MALWARE – PART I

Prof. Dr. Bernhard Tellenbach

Content

Goals

- You know the most important «types» of malware like worms, Trojans, ransomware, rootkits, bootkits,…
- You know several techniques how malware communicates and how malware hides itself from the prying eyes of users and defenders
- You have a basic understanding of why our defences against malware are still quite weak

Malware - Overview

Definition

- Malware is an acronym for **mal**icious soft**ware**
- It comes in different formats; executables, shell code, scripts, firmware, …
- Two definitions:
  - NIST [1]: A program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or operating system or otherwise annoying or disrupting the victim.
  - OR-MEIR et al. [2]: Malware is code running on a computerized system whose presence or behavior the system administrators are unaware of; were the system administrators aware of the code and its behavior, they would not permit it to run. Malware compromises the CIA of the system by exploiting existing vulnerabilities in a system or by creating new ones.
- Most definitions include at least one of the following two main traits:
  - maliciousness or harmfulness
  - the ability to perform actions without the user's consent or knowledge

Most definitions are quite similar to the NIST one (see also below). The definition of OR-MEIR et al. is different in that it defines malware from the system administrators' point of view. They say that previous definitions of malware largely focus on the intentions of malware authors and argue that it is impossible to determine the intentions behind an unknown piece of malware. For example, definitions like "a program inserted into the system with the intent of …" might be incorrect since the program might be dual-use and at the time it was inserted, it was done for a legitimate reason. The definition by OR-MEIR et al captures this case with the "Malware is code running on ….. whose … [malicious] behavior the system administrators are unaware of".

The second part of their definition refers to three principals of security: ***confidentiality***, ***integrity*** , and ***availability*** (known as the *CIA triad*). When a user accesses a computerized service or system, he/she assumes that it is safe in those three respects. In other words, the system is believed to be available at all times, the system and the data it contains is correct and complete, and the private information stored on or used by the system/service is not exposed to unprivileged parties. They argue that any malicious behavior compromises the trust between the user and the attacked system in one (or more) of these respects.

Five slightly different definitions of malware:

- **NIST [1]** - A program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or operating system or otherwise annoying or disrupting the victim.
- **TechTarget [3]** - Any program or file that is harmful to a computer user. Malicious programs can perform a variety of functions, including stealing, encrypting or deleting sensitive data, altering or hijacking core computing functions and monitoring users' computer activity without their permission.
- **BullGuard [4]** - A computer program designed to infiltrate and damage computers without the user's consent.
- **Kaspersky [5]** - A type of computer program designed to infect a legitimate user's computer and inflict harm on it in multiple ways. Malware can infect computers and devices in several ways and comes in a number of forms, just a few of which include viruses, worms, Trojans,

spyware and more.

- **Norton [6]** - Malware is software that is specifically designed to gain access or damage a computer without the knowledge of the owner.

[1] https://csrc.nist.gov/Glossary/?term=5373

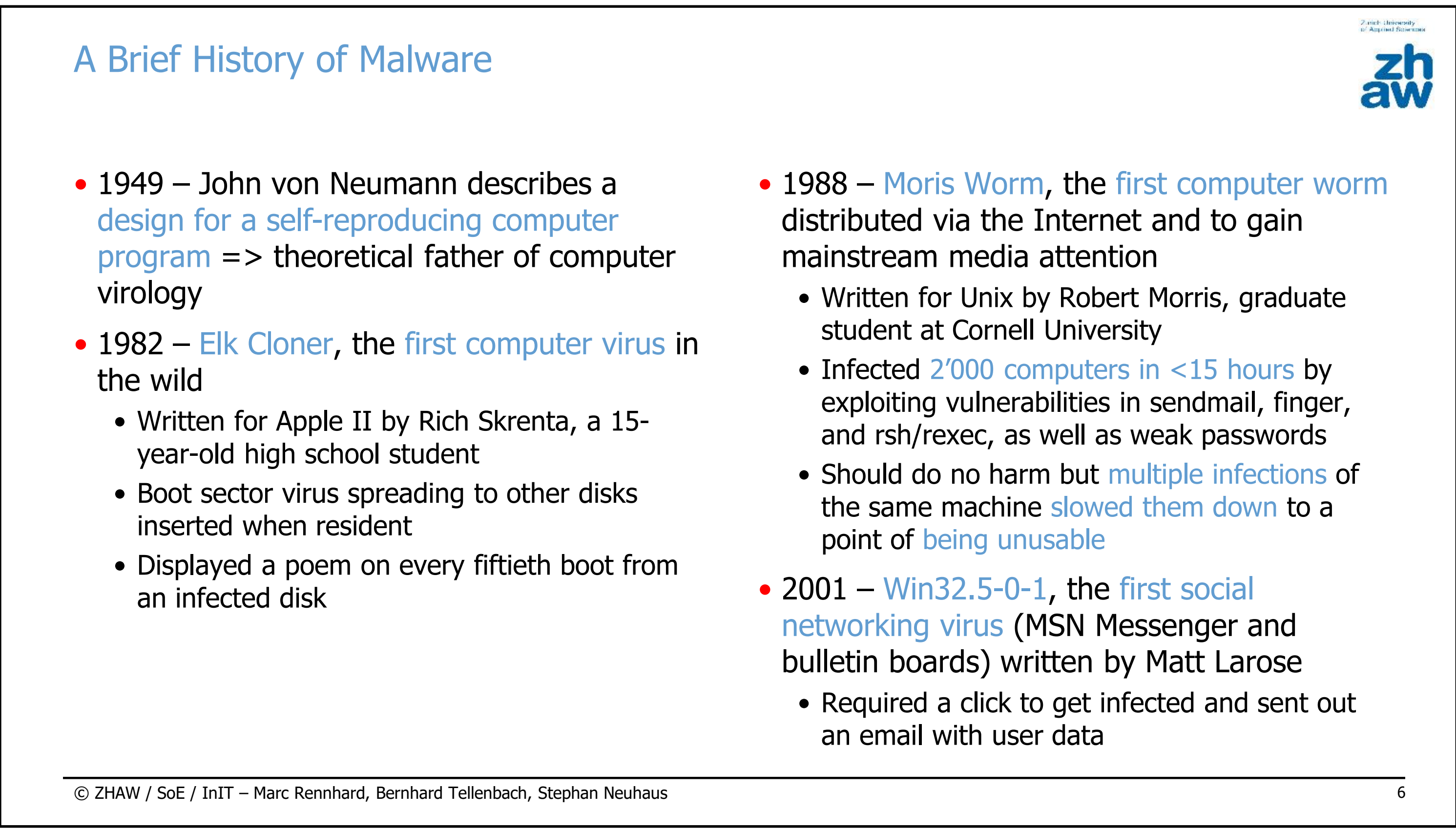
[2] https://dl.acm.org/doi/fullHtml/10.1145/3329786

[3] https://searchsecurity.techtarget.com/definition/malware.

[4] https://www.bullguard.com/bullguard-security-center/pc-security/computer-threats/malware-definition,-history-and-classification.aspx.

[5] https://www.kaspersky.com/resource-center/preemptive-safety/what-is-malware-and-how-to-protect-against-it.

[6] https://us.norton.com/internetsecurity-malware.html.



A Brief History of Malware

- 1949 – John von Neumann describes a design for a self-reproducing computer program => theoretical father of computer virology
- 1982 – Elk Cloner, the first computer virus in the wild
  - Written for Apple II by Rich Skrenta, a 15-year-old high school student
  - Boot sector virus spreading to other disks inserted when resident
  - Displayed a poem on every fiftieth boot from an infected disk
- 1988 – Moris Worm, the first computer worm distributed via the Internet and to gain mainstream media attention
  - Written for Unix by Robert Morris, graduate student at Cornell University
  - Infected 2'000 computers in <15 hours by exploiting vulnerabilities in sendmail, finger, and rsh/rexec, as well as weak passwords
  - Should do no harm but multiple infections of the same machine slowed them down to a point of being unusable
- 2001 – Win32.5-0-1, the first social networking virus (MSN Messenger and bulletin boards) written by Matt Larose
  - Required a click to get infected and sent out an email with user data

It is ***difficult to pinpoint the first malware***. According to [1], the ***Creeper virus*** written by Bob Thomas at BBN Technologies in 1971 was probably the first virus since it had the ***capability to replicate*** itself. Creeper gained access to other systems via the ARPANET and copied itself to the remote system where the message, "I'm the creeper, catch me if you can!" was displayed. The ***Reaper*** program was created to delete Creeper. However, the ***first academic work on self-replicating computer programs*** was done in 1949 by John von Neumann who gave lectures at the University of Illinois about the "Theory and Organization of Complicated Automata". In an essay he described how a computer program could be designed to reproduce itself. Von Neumann's design for a self-reproducing computer program is considered the world's first computer virus, and according to [2] he is considered to be the theoretical father of computer virology.

[1] Thomas Chen, Jean-Marc Robert, The Evolution of Viruses and Worms, Chapter from 'Statistical Methods in Computer Security‘, 2004, ISBN 0-8247-5939-7

[2] Éric Filiol, Computer viruses: from theory to applications, Volume 1, Birkhäuser, 2005, pp. 19–38 ISBN 2-287-23939-1

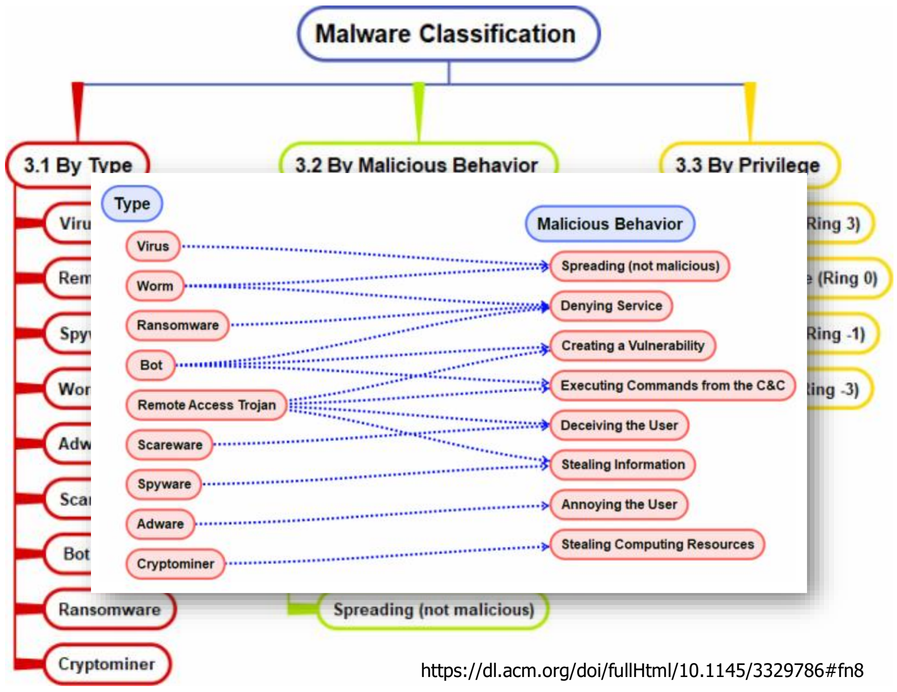
Elk Cloner – An article on the first virus that left the computer or lab where it was created. The article includes statements from the author of Elk Cloner: http://www.theregister.co.uk/2012/12/14/first_virus_elk_cloner_creator_interviewed/

Morris Worm: Robert Tappan Morris later became ***professor at the MIT***!

Malware Classification

- Malware classification helps us in multiple ways, for example to have a common language and understanding of a malware and for risk scoring
- There are multiple types of classification with different end goals
  - By type: Based on the main features implemented by the malware (e.g., replication method)
    - Terms frequently used in mass media, for example bot, virus or worm
  - By behavior: Based on the exhibited behavior
    - For example, "information stealer"
  - By family/lineage: Focus on the authorship and the lineage / evolution
    - For example, by analyzing the code-case for shared or similar code
- The problem with malware classification: there is no generic commonly agreed classification scheme
- For the types, there is some consensus on typical characteristics of a malware of that type, but details might vary

Malware Classification



https://dl.acm.org/doi/fullHtml/10.1145/3329786#fn8

Malware classification illustrated by Or-Meir et al.

By Type – The above classification by malware type can be found in most of the literature; this terminology is also widely known and used by the general public. The terms listed are frequently used in the mass media and among users to describe different types of malware.

By Malicious Behavior - When describing malware to non-professionals, the categories and terms mentioned above are acceptable. However, for analysis purposes, it is more important to focus on malware behavior instead of malware type.

Source: Ori Or-Meir, Nir Nissim, Yuval Elovici, and Lior Rokach. 2019. Dynamic Malware Analysis in the Modern Era—A State of the Art Survey. ACM Comput. Surv. 52, 5, Article 88 (September 2019), 48 pages. DOI:https://doi.org/10.1145/3329786

Malware Types & Terms

- There is no such thing like mutually exclusive malware types but rather categories of malware that are not mutually exclusive
- Aside from the types below, you should also know what these terms mean: Living Off the Land and Fileless Malware

• Trojan Horse	• Bot/Botnet	• Cryptominer
• Backdoor	• Monitor / Spyware	• Spyware
• Remote Access Tool	• Mailer Scareware	• Rootkit / Bootkit
• Downloader	• Adware	• Worm
• Dropper	• Ransomware	• Virus

Trojan Horse

- Trojan or Trojan Horse is frequently used to classify malware samples that employ some mechanism to conceal their true identity.
- Some examples:
  - A whitepaper you downloaded concealing a PDF exploit
  - A binary version of a 3rd party library containing a backdoor
  - Malware embedded within a bundle that pretends to be a printer driver
  - A music player that contains embedded backdoor code when you run it

Backdoor & Remote Access Tools (RAT)

- Traditionally, backdoor functionality provides some level of interactive 1-to-1 access to a compromised system
  - Usually provided through a network connection between adversary and the target's host.
- A Remote Access Tool is frequently used to describe a backdoor having a high level of interactive functionality and providing expansive access to the target's host.
  - Sometimes, the level of access provided is more powerful than the normal end-user interface of the targeted system.

