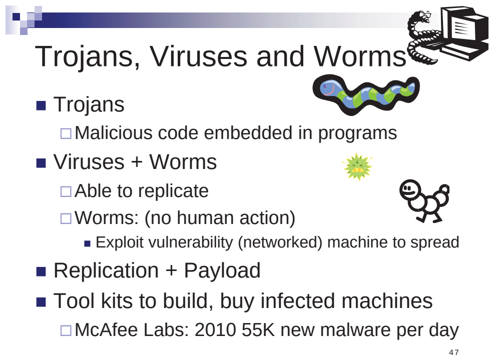
3.3 Malware

Above we have looked at some specific vulnerabilities of networks and the machines on a network. Different type of malware exploit such weaknesses to infiltrate the system, replicate, spread and achieve some malicious goal.

Trojans are legitimate looking programs (or other content) that actually carry malicious code inside. Viruses can replicate, usually involving some action like running an infected program (like biological viruses need a host organism, computer viruses infect programs). Worms are able to replicate by themselves, without the need of human action. Well known worms, such as conficker, spread around networks (e.g. the internet) exploiting vulnerabilities of network services and machines. Classification malware within these categories, however, is sometimes difficult and terms are commonly mixed, using e.g. virus for any type of malware.

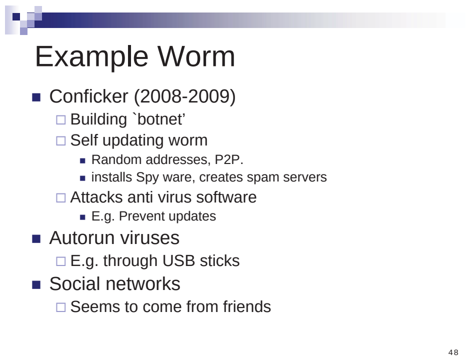
While some hacks, viruses and worms may have been ‘see what I can do’, idealistic, to demonstrate the vulnerability or simple vandalism, modern malware is for the most part big business or even digital warfare. The conficker worm, for instance, installs scareware (showing pop-ups to get user to buy a fake anti-virus solution) and creates a botnet; a network of machines under control of the attacker, which can then be used for sending spam, distributed denial of service attacks, renting out to others, etc. The worm has an update mechanism used to download its software for its malicious activities as well as updates to its spreading mechanism and protection against updates of anti-virus software that would be able to protect against it.



Trojans, Viruses and Worms

- Trojans
  - Malicious code embedded in programs
- Viruses + Worms
  - Able to replicate
  - Worms: (no human action)
    - Exploit vulnerability (networked) machine to spread
- Replication + Payload
- Tool kits to build, buy infected machines
  - McAfee Labs: 2010 55K new malware per day

47



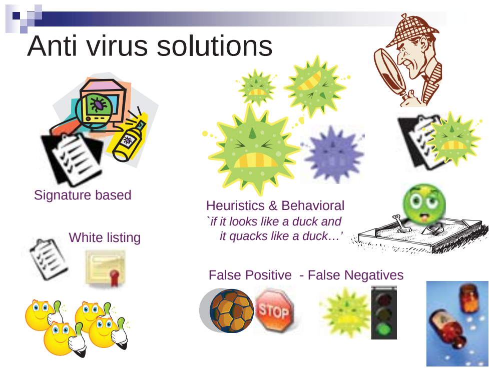
Example Worm

- Conficker (2008-2009)
  - Building `botnet'
  - Self updating worm
    - Random addresses, P2P.
    - installs Spy ware, creates spam servers
  - Attacks anti virus software
    - E.g. Prevent updates
- Autorun viruses
  - E.g. through USB sticks
- Social networks
  - Seems to come from friends

