A Connection Pattern-based Approach to Detect Network Traffic Anomalies in Critical Infrastructures

Béla Genge[1], **Dorin Adrian Rusu**[2], **Piroska Haller**[1]

[1] "Petru Maior" University of Tîrgu Mureș, Romania
[2] VU University Amsterdam, The Netherlands
e-mail: bela.genge@ing.upm.ro, d.rusu@student.vu.nl, phaller@upm.ro

April 13, 2014

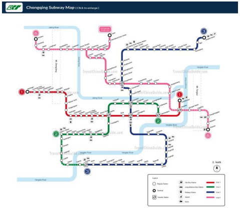
Outline

- Introduction - Critical Infrastructures
- Research motivation
- Proposed approach: SPEAR
- Experimental assessment
- Conclusions and future work

Critical Infrastructures (CI)

The term Critical Infrastructure (CI) underlines the significance of an infrastructure, which *"if disrupted or destroyed, would have a serious impact on the health, safety, security or economic well-being of citizens"*[1]

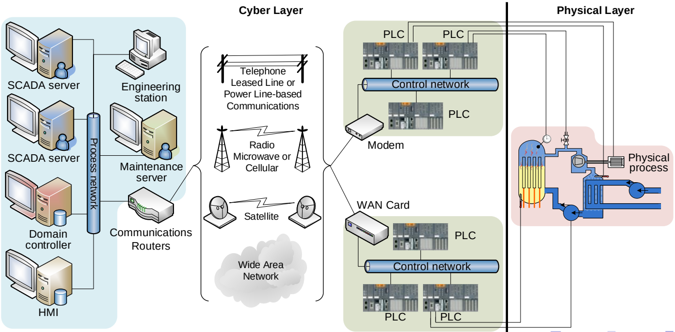
© Can Stock Photo - csp12198429



[1] Communication from the Commission to the Council - Critical Infrastructure Protection in the fight against terrorism. COM(2004)0702., October 2004.

Industrial Control Systems (ICS): the core of CI

- Architecture includes the cyber and physical domains
- Typical components:
  - The physical process: power plant, chemical process, electricity grid
  - Programmable Logical Controllers (PLC)
  - Master Terminal Units (MTU - SCADA servers)
  - Human Machine Interfaces (HMI)
  - Communication infrastructure



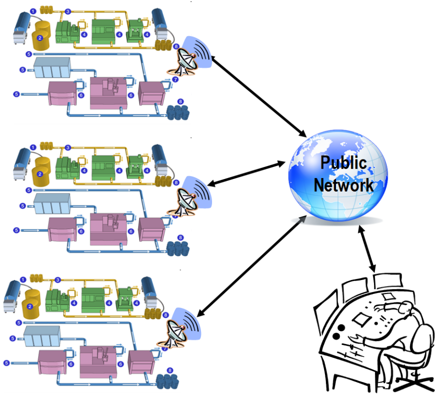
ICS networks vs traditional computer networks[2]

- ICS networks are connected to physical equipment: failure of industrial networks can have severe repercussions
- ICS networks have strong determinism (transmission and reply are predictable)
- ICS communicating nodes are well-known
- ICS include strict real-time requirements, e.g., response time less than 1ms
- ICS installations have longer lifetimes (at least ten years, compared to three years for traditional)

[2] B. Galloway and G.P. Hancke, *Introduction to Industrial Control Networks*, IEEE Communication Surveys & Tutorials, 15(2):860-880, 2013.

Industrial Control Systems (ICS): today

- Adoption of Ethernet and IP-based protocols
- COTS hardware and software
- Advantages: new services and features, remote monitoring and maintenance, energy markets, the newly emerging smart grids

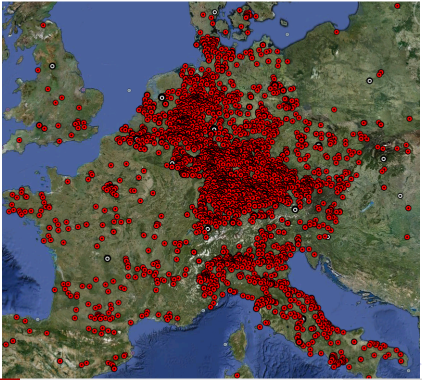


ICS security today

- ICS are not isolated environments
- Traditional ICT hardware and software has been strongly integrated into ICS
- Example security concerns:
  - Old operating systems (Windows NT 3.0/4.0, Windows 2000, BSD)
  - Rare patching
  - Low "ICT security perception"
- ICS are typically prone to traditional ICT attacks (Code RED, NIMDA, SLAMMER)

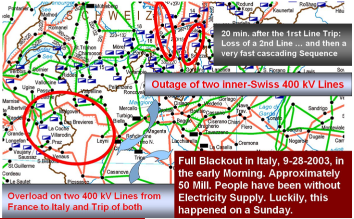
ICS security today

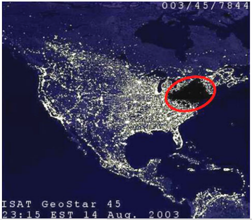
- Unfortunately many ICS components are directly accessible from Internet (see Shodan queries)
- Researchers from Free University Berlin have provided a map with SCADA devices connected to the Internet
- Project SHINE discovered more than 1,000,000 SCADA devices connected to the Internet



Cyber attack impact on ICS

- In 2007, the potential impact of cyber attacks has been highlighted by the Tempe, Arizona incident (improper configuration of load shedding programs) - result: 141 breakers were opened and there was significant loss of load (46 minutes power outage)
- In August 2010 the discovery of a new kind of malware (Stuxnet) constituted a turning point in ICS security - result: more than 100,000 infected stations, target: nuclear enrichment centrifuges
- Early October 2012 a power company reported a virus infection (variant of Mariposa) in a turbine control system - result: downtime for 3 weeks





However ....

