Digital Investigations case study 1 – "Listening for the Grasshopper"

Alec Waters, Dataline Software – alec.waters@dataline.co.uk

In the exciting world of corporate security, you're sometimes blessed with a degree of control and predictability. After all, you built the infrastructure yourself - you know what it's designed to do, you know what kind of traffic flows where, you know its weak points, and you designed the whole lot to be as defensible as possible. Theoretically you should be able to fight your battles on your own terms.

Deploying signature-based security systems like Intrusion Detection Systems (IDS) is, of course, sensible. As with everything in life, nothing is perfect, these included – they require a management effort to keep them up to date, they usually only detect "known badness", and if they're deployed in the wrong place they're of no use whatsoever. They also make no record of anything that they consider benign, which can often leave you with a large gap in your visibility and a false sense of security. It's tempting to think that if the IDS isn't saying anything, then nothing bad is going on, right...?

To supplement your investment in products such as IDS, you may be able to obtain significant intelligence from equipment you've already got. A good example is the use of Netflow (or sflow, or ipfix) exports from your own routers. Netflow exports can be likened to an itemised telephone bill, showing who spoke to who, at what times, and for how long, expressed in terms of IP addresses, protocols and ports. It's a completely passive technique, and doesn't involve installing any software on the machines you want to monitor. You can read more about it here:

http://www.cisco.com/go/netflow

This document tells the tale of how this kind of passive monitoring detected suspicious activity from a public facing web server, and of how the network itself was used to contain and investigate the incident.

The story you are about to read is true. Only the IP addresses and domain names have been changed to protect the innocent.

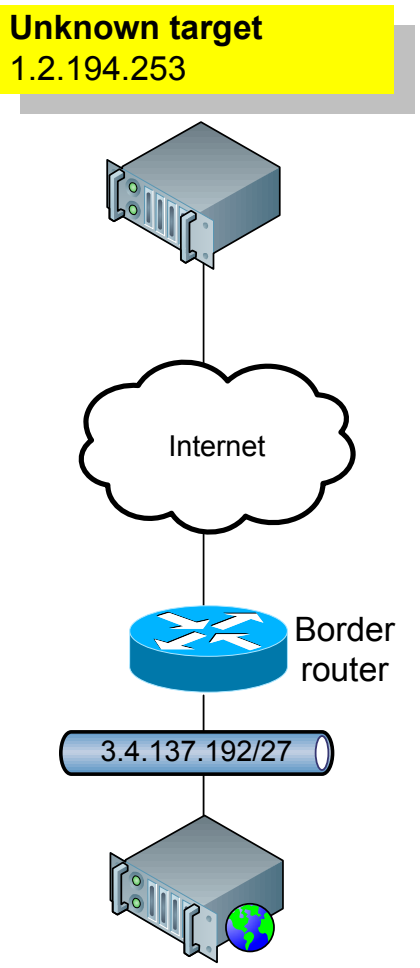
The Mystery

QQSecure is a fictitious web hosting outfit, operating a farm of public web servers that serve up their customers' websites to the masses. Whilst a security analyst should clearly scrutinise all the inbound traffic aimed *at* a web server farm, the outbound traffic *from* a web server should be examined too.

The catalyst for this investigation was a set of daily Netflow-based reports that show internet-bound activity from QQSecure's public web servers. The idea of the reports is to describe all of the activity that these servers have initiated *themselves*, as opposed to the traffic generated by a server doing its intended work (such as serving up web pages, for example).

Typically, these reports show HTTP requests directed at Windows Update, which is the expected behaviour – the servers are making automated checks for updated software. One day however, the reports showed some unexpected traffic from one of the monitored web servers (which we shall call "server1.qqsecure.co.uk", or just "server1" for brevity).

The reports showed that server1 had initiated an outbound TCP connection to port 6101 on a machine at IP address 1.2.192.253 (henceforth referred to as the "unknown target"). It wasn't much – just three outbound SYNs (connection requests) that were answered immediately by RSTs (connection rejections). This means that server1 had tried three times to connect to the unknown target on port 6101, but each time the unknown target had rejected the attempt. Furthermore, the report showed that this pattern was repeated every four hours, and there was no reason to believe it would stop any time soon.