48

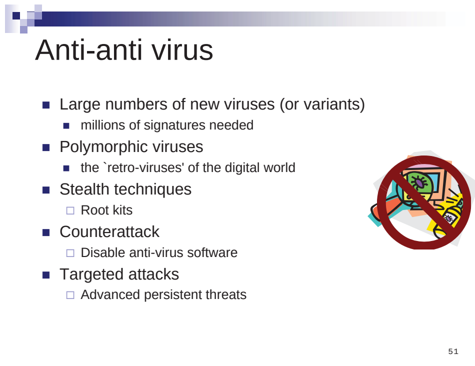
(Zero-day) vulnerabilities, exploits, virus building kits and botnets are commodities that are traded on the black market. One thus does not even have to create one’s own malware or botnet; it is possible to buy or rent infected machines. Botnets are controlled through command and control centers. By using a C&C center to drive a botnet, the malware can easily be updated and adapted, making it hard to take down the network of bots. To prevent the C&C center itself is taken down, it is located in countries where there are no laws to easily do this, its location is hidden e.g. within a list of addresses, using anonymous services in TOR (see lecture on privacy Chapter **??**), and/or redundancy is used so a new C&C can easily be created at a different location. As in many security areas there is an arms race between the taking out botnet C&C centers and new infections, botnets and control methods appearing.

Of course with all the value a botnet represents there are those that will try to take it over, either to dismantle/study it (e.g. `torpig-takeover`) or to use it for their them for illegitimate agenda. It has also been suggested to take this defense strategy a step further; use weaknesses in infected machine to force installation of patches, removing of and protecting against malware but there are many moral, legal and practical and technical issues with such an approach. Related to this is use of ‘hacking’ by authorities which is also the subject of debate what actions are justified, should e.g. ‘hacking’ by the police by allowed? (e.g. German federal court rejects hacking by police in general though leaving open the possibility to do so w.r.t. threats that e.g. endanger public safety such as terrorism. Proposal for a law in the Netherlands allowing the police to break into computers (including mobile phones), installing spyware and software to take control over the device, breaking into and search and destroy data on computers via the internet, even into computer is abroad. `www.nytimes.com/2011/10/15/world/europe/uproar-in-germany-on-police-use-of-surveillance-software.html` `http://jurist.org/thisday/2012/02/german-high-court-rejected-police-computer-hacking-efforts.php` `http://webwereld.nl/nieuws/112654/internationaal-verzet-tegen-terughackplan-opstelten.html` `http://www.nrc.nl/nieuws/2012/12/06/adviseurs-opstelten-hekelen-hackende-politie`)



Anti virus solutions

Signature based
White listing
Heuristics & Behavioral
`if it looks like a duck and it quacks like a duck…'
False Positive - False Negatives
STOP



Anti-anti virus

- Large numbers of new viruses (or variants)
  - millions of signatures needed
- Polymorphic viruses
  - the `retro-viruses' of the digital world
- Stealth techniques
  - Root kits
- Counterattack
  - Disable anti-virus software
- Targeted attacks
  - Advanced persistent threats

51

Anti virus software tries to identify malware and prevent it from causing harm. Typically be periodically scanning the whole system, checking content as it comes in (e.g. downloads in the browser, incomming email) and on access scan; programs get scanned on the fly as they are started. It recognizes malware in different ways; *signature based* recognition compares the scanned content againts the a list of known viruses. This requires constant updating of the list of 'fingerprints' of known viruses. Viruses not on the list yet will be missed. Also variations of the same virus may be missed if the characteristic fingerprint is changed/masked.

The number of viruses is huge and growing quickly; current estimated of number of signatures needed to detect malware exceeds 20.5 million (`http://www.triumfant.com/`, Feb 2012).

Though virus software uses several mechanisms to reduce the performance cost, such as checking for changes (storing e.g. a CRC of a scanned program) and not rechecking unchanged programs, purely signature based detection becomes unfeasible with such numbers.

It is not possible to positively/exactly decide whether content is 'malicious' but if violates some rules and/or looks like known viruses either in code or in behaviour this may indicate that it is indeed malicious. *Heuristic* recognition try to recognize malware; running the program in a sandbox to examine its behaviour or decompiling to look whether the code contains characteristic malware patterns.

