# Global Information Assurance Certification Paper

**Q & A with QAZ**
John Driggers
November 19, 2000

## Introduction

The purpose of this working paper is to describe in as great a detail as possible the QAZ Trojan. This will cover detection methods, infection process, prevention, and a lab test of this trojan.

The QAZ Trojan, also referred to as note.com, QAZ.worm, TROJ_QAZ.A, Trojan/Notepad, and W32.HLLW.Qaz.A was discovered in the wild in early July 2000. It was reported as a "medium risk" trojan originating in Asia. Original reports suggested emphasized the "Back Door" functionality of the virus more than anything else.

The virus did not make many waves in the press at the time - this all changed on October 27th, when a high profile target, Microsoft, announced that the QAZ trojan had been used to infiltrate their corporate network. The initial report indicated that the potential for catastrophic financial loss due to the compromise of their source code was possible.

This virus was first detected on the Corporate Network in early August by alert Systems Administrators who noticed that their central log servers were being inundated with failed login request from machines that would normally not be browsing their networks. This, combined with security conscious users with system level intrusion detection systems reporting NetBios scans, heightened the awareness that this was not a random happening. IT Security began investigating at that point. The processes, procedures, and analysis that grew from that investigation follow.

## Investigation / Detection

Initial reports on the QAZ trojan indicated that it was listening on TCP port 7597 for a "control client". Existing scan tools, based on Fyodor's NMAP were modified to scan the corporate network for machines listening on port 7597. This proved to be a rather challenging task as the network in question is composed of some 20 class b subnets geographically located in over 100 countries in all major timezones. The scripts were moderately successful in identifying infected machines after being modified to scan each region during their "daylight" hours.

There were two major false positives of note. This first involved Hewlett Packard JetDirect cards, the second was Silicon Graphics workstations running IRIX 6.x. It is possible to weed out these false positives by examining the connect string - a true QAZ infected machine will return a colon ":".

Current virus protection (McAfee) did not detect the trojan at the time of its release in the wild, but subsequent updates were successful. Infected machines found could be classified in one of three conditions:

1] No Virus protection install / Virus protection disabled/ Vastly out of date Virus protection.

2] Virus Protection installed / Engine out of date / DAT current

3] Virus Protection installed / Engine current / DAT out of date.

Information regarding the trojan was later published indicating that it was attempting to send mail to an IP address in Asia, 202.106.185.107. Gateway IDS systems were configured to "drop/log" all attempts to connect to this address. Attempts to connect to this address were never successful, but the decision to drop the packets was made in a proactive manner. This was the most effective method by far for detecting infected systems. It should be noted that this server is back online with respect to some networks, but does not appear to be accepting mail for the "nongmin_cn" account.

The general consensus at this point was that the outbreak was confined to our corporate network. This was based on the fact that our perimeter firewalls blocked incoming packets on port 7597 and incoming packets on 139, thus blocking all control and infection attempts. The fact that the virus was "network aware" was not considered an exposure risk at the time as all NT based networks were within our perimeter.

In an effort to verify these conclusions regarding exposure, the decision was made to closely examine one of the infected systems with regard to network traffic. An IDS system specifically targeted an infected machine, and captured all traffic from that system. Due to the location of the infected machine chosen, the "closest" IDS system turned out to be one of the perimeter systems, in this case at an Internet gateway. This was done simply because the infected system was one with little or no impact on day to day operations at the site in question. The decision to use a gateway IDS system turned out to be a stroke of luck - it immediately detected systematic scans on port 139 headed out to the Internet. The IDS was immediately reconfigured to detect this scan activity (TCP 139) in general, resulting in the immediate detection of other scans in progress.

This type of incident is very specifically covered in the corporate Incident Response Policy. In the case, the proper course of action was to immediately open an emergency change management ticket blocking this traffic at all Internet gateways.

This unpublished activity caused an immediate re-evaluation of this trojan. The decision was made to do a detailed analysis of the troian itself to assess its

capabilities.