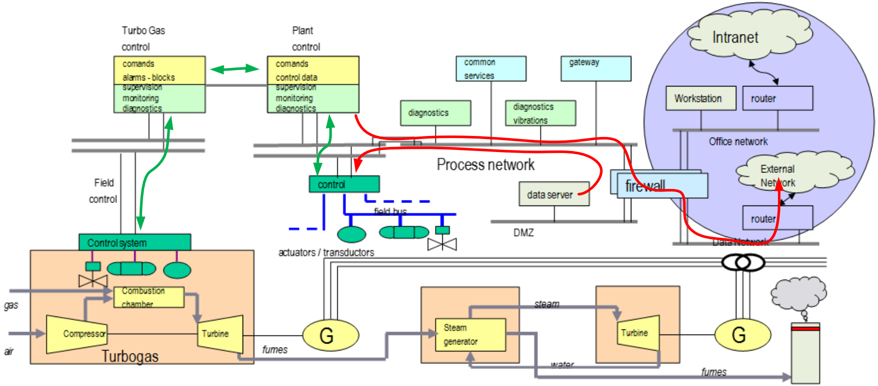
- Traditional ICT shields
- New mitigation techniques
- New policies

In this paper: SPEAR

- SPEAR: systematic approach aimed at modeling the topology of ICS and automatically generating Snort detection rules
- The approach is based on the following assumptions:
  - ICS architectures, once deployed, remain fixed over long time periods (more than 10 years)
  - Communication flows exhibit long-lasting patterns, e.g., **connection patterns**

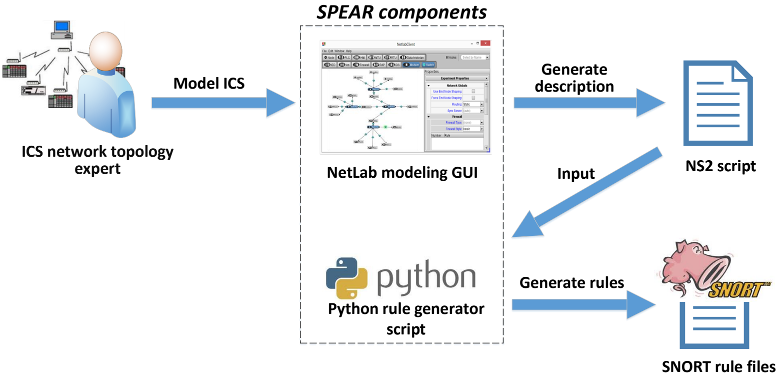
Example ICS architecture and communication flows

- Turbo-Gas power plant
- Green arrows denote allowed communication
- Red arrows denote abnormal communication



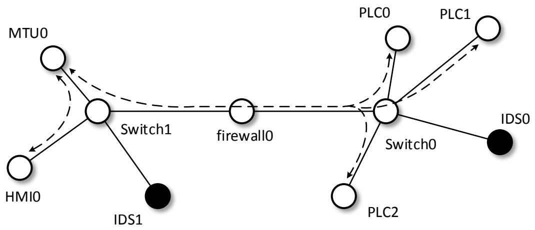
Steps defined in SPEAR

- Step 1: modeling the network of ICS (nodes and traffic flows)
- Step 2: generating Snort detection rules

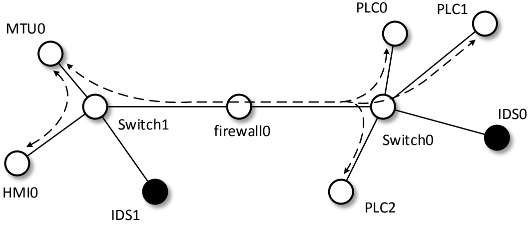


ICS model

- Network architecture as a traditional graph model $G = (V, E)$, where V is the set of *vertices* and $E \subseteq V \times V$ is the set of *edges*
  - Each vertex models typical ICS nodes such as PLC, HMI, RTU, ADS
  - Edges denote typical connections between network components, e.g., wired/wireless links
- Traffic defined as tuple $t = (s, d, k), t \in T$, where $T \subseteq V \times V \times \{\texttt{tcp}, \texttt{udp}\}$



Generating ICS rules



- A breadth-first search (BFS) algorithm is applied to find the path from source to destination (for each traffic flow)
- For each ADS along the path Snort rules are generated to whitelist allowed traffic

Generating ICS rules (contd.)

- Using BFS algorithm, for each traffic flow t the list of ADSs are determined (set A)
- For each ADS $a \in A$ the set of traffic flows is calculated $F^a = \bigcup \{t | t \in T \ and \ a \in adspath(t)\}$
- The set of rules for each ADS $a \in A$ is denoted by R^a
- Rules are generated for Snort. Example:

alert tcp 10.1.1.1 any − > 10.1.1.2 any (msg: "ALERT!")

Generating ICS rules (contd.)

