Targeted cyber attacks: examples and challenges ahead

Levente Buttyán
Budapest University of Technology and Economics
Department of Networked Systems and Services
Laboratory of Cryptography and System Security (**CrySyS Lab**)
www.crysys.hu

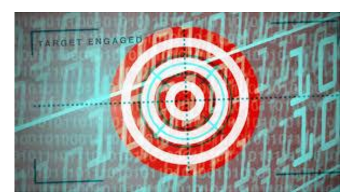
joint work with **Boldizsár Bencsáth**, **Gábor Pék**, and **Márk Félegyházi**

Cyber attacks

- cyber crime
  - financial motivations

- hacktivism (e.g., Anonymous)
  - political motivations

- targeted attacks
  - espionage
  - sabotage
  - DoS



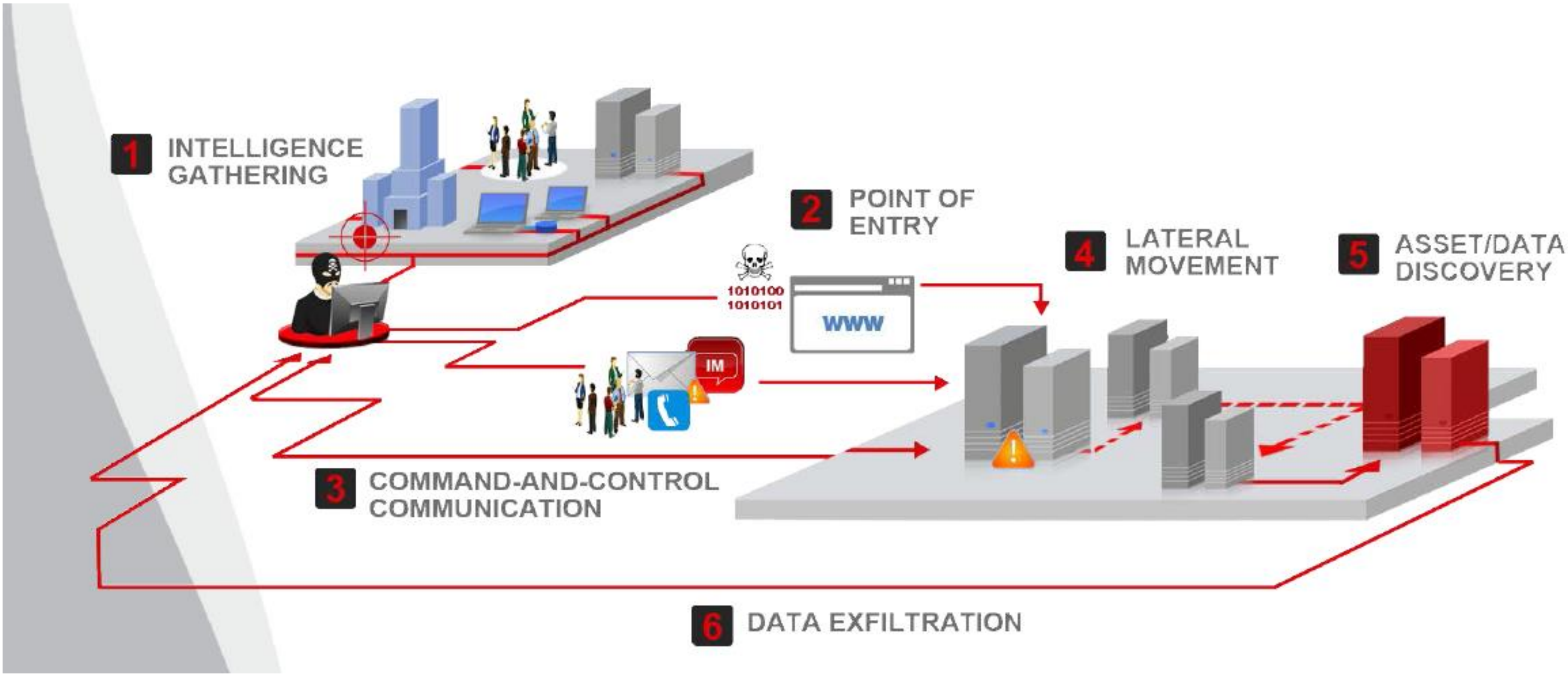
Targeted cyber attacks

- highly customized tools and intrusion techniques
  - multiple different exploits in each campaign
  - often using zero-day (or very fresh) exploits
- stealthy operation and persistence
  - reduced risk of detection
  - average time of undetected compromise is ~1 year
- well-funded and well-staffed organizations behind
  - military or state intelligence organizations
  - large companies

Malware

- often used mechansim in targeted attacks
- types
  - virus, worm, Trojan, ...
- delivery methods
  - attachment of a targeted e-mail
    - spear phishing based on social engineering
  - redirection to a malicious web page
    - drive-by-download
  - through a compromised legitimate site
    - watering hole attacks
  - through an infected USB drive
- infection by exploiting known or publicly unknown vulnerabilities
  - bugs in the OS or in applications (e.g., web browser, office suite)
  - often allow for executing arbitrary code (e.g., installer of the malware)

Phases of targeted attacks



source: TrendMicro Security Intelligence Blog

The problem

source: Symantec Internet Security Threat Report 2013

Questions

- Why are these attacks so successful?
- How do they work? What sort of tricks do they use?
- Why and how do our traditional security tools fail?
- What can be done to mitigate the problem?
- Are we at the dawn of a paradigm shift in security?

