

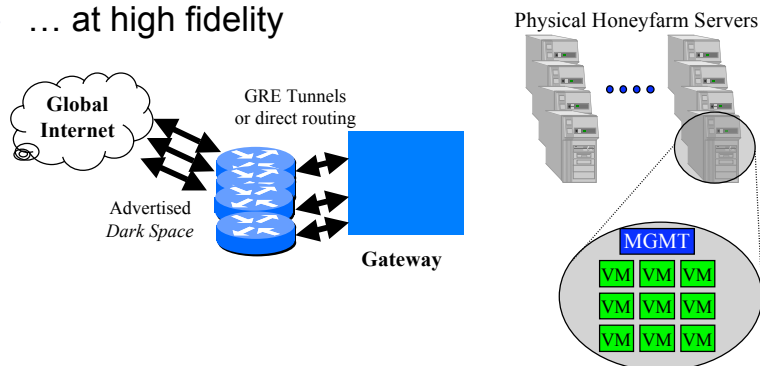
```
#!/usr/bin/perl
while (<>) {
    chomp;
    if ( /^^(get|post|options|head|...)(.*)/i ) {
        # Do not respond if it looks like an exploit
        last if length > 1000;

        my $date = gmtime;
        if ( $1 =~ /get|head/i )
            print "HTTP/1.1 200 OK\r\n";
        elsif ( $1 =~ /search/i )
            print "HTTP/1.1 411 Length Required\r\n";
        elsif ( $1 =~ /options/i ) {
            print "HTTP/1.1 200 OK\r\n";
            print "DASL: \r\nDAV: 1, 2\r\n";
            print "Public: OPTIONS, TRACE, GET, HEAD, DELETE, ... \r\n";
            print "Allow: OPTIONS, TRACE, GET, HEAD, DELETE, ... \r\n";
        }
        elsif ( $1 =~ /propfind/i )
            print "HTTP/1.1 207 Multi-Status\r\n";
        else
            print "HTTP/1.1 405 Method Not Allowed\r\n";
        }
    print <<EOF;
Server: Microsoft-IIS/5.0
Date: $date GMT
Content-Length: 0
Content-Type: text/html
Set-Cookie: ASPSESSIONIDACBAABCQ=BHAMAEOIAIHOMGJCPFLBGO; path=/
Cache-control: private

EOF
```

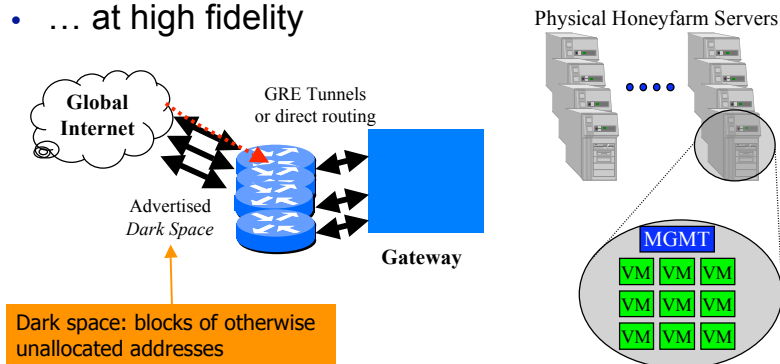
GQ: Building a Large-Scale *Honeyfarm*

- *Honeyfarm*: use a network telescope to route scan traffic to a set of honeypots
- Goal: scale to 100,000s of monitored addresses ...
- ... at high fidelity



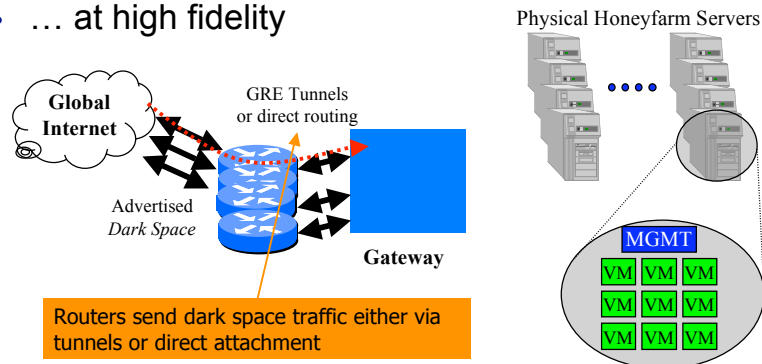
GQ: Building a Large-Scale *Honeyfarm*

- *Honeyfarm*: use a network telescope to route scan traffic to a set of honeypots
- Goal: scale to 100,000s of monitored addresses ...
- ... at high fidelity



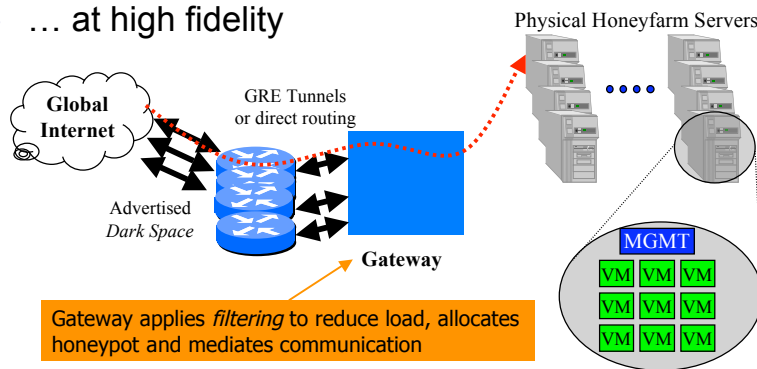
GQ: Building a Large-Scale *Honeyfarm*

- *Honeyfarm*: use a network telescope to route scan traffic to a set of honeypots
- Goal: scale to 100,000s of monitored addresses ...
- ... at high fidelity



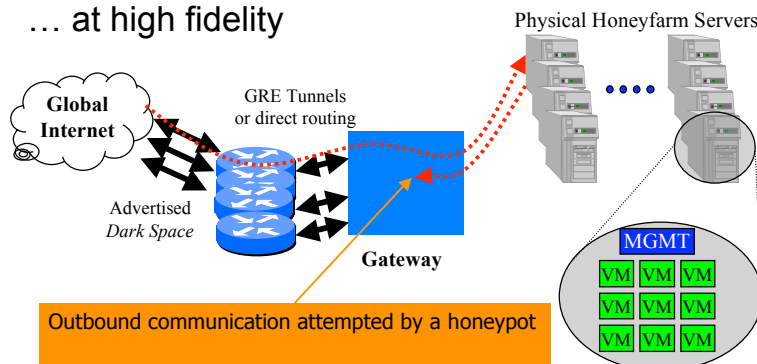
GQ: Building a Large-Scale *Honeyfarm*

- *Honeyfarm*: use a network telescope to route scan traffic to a set of honeypots
- Goal: scale to 100,000s of monitored addresses ...
- ... at high fidelity