**Forensic Examination**

A portion the security research lab was used to conduct the following tests.

- **Code Examination :** Published information indicated that the trojan consisted of a single file, "notepad.exe". It copied the "original" notepad.exe to a file called "note.com". Executing the application (trojan) Notepad caused the trojan code to execute, calling "note.com" as a shell escape to convince the user nothing was amiss.

  The tool Hex Workshop was chosen to examine the binary in detail. This combined with the Unix tool "strings" provided this information:

  1. Offset 00088608:Appeared to be compiled with Microsoft C++
  2. Offset 00090168 : Contained modules "osinfo.c", "aw_str.c", "dospawn.c", "cenvarg,c","wtombenv.c","wcstolbs.c", "chsize.c, "aw_cmp.c", and "setenv.c"
  3.Offset 00094276 : Contains the text "qazwsx.hsq", and what appeared to be a registry key "SOFTWARE\Microsoft\Windows\Current Version\Run startIE"
  4. Offset 00094536 : Contains other interesting entries, including references to "note.com" and "notepab.exe"
  5. Offset 00094605 : Contained the IP address of the system mentioned in the McAfee Report.
  6. Offset 00094630 : Appears to contain the mailing instructions, and the account name to mail to / from "nongmin_cn".
  7. Offset 00109752 : Beginning of what looks like the instruction sequence. This has been cleaned up a bit for readability and included it in Attachment A.

- **Research:** An attempt to locate more information regarding the trojan from various hacking and security sites proved rather fruitless. The most detailed information by far came from distributed.net. It appears that this trojan, or a variant of it, was being used to distribute their distributed computing client. Various anti-virus vendors in general provided the same information and basic cleaning / removal tools.

- **Lab Setup:** Four goals were outlined in the design and configuration of

the lab. These were :

1. Monitor all traffic from the infected machine.
2. Intercept the mail message the trojan sent.
3. Attempt to detect the propagation strategy of the virus.
4. Attempt to detect the exposure based on Microsoft NetBIOS sharing characteristics.

The lab also needed to protect the production environment from the dirty (infected) environment. A diagram of the setup is provided in Illustration A. It was further decided that we would allow the virus to propagate into a contained portion of the production network, using the IDS to block any attempts to actually modify a filesystem (blocking SMBmv and SMBtrans requests). This was a hotly discussed prospect - the decision to allow this "controlled burn" was made largely on the fact that there were infected machines on the network already, combined with test that indicated that the IDS would be able to accomplish the desired result. Initial test on a purely segregated network had indicated that the trojan appeared to "break" - the machine was rendered unusable (frozen) until the "notepad.exe" process was terminated.

To accomplish the above goals, the following equipment was used to create the lab environment.

1] Sun Ultra 10 with dual NIC's. This machine was configured to act as a packet sniffer and IDS monitoring system, and mail host for the destination address.
2] Cisco 2500. This router allowed us to segregate the "dirty" network from the "clean" network, and to allow for less "noise" from the packet sniffers point of view.
3] "Clean" Windows 95 PC to act as the infection agent.
4] "Clean" Windows NT PC to act as a willing (shared drive) host.

The portion of the production network that was chosen to be the test bed was segregated from the rest of the corporate network by confirming that traffic from the "dirty" network would not be allowed to egress the network at its perimeter. This was accomplished by the use of egress filters on the primary site router.

The Sun workstation was configured to with the IP address found in the trojan binary, and an email account was created with the login information also detected in the binary.

- **Observations**: The Windows 95 machine was booted (clean install of Windows 95, no trojan). A domain user was logged into the system. The

trojan file was then executed on the machine. The registry key noted in the binary was written to the system, and the "note.com" was created.

Almost immediately, NetBIOS name request were observed. The trojan then proceeded to walk down the list of domains in the "Network Neighborhood" in alphanumeric order. In each instance, then queries were terminated at the border router, and the trojan moved on to the next name in the list. This continued until the "production / test" domain was reached. At this point the name query was enumerated, and the trojan began to attempt to connect to the machines returned, again in alphanumeric order.