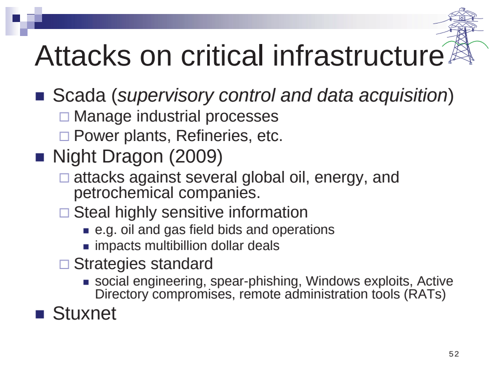
The approaches above are blacklisting approaches; trying to track and recognize the 'bad' cases. There is some problem with false positives; some legitimate programs may match the signature or heuristic rules but the false negative problem is much bigger; e.g. new unknown malware cannot be detected. White listing approaches take the opposite approach; they identify the 'good' pieces of software that are allowed to run. Where blacklisting suffers mainly from false negatives, whitelisting suffers from false positives. Any legitimate program not marked as 'good' will raise an alert, likely leading to a 'the boy who cried wolf' problem. Presenting too many alters to the user will be at least inconvenient and may lead the user to ignore the alerts (or turn of the virus software altogether). Where the problem for blacklisting is keeping track of everything bad, the problem for whitelisting is keeping track of everything good. How would you check whether a program is good? You could start looking up information about the program online; looking for an expert that evaluates it or failing that for the opinion of other users. These approach are also used by anti-malware programs. They can check whether a piece of code is validated by an authority; e.g. on a virus software maker whitelist, certified drivers, etc. Reputation based systems look at how long has the program been around, how often is it used, where and in which context, based on feedback from millions of machines (note the potential privacy risk).

Polymorphic viruses, the 'retro-viruses' of the digital world, try to prevent detection by anti-Virus software by mutating, on occation changing its appearance in new copies. In this way a new signature may no longer be valid. One way of mutating is to encrypt the payload with a (new randomly generated) key (the key used will have to be included in the code to be able to decrypt but a basic pattern analysis will not reveal this). By not mutating too often it also makes the job for analysts harder by not providing many different copies of the same malware.

Stealth techniques employed by viruses to hide from virus scanners include intercepting read requests and replacing reads of infected files by clean copies. Related to this is rootkits; software which operates 'at the root of the system'; it is not visible at the operating system level. Virus protection that uses OS calls will not be able to detect such viruses.

Other viruses take a more active approach actually attacking the anti-virus software; stopping the processes belonging to such software, making anti-virus (update) sites unreachable (e.g. by altering DNS settings).

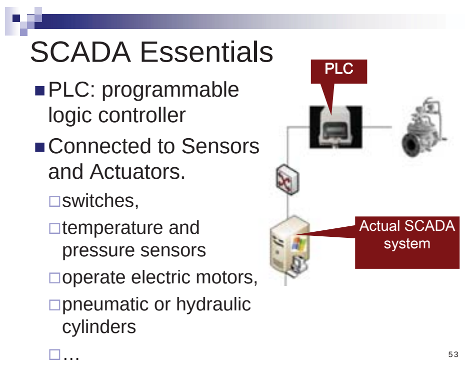
An important trend in the malware landscape is the move from 'blunt tools' doing 'catch what you can' attacks against arbitrary targets to sophisticated, specialized attacks against specific targets; e.g. for data theft, (industrial) espionage. Such Advanced Persistent Threats (APTs) are at the 'frontline' of the (anti-)malware armsrace.



Attacks on critical infrastructure

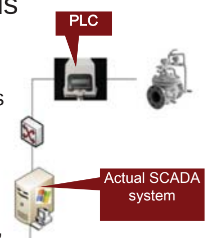
- Scada (*supervisory control and data acquisition*)
  - Manage industrial processes
  - Power plants, Refineries, etc.
- Night Dragon (2009)
  - attacks against several global oil, energy, and petrochemical companies.
  - Steal highly sensitive information
    - e.g. oil and gas field bids and operations
    - impacts multibillion dollar deals
  - Strategies standard
    - social engineering, spear-phishing, Windows exploits, Active Directory compromises, remote administration tools (RATs)
- Stuxnet

52



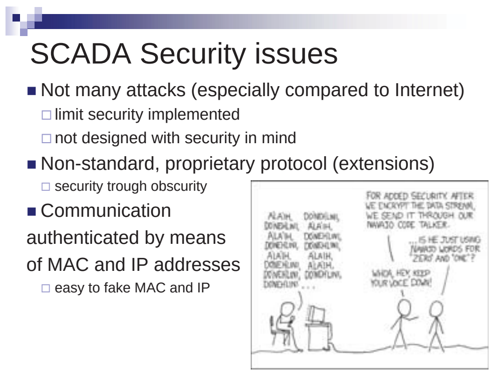
SCADA Essentials

- PLC: programmable logic controller
- Connected to Sensors and Actuators.
  - switches,
  - temperature and pressure sensors
  - operate electric motors,
  - pneumatic or hydraulic cylinders
  - …