Outline

- some personal experience with sophisticated malware used in targeted attacks
- lessons learned in these experiences
- challenges for the security research community

Duqu (2011)

Home / News & Blogs / Zero Day

Hungarian Lab found Stuxnet-like Duqu malware

By Ryan Naraine | October 21, 2011, 9:11am PDT

Summary: *The Laboratory of Cryptography and System Security (CrySyS) in Hungary confirmed its participation in the initial discovery of the Duqu cyber-surveillance Trojan.*

Laboratory of Cryptography and System Security
Budapest University of Technology and Economics
Department of Telecommunications
www.crysys.hu
TO BE ON THE SAFE SIDE

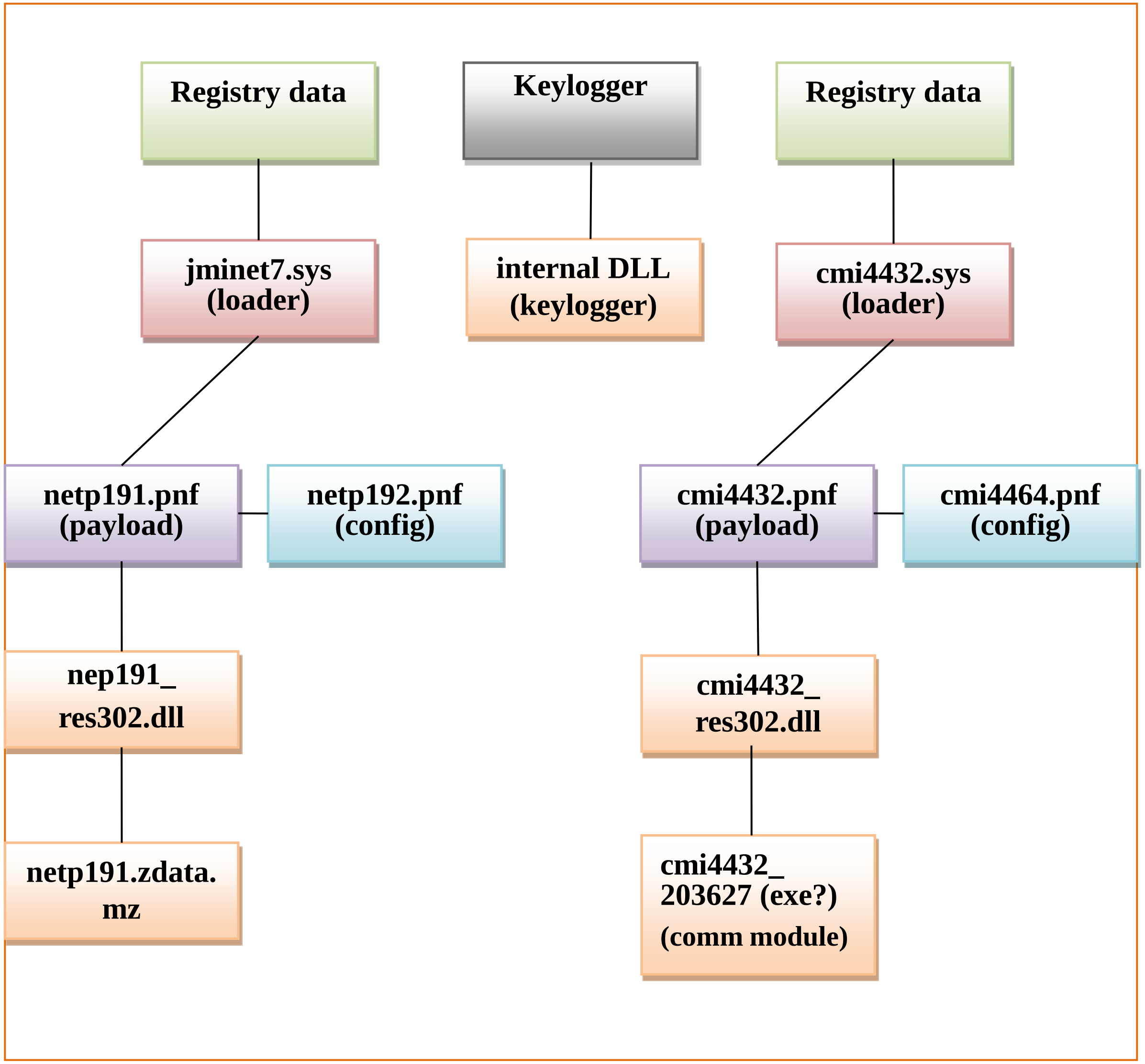
A security lab attached to the Budapest University of Technology and Economics in Hungary has come forward as the mystery outfit that found the Stuxnet-like "Duqu" cyber-surveillance Trojan.

