

Kerberos V Security: Replay Attacks

Abstract

Kerberos V is a trusted third-party authentication mechanism designed for TCP/IP networks. It uses strong symmetric cryptography to enable secure authentication in an insecure network. In a Microsoft Windows domain, many protocols use Kerberos V as the primary authentication mechanism. We use SMB (Server Message Block) and LDAPv3 (Lightweight Directory Access Protocol) as examples of such protocols. We discuss how an attacker could use a replay attack to steal other users' identities on the local network. In addition to Windows 2000 and Windows 2003 we examine one open source (Samba 3.0) and one commercial (NetApp Data ONTAP 6.4R1) product and discuss implementation and configuration specifics that affect the feasibility of replay attacks.

Keywords

Kerberos V, replay attacks, authentication, SMB, LDAPv3

INTRODUCTION

Kerberos V is the authentication mechanism primarily used in a Microsoft Windows domain. In this paper we use SMB (Server Message Block) and LDAPv3 (Lightweight Directory Access Protocol) as examples of protocols that use Kerberos V for authentication. SMB is the protocol used for file and print services. Active Directory resources are accessed with LDAPv3.

Kerberos was developed at MIT as a part of Project Athena. It is based on a key distribution model invented by Roger Needham and Michael Schroeder (Needham, 1978). Symmetric cryptography and a trusted third-party are the basis of this authentication mechanism. There have been two versions of the protocol in public use, namely Kerberos IV and V. In this paper we discuss only Kerberos V.

The security of Kerberos has been discussed in several papers: see (Bellare, 1991) for an example. Possible weak points include replay attacks, password attacks against Ticket-Granting tickets or pre-authentication data, attacks against network time protocols (Kerberos requires time synchronization) and malicious client software. In this paper, we focus on the first scenario: replay attacks. We examine several server products (Windows 2000, Windows 2003, Samba 3.0, NetApp Data ONTAP 6.4R1) that operate in a Windows domain and discuss implementation specifics that affect the feasibility of replay attacks.

We show that replay attacks are feasible by presenting attacks on both SMB and LDAPv3. An attacker will be able to access file shares and modify directory entries with the victim's credentials. Some server implementations have actual weaknesses, while others have default configurations that make the attack possible.

BACKGROUND

This chapter will cover the technical details behind protocols and mechanisms that are relevant to this paper. We discuss the Kerberos V protocol itself and related mechanisms such as GSS-API (Generic Security Service Application Program Interface) and SPNEGO (Simple and Protected GSS-API Negotiation Mechanism). We also briefly explain the basics of SMB (Server Message Block) and LDAPv3 (Lightweight Directory Access Protocol).

KERBEROS V

Kerberos V authentication is based on symmetric cryptography and a trusted third-party that has access to all passwords. If a client wishes to authenticate himself to a network service, the typical process shown in Figure 1 includes the following steps:

- (i) The client requests for a Ticket-Granting Ticket (TGT) from an Authentication Server (AS)
- (ii) The AS will reply with the TGT
- (iii) The client will use the TGT to request for a ticket for the service he wishes to use from a Ticket-Granting Server (TGS)
- (iv) The TGS will reply with a ticket for the service
- (v) The client will use the ticket and some other information to authenticate himself to the server.

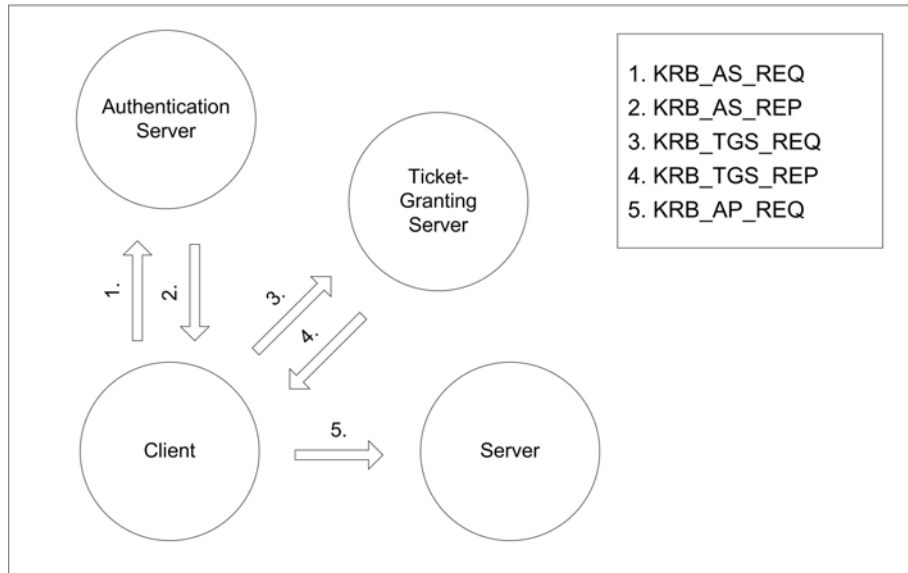


Figure 1: Kerberos V authentication messages.

