599.1
Defeating Advanced Adversaries

SANS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

SANS

Defeating Advanced Adversaries

Welcome to SANS Security SEC599.1: Defeating Advanced Adversaries.

In this course, you will build essential skills required to fend off today's advanced cyber attacks. The course will be highly hands-on, as we help you develop skills by exercising them in hands-on, realistic lab settings. Although this is not a penetration testing course, we will have sufficient attention for the offensive side of the spectrum. We will provide you with a deep technical understanding of how advanced adversaries work, as this will help us be more efficient defenders. Likewise, we will inform you on how to respond to cyber security attacks, but will primarily focus on how to prevent and detect them.

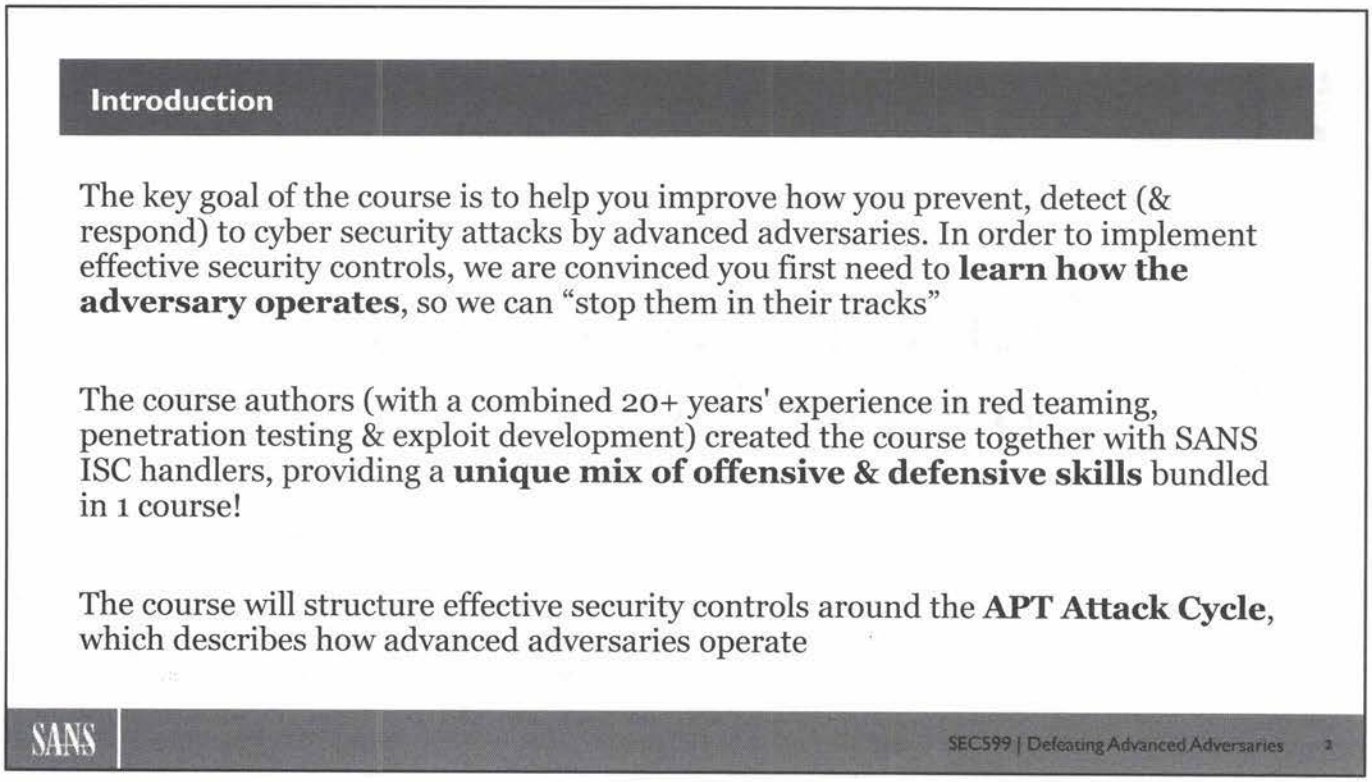
Our goal is to keep the course as interactive as possible. If you have a question, please let the instructor know. Discussions about relevant topics are incredibly important in a class like this, as we have numerous attendees with various levels of skill coming into the class. Share your insights and ask questions. The instructor does reserve the right, however, to take a conversation offline during a break or outside of class in the interest of time and applicability of the topic.

As course authors, we welcome any comments, questions, or suggestions pertaining to the course material. We would also like to extend our thanks to Didier Stevens (a SANS ISC handler), whose contributions greatly helped improve the course.

Erik Van Buggenhout
erik.van.buggenhout@gmail.com
www.nviso.be

Stephen Sims
ssims@sans.org
www.sans.org

Update: C01_03

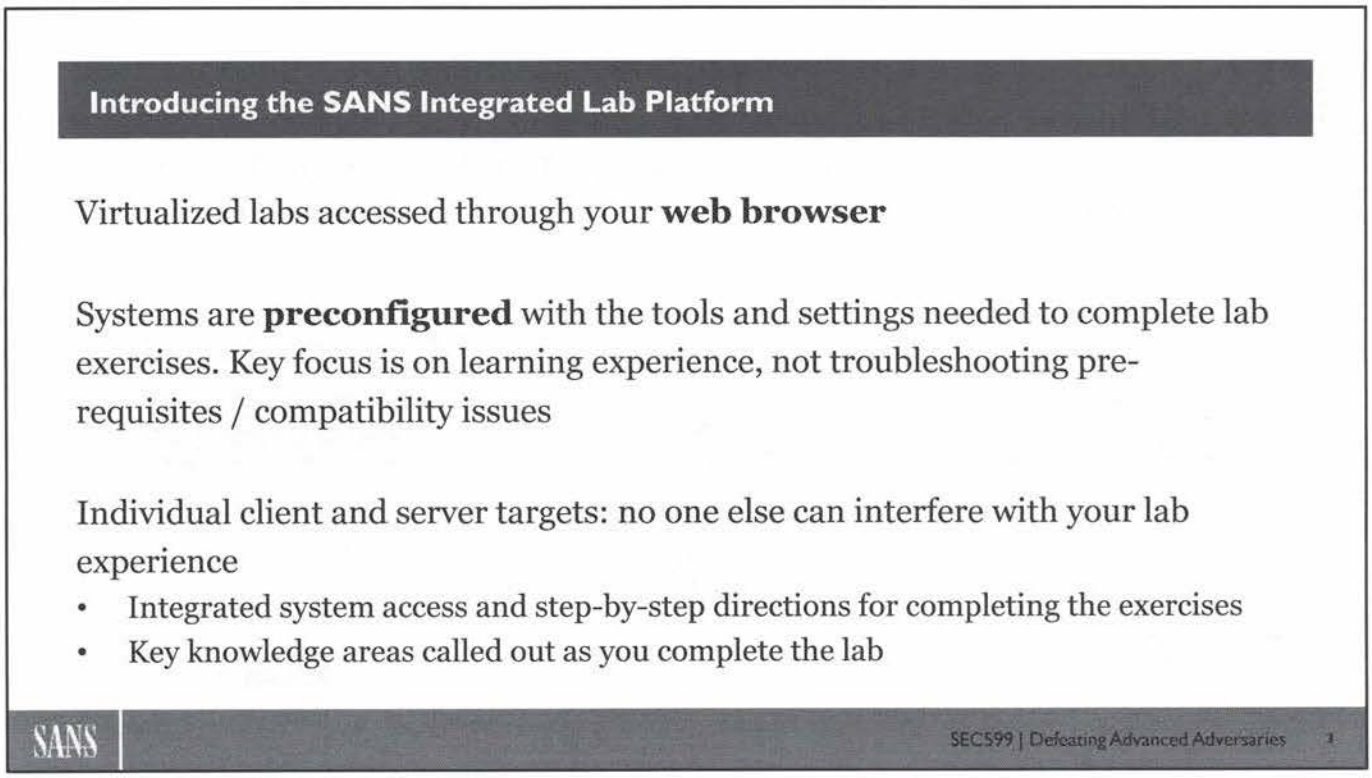


Introduction

The key goal of SEC599 "Defeating Advanced Adversaries" is to help you improve how you prevent and detect cyber security attacks by advanced adversaries. We will also cover techniques for effective incident response, although not in-depth, as SANS has dedicated courses that cover this topic (such as FOR508: Incident Response & Threat Hunting).

The course authors (with a combined 20+ years' experience in red teaming, penetration testing & exploit development) created the course together with SANS ISC handlers, providing a unique mix of offensive & defensive skills bundled in 1 course!

In order to implement effective security controls, we are convinced it is vital to first understand how adversaries operate. We will thus first explain offensive security techniques, explaining how organizations are currently being compromised. Based on this understanding, we will structure the attack in an APT Attack Cycle, we will list the different stages of the attack. Using these stages, we can understand how the attack operates and where / how effective security controls can be implemented.



Introducing the SANS Integrated Lab Platform

SANS is committed to providing a superior course with skills that you can use immediately when you get back to the office. A significant part of this course experience is the use of hands-on lab exercises designed to reinforce the topics we cover during lecture.

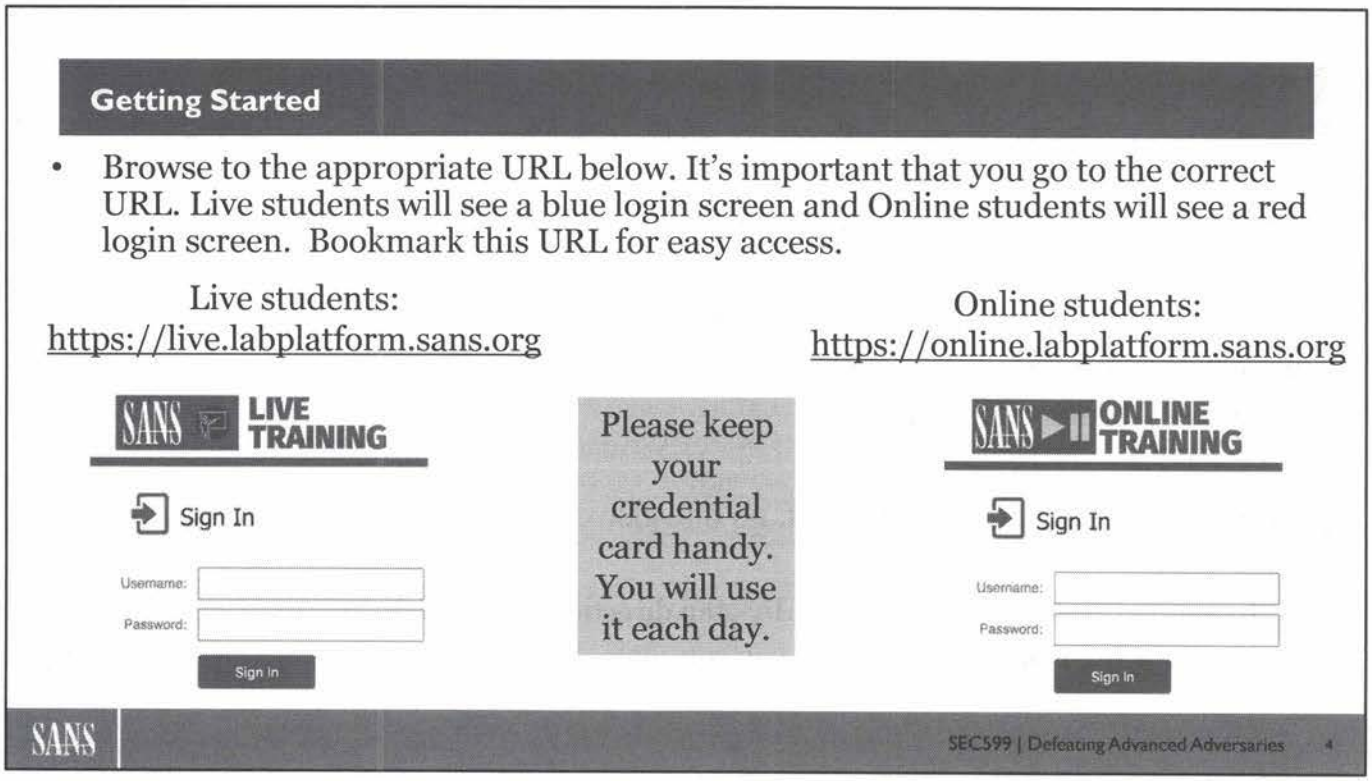
The SANS Integrated Lab Platform is an integrated lab and workbook environment, providing consistent and easy access to the client systems and server targets through your web browser. Through this platform, you simply browse to a URL and login, then you will be able to access all the client and server systems, and see the lab step-by-step directions needed to complete the lab exercises, in a single browser window.

The systems you will access through this platform have been preconfigured with the tools, software, and files needed to complete all the exercises. This allows you to focus on applying the learning objectives for the lab instead of spending valuable time configuring your laptop, troubleshooting network or conflicting software settings, and clicking Next, Next, Next, Next, Next, Finish over and over again.

Another benefit of the SANS Integrated Lab Platform is that you have individualized access to client and server systems. When you start a lab, the servers supporting the platform spin up a duplicate copy of the server and client systems needed to complete the lab, uniquely accessible to you. This stops other people in the classroom from interfering with your lab experience (intentionally or unintentionally), making the lab exercises more consistent and accessible.
Instead of flipping back and forth between a printed lab workbook and your laptop, the SANS Integrated Lab Platform integrates both the client or attacker system view with the step-by-step exercises.

The step-by-step directions in the lab call out key knowledge areas that are important to recognize, as well as alerts to make you aware of the need for caution when using a tool or completing a specific lab step, and screenshots to help you stay on track with the exercises.

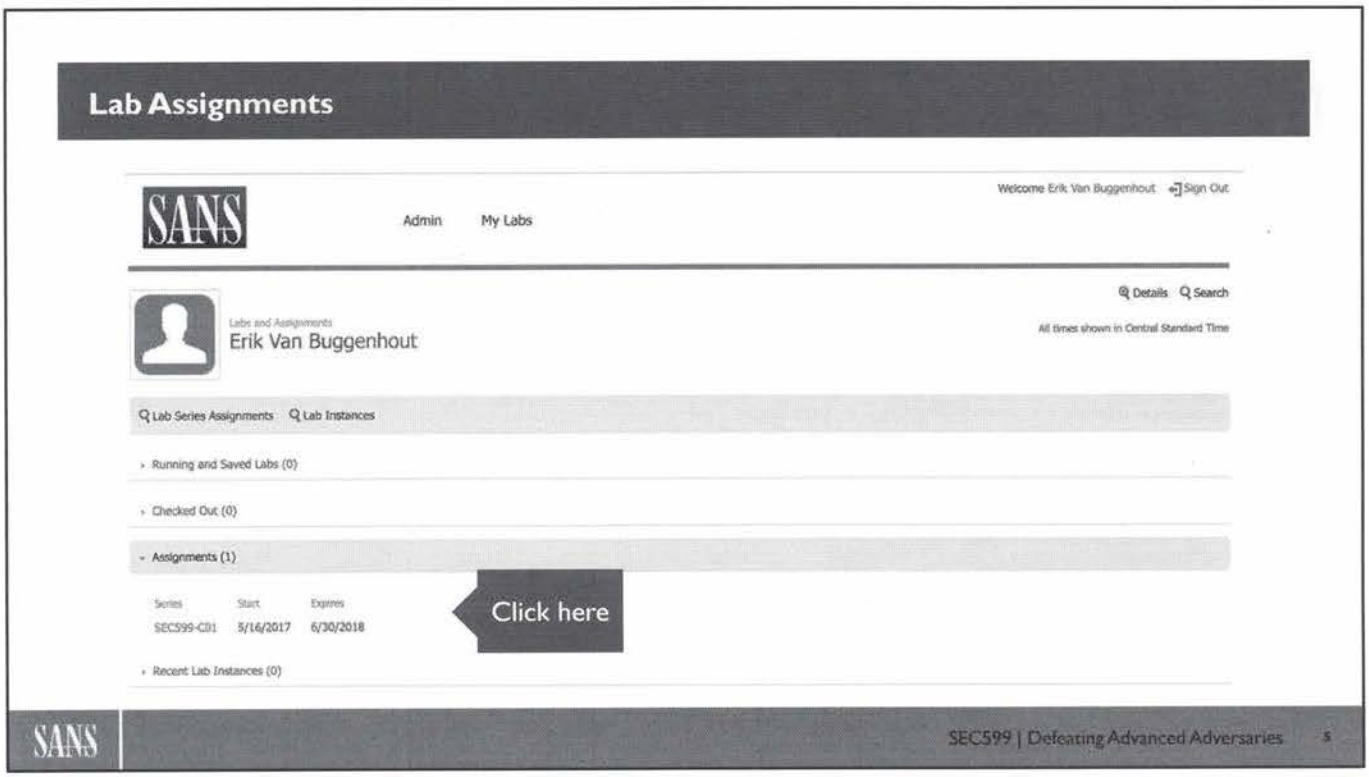
Fundamentally, the SANS Integrated Lab Platform is a way for SANS to deliver a consistent lab experience that focuses on helping you build your skills while minimizing system setup needs.



Getting Started

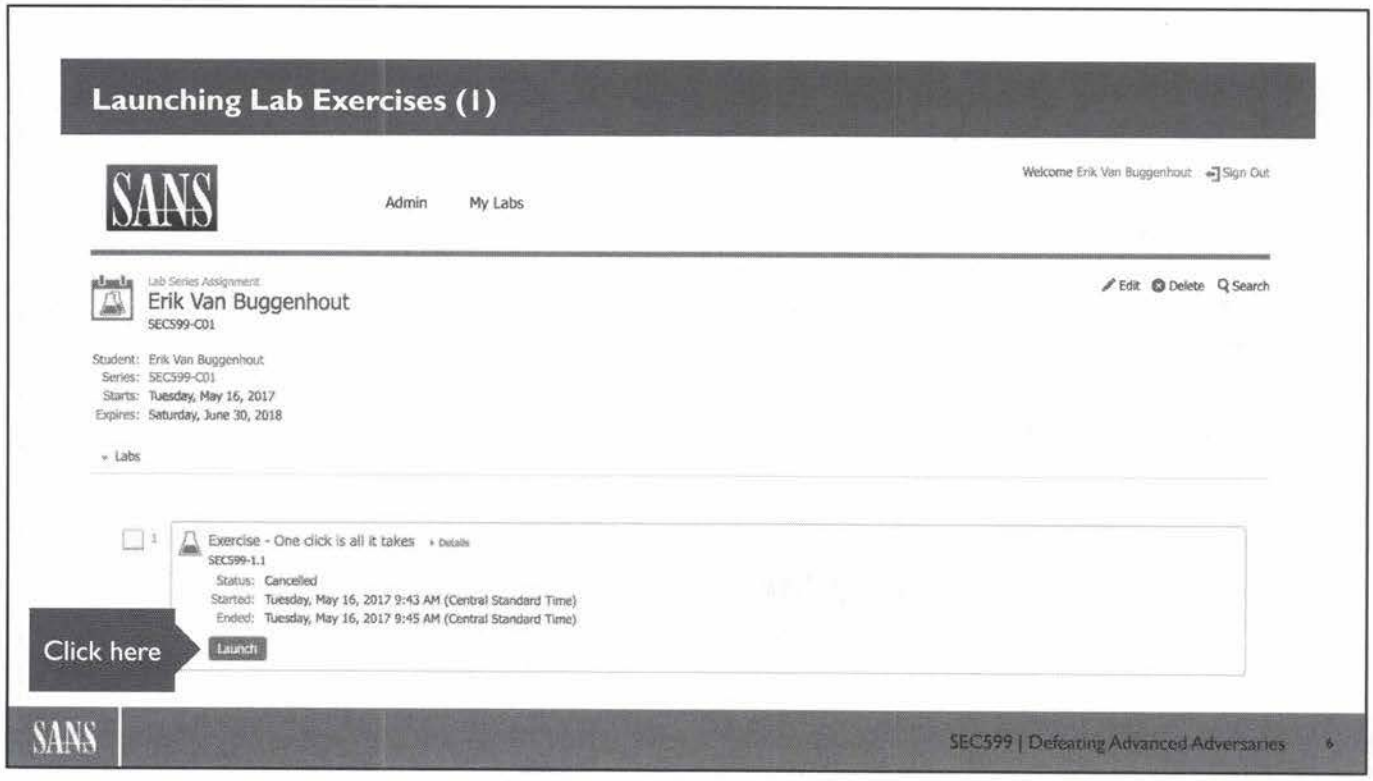
In the beginning of class, your instructor will hand out a login card with your username and password information needed to access the lab server. Please keep this card handy, as you'll use it each day for labs.

Simply browse to the URL on this page (and printed on the login card). When prompted, enter your username and password information, then click Sign In.



Lab Assignments

When you log in to the system, you will see the "My Labs" page. In the Assignments group, you will see your course assignment. Click the course assignment link to see the exercises.

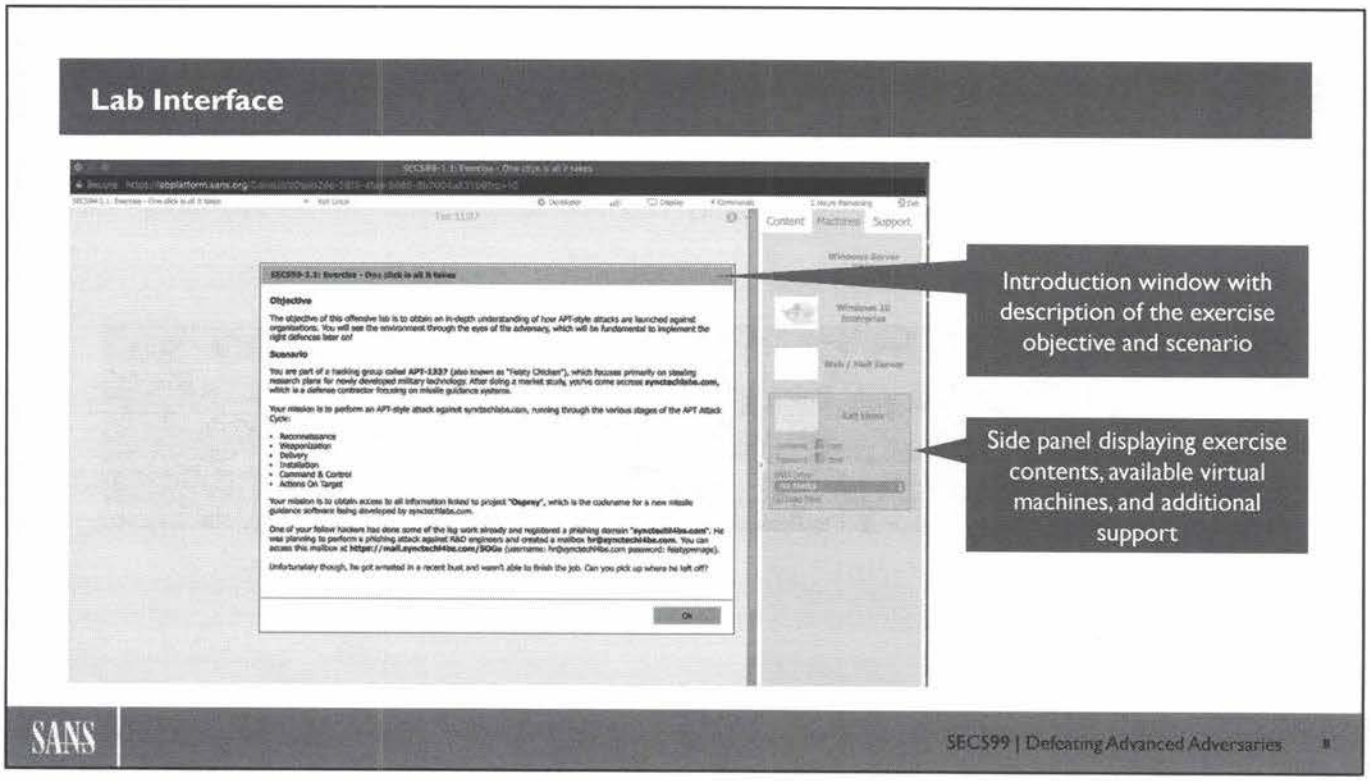


Launching Lab Exercises (1)

After clicking on your lab assignment, you will see a list of all the exercises in the lab assignment. Click the Launch button to start the desired exercises. The exercise will open a new window and kick off the virtual machines needed for the exercise automatically.

Launching Lab Exercises (2)

This page intentionally left blank.



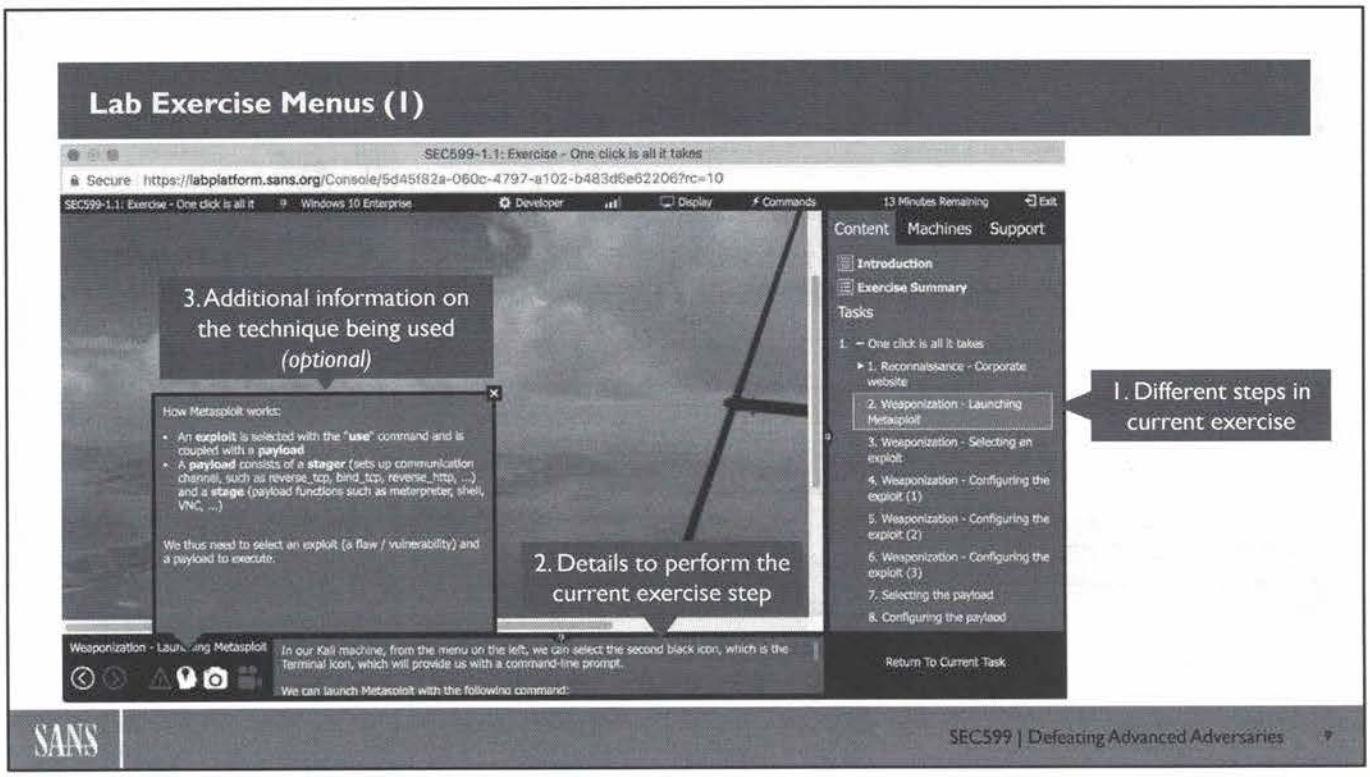
Lab Interface

The window that opens when you click the Launch button will provide the interface for access one or more virtual machines, and the step-by-step directions for the exercise.

First, you will see the objective and scenario information for the lab. Read this material, then click the OK button. Next, you will see an introduction to this specific exercise (a lab can have more than one exercise). Read through this material, then click Next***.