- Generated rule 1 (for bidirectional UDP/TCP communications):

$$(\{k\}, \{v\}, NOT(H_v^a), \texttt{anyp}, \texttt{anyp}, \texttt{ALERT!}) \Rightarrow R^a$$

- $k \in \{\texttt{tcp}, \texttt{udp}\}$: protocol
- $v \in V$: host
- H_v^a: the set of hosts that exchange packets with host v, monitored by ADS a
- ... meaning: generate alert if host v exchanges TCP/UDP packets with any host outside H_v^a

Generating ICS rules (contd.)

- Generated rule 2 and rule 3 (for unidirectional UDP or no TCP packets):

$$(\{k\}, \{v\}, NOT(\{v\}), \texttt{anyp}, \texttt{anyp}, \texttt{ALERT!}) \Rightarrow R^a$$
$$(\{k\}, NOT(\{v\}), \{v\}, \texttt{anyp}, \texttt{anyp}, \texttt{ALERT!}) \Rightarrow R^a$$

- $k \in \{\texttt{tcp}, \texttt{udp}\}$: protocol
- $v \in V$: host
- ... meaning: generate alert if host v sends or receives TCP/UDP packets to/from any host

Generating ICS rules (contd.)

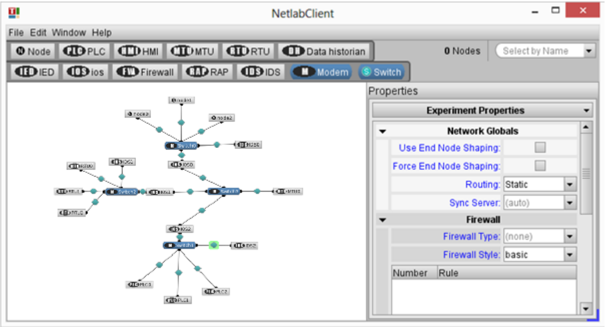
- Generated rule 4 (no UDP/TCP packets):

$$(\{k\}, \{NOT(V)\}, \{NOT(V)\}, \texttt{anyp}, \texttt{anyp}, \texttt{ALERT!}) \Rightarrow R^a$$

- $k \in \{\texttt{tcp}, \texttt{udp}\}$: protocol
- ... meaning: generate alert if any other hosts (outside the monitored set) exchange TCP or UDP packets

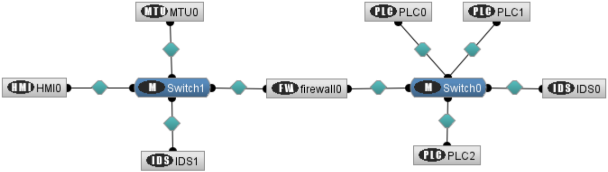
Implementation details

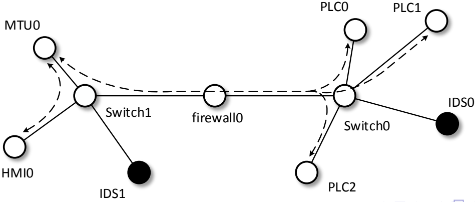
- We adopted the Emulab NetLab GUI
  - Was developed within the Emulab project
  - SPEAR extends the basic GUI with components specific to ICS



Implementation details - example

- We defined ICS with process and control network
- Process network: HMI, MTU and ADS
- Control network: three PLCs and ADS
- TCP traffic





Implementation details - example (contd.)

- Typical Emulab ns-2 script

```
set ns [new Simulator]
source tb_compat.tcl
# Nodes
set MTU0 [$ns node]
tb-set-node-os $MTU0 ncSCADA-MTU
set IDS0 [$ns node]
tb-set-node-os $IDS0 ncSCADA-IDS
# Lans
set Switch0 [$ns make-lan "$firewall0 $IDS0 $PLC0 ..."]
set Switch1 [$ns make-lan "$firewall0 $HMI0 $IDS1 $MTU0"]
# Event Agents
set tg0 [new Application/Traffic/CBR]
set tg0sink0 [new Agent/TCPSink]
$ns attach-agent $MTU0 $tg0src0
...
$ns run
```

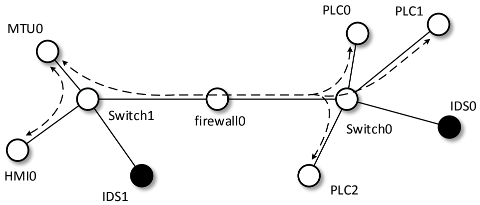
Implementation details - example (contd.)

- Generated Snort rules

ipvar $MTU0 [10.1.1.2]

...

1. alert tcp $MTU0 any − > ![$PLC0,$PLC1,$PLC2,$HMI0] any (...)
2. alert tcp ![$PLC0,$PLC1,$PLC2,$HMI0] any − > $MTU0 any (...)
3. alert tcp $HMI0 any − > !$MTU0 any (...)
4. alert tcp !$MTU0 any − > $HMI0 any (...)
5. alert tcp $firewall0 any − > !$firewall0 any (...)
6. alert tcp !$firewall0 any − > $firewall0 any (...)
7. alert tcp ![$MTU0 $PLC0 ...] any − > ![$MTU0 $PLC0 ...] any (...)
8. alert udp any any − > any any (...)