According to Symantec's initial report on Duqu [PDF], the malware sample was passed along by an unnamed "research lab with strong international connections," a statement that led to speculation about the origins and intent of the threat.

Summary of contributions

- **discovery, naming, and first analysis of Duqu**
  - creates files with names starting with ~DQ
  - striking similarities to Stuxnet, but different objective
  - advanced cyber espionage tool (keystrokes, screen shots, files)
  - 60-page report shared with major AV vendors and Microsoft
  - ~20 known victims, mainly in Iran, but also in Europe
- **identification of the dropper**
  - MS Word document with a 0-day Windows kernel exploit
  - anonymization of the dropper before sharing with Microsoft
- **development of the Duqu Detector Toolkit**
  - focus on heuristic behavior based detection
  - detects live Duqu instances and remains of earlier infections
  - also detects Stuxnet !!
  - open-source distribution (to be usable in critical infrastructures)
  - 12000+ downloads from 150 countries

Duqu components identified

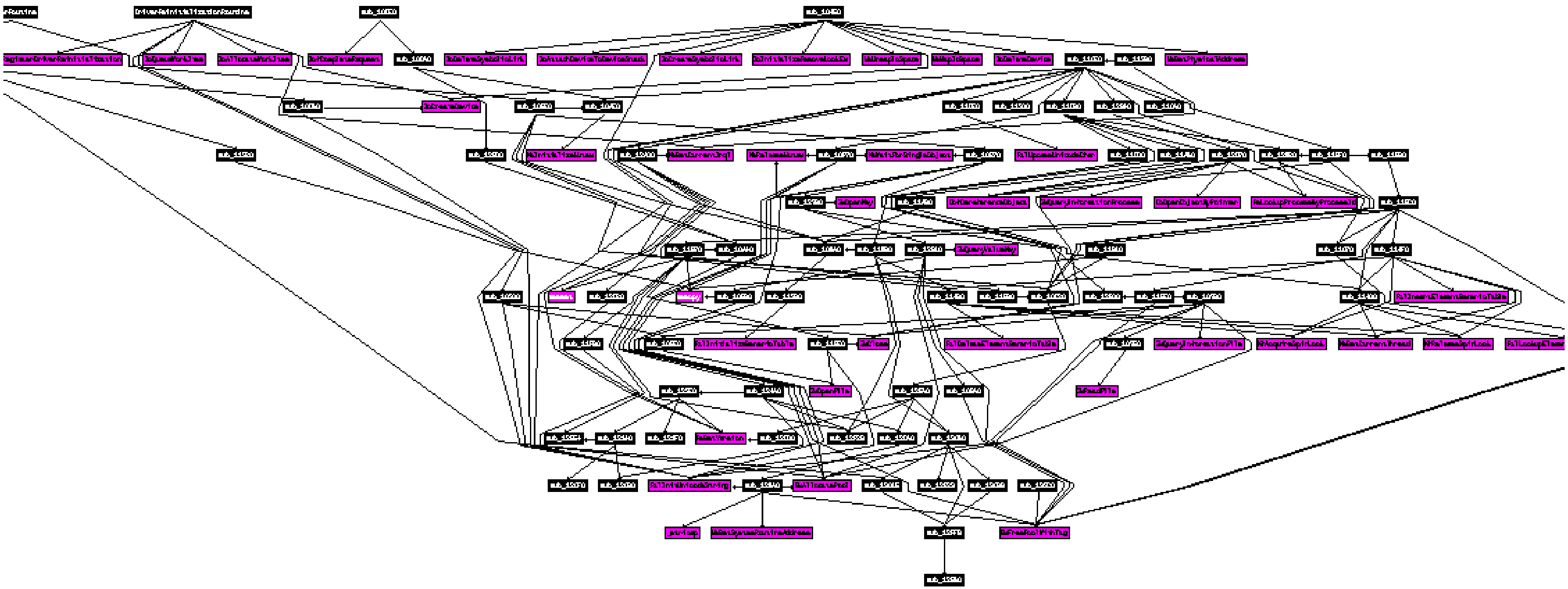


Static analysis of binaries

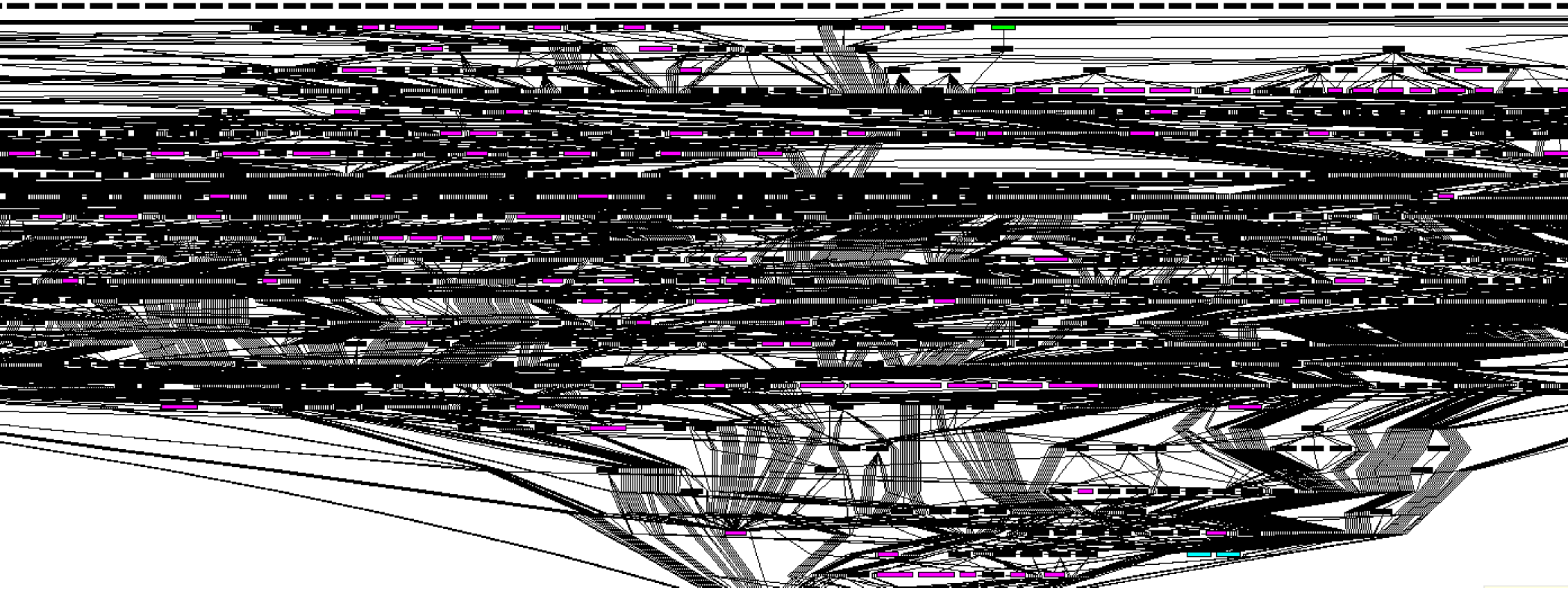
```
text:10001C41                  push    edi
.text:10001C42                 push    790E4013h       ; GetKernelObjectSecurity
.text:10001C47                 mov     [ebp+var_24], eax
.text:10001C4A                 mov     [ebp+var_34], eax
.text:10001C4D                 call    searchin_dll2_100022C7
.text:10001C52                 mov     edi, eax
.text:10001C54                 mov     [esp+10h+var_10], 0E876E6Eh ; GetSecurityDescriptorDacl
.text:10001C5B                 call    searchin_dll2_100022C7
.text:10001C60                 push    0E1BD5137h      ; BuildExplicitAccessWithNameW
.text:10001C65                 mov     [ebp+var_C], eax