On the right-hand side of the overall display, you have three different titles in the menu: "Content" – "Machines" – "Support"

- Content: A step-by-step overview of the activities to be concluded in this exercise
- Machines: A listing of the different machines available to you in this exercise
- Support: Support information on the SANS Integrated Lab environment



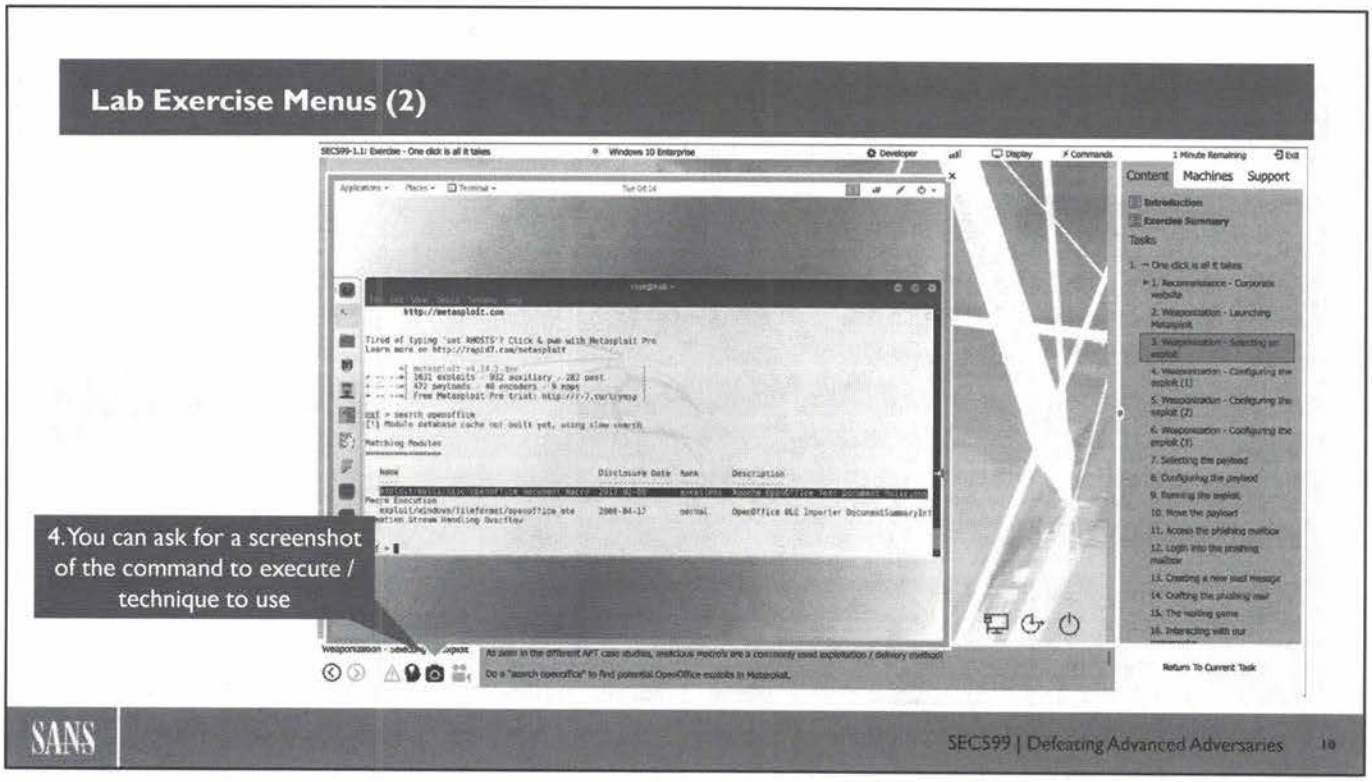
Lab Exercise Menus (1)

Now you are ready to begin the exercise. Let's look at a few of the elements shown on this page. On the right side is a list of the step-by-step directions for the exercise. You can click to jump ahead and explore any of the steps as desired.

In the bottom of the window are the detailed directions for the selected step. As you change to the next step, the detailed instructions will update as well. These detailed instructions tell you what to do to complete the selected step.

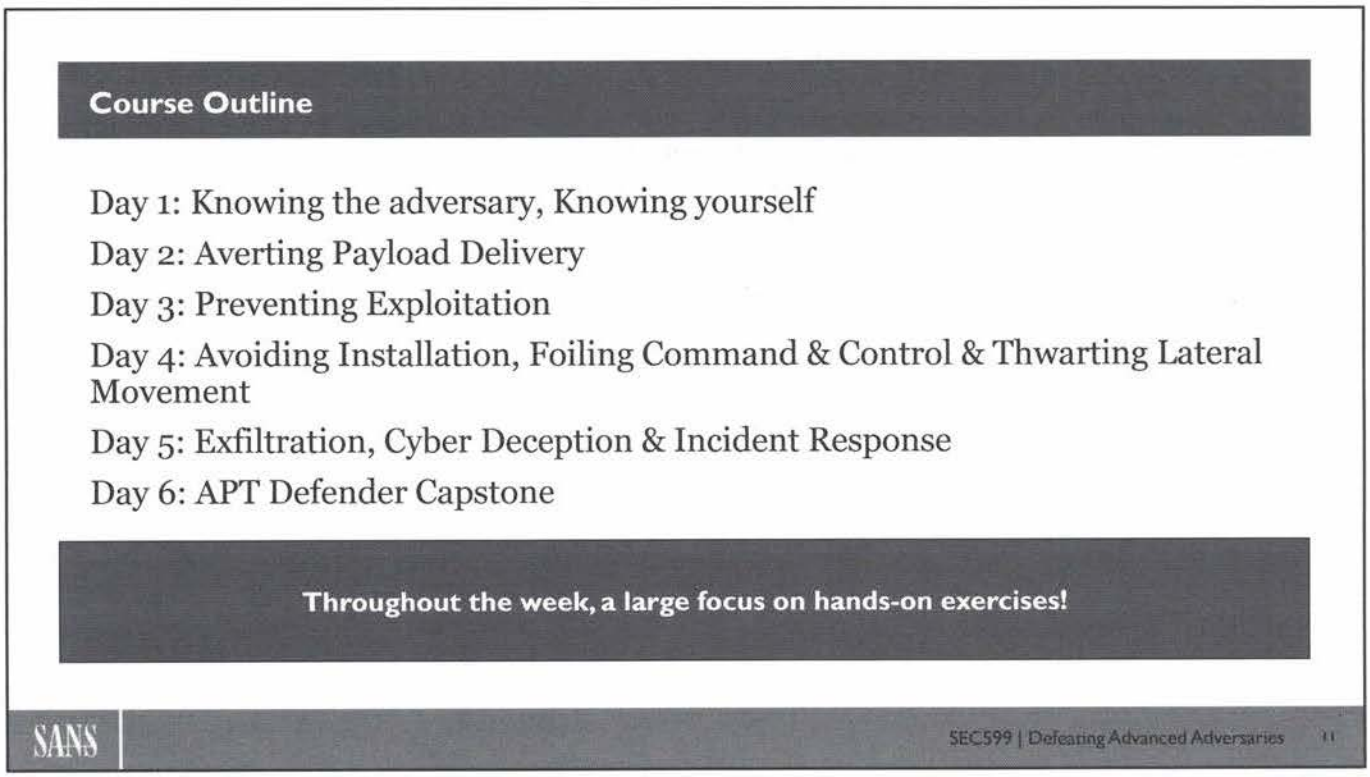
When you complete the instructions in the selected step, you can click the Done button to mark the step as completed. The progress bar in the lower-right corner of the window will show you how many steps are completed, and how many remain.

The main portion of the browser window is your access to the virtual machine that you'll use for this exercise. You can click on this portion of the window and interact with the system like you would for your local system.



Lab Exercise Menus (2)
Finally, the bottom menu also has a "screenshot" button, which you can click to obtain a detailed screenshot of what task is expected of you.

All in all, the LODS platform was set up to be a highly intuitive platform that can help you complete labs without any prerequisite issues. Should you have any further questions or remarks as we go through the different exercises, please don't hesitate to get in touch with your Instructor / TA.



Course Outline
SEC599 has six days of content:

Day 1: Knowing the adversary, Knowing yourself
In Day 1, we will explain what the current threat and attack landscape looks like. We will explain what techniques are being used by our adversaries, so we can prepare ourselves to prevent, detect and respond to them. We will also zoom in on the importance of knowing one's own environment.

Day 2: Averting Payload Delivery
Day 2 will cover how the attacker takes his first steps: how does he perform reconnaissance and what can we do to hinder it? The courseware will cover technical controls, but will also touch upon "soft topics" such as security awareness. After reconnaissance is performed and vulnerabilities are spotted, the adversary will weaponize the payload and deliver it to the target. We will analyze how delivery of the payload can be detected and blocked. We will cover a variety of techniques, including mail-based controls (e.g. SMTP file & URL carving, sandboxing…) and web-based controls (access controls using web proxies).

Day 3: Preventing Exploitation
Day 3 will explain how exploitation can be prevented. Attendees will obtain an in-depth understanding of current exploitation tactics. We will introduce effective security controls to stop exploitation attempts dead in their tracks.

Day 4: Avoiding Installation, Foiling Command & Control & Thwarting Lateral Movement
On day 4, we will zoom in on persistence techniques typically employed by adversaries and how command & control is established. Should the adversary successfully exploit a vulnerability, the next step for them is to attempt persisting their access, escalate privileges and setting up a command & control channel. Once the channel is complete, the adversary can start performing lateral movement, where they pivot throughout the environment looking to accomplish their objectives (e.g. steal sensitive data).

Day 5: Exfiltration, Cyber Deception & Incident Response
Day 5 focuses on stopping the adversary during the final stages of the attack:
-How can data exfiltration be detected and stopped?
-How can cyber deception be used to slow down adversaries?
-How can threat intelligence aid defenders in the APT Attack Cycle?
-How can defenders perform effective incident response?

Day 6: APT Defender Capstone
Day 6 concludes with a hands-on Capstone challenge, applying all the skills you've learned in a friendly, competitive, environment!

SANS

Knowing the Adversary, Knowing Yourself

This page intentionally left blank.

TABLE OF CONTENTS

This page intentionally left blank.

Course Roadmap

- **Day 1: Knowing the adversary, knowing yourself**
- Day 2: Averting Payload Delivery
- Day 3: Preventing Exploitation
- Day 4: Avoiding Installation, foiling Command & Control & thwarting lateral movement
- Day 5: Exfiltration, Cyber Deception & Incident Response
- Day 6: APT Defender Capstone

SEC599.1

Course Outline & Lab Setup
- Course Overview & Objectives
- Attendee System Setup

Current Threat / Attack Landscape
- Key Terminology
- What is happening out there?

Introducing the APT Attack Cycle
- Recent Case Studies – In-Depth
- Exercise: Analyzing The Behavior of Famous Malware
- Exercise: One Click Is All It Takes…

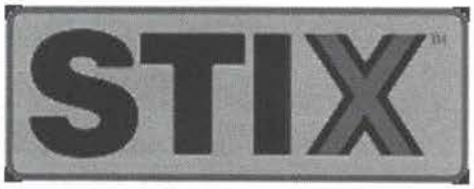
A Defensible Architecture & Environment

Preparation - Knowing Yourself
- Understanding Your Own Environment
- Determining What is "Normal"
- Understanding & Limiting Your Internet Footprint
- A Word on Vulnerability Management

This page intentionally left blank.

Terminology (1)

It is important to speak a common language when discussing cyber security threats. A good framework that serves this purpose is **STIX™**



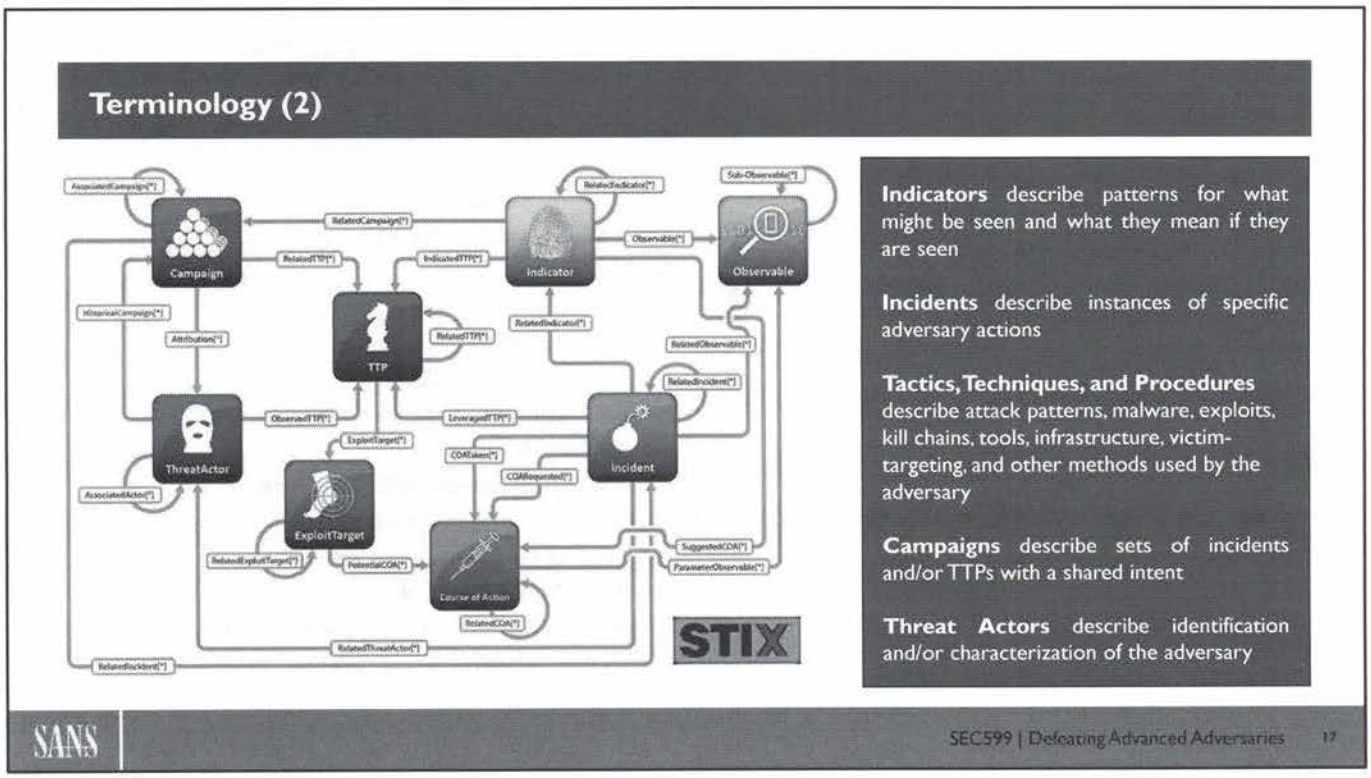
Structured Threat Information Expression (STIX™) is a structured language for describing cyber threat information so it can be shared, stored, and analyzed in a consistent manner

Terminology (1)
When discussing cyber security threats, it's important to speak a common language. Throughout the course, we will base our terminology strongly on the Structured Threat Information eXpression (STIX™). STIX™ was started by MITRE, but is now being maintained by the OASIS Cyber Threat Intelligence (CTI) Technical Committee.

More information on STIX™ can be found here:
- http://stixproject.github.io/about/
- https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cti

STIX™ License
STIX™ is released under a permissive license for any commercial or non-commercial purpose while helper scripts and related tools have individual licenses that typically follow Berkeley Software Distribution.

The MITRE Corporation (MITRE) hereby grants SANS a non-exclusive, royalty-free license to use Structured Threat Information eXpression (STIX™) for research, development, and commercial purposes.



Terminology (2)
The diagram above is a visualization of how the different terms are interpreted in STIX:

- Observables describe what has been or might be seen in cyber
- Indicators describe patterns for what might be seen and what they mean if they are
- Incidents describe instances of specific adversary actions
- Adversary Tactics, Techniques, and Procedures describe attack patterns, malware, exploits, kill chains, tools, infrastructure, victim-targeting, and other methods used by the adversary
- Exploit Targets describe vulnerabilities, weaknesses, or configurations that might be exploited
- Courses of Action describe response actions that may be taken in response to an attack or as a preventative measure
- Campaigns describe sets of incidents and/or TTPs with a shared intent
- Threat Actors describe identification and/or characterization of the adversary
- Reports collect related STIX content and give them shared context

Course Roadmap

- **Day 1: Knowing the adversary, knowing yourself**
- Day 2: Averting Payload Delivery
- Day 3: Preventing Exploitation
- Day 4: Avoiding Installation, foiling Command & Control & thwarting lateral movement
- Day 5: Exfiltration, Cyber Deception & Incident Response
- Day 6: APT Defender Capstone

SEC599.1

Course Outline & Lab Setup
- Course Overview & Objectives
- Attendee System Setup

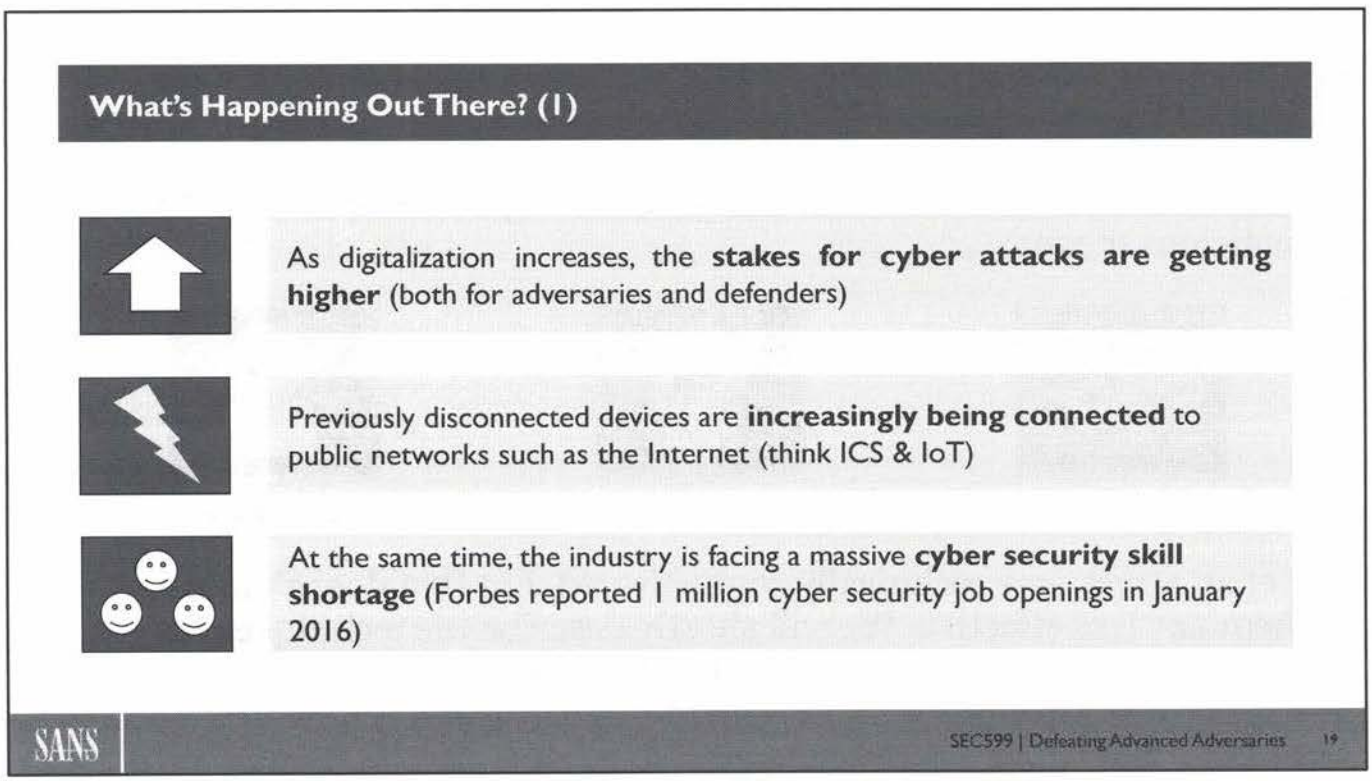
Current Threat / Attack Landscape
- Key Terminology
- What is happening out there?

Introducing the APT Attack Cycle
- Recent Case Studies – In-Depth
- Exercise: Analyzing The Behavior of Famous Malware
- Exercise: One Click Is All It Takes…

A Defensible Architecture & Environment

Preparation - Knowing Yourself
- Understanding Your Own Environment
- Determining What is "Normal"
- Understanding & Limiting Your Internet Footprint
- A Word on Vulnerability Management

This page intentionally left blank.



What's Happening Out There? (1)
As technology plays an increasingly large role in organizations, the stakes for cyber attacks are getting higher. As organizations become more and more dependent on the correct working of their IT systems, defenders need to ensure:

- Mission-critical business systems (& the data stored on them) stay available;
- Data handled by these systems is correct (unaltered);
- Sensitive data handled by these systems is not leaked.

Consequently, the stakes for adversaries are becoming increasingly higher.

Devices that were designed to function in an unconnected fashion are increasingly connected to public networks such as the Internet. This is apparent in the Industrial world, as ICS (Industrial Control Systems) such as SCADA (Supervisory Control And Data Acquisition) devices are being remotely managed and monitored. This trend is however not only in power plants and large industrial complexes: right in our own homes, more and more devices are being connected to the Internet (Internet of Things).

At the same time, organizations are struggling to identify good cyber security experts. In January 2016, Forbes estimated 1 million cyber security job openings globally. A number that is not likely to decrease over the next few years, resulting in a "risky cocktail"!

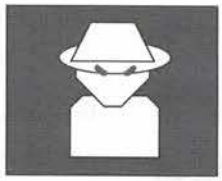
Reference: https://www.forbes.com/sites/stevemorgan/2016/01/02/one-million-cybersecurity-job-openings-in-2016/#7fb603f27ea2

What's Happening Out There? (2)

As a result, more and more bad guys are using "cyber space" for their malicious means:

Cyber Crime

Sabotage



Espionage

Not all attacks are technically sophisticated, but that doesn't make them any less effective. We will shortly describe the techniques they use, with a key focus on **advanced adversaries**

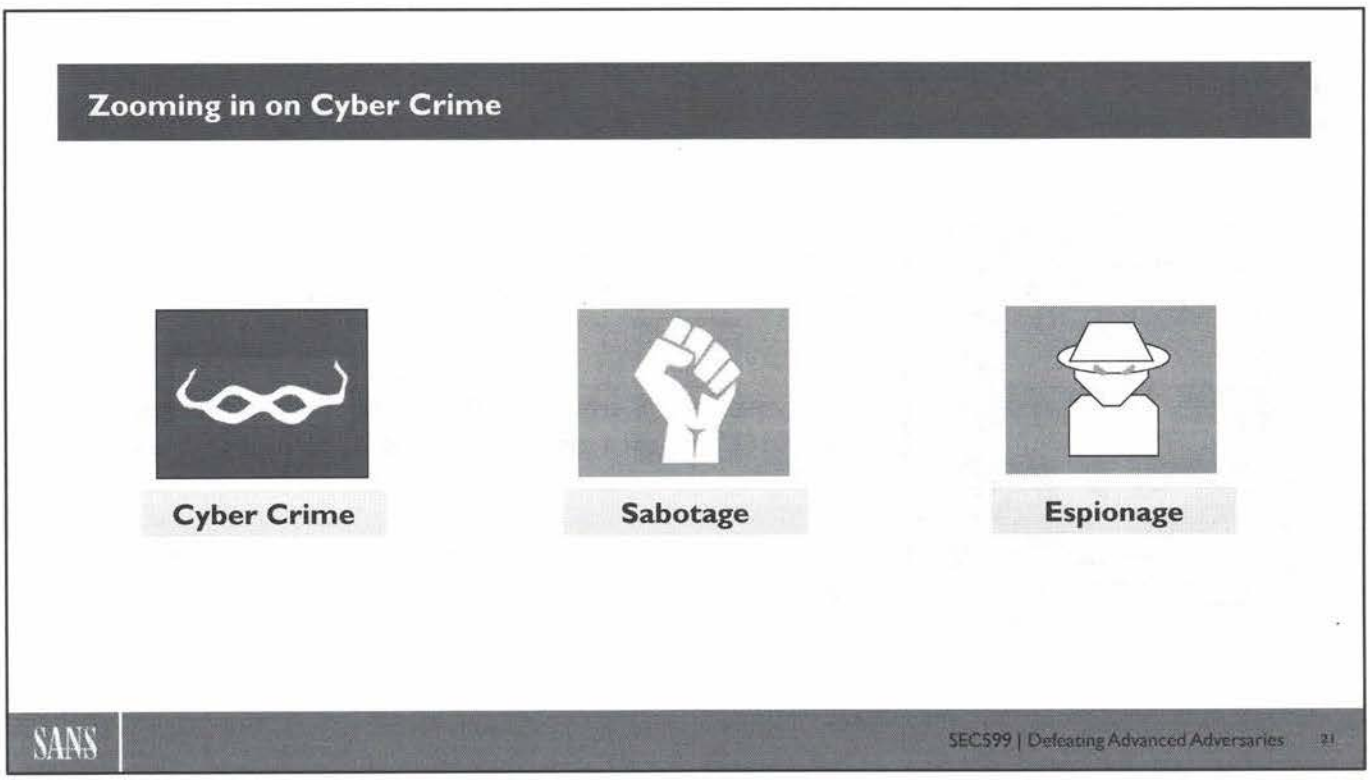
What's Happening Out There? (2)

Over the past few years, more and more organizations have fallen victim to a variety of cyber attacks. Current trends include:

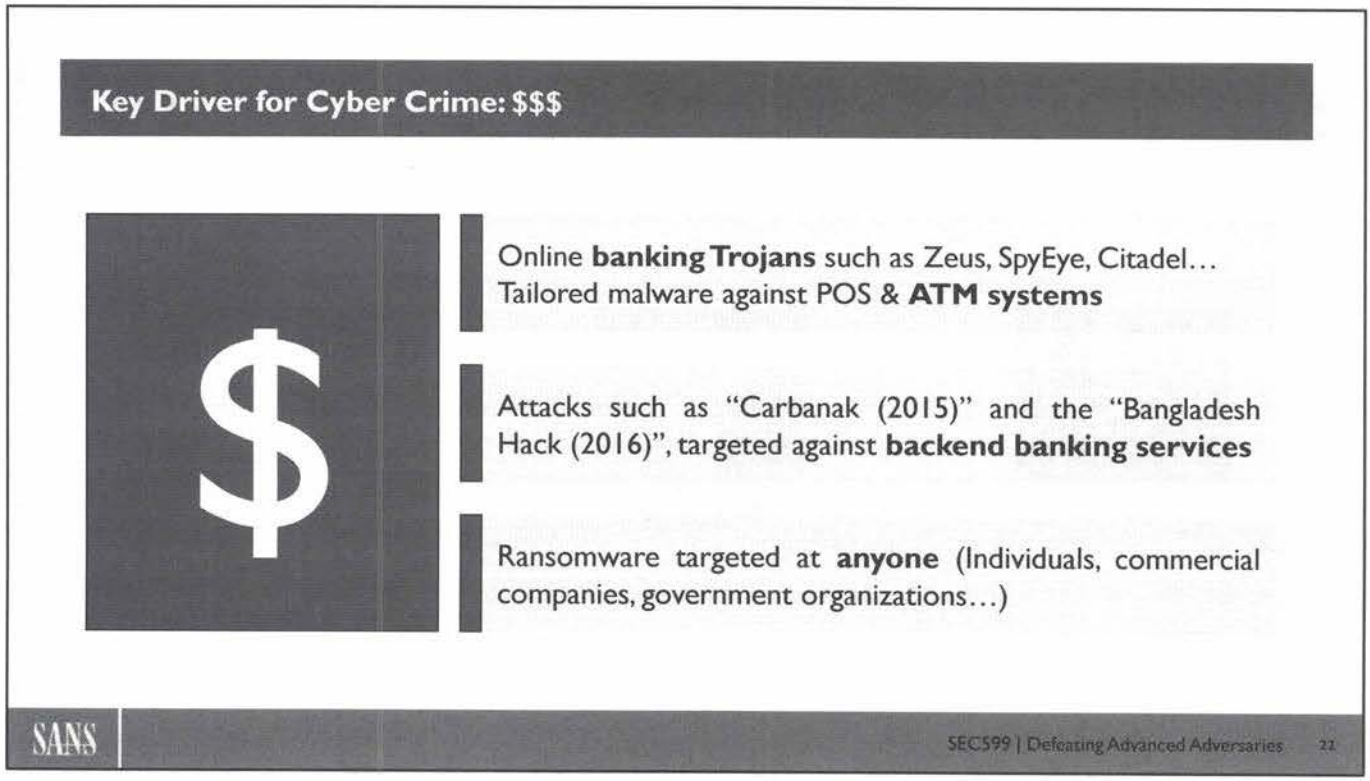
- *Cybercrime*: Attacks focused on earning money / generating revenue for a malicious (group of) perpetrators. Some of the most common attack methods we see these days are ransomware (both against organizations and individuals) and denial of service attacks. With ransomware, business-critical data is encrypted, after which a ransom is asked to allow the data to be recovered. With denial of service, a typical attack technique would be to disrupt the online presence of an organization, after which a ransom is asked to stop the denial of service attack.

- *Sabotage:* Attempts to disrupt your (online) operations. Sabotage is typically executed by hacktivists (e.g. politically motivated attacks against organizations that have different ideological views) or nation-states (e.g. sabotaging the critical infrastructure of other countries in times of war).

- *Espionage:* Could typically include both industrial & political espionage. State-sponsored attacks are not uncommon, as evidenced by a wide variety of discovered APT campaigns. The goal is often to steal data that could result in a commercial advantage (e.g. stealing R&D plans or strategy documents). Next to industrial espionage, political espionage can be focused on obtaining access to sensitive diplomatic intelligence or military technology. It is a "public secret" that most nations are developing offensive cyber capabilities and are using them to their benefit.



Zooming in on Cyber Crime
First, let's have a look at Cyber Crime!