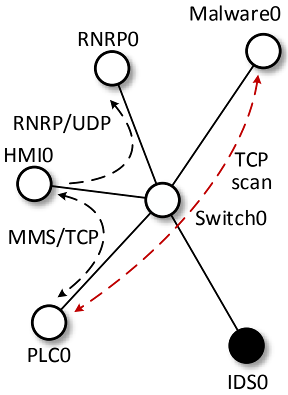


Assessment perspectives

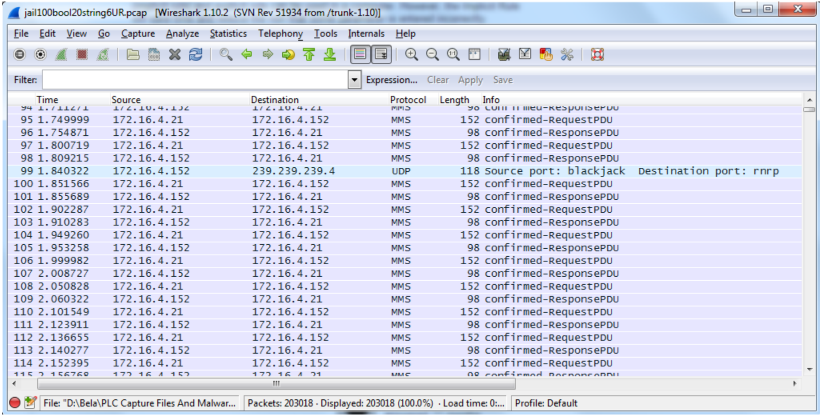
- SPEAR and its generated rules have been assessed from several perspectives:
  - Modeling and generating rules for a laboratory installation with industrial equipment (synthetic attack)
  - Modeling and generating rules for a laboratory installation with traditional PCs (real malware)
  - Modeling and generating rules for simulated infrastructures (synthetic attack)
  - Scalability and execution time of rule generator script

Real industrial equipment + synthetic attack

- We have set-up an experiment consisting of a PLC and HMI software from ABB
- HMI communicated with PLC through Manufacturing Message Specification protocol (MMS)
- HMI also sends Redundant Network Routing Protocol (RNRP) packets over UDP to a specific router
- Generic host to run TCP scans (TCP-SYN, TCP-NULL, TCP-FIN, and TCP-XMAS) with *nmap* software



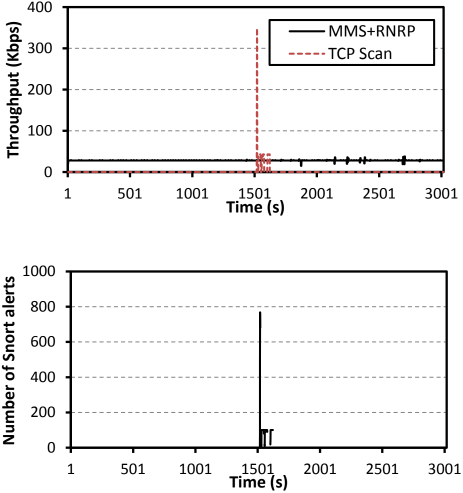
Real industrial equipment + synthetic attack (contd.)



No.	Time	Source	Destination	Protocol	Length	Info
95	1.749999	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
96	1.754871	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
97	1.800719	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
98	1.809215	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
99	1.840322	172.16.4.152	239.239.239.4	UDP	118	Source port: blackjack Destination port: rnrp
100	1.851566	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
101	1.855689	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
102	1.902287	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
103	1.910283	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
104	1.949260	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
105	1.953258	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
106	1.999982	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
107	2.008727	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
108	2.050828	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
109	2.060322	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
110	2.101549	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
111	2.123911	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
112	2.136655	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU
113	2.140277	172.16.4.152	172.16.4.21	MMS	98	confirmed-ResponsePDU
114	2.152395	172.16.4.21	172.16.4.152	MMS	152	confirmed-RequestPDU

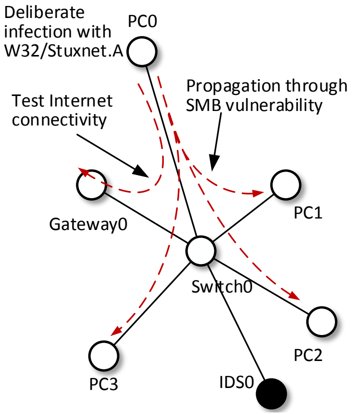
Real industrial equipment + synthetic attack (contd.)

- Detection of the network scan with the rules generated by SPEAR



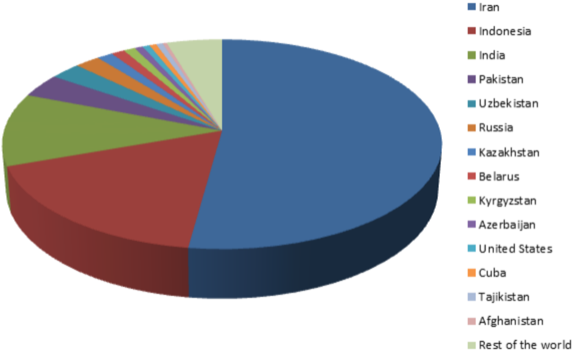
Traditional PCs + real industry-targeting malware

- We set-up a network with 4 PCs and a monitoring box from HP-S8005F (can monitor up to 16 ports)
- We deliberately infected one of the hosts with Stuxnet
- Stuxnet “installed” successfully
- Stuxnet infected (after 8 hours) all other hosts



Stuxnet - overview

- It was reported in August 2010
- The first (known) malware capable to rewrite the logic of control hardware (Siemens PLCs)
- It is believed that the target was Iran's nuclear program
- It affected the normal functioning of centrifuges