Downloader

- During a mission, attackers frequently employ multiple tools
- Downloader functionality provides a tool with a specific feature to download other tools
- In the initial phase (e.g., spear- phishing), adversaries often deliver a lightweight tool
- It may be pre-configured to pull down more feature-full malware
  - Attributes of the compromised host may guide from where and what to retrieve next
- The breaking-up of attacks helps to reduce detectability and facilitates surgical retooling.

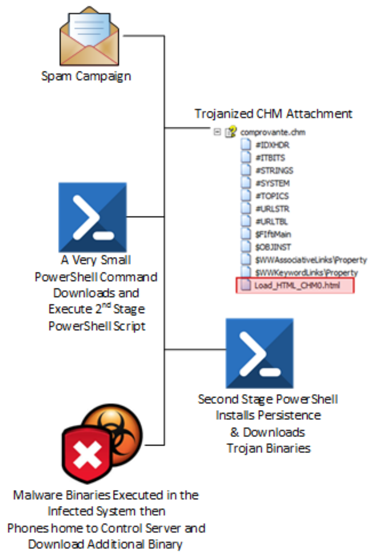

Spam Campaign
Trojanized CHM Attachment
comprovante.chm
#IDXHDR
#ITBITS
#STRINGS
#SYSTEM
#TOPICS
#URLSTR
#URLTBL
$FIftiMain
$OBJINST
$WWAssociativeLinks\Property
$WWKeywordLinks\Property
Load_HTML_CHM0.html
A Very Small PowerShell Command Downloads and Execute 2nd Stage PowerShell Script
Second Stage PowerShell Installs Persistence & Downloads Trojan Binaries
Malware Binaries Executed in the Infected System then Phones home to Control Server and Download Additional Binary

Image source: *https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/chm-badness-delivers-a-banking-trojan/*

CHM file type: Like good old Microsoft Office Macros, Compiled HTML (CHM) Help files have been utilized by malware authors for more than a decade to sneak malicious downloader code into files making them harder to detect. CHMs are a Microsoft proprietary online help file that consist of a collection of HTML pages compiled into a single compressed file format. The most common use of CHMs are for offline software documentation and help guides.

Source: https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/chm-badness-delivers-a-banking-trojan/

Trojan.TrickBot is Malwarebytes' detection name for a banking Trojan targeting Windows machines.. Developed in 2016, TrickBot is one of the more recent banking Trojans, with many of its original features inspired by Dyreza (another banking Trojan).

Besides targeting a wide array of international banks via its webinjects, Trickbot can also steal from Bitcoin wallets. Some of its other capabilities include harvesting emails and credentials using the Mimikatz tool. Its authors also show an ability for constant new features and developments.

Trojan.TrickBot comes in modules accompanied by a configuration file. Each module has a specific task like gaining persistence, propagation, stealing credentials, encryption, and so on. The C&Cs are set up on hacked wireless routers.

In early August 2019, researchers noticed a high-volume of malicious spam campaigns delivering Trickbot. Almost all of these campaigns were found using Ostap, a commodity **JavaScript downloader**.

In the previous campaigns, Trickbot relied on **downloaders that used obfuscated Command Shell and later PowerShell** commands to download their payloads. These PowerShell commands were triggered by VBA AutoOpen macros.

The endpoint user will not notice any symptoms of a Trickbot infection. However, a network admin will likely see changes in traffic or attempts to reach out to blacklisted IPs and domains, as the malware will communicate with Trickbot’s command and control infrastructure to exfiltrate data and receive tasks. Trojan.TrickBot gains persistence by creating a Scheduled Task.

TrickBot typically spreads via malicious spam campaigns. It can also spread laterally using the EternalBlue exploit (MS17-010). Other methods of propagation include infected attachments and embedded URLs. Trojan.TrickBot is also seen as a secondary infection dropped by Trojan.Emotet. Due to the way Trickbot uses the EternalBlue vulnerability to spread through a company’s network, any infected machine on the network will re-infect machines that have been previously cleaned when they rejoin the network. Therefore, IT teams need to isolate, patch, and remediate each infected system one-by-one. This can be a long and painstaking process.

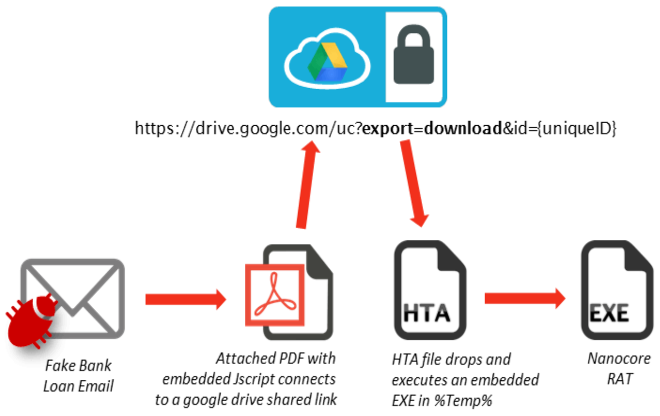
Sources:

- https://blog.malwarebytes.com/detections/trojan-trickbot/

- https://cyware.com/news/demystifying-ostap-a-new-downloader-for-trickbot-trojan-fdbbaeaa

Dropper

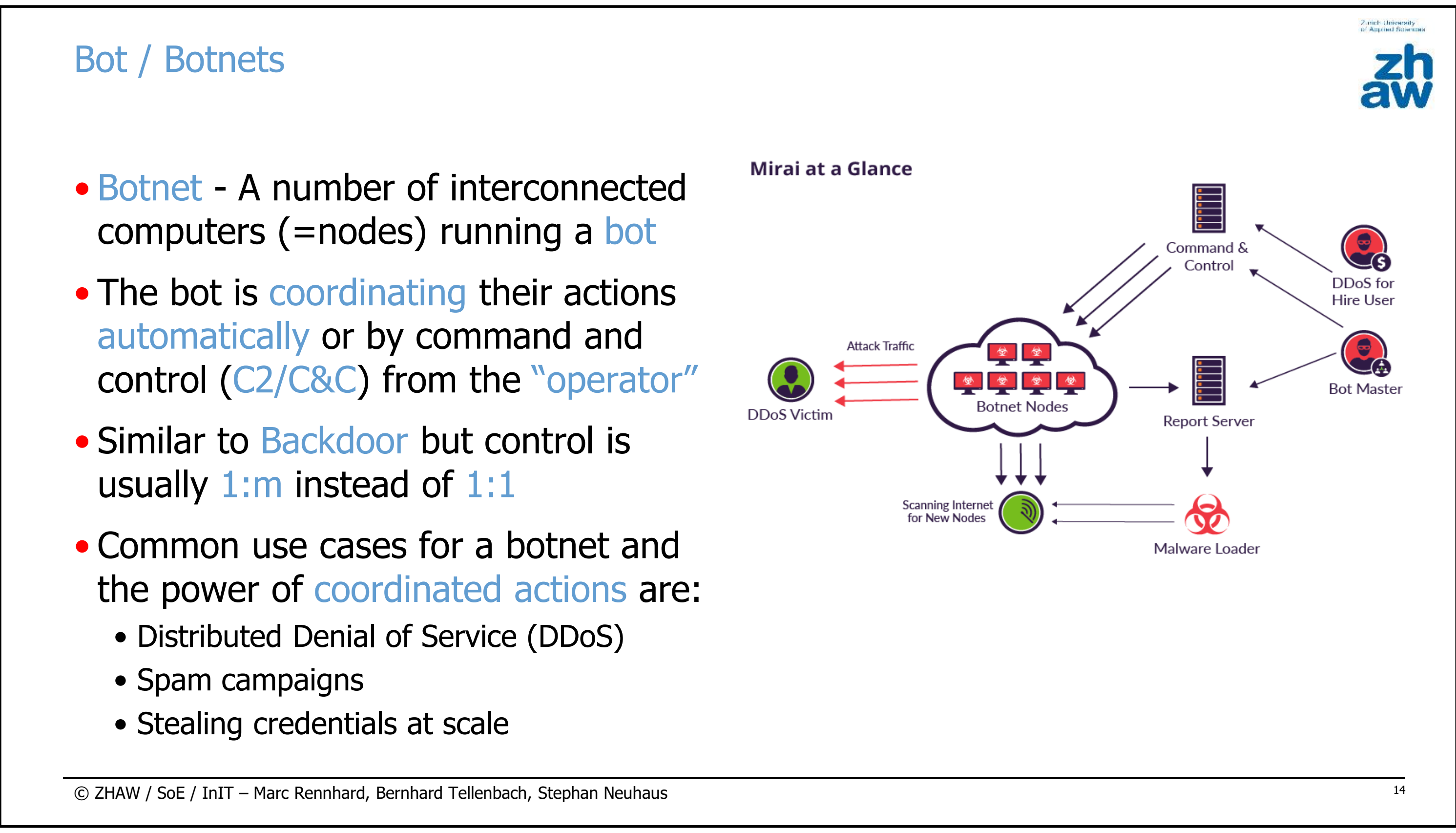
- In a multi-stage attack, one tool may need to write/drop another one to disk and execute it.
- Code must first be extracted and stored in an OS-compatible, native container (e.g., EXE file).
- Example:
  - HTA document that contains embedded, encoded malware
  - When opened, the embedded malware is extracted, dropped to disk and executed
- **Note:** Downloaders are usually also Droppers, but not vice versa


https://drive.google.com/uc?**export=download**&id={uniqueID}
HTA
EXE
Fake Bank Loan Email
Attached PDF with embedded Jscript connects to a google drive shared link
HTA file drops and executes an embedded EXE in %Temp%
Nanocore RAT

HTA is a file extension for an **HTML executable file format**. HTA files are used with Internet Explorer 5 and up. HTA files use HTML syntax to create applications requiring only an additional header and the HTA extension to differentiate.

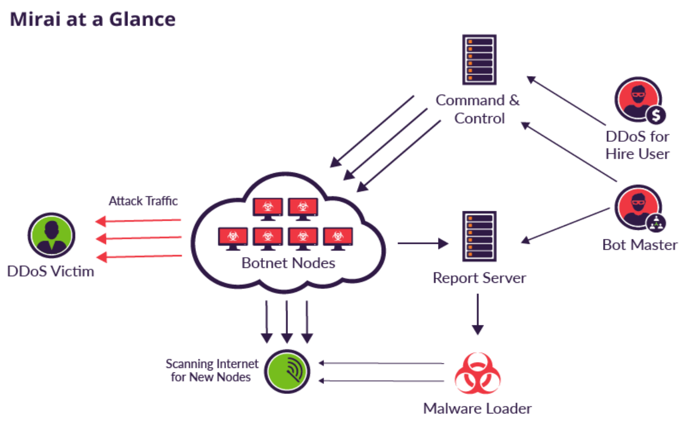
Malware developers use a variety of distribution methods in order to confuse users and evade certain AV solutions. In 2017, FortiGuard Labs found a phishing campaign targeting French Nationals. In this campaign, a PDF file with an embedded javascript is used to download the payload from a Google Drive shared link. As it turns out, the downloaded file is an HTA (HTML Application) file, a format that is becoming more and more common as a malware launch point. It is usually used as a downloader for the actual binary payload. However, in this campaign, the binary payload, which was later found to be a NanoCore RAT client, is actually embedded in the obfuscated HTA. This way, the **HTA effectively serves as a wrapper** to try and slip passed traditional file type-based scanning in the network as well as anti-spam services.

Source: https://www.fortinet.com/blog/threat-research/pdf-phishing-leads-to-nanocore-rat-targets-french-nationals.html



Bot / Botnets

- Botnet - A number of interconnected computers (=nodes) running a bot
- The bot is coordinating their actions automatically or by command and control (C2/C&C) from the "operator"
- Similar to Backdoor but control is usually 1:m instead of 1:1
- Common use cases for a botnet and the power of coordinated actions are:
  - Distributed Denial of Service (DDoS)
  - Spam campaigns
  - Stealing credentials at scale



History: Botnets date back to at least **1999**: Pretty Park worm and Sub7 Trojan demonstrated that controlling victim machines by **listing to Internet Relay Chats (IRC)** to receive **malicious commands** works

Example: Mirai Botnet

The Mirai internet of things (IoT) botnet is infamous for having targeted connected household consumer products. It attached itself to cameras, alarm systems and personal routers, and spread quickly. The damage could be quite substantial. People might not have realized that their internet-enabled webcam was actually responsible for attacking Netflix.

A DDoS botnet attack is pretty straightforward. It gives commands to the control server. And the control server issues attack commands to each of the individual nodes (infected devices) in the botnet. They in turn send the attack traffic to the target. Not all DDoS attacks come from botnets, but here's why botnets are effective.

- **Obfuscation** – The attacker is able to conceal themselves from the victim.
- **Amplification** – By using compromised systems, the attacker can launch a larger attack.
- **Geographical Dispersion** – A large botnet can span the globe making for a massively distributed attack that is hard to mitigate.

There's nothing new about botnets. They've been with us for a long time. In fact, some very large ones existed in the early 2000s that involved millions of nodes.

But there's been a major shift over time in the motivation of the people behind the DDoS attacks. Instead of simply trafficking in spam, botnet operators have figured out a way to monetize their efforts through extortion or by launching a DDoS-for-hire platform like Mirai. When people talk about Mirai they often talk about the emerging threat caused by household IoT devices. This doesn't account for 100 percent of Mirai activity, but certainly there are some aspects that make these personal devices attractive to attackers.