Key Driver for Cyber Crime: $$$
Monetary gain is THE key driver for cyber crime. This makes the attacks somewhat predictable: they are coming for the money… Additionally, it makes the adversaries les persistent: cyber criminal adversaries are looking for the path of least resistance: they will go where the money is easiest to obtain / steal. In order to fend off these adversaries, an age-old safari axiom can be of interest:

'You don't have to be the fastest, just don't be the slowest'

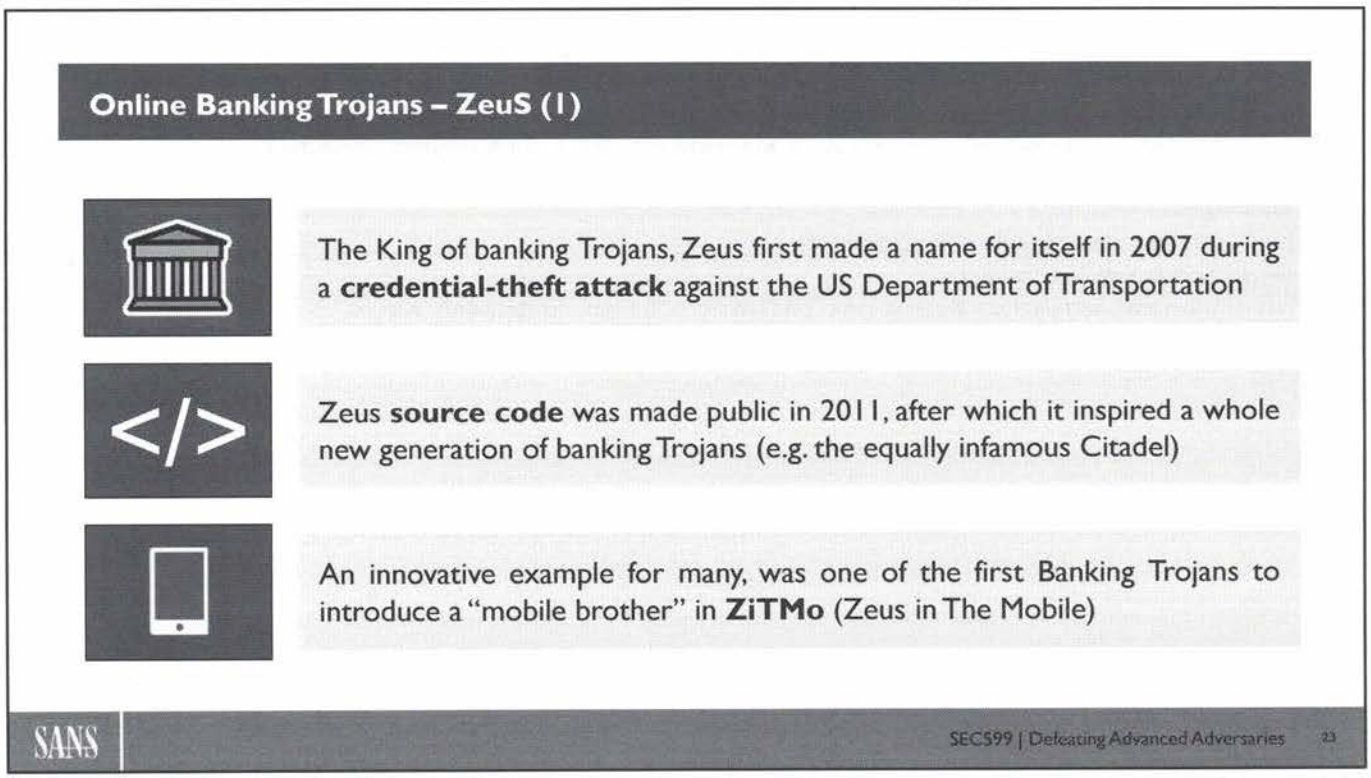
Some interesting attack techniques we've seen over the past couple of years:

- Online banking Trojans such as Zeus, Citadel & Dridex that attempt to infect online / mobile banking users. The idea here is to infect as many users as possible and transfer relatively small amounts per infection. Furthermore, tailored malware is being written that attacks POS (Point of Sales) & ATM systems.

- Somewhat more advanced attacks against banks themselves, where they attempt to infect business users involved in the creation, signing, and approval of larger fund transfers. Key examples of this include the Carbanak attack revealed in 2015 and the "Bangladesh Bank Heist" that occurred in 2016. In both cases, adversaries had obtained a foothold in the internal bank networks and were monitoring the environment to understand how fund transfer approval flows worked and how large fraudulent transactions could be executed.

- Finally, since 2015, we see a very strong rise in the use of ransomware, that is targeted against anyone with data. Ranging from individuals, commercial companies to top-secret government organizations: if you are willing to pay to retrieve your data, you are a target.

References
-https://securelist.com/blog/research/68732/the-great-bank-robbery-the-carbanak-apt/



Online Banking Trojans – ZeuS (1)
In order to describe how Banking Trojans work, we will zoom in on the King of banking Trojans, namely Zeus (aka Zbot).

Zeus is a versatile Trojan horse malware, capable to perform many malicious acts. It is often used as man-in-the-browser. Man-in-the-browser is a word play on man-in-the-middle: in a man-in-the-browser attack, malicious code is injected into the Internet browser process to steal information and to tamper with the rendering of web pages, for example by adding forms for phishing purposes. Man-in-the-browser malware can be written as a browser plugin, or as stand-alone code directly injected into the browser process.

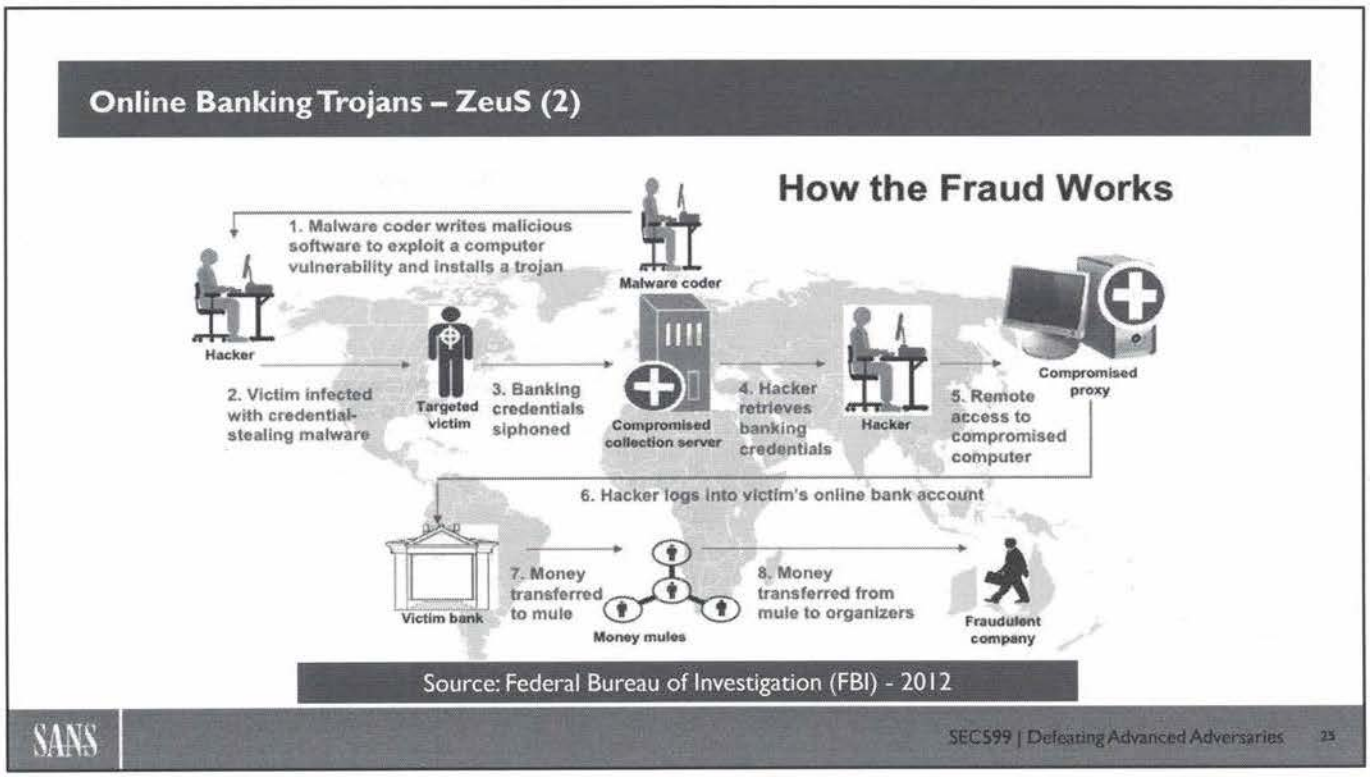
When used in man-in-the-browser mode, Zeus can perform keylogging and form grabbing (stealing data entered into forms). Zeus is also used to spread ransomware (CryptoLocker). It first became known in 2007, when it was used (and detected) in a credential-theft attack against the US Department of Transportation, although its widespread use began in 2009. Another activity of the Zeus authors was the facilitation of tech support scams. When running on a Windows machine, this variant of Zeus would display a pop-up message alerting the user to the simulated presence of a computer virus. The message would instruct users to call a phone number (often claiming to be Microsoft support), where scam artists would "help" users to check for errors with the Windows event viewer (there are always error events in the viewer), claim that this was caused by a virus and get the victim to pay for a fake anti-virus solution.

In 2011, the Zeus source code (a Microsoft Visual Studio project) was made public, spawning many new banking Trojans. It is always easier to copy something than to start from fresh. Initially, the Zeus source code was reused by criminals with minor modifications, requiring little skill: just be able to compile in Visual Studio and replace some strings like IP addresses and domain names. Later on, substantial changes were made resulting in completely new banking Trojans like Citadel.

Zeus-in-the-mobile (ZitMo), appeared in late September 2011. Working together with Zeus on Windows, ZitMo intercepts and steals mobile transaction authentication numbers (mTAN codes) send to the banking

clients' mobile phone. When an online banking transaction is initiated, an mTAN code can be used to authenticate the transaction: the banking software backend sends an mTAN code via mobile messaging to the client, who has to enter it in the online banking application on Windows to validate the transaction. ZitMo steals the mTAN code and passes it on to Zeus on Windows to use it in a fraudulent transaction.

References:
https://en.wikipedia.org/wiki/Zeus_(malware)
https://github.com/Visgean/Zeus
https://securelist.com/analysis/publications/36424/zeus-in-the-mobile-facts-and-theories/

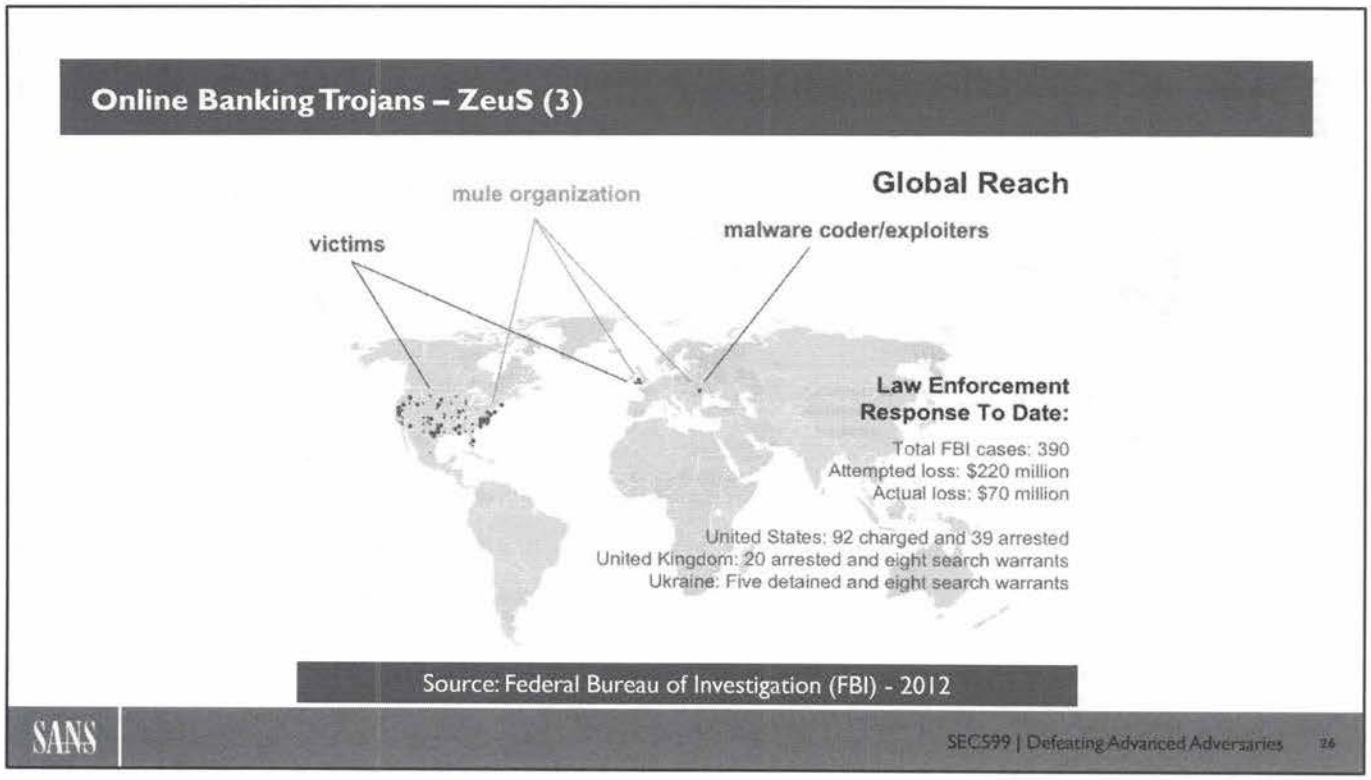


Online Banking Trojans – ZeuS (2)

We will describe here a typical online banking fraud. The different steps are explained in the FBI diagram above.

Via drive-by downloads using exploits or spammed malicious links, the online banking Trojan is installed on a victim's machine. It should be pointed out that online banking Trojans can operate without administrator rights on Windows, there is often no need for a privilege escalation exploit or an administrative user. When the victim uses its infected computer to authenticate to the bank and conduct financial transactions, the Trojan will steal the victims' banking credentials. The credentials harvested on many infected machines are uploaded to a credential collection server, often a compromised web server.

Criminals will regularly connect to the collection server to retrieve new banking credentials. With these credentials, the criminals will initiate fraudulent banking transactions while hiding behind compromised computers and proxies. The fraudulent banking transactions transfer money from the victims' banking accounts to banking accounts of money mules, which will then transfer the money to the criminals.



Online Banking Trojans – ZeuS (3)
Money mules are people recruited by the organization behind the online banking fraud, often under false pretenses.

In online banking fraud, criminals need to obtain the money of their victims, while trying to stay out of reach of law enforcement. Rather sooner than later, victims will notice that money disappeared from their banking accounts, and alert their bank and law enforcement. As financial transactions always leave a trace (account numbers), the transactions that were used to siphon money out of the victims account can be traced back. And if money is directly transferred from the victim accounts to the criminal accounts, the criminals will get caught soon by law enforcement.

To make tracing back financial transactions harder, money is transferred via intermediary accounts. These intermediary accounts are opened by money mules, often in the same country as the victims. Criminals perform fraudulent transactions destined to the intermediary accounts and then instruct the money mules to transfer the money on their accounts. Money mules are recruited by criminals, often under false pretense of a work-at-home scheme or get-rich-quick scheme. For example, the money mules are told that they get an accounting job, where they have to receive payments and then transfer the money at fixed times to their employers (the criminals). In lieu of a salary, they get to keep a part of the money, for example, a few percent on the total sum. This goes on until the money mules get caught by law enforcement, but by then the criminals have already recruited new money mules.

To further increase the difficulty of tracing financial transactions, financial transactions are made between different countries, requiring the involvement and cooperation of law enforcement of different countries; also by using specialized money transfer services like Western Union, that allow money transfers without bank accounts. The money mule takes cash out of his account, hands if over to the specialized service with instructions to deliver it to the criminals. To receive the funds in a Western Union office in their country, the criminals just need a secret transaction number given to them by the money mule, together with ID. In some countries, IDs are not required, or fake IDs are easily obtained. This makes it very hard to trace back the chain of money.

Online Banking Trojans – Defeating Two-Factor Authentication

"FakeToken" mimics the interface of the bank used by the victim

Note that the actual "text" is always the same, the only differences are in the style

Key goal is to steal online banking tokens sent "Out of Band"

Recent evolution: Now also encrypts end-user data for ransom

Installs a **backdoor** on the mobile device capable of executing arbitrary commands, adding Command & Control servers, intercepting & sending SMS, downloading & installing packages…

Online Banking Trojans – Defeating Two-Factor Authentication

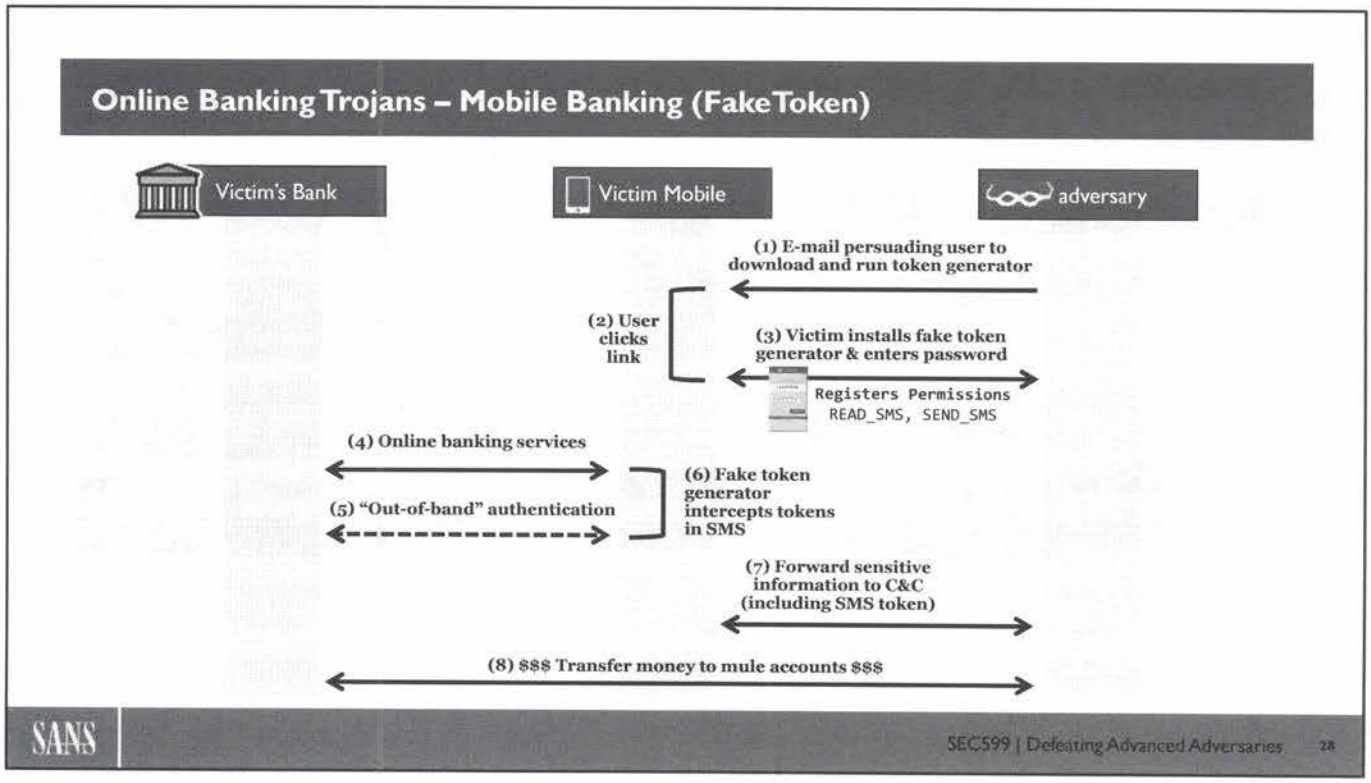
As more and more online banks switched to two-factor authentication for transaction signing, malware authors had to keep up: introducing FakeToken.

FakeToken is malware targeting mobile devices (smartphones), originally intended to steal mTAN codes we discussed before. It mimics the user interface that the genuine mobile banking application would use. In the screenshots above, we see user interfaces for different banks, but taking a close look, we see that this is actually the same application: mToken, Generar…

The malware pretends to generate mTAN codes for the client, but in reality, covertly intercepts mobile messages send to the client's mobile phone with the mTAN code, and then forwards the mTAN code to the criminal, who can use it to conduct a fraudulent financial transaction.

Such fake token malicious mobile applications evolved to diversify criminal activities, like charging money for expensive paying SMS services or party lines or even encrypting the user's data on the phone for a ransom.

Command and control features were later added to achieve this goal.



Online Banking Trojans – Mobile Banking (FakeToken)
We will now provide an overview of the life cycle of a FakeToken infection.

First, the adversaries send emails to potential victims. The email pretends to be from the victim's bank and instructs the victim to install an application needed to increase the security of financial transactions. Often, a form of duress is used by claiming that the victim will lose access to their financial accounts if they do not comply within the allotted time.

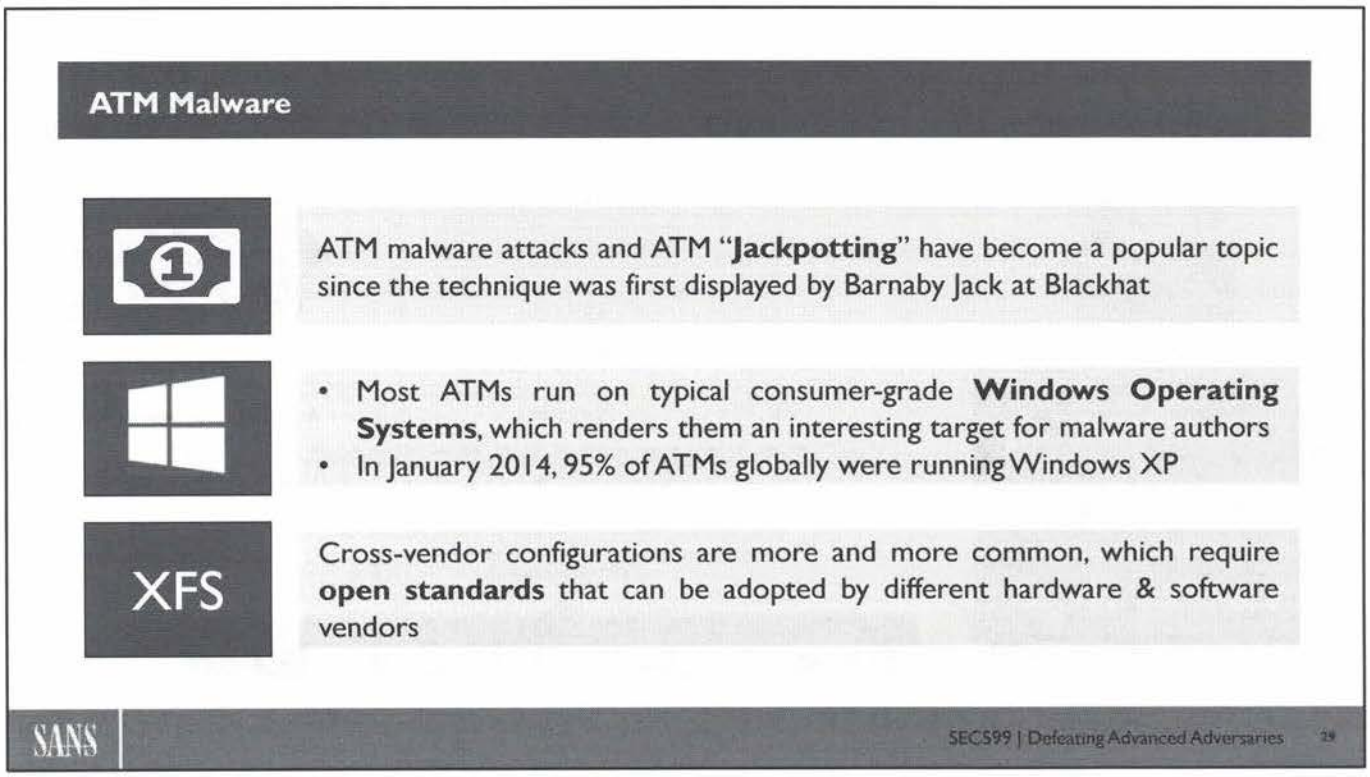
The victim clicks on the link and installs the fake token malware. On Android smartphones, the malware will require permissions to send and receive mobile messages (SMS). After installation, the fake token malware asks the victims to enter their mobile banking credentials.

Later, when the victim conducts mobile financial transactions with the genuine mobile banking application or website, financial transactions have to be authenticated via an mTAN code communicated out-of-band. Out-of-band means that the code is communicated via another communication channel (band) than the one used by the mobile banking application. Here mTAN codes are communicated via SMS.

The fake token malware will intercept the received SMS messages with the mTAN codes, and forward them to the adversary together with the credentials.

As mTAN codes have a limited lifespan, the criminals have to use them quickly to perform fraudulent transactions (siphoning money from the victim's account to a money mule account): this process is therefore automated.

References:
http://securityaffairs.co/wordpress/54563/cyber-crime/faketoken-ransomware-banker.html
https://www.f-secure.com/v-descs/trojan_android_faketoken.shtml
https://threats.kaspersky.com/en/threat/Trojan-Banker.AndroidOS.Faketoken/

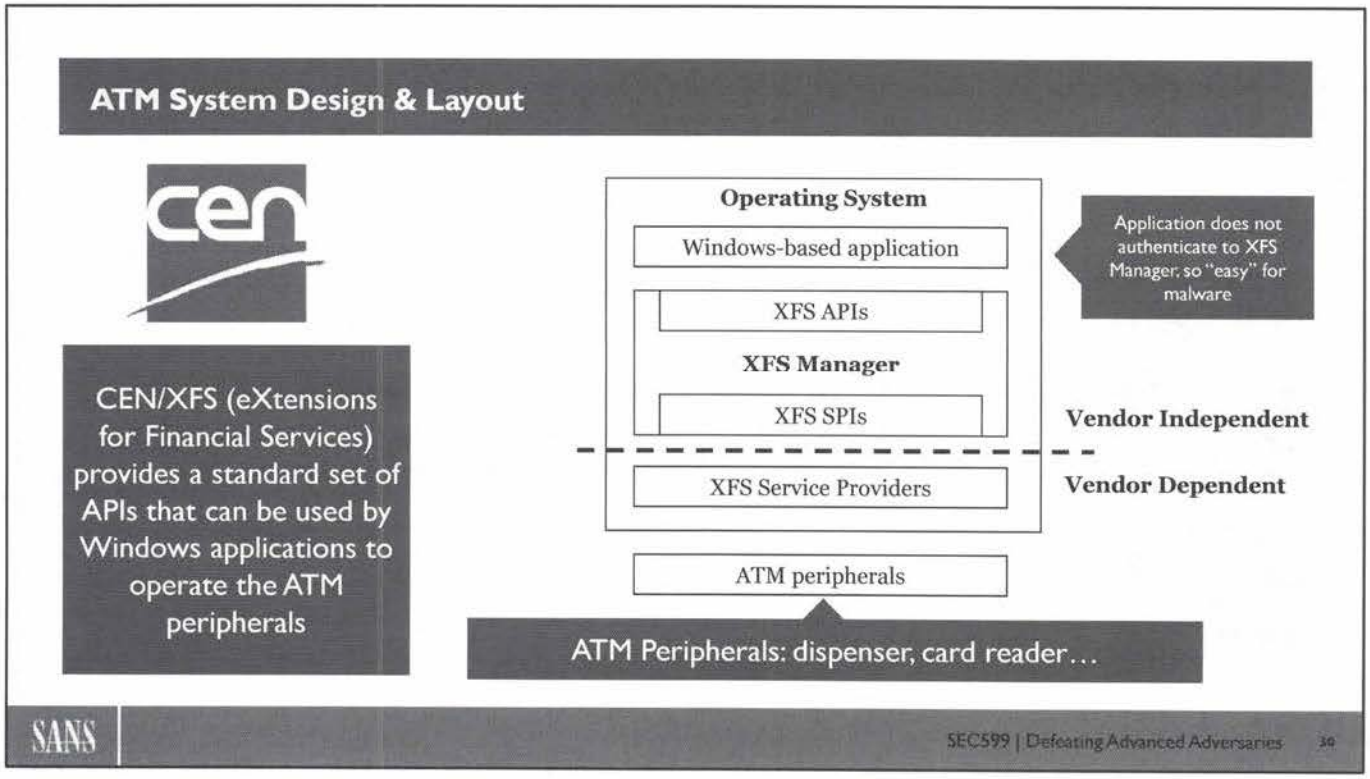


ATM Malware

While online banking has been around for quite some time, since +- 2013 adversaries have found an interesting new strategy to steal funds: directly attack ATM's. Back in 2010, the late Barnaby Jack presented his "ATM Jackpotting" research, where he made several ATMs dispense cash at will on stage.

In order to understand the attack surface of ATMs, it's important we know how they are designed:

- Most ATMs run a typical consumer-grade Windows Operating System. As an illustration, in January 2014, 95% of the world's ATMs were running Windows XP.
- Increased competition in the ATM software and hardware manufacturing industries has led to cross-vendor ATM configurations (where software from Diebold's Agilis software could for example run on an NCR hardware ATM). These cross-vendor configurations require open standards and development frameworks to ensure interoperability.



ATM System Design & Layout

In this slide, we will further zoom in on the "inner workings" of the modern ATM. The CEN/XFS standard provides a standard set of API's that can be used by Windows applications to interact with ATM peripherals (such as the dispenser, the card reader …).

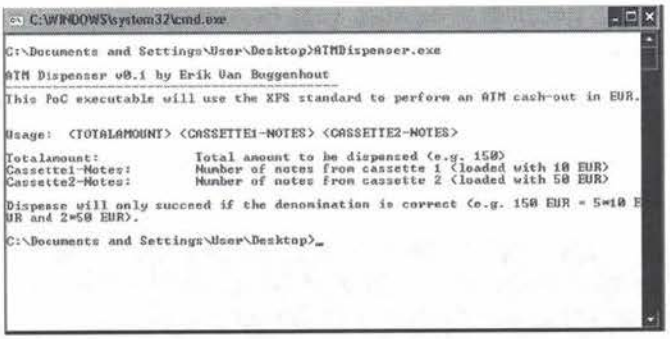
The architecture is design as follows:

- Windows-based applications interact with generic XFS API's (e.g. "Dispense") provided to them by the XFS Manager;
- The XFS Managers uses a set of SPI's (Service Programming Interfaces) that will translate these to interact with the vendor-dependent XFS Service providers;
- Finally, the XFS Service providers interact with the ATM peripherals to perform the actual task.

It's important to note that any Windows-based application running on the Operating System can set up a session with the XFS Manager and thus control the ATM peripherals.