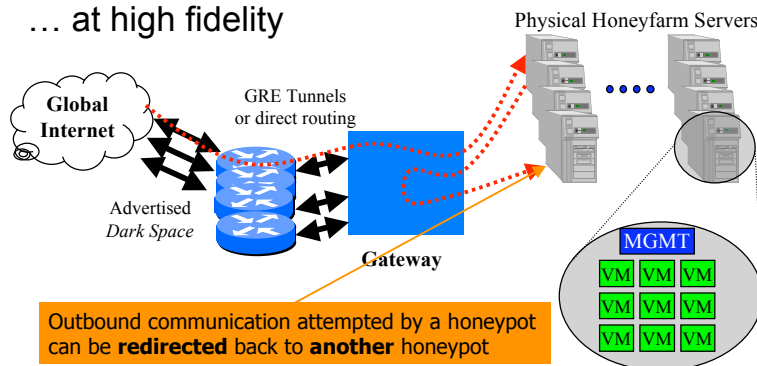
GQ: Building a Large-Scale *Honeyfarm*

- *Honeyfarm*: use a network telescope to route scan traffic to a set of honeypots
- Goal: scale to 100,000s of monitored addresses ...
- ... at high fidelity



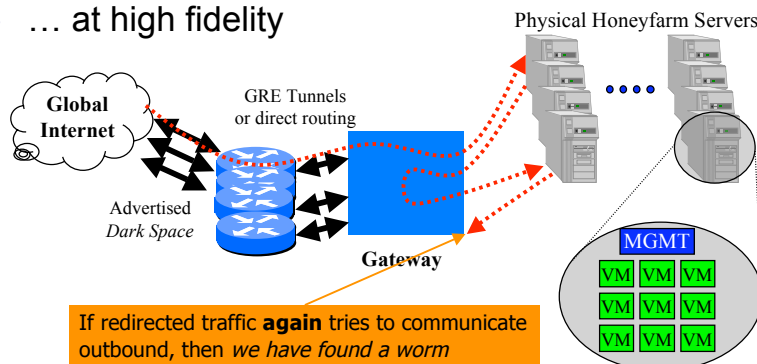
GQ: Building a Large-Scale *Honeyfarm*

- *Honeyfarm*: use a network telescope to route scan traffic to a set of honeypots
- Goal: scale to 100,000s of monitored addresses ...
- ... at high fidelity

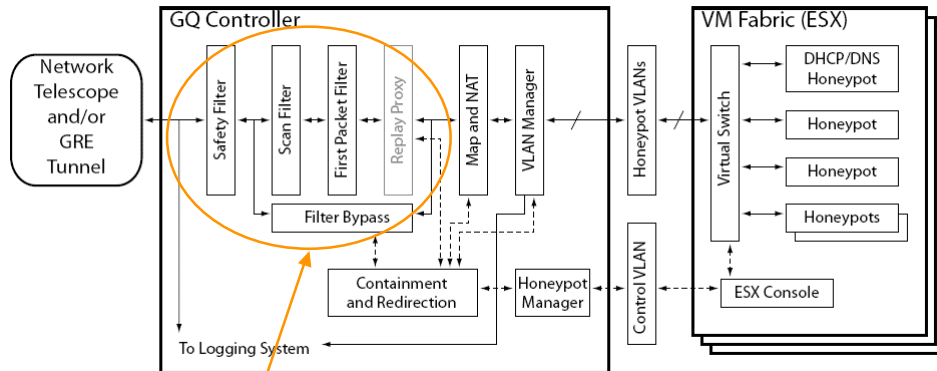


GQ: Building a Large-Scale *Honeyfarm*

- *Honeyfarm*: use a network telescope to route scan traffic to a set of honeypots
- Goal: scale to 100,000s of monitored addresses ...
- ... at high fidelity



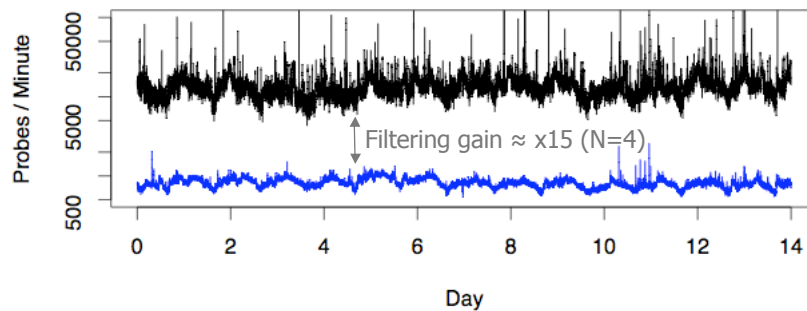
GQ Architecture



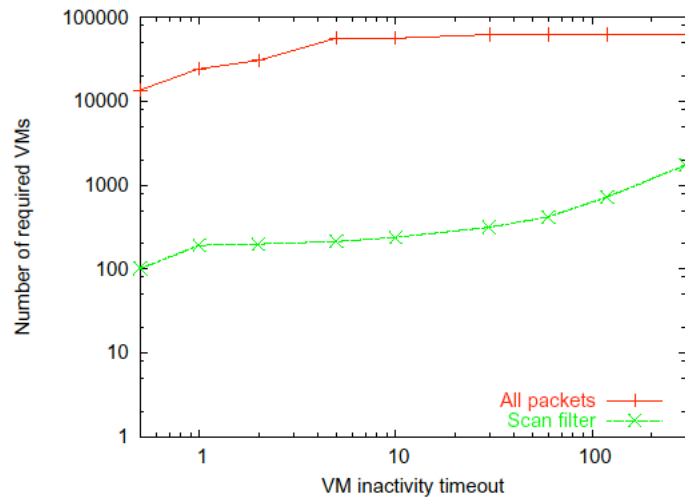
- Controller: VM independent
 - Aggressive filtering
 - Containment and redirection
 - Mapping and NAT: link incoming traffic to selected VM
- Honeypot Manager: VM dependent

Scan Filtering

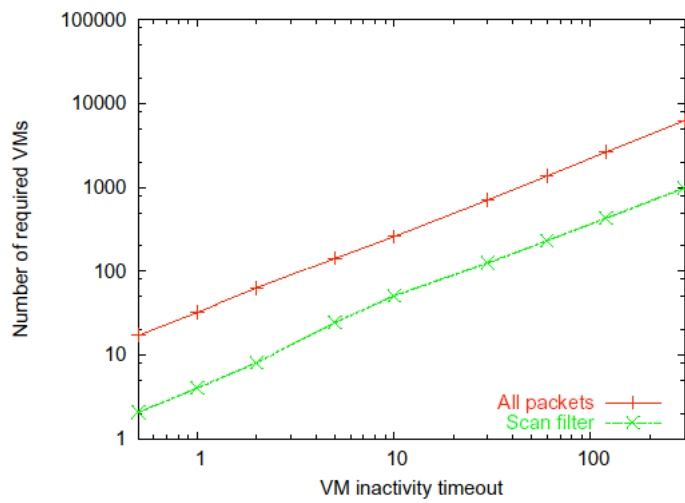
- GQ telescope: 250,000+ Internet addresses



- 10-20K probes/min: can't answer each with a VM!
- Simple filter: each origin gets N probes answered
- Major gain, but still need $\sim$ dozen VM's/sec

