The SANS Institute Author retains full rights.

Appendix A

Request.inf template file

```
;----------------- request.inf -----------------

[Version]

Signature="$Windows NT$

[NewRequest]

Subject = "E={email},CN={FQDN of DC},O={organisation},L={Location},S={State},C={Country}"
KeySpec = 1
KeyLength = 2048
; Can be 1024, 2048, 4096, 8192, or 16384.
; Larger key sizes are more secure, but have
; a greater impact on performance.
Exportable = TRUE
MachineKeySet = TRUE
SMIME = False
PrivateKeyArchive = FALSE
UserProtected = FALSE
UseExistingKeySet = FALSE
ProviderName = "Microsoft RSA SChannel Cryptographic Provider"
ProviderType = 12
RequestType = PKCS10
KeyUsage = 0xa0

[EnhancedKeyUsageExtension]

OID=1.3.6.1.5.5.7.3.1 ; this is for Server Authentication

;-------------------------------------------
```

Andrew Reid;andy@andyreid.co.uk

© 2011 The SANS Institute Author retains full rights.