For a detailed description of all the messages, we will refer the reader to (Kohl, 1993). To help understand the replay attack, we discuss the final message, KRB_AP_REQ, in more detail.

KRB_AP_REQ includes a ticket for the service the client is accessing and something called an authenticator. The ticket includes the following information (note that $\{x\}K_y$ means x is encrypted with key K_y):

$$T_{c,s} = server, \{client, addr, valid, K_{c,s}, flags\}K_s$$

The fields are defined as follows:

- *server* is the server's name
- *client* is the client's name
- *addr* is an array of valid source network addresses
- *valid* is the validity time of the ticket
- $K_{c,s}$ is a session key for the client and server to use
- *flags* describe ticket options
- K_s is the server's key

A new authenticator is created for each authentication attempt. Authenticators are of the following form:

$$A_{c,s} = \{client, time, checksum, seskey\}K_{c,s}$$

The authenticator is encrypted with the session key $K_{c,s}$. The fields are defined as follows:

- *client* is the client's name
- *time* is a timestamp
- *checksum* an optional checksum
- *seskey* is an optional additional session key

The replay attack will exploit this final message sent to the server.

GSS-API AND SPNEGO

The Generic Security Service Application Program Interface (Linn, 1997), as its name implies, provides software developers a generic security service framework having a well defined interface that is independent of the specific implementation details of the underlying security mechanisms. The provided services include security context initialization, entity authentication, data origin authentication and data integrity and confidentiality protection.

One security mechanism supporting all the mentioned features is the Kerberos Version V GSS-API Mechanism defined in (Linn, 1996).

GSS-API does not define any method that the participating peers can use to determine a common security mechanism they are sharing. To solve this problem the Simple and Protected GSS-API Negotiation Mechanism (Baize, 1998) was introduced.

GSS-API and SPNEGO are relevant when we discuss data integrity protection which has a large impact on the success of replay attacks.

Figure 2 describes the relationships between upper layer protocols, GSS-API, SPNEGO and Kerberos V.

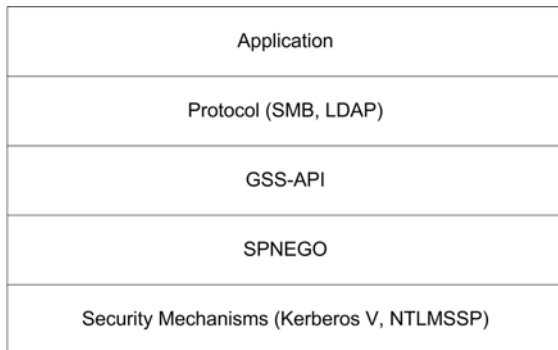


Figure 2: GSS-API and SPNEGO layers.

SMB

SMB (Server Message Block) is a protocol used for file and print services. It is a client-server, request-response protocol that enables servers to share file systems and other services. SMB primarily uses Kerberos V as an authentication mechanism in a Microsoft Windows domain. To those who are familiar with the acronym CIFS, it is the open version of SMB that some vendors are currently developing. The CIFS standard (SNIA, 2002) can be used as a reference for the authentication process.

We will go through the session negotiation and authentication steps, which are important in understanding the replay attack we implemented. These steps are shown in Figure 3.

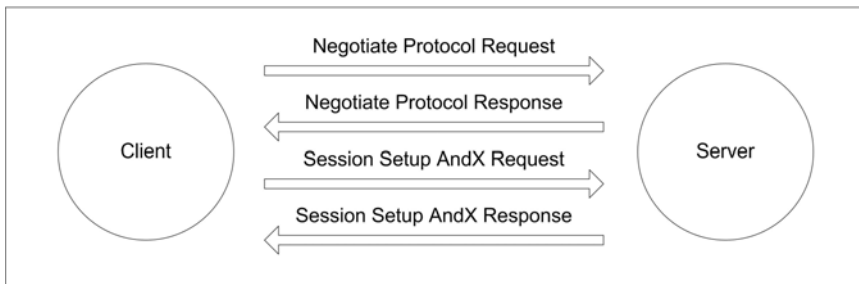


Figure 3: Messages transmitted in SMB session setup.