The diagram above, which we shall build on, shows server1 and the unknown target that it is resolutely trying to contact.

We now have a mystery on our hands - server1 has started behaving in an unexpected manner for reasons unknown. Perhaps it has been attacked and compromised and is now part of an evil botnet, "phoning home" to the attackers for instructions or additional malicious software. Alternatively, it could be something much more benign; in either case, it can't be explained away, and the cause must be understood!

The Plan

At this point, we need to decide what to do. An organisation's pre-prepared Incident Response (IR) plan might list some options for scenarios like this:

- Option 1 might be to assume that the server is compromised and rebuild it from scratch. However, if we did that we'd never know what was causing the issue, nor would we be able to understand how the assumed compromise had happened in the first place.
- Option 2 could be to obtain and examine forensic duplicates of the server's disks and the contents of its memory. These can be examined for traces of compromise, but this process is time consuming and will involve downtime for the machine in question.
- The final option is to leave the server alone, and try to understand the incident by observing the suspect network traffic. The risk here is that server1 will eventually establish a successful connection to the target (or it will connect to a different target) and something Bad will happen. Should definite malicious activity be observed, the IR plan might dictate disconnecting server1 from the network and proceeding with Option 2 and eventually Option 1 to restore service.

The third option was chosen, as in this case the associated risks were deemed acceptable for the following reasons:

- No actual data had actually left server1, despite repeated attempts to connect to the unknown target (i.e., server1 was not (yet!) the victim of a data breach)
- Even if data was taken from server1 at a later time, none of it was in any way sensitive (server1 held no credit card details, no personally identifying information, etc.)
- Server1 was contained by firewalls and could not be used as a stepping stone to compromise other local machines at QQSecure
- Server1 could be rebuilt from known-good sources even if something happened to cause it to become a total loss.

We therefore left the server alone and opted for a "hands off" approach – we would learn as much as we could from the network itself without resorting to logging into the server and digging around. We certainly wouldn't run any anti-virus scans at this point – they touch the last-accessed time of every file on the machine, thereby destroying part of a file system chronology that could be created forensically later on.

