

Microsoft Windows Remote Desktop Protocol checksum and keystroke vulnerabilities

Source: <http://www.derkeiler.com/Mailing-Lists/securityfocus/bugtraq/2002-09/0195.html>

From: Ben Cohen (bc@skygate.co.uk)

Date: 09/16/02

Date: Mon, 16 Sep 2002 09:52:00 +0100 (BST)

From: Ben Cohen <bc@skygate.co.uk>

To: <bugtraq@securityfocus.com>

Vulnerable

All versions of Microsoft Windows using encrypted RDP are vulnerable to the checksum vulnerability. The following versions are also vulnerable to the keystroke vulnerability:

Microsoft Windows XP Professional
Microsoft Windows 2000 Server
Microsoft Windows 2000 Advanced Server
Microsoft Windows .NET Standard Server Beta 3

Background

A design mistake in Microsoft's Remote Desktop Protocol (RDP) leaks information about the contents of encrypted packets through their checksums, since packets with the same plaintext have matching checksums.

A form of input packet used in version 5.0 of the protocol allows an attacker to watch the network traffic from an encrypted session and work out what keystrokes a user makes. Further, it is then possible for the attacker to manipulate subsequent keystrokes sent across the network.

Checksum vulnerability

When RDP has encryption enabled, packets are first encrypted using RC4, then an 8 byte HMAC checksum of the plaintext is prepended to the cyphertext. The encryption key for RC4 is refreshed every 4096 packets, but the HMAC key is apparently not changed during the session. Note that the checksum relies only on the packet length, the packet contents and the HMAC key; if a packet with identical contents is sent twice, the two checksums will be the same, potentially leaking information if the same packet is sent repeatedly. This is known as encrypt-then-authenticate, and it is clearly a general design flaw in RDP. (This problem is further discussed in Hugo Krawczyk's paper.)

This leakage can be observed in the RDP 5.0 style of input packets (which leads to the keystroke vulnerability below). Other examples include the same rendering operation being sent multiple times, for example to make a cursor flash; and Terminal Services Client's device redirection, such as identical data repeatedly transmitted from the serial port. Plugins using Microsoft's Terminal Services Virtual Channels are also vulnerable.

Keystroke vulnerability

In RDP 4.0, input events are sent with a unique timestamp as specified in the T.128 standard; this packet is then framed using lower level protocols T.125 and T.123 and transmitted using TCP. (T.125 is used by Microsoft's RDP Virtual Channel API to multiplex other data such as audio and printing across the same connection.) With this many protocol layers, a single key press requires a packet 56 bytes long to be transmitted over TCP, which is inefficient.

In RDP 5.0 Microsoft added an abbreviated way to specify common commands including input events. These packets start 0x84 (and other values) rather than 0x03 which is used to start T.123. Encryption is done in the same way, but only the key code and key event type (press, repeat or release) is sent — there is no timestamp — so the packets are now only 12 bytes long.

Therefore the checksum is the same for each key event type for the same key. This only fails when the client concatenates input events into a single packet but this is uncommon. This reasoning does not apply to RDP 4.0 input packets because a timestamp is sent so the packet contents and therefore the checksum will change.

Here is an example taken from a real session:

Client to server:

84 0c c5 db ec cd 6c 7e 31 6c 47 a2

Client to server:

84 0c ef 6a bb 01 21 b0 c3 c6 f6 e5

...

Client to server:

84 0c c5 db ec cd 6c 7e 31 6c c1 0a

Client to server:

84 0c ef 6a bb 01 21 b0 c3 c6 fe bd

...

Client to server:

84 0c c5 db ec cd 6c 7e 31 6c 8e 41

Client to server:

```
84 0c ef 6a bb 01 21 b0 c3 c6 22 f8
```

The first byte (0x84) shows that these are RDP 5.0 style packets, not T.123. The second shows the length (i.e., the packets are 0x0c bytes long). The next 8 bytes are the HMAC checksum, and the last two are the encrypted packet contents for the input event.

Since the two pairs of checksums match, the same thing was being sent in each case. Indeed, the same key had been pressed several times, although it isn't possible to tell which key it was from the data given above. Given a larger dump of session data it would be possible to make a good guess for which key each checksum corresponds to by looking at the frequencies and ordering of the different checksums; this is simply a mono-alphabetic substitution cipher on the scan code plus key event type. This knowledge could then be used to work out what the user types, and the attacker might then be able to capture the user's passwords.

(Data from packets which do not start "84 0c" are a different type or length so can be ignored. Apparently the only packets of that length sent by the client are key events.)

Pete Chown pointed out that this is also subject to a substitution attack since the attacker now knows enough to be able to change a key input event to another one. Suppose the user presses "A" and the attacker wants a "B" key press to be sent instead. The attacker knows that the checksum and the plaintext for each of these two events. Since the cypher is RC4 and the plaintext for the events are the same, the desired cyphertext is obtained from XORing the "A" press plaintext and then XORing the "B" press plaintext. So the substitution is made simply by swapping the "A" press checksum with the "B" press checksum, and replacing the cyphertext.

Workaround

Use the Microsoft Terminal Services Client version 4.0, rather than later versions of Terminal Services Client or the Advanced Terminal Services Client. This only cures the keystroke vulnerability and does not address the problem for multiple packets with the same contents.

References

The Order of Encryption and Authentication for Protecting Communications (or: How Secure is SSL?), Hugo Krawczyk,
<http://link.springer.de/link/service/series/0558/bibs/2139/21390310.htm>

Section 8.18 from T.128 Multipoint application sharing, Series T:
Terminals for telematic services, ITU-T.

T.125 Multipoint communication service protocol specification, Series T:
Terminals for telematic services, ITU-T.

SecurityFocus Bugtraq: Microsoft Windows Remote Desktop Protocol checksum and keystroke vulnerabilities

T.123 Network-specific data protocol stacks for multimedia conferencing,
Series T: Terminals for telematic services, ITU-T (also RFCs 905 and 1006)

Using Terminal Services Virtual Channels, MSDN Platform SDK: Terminal
Services,

[http://msdn.microsoft.com/library/en-us/termserv/termserv/
using_terminal_services_virtual_channels.asp](http://msdn.microsoft.com/library/en-us/termserv/termserv/using_terminal_services_virtual_channels.asp)

HMAC: Keyed-Hashing for Message Authentication, RFC 2104,
<http://www.ietf.org/rfc/rfc2104.txt>

Microsoft was notified on 16 April 2002.

Credits

Ben Cohen
Software Developer
ben.cohen@skygate.co.uk

Pete Chown
Chief Security Consultant
pete.chown@skygate.co.uk

Skygate Technology Ltd.
<http://www.skygate.co.uk/>
+44 (0)20 8542 7856

- **Previous message:** [Ben Cohen: "Microsoft Windows XP Remote Desktop denial of service vulnerability"](#)
- **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#) [\[attachment \]](#)