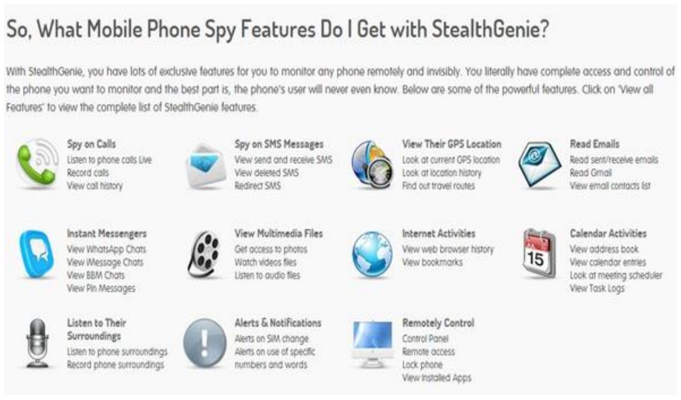
- Large Numbers of Devices – Most people have a single computer, but they probably also have multiple internet-enabled appliances.

- Vendor Security Is Weak – IoT appliances have historically not been very secure, causing a proliferation of insecure devices.
- Consumers Neglect Security – Consumers are less likely to secure their internet toaster than their personal computer.
- Emerging Market Means Opportunity – With the proliferation of the IoT, the pool of possible botnet nodes is growing. The numbers are in the attacker's favor.
- Homogenous Platforms – Unlike personal computers, IoT platforms are generally identical.

Source and more details: https://www.imperva.com/blog/how-to-identify-a-mirai-style-ddos-attack/

Spyware (Monitor)

- Malware with mechanism to record user activity or the environment employs monitor functionality
- The recorded data is delivered back to the adversary
- Example monitor sources on a target:
  - Webcam / Microphone
  - Desktop recording
  - Password prompts
  - Common data folders (My Documents, …)
  - Keyboard (keylogging)
  - Web browser traffic
  - Network traffic



So, What Mobile Phone Spy Features Do I Get with StealthGenie?

With StealthGenie, you have lots of exclusive features for you to monitor any phone remotely and invisibly. You literally have complete access and control of the phone you want to monitor and the best part is, the phone's user will never even know. Below are some of the powerful features. Click on 'View all Features' to view the complete list of StealthGenie features.

Spy on Calls
Listen to phone calls Live
Record calls
View call history

Spy on SMS Messages
View send and receive SMS
View deleted SMS
Redirect SMS

View Their GPS Location
Look at current GPS location
Look at location history
Find out travel routes

Read Emails
Read sent/receive emails
Read Gmail
View email contacts list

Instant Messengers
View WhatsApp Chats
View iMessage Chats
View BBM Chats
View Pin Messages

View Multimedia Files
Get access to photos
Watch videos files
Listen to audio files

Internet Activities
View web browser history
View bookmarks

Calendar Activities
View address book
View calendar entries
Look at meeting scheduler
View Task Logs

Listen to Their Surroundings
Listen to phone surroundings
Record phone surroundings

Alerts & Notifications
Alerts on SIM change
Alerts on use of specific numbers and words

Remotely Control
Control Panel
Remote access
Lock phone
View Installed Apps

StealthGenie – the app that helps jealous partners and stalkers spy on you and your online conversations

There's a shady industry out there of businesses that sell spyware apps that market themselves to jealous partners, domestic abusers and stalkers, keen to spy upon others. Some market themselves as a way of easily keeping taps on your children, but there's no doubt that many are used to abuse individual's privacy and potentially put innocent people in danger.

One example of such spyware is StealthGenie, sold online by a Pakistani company called InvoCode.

According to a Department of Justice statement, 31-year-old Akbar and his co-conspirators are alleged to have created the spyware, capable of intercepting communications between mobile phones, including iPhones, Androids, and Blackberrys. The StealthGenie spyware was advertised as being "untraceable" and would probably be undetected by the vast majority of victims.

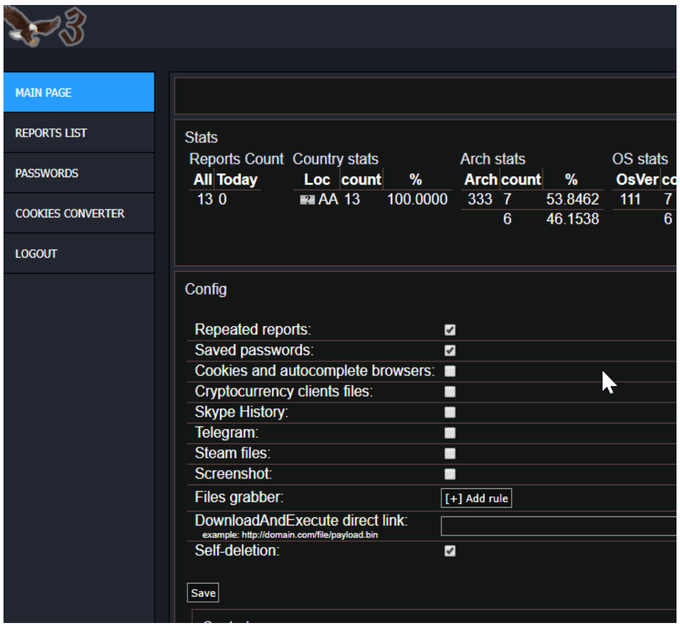
Not only could StealthGenie record incoming and outgoing voice calls, it could also be activated remotely allowing an unauthorised party to record conversations taking place nearby. In addition, snoops who purchased the software were given the ability to track the user's location, rifle through their target's incoming and outgoing emails and text messages, listen to voicemails and access other data on the devices.

Advertising and selling spyware technology is a criminal offence, but more than that – it helps others break the law too by assisting them in illegally tracking and surveilling individuals without their knowledge.

Source and more details: https://hotforsecurity.bitdefender.com/blog/stealthgenie-the-app-that-helps-jealous-partners-and-stalkers-spy-on-you-and-your-online-conversations-10331.html

Information Stealer (Spyware)

- Malware with functions to automate the process of stealing information
- It searches data using a hard-coded or network-provided collection plan
- Common approaches include:
  - Contact list theft
  - Browser cookies/history
  - Targeted document theft
  - Credential- / Keystores
- Similar to Monitor but with automated search & extraction of relevant data only



Stats

Reports Count		Country stats			Arch stats			OS stats	
All	**Today**	**Loc**	**count**	**%**	**Arch**	**count**	**%**	**OsVer**	**co**
13	0	AA	13	100.0000	333	7	53.8462	111	7
						6	46.1538		6

Config

Repeated reports:	✓
Saved passwords:	✓
Cookies and autocomplete browsers:	☐
Cryptocurrency clients files:	☐
Skype History:	☐
Telegram:	☐
Steam files:	☐
Screenshot:	☐
Files grabber:	[+] Add rule
DownloadAndExecute direct link: example: http://domain.com/file/payload.bin	
Self-deletion:	✓

Save

Example: AZORult

The AZORULT malware was first discovered in 2016 to be an information stealer that steals browsing history, cookies, ID/passwords, cryptocurrency information and more. It can also act as a downloader of other malware. It was sold on Russian underground forums to collect various types of sensitive information from an infected computer. A variant of this malware was able to create a new, hidden administrator account on the machine to set a registry key to establish a Remote Desktop Protocol (RDP) connection.

Exploit kits such as Fallout Exploit Kit (EK) and phishing mails with social engineering technique are now the major infection vectors of the AZORult malware. Other malware families such as Ramnit and Emotet also download AZORult. The current malspam and phishing emails use fake product order requests, invoice documents and payment information requests. This Trojan-Spyware connects to command and control (C&C) servers of attacker to send and receive information.

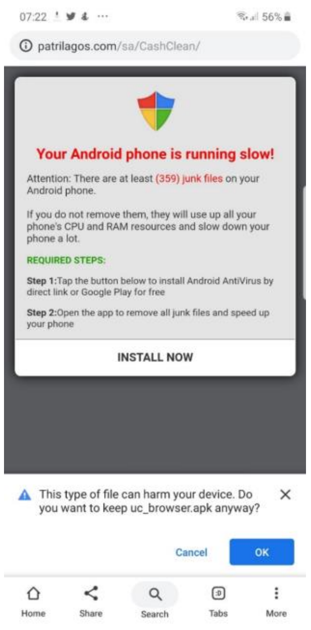
Behaviors

- Steals computer data, such as installed programs, machine globally unique identifier (GUID), system architecture, system language, user name, computer name, and operating system (OS) version
- Steals stored account information used in different installed File Transfer Protocol (FTP) clients or file manager software
- Steals stored email credentials of different mail clients
- Steals user names, passwords, and hostnames from different browsers
- Steals bitcoin wallets - Monero and uCoin
- Steals Steam and telegram credentials
- Steals Skype chat history and messages
- Executes backdoor commands from a remote malicious user to collect host Internet protocol (IP) information, download/execute/delete file

Source: https://success.trendmicro.com/solution/000146108-AZORULT-Malware-Information

Scareware / Adware

- Employed to achieve a social engineering outcome
- Entices or frequently scares the user into taking specific actions
- A common variant is “Fake Anti Virus” programs
  - When visiting a website, visitors are told they have a virus
  - The site offers a (fake) free Anti-Virus program to remove it
  - It then annoys the user (e.g., with popups) until a fee is paid
- Adware is another common variant where the software may not be an illegal enterprise, but is annoying nonetheless


07:22 56%

patrilagos.com/sa/CashClean/

Your Android phone is running slow!

Attention: There are at least (359) junk files on your Android phone.

If you do not remove them, they will use up all your phone's CPU and RAM resources and slow down your phone a lot.

REQUIRED STEPS:

Step 1: Tap the button below to install Android AntiVirus by direct link or Google Play for free

Step 2: Open the app to remove all junk files and speed up your phone

This type of file can harm your device. Do you want to keep uc_browser.apk anyway?

Example: Android Scareware

According to ESET, the new most common threat to mobile devices running Android in 2019 was so-called scareware, which is to scare users and lure money out of it.

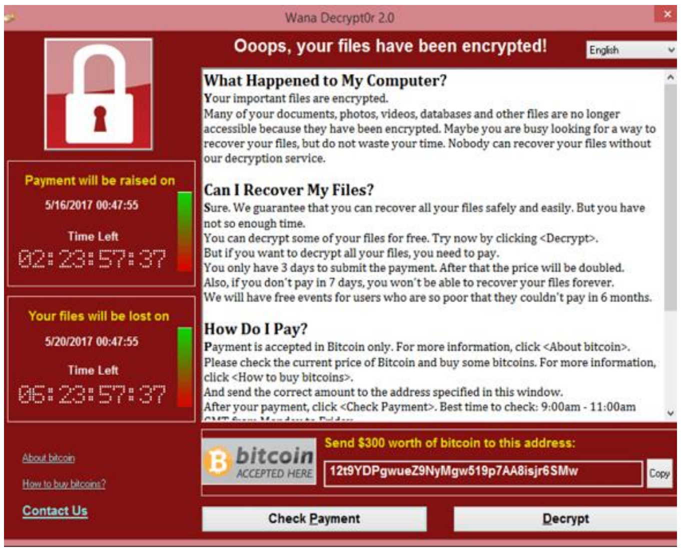
“In December, it reached a 3% share in detection. Advertising malware, which was the most common threat for most of last year, is weakening and spyware is gradually taking its place. This stems from ESET's regular statistics.

The most common threat faced by Android users in December is the so-called FakeAV scareware. It has been on the top rungs of detection since August. It is a fraudulent antivirus application. Once installed and run by the user, the application warns against suspected malware on the phone and offers the removal of the infection for a fee. However, this is a fraudulent challenge. The equipment is fine.”

https://engnews24h.com/eset-warns-against-fake-android-antivirus-attracting-money-from-users/

Ransomware

- Similar end-goals as scareware - social-engineer the target into paying a fee or some other activity
- Typically, ransomware encrypts files on the local system and asks for money to decrypt them
  - Contemporary ransomware leverages strong public-key cryptography
  - The malware only contains the public key => makes it practically impossible to recover the files without paying


Wana Decrypt0r 2.0

Ooops, your files have been encrypted!

What Happened to My Computer?
Your important files are encrypted.
Many of your documents, photos, videos, databases and other files are no longer accessible because they have been encrypted. Maybe you are busy looking for a way to recover your files, but do not waste your time. Nobody can recover your files without our decryption service.

Can I Recover My Files?
Sure. We guarantee that you can recover all your files safely and easily. But you have not so enough time.
You can decrypt some of your files for free. Try now by clicking <Decrypt>.
But if you want to decrypt all your files, you need to pay.
You only have 3 days to submit the payment. After that the price will be doubled.
Also, if you don't pay in 7 days, you won't be able to recover your files forever.
We will have free events for users who are so poor that they couldn't pay in 6 months.