.text:10001C68                 call    searchin_dll2_100022C7
.text:10001C6D                 push    2F03FA6Fh       ; SetEntriesInAclW
.text:10001C72                 mov     ebx, eax
.text:10001C74                 call    searchin_dll2_100022C7
.text:10001C79                 push    0C69CF599h      ; MakeAbsoluteSD
.text:10001C7E                 mov     [ebp+var_4], eax
.text:10001C81                 call    searchin_dll2_100022C7
.text:10001C86                 push    0CE8CAD1Ah      ; SetSecurityDescriptorDacl
.text:10001C8B                 mov     [ebp+var_8], eax
.text:10001C8E                 call    searchin_dll2_100022C7
.text:10001C93                 push    9A71C67h        ; SetKernelObjectSecurity
.text:10001C98                 mov     [ebp+var_10], eax
.text:10001C9B                 call    searchin_dll2_100022C7

ext:10002565                   call    sub_1000211F
.text:1000256A                 mov     ecx, [ebp+var_4]
.text:1000256D                 mov     [ecx], eax
.text:1000256F                 push    4BBFABB8h       ; lstrcmpiW
.text:10002574                 call    searchin_dll1_100022B6
.text:10002579                 pop     ecx
.text:1000257A                 mov     ecx, [ebp+var_4]
.text:1000257D                 mov     [ecx+8], eax
.text:10002580                 push    0A668559Eh      ; VirtualQuery
.text:10002585                 call    searchin_dll1_100022B6
.text:1000258A                 pop     ecx
.text:1000258B                 mov     ecx, [ebp+var_4]
.text:1000258E                 mov     [ecx+0Ch], eax
.text:10002591                 push    4761BB27h       ; VirtualProtect
.text:10002596                 call    searchin_dll1_100022B6
.text:1000259B                 pop     ecx
.text:1000259C                 mov     ecx, [ebp+var_4]
.text:1000259F                 mov     [ecx+10h], eax
.text:100025A2                 push    0D3E360E9h      ; GetProcAddress
```

