

MWR InfoSecurity Security
Advisory

WebSphere MQ TCPReceive
Heap Overflow Vulnerability

12th January 2009



Contents

1	Detailed Vulnerability Description	5
1.1	Introduction	5
1.2	Technical Background.....	5
1.3	Exploit Information	7
1.4	Dependencies	7
2	Recommendations.....	8

WebSphere MQ TCPReceive Heap Overflow

Package Name:	WebSphere MQ
Vendor Notified:	20 th September 2007
Advisory Release Date:	12 th January 2009
Affected Versions:	WebSphere MQ 6.0.0.0 on Windows is confirmed to be vulnerable. Other versions and platforms may also be affected by this issue.

CVE Reference	CVE-2008-4288
Author	M. Ruks
Severity	High Risk
Local/Remote	Remote
Vulnerability Class	Signed Integer Error Leading to Heap Buffer Overflow
Impact	Denial of Service (Code execution will be difficult, although it may be possible)
Vendor Response	Updated packages have been created and are included in MQ release 6.0.2.4.
Exploit Details Included	Limited information about exploitation vectors is included
Affected OS	Microsoft Windows is confirmed to be vulnerable although other platforms are also expected to be affected

Overview:

The WebSphere MQ service can be used to transfer messages between systems and applications. A signed check error and subsequent heap buffer overflow vulnerability has been identified in the TCPReceive function. The vulnerability is associated with the copying of data received in MQ packets on the heap. This could be used to terminate a core MQ process and although this would restart, this technique could still be used to perform a Denial of Service (DoS) attack. Given sufficient time and effort this issue could potentially result in the execution of arbitrary code. The vulnerable function can be reached in a number of ways and could be exploited by unauthenticated attackers.

Impact:

The vulnerability could enable an attacker to create a DoS condition for remote users of the service. In addition, it should be assumed that this vector could provide the ability to execute arbitrary code on the affected system, with the privileges of the MQ process.

Cause:

The vulnerability arises from an error in the checking of an integer value taken from packet data using a signed check. The error enables length checks to be bypassed and allows the



size of a memory copy operation to be controlled. This in turn allows an area of the heap to be overwritten and results in corruption of the data.

Interim Workaround:

One method for mitigating the risk associated with this issue would be to use network filtering to restrict access to WebSphere MQ services to trusted IP addresses only.

Solution:

The vendor supplied patches should be installed to resolve this issue. Links to the updated software can be discovered at the following location: -

<http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg27006037>

1 Detailed Vulnerability Description

1.1 Introduction

WebSphere MQ is an Enterprise level application that can be used to provide a unified platform for messaging within an organisation. IBM describes their technology as follows: -

“WebSphere MQ provides an award-winning messaging backbone for deploying your enterprise service bus (ESB) today as the connectivity layer of a service-orientated architecture (SOA).”

Source: <http://www-306.ibm.com/software/integration/wmq/>

Communication with MQ services can be achieved in a number of ways and the technology requires a feature rich API and extensions in order to integrate with an Enterprise environment.

1.2 Technical Background

The main component of a WebSphere MQ instance is the Queue Manager. This is a process that manages the message queues and provides interfaces (known as channels) to the applications wishing to communicate with them. By default, a Queue Manager will listen on a network interface for incoming connections and process the data accordingly. A Queue Manager will accept a number of types of MQ data and will begin processing it before determining whether the packet is authorised, or has been received at the correct point within the application’s “state machine”.

The result of this fact is that a large amount of MQ code is exposed to unauthenticated users. Consequently, any vulnerabilities in the code used to parse the data after it has been passed from the network socket can potentially be exploited by an attacker.

After receiving data from the network socket the MQ application will process and parse this data in various ways. The exact nature of this parsing is not within the scope of this document; however, it is important to note that a large amount of this activity occurs before a connection to the Queue Manager is fully established.

In certain conditions the ‘TCPReceive’ function will read a value within the packet and use this as the length of a data structure. This value is then used when performing a ‘memmove’ operation later in the processing. This operation is used to move data from a location within a buffer containing the entire packet to the beginning of the buffer. The purpose of this is not currently understood; however, it may be to handle packet fragmentation or another type of operation.

Vulnerability Details

A vulnerability was identified such that an attacker could affect the length of the data used in the ‘memmove’ operation. During the ‘TCPReceive’ function a signed check is performed on the data held within the EAX and EDX registers. The EDX register contains data from the packet currently being processed and it is expected that the EAX register contains the length of the buffer assigned for the packet.