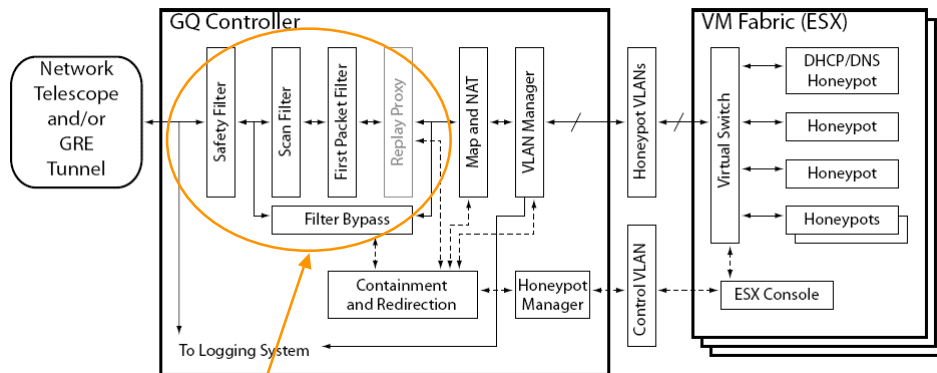


(a) Maximum number of simultaneous VMs



(b) Average number of simultaneous VMs

GQ Architecture



- Controller: VM independent
 - Aggressive filtering
 - Containment and redirection
 - Mapping and NAT: link incoming traffic to selected VM
- Honeypot Manager: VM dependent

⇒ SMB Negotiate Protocol Request (that stuff again)

⇐ SMB Negotiate Protocol Response

⇒ SMB Session Setup AndX Request

⇐ SMB Session Setup AndX Response

⇒ SMB Tree Connect AndX Request
Path: \\XX.128.18.16\IPC\$

⇐ SMB Tree Connect AndX Response

⇒ SMB NT Create AndX Req, Path: \samr

⇐ SMB NT Create AndX Response

⇒ DCERPC Bind: call_id: 1 UUID: SAMR

⇐ DCERPC Bind_ack:

⇒ SAMR Connect4 Request

⇐ SAMR Connect4 Reply

⇒ SAMR EnumDomains Request

⇐ SAMR EnumDomains Reply

⇒ SAMR LookupDomain Request

⇐ SAMR LookupDomain Reply

⇒ SAMR OpenDomain Request

⇐ SAMR OpenDomain Reply

⇒ SAMR EnumDomainUsers Request

⇒ SMB NT Create AndX Request,
Path: \srvsvc

⇐ SMB NT Create AndX Response

⇒ DCERPC Bind: call_id: 1 UUID: SRVSVC

⇐ DCERPC Bind_ack: call_id: 1

⇒ SRVSVC NetrRemoteTOD Request

⇐ SRVSVC NetrRemoteTOD Reply

⇒ SMB Close Request

⇐ SMB Close Response

⇒ SMB Tree Connect AndX Request,
Path: \\XX.128.18.16\ADMIN\$

⇐ SMB Tree Connect AndX Response

⇒ SMB NT Create AndX Request,
Path: \system32\msmqri32.exe

Only here do we find what file they're modifying

⇐ SMB NT Create AndX Response,
FID: 0x74ca

⇒ SMB Trans2Req SET_FILE_INFORMATION

⇐ SMB Trans2Resp SET_FILE_INFORMATION

⇒ SMB Trans2Req QUERY_FS_INFORMATION

⇐ SMB Trans2Resp QUERY_FS_INFORMATION

⇒ SMB Write Request

And only here do we find what code they're injecting into it!

Now start another session, connect to SRVSVC pipe and issue Remote-Time-of-Day Request

Filtering Out Complicated-But-Boring Probes

- Idea #1: for each such worm, craft a “script” that codifies its network activity
 - Doable in principle, but **tedious**: 1000s of both different worms & message types
- Idea #2: **automate** construction of such scripts using copies of the packet exchanges
 - Problem: for many protocols, two semantically identical sessions are **not** byte-wise identical
 - They differ in:
 - Embedded (known) network addresses & host names
 - Transaction identifiers, “cookie” fields
 - *Length* fields corresponding to these

⇒ SMB Negotiate Protocol Request (that stuff again)

⇐ SMB Negotiate Protocol Response

⇒ SMB Session Setup AndX Request

⇐ SMB Session Setup AndX Response

⇒ SMB Tree Connect AndX Request
Path: \\XX.128.18.16\IPC\$

⇐ SMB Tree Connect AndX Response

⇒ SMB NT Create AndX Req, Path: \samr

⇐ SMB NT Create AndX Response

⇒ DCERPC Bind: call_id: 1 UUID: SAMR

⇐ DCERPC Bind_ack:

⇒ SAMR Connect4 Request

⇐ SAMR Connect4 Reply

⇒ SAMR EnumDomains Request

⇐ SAMR EnumDomains Reply

⇒ SAMR LookupDomain Request

⇐ SAMR LookupDomain Reply

⇒ SAMR OpenDomain Request

⇐ SAMR OpenDomain Reply

⇒ SAMR EnumDomainUsers Request

⇒ SMB NT Create AndX Request, Path: \srvsvc

⇐ SMB NT Create AndX Response

⇒ DCERPC Bind: call_id: 1 UUID: SRVSVC

⇐ DCERPC Bind_ack: call_id: 1

⇒ SRVSVC NetrRemoteTOD Request

⇐ SRVSVC NetrRemoteTOD Reply

⇒ SMB Close Request

⇐ SMB Close Response

⇒ SMB Tree Connect AndX Request, Path: \\XX.128.18.16\ADMIN\$

⇐ SMB Tree Connect AndX Response

⇒ SMB NT Create AndX Request, Path: \system32\msmqri32.exe

Only here do we find what file they're modifying

⇐ SMB NT Create AndX Response, FID: 0x74ca

⇒ SMB Trans2Req SET_FILE_INFORMATION

⇐ SMB Trans2Resp SET_FILE_INFORMATION

⇒ SMB Trans2Req QUERY_FS_INFORMATION

⇐ SMB Trans2Resp QUERY_FS_INFORMATION

⇒ SMB Write Request

And only here do we find what code they're injecting into it!

Now start another session, connect to SRVSVC pipe and issue Remote-Time-of-Day Request

⇒ SMB Negotiate Protocol Request
 ⇐ SMB Negotiate Protocol Response
 ⇒ SMB Session Setup AndX Request
 ⇐ SMB Session Setup AndX Response
 ⇒ SMB Tree Connect AndX Request
 Path: \\XX.128.18.16\IPC\$
 ⇐ SMB Tree Connect AndX Response
 ⇒ SMB NT Create AndX Req, Path: \samr
 ⇐ SMB NT Create AndX Response
 ⇒ DCERPC Bind: call_id: 1 UUID: SAMR
 ⇐ DCERPC Bind_ack:
 ⇒ SAMR Connect4 Request
 ⇐ SAMR Connect4 Reply
 ⇒ SAMR EnumDomains Request
 ⇐ SAMR EnumDomains Reply
 ⇒ SAMR LookupDomain Request
 ⇐ SAMR LookupDomain Reply
 ⇒ SAMR OpenDomain Request
 ⇐ SAMR OpenDomain Reply
 ⇒ SAMR EnumDomainUsers Request