Duqu – jminet7 driver structure



Duqu – netp191 main module uncompressed



Similarity to Stuxnet

Feature	Stuxnet	Duqu
Modular malware	✓	✓
Kernel driver based rootkit	✓	✓ very similar
Valid digital signature on driver	Realtek, JMicron	C-Media
Injection based on A/V list	✓	✓ seems based on Stux.
Imports based on checksum	✓	✓ different alg.
3 Config files, all encrypted, etc.	✓	✓ almost the same
Keylogger module	?	✓
PLC functionality	✓	✗ (different goal)
Infection through local shares	✓	No proof, but seems so
Exploits	✓	?
0-day exploits	✓	?
DLL injection to system processes	✓	✓
DLL with modules as resources	✓ (many)	✓ (one)
RPC communication	✓	✓
RPC control in LAN	✓	?
RPC Based C&C	✓	?
Port 80/443, TLS based C&C	?	✓
Special "magic" keys, e.g. 790522, AE	✓	✓ lots of similar
Virtual file based access to modules	✓	✓
Usage of LZO lib	?	✓ multiple
Visual C++ payload	✓	✓
UPX compressed payload,	✓	✓
Careful error handling	✓	✓
Deactivation timer	✓	✓
Initial Delay	? Some	✓ 15 mins
Configurable starting in safe mode/dbg	✓	✓ (exactly same mech.)

Table 1 – Comparing Duqu and Stuxnet at the first glance

Duqu Detector Toolkit

- instead of signature based identification, focus on behavior based anomaly detection
  - e.g., PNF files without corresponding INF files
  - detection of encrypted components and registry entries
  - false positives are acceptable, given the critical nature of potential targets
- simple components (tools) provided in C source code
- evaluation results
  - detect Duqu and Stuxnet.A
  - low number of false positive alarms
- the toolkit has been downloaded from more than 12000 distinct IP addresses distributed over 150 countries

How could Duqu have been detected?

- manual inspection of the system status
  - multiple running instances of lsass.exe
  - suspicious function calls into sortEA74.nls in the stack trace of a malicious lsass instance (checked with Sysinternal's Process Monitor)
  - by checking the bootlog, one can figure out that the parent process of lsass.exe is svchost.exe, whose parent process is services.exe, which injects code into lsass.exe, alg.exe, imapi.exe, spoolsv.exe and other svchost.exe instances
- we also tested the hook detection performance of ~40 freely available rootkit detection tools on Duqu infected computers
  - several tools identified anomalies that would be very suspicious for a knowledgeable system administrator

Hook detection results

Tools with hook detection capabilities	Results on Duqu
CMC CodeWalker (2008)	16 hooks in lsass.exe (BSoD during test)
Gmer (1.0.15.15641)	inline hooks in lsass.exe (PID: 1176, 1236, 1930, 2016) inline hooks in svchost.exe (PID: 996, 1084)
NoVirusThanks Anti-Rootkit v1.2	- (detects only unrelated Message hooks)
McAfee Rootkit Detective 1.0	-
RKDetector v2.0 IAT API Hooks Analyser	IAT hook in explorer.exe
Rootkit Unhooker LE v3.7.300.509	inline hooks in svchost.exe (PID: 996) IAT hook in explorer.exe inline and IAT hooks in lsass.exe (PID: 1236, 1176, 1048, 1416)
Sysinternals RootkitRevealer	-
TrendMicro Rootkit Buster v5.0 2011	-
Usec Radix v1.0.0.13	IAT hook in explorer.exe
XueTr	inline and IAT hooks in svchost.exe (PID: 1084, 996) IAT hook in explorer.exe inline and IAT hooks in lsass.exe (PID:1176, 1920, 2016, 1236) (IAT hooks in every process uses the hooked function)