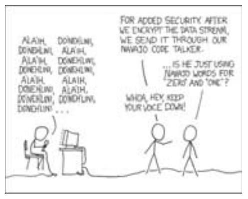
53

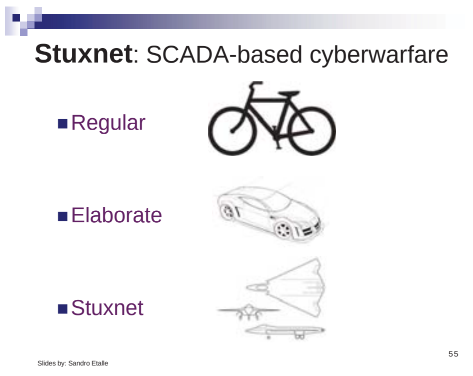
Not only the PCs or ICT networks are potential targets for attacks. Attacks on critical infrastructure can be especially harmful. So called SCADA (supervisory control and data acquisition) systems that manage industrial processes, power plants, locks, bridges, prison cell doors, etc. are vulnerable to attacks. "SCADA Vulnerabilities In Prisons Could Open Cell Doors" (slashdot Jan 2012), "Locks, pumping stations and bridges badly protected" (in Dutch, 14-2-12 een-vandaag), "Attack Code for SCADA Vulnerabilities Released Online" (3-12-11, wired.com).



SCADA Security issues

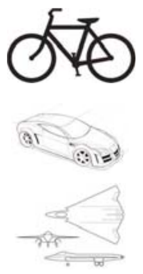
- Not many attacks (especially compared to Internet)
  - limit security implemented
  - not designed with security in mind
- Non-standard, proprietary protocol (extensions)
  - security trough obscurity
- Communication authenticated by means of MAC and IP addresses
  - easy to fake MAC and IP


ALA'IH, DONEHLINI, ALA'IH, DONEHLINI, ALA'IH, DONEHLINI, DONEHLINI, ALA'IH, DONEHLINI, DONEHLINI, ALA'IH, ALA'IH, DONEHLINI, DONEHLINI, DONEHLINI ...
FOR ADDED SECURITY, AFTER WE ENCRYPT THE DATA STREAM, WE SEND IT THROUGH OUR NAVAJO CODE TALKER.
... IS HE JUST USING NAVAJO WORDS FOR "ZERO" AND "ONE"?
WHOA, HEY, KEEP YOUR VOICE DOWN!



Stuxnet: SCADA-based cyberwarfare

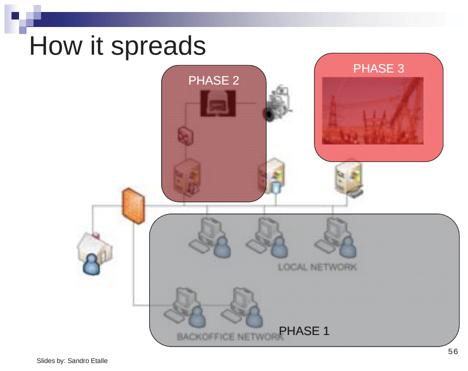
- Regular
- Elaborate
- Stuxnet



55

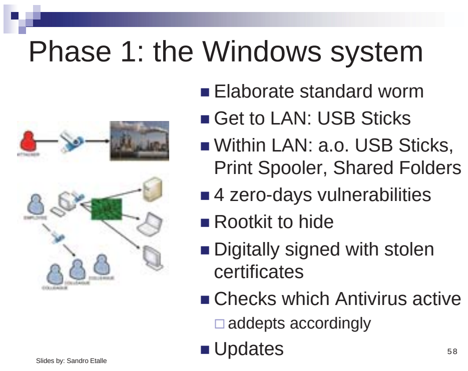
Slides by: Sandro Etalle

A major boost for the media attention for SCADA security was the discovery ‘in the wild’ of a very advanced targeted virus; Stuxnet. Stuxnet is a very sophisticated piece of malware, combining advanced techniques, some of which already introduced above. It uses several zero day vulnerabilities, is aware of virus scanners that might be able to detect it and adapts itself to avoid them, uses techniques such as rootkits to remain hidden and is able to update itself through a command and control center. A stolen certificate is used to pretend to be a valid driver, so it will be installed without prompting the user.