(that stuff again)
 ⇒ SMB NT Create AndX Request,
 Path: \srvsvc
 ⇐ SMB NT Create AndX Response
 ⇒ DCERPC Bind: call_id: 1 UUID: SRVSVC
 ⇐ DCERPC Bind_ack: call_id: 1
 ⇒ SRVSVC NetrRemoteTOD Request
 ⇐ SRVSVC NetrRemoteTOD Reply
 ⇒ SMB Close Request
 ⇐ SMB Close Response
 ⇒ SMB Tree Connect AndX Request,
 Path: \\XX.128.18.16\ADMIN\$
 ⇐ SMB Tree Connect AndX Response
 ⇒ SMB NT Create AndX Request,
 Path: \system32\msmsgri32.exe
 Only here do we find what file they're modifying
 ⇐ SMB NT Create AndX Response,
 FID: 0x74ca
 ⇒ SMB Trans2Req SET_FILE_INFORMATION
 ⇐ SMB Trans2Resp SET_FILE_INFORMATION
 ⇒ SMB Trans2Req QUERY_FS_INFORMATION
 ⇐ SMB Trans2Resp QUERY_FS_INFORMATION
 ⇒ SMB Write Request
 And only here do we find what code they're
 injecting into it!

Now start another session, connect
 to SRVSVC pipe and issue Remote-
 Time-of-Day Request

Matching Protocol Dialog In A New Setting

```
A->V N1|180|S26|0388|S7|96|S20|96|S8|113|S16|R8|96|R6|72|R4|0014CCB8|18|M4|18|M4
|144.165.114.119|M20
A<-V N4|S26|0388|S28|R24|M16|0000000092F3E82470FDD91195F8000C295763F7|M4
A->V N4|S26|0388|S56|R24|0000000092F3E82470FDD91195F8000C295763F7|M8
A<-V N1|180|S26|0388|S7|124|S8|124|S6|125|S1|R8|124|DCERPC-6|100|R4|M4|000B0BB0
|M4|000B27A0|M8|8|10|000B8D18|M8|000BC610|5|M4|4|h0ne|M36
A->V N1|156|S26|0388|S7|72|S20|72|S8|89|S16|R8|72|R6|48|R4|M4
|0000000092F3E82470FDD91195F8000C295763F7|8|10|001503F8|5|M4|4|h0ne
A<-V N4|S26|0388|S28|R24|000B27A0|M32
```

N4 = 4 bytes of NetBIOS
 S26 = 26 bytes of SMB (Server Message Block)
 R24 = 24 bytes of DCE-RPC
 M32 = Security Account Manager

Matching Protocol Dialog In A New Setting

```

A->V N1|180|S26|0388|S7|96|S20|96|S8|113|S16|R8|96|R6|72|R4|0014CCB8|18|M4|18|M4
      |144.165.114.119|M20
A<-V N4|S26|0388|S28|R24|M16|00000000|92F3E82470FDD91195F8000C295763F7|M4
A->V N4|S26|0388|S56|R24|00000000|92F3E82470FDD91195F8000C295763F7|M8
A<-V N1|180|S26|0388|S7|124|S8|124|S6|125|S1|R8|124|DECRPC-6|100|R4|M4|000B0BB0
      |M4|000B27A0|M8|8|10|000B8D18|M8|000BC610|5|M4|4|hone|M36
A->V N1|156|S26|0388|S7|72|S20|72|S8|89|S16|R8|72|R6|48|R4|M4
      |00000000|92F3E82470FDD91195F8000C295763F7|8|10|001503F8|5|M4|4|hone
A<-V N4|S26|0388|S28|R24|000B27A0|M32

```

Grey = embedded length field

Bold = transaction ID / “cookie” field

Bold Italic = embedded IP address or hostname

- How can we accurately identify & adjust all of these?

Two Dialogs for Matching Randex

```

A->V N1|180|S26|0388|S7|96|S20|96|S8|113|S16|R8|96|R6|72|R4|0014CCB8|18|M4|18|M4
      |144.165.114.119|M20
A<-V N4|S26|0388|S28|R24|M16|00000000|92F3E82470FDD91195F8000C295763F7|M4
A->V N4|S26|0388|S56|R24|00000000|92F3E82470FDD91195F8000C295763F7|M8
A<-V N1|180|S26|0388|S7|124|S8|124|S6|125|S1|R8|124|DECRPC-6|100|R4|M4|000B0BB0
      |M4|000B27A0|M8|8|10|000B8D18|M8|000BC610|5|M4|4|hone|M36
A->V N1|156|S26|0388|S7|72|S20|72|S8|89|S16|R8|72|R6|48|R4|M4
      |00000000|92F3E82470FDD91195F8000C295763F7|8|10|001503F8|5|M4|4|hone
A<-V N4|S26|0388|S28|R24|000B27A0|M32

```

```

A->V N1|172|S26|0474|S7|88|S20|88|S8|105|S16|R8|88|R6|64|R4|0014CCB8|14|M4|14|M4
      |48.196.8.48|M20
A<-V N4|S26|0474|S28|R24|M16|00000000|6093917586FDD91195F8000C294A478F|M4
A->V N4|S26|0474|S56|R24|00000000|6093917586FDD91195F8000C294A478F|M8
A<-V N1|184|S26|0474|S7|128|S8|128|S6|129|S1|R8|128|DECRPC-6|104|R4|M4|000B0BB0
      |M4|000B6380|M8|12|14|000B76C0|M8|000C9FA8|7|M4|6|host02|M36
A->V N1|160|S26|0474|S7|76|S20|76|S8|89|S16|R8|76|R6|52|R4|M4
      |00000000|6093917586FDD91195F8000C294A478F|12|14|001503F8|7|M4|6|host02
A<-V N4|S26|0474|S28|R24|000B27A0|M32

```

Replay Proxy

Attacker

Proxy

Honeypot

1

Roleplayer proxy receives the beginning of an exploit attempt

Replay Proxy

Attacker

Proxy

Honeypot

1

2

*Roleplayer finds a match for attacker's first message and replies to it based on its *script**