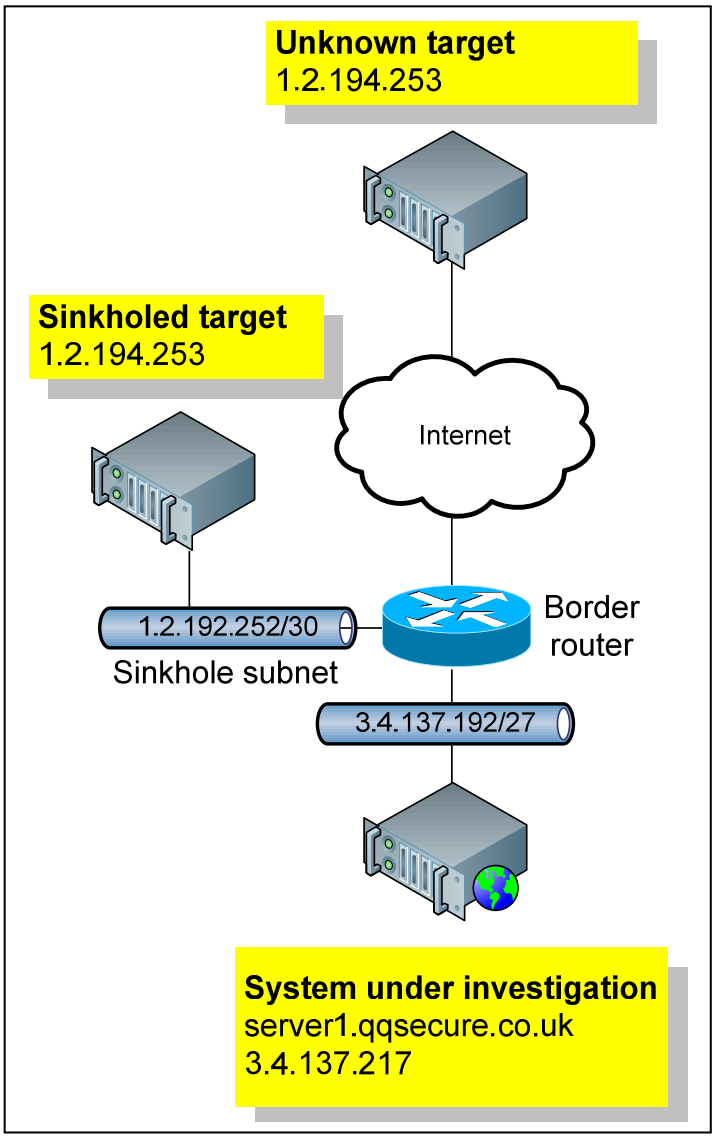
The Plan in Action

The next four-hour interval was imminent, so the level of information gathering was stepped up. Instead of reading the telephone-bill-style Netflow exports, we would capture the traffic we're interested in straight off the wire (to continue the telephone metaphor, this is the equivalent of recording a phone call to tape). The traffic capture was performed by setting up a SPAN port on the network switch that server1 connects to. The SPAN port creates a duplicate of all the traffic to and from server1 which we can capture with an analysis workstation, thereby avoiding the need to log into server1 and run the traffic capture there.

However, the traffic capture didn't show us much more than Netflow hadn't already told us – there were three connection attempts, each of which was rejected by the unknown target. There was no evidence of any sneaky covert channel passing information in the SYN packets, so we had four hours to determine a course of action before the next cycle of activity.

Time to research what few facts we had:

- The unknown target at 1.2.192.253 was a shared server at a hosting centre in the US – just about anything could have been listening for connections there, with or without the consent of the hosting centre's owners or customers.
- Port 6101 seems to be used by the Veritas/Symantec remote agent for Backup Exec, older versions of which have definite security issues.



The above software was installed and running on the server, although the version present was fully patched and not supposed to be vulnerable to any documented security issues – nevertheless, perhaps someone had managed to exploit a different, as-yet undiscovered flaw in it?

The Sting

We're desperately short on information here – the ideal thing would be to capture the traffic that server1 was *trying* to send to the unknown target, provided that the target could be persuaded to accept the connection in the first place. Interacting in any way with the hosting centre in the US didn't seem feasible, so it was time to invoke a little lateral thought. If we can't get to the target, perhaps we can bring the target to us by altering the configuration of the border router in the diagram. What we want to do is convince the router that the target IP address is physically at QQSecure, rather than halfway across the world. We can accomplish this with the following commands on the border router:

```
router#config terminal
router(config)#int vlan 666
router(config-if)#ip address 1.2.192.254 255.255.255.252
router(config-if)#no shut
```

The net result is what's called a 'sinkhole', and is shown in the diagram above and explained below.

With the sinkhole in place, from the point of view of the border router the IP addresses from 1.2.192.252 to 1.2.192.255 are now at QQSecure's location, not at the US hosting centre – traffic sent to these addresses from server1 will not go over the Internet, but will instead stay local. Now we can simulate the unknown target by putting a PC into this network with the target's IP address, 1.2.194.253.

Next, we need to get something on our fake target that will listen on port 6101 and accept a connection. netcat is the tool of choice here, so we set it listening and redirecting anything received to a file:

```
c:\> nc -l -p 6101 > traffic.bin
```

By doing all of this, the connections from server1 will complete successfully and if any data is transferred we'll be able to see what it is. Finally, we set up another traffic capture to record the entire conversation, and the trap is set. We watch the clock. Tick followed tock followed tick followed tock....

The Payout

Bingo! Right on time, server1 pipes up and the trap is sprung; it has connected to the fake target machine in the sinkhole, and we have captured some packets.

The conversation was straightforward:

- Server1 opened up a TCP connection to the fake target with the usual three-way handshake
- Server1 sent one packet of data to the fake target
- The connection was torn down gracefully with FINs.