ATM Malware – Sample Source Code

As part of ongoing research, SANS Instructor Erik Van Buggenhout wrote a small Proof of Concept Windows executable to dispense money from ATM:




```
ulaValues[0] = cassette1;   // # notes from
cassette1 is supplied by a command line parameter
ulaValues[1] = cassette2;   // # notes from
cassette2 is supplied by a command line parameter

tDenomination.lpulValues = ulaValues;
tDispense.lpDenomination= &tDenomination;

HRESULT hResult =
WFSExecute(service,WFS_CMD_CDM_DISPENSE,&tDispens
e,WFS_INDEFINITE_WAIT,&lpResult);
```

SANS SEC599 | Defeating Advanced Adversaries 31

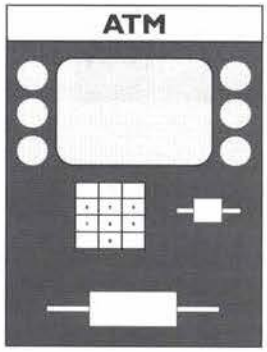
ATM Malware – Sample Source Code

In 2014, SANS Instructor and course author Erik Van Buggenhout spent some time researching ATM malware. By using the publicly referenced CEN/XFS standard (eXtensions for Financial Services), he wrote a Proof-of-Concept Windows application that forced ATMs to perform dispense operations without entering a debit / credit card.

An extract of the source code and the application in action is shown below. It's important to note the trivial nature of the malware development: it's using a standard, public, framework and works natively on one of the most commonly used Operating System families: Windows. The ease of its development meant the rise of ATM malware was thus only a matter of time…

References:
https://www.cen.eu/work/areas/ICT/eBusiness/Pages/WS-XFS.aspx

Typical ATM Malware Attack



Adversaries obtain physical access to the ATM computer (through the back or through the front panel)

Adversaries connect peripheral devices (e.g. keyboard, USB stick…) to deploy & install malware on the ATM

Mules are sent to perform illegitimate dispenses by interacting with the ATM keypad, entering "tokens" they receive from the actual criminals

In 2014, a campaign was observed where adversaries connected a phone in USB tethering mode to the ATM computer, the mules now didn't have to physically interact with the ATM, instead, they used SMS messages to start the dispense

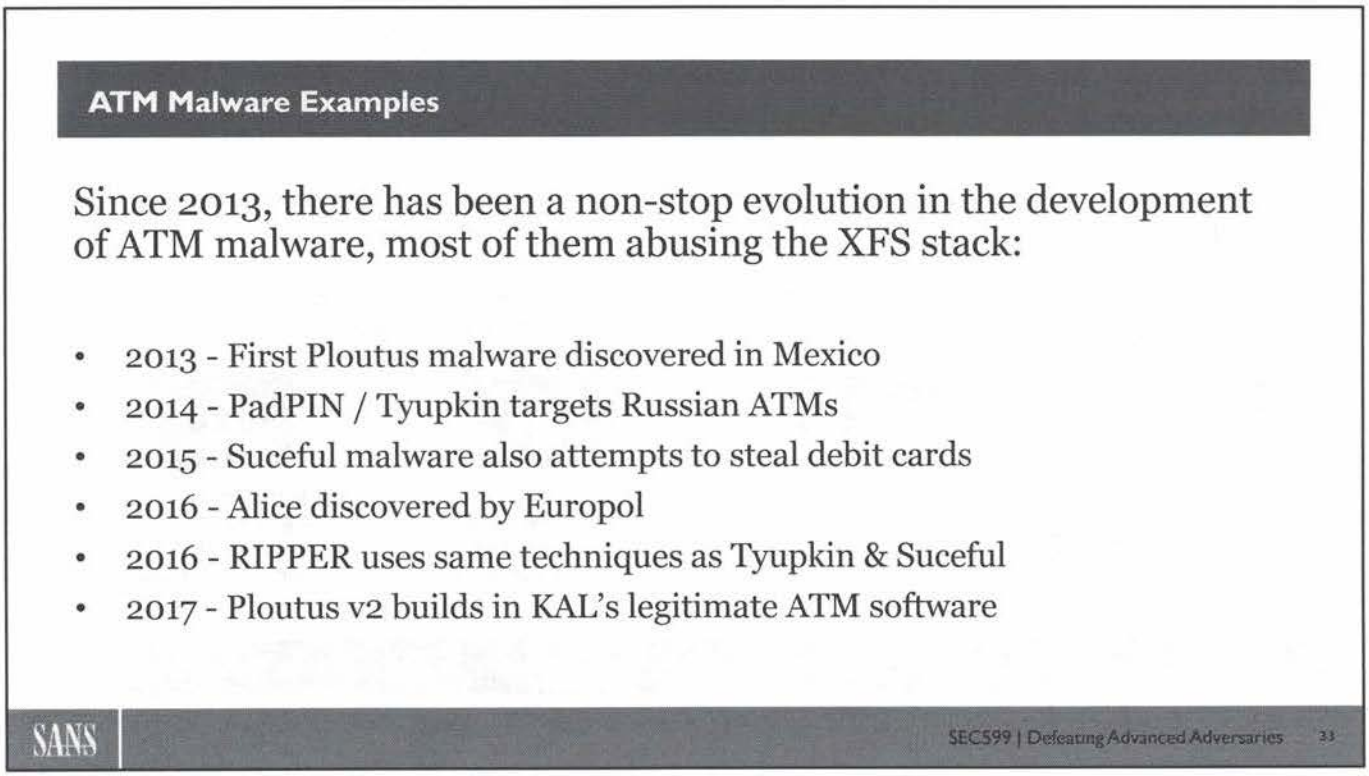
Typical ATM Malware attack

So, we've seen the development of ATM malware is rather straightforward. So how do adversaries infect ATMs with malware?

- Step 1: In the majority of cases, adversaries obtain physical access to the ATM computer. Historically, the security of ATMs was mostly focused on the vault and making sure the vault couldn't be breached easily. The security of the actual computer controlling the ATM was considered less important. There have been cases where the front-end of the ATM was breached (e.g. by drilling a small hole), but adversaries have also managed to gain access to the back of the ATM (e.g. by social engineering employees and getting physical access)

- Step 2: Once physical access is obtained, it is abused to connect peripheral devices (e.g. keyboard, USB stick…) to deploy & install malware on the ATM. One could ask himself why ATMs don't have their USB ports disabled: normal technical support operations require this type of support as well, hence it's "easier" to leave the USB ports enabled

- Step 3: Mules are sent to perform illegitimate dispenses by interacting with the ATM keypad, entering "tokens" they receive from the actual criminals. In 2014, a campaign was observed where adversaries connected a phone in USB tethering mode to the ATM computer, the mules now didn't have to physically interact with the ATM; instead, they used SMS messages to start the dispense



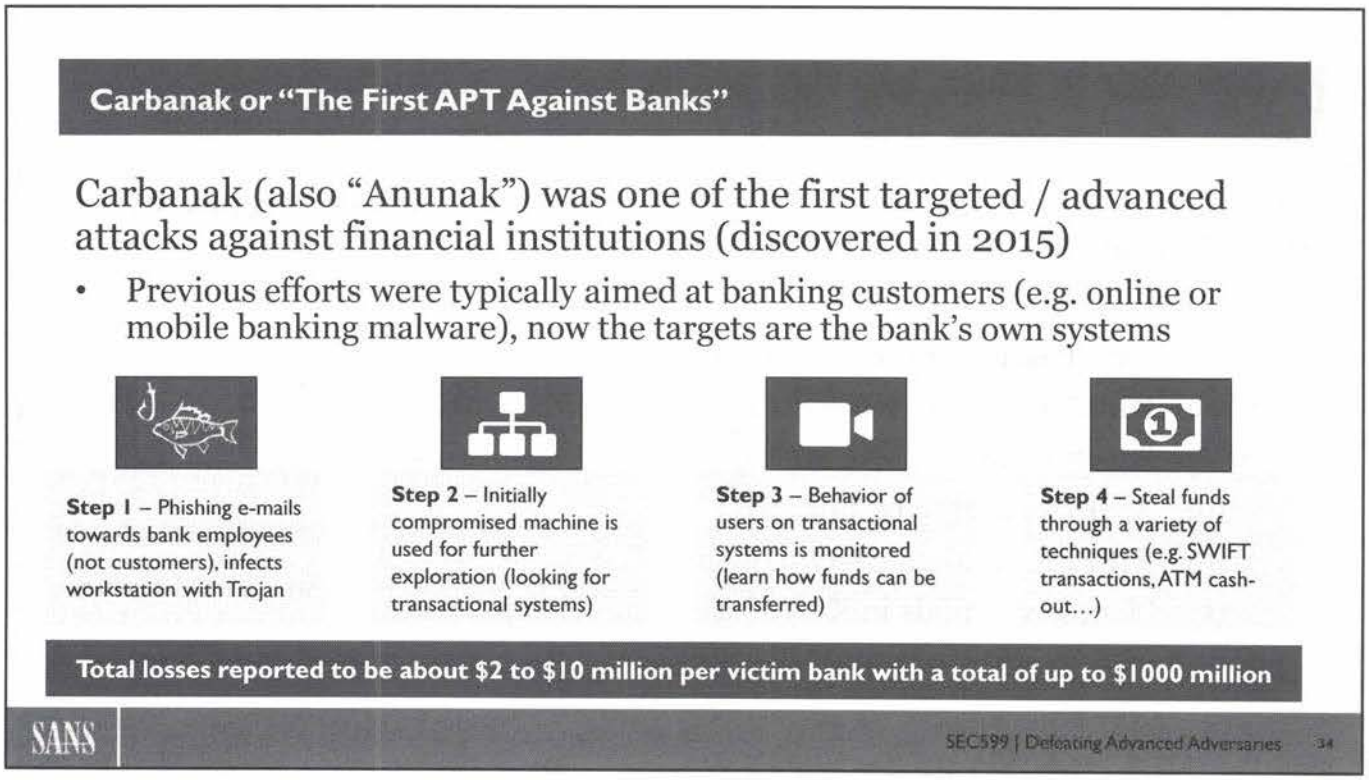
ATM Malware Examples

Since 2013, we have seen a relentless evolution in the development of new malware samples. We list a few interesting examples on this slide, in which we clearly see the evolution of adversary tactics:

- In 2013, Ploutus became one of the very first banking malware samples seen in the wild. It mainly targeted Latin-American banks and was first identified in Mexico. Once ATMs were infected with Ploutus, adversaries could interact using the front PIN pad to enter specific commands to make the ATM dispense cash. This was later further adapted to also support dispensing via SMS (using USB tethered mobile phones);
- In 2015, the Suceful ATM malware not only allowed for cash dispensing, it also supports the stealing of debit cards (it would block the ATM card upon insertion);
- In 2017, the "second generation" Ploutus including the legitimate KAL software stack for increased compatibility.

References:

https://www.fireeye.com/blog/threat-research/2015/09/suceful_next_genera.html
https://www.bleepingcomputer.com/news/security/new-alice-malware-makes-atms-spit-out-cash/
https://www.fireeye.com/blog/threat-research/2016/08/ripper_atm_malwarea.html
https://www.fireeye.com/blog/threat-research/2017/01/new_ploutus_variant.html



Carbanak or "The first APT Against Banks"
Carbanak is the name of an APT attack and associated malware, performed against financial institutions and discovered in 2015 by anti-virus company Kaspersky. The Carbanak gang managed to steal at least 500 million dollars from financial institutions and their clients, through various means. The malware was often delivered via phishing emails.

Via phishing emails with executable attachments like CPLs (Malicious Control Panel items) or Word documents with exploits, a backdoor (Carbanak) is installed on the victim's machine. This malware is based on the Carberp malware. The malware is designed to support the following functions: espionage, data exfiltration, and remote control. Once they gained access to the victim's machine, the criminals used this beachhead in search of computers that could help them perform fraudulent financial transactions, like computers operated by administrators. This lateral movement led them to computers that could perform financial transactions.

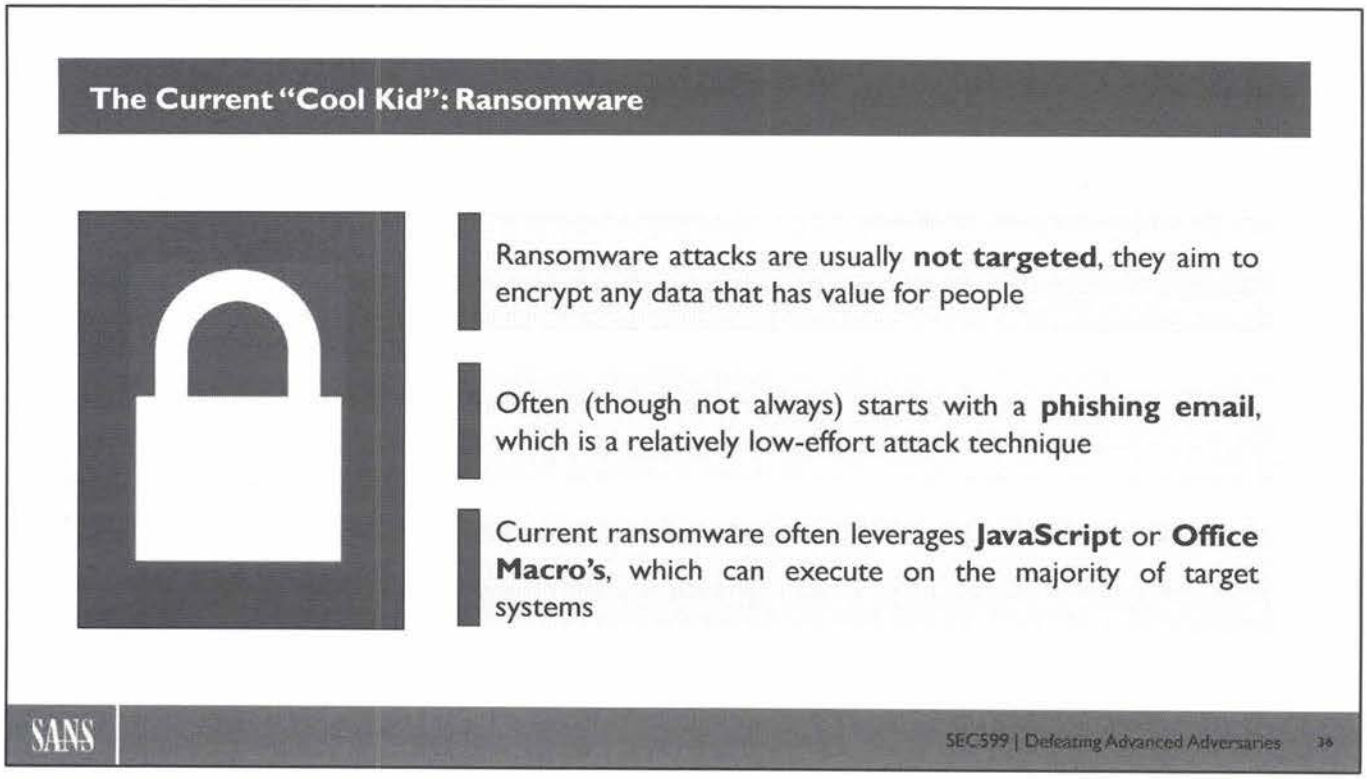
Often the criminals behind the Carbanak gang would not have financial knowledge and procedures of the bank they were targeting, but would quickly learn by recording the screens and keyboard strokes of the compromised machines, and learn via videos how to operate the financial systems. Armed with this knowledge and credentials, they would perform operations to obtain money.

Money would be obtained through different scenarios, depending on the environment they discovered at the bank they targeted. They are known to have:

- Programmed ATMs to cash out money without any interaction;
- Transferred money to mule accounts;
- Used the SWIFT network to inject financial transactions;
- Create fake bank accounts with a high balance

The losses per bank were between 2 and 10 million dollars per bank and could be as high as 1000 million dollars in total.

References:
https://securelist.com/blog/research/68732/the-great-bank-robbery-the-carbanak-apt/
https://en.wikipedia.org/wiki/Carbanak



The Current "Cool Kid": Ransomware

One of the most commonly used attack methods of cyber crime gangs these days is ransomware. Ransomware is a devilishly easy principle: make (preferably critical) data inaccessible and demand a ransom for its release. This is typically achieved using a security technique we usually use to **protect** ourselves: encryption. The key properties of ransomware campaigns are the following:

- It is a not-targeted attack: it is directed at anyone with data, including individuals, commercial companies, governments…
- It typically starts with a phishing email (though not always). A recently identified infection vector is to target insecurely configured databases (MongoDB, MySQL…) that are exposed on the Internet. Instead of stealing or removing the data, it is encrypted and held for ransom.
- It is technically rather trivial: Ransomware is no advanced "Trojan" that needs to support multiple commands, it just has to encrypt data and leave a ransomware note.

We will analyze an example ransomware campaign in the next few slides.

Some Ransomware Families

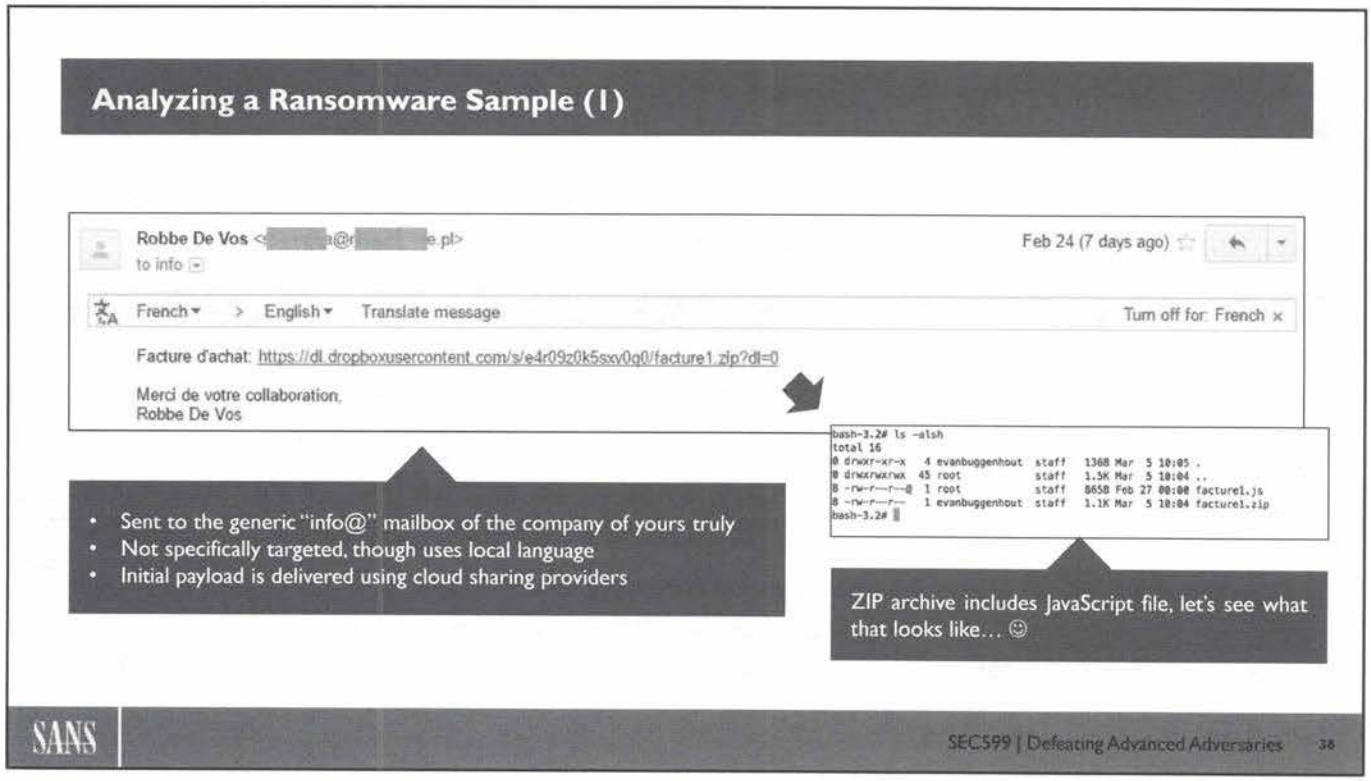
New ransomware families are popping up on a frequent basis, we list some interesting examples with particular behavior below:

Name	First appeared?	Specifics?
Locky	2016	Can leverage different exploit kits, highly flexible
Cerber	2016	Includes non-typical ransomware features like DDoS attacks
Jigsaw	2016	Both steals & encrypts your data, focuses on "victim service"
Crysis / LeChiffre	2015	Uses RDP brute forcing to obtain access to target systems
Goldeneye / Petya / HDDCryptor	2016	If ran with administrative privileges, will encrypt entire drive and overwrite Master Boot Record
Popcorn Time	2016	"Infect-a-friend" in exchange for decryption key
Wcry	2017	Uses ShadowBrokers SMB exploit to spread in worm-like fashion
(Not)Petya	2017	Uses ShadowBrokers SMB exploit, Mimikatz-alike features + PSExec & WMIC

Some Ransomware Families

Due to its high effectiveness, new ransomware families are popping up on a very frequent basis these days. We list some interesting variants in this slide:

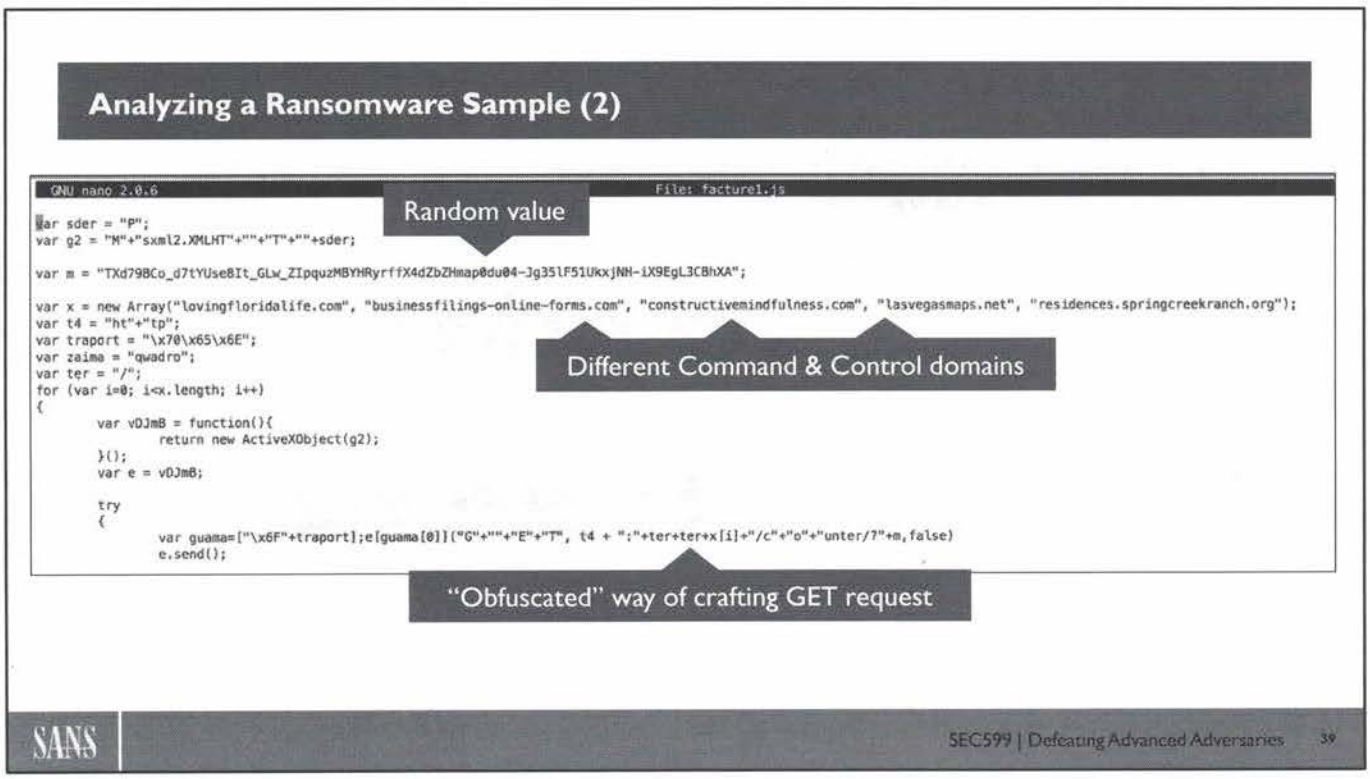
- Locky is one of the most popular ransomware samples out there. It's highly flexible and can be delivered using multiple exploit kits (drive-by downloads), or just using the traditional phishing scheme;
- Cerber is not your typical ransomware: it also includes other features / attack methods such as DDoS support;
- Jigsaw (themed like the movie "Saw") doesn't limit itself to only encrypting your data: it also steals it!
- The Crysis & LeChiffre ransomware variants have something interesting in common: they use brute force attacks against Windows RDP (Remote Desktop Protocol) to obtain access to victim systems (instead of the usual phishing techniques);
- Goldeneye, Petya and HDDCryptor attempt to not only encrypt individual files: When ran with administrative privileges, they will attempt to encrypt the entire hard drive and overwrite the Master Boot Record;
- Popcorn Time implements the interesting "infect-a-friend" function, where victims receive the decryption key for free provided they infect a number of other users / friends.
- Wcry caused a major impact in May 2017, holding several large organizations hostage. The "innovative" part of the attack was the use of an SMB exploit (published by the ShadowBrokers) to spread the ransomware throughout victim networks.
- (Not)Petya rose to stardom in June 2017, as it impacted several large organizations. While also relying on an SMB exploit (published by the ShadowBrokers), it coupled this with a highly effective combination of Mimikatz-alike techniques (to steal credentials) and PsExec / WMIC to perform lateral movement. Several experts claim that the ransomware-part of the malware was only a distraction of its actual intent, which was to cause as much downtime / damage as possible.



Analyzing a Ransomware Sample (1)

The above picture is an example of a typical phishing email carrying a ransomware payload sent to the "info@" mailbox of the company of yours truly (a cyber security consulting organization). It is not targeted specifically against our organization. The attack is not really tailored (although the email message is in French and the name of the sender appears to be "local"). An interesting trick here is that it's using typical cloud sharing providers (who usually aren't blacklisted at proxy level) to deliver its payload.

Once the ZIP file is downloaded and extracted, a JavaScript file is visible. Let's analyze what this JavaScript does…



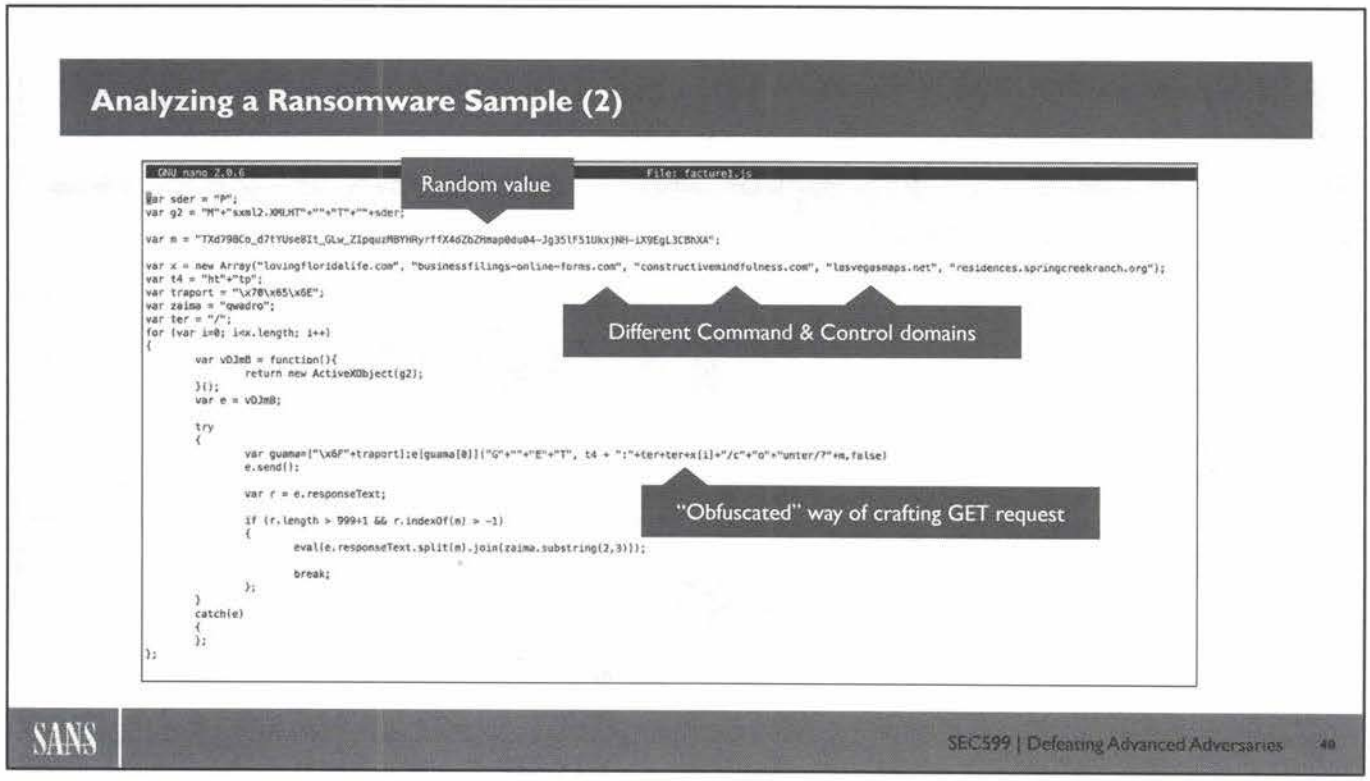
Analyzing a Ransomware Sample (2)

The sample above is a JavaScript script file that will download a payload which will start the next steps in the process (typically removing any available Volume Shadow Copies and starting the encryption process). Note that in the summer of 2016, samples were discovered that included the full encryption functionality inside the JavaScript itself (so without downloading an additional payload).