Replay Proxy

Attacker

Proxy

Honeypot

1

2

3

In response to script
reply, attacker sends
along next stage of
the attack

Replay Proxy

Attacker

Proxy

Honeypot

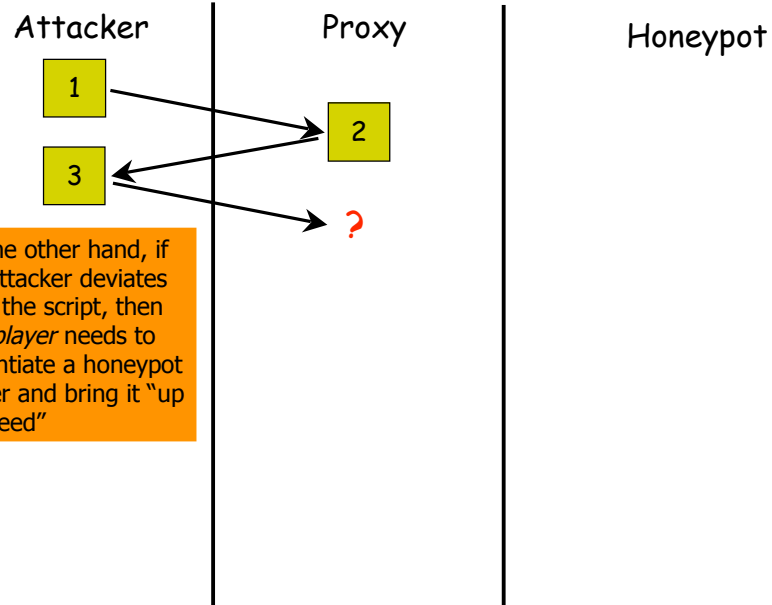
1

2

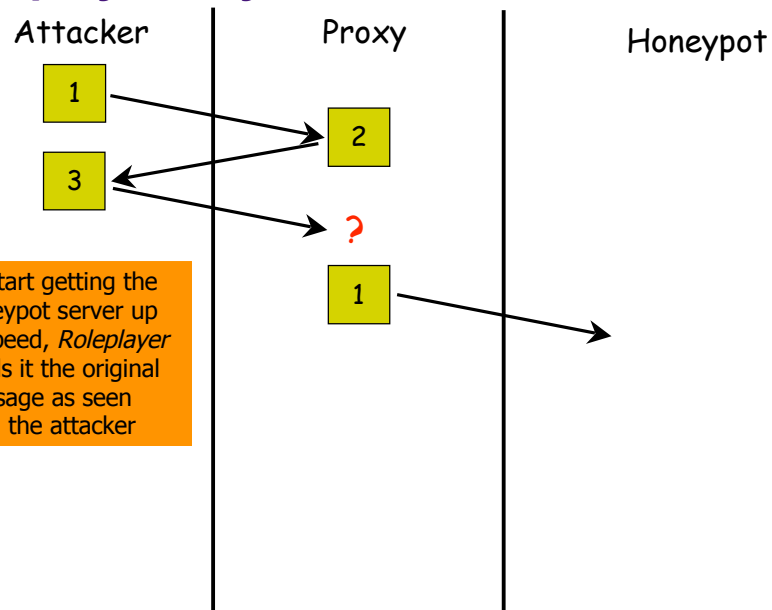
3

At this point, the
process continues using
the script. If it matches
all the way until the
end, then there's no
need to instantiate a
honeypot ...