How it spreads

Slides by: Sandro Etalle

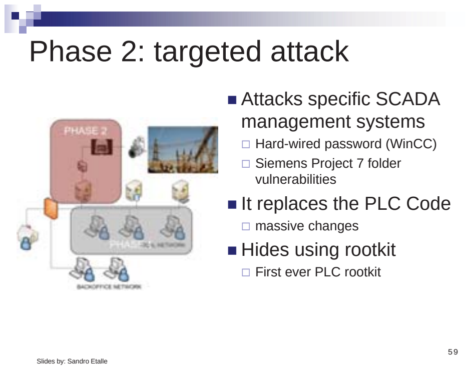
56


Phase 1: the Windows system

- Elaborate standard worm
- Get to LAN: USB Sticks
- Within LAN: a.o. USB Sticks, Print Spooler, Shared Folders
- 4 zero-days vulnerabilities
- Rootkit to hide
- Digitally signed with stolen certificates
- Checks which Antivirus active
  - addepts accordingly
- Updates

Slides by: Sandro Etalle

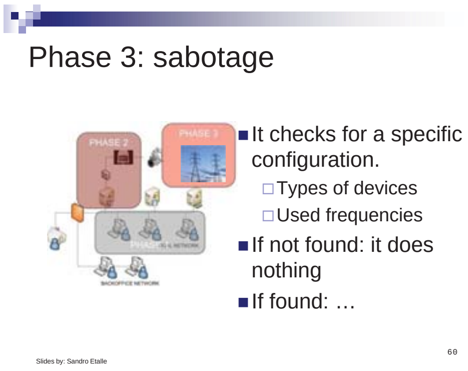
58


Phase 2: targeted attack

- Attacks specific SCADA management systems
  - Hard-wired password (WinCC)
  - Siemens Project 7 folder vulnerabilities
- It replaces the PLC Code
  - massive changes
- Hides using rootkit
  - First ever PLC rootkit

Slides by: Sandro Etalle

59

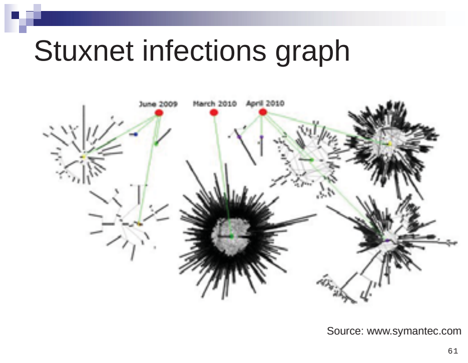

Phase 3: sabotage

- It checks for a specific configuration.
  - Types of devices
  - Used frequencies
- If not found: it does nothing
- If found: …

Slides by: Sandro Etalle

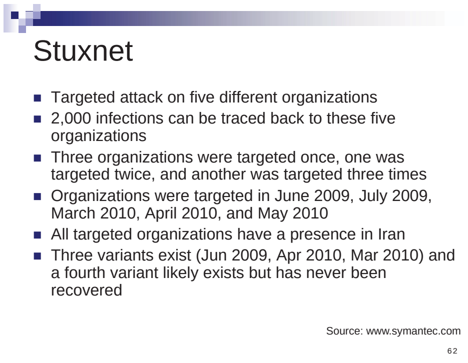
60

The virus goes through several stages; it tries to get on to a local network by using e.g. usb sticks and then spreads on this local network using vulnerabilities in attached devices and their software such as print spoolers. It then looks for PLCs to infect, making massive changes to the PLC code. One on the PLC it checks for a specific configuration considering e.g. types of devices and frequencies used, doing nothing if the configuration does not match. When it does find the configuration it sabotages the processes controlled by the PLC.