How Do I Pay?
Payment is accepted in Bitcoin only. For more information, click <About bitcoin>.
Please check the current price of Bitcoin and buy some bitcoins. For more information, click <How to buy bitcoins>.
And send the correct amount to the address specified in this window.
After your payment, click <Check Payment>. Best time to check: 9:00am - 11:00am

Payment will be raised on
5/16/2017 00:47:55
Time Left
02:23:57:37

Your files will be lost on
5/20/2017 00:47:55
Time Left
06:23:57:37

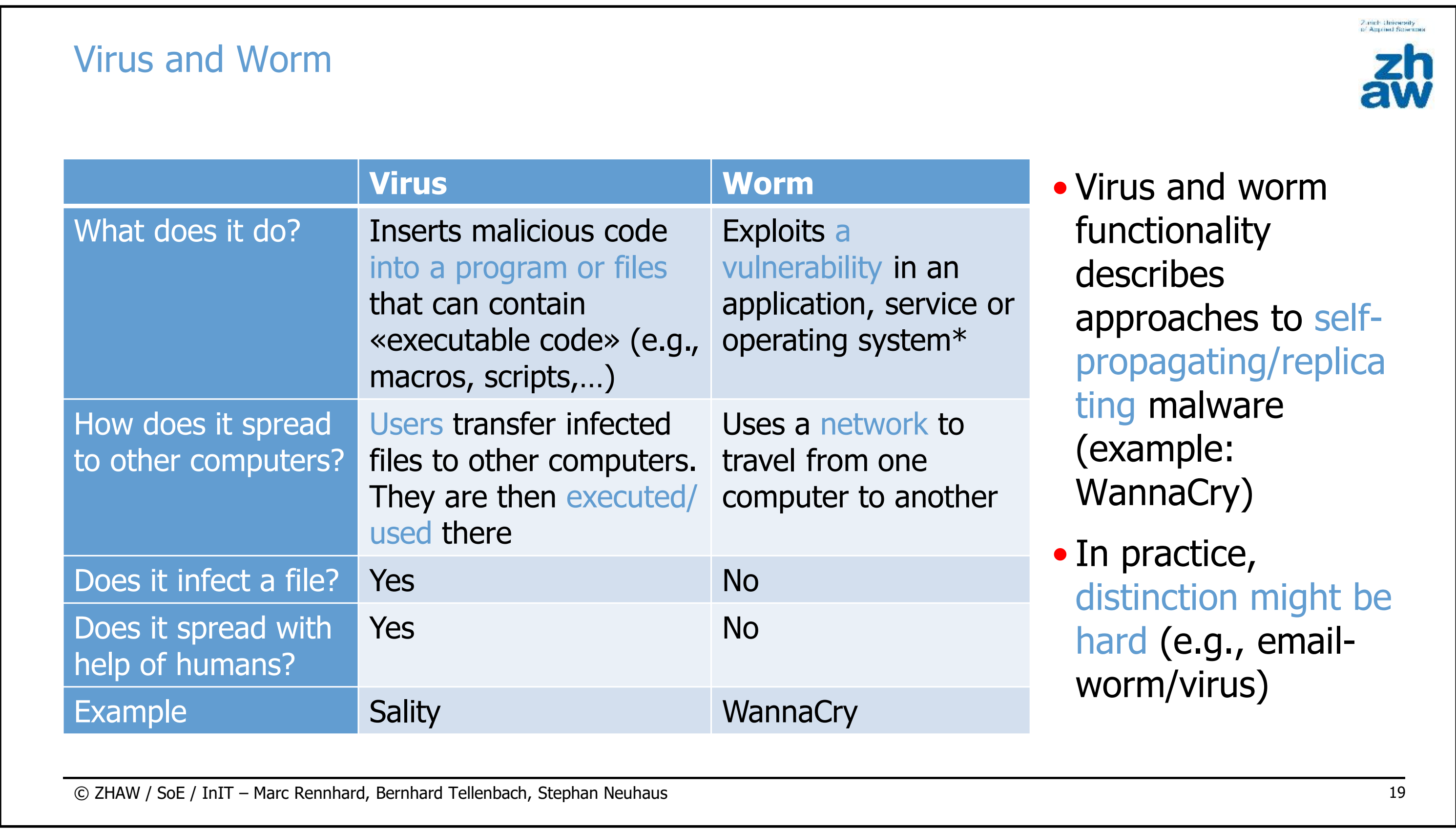
Send $300 worth of bitcoin to this address:
12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw

Example: WannaCry is a ransomware cryptoworm, which targeted computers running the Microsoft Windows operating system by encrypting data and demanding ransom payments in the Bitcoin cryptocurrency. The worm is also known as WannaCrypt, Wana Decrypt0r 2.0, WanaCrypt0r 2.0, and Wanna Decryptor. It is considered a network worm because it also includes a "transport" mechanism to automatically spread itself. This transport code scans for vulnerable systems, then uses the EternalBlue exploit to gain access, and the DoublePulsar tool to install and execute a copy of itself. WannaCry versions 0 to 2 were created using Microsoft Visual C++ 6.0.

EternalBlue is an exploit of Windows' Server Message Block (SMB) protocol released by The Shadow Brokers. Much of the attention and comment around the event was occasioned by the fact that the U.S. National Security Agency (NSA) (from whom the exploit was likely stolen) had already discovered the vulnerability, but used it to create an exploit for its own offensive work, rather than report it to Microsoft. Microsoft eventually discovered the vulnerability, and on Tuesday, 14 March 2017, they issued security bulletin MS17-010, which detailed the flaw and announced that patches had been released for all Windows versions that were currently supported at that time, these being Windows Vista/7/8.1/10/, Windows Server 2008/2012/2016.

DoublePulsar is a backdoor tool, also released by The Shadow Brokers on 14 April 2017. Starting from 21 April 2017, security researchers reported that there were tens of thousands of computers with the DoublePulsar backdoor installed. By 25 April, reports estimated that the number of infected computers could be up to several hundred thousand, with numbers increasing every day. The WannaCry code can take advantage of any existing DoublePulsar infection or installs it itself. On 9 May 2017, private cybersecurity company RiskSense released code on the website github.com with the stated purpose of allowing legal “white hat” penetration testers to test the CVE-2017-0144 exploit on unpatched systems.

When executed, the WannaCry malware first checks the "kill switch" domain name; if it is not found, then the ransomware encrypts the computer's data, then attempts to exploit the SMB vulnerability to spread out to random computers on the Internet, and "laterally" to computers on the same network. As with other modern ransomware, the payload displays a message informing the user that files have been encrypted and demands a payment of around US$300 in bitcoin within three days, or US$600 within seven days. Three hardcoded bitcoin addresses, or "wallets", are used to receive the payments of victims. As with all such wallets, their transactions and balances are publicly accessible even though the cryptocurrency wallet owners remain unknown.



Virus and Worm

	Virus	Worm
What does it do?	Inserts malicious code into a program or files that can contain «executable code» (e.g., macros, scripts,…)	Exploits a vulnerability in an application, service or operating system*
How does it spread to other computers?	Users transfer infected files to other computers. They are then executed/ used there	Uses a network to travel from one computer to another
Does it infect a file?	Yes	No
Does it spread with help of humans?	Yes	No
Example	Sality	WannaCry

- Virus and worm functionality describes approaches to self-propagating/replicating malware (example: WannaCry)
- In practice, distinction might be hard (e.g., email-worm/virus)

Computer viruses are named after human viruses that spread from person to person. A computer virus is a program made of malicious code that can propagate itself from device to device. Like a cold that alters your well-being, when your computer is infected, it alters the way your computer operates, can destroy your files, or prevent it from working altogether.

A virus typically attaches itself to a program, file, or the boot sector of the hard drive. Once the virus attaches itself to that file or program (aka, the host), they're infected.

When the infected application or file runs in the computer, the virus activates and executes in the system. It continues to replicate and spread by attaching replicas of itself to other files and applications in the system.

A virus spreads when the infected file or program migrates through networks, file collaboration apps, email attachments, and USB drives. Once a user opens the infected file or program, the vicious cycle repeats itself all over again.

Typically, the host program continues to function after the viral infection, but some viruses overwrite entire programs with copies of themselves, which corrupts and destroys the host program altogether. Viruses can also attack data: they can disrupt access, corrupt, or destroy it.

Worms are a self-replicating type of malware that enter networks by exploiting vulnerabilities, moving quickly from one computer to another. Because of this, worms can propagate themselves and spread very quickly – not only locally, but have the potential to disrupt systems worldwide. Unlike a typical virus, worms don't attach to a file or program. Instead, they slither and enter computers through a vulnerability in the network, self-replicating and spreading before you're able to remove the worm. But by then, they'll already achieved their goal, for example consumed all the bandwidth of the network, interrupting and arresting large network and web servers.

WannaCry: In 2017, the WannaCry worm attack caused damage worth hundreds of millions to billions of dollars. Also known as WannaCry ransomware, this attack is a hybrid of ransomware and a worm – specifically cryptoworm.

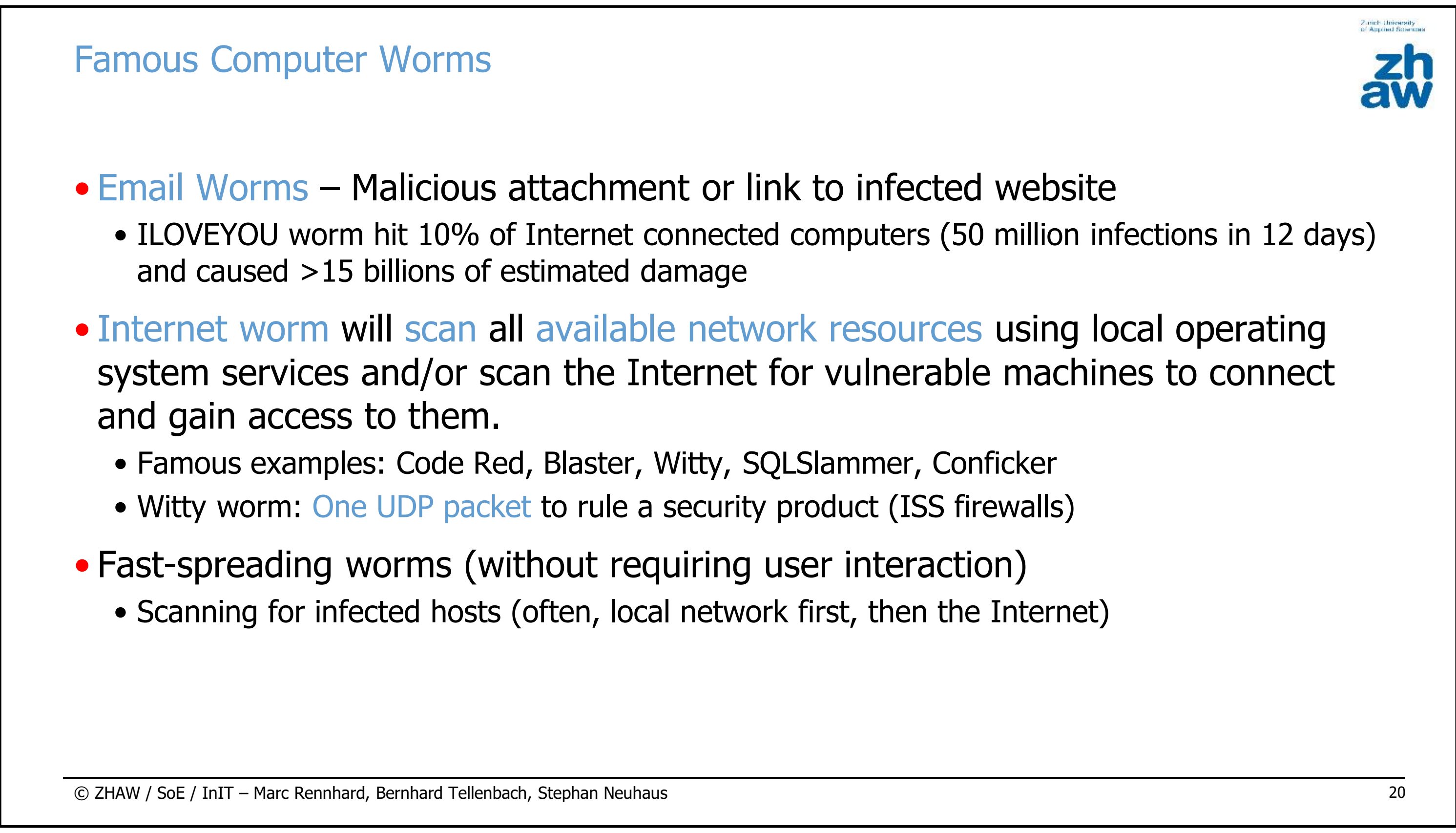
Instead, WannaCry took advantage of a vulnerability in Microsoft's SMB Version 1 file sharing protocol, typically used by Windows machines to communicate with file systems over a network. Those who didn't patch SMB Version 1 learned the hard way about the perils of forgetting to patch their systems. WannaCry leveraged EternalBlue, a Windows SMB protocol exploit, to gain access, install a backdoor, and download software – infecting the systems.

In short, WannaCry self-propagated, self-replicated, and quickly traversed entire networks, causing worldwide damage.

Source: https://www.varonis.com/blog/what-is-a-computer-virus-and-computer-worm/

Sality: Sality is the classification for a family of malicious software (malware), which infects files on Microsoft Windows systems. Sality was first discovered in 2003 and has advanced over the years to become a dynamic, enduring and full-featured form of malicious code. Systems infected with Sality may communicate over a peer-to-peer (P2P) network to form a botnet for the purpose of relaying spam, proxying of communications, exfiltrating sensitive data, compromising web servers and/or coordinating distributed computing tasks for the purpose of processing intensive tasks (e.g. password cracking). Since 2010, certain variants of Sality have also incorporated the use of rootkit functions as part of an ongoing evolution of the malware family. Because of its continued development and capabilities, Sality is considered to be one of the most complex and formidable forms of malware to date.