The Duqu experience

- we have never done this sort of work before
  - no hands-on experience with state-of-the-art tools (e.g., IDApro)
  - didn’t know the information sharing practices in the AV industry
  - not sure about our expected role given the supposed creator(s) and purpose of Duqu
- we worked under extreme time pressure
- “Luck favors the prepared mind” (Pasteur)
- increased visibility
  - media coverage
  - invitations to various places and visits of different “entities”

Flame (aka sKyWIper) (2012)

BBC | News | Sport | Weather | Travel | Future

NEWS TECHNOLOGY

Home | UK | Africa | Asia | Europe | Latin America | Mid-East | US & Canada | Business | Health

An **in-depth look at Flame by the Laboratory of Cryptography and System Security** at Hungary's University of Technology and Economics in Budapest, said it stayed hidden because it was so different to the viruses, worms and trojans that most security programmes were designed to catch.

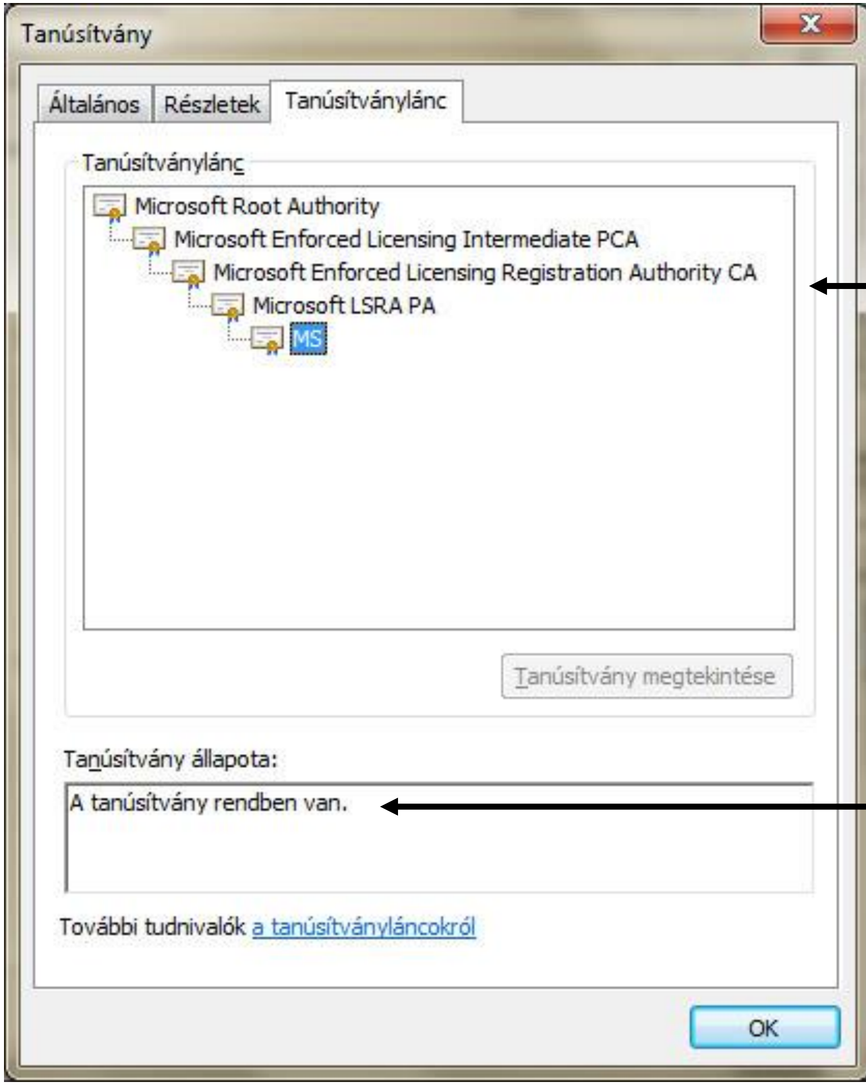
Flame (aka sKyWIper) (2012)

- another info-stealer malware
  - activates microphones and web cam
  - logs key strokes
  - takes screen shots
  - extracts geolocation data from images
  - sends and receives commands and data through Bluetooth
- data saved in SQL databases
- data transport via network connections and USB pen drive
- **infects computers by masquerading as a proxy for Windows update**
  - uses a fake certificate that looks like a valid Microsoft certificate
  - needed advanced collision attack on the MD5 hash function
- thousands of victims, mostly in Iran, Israel (Palestine territories), and Sudan, but also in Hungary!