In our sample above, we can easily spot:

- Different command & control domains as part of the "X" variable;
- The GET request to a "/counter/?" page being constructed in the "guama" variable;
- An apparently random value in variable "m" that is appended as a URL parameter to the GET request;

Some very basic obfuscation is used throughout the sample (e.g. the GET method name is constructed by concatenating "G", "E" and "T".)



Analyzing a Ransomware Sample (2)

The sample above is a JavaScript script file that will download a payload which will start the next steps in the process (typically removing any available Volume Shadow Copies and starting the encryption process). Note that in the summer of 2016, samples were discovered that included the full encryption functionality inside the JavaScript itself (so without downloading an additional payload).

In our sample above, we can easily spot:

- Different command & control domains as part of the "X" variable;
- The GET request to a "/counter/?" page being constructed in the "guama" variable;
- A apparently random value in variable "m" that is appended as a URL parameter to the GET request;

Some very basic obfuscation is used throughout the sample (e.g. the GET method name is constructed by concatenating "G", "E" and "T".)

Zooming in on Sabotage

As a second attack motivator, let's see how sabotage is typically performed in cyber space!

Key Drivers for Sabotage

Sabotage attacks can be community-based ("hacktivism") or state-sponsored, launching attacks aimed at undermining or disrupting opponents

A variety of tools is typically used in order to achieve this goal (e.g. DDoS, defacements, erasing of documents,…)

Some of the most infamous hacktivist groups include Anonymous ("motivated by their beliefs"), LulzSec ("for fun / fame") and Lizard Squad

Key Drivers for Sabotage
Sabotage attacks can be "community-based" (hacktivism) or state-sponsored, typically launching attacks aimed at undermining or disrupting opponents:

- Hacktivism is a portmanteau word: hack and activism; activists that engage in hacking to convey a political message. Typical tools used by such groups include DDoS (Distributed Denial of Service) or website defacements. These attacks are typically less complex in nature and are thus "easier" to fend off as opposed to sabotage being performed by state-sponsored adversaries;

- State-sponsored attacks are often more complex and targeted in nature: the adversary has a specific purpose and objective. Typical examples include sabotage of critical infrastructure such as utilities, military installation, banks… Sabotage could be performed by erasing data / information, or even just by stealing & exposing it or making modifications (confidentiality, integrity & availability can be impacted).

Some infamous hacktivist groups include Anonymous (idealistic motive), LulzSec and Lizard Squad.

LulzSec

LulzSec was a splinter group of Anonymous, infamous for its short streak of data breaches they did "for the lulz". As a company, being compromised by such a group will at the very least lead to reputational damage.

Date	Target	Impact	Techniques used?
May 2011	Fox News	X-factor participant user data released	SQL Injection
May 2011	Sony	End-user data released	SQL Injection
June 2011	www.senate.gov	End-user data released	SQL Injection
June 2011	www.cia.gov	Website down for +-2 hours	DDoS
June 2011	www.pron.com	End-user data released	SQL Injection
June 2011	Bethesda Games Studio	End-user data obtained	SQL Injection

LulzSec ceased operations in Summer 2011, after a number of suspected members were arrested and brought to trial

LulzSec

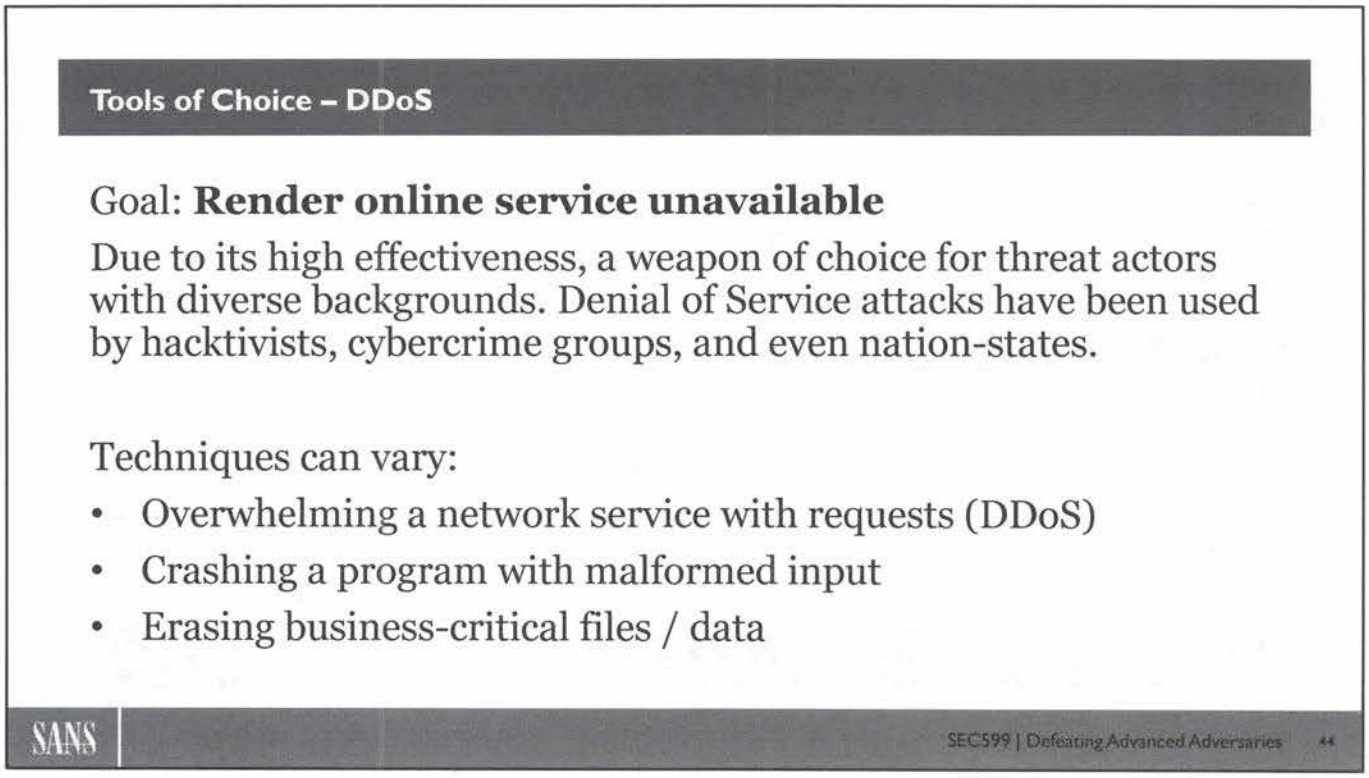
Founded in 2011 as a splinter group of "Anonymous", the hacktivist group LulzSec was notorious for their data breaches. They claimed to do this "for the lulz" (for the fun), which explains their name (Lulz Security, abbreviated to LulzSec). The techniques they used were often rather "basic", including for example SQL injection or DDoS attacks against public facing websites.

That didn't make them any less effective though:

- Their first target was Fox.com because they felt that Fox News had discredited a rap artist. LulzSec members obtained credentials and personal data of 73,000 X Factor participants. This information was leaked in an attempt to harm Fox. The Public Broadcasting System website was also compromised by LulzSec. They obtained user credentials which they disclosed, claiming to do this to defend WikiLeaks.

- LulzSec is most known for its attack on Sony, obtaining credentials, e-mail and physical addresses of tens of thousands of people (one million according to LulzSec). This was done through a SQL injection.

After the attacks against Fox News and Sony, LulzSec rose to fame and followed up quickly with many more SQL Injection and DDoS attacks. They, however, ceased operations in July 2011, after a number of suspected members were arrested and brought to trial.



Tools of Choice – DDoS

A denial of service is an attack where the offered service is impaired or rendered unavailable due to an attack. It's a tool of choice for a number of threat actors, such as:

- Hacktivists (e.g. taking down websites or services to make a statement);
- Cybercrime groups (where they could ask for a ransom to restore services);
- Nation states (e.g. take down a component of an adversary's critical infrastructure).

Depending on what type of service is rendered unavailable, the impact of such an attack can be diverse. Consider the difference between taking down an online "brochureware" website and taking down the power grid for an entire state…

Note that the technique behind a denial of service attack can be diverse:

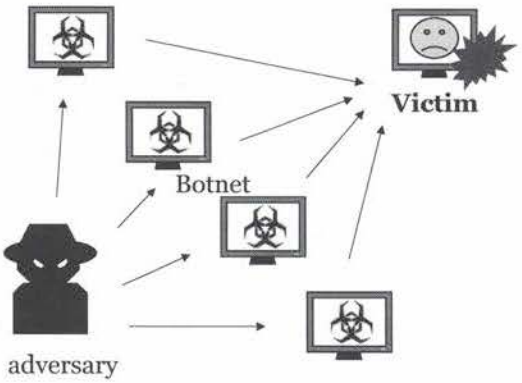
- Crashing a program with malformed input;
- Overwhelming a network service with requests (DDoS);
- Deleting business-critical files;
- …

We will zoom in on some interesting examples in the next few slides.

Analyzing a DDoS attack: Dyn (1)

Common attack technique: Distributed Denial of Service (DDoS)

- **Case: 2016 Dyn DDoS Attack**
- Target: DNS Provider Dyn
- Generated 1.2 terabits per second
- Major organizations affected
- Mirai Botnet (mostly consisted of IoT devices)

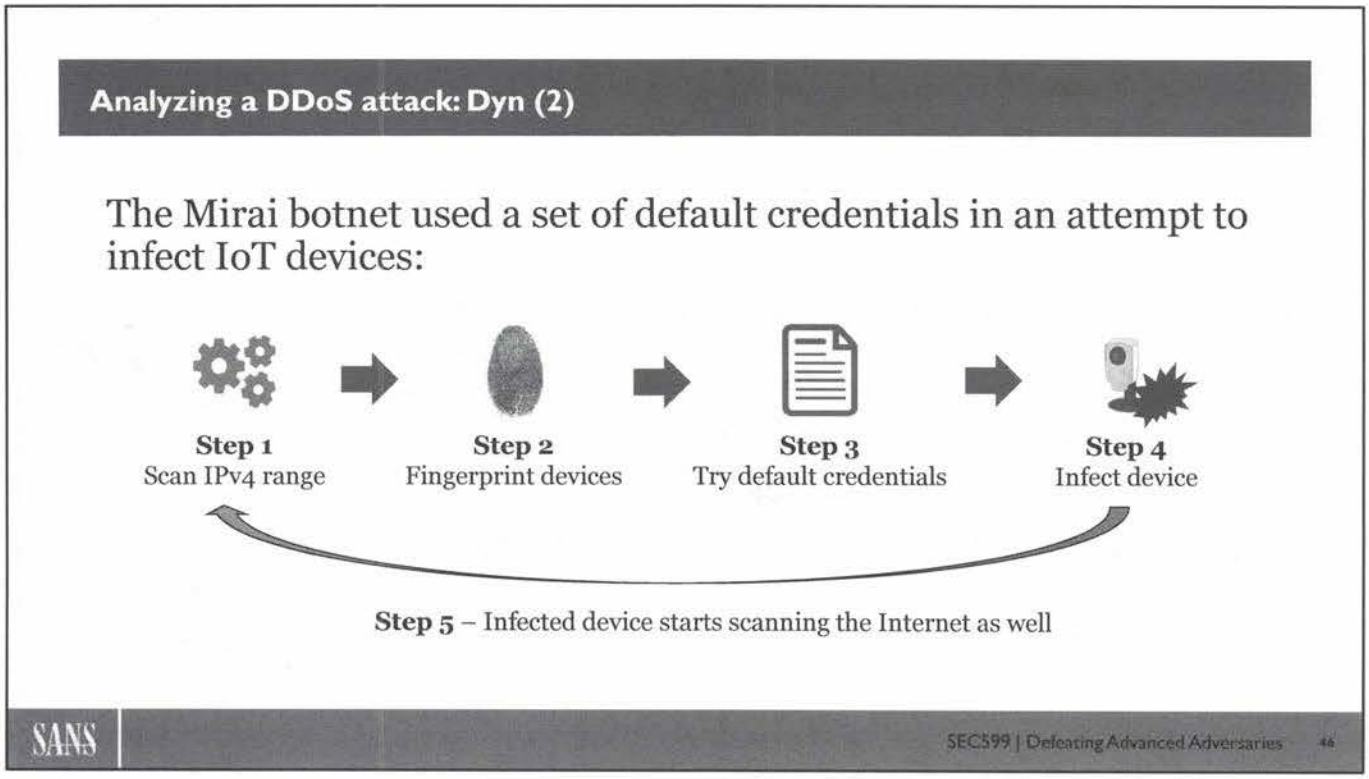


Analyzing a DDoS Attack: Dyn (1)
The example here covers a network denial of service. In a network denial of service, the service is sent so much network traffic by the adversary(s) that the service is not available at 100% of its capacity. It can be slow in responding, drop requests, or stop responding altogether. When more than one network endpoint is used to attack the service, we use the term distributed denial of service (DDoS).

On October 21, 2016, several Distributed Denial of Service attacks were conducted against Domain Name System provider Dyn. As you know, the DNS protocol is a vital component of the Internet as we know it today.

Dyn provides DNS services for major Internet services. The attack was performed by bombarding Dyn's DNS servers with lookup requests from 10s of millions of IP addresses. It is assumed that most of these requests originated from compromised IoT (Internet of Things) devices. At one point the attack culminated in an estimated traffic of 1.2 terabits per second, the largest DDoS attack ever at that time. This onslaught of traffic resulted in Dyn's DNS servers failing to respond to legitimate DNS queries. The impacted services included amongst others Amazon.com, PayPal, Visa, …

The compromised IoT devices were under control of a botnet called Mirai. A botnet (robot network) is a network of compromised devices under the control of criminals.



Analyzing a DDoS Attack: Dyn (2)
The Mirai botnet is a collection of IoT devices such as IP cameras, network connected printers, … that have been compromised. Mirai is malware that compromises IoT devices running Linux. An IoT device compromised with Mirai scans the Internet for devices with public IP addresses.

The process of scanning a network involves sending network packets to a list or range of IP addresses to see which IP addresses reply. A reply indicates that the IP address is assigned to a device and that the device is active and answering to queries. It is feasible to scan the entire IPv4 address range: IPv4 uses 4 bytes (32 bits), creating an address space of 4,294,967,296 addresses.

Mirai scans the entire IPv4 address range, excluding private IP ranges and some government-owned IP ranges. When a device infected with Mirai receives a reply to its scan, it will try to fingerprint the device it discovered. If the device appears to be an IoT device (like an IP camera) for which Mirai has the default credentials, it will attempt to remotely log in to the fingerprinted device with these credentials.

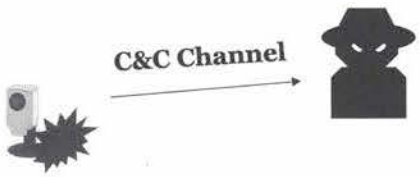
Many IoT devices can be remotely controlled by establishing an HTTP(S), SSH or even telnet connection. They often require authentication, but many IoT devices have default credentials, like username Admin and password Admin. IoT device manufacturers use default credentials as a simple method to secure the device. With some IoT devices, users are expected to change the default credentials after the initial configuration, but this step is often omitted, leaving the device accessible to anyone with knowledge of the default credentials. Other IoT devices have hard-coded default credentials: these cannot be changed by the user. Estimations indicate that there are millions of IoT devices with default credentials.

Mirai has a list of more than 50 default credentials. When Mirai successfully authenticates to an IoT device with default credentials, it infects the IoT device by uploading the Mirai malware to the device, persisting the malware and then starting it. A newly infected IoT device becomes a member of the botnet and falls under the control of the criminals. It will also start to propagate Mirai by scanning the Internet.

Analyzing a DDoS attack: Dyn (3)

An infected host will communicate with a Command & Control system to receive instructions:

- Typically a commonly used protocol (DNS, HTTP...)
- A set of commands included, defining the botnet capabilities
- For Mirai, focus was on DDoS



Mirai malware source code was open sourced, resulting in many variants

Analyzing a DDoS Attack: Dyn (3)

A compromised IoT device will connect to a command and control server (aka as C&C or C2). A C&C server is operated by the criminals and used to control the devices connected to it by sending them commands. The set of devices connecting to the same C&C server define a botnet. Communication between C&C and bots (compromised devices) can take many forms such as custom TCP protocols, HTTP(S), Telnet, DNS… Adversaries typically favor commonly used protocols such as DNS or HTTP as these are commonly allowed at perimeter level and their presence does not raise too much suspicion.

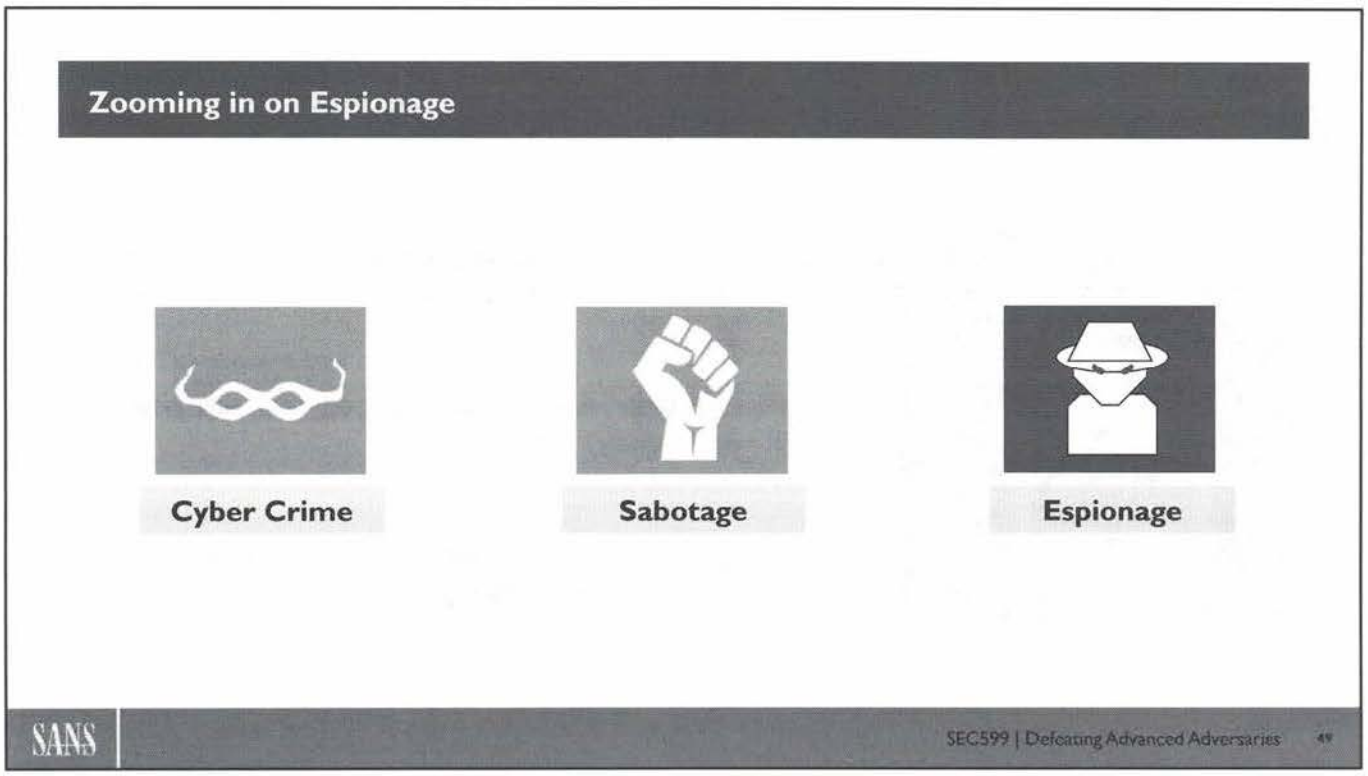
The type of commands that a bot can receive and execute define the capabilities of a botnet. Capabilities often found in botnets are performing DDoS attacks and sending emails. Sending emails is used for all types of unwanted emails, not only SPAM (unwanted advertisement emails) but also phishing emails, emails with links to malware and emails with malicious attachments.

In the 2016 Dyn attack, Mirai bots received commands from the C&C to continuously send DNS requests to the Dyn DNS servers. A single compromised IoT device cannot impair the correct and timely functioning of a DNS server with continuous DNS requests. The bandwidth of DNS traffic from a single device is too small to have a performance impact on DNS servers. DNS server software is designed to handle a lot of requests efficiently, and DNS server hardware and network connections are dimensioned to handle large peaks in network traffic, as well as sustained traffic. Hundreds of thousands of compromised IoT devices, however, can generate enough DNS requests to bring down even the most powerful of DNS servers.

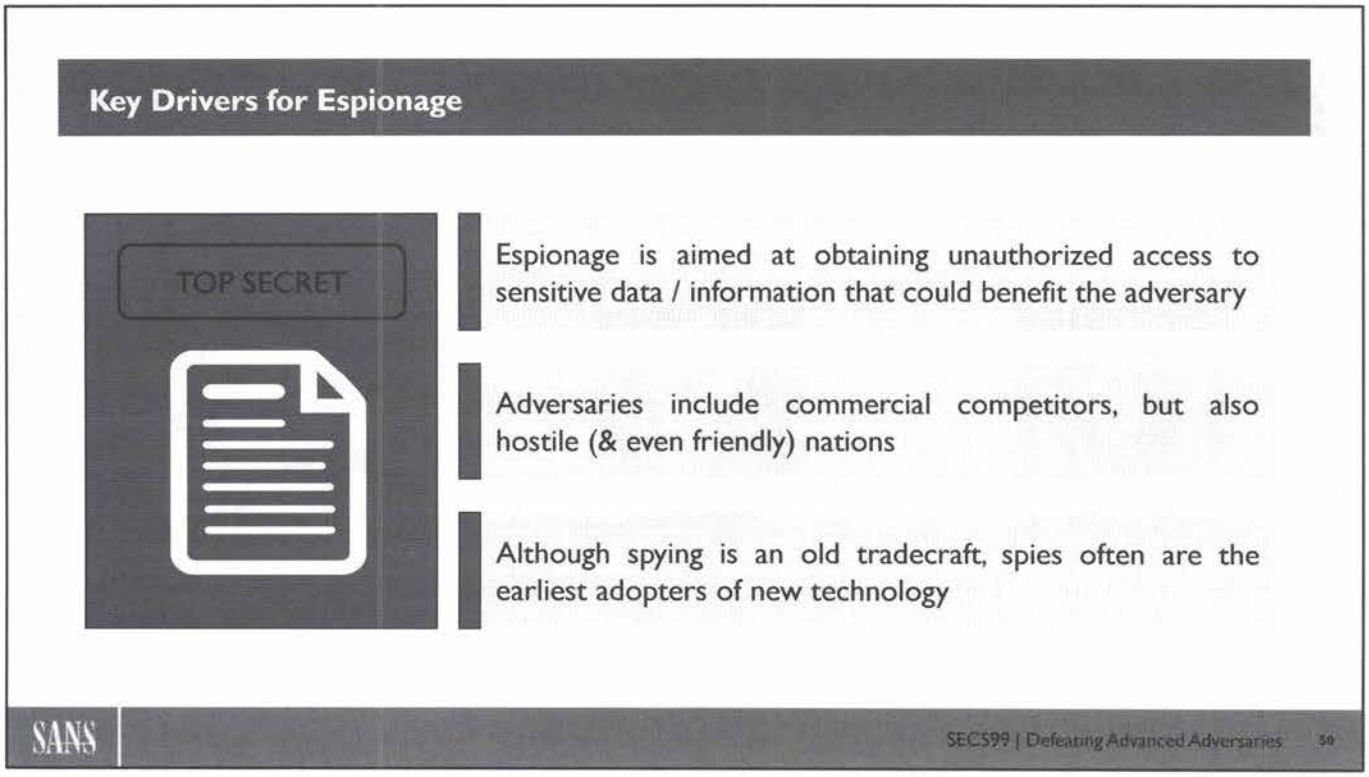
The source code of the Mirai malware was open sourced: it was released on hacker forums. This resulted in the appearance of many variants of Mirai (malware versions with different characteristics, features, and capabilities) and the adoption of Mirai's techniques in other, existing malware. It also allowed security researchers to better understand Mirai by examining the source code. It is speculated that the author(s) of the Mirai source code decided to open source the Mirai malware to make attribution harder.

Attribution is the act of ascribing an action or work to a particular person or group. As long as Mirai was closed source, all actions by the Mirai botnet could be attributed to the Mirai authors, and from that infer the motives of the authors. Because of the adoption of the source code of Mirai by many persons and groups,

many similar botnets have appeared and attribution of actions to a particular individual or group based on botnet and malware characteristics has become harder, because of the many-to-many relationships that exist now.



Zooming in on Espionage
Finally, let's turn our eyes to the interesting world of cyber espionage!



Key Drivers for Espionage

Espionage is the act of spying. Spies obtain secret information without permission and knowledge from their targets. Spying is done by many actors and at many levels.

Nation states have always had espionage agencies, like the CIA in the USA and KGB in the Soviet Union. These agencies collect information about their targets to further the cause of the nation. This is military information to be better prepared in case of an armed conflict, and political information to have an advantage in political negotiations with insider information. For example, if you know to what level your opponent is willing to make concessions during negotiations, you can use this to your advantage to get a better deal.

Companies and organizations also spy on each other, within and outside national borders. This industrial espionage is performed to have an advantage over the competition. Research and development is a very costly activity companies engage in, and it doesn't always yield expected results. But R&D is vital for a companies' growth. A cheaper way is to obtain research from a competitor and develop new products before the competitors do. This is illegal in most countries, but it will not stop companies from doing this. Sometimes the cost and risk of espionage outweigh the cost of research, and companies engage in industrial espionage and factor in the fines they will have to pay if they get caught.

Spying is a millennium-old practice; however, it is not behind in its use of technology. Spies are often early adopters of new technology to improve their practices. With the digitalization of technology came the need to have the capability to steal information from digital devices. This resulted in espionage actors adopting sophisticated technology to infiltrate these digital devices.

Some Known Espionage Groups

The current cyber espionage landscape includes actors from all parts of the world (**"everybody is doing it"**), we listed some of the most known groups below:

Name	Other Names	Known Campaigns	Main Targets
APT-28	Sofacy, Fancy Bear	Grizzly Steppe, DNC Hack, Bundestag	US & European Governments
APT-29	Dukes, Cozy Bear	Grizzly Steppe	US & European Businesses
Turla	Snake	Satellite Turla, Epic Turla	US & European Governments & Businesses
Sandworm	TEMP.Noble	Black Energy	Eastern European Utilities
APT-1	Comment Panda	ShadyRAT	English-speaking high-tech firms
APT-3	Gothic Panda	Clandestine Fox, Double Tap	Worldwide defense contractors
APT-27	Emissary Panda	Operation Iron Tiger, A Tale of 2 Targets	US Government & defense contractors
Charming Kitten	Parastoo	Stonedrill, Shamoon	Saudi & US Interests (focus on utilities & defense contractors)
Copy Kitten	Slayer Kitten	Matryoshka	Israeli Interests in the Middle-East
Rocket Kitten	TEMP.Beanie	Operation Woolen Goldfish	Saudi Arabian, Israeli & US Interests in the Middle-East
Equation Group	Tilded Team	Stuxnet, Regin	Worldwide

Some Known Espionage Groups

The current cyber espionage landscape includes actors from all parts of the world. It's safe to assume that the vast majority of states are developing and using cyber espionage capabilities. The table in the slide lists a number of different groups, coupled with some of their best-known campaigns and the types of organizations they usually target.

It's important to note that attribution based on pure "technical facts" is often difficult. For example, "the source IP address is from Russia" or "the command & control server is hosted in China" are highly unreliable elements in attribution (as they can be easily forged). Some more interesting elements that are currently being used for attribution:

- Similarities in coding style (or even copy / paste work);
- Similarities in tools that are used (e.g. the use of Mimikatz during post-exploitation);
- Exploitation of the same vulnerabilities (e.g. 0-days that are used in different campaigns);
- Sophistication of the malware;
- Artefacts identified during analysis (e.g. PDB path, compilation times…);
- The target of the attack (correlated with the current geopolitical situation);

Still, attribution remains a difficult topic in today's world.

References

https://docs.google.com/spreadsheets/d/1H9_xaxQHpWaa4O_Son4Gx0YOIzlcBWMsdvePFX68EKU/edit#gid=1864660085

The Equation Group – "Omnipotent" Adversaries (1)

$$\sum \left(\bar{x} - \bar{x}_G\right)^2 n$$

- The Equation group
- An espionage group active all over the world
- First appearance in early 2000's
- Discovered and reported by Kaspersky
- Very sophisticated in its use of technology
- Linked to high-profile campaigns such as Stuxnet & Regin

The Equation Group – "Omnipotent" Adversaries (1)
The Equation group is an industrial espionage group active all over the world. They use very sophisticated malware to spy on their targets. Kaspersky, a Russia-based (though globally active) anti-virus company, discovered and analyzed their malware. The extensive report published by Kaspersky Lab calls them the most sophisticated espionage group they have seen to date.

Kaspersky christened this espionage group "The Equation group" because of their predilection to use strong encryption in their malware and methods of operation. The group is active in many countries and national and economical sectors.