Now, what's in the data packet? Ignoring the IP and TCP headers, the actual packet payload looked like this:

```
0000 be 01 00 41 00 00 00 9f 00 0c 00 09 00 1a 52 4e ...A..........RN
0010 73 65 72 76 65 72 31 2e 71 71 73 65 63 75 72 65 server1.qqsecure
0020 2e 63 6f 2e 75 6b 00 0d 52 54 00 00 00 00 46 ca .co.uk..RT....F.
0030 ea f4 00 00 0b 49 34 03 04 89 d9 27 10 1b 00 0b .....I4....'....
0040 49 34 03 04 89 d8 27 10 1b 00 0b 49 34 03 04 89 I4....'....I4...
0050 cf 27 10 1b 00 0b 49 34 03 04 89 d4 27 10 1b 00 .'....I4....'...
0060 14 52 50 00 00 01 f4 00 02 90 03 00 00 00 05 00 .RP.............
0070 00 00 00 00 14 52 44 00 00 00 07 00 00 00 02 00 .....RD.........
0080 10 c0 82 00 00 02 80 00 18 52 4d 00 00 00 0b 00 .........RM.....
0090 00 00 00 00 00 18 5b 00 00 00 00 00 00 00 f0     ......[........
```

Most of it is total gibberish at first glance, but there are plenty of clues there if we look hard enough. We can see:

- server1.qqsecure.co.uk - this is the DNS name of the server
- Four occurrences of 'I4'. Looking at these and the following four bytes in hex we see:

```
49 34 03 04 89 d9
49 34 03 04 89 d8
49 34 03 04 89 cf
49 34 03 04 89 d4
```

Translating this to ASCII, we can see some IP addresses:

```
I4 3.4.137.217
I4 3.4.137.216
I4 3.4.137.207
I4 3.4.137.212
```

These are the four IP addresses assigned to server1 (I4 could mean "four byte IP address" or "IPv4 address"). We couldn't make much sense of the rest, but seeing a server export its contact details for reasons unknown is deeply concerning.

Just the facts, ma'am

To recap what we've found out so far:

- Server1 is trying to communicate with an unknown target
- The communication attempts use a port known to be used by the Backup Exec remote agent
- The Backup Exec remote agent is installed on server1
- Earlier versions of this software have had security issues
- Having tricked server1 into connecting to a fake target, we can see it exporting its contact details

The situation isn't looking very good at this point.

Digging deeper

Having captured the conversation between server1 and the fake unknown target, it's finally time to log into the server and have a gentle nose around. The first thing to do is to make sure that the version of the Backup Exec remote agent software isn't affected by the documented vulnerabilities; we're in the clear here, and checksums of the binaries involved correlate with those of known-good files.

The server admin says that nobody has logged into server1 for a long time, but the timestamps on the Backup Exec files are recent. This jogs the server admin's memory – the Backup Exec remote agent *was* recently upgraded on this machine as a side effect of upgrading the Backup Exec server elsewhere (it all happened automatically without the need for the server admin to log in). The backups of server1 had been working properly, so nobody had cause to check out the new version of the remote agent, which now sports a GUI (the old version didn't have one). Time to take a look...

There are several tabs in the GUI, one of which is labelled "Publishing" – the new remote agent can publish information about itself to Backup Exec Media Servers. This option was turned on, and QQSecure's backup server called qqbackup is listed as a destination. Could this "publishing" option be the cause of our mystery traffic? It's certainly a fit with the port number in use.

The trouble is though, qqbackup 's IP address isn't 1.2.194.253. If our mystery traffic is indeed caused by the Remote Agent doing a "publish", why isn't the traffic headed to qqbackup? Why is it going to the unknown target at the hosting centre in the US?

The Answer

If server1 is trying to talk to qqbackup, the first thing it has to do is convert the computer name "qqbackup" into an IP address by means of a DNS query. Given an incomplete DNS name like this, Windows will append a series of domain name suffixes to it and will try a DNS lookup for each. You can see what the list contains on your own computer by opening up a command prompt window and typing:

```
C:\>ipconfig /all
```

The top part of the output will look something like this:

```
Windows IP Configuration

        Host Name . . . . . . . . . . . . : server1
        Primary Dns Suffix  . . . . . . . : qqsecure.co.uk
        Node Type . . . . . . . . . . . . : Unknown
        IP Routing Enabled. . . . . . . . : No
        WINS Proxy Enabled. . . . . . . . : No
        DNS Suffix Search List. . . . . . : qqsecure.co.uk
                                            co.uk
```

Server1's list of suffixes started with qqsecure.co.uk, so it tried to lookup qqbackup.qqsecure.co.uk – this failed, as qqbackup's IP address can only be looked up on a private DNS server that server1 can't talk to.