Flame vs. Duqu (Stuxnet)

- Flame is a platform different form Duqu (and Stuxnet)
  - larger code size
  - use of Lua scripting language
  - use of SQLite databases
  - larger C&C infrastructure
  - C&C servers run different OS (Ubuntu vs. CentOS Linux)
- they may be two implementations for the same requirement specification developed by two different teams
- the two teams may not be independent
  - Kaspersky researchers found chunks of code from a 2009 Stuxnet variant inside Flame
  - CrySyS Lab identified an encryption routine that is the same in Flame and Stuxnet

The fake certificate used by Flame



chains up to the MS root

can be used for
code signing!

looks valid

MD5 collision attack

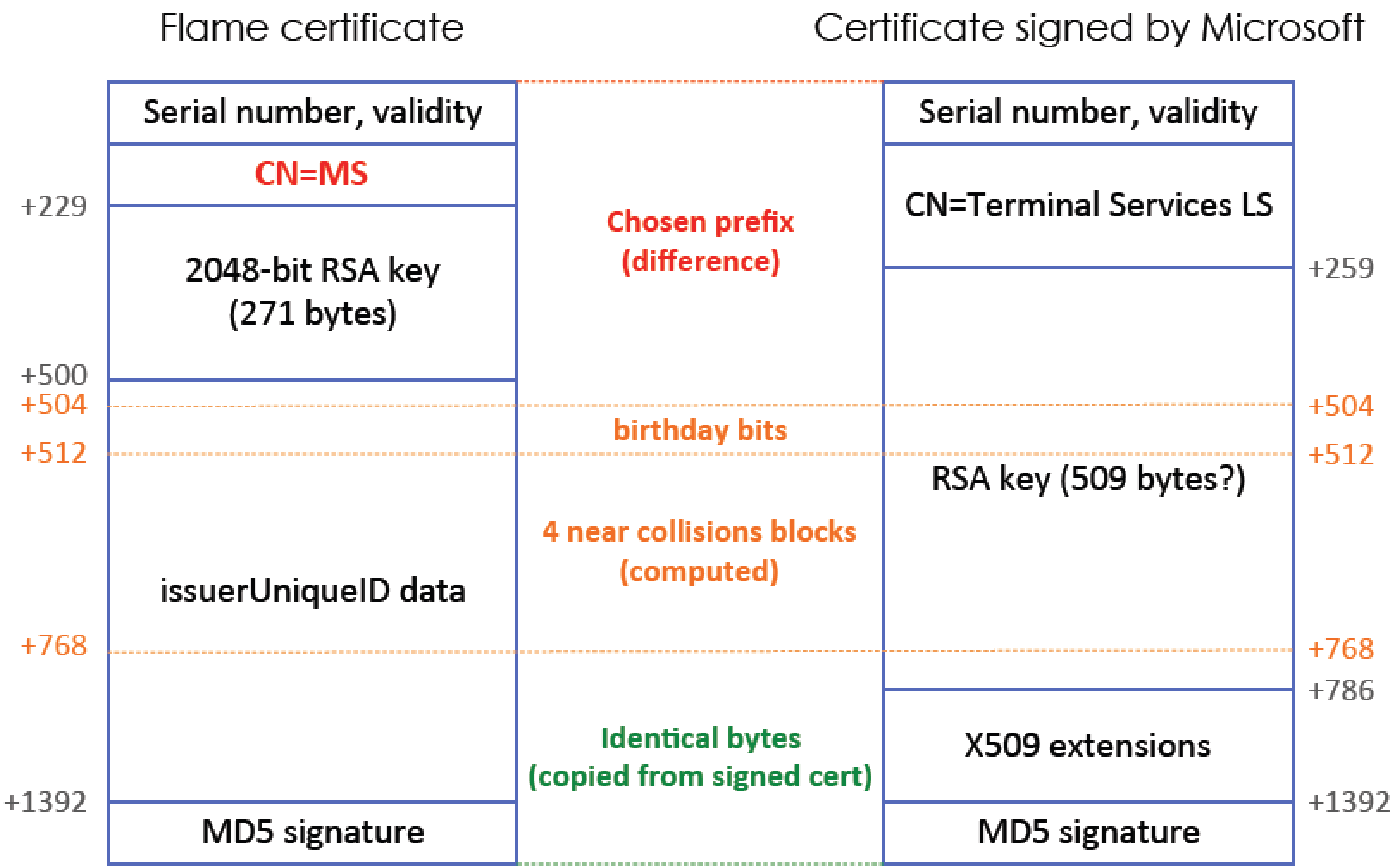


figure taken from a presentation of Alex Sotirov

Rapid SSL attack in 2008

- collaboration of hackers and academics led by Alex Sotirov and Marc Stevens
- demonstrated a practical MD5 collision attack against the RapidSSL CA
  - resulted in a fake SSL certificate trusted by all browsers
- **generating a collision required 2 days on a cluster of 200 PS3s**
  - **equivalent to about $20K computing capacity on Amazon EC2**

source: presentation of Alex Sotirov

Challenges for the attackers

- serial number and validity values are not controlled by the attacker
  - they depend on the issue time of the certificate
- the attackers needed to get the certificate issued at the right moment (matching the preset serial number and validity)
  - they had a 1-millisecond window to submit their query
  - probably large number of attempts were required
- questions
  - Did the attackers have a fast collision generation algorithm or a large cluster for computations?
  - Were they located close to Microsoft’s certificate server to minimize packet jitter?