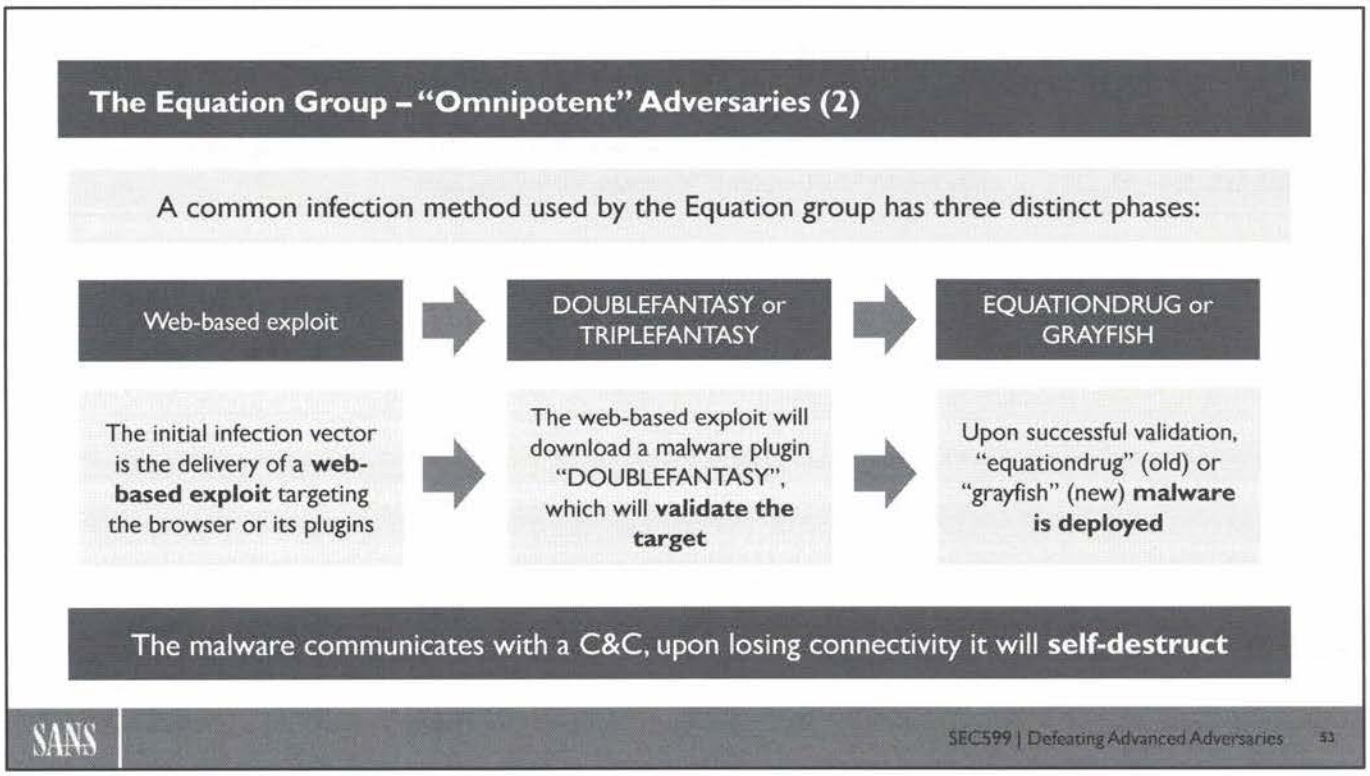
Taking the infection detection rate as an indicator, the group seems to be very active in Russia, Iran and other countries of that region. Infection targets were observed in more than 30 countries.

The group is not only active in industrial espionage, targeting sectors like finance, energy, telecommunications, …, but also targets governments, military, diplomats, …

The Equation group uses very sophisticated malware to infect their victims. Kaspersky has identified many malware families attributed to the Equation group, giving these families names like EQUATIONDRUG, DOUBLEFANTASY, GRAYFISH, FANNY. These families go back to the early 2000's, maybe even the 90's. This extensive collection of malware families indicates the vast resources the Equation group can command. The use of zero-day exploits is more evidence of their sophistication. A zero-day is an exploit for a vulnerability that is not publicly known and has not been patched by the vendor.

Equation group's malware primarily targets the Windows operating system, although OSX also seems to be targeted. They also have the capability to infect firmware of hard disks, a very sophisticated attack vector.

Reference: https://securelist.com/files/2015/02/Equation_group_questions_and_answers.pdf



The Equation Group – "Omnipotent" Adversaries (2)
This diagram illustrates one of the sophisticated methods used by the Equation group to infect the Windows machine of their victim.

First, the victim is directed to a website with an exploit. This can be done via a link in an email, via social media or even SMS. The website contains an exploit for a vulnerability in the browsers victim. The group has been known to use various exploits, like exploits for vulnerabilities CVE-2013-3918 (Internet Explorer), and CVE-2012-1723 and CVE-2012-4681 (Java). These exploits allow for remote code execution on the victim's machine. This is the first step in the compromise life cycle of this particular method used by the Equation group.

Next, the payload of the web-based exploit will download DOUBLEFANTASY. DOUBLEFANTASY is a malware plugin that will first validate the target. Before installing the full-featured espionage platform, the Equation group wants to validate that they infected their intended victim and that it is sufficiently interesting to them to warrant the deployment of the espionage platform.

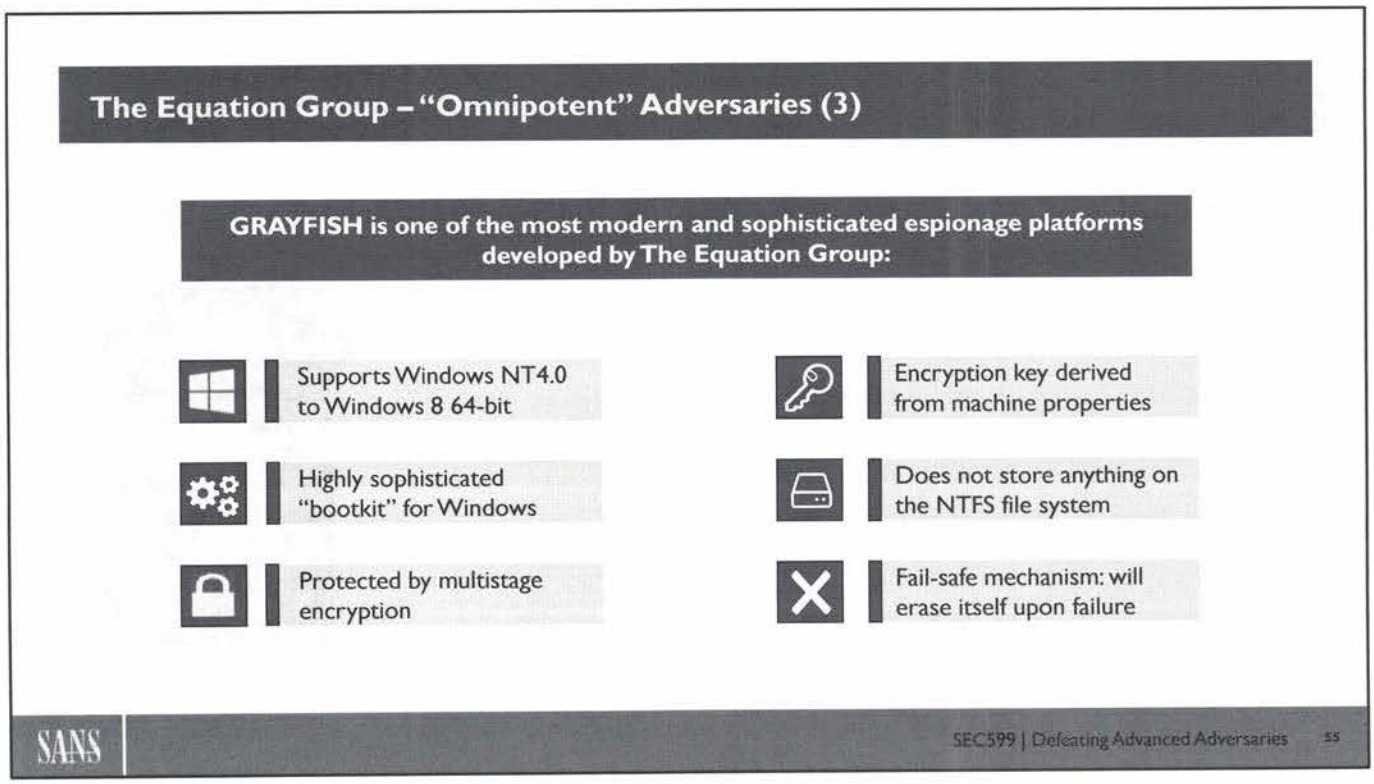
After validation, espionage platforms EQUATIONDRUG or GRAYFISH are deployed. EQUATIONDRUG seems to be developed by the group between 2003 and 2013, while GRAYFISH is a later platform replacing EQUATIONDRUG. Development of GRAYFISH started in 2008. It is the most modern and sophisticated malware from the group.

EQUATIONDRUG and GRAYFISH are modular malware, using plugins and drivers to extend their features.

When deployed, EQUATIONDRUG contains core functionality giving the adversaries full control over the Windows operating system of the infected machine. In cases where the core functionality is not enough for the intended espionage activities, extra plugins and drivers can be installed. Kaspersky identified 35 different plugins and 18 different drivers.

Information stolen from the victim is stored in encrypted font files that are uploaded to the C&C. If EQUATIONDRUG can no longer communicate with the C&C, it will self-destruct.

EQUATIONDRUG operates on Windows operating systems up to Windows 2000 (including pre-NT versions like Windows 95, 98, …), while GRAYFISH supports Windows NT 4.0 and later.



The Equation Group – "Omnipotent" Adversaries (3)

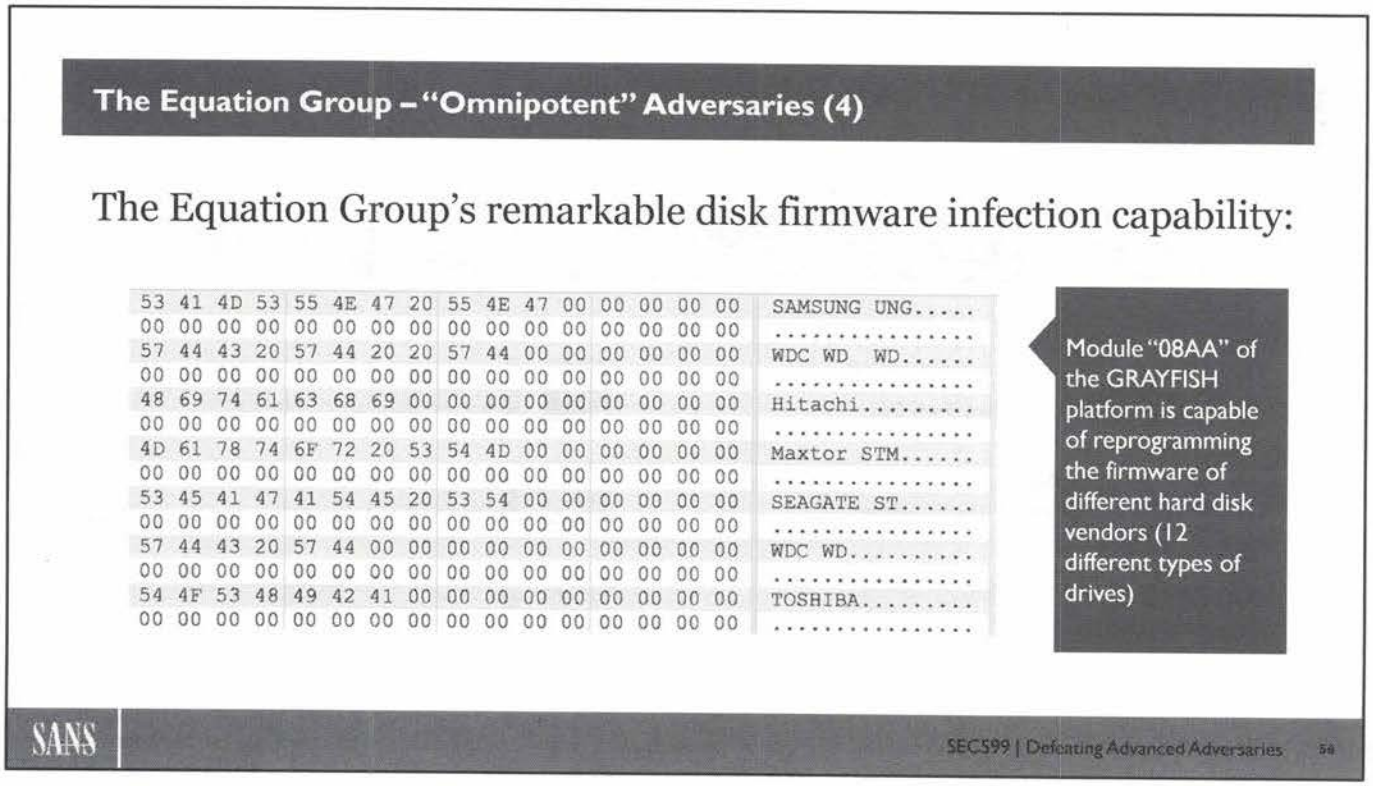
GRAYFISH is, according to Kaspersky Lab, the most modern and sophisticated espionage platform from the Equation group. It appears to have been developed between 2008 and 2013. The sophistication of the techniques used strongly indicates that very skilled developers worked on this malware. It is compatible with all Windows versions from Windows NT 4.0 to Windows 8, 32-bit and 64-bit versions.

When a computer infected with GRAYFISH is started, GRAYFISH activates very early in the boot process. GRAYFISH injects its code into the master boot record, the very first piece of code that is executed after the BIOS is started. The different stages of GRAYFISH will activate during the Windows boot process: loading of Volume Boot Record, loading of Windows Boot Manager, loading of Windows kernel and boot drivers, running Windows processes and services.

Each stage is encrypted with a key derived from the machines properties (like the object ID of the NTFS System folder, similar to Gauss), and GRAYFISH will self-destruct if any stage fails.

GRAYFISH does not store its components directly in files on the NTFS file system but uses boot records, sectors on the hard disk, and an encrypted virtual file system stored in the registry. This VFS is not only used to store components of GRAYFISH but also to store stolen data.

The Equation group has used zero-day exploits that were also used in Stuxnet, making Kaspersky conclude that the Equation group developers are somehow linked to the Stuxnet developers. This observation was reported by various media as the attribution of the Equation group to the NSA.



The Equation Group – "Omnipotent" Adversaries (4)
One salient feature of the Equation groups technological capabilities, it's the technology they developed to infect the firmware of hard disks.

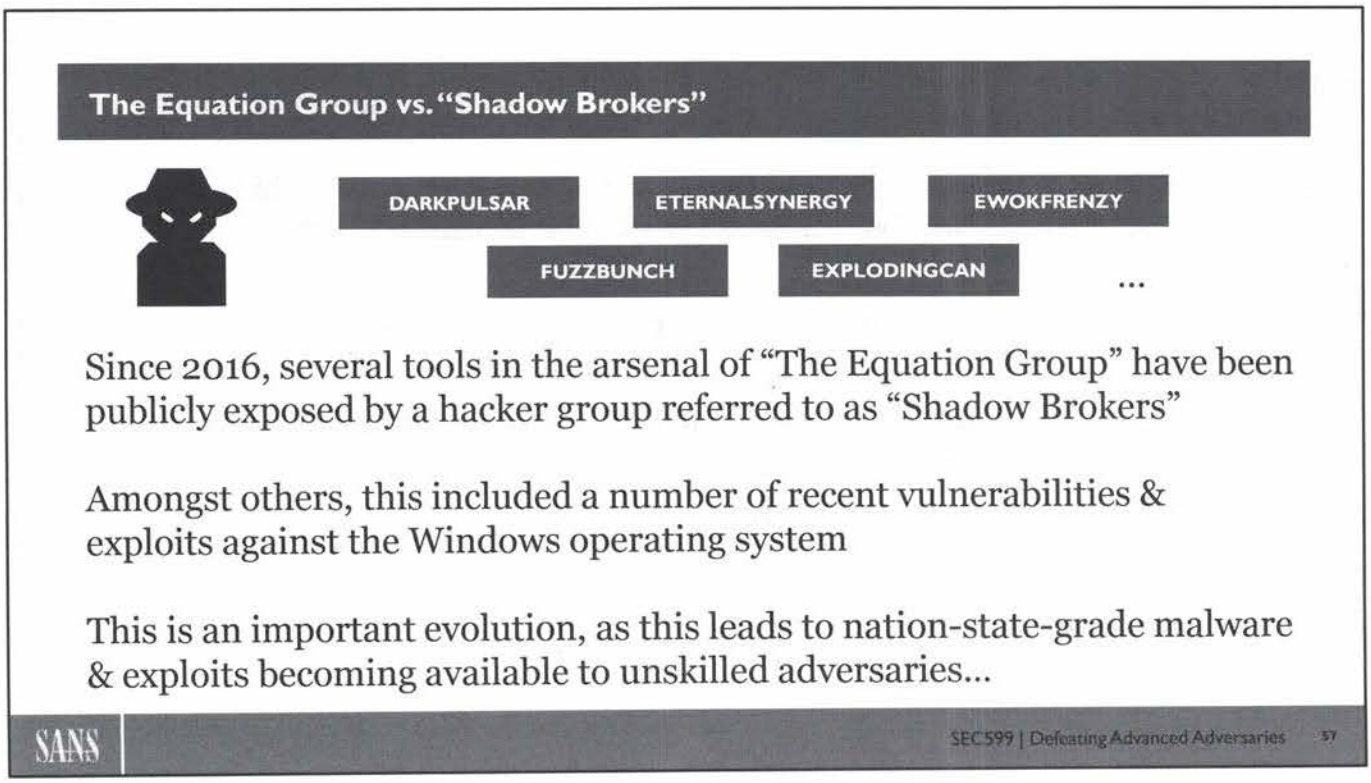
While with old computer technology, the hard disk was just an electro-mechanic device controlled by the operating system or a dedicated controller, modern hard disks include their own controller software. This software is called firmware.

Kaspersky identified a module (ID 80AA) of the EQUATIONDRUG and GRAYFISH platforms capable of reprogramming the firmware of different hard disk vendors. This plugin with version number 3 for EQUATIONDRUG was capable of reprogramming 6 types of drives, while plugin version 4 for GRAYFISH has the capability to reprogram 12 different types of drive. Both 32-bit and 64-bit versions were discovered.

Reprogramming the firmware of a hard disk is already an astounding technological feat. Having the resources to reverse engineer 12 different types of drives to develop the technology to reprogram them is even more astounding.

The goal of reprogramming the hard disk's firmware is to be able to store hidden data and achieve persistence even after reformatting the hard drive or reinstallation of the operating system. The Equation group's developers achieved this goal by programming a custom API in the hard disk's firmware, providing them access to a set of hidden disk sectors.

Kaspersky has only identified a couple of victims with this module, it is extremely rare. This could indicate that this technology is very valuable to the Equation group and is only used on targets they consider highly valuable.



The Equation Group vs. "Shadow Brokers"
Since 2016, several tools in the arsenal of "The Equation Group" have been publicly exposed by a hacker group referred to as "Shadow Brokers".

In April 2017, a number of exploitation tools used by "The Equation Group" were released. Although not "officially" 0-day, these included exploits against the Windows Operating System for vulnerabilities patched only one week before the release date.

This is an important evolution, as this leads to nation-state-grade malware & exploits becoming available to unskilled adversaries… In the weeks after the release, many organizations were compromised by unskilled adversaries using the state-of-the-art exploit tools released by the Shadow Brokers.

Current Threat / Attack Landscape – Additional Resources

Some additional resources related to the current threat landscape include:

- https://avien.net/blog/ransomware-resources/ransomware-families-and-types/
  List of malware families
- http://hackaday.com/2016/01/26/the-dark-arts-meet-the-lulzsec-hackers/
  About the LulzSec hackers
- https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/
  Dyn DDOS analysis summary
- https://www2.fireeye.com/rs/fireye/images/rpt-apt28.pdf
  APT28 and their targets
- http://blog.airbuscybersecurity.com/post/2016/10/Playing-defence-against-the-Equation-Group
  Analysis of Equation Group exploits

Current Threat / Attack Landscape – Additional Resources
Some additional resources related to the current threat landscape include:

https://avien.net/blog/ransomware-resources/ransomware-families-and-types/
List of malware families

http://hackaday.com/2016/01/26/the-dark-arts-meet-the-lulzsec-hackers/
About the LulzSec hackers

https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/
Dyn DDOS analysis summary

https://www2.fireeye.com/rs/fireye/images/rpt-apt28.pdf
APT28 and their targets

http://blog.airbuscybersecurity.com/post/2016/10/Playing-defence-against-the-Equation-Group
Analysis of Equation Group exploits

Prevent or Detect? (1)

So... There's quite some bad stuff happening out there. This brings us to an interesting question: Should we focus on preventing these adversaries from obtaining access or should we attempt to detect them? **We should do both!**

If we spot the adversary completing first phases of the APT attack cycle, we could still prevent later (and more damaging) phases from taking place!

Furthermore, controls aimed at detection allow us a higher "fault margin", as too stringent security controls implemented in a detection mode don't lead to business disruption. A good strategy is:

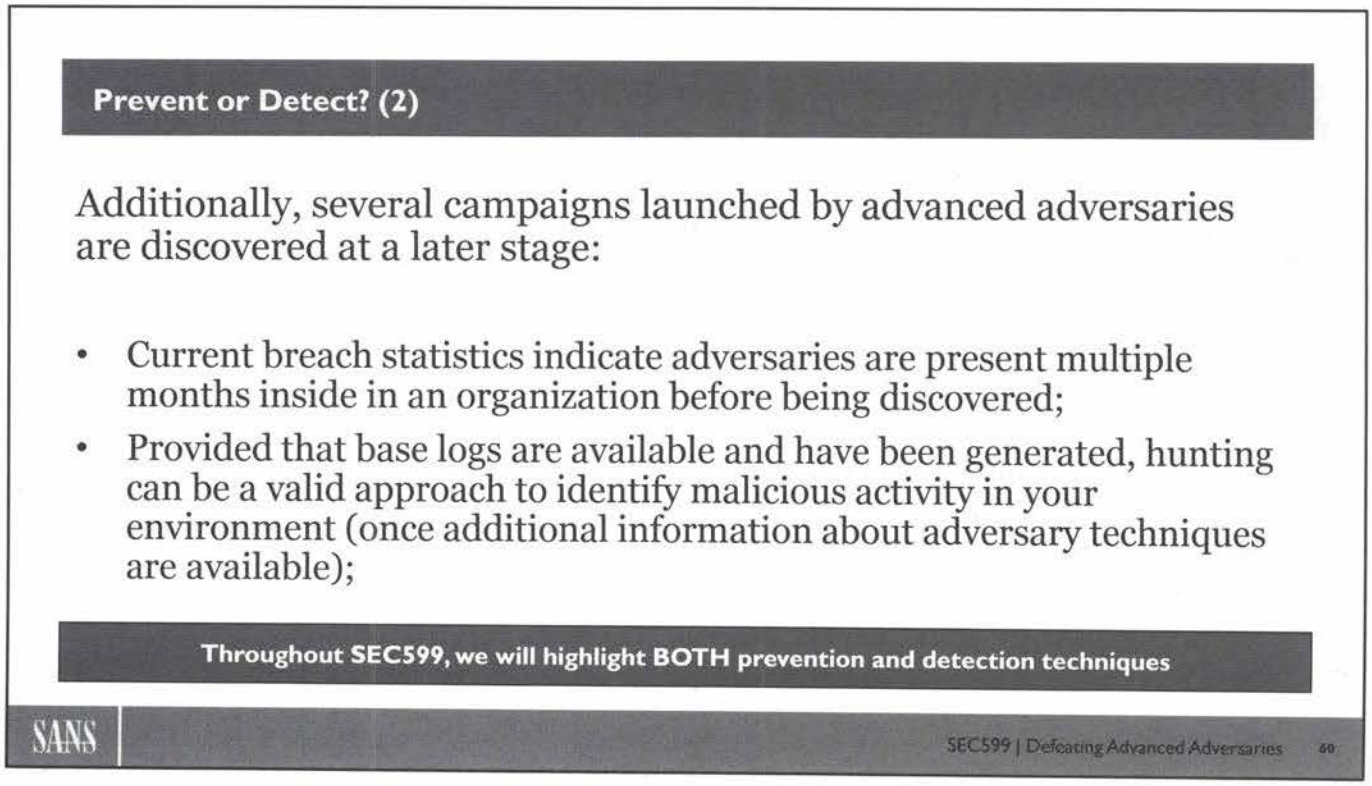
- Implement both detection & prevention controls;
- New security controls can be tested in detection mode first;
- Once false positive rate is low in detection, replicate control in prevention mode

Prevent or Detect? (1)

Work under the assumption that persistent adversaries with enough resources will succeed in the initial intrusion. When adversaries have months to prepare and execute the initial intrusion phase of the attack, it is safe to assume they will succeed, regardless of how good your defenses are. Complex systems like your network and computers always have vulnerabilities (through bugs, configuration errors or even a lack of security awareness of your staff) and a persistent attacker will have the time to discover and exploit them.

Prevention is important. However, working under the assumption that you will not be able to prevent all attacks, detection is even more important. This can be the detection of the attack itself or the actions of the adversaries after the initial intrusion (like lateral movement). Even though the attacker could have successfully completed the first steps of the kill chain, we might be able to prevent a more damaging phase, such as sensitive data exfiltration, from happening.

Additionally, controls aimed at detection are more forgiving than prevention mechanisms. A strict prevention control causing a lot of false positives will have a negative impact on business operations, as legitimate actions will be blocked. In case of a strict detection control, there might still be a large number of false positives reported, but the operational impact will be limited. As a result, it is a good idea to test a new control in detection mode first and replicate it in prevention mode once it has proven its worth and the false positive rate is reduced.



Prevent or Detect? (2)

Detecting all attacks in an automated fashion as actionable incidents is also unrealistic. Studies have repeatedly shown that the average time to detect a breach is high (although there appears to be a positive downwards trend). Unsurprisingly, a lot of these attacks were only discovered at a later stage, long after the initial attack phases had taken place. Once the attack is identified, it's usually deeply investigated and a whole lot of technical information is discovered in the form of IoC's (Indicators of Compromise). These IoC's (e.g. domain names, IP addresses, file hashes,) are shared with trusted parties and can be used by others to assess whether they have been impacted by the same adversaries.

Now imagine you receive a number of IoC's from a trusted party about an attack that took place 1 year ago. In order to leverage this information, it's vital that sufficient logs were generated at the time and were retained long enough. For this reason, basic logging is your next "safety net": when prevention and real-time detection fails, logging can help you retro-actively investigate your environment for compromises.

Prevent or Detect? – Additional resources

Some additional resources that can prove to be useful for prevention vs. detection include:

- https://securityintelligence.com/detection-not-new-prevention-advanced-threat-protection/
  Is detection the new prevention?
- https://www.cylance.com/content/dam/cylance/pdfs/white_papers/PreventionvsDetectandRespond.pdf
  Prevention vs. detect and respond
- http://www.csoonline.com/article/3186731/techology-business/prevent-or-detect-what-to-do-about-vulnerabilities.html
  Prevent or detect for CSOs

Prevent or Detect? – Additional Resources
Some additional resources that can prove to be useful for prevention vs. detection include:

https://securityintelligence.com/detection-not-new-prevention-advanced-threat-protection/
Is detection the new prevention?

https://www.cylance.com/content/dam/cylance/pdfs/white_papers/PreventionvsDetectandRespond.pdf
Prevention vs. detect and respond

http://www.csoonline.com/article/3186731/techology-business/prevent-or-detect-what-to-do-about-vulnerabilities.html
Prevent or detect for CSOs

Course Roadmap

- **Day 1: Knowing the adversary, knowing yourself**
- Day 2: Averting Payload Delivery
- Day 3: Preventing Exploitation
- Day 4: Avoiding Installation, foiling Command & Control & thwarting lateral movement
- Day 5: Exfiltration, Cyber Deception & Incident Response
- Day 6: APT Defender Capstone

SEC599.1

Course Outline & Lab Setup
- Course Overview & Objectives
- Attendee System Setup

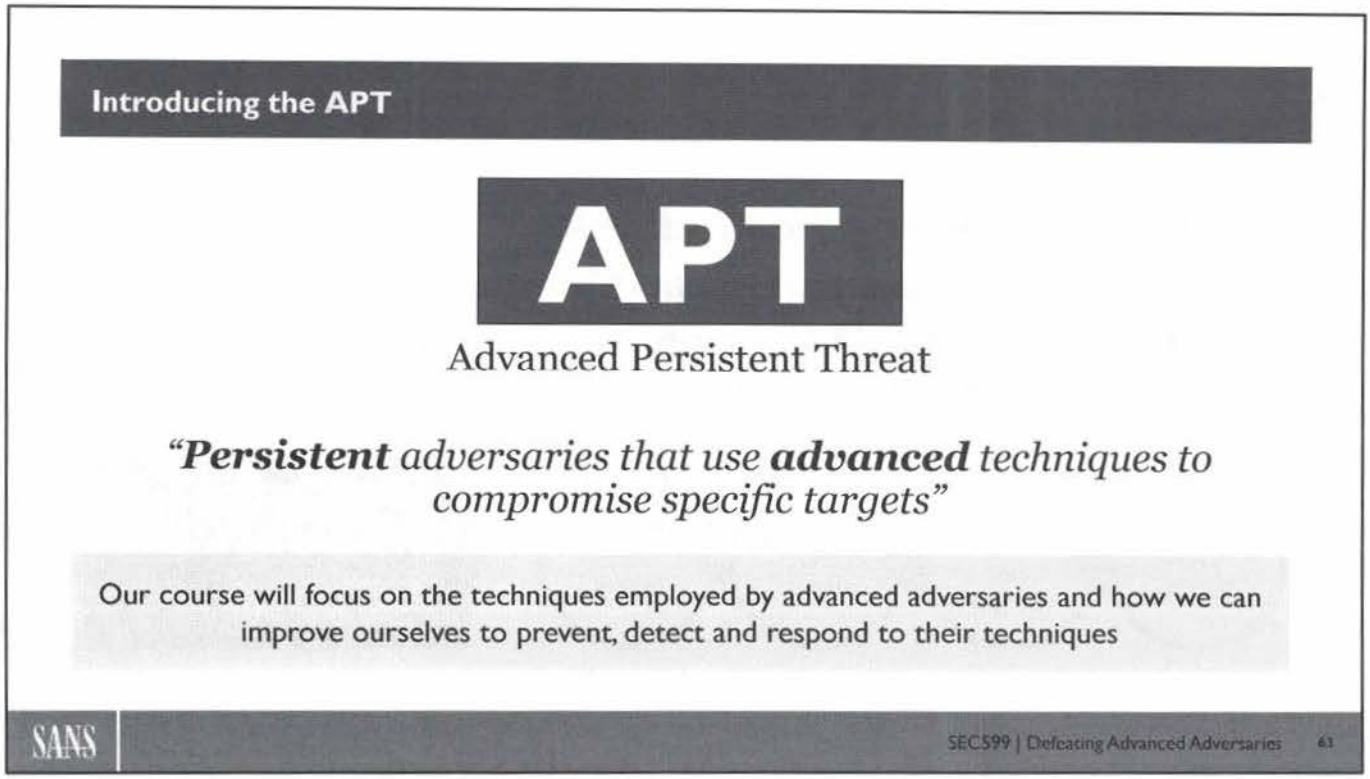
Current Threat / Attack Landscape
- Key Terminology
- What is happening out there?

