CVE-1999-0015: Teardrop Denial-of-Service

Paper in Computer Security (MTAT.03.134)

Author: Rain Vagel
Supervisor: Meelis Roos

TARTU 2016
The "Teardrop" vulnerability first affected Windows 3.1, 95 and NT machines, as well as Linux versions older than 2.0.32 and 2.1.63. (University of Hamburg). It was first entered into the National Vulnerability Database in the year 1999. This vulnerability resurfaced ten years later in Windows Vista and Windows 7. This time due to vulnerabilities in Server Message Block Version 2 (SMBv2) (Microsoft, 2009). Other operating systems were spared thanks to not having the vulnerable drivers. (Naraine, 2009). This vulnerability is called the SMBv2 Negotiation Vulnerability - CVE - 2009-3103. Microsoft ranked the severity of this vulnerability as "critical".
DESCRIPTION OF THE PROBLEM

A "teardrop" attack consists of an attacker sending a series of fragmented IP datagram pairs to the target system with the amount of pairs required for a successful attack depending on the system at hand. For example a Windows NT machine requires up to 50 pairs, while one pair is enough to crash a Linux machine. (IBM) The fragments themselves are sent with different offsets, the first one with an offset of 0 and a payload of N. The next fragments are sent with offsets that tells the IP that is should overlap inside the previous fragment. However the packets have very small or non-existent payloads. The target system will then restart or crash.

This is possible due to a bug in the TCP/IP fragmentation reassembly code. Usually networks have different hardware with different transmission speeds and different sized maximum transmission units (MTU). So when a network sends another network with a smaller MTU some datagrams, it must fragment the datagrams.

Each datagram has a length in bytes, an ID and a fragflag that tells the reassembly process if there is a fragmented datagram coming after it. Finally it has the offset value, which tells the reassembly code where the datagram must be located. So when we have a network with an MTU value of 1500 bytes and we want to send a datagram with a length of 4000 bytes, we must naturally fragment it. Since the IP header in a datagram consists of 20 bytes, the actual data is the size of 3980 bytes. This would normally be fragmented into three different datagrams. The first two would have byte length of 1500 and the third a byte length of 1040. The offset value comes from where the last datagram ended and as the offset is measured in units of 8 octets (64 bits), we will do the calculation firstOffset = 1480 / 8. 1480 since the IP header is not counted in. This will give us an offset value of 185. The third datagram will have an offset of 2 * 1480 / 8 = 370. Every fragment except for the last must contain a multiple of 8 bytes of data (Information Sciences Institute, University of California, 1981).
IP Fragmentation and Reassembly
(Example)

MTU = 1500 Bytes, Offset = 1480/8

<table>
<thead>
<tr>
<th>Length</th>
<th>ID</th>
<th>fragflag</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>4000</td>
<td>X</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1500</td>
<td>X</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1500</td>
<td>X</td>
<td>1</td>
<td>185</td>
</tr>
<tr>
<td>1040</td>
<td>X</td>
<td>0</td>
<td>370</td>
</tr>
</tbody>
</table>

**Length** - The size of the fragmented datagram

**ID** - The ID of the datagram being fragmented

**Fragflag** - Indicates whether there are more incoming fragments

**Offset** - Details the order the fragments should be placed in during reassembly

*Figure 1 IP fragmentation (Imperva Incapsula)*
ATTACK VECTOR

The weakness is in certain operating systems in which a bug causes the TCP/IP fragmentation reassembly code to improperly handle overlapping IP fragments (University of Hamburg). Let an attacker send n pairs of datagrams to a victim's computer in which the first datagram has an offset of 0, with a fragflag set to 1 and a payload the size of \( N_1 \) and follows that with a datagram with an offset of \( N_1 / 8 - x \), a fragflag of 0 and a payload \( N_2 \). Let \( N_1, N_2, x, n \) be some integers where \( N_1 \mod 8 = 0 = N_2 \mod 8 \). In this case there is a very obvious case of fragment overlap, although everything else is in order. When an operating system that is vulnerable to "teardrop" attacks receives a datagram fragmented in such a way, it will improperly handle it and the system will either crash or restart. The total number of improperly fragmented datagrams depends on the system under attack. (IBM)

SOLUTION

The most common way to mitigate this type of attack is to monitor incoming data packets before they reach their destination for violations of fragmentation rules. This can be done with either a router or a security proxy (Imperva Incapsula). Since the overlapping fragments must first reach the system, which will then start the reassembly process, not letting them reach the system at all will stop the attack from affecting even the systems which have this vulnerability in them. Another general strategy of mitigating denial of service attacks in general is by the use of access control lists (ACL). ACL is a tool which can be used to control incoming and outgoing traffic of a network as it can stop specific sets of IP packets. However there is a loophole in this defense mechanism as it stops service to an entire network and not just one attacker (rao, 2011). Another method is the use of rate limiting., which unlike ACL does not separate the attacking network completely off the victim. It instead sets up a cap or limit on what the network can withstand. It is a method that is implemented by most data providers as it is extremely effective. With the use of this technique the network administrator can choose how much traffic will be allowed on the network at any given time (rao, 2011).
However while the last two methods are effective in use against mitigating denial of service attacks, they do not specifically handle "teardrop", as in TCP/IP fragmentation, attacks. Since this vulnerability is well known, the most useful suggestion is to update or upgrade your software or operating system. In the form of Linux machines, you should 2.0.32 / 2.1.63 or later (University of Hamburg). Although for a short period of time this sort of attack was possible on newer Windows Vista and Windows 7 machines, the problem has now been fixed with a simple software update (Microsoft, 2009).

CONCLUSION

This specific denial of service attack is and was possible on older machines, such as Windows 95, Windows 2000 and Linux versions previous to 2.0.32 and 2.1.63. While general methods of mitigation denial of service attacks, such as the use of rate limiting and ACL, are of great use. A more specific method such as scanning incoming fragments in a router or a security proxy for violation of fragmentation rules might be of more use. Then again as mentioned in previous cited works, the most effective solution for this specific old attack method, is to upgrade or update your software.
Works Cited