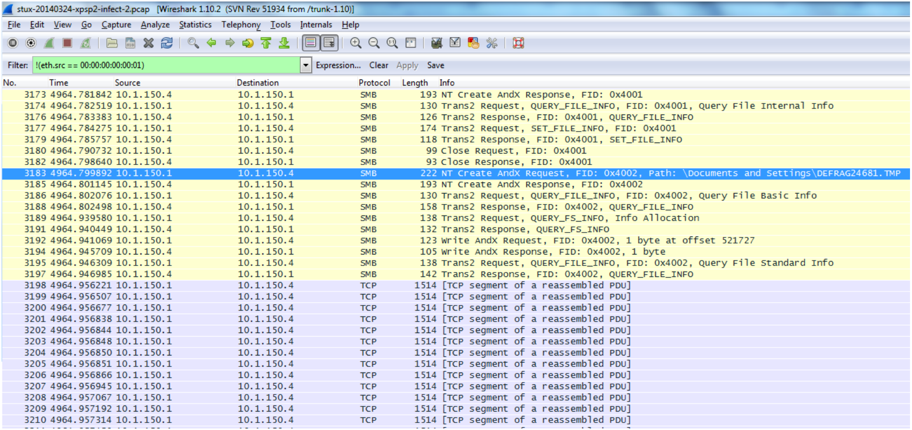


Stuxnet - zero-day vulnerabilities

- It exploited 4 zero-day vulnerabilities
  - Exploit LNK vulnerability MS10-046 (Windows 2000, Windows Server 2003 and 2008, Windows Vista, Windows XP, Windows 7)
  - Exploit MS Spooler vulnerability MS10-061 (Windows 2000, Windows Server 2003 and 2008, Windows Vista, Windows XP, Windows 7)
  - **Exploit Network Shared Folders and RPC vulnerability MS08-067** (Windows 2000, Windows Server 2003 and 2008, Windows Vista, Windows XP)
  - Exploit `win32k.sys` vulnerability MS10-73 (Windows Server 2003 and 2008, Windows Vista, Windows XP, Windows 7)

Traditional PCs + real industry-targeting malware (contd.)

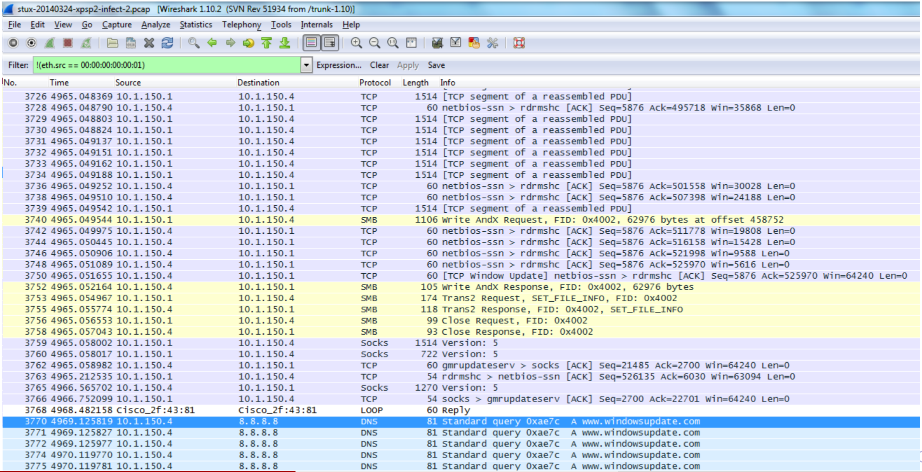
- Stuxnet creates remote file DEFRAG24681.TMP
- It copies itself on the other host



No.	Time	Source	Destination	Protocol	Length	Info
3173	4964.781842	10.1.150.4	10.1.150.1	SMB	193	NT Create AndX Response, FID: 0x4001
3174	4964.782519	10.1.150.1	10.1.150.4	SMB	130	Trans2 Request, QUERY_FILE_INFO, FID: 0x4001, Query File Internal Info
3176	4964.783383	10.1.150.4	10.1.150.1	SMB	126	Trans2 Response, FID: 0x4001, QUERY_FILE_INFO
3177	4964.784275	10.1.150.1	10.1.150.4	SMB	174	Trans2 Request, SET_FILE_INFO, FID: 0x4001
3179	4964.785757	10.1.150.4	10.1.150.1	SMB	118	Trans2 Response, FID: 0x4001, SET_FILE_INFO
3180	4964.790732	10.1.150.1	10.1.150.4	SMB	99	Close Request, FID: 0x4001
3182	4964.798640	10.1.150.4	10.1.150.1	SMB	93	Close Response, FID: 0x4001
3183	4964.799892	10.1.150.1	10.1.150.4	SMB	222	NT Create AndX Request, FID: 0x4002, Path: \Documents and Settings\DEFRAG24681.TMP
3185	4964.801145	10.1.150.4	10.1.150.1	SMB	193	NT Create AndX Response, FID: 0x4002
3186	4964.802076	10.1.150.1	10.1.150.4	SMB	130	Trans2 Request, QUERY_FILE_INFO, FID: 0x4002, Query File Basic Info
3188	4964.802498	10.1.150.4	10.1.150.1	SMB	158	Trans2 Response, FID: 0x4002, QUERY_FILE_INFO
3189	4964.939580	10.1.150.1	10.1.150.4	SMB	138	Trans2 Request, QUERY_FS_INFO, Info Allocation
3191	4964.940449	10.1.150.4	10.1.150.1	SMB	132	Trans2 Response, QUERY_FS_INFO
3192	4964.941069	10.1.150.1	10.1.150.4	SMB	123	Write AndX Request, FID: 0x4002, 1 byte at offset 521727
3194	4964.945709	10.1.150.4	10.1.150.1	SMB	105	Write AndX Response, FID: 0x4002, 1 byte
3195	4964.946309	10.1.150.1	10.1.150.4	SMB	138	Trans2 Request, QUERY_FILE_INFO, FID: 0x4002, Query File Standard Info
3197	4964.946985	10.1.150.4	10.1.150.1	SMB	142	Trans2 Response, FID: 0x4002, QUERY_FILE_INFO
3198	4964.956221	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3199	4964.956507	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3200	4964.956677	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3201	4964.956838	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3202	4964.956844	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3203	4964.956848	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3204	4964.956850	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3205	4964.956851	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3206	4964.956866	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3207	4964.956945	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3208	4964.957067	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3209	4964.957192	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3210	4964.957314	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]