MiniDuke (February 2013)

- targeted multiple government entities and human right organisations
  - governmental victims have been identified in Ukraine, Belgium, Portugal, Romania, the Czech Republic, and Ireland
- uses targeted pdf documents as droppers
  - exploits the Acrobat Reader 0-day vulnerability that was published by FireEye on February 12, 2013
- drops a highly customized backdoor written in assembly language
  - very small in size (only 20kb) and unique for all victims
- this backdoor uses Twitter and Google to locate C&C servers
  - The weather is good today. Sunny! uri!wp07VkkxYt3Mne5uiDkz4Il/Iw48Ge/EWg==
- downloads stage 2 and 3 codes from C&C server disguised as GIF files

Lessons learned

- current practice to defend systems against targeted attacks have limitations
  - Duqu, Flame, and MiniDuke passed signature based virus scanners
  - code signing did not help either
    - compromised signature key in case of Duqu (and Stuxnet)
    - fake certificate for signature verification key in case of Flame
- the attackers are in possession of advanced cryptographic knowledge
  - MD5 collision attack in Flame
- vigilant system administrators could detect advanced attacks with simple approaches and available tools
  - manual inspection of system status, resource usage, and bootlog
  - available rootkit detector tools

Lessons learned

- mainstream security products are bypassed by attackers
  - McAfee, TrendMicro, Sysinternals tools failed to detect Duqu and Flame
  - attackers can fine-tune their code until it pass mainstream products
- information sharing is crucial for identification of droppers (and potentially 0-day exploits)
  - we played the role of a trusted mediator between Microsoft and the Duqu victim
  - this led to efficient handling of the incident and the discovery of a 0-day Windows kernel exploit
  - unlike in our case, security vendors are often unable to obtain forensics information even when their product detected infection
    - droppers of Flame?

Challenge #1: Prevention

- signature based scanning does not work
  - assumes that AV company already detected the threat before it affects the user
  - not true for targeted attacks (you may be the first and only victim)
  - polymorphism allows for individualized malware samples
  - attacker can test AV products available on the market
- on-the-fly dynamic analysis of behavior?
  - open document / download web page in closely monitored virtual machine
  - examples: FireEye, LastLine
  - problems:
    - malware can detect the virtualized environment
    - ensuring transparency is very difficult
- sandboxing and virtualization?
  - create isolated execution environments and monitor accesses through their boundary to system resources
  - examples: Java VM, VMware, VirtualBox, Bromium vSentry, Qubes OS
  - problems:
    - usability, convenience, and performance issues

Challenge #2: Fast detection

- determined and resourceful attackers will eventually succeed in compromising any system
- detection of being compromised can focus on lateral movement, C&C communications, and data exfiltration
- Security Information and Event Management (SIEM) systems?
  - collect and correlate event logs
  - raise alarms
  - problems:
    - false positives
    - need expert knowledge to configure properly
- defender can also take advantage of the home ground
  - can use traps and baits to mislead the attacker
  - examples: honeypots and honeytokens
  - also allow for observation of attacker activities, tools, and tactics
  - problems:
    - management of honeypots and honeytokens can be cumbersome
    - is this only a question of lacking good management tools?

Challenge #3: Information asymmetry

attackers have knowledge on available security tools, and they may also know the security posture of the defender

⇕

defenders know very little about the attacker

- how can we decrease this asymmetry?

- hide some part of the security posture
  - custom security tools and configurations
  - honeypots and honeytokens could also be used here

- try to obtain more information about the attacker
  - threat intelligence gathering from different sources
  - client honeypots

Challenge #4: Information sharing

- victims of attacks are reluctant to share incident information and forensics material
  - privacy issues, sensitive company related information
- security vendors could collect lot of information on potential new threats
  - e.g., silent detection by AV vendors, Cisco SenderBase, cloud based protection by SIEM vendors, ...
- but security vendors may not want to share their collected intelligence to preserve market advantage
- standards for exchange of incident data are emerging
  - STIX - Structured Threat Information Expression
  - TAXII - Trusted Automated Exchange of Indicator Information
- legal issues may impede data sharing
  - incident reports, traffic logs, DNS data include personally identifiable information

Challenge #5: Lack of good experts

- a good expert
  - knows about old tricks and new trends in cyber attacks
  - knows the systems and networks he is responsible for
  - can use and configure available security tools
  - can select new security tools to buy
  - has time to look into the tools' output

- better education, training, simulation exercises are needed

Questions?

Dr. Levente Buttyán

CrySyS Lab, Budapest

contact info: www.crysys.hu