Source: https://en.wikipedia.org/wiki/Sality



Famous Computer Worms

- Email Worms – Malicious attachment or link to infected website
  - ILOVEYOU worm hit 10% of Internet connected computers (50 million infections in 12 days) and caused >15 billions of estimated damage
- Internet worm will scan all available network resources using local operating system services and/or scan the Internet for vulnerable machines to connect and gain access to them.
  - Famous examples: Code Red, Blaster, Witty, SQLSlammer, Conficker
  - Witty worm: One UDP packet to rule a security product (ISS firewalls)
- Fast-spreading worms (without requiring user interaction)
  - Scanning for infected hosts (often, local network first, then the Internet)

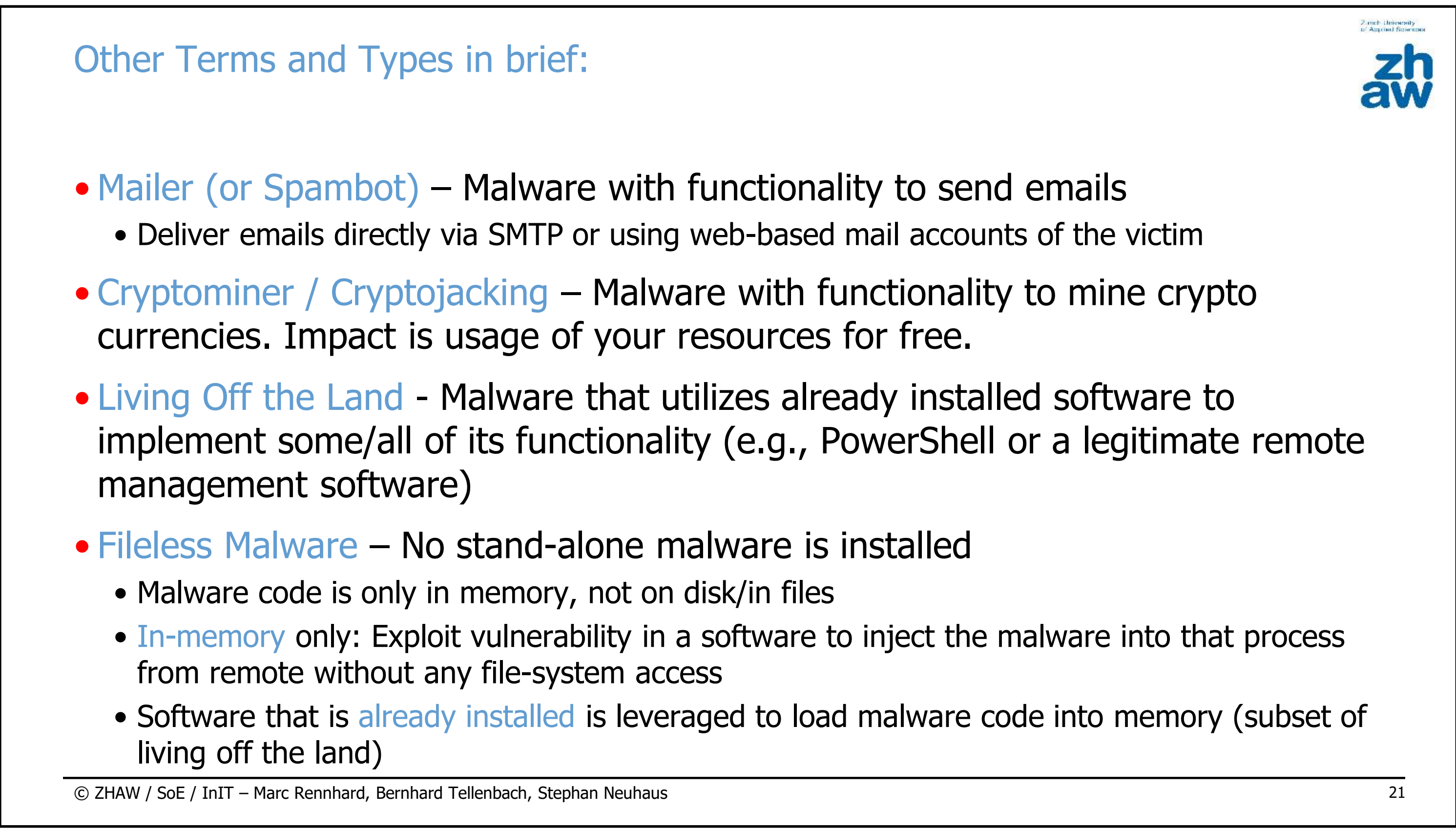
ILOVEYOU Worm: Sometimes referred to as Love Bug or Love Letter for you, is a computer worm that infected over ten million Windows personal computers on and after 5 May 2000[1] local time in the Philippines when it started spreading as an email message with the subject line "ILOVEYOU" and the attachment "LOVE-LETTER-FOR-YOU.txt.vbs". The latter file extension ('vbs', a type of interpreted file) was most often hidden by default on Windows computers of the time (as it is an extension for a file type that is known by Windows), leading unwitting users to think it was a normal text file. Opening the attachment activates the Visual Basic script. The worm inflicts damage on the local machine, overwriting random types of files (including Office files, image files, and audio files; however, after overwriting MP3 files the virus hides the file), and sends a copy of itself to all addresses in the Windows Address Book used by Microsoft Outlook. This made it spread much faster than any other previous email worm.

Source: https://en.wikipedia.org/wiki/ILOVEYOU

Witty Worm: The Witty worm is a computer worm that **attacks the firewall and other computer security products** written by a particular company, Internet Security Systems (ISS). It was the first worm to take advantage of vulnerabilities in the very pieces of software designed to enhance network security, and carried a destructive payload, unlike previous worms. It is so named because the phrase "(^.^) insert witty message here (^.^)" appears in the worm's payload.

On March 19, 2004, it began infecting hosts connected to the Internet (and running the vulnerable ISS software) without any seed population. Within a half-hour it infected 12,000 computers and was generating ***90 Gbit/s (gigabits per second) of UDP traffic***. Witty uses a vulnerability in ICQ instant messaging protocol parsing routines of the ISS Protocol Analysis Module (PAM).

Based on source: https://www.caida.org/research/security/witty/



Other Terms and Types in brief:

- Mailer (or Spambot) – Malware with functionality to send emails
  - Deliver emails directly via SMTP or using web-based mail accounts of the victim
- Cryptominer / Cryptojacking – Malware with functionality to mine crypto currencies. Impact is usage of your resources for free.
- Living Off the Land - Malware that utilizes already installed software to implement some/all of its functionality (e.g., PowerShell or a legitimate remote management software)
- Fileless Malware – No stand-alone malware is installed
  - Malware code is only in memory, not on disk/in files
  - In-memory only: Exploit vulnerability in a software to inject the malware into that process from remote without any file-system access
  - Software that is already installed is leveraged to load malware code into memory (subset of living off the land)

Living Off the Land is a term used to describe when malware utilizes already-installed software to implement some or all of its functionality. As Microsoft has improved the capability of interfaces such as WMI, PowerShell, Windows Subsystem for Linux, and others, this technique has become more popular as adversaries can rely upon Microsoft-licensed code to do a lot more heavy-lifting. The benefit is often two-fold: Less code needs to be authored in-house by malware authors

Genuine software processes distributed by trusted vendors will do the work, which is less likely to be caught as malicious activity.

A common use of this is, once an adversary gets a backdoor installed onto a system, the may resort to these techniques to fill gaps in the backdoor or RAT's capabilities. For instance, they may resort to using reg.exe for modifying the Windows Registry, or crontab on a UNIX system to provide a mechanism for restarting the backdoor channel when it gets disconnected.

Some great documentation is available here:

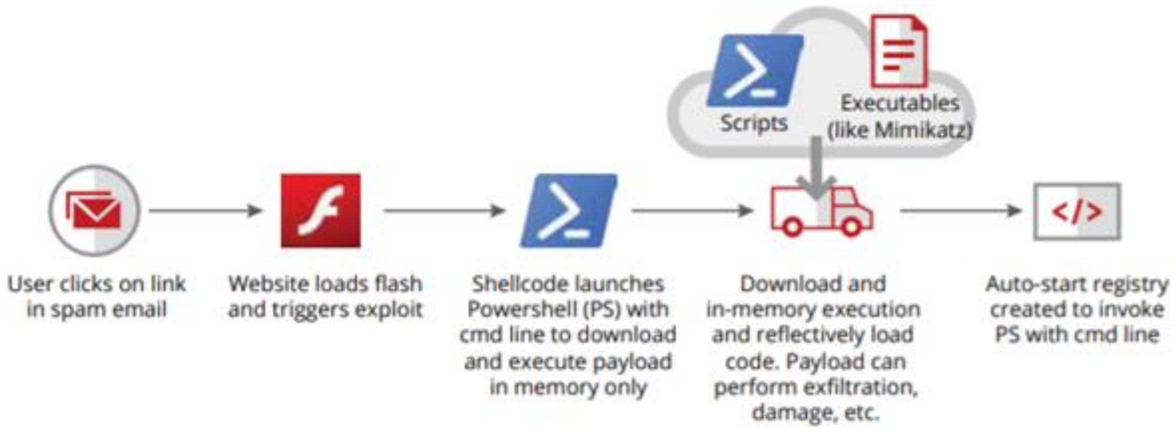
- https://lolbas-project.github.io/ (Windows-focused)
- https://gtfobins.github.io/ (Linux-focused, many would likely work on MacOS X too)

Source: https://class.malware.re/2020/01/19/malware-taxonomy.html

Fileless Malware - An emerging term that you will encounter is "Fileless" or "File-less" malware. The term often is used to describe attacks that employ a lot of the existing software on a system to execute malware, largely in memory. This can largely be seen as a very specific subset of *Living Off the Land* techniques, where the existing infrastructure of the OS is used even further to establish the system compromise in a manner intended to evade forensic analysis. While it is named *fileless*, the truth is that it is largely *file-limited*. It is truly difficult to get away from files altogether, because most underlying operating systems rely upon file-like objects at a low level to get data into or out of the system.

Be advised that different vendors have differing opinions on what constitutes *fileless…*

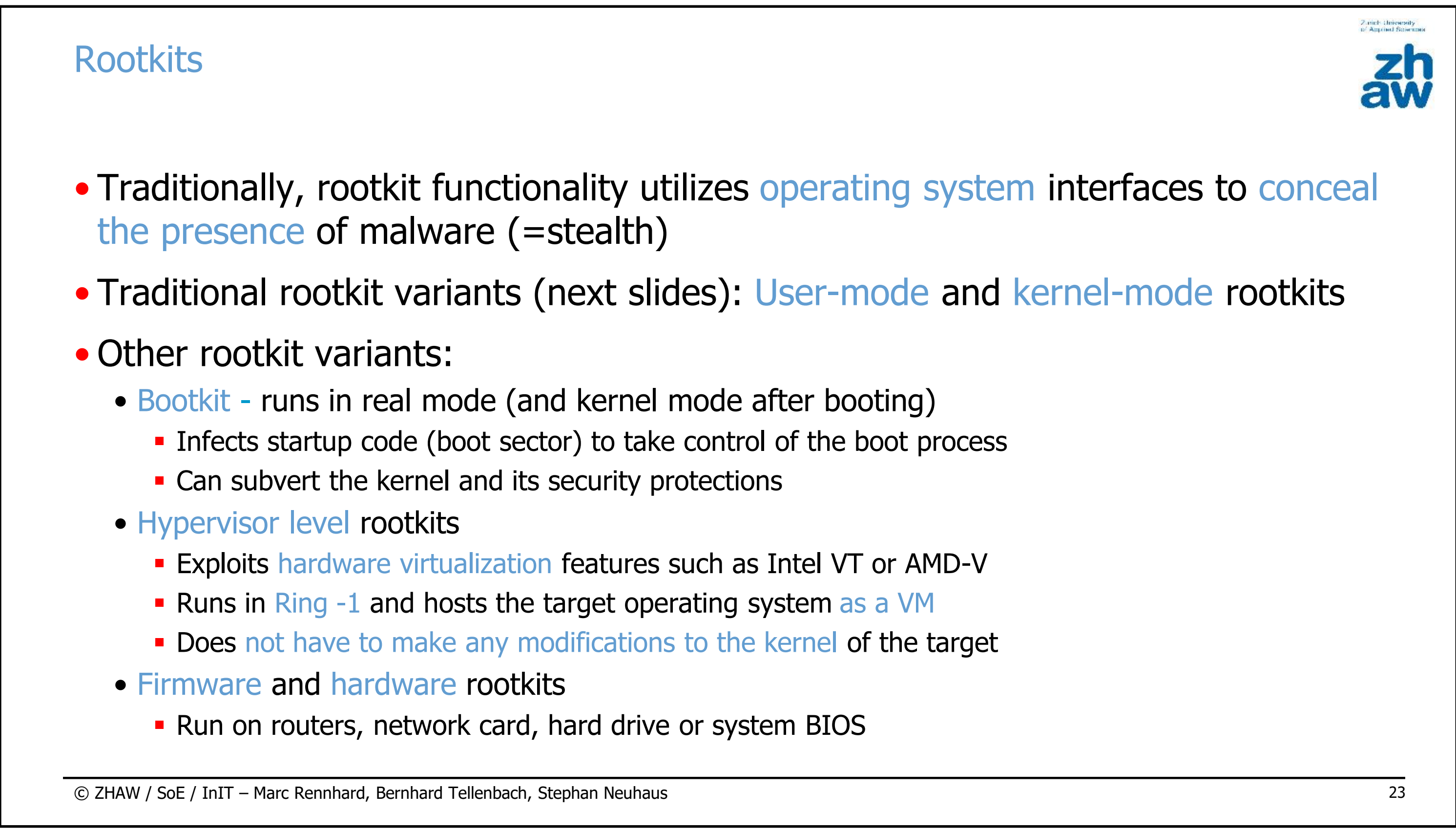
Fileless or not Fileless?


Scripts
Executables (like Mimikatz)
User clicks on link in spam email
Website loads flash and triggers exploit
Shellcode launches Powershell (PS) with cmd line to download and execute payload in memory only
Download and in-memory execution and reflectively load code. Payload can perform exfiltration, damage, etc.
Auto-start registry created to invoke PS with cmd line

In the above “fileless” attack, you can clearly see that the user received an email (which is a file, of sorts, that lives in the user’s inbox), and this contained a link to an HTML page on a malicious website, which contains an embedded tag to play an Adobe Flash animation. The HTML page and the Adobe Flash (SWF) animation are both files, as well, with the SWF being a small compiled application that executes within the Adobe Flash context, inside of the user’s web browser. The flash animation then executes PowerShell to download and execute a payload entirely within memory.