We assume that both parties accept Kerberos as the authentication mechanism. There are normally four steps:

- (i) A client will send a Negotiate Protocol Request to the server, telling which dialects it supports.
- (ii) The server responds with a Negotiate Protocol Response. It tells the client which dialect to use and other options it supports including support or requirement for data integrity.
- (iii) The client sends a Session Setup AndX Request to the server. In addition to other data, this message includes the `KRB_AP_REQ` described in the previous chapter.
- (iv) The server responds with a Session Setup AndX Response indicating everything was acceptable

LDAPV3

Lightweight Directory Access Protocol, more specifically LDAPv3, is defined in (Wahl, 1997). It is used to read and modify directory information. In a Microsoft Windows domain, nearly all relevant domain information, such as user and computer objects, is stored in an Active Directory server. Active Directory serves its clients using LDAPv3. The base authentication mechanism used is Kerberos V.

Authentication to an LDAPv3 server is done using a Bind Request message, which will include a KRB_AP_REQ from the client. With Windows clients, the KRB_AP_REQ will be enclosed in a SPNEGO token, which in turn is inside a GSS-API token. The replay attack will exploit this Bind Request message.

REPLAY ATTACKS ON KERBEROS V

A replay attack on Kerberos V exploits the final message, KRB_AP_REQ, presented in Figure 1. The contents of this message were discussed earlier. The KRB_AP_REQ will usually be inside an upper layer protocol message, such as the SMB Session Setup AndX or the LDAPv3 Bind Request. If an attacker is able to access the network traffic from the victim, he will be able to extract the KRB_AP_REQ sent by the victim. In a replay attack the attacker simply attempts to re-use this message to authenticate himself to a server. In some cases the server will accept the replayed message sent by the attacker allowing him full access to the service with the victim's credentials.

Accessing network traffic from the victim is essential for the attack to succeed. This can be accomplished in many ways. A common attack known as ARP spoofing uses forged ARP-reply packets allowing the attacker to impersonate the victim's default gateway. This method is discussed in more detail in (Kasslin, 2003).

Kerberos V does include mechanisms that are aimed at making replay attacks difficult:

- The ticket inside KRB_AP_REQ should include the network address of the client. The server should verify that the source address of the message equals the address in the ticket.
- The authenticator includes a timestamp. The server should verify that this timestamp falls inside the allowable time skew.
- The server should use a cache to store used authenticators. This cache should hold all authenticators that have been used within the allowable time skew.

However, as we will show, these mechanisms are clearly insufficient.

In many environments, the Kerberos KDC will not include network addresses in the tickets it gives out at all. Even if the tickets did include a network address, an attacker could easily forge his source address during the attack.

The authenticator timestamp only enforces a maximum time skew of a few minutes. In practice, this will not be of any help. An attacker will be able to mount the attack within seconds of recovering the KRB_AP_REQ. If the maximum time skew is configured to be too small, the nodes will face problems with time synchronization.

If the server uses a cache for the used authenticators, a passive attack becomes impossible. A server will reject all authenticators it has already seen. The attacker must now resort to an active attack. If he is able to sit between the victim and the server during session setup, the attacker can refuse to forward the initial KRB_AP_REQ to the server. After this, he can mount the attack by using the captured message. The server will not have seen the authenticator and the attack will succeed. Authenticator caching makes a replay attack slightly more difficult, but is not a sufficient protective mechanism.

In the chapter "Test Results and Analysis" we will discuss how server configuration and implementation affect the feasibility of replay attacks in practice.

PROTECTION AGAINST REPLAY ATTACKS

In this chapter we will examine different methods to defend against replay attacks. Some of them can be challenging to implement, especially if the environment consists of products from different vendors.

Network Connection Hijacking

For the replay attack to be feasible, the attacker must be able to listen to the network traffic between the victim and the server. In some cases he must also be able to actively control the traffic flow from the victim. This is more commonly known as network connection hijacking. Network connection hijacking can be done in many ways. In this

chapter we discuss the solutions against ARP spoofing, which is probably the most efficient and common technique to subvert traffic from a node within a network segment.