Stuxnet infections graph

Source: www.symantec.com

61

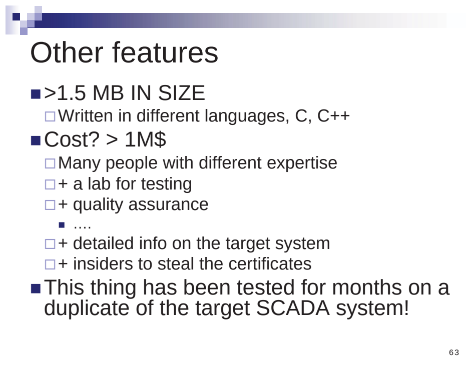

Stuxnet

- Targeted attack on five different organizations
- 2,000 infections can be traced back to these five organizations
- Three organizations were targeted once, one was targeted twice, and another was targeted three times
- Organizations were targeted in June 2009, July 2009, March 2010, April 2010, and May 2010
- All targeted organizations have a presence in Iran
- Three variants exist (Jun 2009, Apr 2010, Mar 2010) and a fourth variant likely exists but has never been recovered

Source: www.symantec.com

62

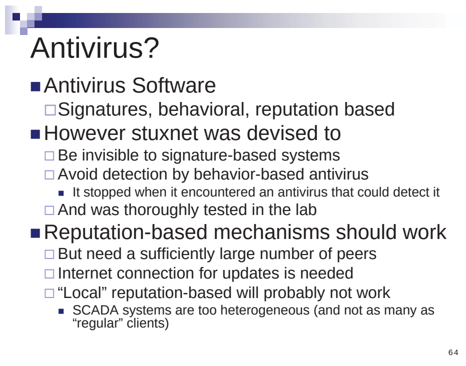
Note that several different expertises need to be combined to make a sophisticated piece of malware like stuxnet. Ofcourse there is the creation of an advanced worm for penetrating and spreading on the target organizations LAN. However, there is also the programming of PLC code malware that should not only infest and remain hidden but also disrupt the process being controlled in a way that is not easy to detect but will have serious effects. Developing and testing such code would very likely require a duplicate of the target SCADA system to be built.



Other features

- >1.5 MB IN SIZE
  - Written in different languages, C, C++
- Cost? > 1M$
  - Many people with different expertise
  - + a lab for testing
  - + quality assurance
    - ....
  - + detailed info on the target system
  - + insiders to steal the certificates
- This thing has been tested for months on a duplicate of the target SCADA system!

63



Antivirus?

- Antivirus Software
  - Signatures, behavioral, reputation based
- However stuxnet was devised to
  - Be invisible to signature-based systems
  - Avoid detection by behavior-based antivirus
    - It stopped when it encountered an antivirus that could detect it
  - And was thoroughly tested in the lab
- Reputation-based mechanisms should work
  - But need a sufficiently large number of peers
  - Internet connection for updates is needed
  - “Local” reputation-based will probably not work
    - SCADA systems are too heterogeneous (and not as many as “regular” clients)

64

Current anti-virus measures are not able to deal with sophisticated attacks such as stuxnet. Advanced persistent threats and digital warfare continue to grow. New protection methods will be needed to defend against such targeted digital attacks.

3.4 Conclusions

In this chapter we have looked at malware and key network threats along with some methods that address them. When considering specific (network) attacks, it is important to keep the right attacker model in mind (some attacks make no sense for some attacker models as they are either bring no gain to such an attacker or the attack is not possible for that attacker).

Given all the network threats it should be clear that secure communication requires carefully designed security protocols to govern the interaction. In a later lecture (see Chapter **??**) we will look into how to evaluate security protocols against given security goals.

If you wish to learn more about network security there are many books on the subject e.g. [10] there is also a Master course on this subject (taught in Twente) in the Kerckhoffs Masters (see `http://www.kerckhoffs-institute.org`) program.

3.4.1 Literature

Suggested reading (check the course page [2] for the most up to date list of suggested reading materials):

- Security Engineering Introduction [3, Ch 18]

3.5 Exercises

1. Consider again the online music store of the previous two chapters. Review your requirements analysis taking network and malware threats and countermeasures into account.

2. The WebGoat exercises are focused on web server vulnerabilities, see lab sessions 2 and further.