While no file is written to disk during this step, a file hosted on a remote server is fetched by the PowerShell process, and stored entirely in memory. Additional tools are referenced, like mimikatz (a popular password [info] stealer), that are hosted remotely as distinct files, and downloaded to the local system into RAM. Some of this RAM may additionally be swapped to disk in pagefile.sys, or a compromised system may be hibernated to create a hiberfile.sys that may also contain useful forensic artifacts. Likewise, if an HTTP or HTTPS proxy is employed by the target, there is an opportunity for all of the downloaded files to have been archived locally, either by forensic data retention, or possibly by caching logic intended to speed up web traffic.

So, in the above McAfee example, the attack they describe as *fileless* really seems to consist of no less than 5 files, but limits the storage of them into volatile memory on the host, expecting that, in a panic, the target may simply power off all compromised systems in reaction to discovering malicious activity, destroying a lot of the evidence in the process.



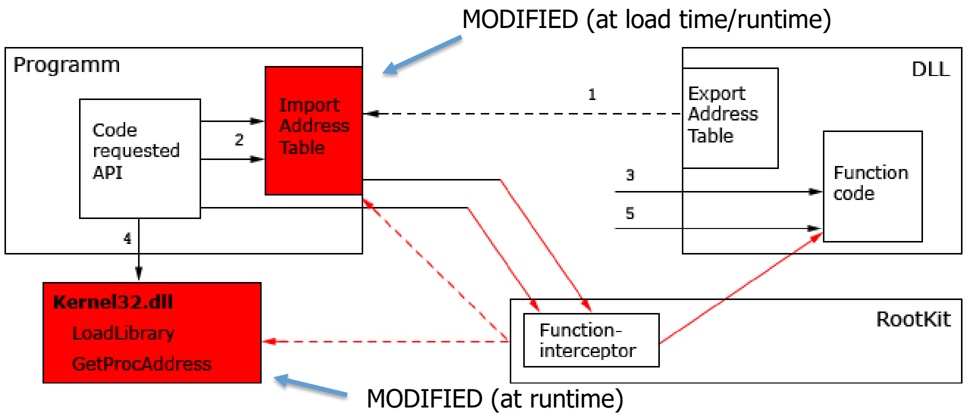
Rootkits

- Traditionally, rootkit functionality utilizes operating system interfaces to conceal the presence of malware (=stealth)
- Traditional rootkit variants (next slides): User-mode and kernel-mode rootkits
- Other rootkit variants:
  - Bootkit - runs in real mode (and kernel mode after booting)
    - Infects startup code (boot sector) to take control of the boot process
    - Can subvert the kernel and its security protections
  - Hypervisor level rootkits
    - Exploits hardware virtualization features such as Intel VT or AMD-V
    - Runs in Ring -1 and hosts the target operating system as a VM
    - Does not have to make any modifications to the kernel of the target
  - Firmware and hardware rootkits
    - Run on routers, network card, hard drive or system BIOS

A good starting point for further reading on this topic is again wikipedia (the referenced sources).

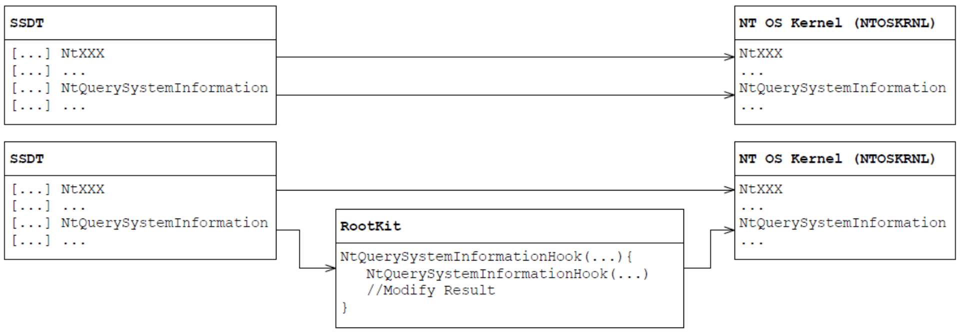
https://en.wikipedia.org/wiki/Rootkit

Rootkits – User Mode



- Run in user space and work best with administrative privileges
- Alter security settings and hide processes, files, system drivers, network ports, and even system services
- Hiding techniques: E.g., function hooking of widely used API in every process

Rootkits – Kernel-Mode



- Loaded as device drivers / loadable kernel modules => requires to be able to install a driver (kernel module)
- Many different hiding techniques
- E.g., SSDT (System Service Descriptor Table) hooking

SSDT (System Service Descriptor Table) is a system service dispatch table that contains entry point addresses of NT kernel services in Windows. The above figure show normal execution of a system routine “NtQuerySystemInformation” vs. the execution of the same system routine after the rootkit replaced an SSDT pointer to its own routine. This way, a rootkit can control system routines, filtering out “unwanted” data.

Detection of SSDT Hooking:

All pointers referenced in SSDT must refer to routines implemented in either “nt” or “win32k” library. Therefore, when checking these pointers for interceptions, one must verify whether SSDT pointers actually refer to one of those memory areas. However, there might be “legitimate” use of such hooking.

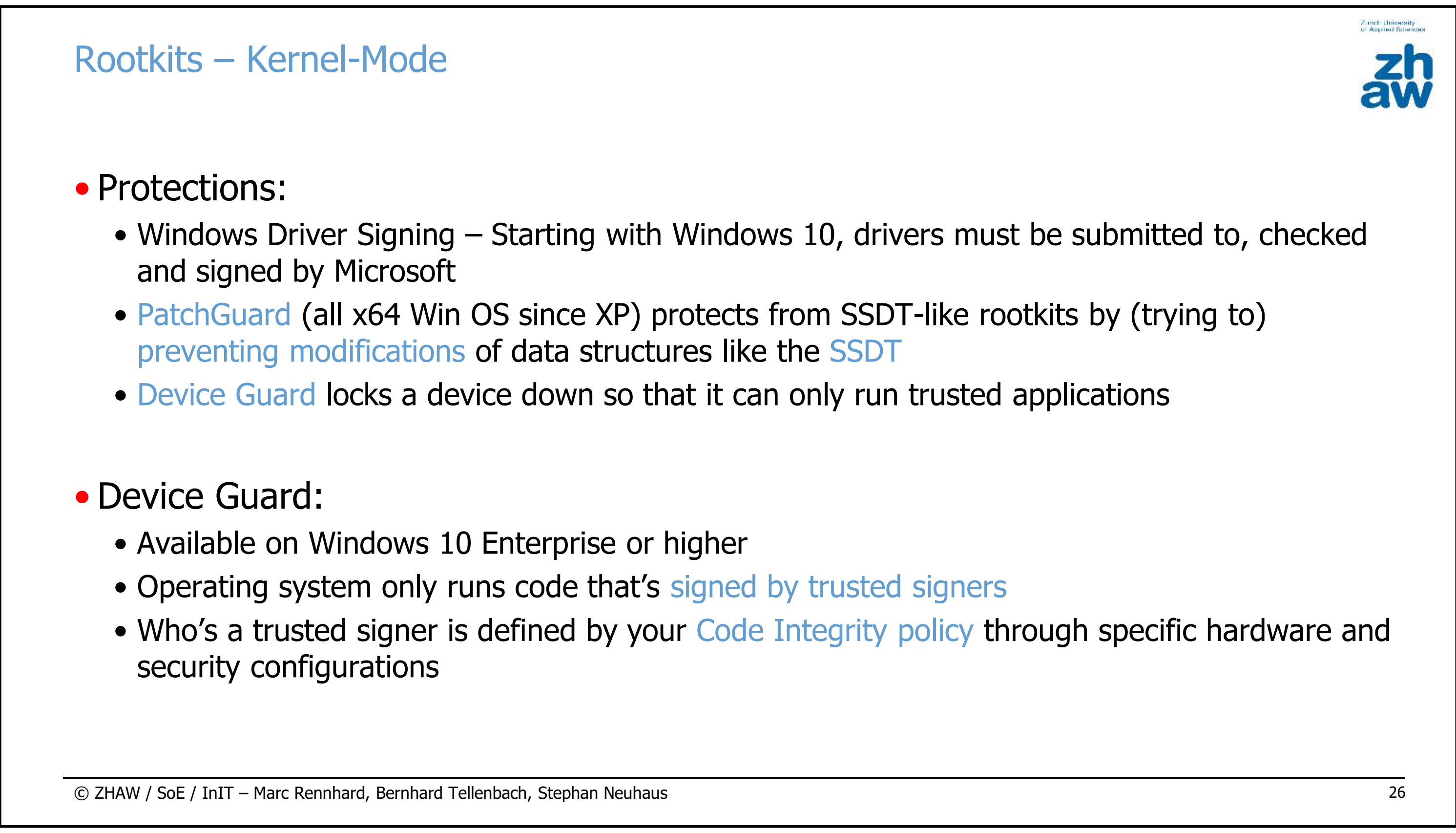
Driver Signing

By default, 64-bit versions of Windows Vista and later versions of Windows will load a kernel-mode driver only if the kernel can verify the driver signature. However, this default behaviour can be disabled to during early driver development and for non-automated testing.

https://msdn.microsoft.com/en-us/library/windows/hardware/ff547565%28v=vs.85%29.aspx

There are other options not mentioned in the Microsoft article:

https://techjourney.net/workarounds-to-load-unsigned-driver-in-windows-cannot-verify-the-digital-signature-of-low-level-driver/



Rootkits – Kernel-Mode

- Protections:
  - Windows Driver Signing – Starting with Windows 10, drivers must be submitted to, checked and signed by Microsoft
  - PatchGuard (all x64 Win OS since XP) protects from SSDT-like rootkits by (trying to) preventing modifications of data structures like the SSDT
  - Device Guard locks a device down so that it can only run trusted applications
- Device Guard:
  - Available on Windows 10 Enterprise or higher
  - Operating system only runs code that's signed by trusted signers
  - Who's a trusted signer is defined by your Code Integrity policy through specific hardware and security configurations

Device Guard in Windows 10 Enterprise

Device Guard is a combination of enterprise-related hardware and software security features that, when configured together, will lock a device down so that it can only run trusted applications. If the app isn't trusted it can't run, period. It also means that even if an attacker manages to get control of the Windows kernel, he or she will be much less likely to be able to run malicious executable code after the computer restarts because of how decisions are made about what can run and when.

Device Guard uses the new virtualization-based security in Windows 10 Enterprise to isolate the Code Integrity service from the Microsoft Windows kernel itself, letting the service use signatures defined by your enterprise-controlled policy to help determine what is trustworthy. In effect, the Code Integrity service runs alongside the kernel in a Windows hypervisor-protected container.

How Device Guard works

Device Guard restricts the Windows 10 Enterprise operating system to only running code that's signed by trusted signers, as defined by your Code Integrity policy through specific hardware and security configurations, including:

- User Mode Code Integrity (UMCI)
- New kernel code integrity rules (including the new Windows Hardware Quality Labs (WHQL) signing constraints)
- Secure Boot with database (db/dbx) restrictions
- Virtualization-based security to help protect system memory and kernel mode apps and drivers from possible tampering.
- **Optional:** Trusted Platform Module (TPM) 1.2 or 2.0

Device Guard works with your image-building process, so you can turn the virtualization-based security feature on for capable devices, configure your Code Integrity policy, and set any other operating system settings you require for Windows 10 Enterprise. After that, Device Guard works to help protect your devices:

- Your device starts up using Universal Extensible Firmware Interface (UEFI) Secure Boot, so that boot kits can't run and so that Windows 10 Enterprise starts before anything else.
- After securely starting up the Windows boot components, Windows 10 Enterprise can start the Hyper-V virtualization-based security services, including Kernel Mode Code Integrity. These services help protect the system core (kernel), privileged drivers, and system defenses, like anti-malware solutions, by preventing malware from running early in the boot process, or in kernel after startup.
- Device Guard uses UMCI to make sure that anything that runs in User mode, such as a service, a Universal Windows Platform (UWP) app, or a Classic Windows application is trusted, allowing only trusted binaries to run.
- At the same time that Windows 10 Enterprise starts up, so too does the trusted platform module (TPM). TPM provides an isolated hardware component that helps protect sensitive information, such as user credentials and certificates.

Source: https://msdn.microsoft.com/en-us/library/dn986865%28v=vs.85%29.aspx

Detecting Rootkits

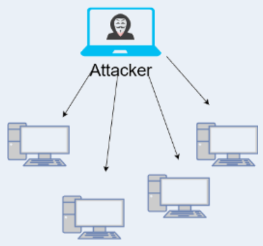
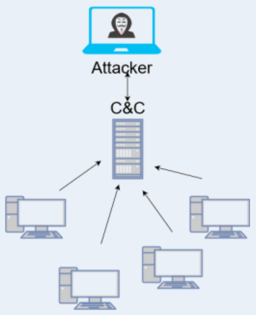
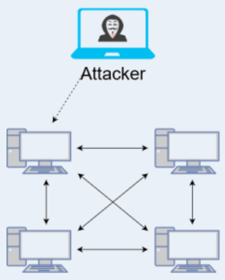
- Based on the «concept» used for hiding
  - E.g., look for alteration of the SSDT
- Detection of sophisticated rootkits at ring 0 or lower is difficult
- Timing analysis as a generic detection method for rootkits
  - Hooked/modified operations show a different timing behavior
  - Baseline to compare to is the challenge
  - In case of hypervisor level rootkits, external time sources must be used

Malware Communication

Introduction

- Many malware types need to communicate with the attacker(s) or other malware instances to fulfil their purpose
- Communication consists of the communication architecture and the properties of the communication channel itself
- Important goals for malware communication:
  - Communication should be resilient vs. blocking/takedowns
  - Communication should be hard to detect and/or identify

Communication Architectures

	Direct	Client-Server	P2P
Communication initiated by	attacker	malware	attacker/malware
Firewall compatibility	low	high	low-medium
Resilience / Ease of takeout	easy [hard]	easy [hard]	hard
Architecture Outline			

Direct: In this architecture, the attacker exert direct control over their victims. The attacker connects to the victim and disseminates commands.