There are generally two well known ways to detect ARP spoofing attempts – monitoring the local ARP cache and monitoring the network traffic on the wire.

ARP cache monitoring on a local machine can be accomplished on most systems with the *arp* command. It is quite trivial to notice if the gateway’s hardware address has changed (assuming the real address of the gateway is known). This can be done automatically with a tool called *arpwatch*, available from <<http://www-nrg.ee.lbl.gov/>>.

Network traffic monitoring can be implemented with certain Intrusion Detection Systems. The Open Source IDS called Snort is able to do this in real time. Snort is available from <<http://www.snort.org/>>.

One of the best ways to protect machines against ARP spoofing attacks is to enforce static ARP entries on the local machines; especially the entry for the local gateway should be static. This should be possible on most systems.

IPSec

To protect your network against replay attacks, one efficient solution is deploying encryption on the IP layer. The use of IPSec (Kent, 1998) would be a sufficient protective action. If all IP traffic is encrypted the attacker will have no way of intercepting the required data. In many environments deploying IPSec to secure all client-server traffic is a challenging task which often requires a working PKI (public key infrastructure).

It should be noted that all the network nodes must be configured to require IPSec on all relevant connections. An active attacker can easily block all IPSec IKE negotiation traffic between the victim and the server by simply dropping all traffic to UDP port 500. If the nodes are then allowed to fall back to plaintext communication, they will be vulnerable to replay attacks.

Data Integrity with SMB and LDAPv3

During a replay attack the attacker is simply re-using data sent by the victim. Inside the service ticket, which is encrypted using the server’s key, is a session key that the victim and server will share. The attacker cannot gain information about this session key at any point.

Upper layer protocols such as SMB and LDAPv3 offer data integrity protection that relies on this session key. A keyed cryptographic checksum of the contents can be included with each packet. An attacker cannot create these checksums, because he does not know the session key. Therefore, data integrity protection, if configured correctly, can also be used to protect against replay attacks.

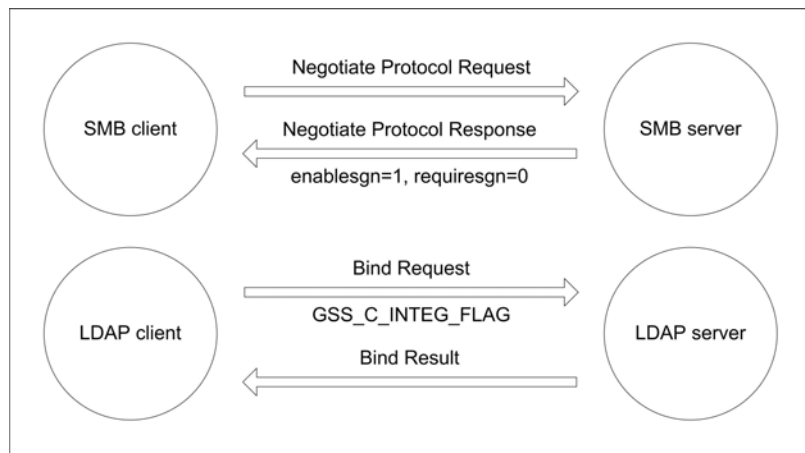


Figure 4: Integrity negotiation with SMB and LDAPv3.

The SMB protocol itself has support for data integrity. During session negotiations, the server Negotiate Protocol Response includes two flags that indicate if the server supports integrity and if the client is required to use it. However, the client has no means to declare its support to the server. If the parties agree on using SMB signing, all following packets will include a checksum calculated with the session key. Client and server implementations can usually be configured to disable, enable or require signatures.

LDAPv3 uses the GSS-API layer for data integrity. The GSS-API token in the client's Bind Request will include the session options it supports. The Kerberos V mechanism for GSS-API (Linn, 1996) defines a new authenticator checksum type (0x8003) for the authenticator inside the KRB_AP_REQ that is sent with the initial token. The checksum will include option flags, including the data integrity option (GSS_C_INTEG_FLAG). If both parties agree, subsequent packets will include a cryptographic checksum to ensure their integrity.

It should be noted that even though SMB signing does not use GSS-API token data integrity, some clients will use the GSS-API checksum inside the authenticator to declare support for this feature.