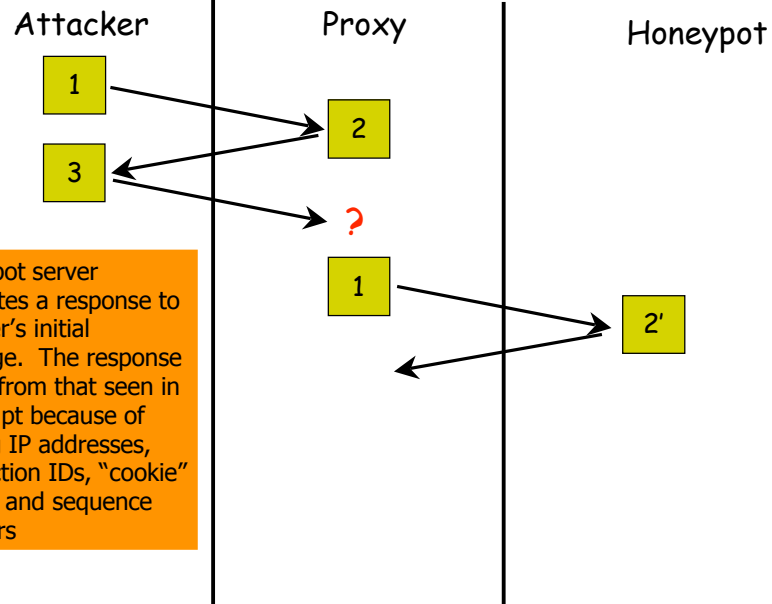
Replay Proxy



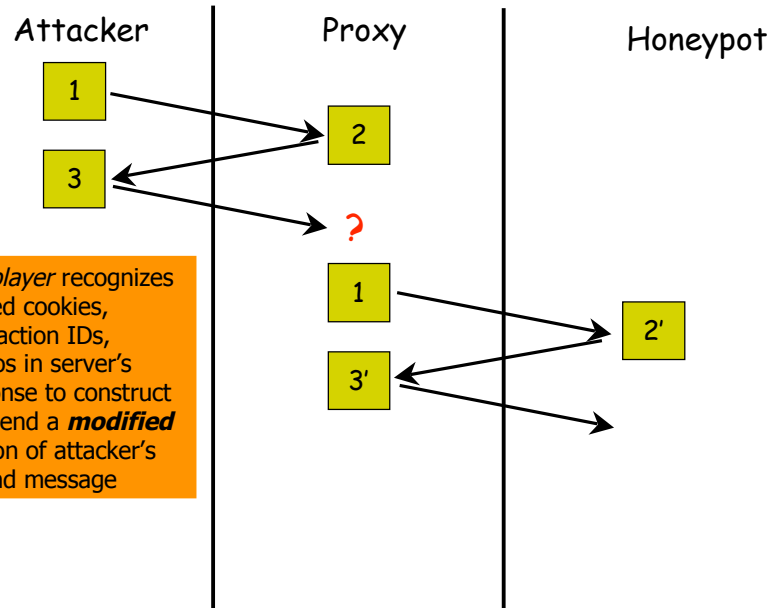
Replay Proxy



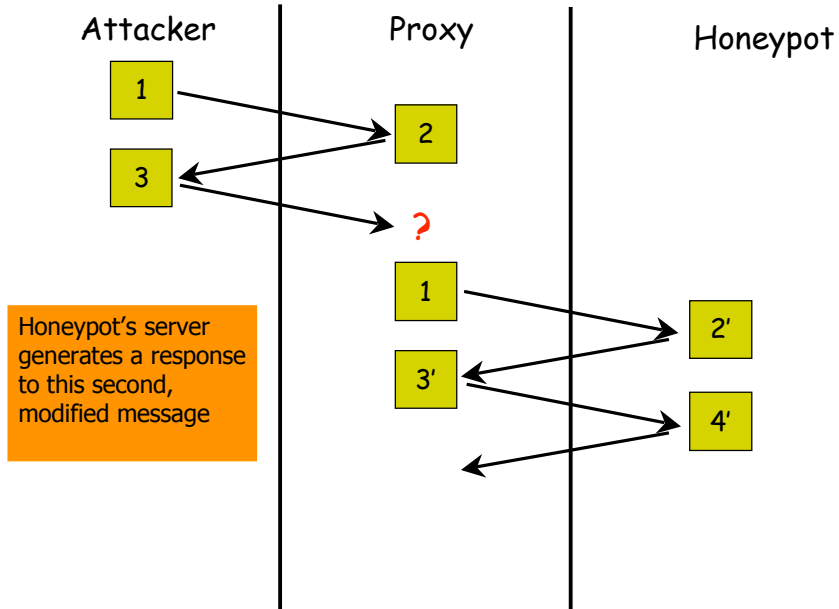
Replay Proxy



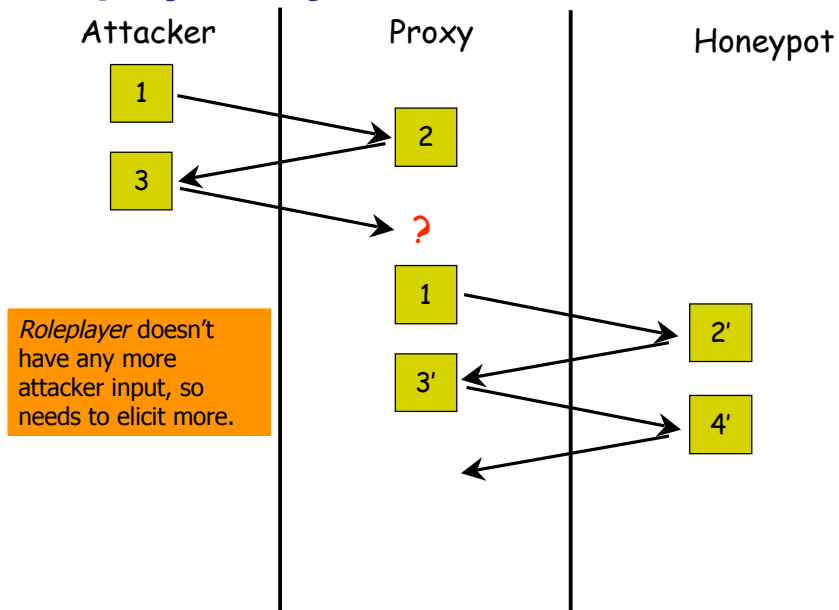
Replay Proxy



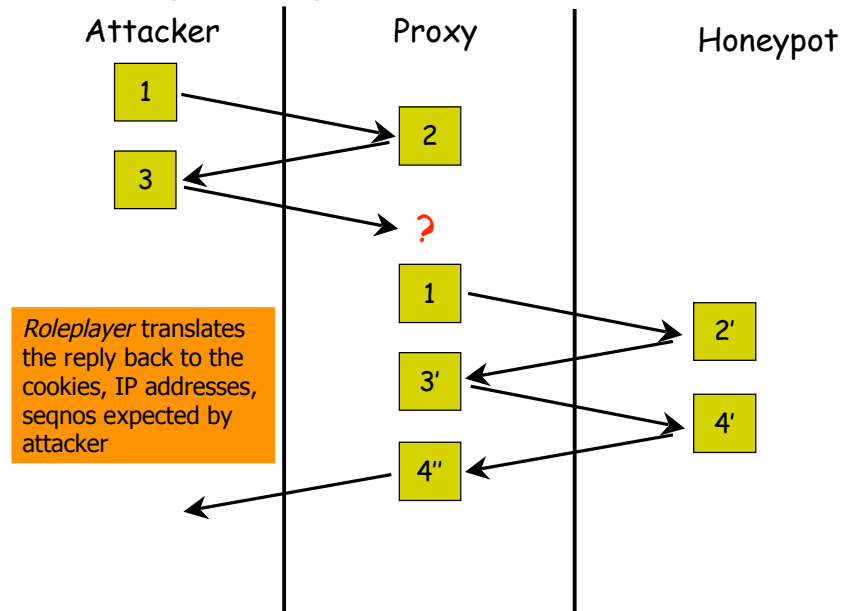
Replay Proxy



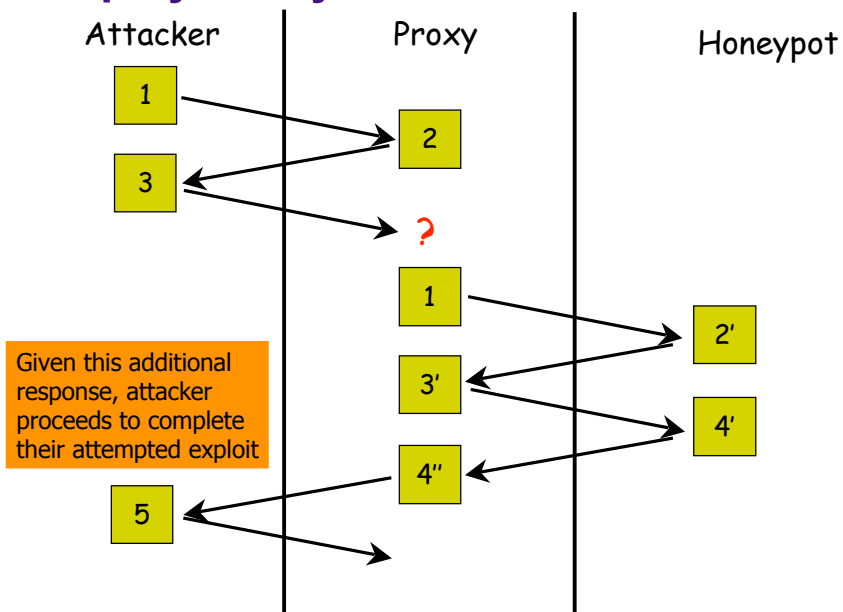
Replay Proxy



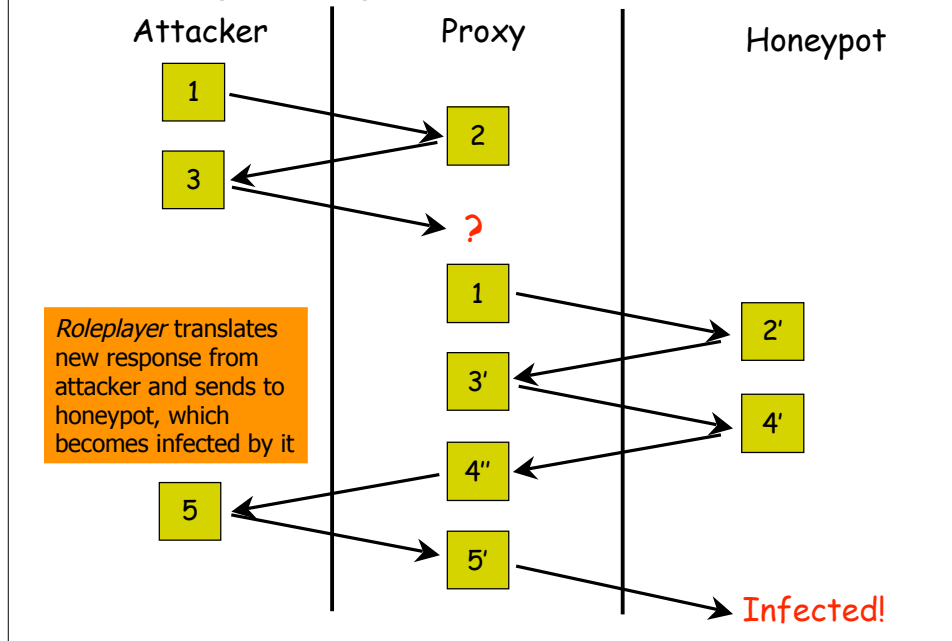
Replay Proxy



Replay Proxy



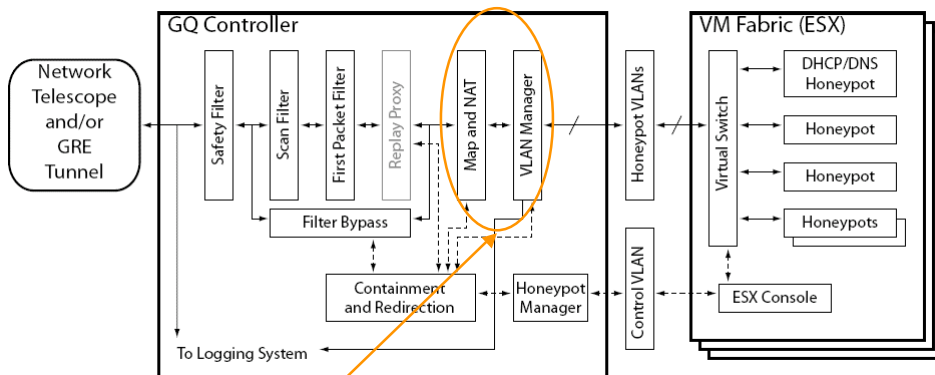
Replay Proxy



Executable Name	Size (B)	MD5Sum	Worm Name	# Events	# Conns	Time (s)
a####.exe	10366	7a67f7c8...	W32.Zotob.E	4	3	29.0
a####.exe	10878	bf47cfe2...	W32.Zotob.H	9	3	25.2
a####.exe	25726	62697686...	Quarantined but no name	1	3	223.2
cpufanctrl.exe	191150	1737ec9a...	Backdoor.Sdbot	1	4	111.2
chkdisk32.exe	73728	27764a5d...	Quarantined but no name	1	4	134.7
dlhhost.exe	10240	53bfe15e...	W32.Welchia.Worm	297	4 or 6	24.5
enbiei.exe	11808	d1ee9d2e...	W32.Blaster.F.Worm	1	3	28.9
msblast.exe	6176	5ae700c1...	W32.Balster.Worm	1	3	43.8
lsd	18432	17028f1e...	W32.Poxdar	11	8	32.4
NeroFil.EXE	78480	5ca9a953...	W32.Spybot.Worm	1	5	237.5
sysmsn.exe	93184	5f6c8c40...	W32.Spybot.Worm	3	3	79.6
MsUpdaters.exe	107008	aa0ee4b0...	W32.Spybot.Worm	1	5	57.0
RealPlayer.exe	120320	4995eb34...	W32.Spybot.Worm	2	5	95.4
WinTemp.exe	209920	9e74a7b4...	W32.Spybot.Worm	1	5	178.4
wins.exe	214528	7a9aee7b...	W32.Spybot.Worm	1	5	118.2
msnet.exe	238592	6355d4d5...	W32.Spybot.Worm	1	7	189.4
MSGUPDATES.EXE	241152	65b401eb...	W32.Spybot.Worm	2	5	125.3
ntsf.exe	211968	5ac5998e...	Quarantined but no name	1	5	459.4
scardsvr32.exe	33169	1a570b48...	W32.Femot.Worm	4	3	46.2
scardsvr32.exe	34304	b10069a8...	W32.Femot.Worm	1	3	66.5
scardsvr32.exe	34816	ba599948...	W32.Femot.Worm	55	3	96.6
scardsvr32.exe	35328	617b4056...	W32.Femot.Worm	2	3	179.6
scardsvr32.exe	36864	0372809c...	W32.Femot.Worm	1	5	49.3
scardsvr32.exe	39689	470de280...	W32.Femot.Worm	4	3	41.4