Example of failed enumeration:

905 0.01301 infested.machine -> pdc.machine NBT NS Query Request for
ANADRILL[1b], Success
907 0.00061 blocking.router -> infested.machine ICMP Destination unreachable
(Communication administratively prohibited)

As each machine was queried, all of its shares were enumerated, and a listing of all files in the root of each share was returned. If a share contained a directory named "Windows" or "WINNT", the files "notepad.exe" & "note.com" were enumerated. If the "note.com" file did not exist, the trojan then attempted (and succeeded if the file/share permissions allowed) to move the original "notepad.exe" to "note.com", and then copy itself over as "notepad.exe". It is ***very important*** to note that the connection attempts were made with the user account that was logged into the infected machine. This has very bad implications if the infected machine is logged in as a domain administrator in an NT domain for obvious reasons as "$" shares were also enumerated.

Example of Successful Enumeration with file and share information - Note the session setup username.

1 0.00000 infested machine -> targeted.machine NBT C port=1033
2 0.00153 targeted.machine -> infested machineNBT R port=1033
3 0.00077 infested machine -> targeted.machine NBT C port=1033
4 0.00028 infested machine -> targeted.machine NBT Type=SESSION
REQUEST Dest=7KEYS-APC[20] Source=JDRIGGERS-APC[0] Length=68
5 0.00064 targeted.machine -> infested machineNBT Type=POSITIVE

SESSION RESPONSE Length=0
6 0.00118 infested machine -> targeted.machine SMB C
Code=0x72
Name=SMBnegprot LastDialect=NT LM 0.12 Error=0
7 0.00098 targeted.machine -> infested machineSMB R
Code=0x72
Name=SMBnegprot Dialect#=5 Error=0
8 0.00191 infested machine -> targeted.machine SMB C
Code=0x73
Name=SMBsesssetupX Username=DRIGGERS Error=0
9 0.02591 targeted.machine -> infested machineSMB R
Code=0x73
Name=SMBsesssetupX Error=0
10 0.00123 infested machine -> targeted.machine SMB C
Code=0x25
Name=SMBtrans Error=0
11 0.00238 targeted.machine -> infested machineSMB R
Code=0x25
Name=SMBtrans Error=0
12 0.00288 infested machine -> targeted.machine SMB C
Code=0x75
Name=SMBtconX Share=\\TARGET NETBIOSNAME\CD Error=0
13 0.00074 targeted.machine -> infested machineSMB R
Code=0x75
Name=SMBtconX Type=A: Error=0
14 0.00103 infested machine -> targeted.machine SMB C
Code=0x8
Name=SMBgetatr FileName= Error=0
15 0.00232 targeted.machine -> infested machineSMB R
Code=0x8
Name=SMBgetatr Error=15
16 0.09106 infested machine -> targeted.machine SMB C
Code=0x75
Name=SMBtconX Share=\\TARGET NETBIOSNAME\PUBLIC
Error=0
17 0.00086 targeted.machine -> infested machineSMB R
Code=0x75
Name=SMBtconX Type=A: Error=0
18 0.00097 infested machine -> targeted.machine SMB C
Code=0x8
Name=SMBgetatr FileName= Error=0
19 0.00069 targeted.machine -> infested machineSMB R
Code=0x8
Name=SMBgetatr Error=0
20 0.00131 infested machine -> targeted.machine SMB C
Code=0x32

```
  Name=SMBtrans2 Findfirst File=\* Error=0
21 0.00200 targeted.machine -> infested machineSMB R
Code=0x32
  Name=SMBtrans2 Error=0
22 0.00157 infested machine -> targeted.machine SMB C
Code=0x8
  Name=SMBgetatr FileName= Error=0
23 0.00067 targeted.machine -> infested machineSMB R
Code=0x8
  Name=SMBgetatr Error=0
24 0.00109 infested machine -> targeted.machine SMB C
Code=0x8
  Name=SMBgetatr FileName= Error=0
25 0.00066 targeted.machine -> infested machineSMB R
Code=0x8
  Name=SMBgetatr Error=0
26 0.00114 infested machine -> targeted.machine SMB C
Code=0x8
  Name=SMBgetatr FileName= Error=0
27 0.00071 targeted.machine -> infested machineSMB R
Code=0x8
  Name=SMBgetatr Error=0
28 0.00109 infested machine -> targeted.machine SMB C
Code=0x8
  Name=SMBgetatr FileName= Error=0
29 0.00069 targeted.machine -> infested machineSMB R
Code=0x8
  Name=SMBgetatr Error=0
30 0.00118 infested machine -> targeted.machine SMB C
Code=0x8
  Name=SMBgetatr FileName=\ARGS.EXE Error=0
31 0.00075 targeted.machine -> infested machineSMB R
Code=0x8
  Name=SMBgetatr Error=0
32 0.00121 infested machine -> targeted.machine SMB C
Code=0x8
  Name=SMBgetatr FileName=\COMSHELL.EXE Error=0
```