We will discuss correct server and client configuration in the next chapters. However, as a rule of thumb, servers should always require data integrity from all clients. Deploying data integrity protection should be relatively simple in a domain environment if all members fully support it.

TEST RESULTS AND ANALYSIS

Windows 2000 Server and Windows 2003 Server

Windows 2003 Server is the latest server operating system from Microsoft (<<http://www.microsoft.com/windowsserver2003>>). It will replace the older but still widely used Windows 2000 Server (<<http://www.microsoft.com/windows2000/server>>). Our results show that replay attacks against SMB and LDAPv3 protocols using Kerberos V for authentication are possible in both environments.

Authenticator Caching

In our tests we discovered that Windows 2000 Server SP4 and Windows 2003 Server do cache used authenticators. This makes replay attacks more difficult, as the attacker has to prevent the server from receiving the captured authenticator. In practice, instead of just passively listening to network traffic, the attacker must now use an active man-in-the-middle attack.

Enforcing Ticket Addresses

We also noticed that the attacks are simplified by not enforcing matching network addresses in the service tickets. A Windows 2000 Server and Windows 2003 Server acting as a Kerberos KDC will never include network addresses in the tickets it gives out. This means that it is not necessary for the attacker to steal the victim's network address at any point.

Handling of Kerberos Principals

In our tests we discovered that a Windows 2000 Server and Windows 2003 Server will accept connections to different services (SMB, LDAPv3) with a single service ticket. In other words, you will be able to authenticate to an LDAPv3 service using the same KRB_AP_REQ you used with an SMB service as long as they reside on the same server. All service principals are mapped to a single computer account. This means that an attacker can use a KRB_AP_REQ captured from an LDAPv3 connection and use it to connect to a SMB file shares and vice versa.

Data Integrity

The result of the replay attack depends heavily on the fact if integrity protection is used and configured correctly. Windows 2000 and Windows 2003 have support for integrity protection in both client and server implementations. The figure below shows the default settings of the relevant configuration options on the client and server regarding SMB and LDAPv3 signing.

	Windows 2000 Pro SP4	Windows 2000 Server SP4	Windows XP SP1	Windows 2003 Server
SMB client: enable signing	enabled	enabled	enabled	enabled
SMB client: require signing	disabled	disabled	disabled	disabled
SMB server: enable signing	disabled	enabled	disabled	enabled
SMB server: require signing	disabled	disabled	disabled	enabled
LDAP client: signing requirements	negotiate	negotiate	negotiate	negotiate
LDAP server: signing requirements	-	negotiate	-	negotiate

Figure 5: Default integrity settings on Windows 2000 and Windows 2003.

Default Windows 2000 Server settings do not provide any protection against the replay attack on the SMB service. With SMB, the service has to require integrity protection to prevent the attack, which is the case with Windows 2003 Server.

By default, the LDAPv3 service is not vulnerable to a replay attack. In this case it is enough if both the server and the client are configured to support integrity protection. If an attacker attempts to replay a captured KRB_AP_REQ message with the GSS-API GSS_C_INTEG_FLAG turned on, the server will require data integrity and the attack will fail.

However, we would recommend that LDAPv3 servers be configured to require signing from all clients. Because of the way Windows 2000 Server and Windows 2003 Server handle Kerberos principals, the attacker will be able to use captured KRB_AP_REQ messages from other connections such as SMB. If a client does not have support for integrity, the attack against LDAPv3 becomes possible by capturing traffic from this client.

The complete analysis of the effects the permutations of SMB signing settings have on client-server communication and the feasibility of replay attacks is included in Appendix 1. In short, requiring integrity on the server gives sufficient protection from replay attacks.

Samba 3.0

Samba is an open source product available from <<http://www.samba.org>>. Among other features it is able to act as a file server in a native Windows domain.

The tests were conducted with Samba 3.0 beta1, which was the latest version available at the time. After contacting developers, some tests were conducted with authenticator caching on development versions to verify changes.

Authenticator Caching

Our tests verified that Samba 3.0 beta1 does not cache used authenticators. The replay attack is possible by passively listening to the traffic and replaying a KRB_AP_REQ the server had already seen. This makes Samba 3.0 beta1 more vulnerable to a replay attack.

In later beta versions and in the final 3.0 release the authenticator cache should be implemented. This means that a passive attack will not work. An active attack is still possible, as is the case with Windows 2000 Server and Windows 2003 Server.