Here, the knowledge which victims exist and how to reach them (e.g., hostname / IP, email address,…) is with the attacker only.

This architecture lost popularity because if the attacker directly communicates with the victim, it is easy to identify the attacker (e.g., by its IP address). That’s also why takedown is easy – arrest the attacker or block communication from the attacker. However, if the attacker makes use of indirection techniques to hide the true origin of the connection to make it resilient vs. take-down, the communication architecture might still have some applications (e.g., targeted attacks with server compromises).

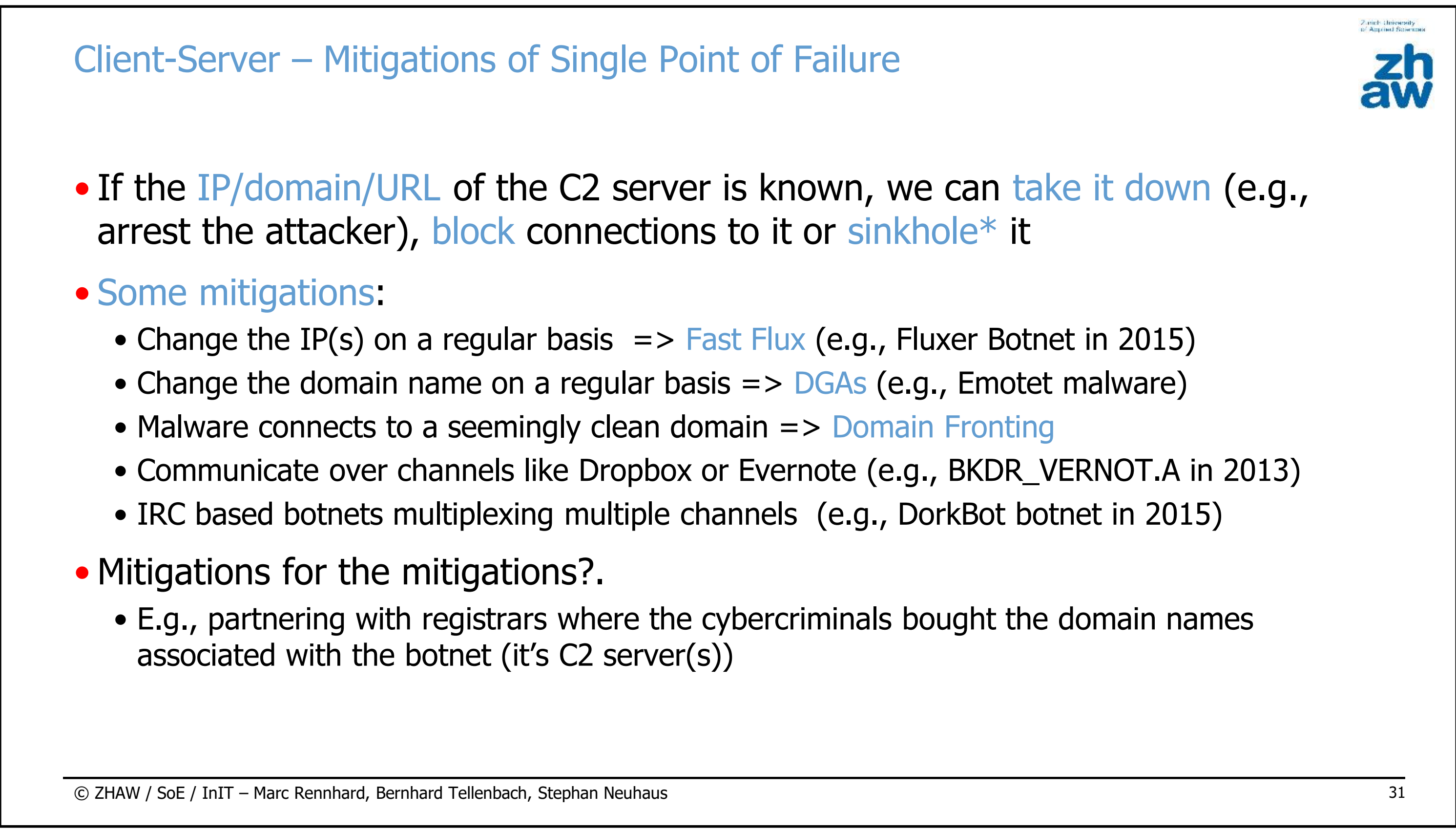
Another drawback of this architecture is that it has some difficulties with firewalls since most firewall configurations today permit outgoing connections only by default. At least for workstations and other non-server devices.

Client-Server: In the Client-Server architecture all victims communicate with a server controlled by the attacker. This server is usually called the command-and-control (C&C/C2) server. Connections to it are often initiated on a regular basis to check whether new instructions/commands are queued for execution by the victim.

As with the direct architecture, this architecture lost popularity because it is easy to locate the C&C server and therefore also to take it down or block it. However, the picture changes if indirection techniques to hide the true location of the C&C server are used. Client-Sever in combination with indirection techniques is probably the most popular architecture right now.

The high level of compatibility with firewalls and the “natural” communication pattern contributes to its merit. With natural communication patter we refer to the pattern that a computer contacts a server to do something, which is for most non-server machines the “normal” behavior.

Peer-to-Peer (P2P): In the P2P architecture, each victim has the capability to be a client (e.g., doing malicious stuff on the victim’s machine) and a server connecting to at least one other victim. Using the P2P method, the attacker has no need to maintain communication with all of the victims. Especially tasks like searching for new victims and adding them to the architecture is often automated and does not need any back-channel to the attacker. To distribute commands, an attacker can run itself an instance of the malware. While this makes locating the attacker already difficult (depending on the routing and addressing scheme of the architecture), an attacker can make this even more difficult by using indirection techniques (e.g., public VPNs or TOR) to connect to the P2P network. But its biggest advantage is the lack of a single point of failure – taking down a node or a set of nodes or blocking communication to some nodes won't help to take down the entire P2P network. At least not, if the P2P architecture and protocol is well-designed and implemented.



Client-Server – Mitigations of Single Point of Failure

- If the IP/domain/URL of the C2 server is known, we can take it down (e.g., arrest the attacker), block connections to it or sinkhole* it
- Some mitigations:
  - Change the IP(s) on a regular basis => Fast Flux (e.g., Fluxer Botnet in 2015)
  - Change the domain name on a regular basis => DGAs (e.g., Emotet malware)
  - Malware connects to a seemingly clean domain => Domain Fronting
  - Communicate over channels like Dropbox or Evernote (e.g., BKDR_VERNOT.A in 2013)
  - IRC based botnets multiplexing multiple channels (e.g., DorkBot botnet in 2015)
- Mitigations for the mitigations?.
  - E.g., partnering with registrars where the cybercriminals bought the domain names associated with the botnet (it's C2 server(s))

***Sinkhole** – This is the practice to redirect traffic from its original destination (e.g., the C2 server) to one specified by the sinkhole owner. This is done by cooperating with the Internet service providers, DNS operators and registrars. For example, within the SWITCH network, you are redirected you to a landing page with a warning, if you try to access a known C&C server. Instead of redirection you to a landing page and blocking the access, in the case of sinkholing you are redirected to an "alternative" C&C server. This might be desirable in at least two cases:

- If the malware would start destroying stuff on the victim's machine if it can no longer reach the C&C server for some time
- If one does not know yet whether the malware would do the former because more time for analysis is needed. To cut-off the attacker in the meantime, an "alternative" C&C server that does only tell the malware to do "nothing" is used.

Fast Flux – What it is

- Basic idea: A single fully qualified domain name, where IP addresses are swapped in and out with high frequency (TTL <300s), through changing DNS records.
- Fast flux is a DNS technique to hide a machine behind an ever-changing network of compromised hosts acting as proxies.
  - Makes malware networks more resistant to IP based blocking
  - The Storm Worm (2007) was one of the first malware variants to make use of this technique.
  - It can also refer to the combination of peer-to-peer networking, distributed command and control, web-based load balancing and proxy redirection
- How to defend?
  - Block the fully qualified domain or take it down with the help of the registrars/providers where the domains were registered

Fast Flux - Visualization

DNS Robtex Analysis of a Fast flux domain

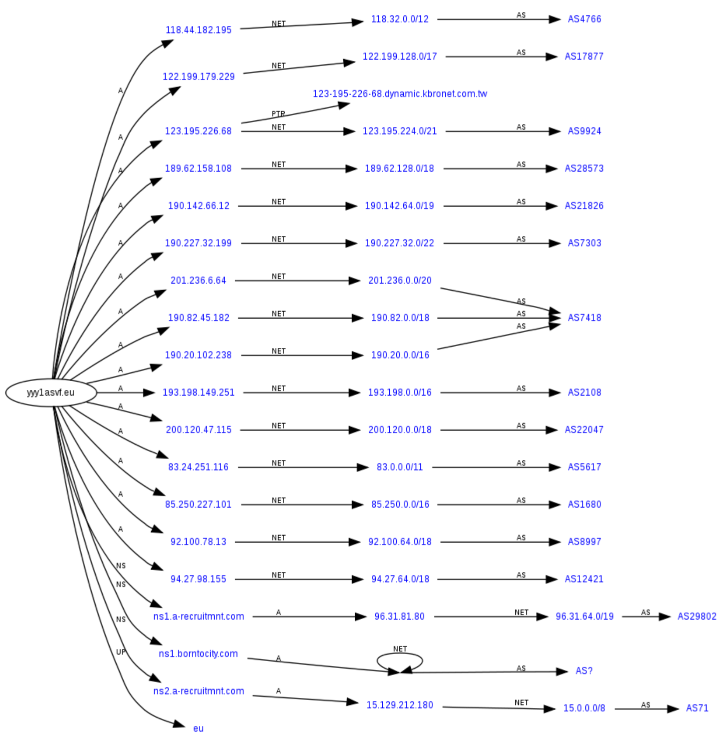


Image source: Source: https://en.wikipedia.org/wiki/Fast_flux

Domain Generation Algorithm (DGA)

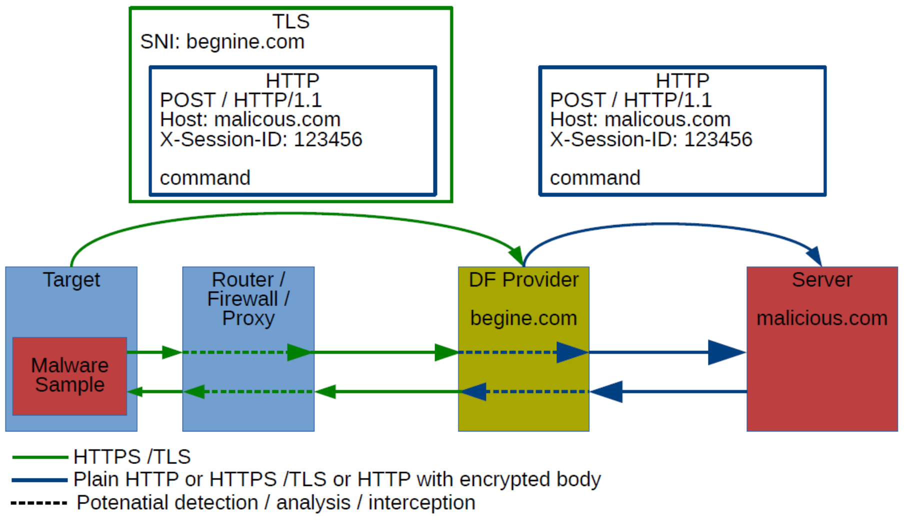
- Thwart “bad domain” list defense with domain generation algorithm
- Used to periodically generate many domain names that can be used as rendezvous points with their C2 servers.
- The large number of potential rendezvous points makes it difficult for law enforcement to effectively shut down botnets

```python
def generate_domain(year, month, day):
    """Generates a domain name for the given date."""
    domain = ""

    for i in range(16):
        year = ((year ^ 8 * year) >> 11) ^ ((year & 0xFFFFFFF0) << 17)
        month = ((month ^ 4 * month) >> 25) ^ 16 * (month & 0xFFFFFFF8)
        day = ((day ^ (day << 13)) >> 19) ^ ((day & 0xFFFFFFFE) << 12)
        domain += chr(((year ^ month ^ day) % 25) + 97)

    return domain
```

Domain Fronting



- Communicate with your C2 server using a totally legitimate domain

Domain fronting is a technique that circumvents Internet censorship by hiding the true endpoint of a connection. Working in the application layer, domain fronting allows a user to connect to a blocked service over HTTPS, while appearing to communicate with an entirely different site.[1]

The technique works by using different domain names at different layers of communication. The domain name of an innocuous site is used to initialize the connection. This domain name is exposed to the censor in clear-text as part of the DNS request and the TLS Server Name Indication. The domain name of the actual, blocked endpoint is only communicated after the establishment of an encrypted HTTPS connection, in the HTTP Host header, making it invisible to censors. This can be done if the blocked and the innocuous sites are both hosted by the same large provider, such as Google App Engine. For any given domain name, censors are typically unable to differentiate circumvention traffic from legitimate traffic. As such, they are forced to either allow all traffic to the domain name, including circumvention traffic, or block the domain name entirely, which may result in expensive collateral damage.