At seemingly random intervals, the infected machine connected to the mailserver and sent a message to the "nonmin_cn" user. These intervals seemed to be triggered by two events :

1] At the end of an IP based scan of a class "c" network.
2] When a machine was successfully infected.

Messages intercepted fell into one of three categories.

1] Messages timed out during the data phase of the SMTP session.
2] Messages that successfully transmitted a "blank" message.
3] Messages that reported the IP of an infected machine.

Example of message sent to target IP with no body :

```
From nongmin_cn Tue Oct 31 23:38:51 2000
Received: from infested.machine (infested.machine [10.0.0.1])
by SUN.mailhost (8.9.3+Sun/8.9.3) with SMTP id XAA01642
for nongmin_cn; Tue, 31 Oct 2000 23:38:51 -0600 (CST)
Date: Tue, 31 Oct 2000 23:38:51 -0600 (CST)
From: virus test <nongmin_cn>
Message-Id: <200011010538.XAA01642@SUN.mailhost>
X-Authentication-Warning: SUN.mailhost: infested.machine
[10.0.0.1]
  didn't use HELO protocol
Content-Length: 0
```

Example of message sent to target IP with IP of a newly infected machine. :

```
From nongmin_cn Wed Nov 8 12:43:12 2000
Received: from infested.machine (infested [XXX.XXX.XXX.48])
by SUN.mailhost (8.9.3+Sun/8.9.3) with SMTP id MAA15288
for nongmin_cn; Wed, 8 Nov 2000 12:43:11 -0600 (CST)
Date: Wed, 8 Nov 2000 12:43:11 -0600 (CST)
From: virus test <nongmin_cn>
Message-Id: <200011081843.MAA15288@SUN.mailhost>
X-Authentication-Warning: SUN.mailhost: infested.machine
  [XXX.XXX.XXX.48]
  didn't use HELO protocol
Content-Length: 18
\\XXX.XXX.XXX.151
```

After the machines in the "control" domain had been exhausted, the trojan proceeded to attempt to enumerate the remaining domains. In this test, these attempts failed due to the site routers ACLs.

When the list of domains had been exhausted, the trojan then exhibited the activity first noticed by the perimeter IDS systems - it started to scan for shares by IP address. These address were sequential, starting at the first IP address range outside of the

infected systems subnet mask, and then proceeding upward incrementing an octet at a time. It also appears that there is at least some randomization of the networks scanned - due to the size of the packet dumps, it will take some time to correlate this activity.

It was also observed that upon reaching the end of a block (.254), the trojan sent an email message to the mailhost. The message had an empty body – the supposition is that it is simply a counter and re-notification source for infected machines.