Enforcing Ticket Addresses

The test environment for Samba 3.0 used Windows 2000 Server as the Kerberos KDC. As already stated before, network addresses are not included in tickets from any Windows Server KDC. Therefore, Samba 3.0 beta1 cannot enforce ticket addresses even though code was added to implement this feature.

Data Integrity

Samba 3.0 beta1 does not include server-side support for SMB data integrity with Kerberos V. This means that the replay attack cannot be prevented by server configuration. It is not known if support for this feature will make it to the final Samba 3.0 release.

Also, the SMB client-side application (*smclient*) included with Samba 3.0 beta1 does not include support for SMB data integrity with Kerberos V. Naturally this means that it cannot communicate with servers that require data integrity. However, if we consider interoperability with Windows and remember the handling of Kerberos Principals discussed earlier, another small attack vector against LDAPv3 should be noted. Consider an environment where all LDAPv3 and SMB clients except *smclient* declare support for GSS-API data integrity. They do this by setting the GSS_C_INTEG_FLAG flag inside the KRB_AP_REQ. Windows SMB clients do this, even though SMB integrity is not negotiated this way. Now, if the LDAPv3 service is configured not to require data integrity, the only possibility for a replay attack to succeed is by using a captured KRB_AP_REQ from *smclient* to an SMB service running on the same server. Because the GSS-API integrity flag is not set, the LDAP server will correctly allow the attacker to connect without data integrity thus making the attack possible.

NetApp Data ONTAP 6.4R1

NetApp provides highly available storage solutions which have full interoperability with Microsoft Windows. Data ONTAP is the operating system used in NetApp file servers (<<http://www.netapp.com/products/filer/ontap.html>>).

The Data ONTAP 6.4R1 was the latest software version available at the time this paper was written. NetApp was contacted and they confirmed all the found weaknesses. The latest information we have is that the problems will be corrected in the next release.

Authenticator Caching

Our tests showed that NetApp suffers from the same problem as Samba 3.0 beta1. It does not cache used authenticators and so the attack was possible by just monitoring the network traffic. Used authenticators could be exploited multiple times within the time skew.

Enforcing Ticket Addresses

During the tests NetApp was a member of a domain with Windows 2000 Server as the KDC. It was stated before that the KDC does not include the network addresses in the service tickets it provides. This means that NetApp is not the source of the problem of not enforcing ticket address validation.

Data Integrity

NetApp Data ONTAP 6.4R1 does not provide server-side support for SMB data integrity at all. This leaves the product completely open for replay attacks. It should be noted that the only protection available is to use IPSec. However, at this moment PSK (Pre-Shared Key) is the only authentication method supported by Data ONTAP 6.4R1 and no hardware acceleration is available.

CONCLUSIONS

Our research shows that replay attacks against protocols that use Kerberos V for authentication are possible with modest resources. Some implementations contain weaknesses in their implementation, while others have default configurations that make replay attacks possible. An attacker will be able to access network resources such as directories and file shares with stolen user credentials.

If an attacker has access to the network, securing the environment against a replay attack is not trivial and in practice requires deployment of additional cryptographic mechanisms. Networks, such as campus environments, where physical access point security cannot be guaranteed are especially problematic. Administrators should consider enforcing upper layer data integrity if possible. IPSec is also a solution, but extensive and correct deployment is not trivial.

When evaluating new services that support Kerberos V, some attention should be paid on implementation details we have discussed. The upper layer protocol specifics are also important, because features such as data integrity are extremely significant in protecting against replay attacks.

REFERENCES

- Baize E. and Pinkas D. (1998). The Simple and Protected GSS-API Negotiation Mechanism. Request for Comments RFC 2478.
- Bellovin S. and Merritt M (1991). Limitations of the Kerberos Protocol. Winter 1991 USENIX Conference Proceedings
- Kasslin K. and Tikkanen A. (2003). Hijacking a Network Connection on a Switched Network. <http://www.hut.fi/u/autikkan/kerberos/docs/phase1/pdf/LATEST_hijacking_attack.pdf>
- Kent S. and Atkinson R. (1998). Security Architecture for the Internet Protocol. Request for Comments RFC 2401.
- Kohl J. and Neuman B. C. (1993). The Kerberos Network Authentication Service (Version 5). Request for Comments RFC 1510.
- Linn J. (1996). The Kerberos Version 5 GSS-API Mechanism. Request for Comments RFC 1964.
- Linn J. (1997). Generic Security Service Application Program Interface, Version 2. Request for Comments RFC 2078.
- Needham, R.M and Schroeder M.D. (1978). Using Encryption for Authentication in Large Networks of Computers. Communications of the ACM, v. 21, n. 12, Dec 1978, pp. 993-999.