Traditional PCs + real industry-targeting malware (contd.)

- The newly infected host begins to test for Internet connectivity (`www.windowsupdate.com` and `www.msn.com`)



stux-20140324-xpsp2-infect-2.pcap [Wireshark 1.10.2 (SVN Rev 51934 from /trunk-1.10)]

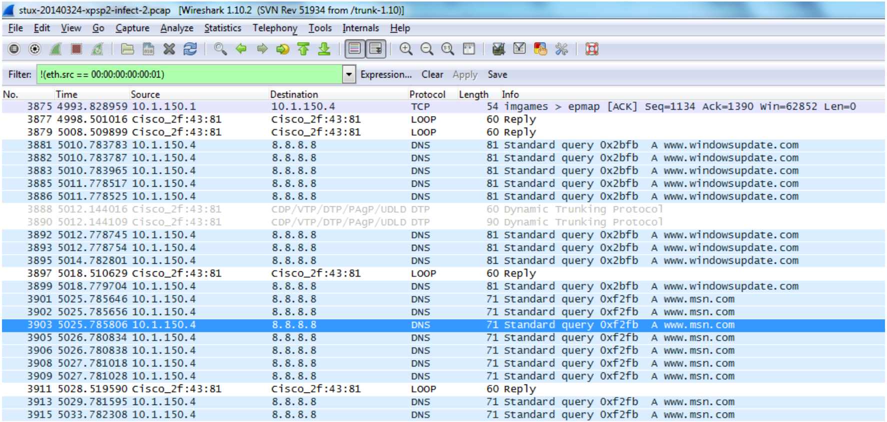
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: !(eth.src == 00:00:00:00:00:01) Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
3726	4965.048369	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3728	4965.048790	10.1.150.4	10.1.150.1	TCP	60	netbios-ssn > rdrmshc [ACK] Seq=5876 Ack=495718 Win=35868 Len=0
3729	4965.048803	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3730	4965.048824	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3731	4965.049137	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3732	4965.049151	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3733	4965.049162	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3734	4965.049188	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3736	4965.049252	10.1.150.4	10.1.150.1	TCP	60	netbios-ssn > rdrmshc [ACK] Seq=5876 Ack=501558 Win=30028 Len=0
3738	4965.049510	10.1.150.4	10.1.150.1	TCP	60	netbios-ssn > rdrmshc [ACK] Seq=5876 Ack=507398 Win=24188 Len=0
3739	4965.049542	10.1.150.1	10.1.150.4	TCP	1514	[TCP segment of a reassembled PDU]
3740	4965.049544	10.1.150.1	10.1.150.4	SMB	1106	Write AndX Request, FID: 0x4002, 62976 bytes at offset 458752
3742	4965.049975	10.1.150.4	10.1.150.1	TCP	60	netbios-ssn > rdrmshc [ACK] Seq=5876 Ack=511778 Win=19808 Len=0
3744	4965.050445	10.1.150.4	10.1.150.1	TCP	60	netbios-ssn > rdrmshc [ACK] Seq=5876 Ack=516158 Win=15428 Len=0
3746	4965.050906	10.1.150.4	10.1.150.1	TCP	60	netbios-ssn > rdrmshc [ACK] Seq=5876 Ack=521998 Win=9588 Len=0
3748	4965.051089	10.1.150.4	10.1.150.1	TCP	60	netbios-ssn > rdrmshc [ACK] Seq=5876 Ack=525970 Win=5616 Len=0
3750	4965.051655	10.1.150.4	10.1.150.1	TCP	60	[TCP Window Update] netbios-ssn > rdrmshc [ACK] Seq=5876 Ack=525970 Win=64240 Len=0
3752	4965.052164	10.1.150.4	10.1.150.1	SMB	105	Write AndX Response, FID: 0x4002, 62976 bytes
3753	4965.054967	10.1.150.1	10.1.150.4	SMB	174	Trans2 Request, SET_FILE_INFO, FID: 0x4002
3755	4965.055774	10.1.150.4	10.1.150.1	SMB	118	Trans2 Response, FID: 0x4002, SET_FILE_INFO
3756	4965.056553	10.1.150.1	10.1.150.4	SMB	99	Close Request, FID: 0x4002
3758	4965.057043	10.1.150.4	10.1.150.1	SMB	93	Close Response, FID: 0x4002
3759	4965.058002	10.1.150.1	10.1.150.4	Socks	1514	Version: 5
3760	4965.058017	10.1.150.1	10.1.150.4	Socks	722	Version: 5
3762	4965.058982	10.1.150.4	10.1.150.1	TCP	60	gmrupdateserv > socks [ACK] Seq=21485 Ack=2700 Win=64240 Len=0
3763	4965.212535	10.1.150.1	10.1.150.4	TCP	54	rdrmshc > netbios-ssn [ACK] Seq=526135 Ack=6030 Win=63094 Len=0
3765	4966.565702	10.1.150.4	10.1.150.1	Socks	1270	Version: 5
3766	4966.752099	10.1.150.1	10.1.150.4	TCP	54	socks > gmrupdateserv [ACK] Seq=2700 Ack=22701 Win=64240 Len=0
3768	4968.482158	Cisco_2f:43:81	Cisco_2f:43:81	LOOP	60	Reply
3770	4969.125819	10.1.150.4	8.8.8.8	DNS	81	Standard query 0xae7c A www.windowsupdate.com
3771	4969.125827	10.1.150.4	8.8.8.8	DNS	81	Standard query 0xae7c A www.windowsupdate.com
3772	4969.125977	10.1.150.4	8.8.8.8	DNS	81	Standard query 0xae7c A www.windowsupdate.com
3774	4970.119770	10.1.150.4	8.8.8.8	DNS	81	Standard query 0xae7c A www.windowsupdate.com
3775	4970.119781	10.1.150.4	8.8.8.8	DNS	81	Standard query 0xae7c A www.windowsupdate.com