scardsvr32.exe	40504	23055595...	W32.Femot.Worm	1	3	41.1
scardsvr32.exe	43008	ff20f56b...	W32.Valla.2048	1	5	32.2
scardsvr32.exe	66374	f7a00ef5...	Quarantined but no name	1	7	54.8

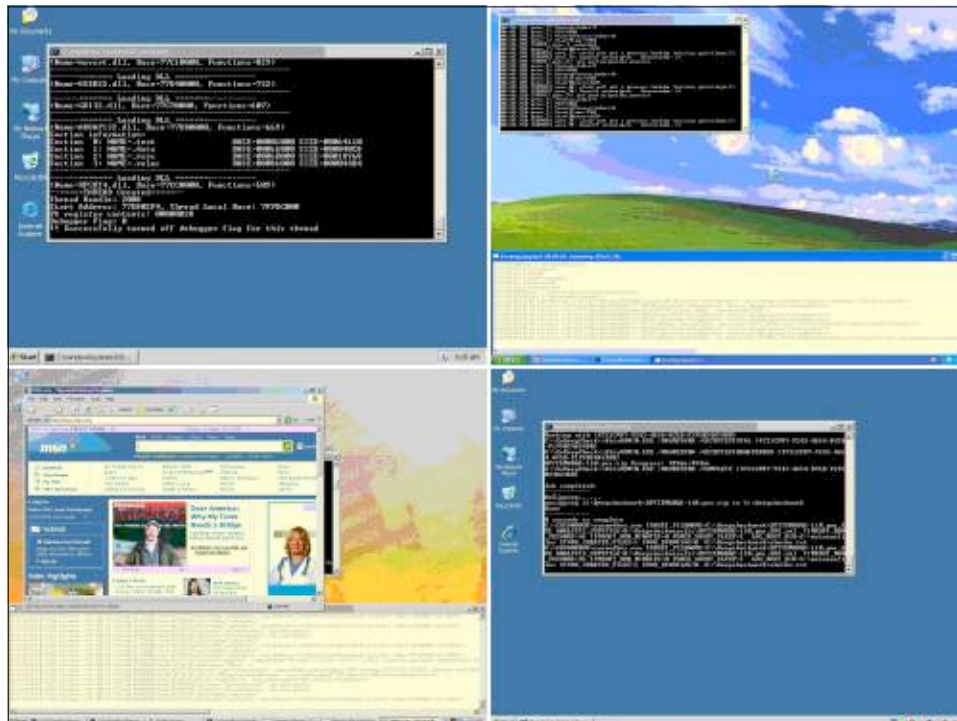
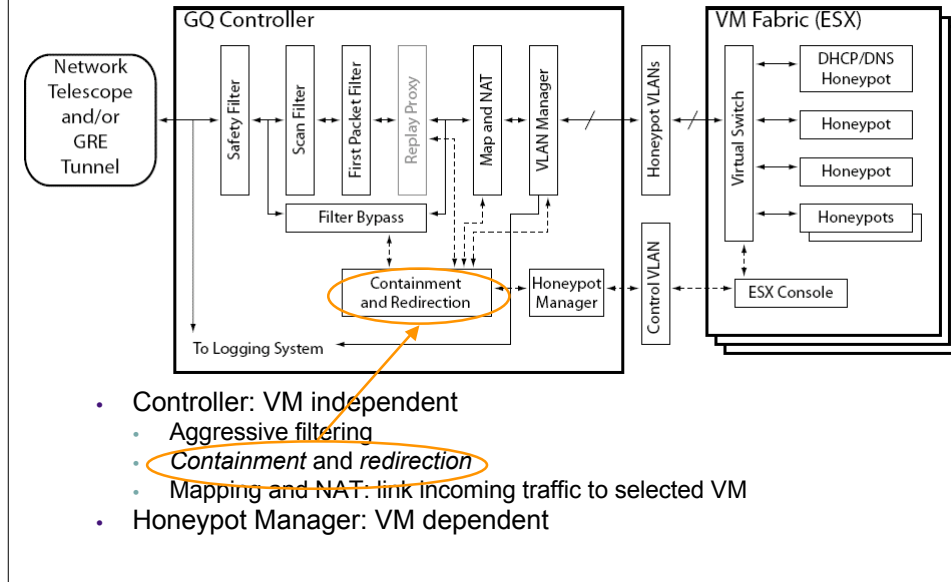
x.exe	9343	986b5970...	W32.Korgo.Q	17	2	6.6
x.exe	9344	d6df3972...	W32.Korgo.T	7	2	9.5
x.exe	9353	7d99b0e9...	W32.Korgo.V	102	2	6.0
x.exe	9359	a0139d7a...	W32.Korgo.W	31	2	5.9
x.exe	9728	c05385e6...	W32.Korgo.Z	20	2	6.6
x.exe	11391	7f60162c...	W32.Korgo.S	169	2	6.6
x.exe	11776	c0610a0d...	W32.Korgo.S	15	2	8.6
x.exe	13825	0b80b637...	W32.Korgo.V	2	2	24.4
x.exe	20992	31385818...	W32.Licum	2	2	7.9
x.exe	23040	e0989c83...	W32.Korgo.S	3	2	10.4
x.exe	187348	384c6289...	W32.Pinfi	1	2	329.7
x.exe	187350	a4410431...	W32.Korgo.V	6	2	11.3
x.exe	187352	b3673398...	W32.Pinfi	5	2	20.1
x.exe	187354	c132582a...	W32.Pinfi	5	2	24.9
x.exe	187356	d586e6c2...	W32.Pinfi	2	2	27.5
x.exe	187358	2430c64c...	W32.Korgo.V	1	2	27.5
x.exe	187360	eb1d07c1...	W32.Pinfi	1	2	63.1
x.exe	187392	2d9951ca...	W32.Korgo.W	1	2	76.1
x.exe	189400	7d195c0a...	W32.Korgo.S	1	2	18.0
x.exe	189402	c03b5262...	W32.Pinfi	1	2	58.2
x.exe	189406	4957f2e3...	W32.Korgo.S	1	2	210.9
xxxx...x	46592	a12cab51...	Backdoor.Berbew.N	844	2	9.4
xxxx...x	56832	b783511e...	W32.Info.A	34	2	7.2
xxxx...x	57856	ab5e47bf...	Trojan.Dropper	685	3	10.0
xxxx...x	224218	d009d6e5...	W32.Pinfi	1	3	32.5
xxxx...x	224220	af79e0c6...	W32.Pinfi	3	2	34.2
n/a	10240	7623c942...	W32.Korgo.C	3	2	4.8
n/a	10752	1b90cc9f...	W32.Korgo.L	1	2	7.0
n/a	10752	32a0d7d0...	W32.Korgo.G	8	2	4.1
n/a	10752	ab7ecc7a...	W32.Korgo.N	2	2	5.3
n/a	10752	d175bad0...	W32.Korgo.G	3	2	5.4
n/a	10752	d85bf0c5...	W32.Korgo.E	1	2	5.6
n/a	10752	b1e7d9ba...	W32.Korgo.gen	1	2	5.0
n/a	10879	042774a2...	W32.Korgo.I	15	2	4.3
n/a	11264	a36ba4a2...	W32.Korgo.I	1	2	5.4
multiple	n/a	n/a	W32.Muma.A	2	7	186.7
multiple	n/a	n/a	W32.Muma.B	2	7	208.9
multiple	n/a	n/a	BAT.Boohoo.Worm	1	72	384.9