**Recommendations**

1] Detection for the current variant is made easy with the use of perimeter IDS systems. Trap and log traffic directed to 202.106.185.107. Note that the machine at this address does not accept mail for the "nongmin_cn" at this time. If NetBIOS is not allowed outside of the network perimeter, monitor for outgoing connection attempts with varied destination and unique sources.

2] If perimeter IDS is not possible (router ACL's etc), scanning for machines listening on port 7595 is a viable method of detection. Avoid false positives by testing for the colon response to the connection attempt.

3] Utilize centralized log systems to track failed connection attempts to Windows hosts. If personal firewalls are present on the network, make sure users have method, means, and knowledge of the IRT processes.

4] Current virus protection is a must. Most anti-virus vendors are supplying a "stand alone" cleaner to remove the registry entries and rename the "note.com" file.

5] Keep in mind that there are reported to be both a "control client" and configuration utility for this trojan. If the trojan is configured to "listen" on a different port, or send mail to a different server, it will become more difficult to detect. A HexDump utility or Unix "strings" function should be used to attempt to locate these modifications.

**Conclusions**

In the post mortem evaluation, the following observations were made.

1] Centralized log servers and System Administrator security awareness training was very worthwhile. The outbreak was

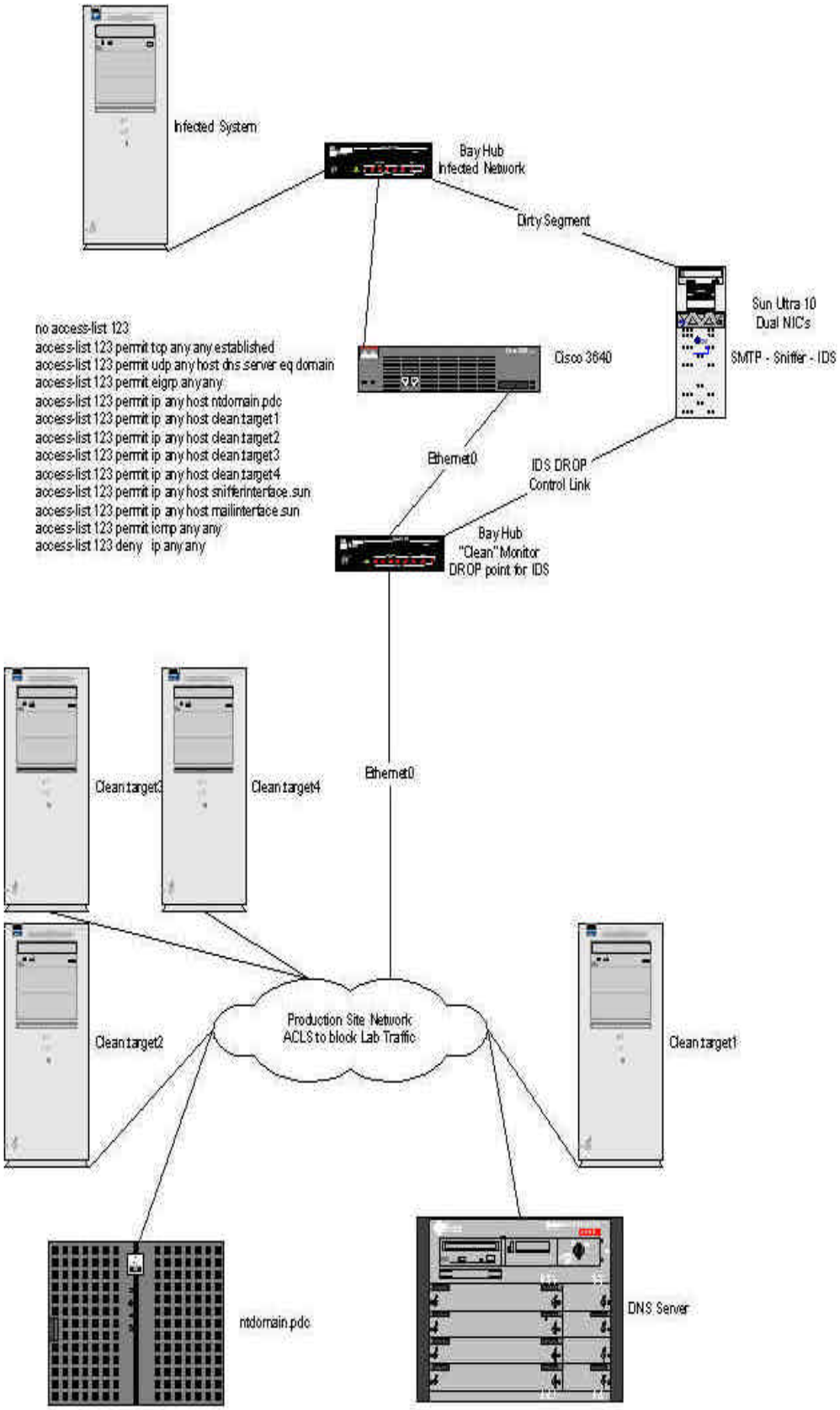
detected early due to the diligence of these "carbon based IDS systems". Follow-up recommendation is to continue user awareness training; evaluate PC based "personal" firewall products with eventual corporate wide deployment. "ATTA-Boys" to the System Administrators who noticed the trojan.