Introducing the APT Attack Cycle
- Recent Case Studies – In-Depth
- Exercise: Analyzing The Behavior of Famous Malware
- Exercise: One Click Is All It Takes…

A Defensible Architecture & Environment

Preparation - Knowing Yourself
- Understanding Your Own Environment
- Determining What is "Normal"
- Understanding & Limiting Your Internet Footprint
- A Word on Vulnerability Management

This page intentionally left blank.



Introducing the APT

The term Advanced Persistent Threat was introduced by Colonel Greg Rattray from the United States Air Force in 2006. Although it has a bit of a "buzz-word" feeling to it, it does cover our topic quite well:

*"**Persistent** adversaries that use **advanced** techniques to compromise **specific targets**"*

Persistent

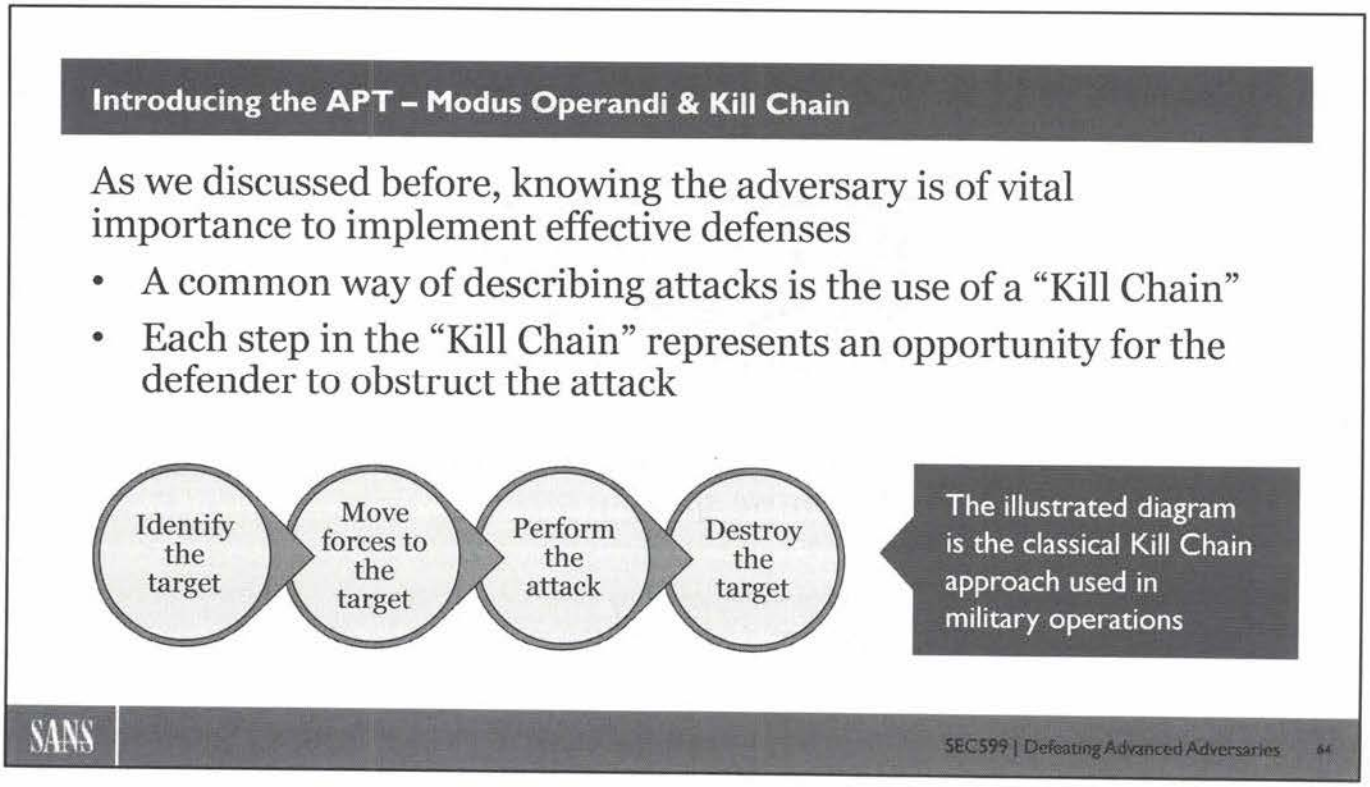
The term "persistent" indicates that the APT group has clear objectives and will not retreat after the first (failed) attack. APT groups knows what objectives to reach, and coordinate their attack in a persistent manner to reach the objectives.

Advanced

The term "advanced" does not necessarily mean that all attacks conducted by an APT group are sophisticated. They regularly start an attack with a simple phishing e-mail or common malware. The advanced part of APT means that the group can command advanced techniques, when necessary. Like using zero-day exploits that have no patches. But the APT group will only resort to advance techniques when necessary to reach the objective.

Threat

APT groups are considered a threat to your organization because they are motivated adversaries. They have clear objectives and the technical skills and resources necessary to reach them. This makes them a threat to your organization, that will not easily back off.



Introducing the APT – Modus Operandi & Kill Chain

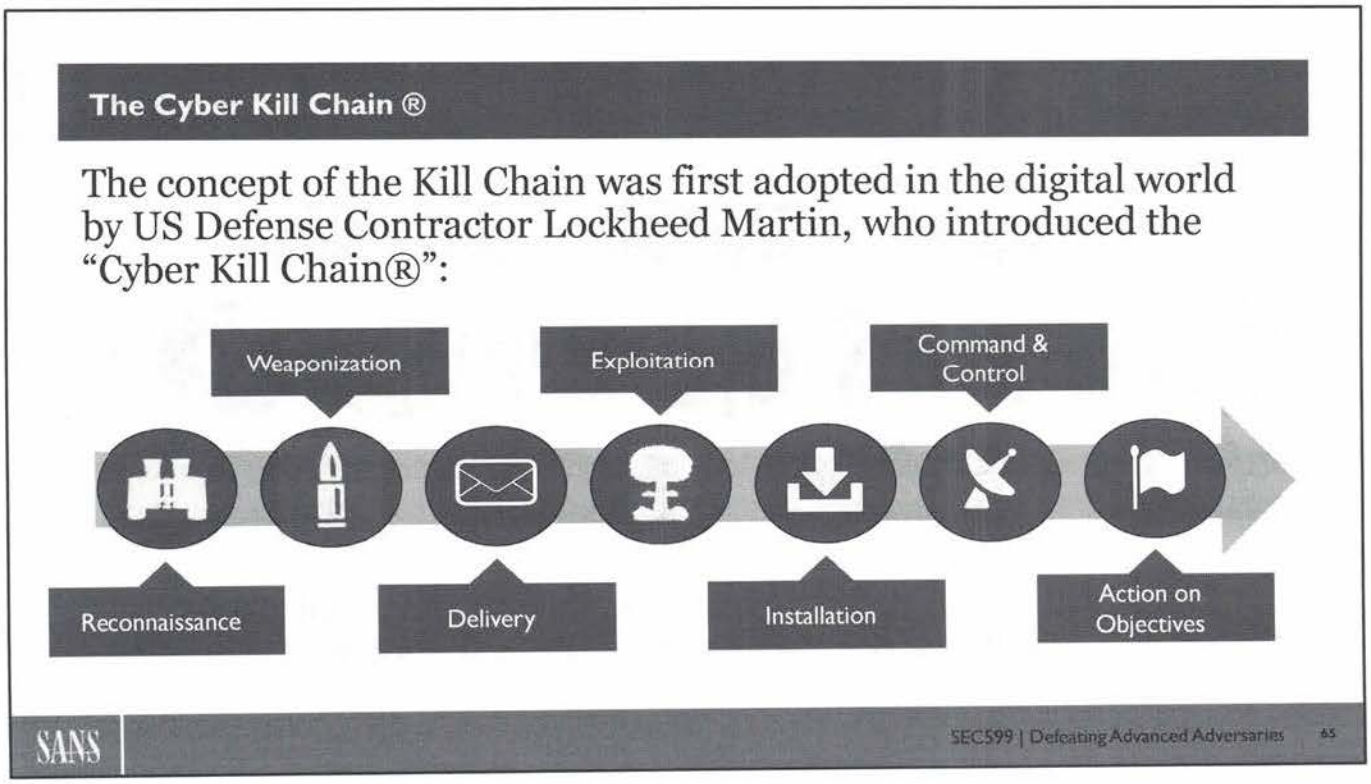
Understanding how our adversaries operate is of crucial importance. We can get some inspiration from the military industry, where adversaries and how they operate have been studied at length!

The term "kill chain" was introduced by military forces as they try to be as efficient as possible, working out processes and methods to reach their targets in a standardized way that can be taught and reproduced. The kill chain describes the different steps in an attack on a target. A standard military kill chain has the following elements:

- The first step is to identify the target: know what the target is, and gather information about the target.
- The second step is to decide which forces are needed to attack the target, and move those forces to the target.
- The third step is to give the order to the forces to attack the target.
- The final step is to destroy the target.

Military organizations prefer to standardize their procedures so that they can be more easily instructed to soldiers and executed. Codifying the attack with such a kill chain makes it easier to teach officers how to plan an attack, and soldiers to execute the attack.

Knowing the different steps of an attack is not only advantageous to the adversary. It is also useful to the defender. The method allows the defender to know how a typical attack will proceed (to a certain degree, without knowing all the details). And by knowing this method, the defender can also structure its defenses. From a defensive point of view, an adversary's attack can be stopped by "breaking" the kill chain.



The Cyber Kill Chain ®

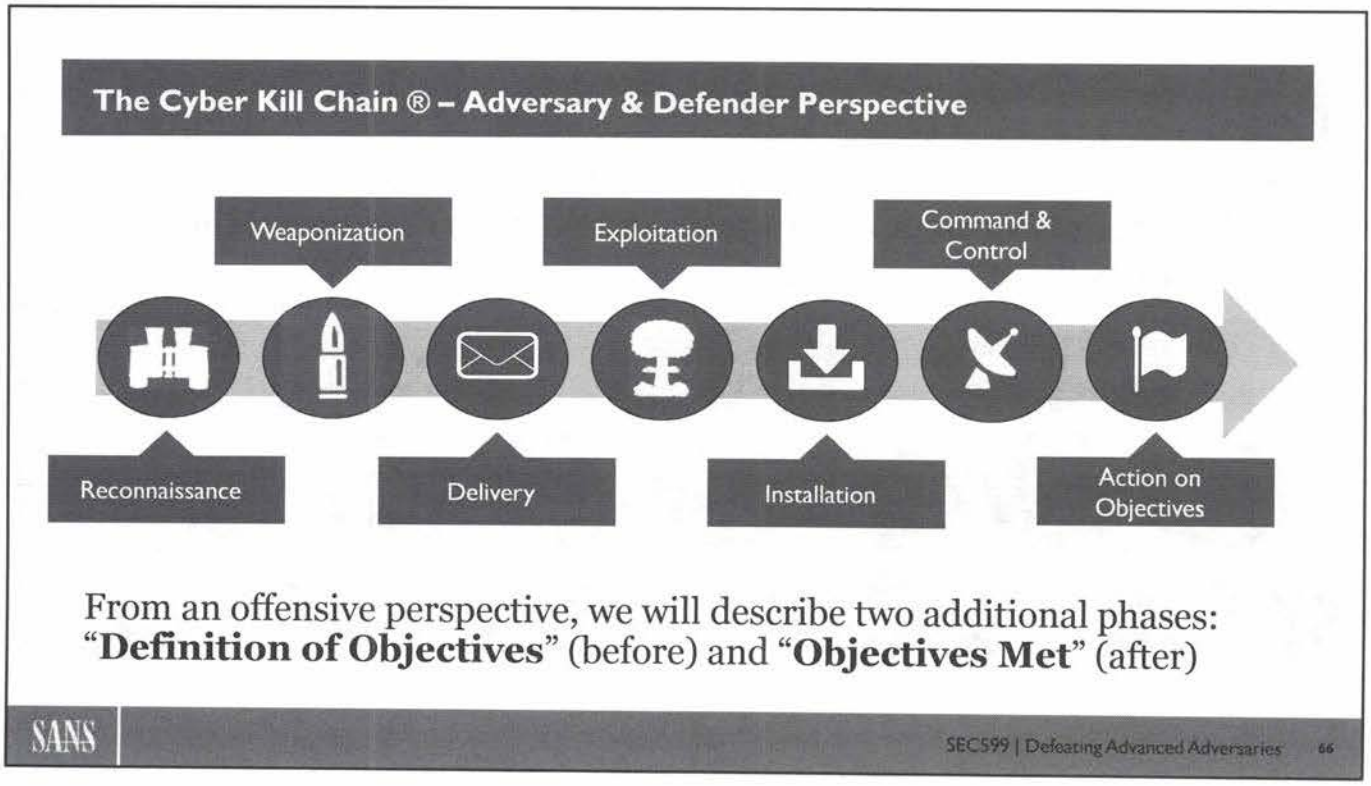
As we are the defenders of digital assets of our company or organization, we face adversaries using digital methods to attack our digital assets. It would be useful to have a digital equivalent of the military kill chain so that we can structure our defenses accordingly.

Different groups and organizations have worked on documenting adversaries' methods in a digital kill chain. Lockheed Martin developed the trademarked "Cyber Kill Chain ®", which has risen in popularity to become one of the most used frameworks to describe cyber attacks. An alternative, slightly adopted variant is Dell SecureWorks' "Cyber Kill Chain". Both chains have more steps than the military kill chain.

Lockheed Martin: Reconnaissance, Weaponization, Delivery, Exploitation, Installation, Command & Control, Actions On Objectives.
Dell SecureWorks: Target Defined, Recon, Development, Weaponization, Delivery, Exploitation, Installation, Command & Control, Actions on Objectives, Objective Met

For the purpose of our course, we will follow a similar structure, as most online publications related to cyber attacks do the same.

References:
http://www.lockheedmartin.com/us/what-we-do/aerospace-defense/cyber/cyber-kill-chain.html
https://www.secureworks.com/resources/wp-breaking-the-kill-chain



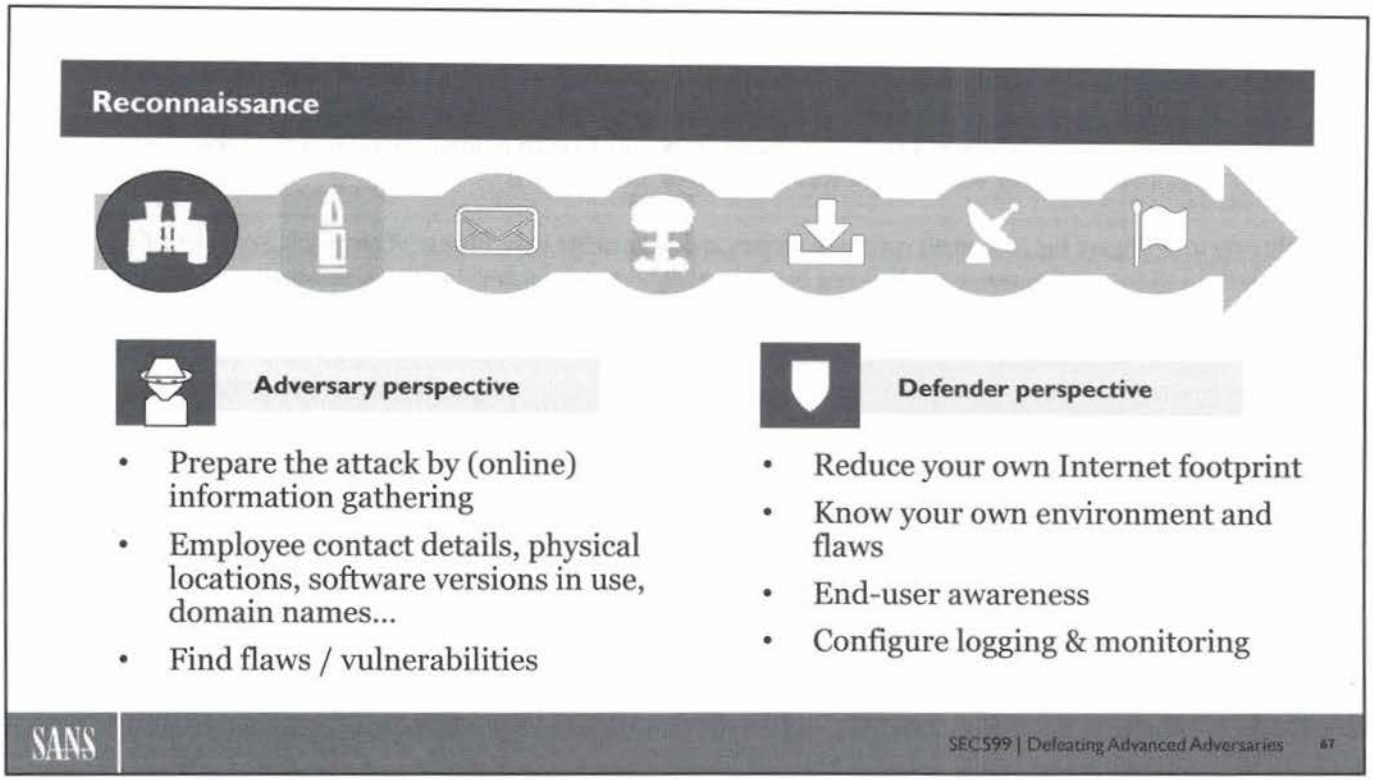
The Cyber Kill Chain ® - Adversary & Defender Perspective
We will now discuss the digital kill chain step by step, both from an adversary and a defender perspective. In a normal, streamlined process without problems, these steps are taken sequentially. But when issues arise, adversaries will revert to previous steps. For example, if exploitation fails, adversaries will return to the weaponization step.

We are convinced that it is important to clearly define the objectives of the attack, and that is why we start with the step "Define Objectives" and end with the step "Objectives Met". It also allows to define a feedback steering loop: when an objective is not met, the attack can start again with modified implementations of the different steps, up to redefinition of the objectives.

Definition of Objectives
Objectives are not the same as targets. As an example, let's assume adversaries want to steal technology to build better fighter jets. They would first list organizations that appear to have this type of information these organizations would become targets), while the objective will be the "fighter jet technology".

In a military kill chain, the goal is often the destruction of the target. This is most often not the case in an attack following the digital kill chain. With the digital kill chain, the objectives are often to obtain data and intellectual property, or to disrupt operations. Disrupting operations often takes the form of a denial of service attack, which is not the same as destructing the target. Coming close to destruction is erasing systems and data, but it is still not the same as physical destruction. Only rarely will digital attacks result in physical destruction, Stuxnet coming obviously to mind as an example.

Objectives Met
Finally, adversaries will evaluate if they have met their objectives through the actions they performed on the targets. If this is the case, they will retreat but might leave their C&C infrastructure and persistent malware in place for future plans. As this increases the chances of discovery of the attack, it is more frequent that all traces of the attack are removed as best as possible: removing malware and persistence mechanisms, dismantling of the C&C infrastructure, and sometimes even altering logs like Windows event logs to hide all traces. In extreme cases, computer systems might be completely wiped to leave no trace behind.



Reconnaissance
Once the objectives are identified, an adversary will start activities by performing "Reconnaissance". The goal of the reconnaissance phase is to collect information about the target to increase the chances of a successful attack.

Adversary perspective
In our digital kill chain, reconnaissance is the process of selecting targets and obtaining information to achieve the objectives. The attack is still in the planning phase. Targets can be all kinds of Internet-facing services owned or used by the target company. The goal of reconnaissance is not limited to assets of the target company. It can for example also include information gathering about the hosting company where the target company hosts its servers.

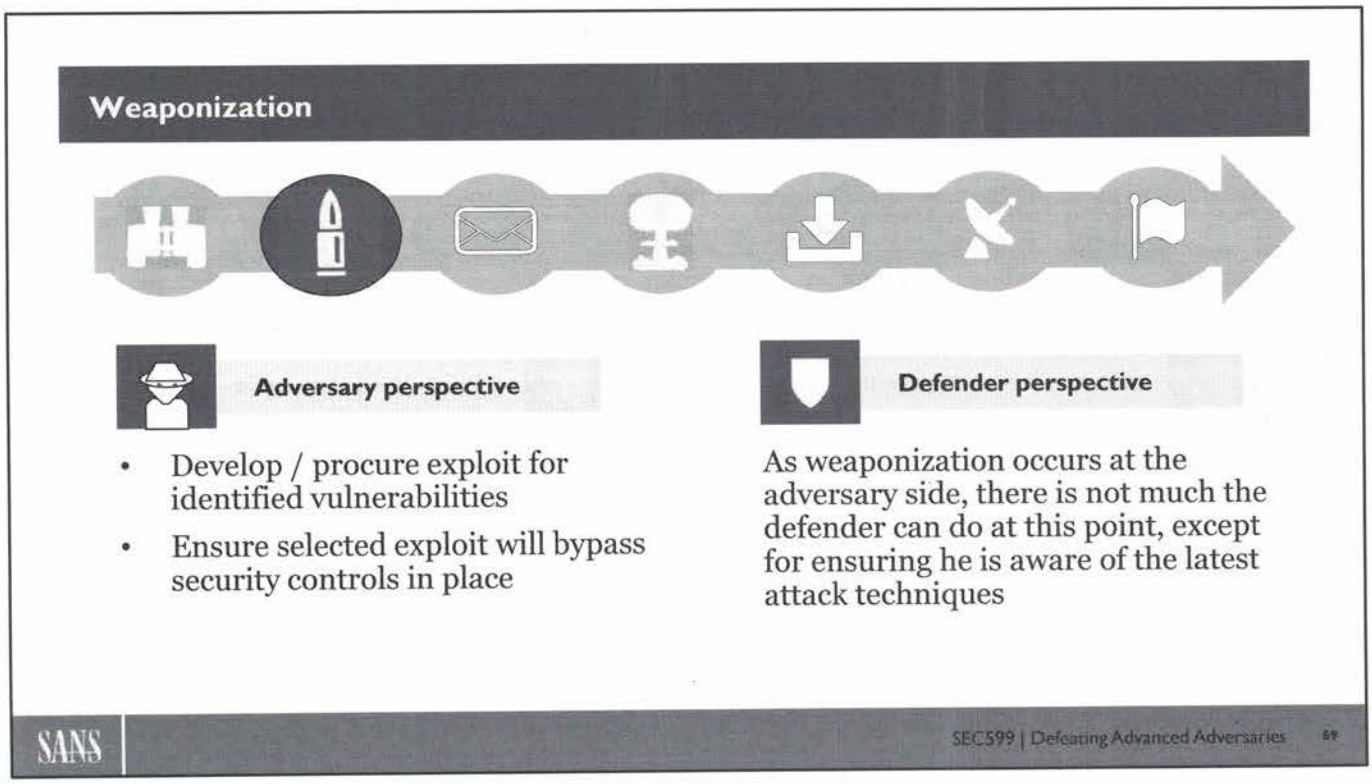
There's a variety of information that could be useful to an attacker:
- Names of employees collected from social media like LinkedIn:
- Email addresses published on various websites;
- Domain names and IP addresses owned by the target company;
- Technology used by the company (e.g. extract software version from metadata, job openings…)
- …

For adversaries, the goal of the reconnaissance phase is to identify flaws or vulnerabilities that can be abused for the attack.

Defender perspective
In this planning phase of the attack, it is extremely hard to detect reconnaissance activities by the adversary. They will scrape the content of your websites looking for information to prepare their attack, but so will (potential) customers at the same time. Distinguishing reconnaissance from simple browsing is not straightforward. It is however highly advised to ensure logging on all Internet-facing systems is well-configured: even if you might not initially detect the reconnaissance activity, they could still be used later for further (possibly post-mortem) analysis.

As a defender, you can, however, reduce the Internet footprint of your company: Investigate what you (and your employees) are exposing online and limit it where required. Furthermore, ensure your employees are aware of the sensitive data they are handling and how they should handle it. An interesting example about this: The US Navy recently re-adopted a slogan to remind everybody to keep information classified: "Loose Tweets Sink Fleets" (the old World War 2 slogan was "Loose Lips Sink Ships").

Be sure to configure logging on all your web servers and to keep the logs. In case of an attack, you might find backtraces of reconnaissance in the logs and better be able to reveal the intent of the adversaries.



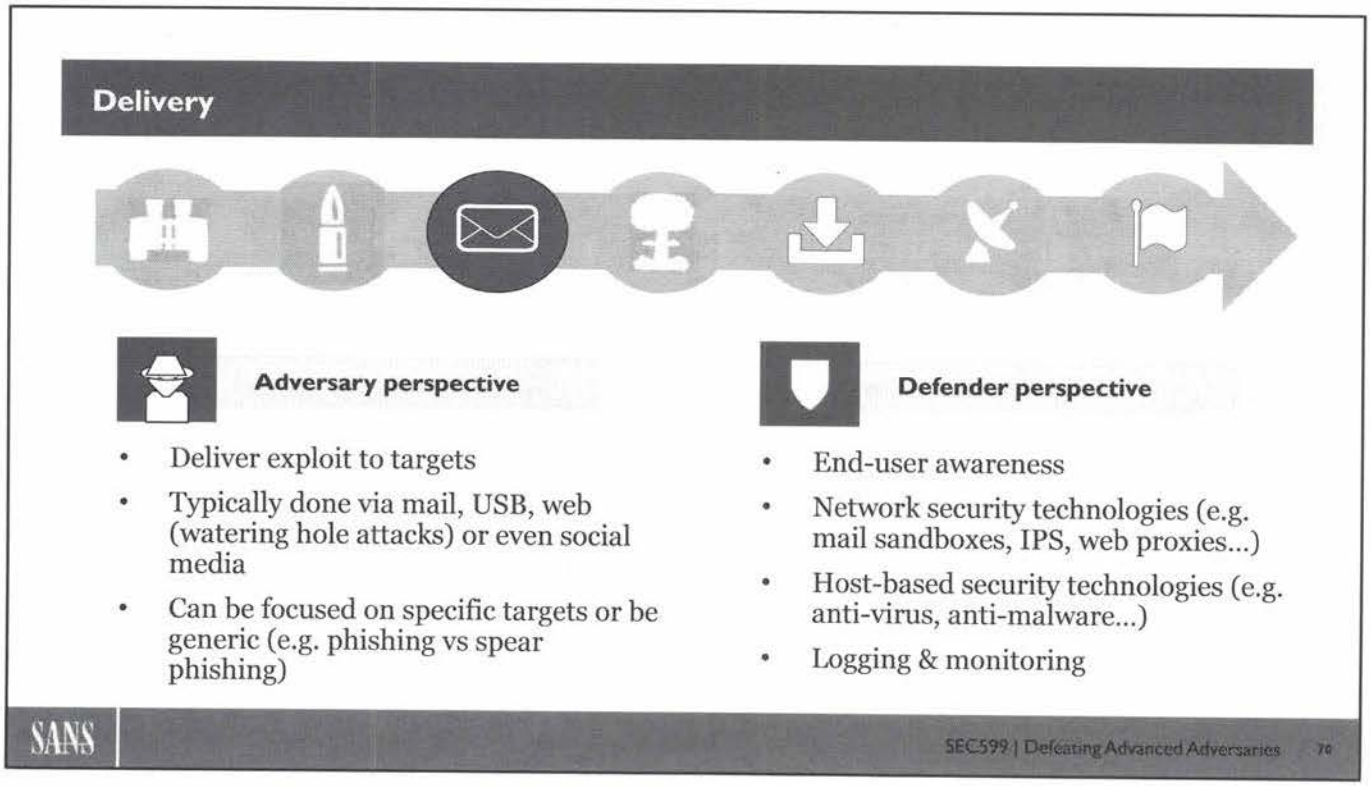
Weaponization
Once reconnaissance is performed, the adversary hopes to have identified a number of (potential) flaws he can leverage.

Adversary perspective
Weaponization is the process of combining an exploit with a suitable vector. An exploit could, for example, be the development of an Office document with malicious macros included. A vector is a delivery mechanism. It can be as simple as attaching the malicious document to an email and sending this to a target company employee's mailbox (note: actual sending of the email would be delivery). The executable is the malware and the email is the vector. Some other examples of weaponization:

- Generating a stand-alone malicious payload as executable (and luring the target in the next phase to execute it);
- Development of a custom exploit for a newly identified vulnerability in software used by the targets;

It's important to note that the more successful the reconnaissance phase was, the more tailored the weaponization phase can be (and thus: the higher the chance for a successful attack). If the adversary knows the target employees are using a specific version of Microsoft Office, he could tailor his attack. Another good example would be knowledge of the installed Anti-Virus software.

Defender perspective
The weaponization phase occurs at the attacker's side and is part of his attack planning activities. As long as the exploit is not delivered, there is not much the defender can do. On a continuous basis, the defender should, however, ensure he knows how attack techniques are evolving, so the eventual "attack" does not come as a total surprise.