GQ Architecture



- Controller: VM independent
 - Aggressive filtering
 - Containment and redirection
 - Mapping and NAT: link incoming traffic to selected VM
- Honeypot Manager: VM dependent

GQ Architecture



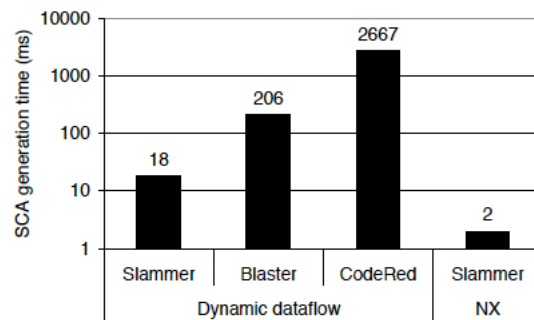
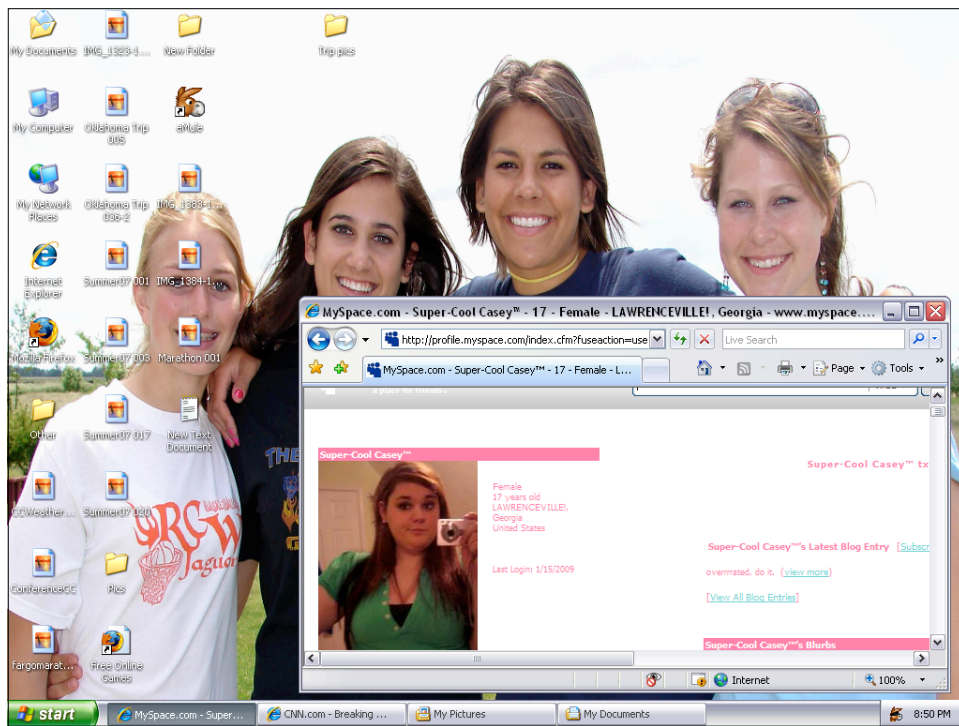


Figure 8: SCA generation time in milliseconds for real worms using two detectors.

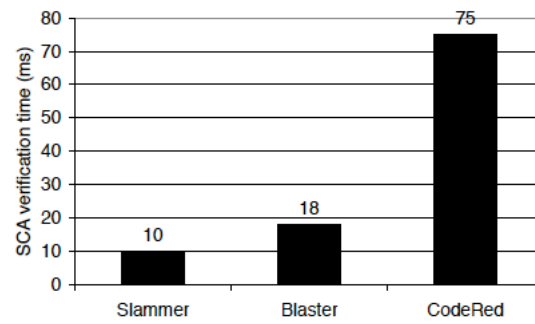


Figure 10: SCA verification time in milliseconds for real worms.