2] Initial reports of this virus tended to minimize its importance. Follow-up recommendation is treat all virus / trojan detects as a major incident and to analyze the code in detail.

3] Emergency Change Management request on the perimeter firewalls functioned properly, and well within the Service Level Agreement parameters.

4] Incident Response Plan functioned properly and IRT reacted in a timely manner.

Illustration A:




 Attachment A: Strings output of apparent code sequence. 

Attachment A: Strings output of apparent code sequence.

```
CreateThread
CloseHandle
GetLastError
CreateSemaphoreA
GetCommandLineA
GetModuleFileNameA
ExitThread
Sleep
GetFileAttributesA
CopyFileA
WriteFile
CreateFileA
KERNEL32.dll
LoadIconA
USER32.dll
WNetCloseEnum
WNetEnumResourceA
WNetOpenEnumA
MPR.dll
timeBeginPeriod
timeEndPeriod
timeKillEvent
timeSetEvent
WINMM.dll
WSOCK32.dll
RegCloseKey
RegSetValueExA
RegCreateKeyA
RegQueryValueExA
ADVAPI32.dll
ShellExecuteA
SHELL32.dll
ExitProcess
TerminateProcess
GetCurrentProcess
FindFirstFileA
FindNextFileA
FindClose
FileTimeToSystemTime
FileTimeToLocalFileTime
SetEnvironmentVariableA
GetCurrentDirectoryA
SetCurrentDirectoryA
```

MoveFileA
GetModuleHandleA
GetStartupInfoA
GetVersion
IsBadWritePtr
IsBadReadPtr
HeapValidate
DebugBreak
GetStdHandle
InterlockedDecrement
OutputDebugStringA
GetProcAddress
LoadLibraryA
InterlockedIncrement
UnhandledExceptionFilter
FreeEnvironmentStringsA
MultiByteToWideChar
FreeEnvironmentStringsW
GetEnvironmentStrings
GetEnvironmentStringsW
WideCharToMultiByte
GetCPInfo
GetACP
GetOEMCP
SetHandleCount
GetFileType
HeapDestroy
HeapCreate
VirtualFree
RtlUnwind
HeapAlloc
HeapReAlloc
HeapFree
VirtualAlloc
SetFilePointer
SetConsoleCtrlHandler
GetTimeZoneInformation
LCMapStringA
LCMapStringW
SetStdHandle
FlushFileBuffers
GetStringTypeA
GetStringTypeW
GetExitCodeProcess
WaitForSingleObject
CreateProcessA

SetEndOfFile
ReadFile
CompareStringA
CompareStringW