Within the TCPReceive function the length field in the packet is copied from the packet data byte by byte and stored in the EDX register as can be observed in the disassembly here: -

```
.text:4D465C5C      movzx  edx, byte ptr [esi+4]
.text:4D465C60      movzx  edi, byte ptr [esi+5]
.text:4D465C64      shl    edx, 8
.text:4D465C67      add    edx, edi
.text:4D465C69      movzx  edi, byte ptr [esi+6]
.text:4D465C6D      shl    edi, 8
.text:4D465C70      add    edx, edi
.text:4D465C72      movzx  edi, byte ptr [esi+7]
.text:4D465C76      shl    edi, 8
.text:4D465C79      add    edx, edi
```

The value in EDX (length of packet) is then compared to the length of the assigned buffer which is contained within the EAX register. The length of this buffer is copied to the EDI register before the jump instruction and is used later in the processing.

```
.text:4D465C7B      cmp    eax, edx
.text:4D465C7D      mov    [esp+1DCh+var 1C0], edx
.text:4D465C81      mov    edi, eax
.text:4D465C83      jnl   short loc_4D465C87
```

The compare and subsequent jump (cmp and jnl) instructions perform a signed check on the values in EAX and EDX. Therefore, if a value between half maximum integer and full maximum integer (0x7fffffff and 0xffffffff) is specified in the packet's payload the length will be interpreted as a signed value and therefore negative. As EDX is less than EAX the jump will therefore not occur and processing will continue.

If the jump is not taken the value in EDI will be overwritten with the length of the packet which is stored in EDX (this is intended to occur when the length of the data is less than the size of the buffer). The disassembly output of the relevant instructions is included here: -

```
.text:4D465C85      mov    edi, edx
```

By executing this instruction the subsequent memory move operation will use the actual length of data in the packet taken from the EDX register. However, the length of the data in EDI can now be greater than the size of the buffer due to the signed check which was performed.

Later in the TCPReceive function a call to the 'memmove' function is made which uses the values in the ECX, ESI and EDI registers. The ECX register contains a pointer to the beginning of the buffer containing the packet data, the ESI register contains a pointer to the section of the packet which is to be moved and the EDI register contains the length of data to be copied. The value in the EDI register is obtained from the data packet and therefore is controllable by an attacker. If the value in the EDI register is greater than the length of the buffer then data on the heap will be overwritten. The disassembly output of the relevant instructions is included here: -

```
.text:4D465D56      mov     ecx, [ebp+148h]
.text:4D465D5C      push   edi           ; size_t
.text:4D465D5D      push   esi           ; void *
.text:4D465D5E      push   ecx           ; void *
.text:4D465D5F      call   ds:memmove
```

However, as the packet data is written to an earlier location within the same buffer the overwrite operation will include the data already located after the end of this buffer and it is not trivial to control the contents of any of the data located in memory directly after this buffer. If this data is not valid, heap corruption will occur and the “amqrmppa” process which handles data from network connections will be terminated. The application handles this exception and restarts the process; however, this will affect any previously established connections.

Vulnerability Impact

An attacker capable of exploiting the vulnerability would be able to cause a DoS to established connections by causing the process to restart. If the condition continues to be triggered this can result in a DoS to all legitimate users. If a method were discovered by which the data after the packet could be controlled it could also be possible to gain execution within the process and thereby execute arbitrary code. The “amqrmppa” process runs with the privileges of the MQ user (this is dependent on the platform, for example, mqm on UNIX or MUSR_MQADMIN on Microsoft Windows). Therefore, in the circumstances described above an attacker would be able to execute arbitrary code with the privileges of that user.

1.3 Exploit Information

The vulnerability is located within the TCPReceive function and therefore a number of different vectors can be used to trigger the condition. Termination of the “amqrmppa” process has been demonstrated using custom crafted MQCONN and MQUSERID packets and other packet types are also expected to be valid for this attack.

At the present time, MWR InfoSecurity have not been able to produce working exploit code for this vulnerability (although the DoS is trivial to trigger); however, this may change if further time and effort is invested in the process. It is therefore important that this issue be addressed as an exploitable issue although the likelihood of exploit code being easily produced is probably low. It should also be noted that when taking the decision to publish any exploit code MWR InfoSecurity are mindful of their obligation to protect their customers and Critical National Infrastructure (CNI) whilst also enabling the security community to accurately assess the vulnerability of systems running affected software.

1.4 Dependencies

This vulnerability has been tested on WebSphere MQ version 6.0 on multiple Microsoft Windows and Linux platforms. However, as the vulnerability is present within the TCPReceive function of the MQ product it is expected it will be present in every version of the software.

2 Recommendations

It is recommended that all users install the appropriate security patches released by the vendor in response to this issue. Links to the updated software can be discovered at the following location: -

<http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg27006037>

Interim Workaround

Both network and host based traffic filtering controls could be implemented to protect access to the Queue Manager service on the affected hosts. This should be considered at both the packet filtering level and by the use of IPSec tunnels between hosts.

MWR InfoSecurity
St. Clement House
1-3 Alencon Link
Basingstoke, RG21 7SB
Tel: +44 (0)1256 300920
Fax: +44 (0)1256 844083
mwrinfosecurity.com