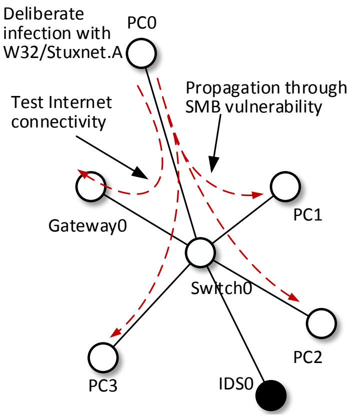
Traditional PCs + real industry-targeting malware (contd.)

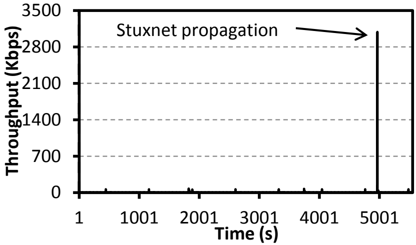
- The newly infected host begins to test for Internet connectivity (`www.windowsupdate.com` and `www.msn.com`)

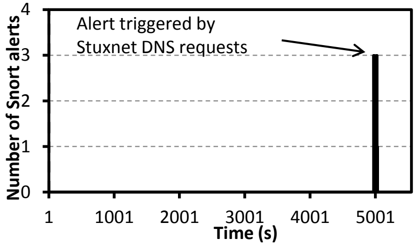


No.	Time	Source	Destination	Protocol	Length	Info
3875	4993.828959	10.1.150.1	10.1.150.4	TCP	54	imgames > epmap [ACK] Seq=1134 Ack=1390 Win=62852 Len=0
3877	4998.501016	Cisco_2f:43:81	Cisco_2f:43:81	LOOP	60	Reply
3879	5008.509899	Cisco_2f:43:81	Cisco_2f:43:81	LOOP	60	Reply
3881	5010.783783	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3882	5010.783787	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3883	5010.783965	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3885	5011.778517	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3886	5011.778525	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3888	5012.144016	Cisco_2f:43:81	CDP/VTP/DTP/PAgP/UDLD	DTP	60	Dynamic Trunking Protocol
3890	5012.144109	Cisco_2f:43:81	CDP/VTP/DTP/PAgP/UDLD	DTP	90	Dynamic Trunking Protocol
3892	5012.778745	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3893	5012.778754	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3895	5014.782801	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3897	5018.510629	Cisco_2f:43:81	Cisco_2f:43:81	LOOP	60	Reply
3899	5018.779704	10.1.150.4	8.8.8.8	DNS	81	Standard query 0x2bfb A www.windowsupdate.com
3901	5025.785646	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com
3902	5025.785656	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com
3903	5025.785806	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com
3905	5026.780834	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com
3906	5026.780838	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com
3908	5027.781018	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com
3909	5027.781028	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com
3911	5028.519590	Cisco_2f:43:81	Cisco_2f:43:81	LOOP	60	Reply
3913	5029.781595	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com
3915	5033.782308	10.1.150.4	8.8.8.8	DNS	71	Standard query 0xf2fb A www.msn.com

Traditional PCs + real industry-targeting malware (contd.)

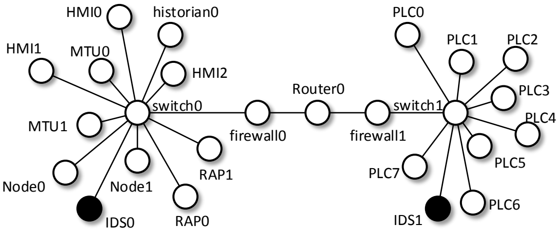






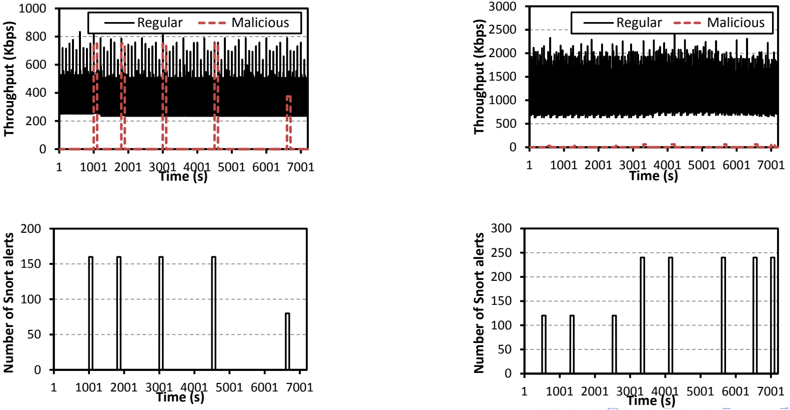
Simulated networks, traffic and attack

- We recreated in `ns-3` a (larger) typical ICS topology with two networks: control and process
- For each network we defined 10 nodes, regular UDP and malicious traffic
- We used SPEAR to model the topology and to generate detection rules
- The `ns-3` traffic was exported to `pcap` files and we used Snort (configured with SPEAR's rules) to detect the attack



Simulated networks, traffic and attack (contd.)