Source: https://en.wikipedia.org/wiki/Domain_fronting

A **Content Delivery Network** is a system of distributed servers that deliver web pages and other web content to users based on their geographic locations, to improve availability and performance.

Example in the wild: APT29 has used The Onion Router (TOR) and the TOR domain fronting plugin meek to create a hidden, encrypted network tunnel that appeared to connect to Google services over TLS. This tunnel provided the attacker remote access to the host system using the Terminal Services (TS), NetBIOS, and Server Message Block (SMB) services, while appearing to be traffic to legitimate websites. The attackers also leveraged a common Windows exploit to access a privileged command shell without authenticating.

Source and more information: https://www.fireeye.com/blog/threat-research/2017/03/apt29_domain_frontin.html

Malware Communication

- Most common: HTTPS with or without content-level encryption
- Advanced concept: Smart communication/exfiltration
  - What protocols are used [HTTP, HTTPS, SSH, …]?
  - What applications are used [Dropbox, Google docs, Citrix, …]?
  - How much data is sent/received and when?

  => Communication should adapt to what is "normal"
- Side Channels to send out data even if communication is over an encrypted tunnel only (e.g., VPN only)
  - E.g., when machine is on WLAN, malware could delay packets to create specific packet inter-arrival patterns
- DNS Covert Channel when isolated from the Internet might work

Why content-level encryption?

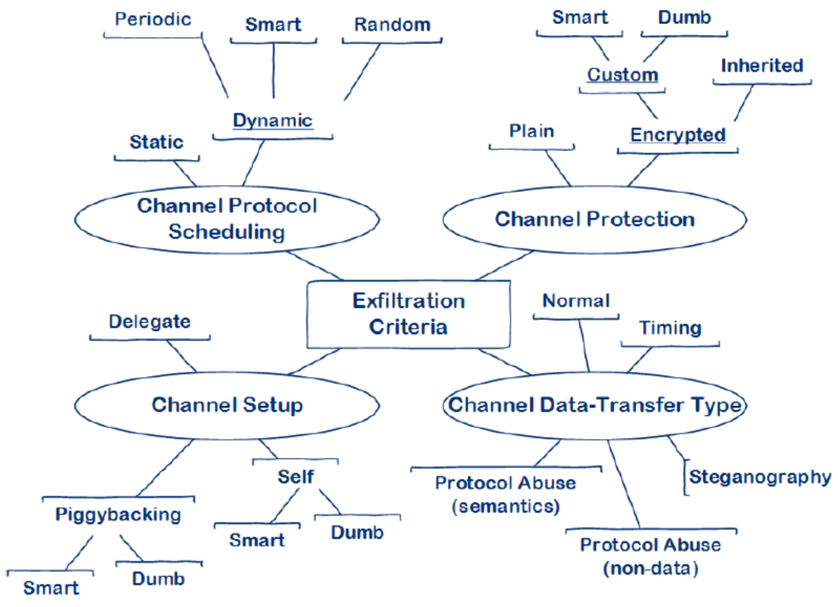
In company setups, https traffic is not necessarily end-to-end encrypted. Companies might use secure web gateways (see [1]) or SASE (see [2]) solutions to "break up" such connections and do deep packet inspection (DPI). To avoid detection of malicious activities, attackers might consider content-level encryption.

However, since content-level encryption itself might be considered suspicious, attackers might have to choose encryption methods (or steganography) that make the encrypted content look like normal content.

[1] https://www.cloudflare.com/en-gb/learning/access-management/what-is-a-secure-web-gateway/

[2] https://en.wikipedia.org/wiki/Secure_access_service_edge

Attributes of Communication Channels



Channel Protocol Scheduling

- Static - The protocol used never changes.
- Dynamic – The protocol/technique changes as follows:
  - Random: A protocol is chosen randomly from a set of available protocols and might be changed after a certain amount of time or data
  - Smart – In contrast to the random method, the selection of the protocol is done based on information collected by the malware and a set of rules. Network usage on the infected machine to detect common used protocols and timing properties might be analysed. This method is less prone to detection by anomaly-detection solutions
  - Periodic: Protocol might be switched in regular intervals

Channel Protection

- Different types of encryption can be used including encryption that might change the algorithm used (e.g. use weaker encryption mechanisms or steganography to avoid detection by entroyp methods).

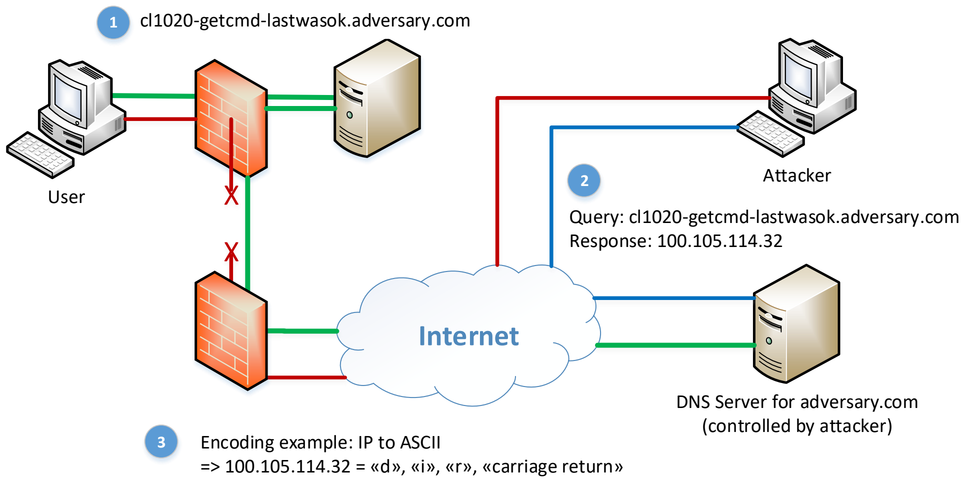
Channel Setup

- The malware can create a channel itself or piggiback a channel already there (e.g., modify timings of packets sent by delaying them in a specific pattern to encode information in the timings)

Channel Data-Transfer Type

- Information can be encoded in different ways – In the content in plain sight or hidden with steganography, encoded in timings such as packet inter-arrival times or by abusing (unused) protocol fields or semantics not intended to be used for data transfer.

Communication – Covert Channel using DNS


1
cl1020-getcmd-lastwasok.adversary.com
User
Attacker
2
Query: cl1020-getcmd-lastwasok.adversary.com
Response: 100.105.114.32
Internet
DNS Server for adversary.com
(controlled by attacker)
3
Encoding example: IP to ASCII
=> 100.105.114.32 = «d», «i», «r», «carriage return»

If an infected system needs any information about external systems, an internal DNS server resolving these queries is available to them. Consequently, even if the infected system is allowed no direct access to the Internet at all, it can still communicate by implementing a covert channel utilizing recursive DNS lookups. At least as long as the DNS server is not configured to allow the lookup of internal hostnames or domain names only for such hosts (or a filter for such requests is in place) => DNS must be configured accordingly!

Note that this channel is quite easy to detect, if it is used to exfiltrate data since the number of DNS queries will be quite high. Furthermore, queries are likely to show strange DNS name patterns. However, one has to monitor DNS traffic for such anomalies and block corresponding queries (or redirect them to a «non-malicious» DNS if we know how to respond to a query from the malware so that it remains «idle»).

Zurich University of Applied Sciences

Appendix

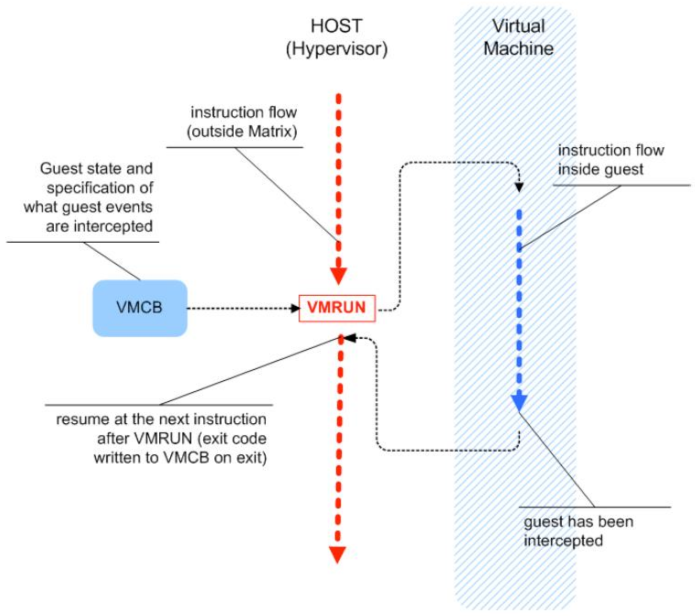
Famous Example of a Hypervisor-level Rootkit

- Blue Pill rootkit for Windows Vista x64
- Introduced by researcher Joanna Rutkowska in 2006
- Blue Pill idea:
  - Exploit AMD64 SVM extension (later also Intel VT) to move the operating system into a virtual machine ‘on-the-fly’
  - Provide thin hypervisor to control the OS
  - Hypervisor is responsible for controlling interesting events inside guest OS
- The VMRUN instruction is the heart of AMD64 SVM (and Blue Pill)

Secure Virtual Machine (AMD SVM) extension (AKA Pacifica released in 2006) is a set of instructions which can be used to implement Secure Virtual Machines on AMD64 machines. *support.amd.com/TechDocs/24594.pdf*

The **Blue Pill** concept is to trap a running instance of the operating system by starting a thin hypervisor and virtualizing the rest of the machine under it. The previous operating system would still maintain its existing references to all devices and files, but nearly anything, including hardware interrupts, requests for data and even the system time could be intercepted (and a fake response sent) by the hypervisor. The original concept of Blue Pill was published by another researcher at IEEE Oakland on May 2006, under the name VMBR (virtual-machine based rootkit).

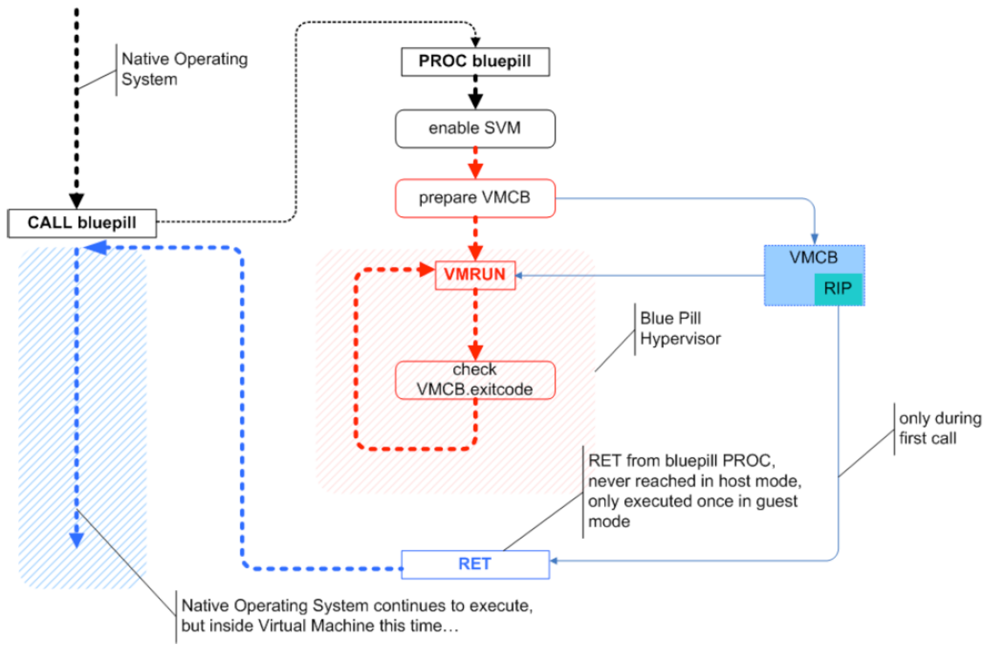
Blue Pill – The VMRUN Instruction (heart of AMD64 SVM)


HOST (Hypervisor)
Virtual Machine
instruction flow (outside Matrix)
Guest state and specification of what guest events are intercepted
instruction flow inside guest
VMCB
VMRUN
resume at the next instruction after VMRUN (exit code written to VMCB on exit)
guest has been intercepted

Source: Joanna Rutkowska, Subverting Vista Kernel for Fun and Profit, SyScan 2006, Singapore

VMCB = Virtual Machine Control Block

Blue Pill – The Basic Idea (simplified)



Native Operating System
CALL bluepill
PROC bluepill
enable SVM
prepare VMCB
VMRUN
check VMCB.exitcode
VMCB
RIP
Blue Pill Hypervisor
only during first call
RET from bluepill PROC, never reached in host mode, only executed once in guest mode
RET
Native Operating System continues to execute, but inside Virtual Machine this time…

Source: Joanna Rutkowska, Subverting Vista Kernel for Fun and Profit, SyScan 2006, Singapore

Note: With the VMCB already established for each core of the CPU and initialized with the state of the OS, shifting of the OS to guest mode is done very simply by running the VMRUN instruction with the RAX register as its single required operand. The RAX register is a pointer to the 64-bit physical address of the VMCB. At this point, code execution and control flow of the guest OS will continue seamlessly and transparently without it ever being aware that it has been migrated away from having direct control of hardware to within a VM under the control of a hypervisor.

A detailed analysis and comprehensive guide to Blue Pill can be found here:

Robert C. Fannon, AN ANALYSIS OF HARDWARE-ASSISTED VIRTUAL MACHINE BASED ROOTKITS, Thesis, Naval Postgraduate School, June 2014
http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA607759