SNIA Storage Networking Industry Association (2002). Common Internet File System (CIFS) Technical Reference Revision 1.0.
 <http://www.snia.org/tech_activities/CIFS/CIFS-TR-1p00_FINAL.pdf> (2003, Aug 2).

Wahl, M., Howes T. and Kille S. (1997). Lightweight Directory Access Protocol (v3). Request for Comments RFC 2251.

ACKNOWLEDGEMENTS

The authors wish to thank Timo P. Aalto for his valuable comments regarding this paper. We also express our gratitude to Helsinki University of Technology Computing Centre for supporting our research.

APPENDIX 1

The tables describe all SMB signing configuration permutations between the two client-server pairs.

The configuration options refer to registry keys *enablesecuritysignature* and *requiresecuritysignature* at *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\lanmanworkstation* for clients and *\lanmanserver* for servers.

The results indicate if a connection is possible between a client and a server, if SMB signing is used during this connection and if a replay attack is possible. The large differences in results can be explained by different interpretations of the configuration options. For both parties in the first table, the *enable* registry key is a precondition for *require*. This does not hold for the parties in the second table.

CLIENT: Windows 2000 SP4		SERVER: Windows 2000 Server SP4		RESULTS		
Required	Enabled	Required	Enabled	Connection	SMB signing	Attack
No	No	No	No	OK	No	OK
No	No	No	Yes	OK	No	OK
No	No	Yes	No	OK	No	OK
No	No	Yes	Yes	Fails	-	Fails
No	Yes	No	No	OK	No	OK
No	Yes	No	Yes	OK	Yes	OK
No	Yes	Yes	No	OK	No	OK
No	Yes	Yes	Yes	OK	Yes	Fails
Yes	No	No	No	OK	No	OK
Yes	No	No	Yes	OK	No	OK
Yes	No	Yes	No	OK	No	OK
Yes	No	Yes	Yes	Fails	-	Fails
Yes	Yes	No	No	Fails	-	OK ¹
Yes	Yes	No	Yes	OK	Yes	OK
Yes	Yes	Yes	No	Fails	-	OK ¹
Yes	Yes	Yes	Yes	OK	Yes	Fails

Figure 6: SMB signing configurations: Windows 2000 SP4 and Windows 2000 Server SP4.

CLIENT: Windows XP SP1		SERVER: Windows 2003 Server		RESULTS		
Required	Enabled	Required	Enabled	Connection	SMB signing	Attack
No	No	No	No	OK	No	OK
No	No	No	Yes	OK	No	OK
No	No	Yes	No	Fails	-	Fails
No	No	Yes	Yes	Fails	-	Fails
No	Yes	No	No	OK	No	OK
No	Yes	No	Yes	OK	Yes	OK
No	Yes	Yes	No	OK	Yes	Fails
No	Yes	Yes	Yes	OK	Yes	Fails
Yes	No	No	No	Fails	-	OK ¹
Yes	No	No	Yes	OK	Yes	OK
Yes	No	Yes	No	OK	Yes	Fails

Yes	No	Yes	Yes	OK	Yes	Fails
Yes	Yes	No	No	Fails	-	OK ¹
Yes	Yes	No	Yes	OK	Yes	OK
Yes	Yes	Yes	No	OK	Yes	Fails
Yes	Yes	Yes	Yes	OK	Yes	Fails

Figure 7: SMB signing configurations: Windows XP SP1 and Windows 2003 Server.

¹ Requires modifying the server's Negotiate Protocol Response to include support for signing.

COPYRIGHT

[Kimmo Kasslin, Antti Tikkanen and Teemupekka Virtanen] © 2003. The author/s assign the AIWSC03 & University of South Australia a non-exclusive license to use this document for personal use provided that the article is used in full and this copyright statement is reproduced. The authors also grant a non-exclusive license to the AIWSC03 & UniSA to publish this document in full in the Conference Proceedings. Such documents may be published on the World Wide Web, CD-ROM, in printed form, and on mirror sites on the World Wide Web. Any other usage is prohibited without the express permission of the authors.