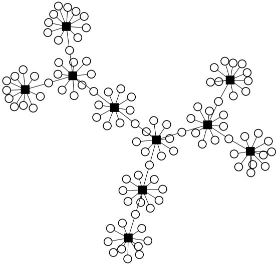
- Two settings:
  - Setting 1: attack rate similar to regular traffic
  - Setting 2: attack rate 30 times smaller than regular traffic

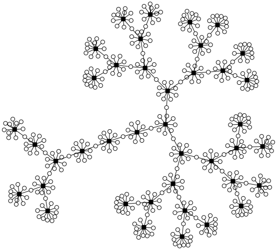


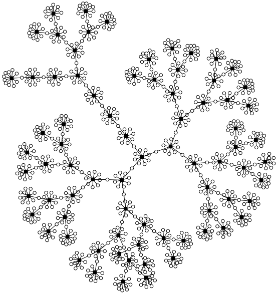
Rule generator execution time

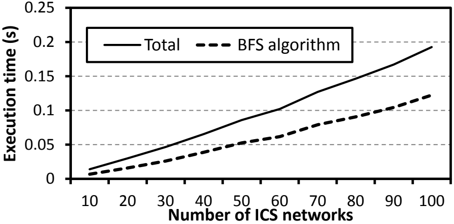
- We evaluated the execution time of SPEAR's rule generator
- We generated a total of 10 different topology descriptions (`ns-2`)
- For each topology the dimension was gradually increased with a number of 10 networks (10hosts + 1 ADS/network)
- Traffic was defined between hosts and between networks

Rule generator execution time (contd.)







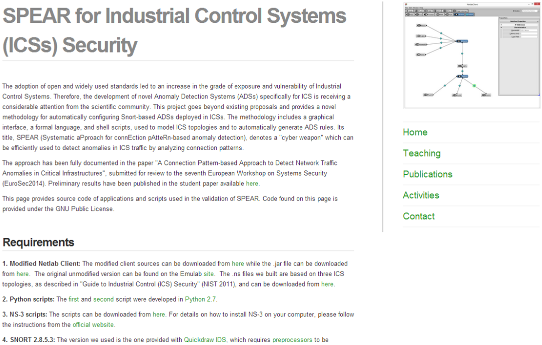


Conclusions

- The definition of connection patterns in the core of CI can lead to effective detection of traffic anomalies
- The learning phase from other approaches is replaced by expert knowledge and formal description language
- Detection rules are generated for a well-known detection engine: Snort
- SPEAR's main contribution:
    - It automatizes the rule generation procedure for ICSs and a well-established detection engine, i.e., Snort, by employing available open-source tools

Conclusions (contd.)

- Future work:
  - Extend the supported protocols for more expressive modeling capabilities
  - Integrate automated traffic learning techniques (carefully planned)
- SPEAR is available as open-source (`http://www.ibs.ro/~bela/conpat.html`)



SPEAR for Industrial Control Systems (ICSs) Security

The adoption of open and widely used standards led to an increase in the grade of exposure and vulnerability of Industrial Control Systems. Therefore, the development of novel Anomaly Detection Systems (ADSs) specifically for ICS is receiving a considerable attention from the scientific community. This project goes beyond existing proposals and provides a novel methodology for automatically configuring Snort-based ADSs deployed in ICSs. The methodology includes a graphical interface, a formal language, and shell scripts, used to model ICS topologies and to automatically generate ADS rules. Its title, SPEAR (Systematic aPproach for connEction pAtteRn-based anomaly detection), denotes a "cyber weapon" which can be efficiently used to detect anomalies in ICS traffic by analyzing connection patterns.

The approach has been fully documented in the paper "A Connection Pattern-based Approach to Detect Network Traffic Anomalies in Critical Infrastructures", submitted for review to the seventh European Workshop on Systems Security (EuroSec2014). Preliminary results have been published in the student paper available here.

This page provides source code of applications and scripts used in the validation of SPEAR. Code found on this page is provided under the GNU Public License.

Requirements

1. **Modified Netlab Client:** The modified client sources can be downloaded from here while the .jar file can be downloaded from here. The original unmodified version can be found on the Emulab site. The .ns files we built are based on three ICS topologies, as described in "Guide to Industrial Control (ICS) Security" (NIST 2011), and can be downloaded from here.

2. **Python scripts:** The first and second script were developed in Python 2.7.

3. **NS-3 scripts:** The scripts can be downloaded from here. For details on how to install NS-3 on your computer, please follow the instructions from the official website.

4. **SNORT 2.8.5.3:** The version we used is the one provided with Quickdraw IDS, which requires preprocessors to be

Home

Teaching

Publications

Activities

Contact

Thank you!