Moving down the list, the next suffix (put there automatically by Windows) was just co.uk – server1 therefore does a DNS lookup for qqbackup.co.uk which comes back as 1.2.194.253, the IP address of our unknown target!

Quirks

Having co.uk in the suffix list is a quirk of the Windows DNS client, which builds the list from the primary DNS suffix you specify when you configure the machine. If you're a .com, everything works fine – for example, if your machine name is myserver.salesdept.eastcoast.mycompany.com your suffix list will look like this:

```
salesdept.eastcoast.mycompany.com
eastcoast.mycompany.com
mycompany.com
```

This is fine – all of these DNS suffixes are under the control of mycompany.com. However, if it were a co.uk instead, it would look like this:

```
salesdept.eastcoast.mycompany.co.uk
eastcoast.mycompany.co.uk
mycompany.co.uk
co.uk
```

If the co.uk suffix is used (because all the others haven't turned up any results), you could end up doing a DNS lookup for something that is outside of mycompany.co.uk's control, as has happened in this investigation – qqbackup.co.uk is nothing to do with qqsecure.co.uk! This could well result in an inadvertent leak of information, or worse.

The Fix

To resolve the issue:

- The publishing facility within the remote agent on server1 was disabled, as it wasn't needed
- The redundant DNS suffix of co.uk was removed.

After these steps were taken, server1 stopped trying to contact 1.2.194.253.

In Summary

- Server1 had a new Backup Exec Remote Agent installed on it by virtue of an upgrade to the Backup Exec server running on a machine called qqbackup
- This new remote agent software tried to "publish" information about server1 to qqbackup
- Server1 did a DNS lookup for qqbackup and, through a quirk in the way Windows builds the list of DNS suffixes, ends up thinking that qqbackup's IP address is 1.2.194.253
- At four-hour intervals, server1 tried to contact 1.2.194.253, believing it to be the backup server running on qqbackup
- 1.2.194.253 rejects these connection attempts
- Netflow activity for these connection attempts is collected, and appears on a daily report
- The investigation begins!

What did we learn?

- Looking for unexpected traffic patterns *will* bear fruit. Develop the means to understand what traffic your network carries in as much detail as possible. You can only detect abnormal traffic if you have a solid understanding of what is normal
- You can learn a lot from listening to the network; you don't always need to touch a suspect host to understand what's going on
- Sink holing can be used to move parts of the Internet around so you can take a better look
- Understand fully the software you put on your machines, or you'll end up with an unnecessary security incident. The new version of the Backup Exec Remote Agent did something that the previous version didn't – knowing about the publishing facility would have made the investigation a good deal shorter
- Documented change control of servers is important. Server1's admin had assumed that there had been no recent changes to the server because nobody had physically gone to it and run an installer - the updated Backup Exec Remote Agent had instead been "pushed" out by the backup server
- There needs to be good lines of communication between the server admin team and the security team. If the security team knew about the upgrade, they'd have seen that the traffic to the unknown target only started after the upgrade had taken place and would have had a better idea of where to start looking
- Egress filtering should be in place for servers like server1 that have a well-defined role. This machine has no legitimate need to use port 6101, so why allow it at all?

Finally…

Here's a popular quote from the TV series "Kung Fu":

Master Po: Close your eyes. What do you hear?
Young Caine: I hear the water, I hear the birds.
Master Po: Do you hear your own heartbeat?
Young Caine: No.
Master Po: Do you hear the grasshopper that is at your feet?
Young Caine: Old man, how is it that you hear these things?
Master Po: Young man, how is it that you do not?

Start listening for the grasshoppers in your own network!