Delivery
During the delivery phase, the malicious payload that was prepared in the weaponization phase is delivered to the target.

Adversary perspective
With this step, the attack leaves the planning phase to enter the execution phase. The payload created in the previous step must be delivered to the victim(s) selected in the reconnaissance phase. This delivery can be done through various vectors:

- Sending emails to the victims with malicious payloads (or links to download the payload);
- Interact via social media like Facebook or Twitter and send malicious links to the victims;
- Copy the malicious payload to removable media such as USB sticks, and deliver the media to the victims. This can be delivered via regular mail or courier, or more surreptitiously by dropping some USB sticks were the victims tend to gather, like a staff parking lot or near vending machine;
- Another interesting mechanism is the "watering hole" attack. In a watering hole attack, the adversaries will first compromise other, unrelated, websites that tend to be visited by the victims.

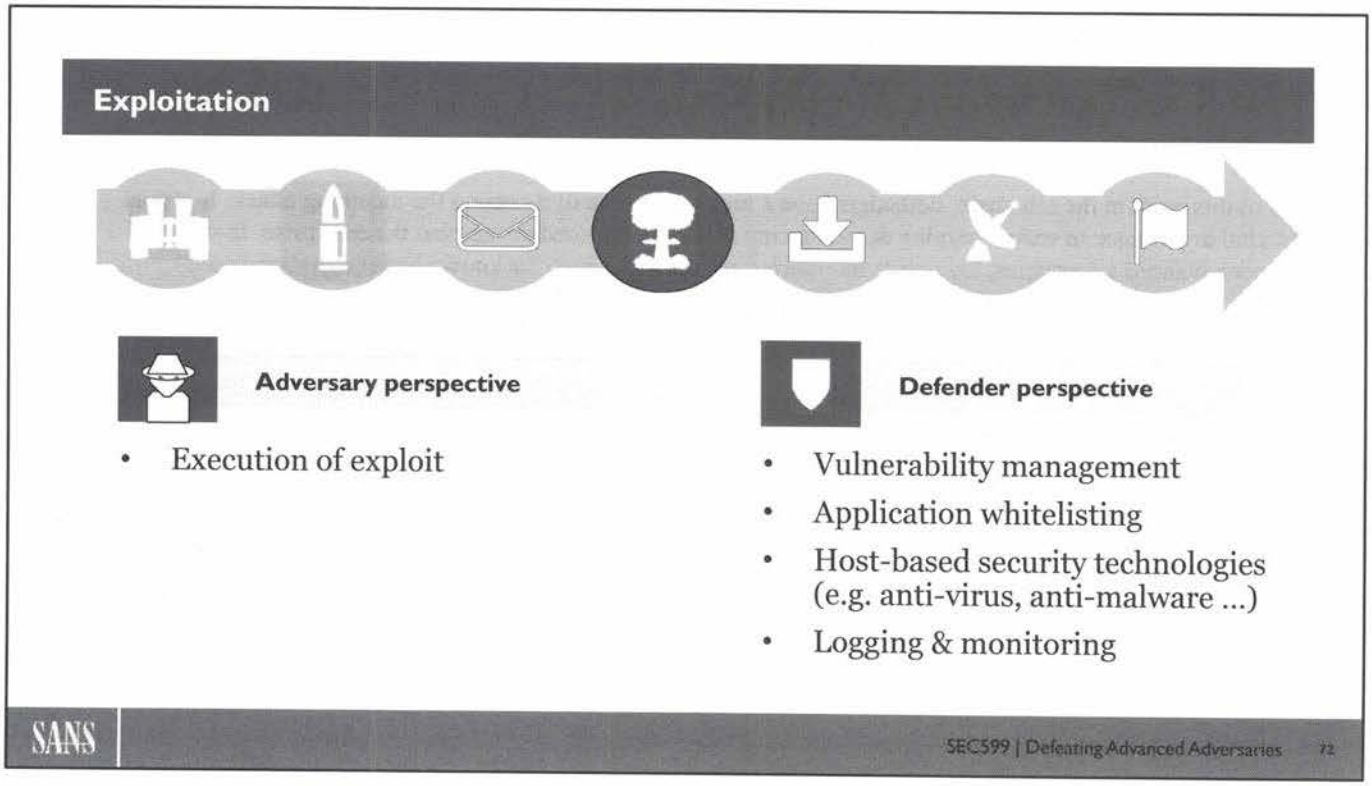
Defender perspective
Being able to detect attacks this early in the digital kill chain is a key capability for defenders: the earlier we can detect adversaries in the kill chain, the less they will be able to reach their objectives. End-user awareness is a key security control here: if people understand how advanced adversaries operate, they can be the first layer of defense. Next to end-user awareness, there's also a number of technical controls that can be implemented:

Network security technologies such as mail sandboxes, IPS engines or web proxies.
- Mail sandboxes will allow us to investigate incoming email and block malicious attachments or URLs;
- IPS engines can block known attack signatures at network level;
- Web proxies can be used to block access to suspicious / malicious websites.

Host-based security technologies such as Anti-Virus or Anti-Malware software;

- Anti-Virus engines to detect the low-hanging fruit and generic payloads;
- Anti-Malware technology (typically agents) that can analyze the system for suspicious application behavior.

As of this point in the kill chain, defenders have a realistic chance of detecting the incoming attack. It is thus of vital importance to ensure logging & monitoring is well-configured throughout the enterprise. In order to detect advanced adversaries, logging & monitoring should focus both for known bads (signature-based, IOCs…) and unknown bads (behavior-based, TTPs…). We will further discuss this as we proceed through the course.



Exploitation
Upon successful delivery of the payload, exploitation is performed:

Adversary perspective
After the delivery, the malware will be executed on devices used by the victims. When a user received a payload and opens / runs it, it will run with the privileges of that user. Depending on the goals of the adversaries, running the executable with the user's privileges might be sufficient or not (we assume that the target user is a least privilege user, e.g. not a Windows administrator). If higher privileges are needed, the adversaries will need to use exploits to gain privilege escalation. This can be as simple as a User Account Control bypass, or as sophisticated as a zero-day to achieve code execution in the Windows kernel.

Adversaries often use a combination of technical and people exploits. Consider the example of malicious macro in an office document: it will not run as long as the human doesn't open the document (and enables macros, if required). Next to generating the malicious payload, the adversary will still need to convince the target to open / run it.

Defender perspective
As defenders, we have two opportunities during this step to combat adversaries: prevent exploitation and detect exploitation.

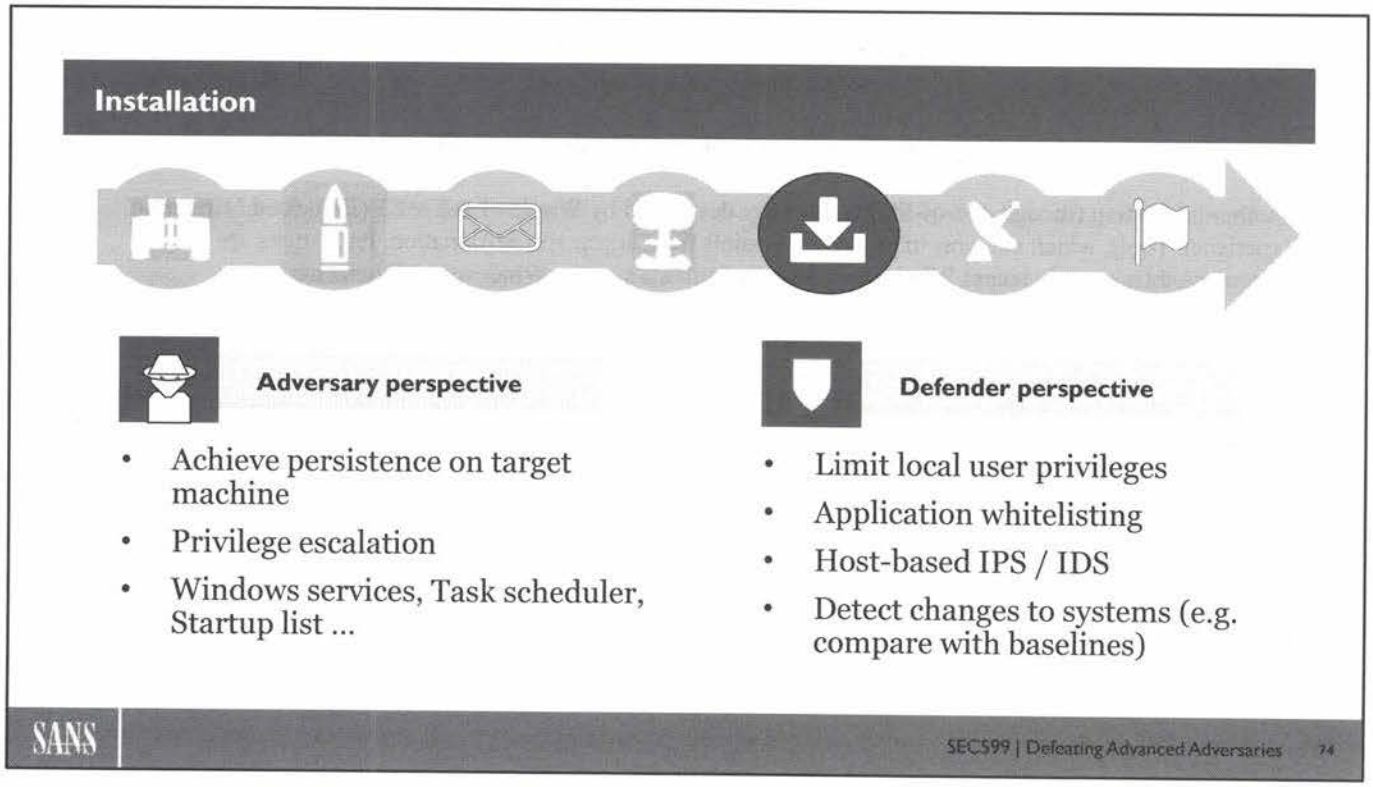
As part of vulnerability management, we should:

- Harden our systems according to a baseline configuration (shut down unneeded services, change default credentials…). NIST is well-known for producing quality hardening guidelines;
- Ensure patches for third-party software are timely installed;
- Ensure the software we develop ourselves is developed according to secure development standards;
- Ensure your entire IT environment (including infrastructure and applications) is regularly assessed for vulnerabilities (and that identified flaws are fixed);
- Have a formal vulnerability management process in place, where vulnerabilities have to be formally accepted if they are not mitigated.

Application whitelisting can be of great help to prevent arbitrary code execution, but it requires a standardized software environment. If users are expected to install and run their own software according to their business needs, then application whitelisting will be extremely hard to successfully implement. Windows provides free-to-use application whitelisting technology via AppLocker and Software Restriction Policies.

Another interesting (though end-of-life) technology developed by Windows is EMET (Enhanced Mitigation Experience Tool), which attempts to protect end-points from successful exploitation. Its features are implemented (in part) in recent Windows Operating Systems such as Windows 10 Enterprise.

Monitoring all code execution on a system is not only a great way to detect exploitation, but it also provides logs that can be used in forensic investigations of successful attacks. Microsoft's Sysmon is a free tool that collects information of code execution and writes it to a Windows event log. For example, Sysmon can create Windows event log entries for executed programs, including a cryptographic hash of the executable image.



Installation

If exploitation was successful, the adversary will have an initial payload running on the target system. His goal will now be to persist his access to the system (e.g. so the target remains compromised after a reboot). Note that not all attacks require persistence. Ransomware attacks for example typically don't require persistence: once the documents are encrypted and the user informed, the ransomware has no need for further execution.

Adversary perspective

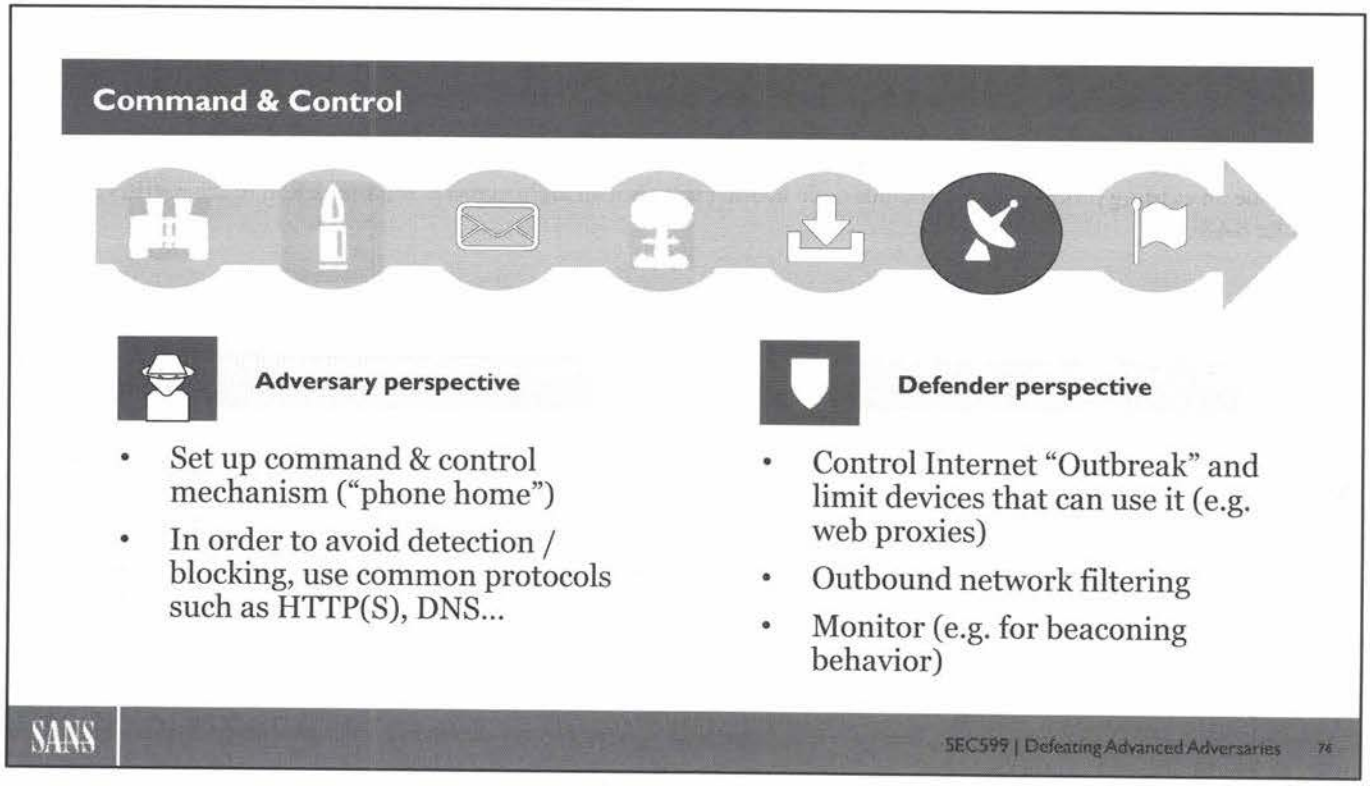
During the installation step of the attack cycle, adversaries try to achieve persistence on the target machines (if it is part of the goal). When persistence is required, changes must be made to the configuration or software of the machine to achieve persistence. This is necessary because typical computer operating systems like Windows only run started programs as long as the user is logged on. If the user logs off (or the machine is restarted), running programs are terminated. To restart the malicious programs automatically when a user logs on again, persistence mechanisms must be used.

Windows has a large and diverse set of "autorun" options that can be used for persistence. This can be done in the context of a user, so that persistence is achieved only when the same user logs in again, or in the context of a machine, so that persistence is achieved when the machine is started. Persistence can be as simple as a Start entry in the user's Windows menu configured to run the malicious payload again or as complex as a dedicated backdoor running as a service or even installed in the firmware of the computer. Webshells are typical backdoors left behind on compromised web servers.

Defender perspective

To achieve persistence on a target system, adversaries must make changes to the configuration of said systems. Not only can many of these changes be prevented by hardening, but they can also be detected by monitoring applications like Microsoft's Sysmon. In homogenous environments, configuration baselining can help detect this type of changes. Some host-based Intrusion Prevention Systems monitor autorun configurations too and alert on any changes made to them.

The Windows operating system provides an abundant set of configuration options that can be used to achieve persistence. It should be noted that achieving persistence does not necessarily require the malware to be stored in files on the file system. So-called "fileless" malware can achieve persistence by storing commands inside autorun entries in the registry. When executed at startup or login, these commands will inject malicious code inside an existing process. The malicious code is often stored in an alphanumeric representation in the registry, like BASE64.



Command & Control

Programming malware to perform all malicious actions automatically and autonomously can be quite challenging for adversaries, especially when they have incomplete information about their targets. Often the adversaries will want the malware to have capabilities to be controlled remotely so that they can instruct the malware with the appropriate actions to take. Adversaries attempt to keep this control by the use of a Command & Control infrastructure and channel.

Adversary perspective

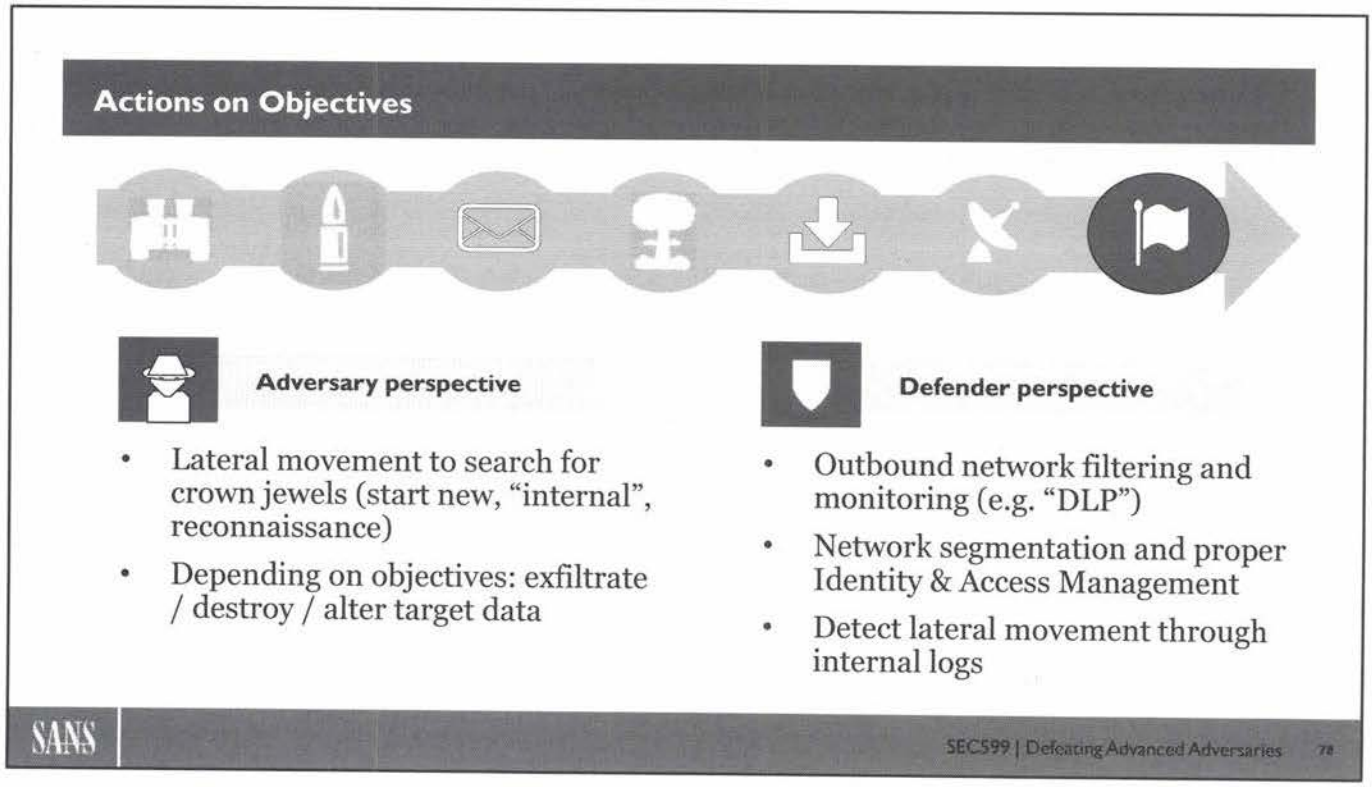
During weaponization, adversaries will already decide on the communication channel to be used, as it will be built into the malicious payload. In order to avoid detection and to increase the chances of the outbound connectivity being allowed, adversaries will select a commonly used protocol such as HTTP(S), DNS, e-mail or even social media. Cases have also been identified where custom TCP protocols were developed. The endpoint of communication channels are called Command & Control servers; these are servers under the control of the adversaries, but not necessarily owned by the adversaries. In targeted attacks, adversaries might first compromise other systems and use these as Command & Control servers.

A varying degree of stealth can be built into these C&C channels, for example masquerading the communication as an HTTP connection with a music streaming service. Another example we've seen is the use of steganography in pictures. This doesn't necessarily imply that the communication is complex. For ransomware, for example, the C&C infrastructure is often only used to report back the encryption keys.

Defender perspective

Communication between the malware on target systems and adversaries is a good opportunity for us to stop the attack, provided we detect and block it almost instantaneously. We can limit network communications from our network to the Internet via control points like proxies and firewall. In an open network where internal clients have full access to the Internet via a NAT gateway, controlling, filtering and monitoring is exponentially harder than in a network where all communications go through filtering devices. Of course, an open network can be a business requirement and something you have to live with.

Proxies not only allow us to block and filter traffic, but it also gives us the opportunity to log and inspect the traffic for patterns or anomalies (e.g. beaconing behavior). Beaconing behavior is when malware periodically attempts to connect back to its Command & Control server. If this is done using a fixed time interval, it could form a pattern we can attempt to detect.



Actions on Objectives
Once the adversary has managed to persist malware on an initial system, he can now gear up to start working on his actual objectives.

Adversary perspective
Once the attack reaches this step of the kill chain, everything has been put into place to enable the adversaries to perform the necessary actions on the targets to reach their objectives. The actions they will take depend on these objectives, and can be all sort.

For example, once they have a foothold in the infrastructure, adversaries can start a new digital kill chain: they start with reconnaissance of the internal network to identify interesting targets to attack. This will typically be followed by lateral movement. Lateral movement is the term used to indicate that adversaries are spreading in the network, moving from computer to computer. Once inside, lateral movement is often facilitated by the "openness" of the internal network (so-called "egg-shell" problem). Old school design of a secure network puts many of the security controls at the perimeter of the network, and not inside the network. Once adversaries penetrated the perimeter without detection, they encounter fewer obstacles to move inside the internal network.

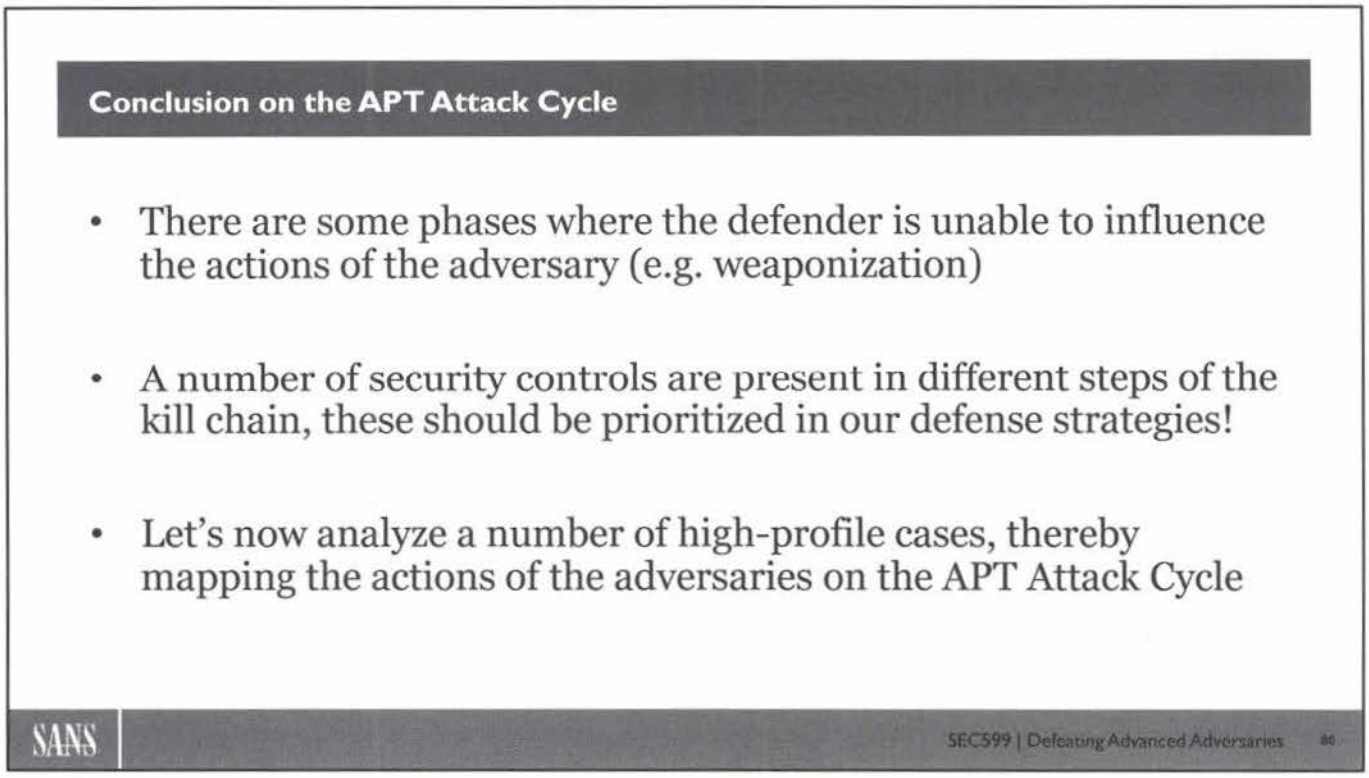
When adversaries reach their targets through lateral movement, they will "finalize the kill". If the objective is espionage, they will collect and exfiltrate data. If the objective is to interfere with the target, they will start making modifications. This can be corrupting, deleting or overwriting of data and systems, or covertly modify data and configurations to change operations within the target. For example, data modifications can be introduced in payment systems to steal money by wire transfer. We have even observed malware samples that modify payroll data on cloud systems to introduce new, fake, employees in the staff database and have their wages paid into bank accounts owned by criminals or their money mules.

Defender perspective
When adversaries progress this far in the kill chain, they have defeated the majority of previous defenses. For the adversary, everything is in place for the final push.

Depending on the objectives and the complexity of the attack, there might be a lot of activity required from the adversary, which could give us more opportunities to detect or block the attack. During this step, which can be the longest in absolute time of all steps in the digital kill chain, adversaries typical perform following actions (not all during the same attack):

Lateral movement: adversaries do reconnaissance of the internal network and move from system to system to reach the systems they target. Lateral movement can generate a lot of evidence, offering opportunities to defenders for detection. To be able to detect lateral movement, appropriate controls inside the internal network need to be put in place, such as firewalls and intrusion detection system between different segments of the internal network. Furthermore, adversaries could attempt to reuse previously compromised credentials, which could raise suspicion. For example, if user A is currently working in the New York office and a login is detected for user A via VPN from a location in Turkey, something unusual is going on and should be further investigated.

Data exfiltration: when the objective is to obtain information, it has to be transferred to the adversaries' systems once it is located and accessed. Exfiltration of data is typically a network activity, and as such, leaves traces. Large amounts of data exfiltration (gigabytes or even terabytes) are detectable by graphing the consumed network bandwidth versus a time axis. Dedicated system can be put in place to monitor for data exfiltration: Data Loss Prevention systems. DLP can be as simple as looking for tags on the network, such as the string "strictly confidential" in uploaded documents. But such simple detections are also simple to bypass. For example, just compressing or encrypting a document before uploading hides all strings inside the document.



Conclusions on the APT Attack Cycle
Throughout the APT Attack Cycle, several phases exist where the defender is unable to interfere or influence the adversary's actions. For example, the actions taken during the "preparation" or "planning" (e.g. definition of objectives, weaponization) of the attack offer limited opportunities to disrupt the attack. Once the payload / exploit is delivered, the defensive options increase and a broad range of controls can be implemented.

Furthermore, a number of security controls are present in different steps of the kill chain:

- End-user awareness;
- Logging & monitoring;
- Application whitelisting;
- Host-based security technologies; (antivirus, antimalware…)
- Network-based security technologies (IPS, Proxies, Mail gateways…)

These should take priority when we are assessing our defensive options! Let's now analyze a number of high-profile cases, thereby mapping the actions of the adversaries on the APT Attack Cycle.

APT Attack Cycle – Additional Resources

Some additional resources that can prove to be useful for the APT attack cycle include:

- https://www.sans.org/reading-room/whitepapers/analyst/killing-advanced-threats-tracks-intelligent-approach-attack-prevention-35302
  Whitepaper on attack prevention
- http://blog.airbuscybersecurity.com/?q=apt+kill+chain
  Blog series on the APT kill chain
- http://www.darkreading.com/attacks-breaches/leveraging-the-kill-chain-for-awesome/a/d-id/1317810
  Some more reasons to use the cyber kill chain

APT Attack Cycle – Additional Resources
Some additional resources that can prove to be useful for the APT attack cycle include:

https://www.sans.org/reading-room/whitepapers/analyst/killing-advanced-threats-tracks-intelligent-approach-attack-prevention-35302
Whitepaper on attack prevention

http://blog.airbuscybersecurity.com/?q=apt+kill+chain
Blog series on the APT kill chain

http://www.darkreading.com/attacks-breaches/leveraging-the-kill-chain-for-awesome/a/d-id/1317810
Some more reasons to use the cyber kill chain

Course Roadmap

- **Day 1: Knowing the adversary, knowing yourself**
- Day 2: Averting Payload Delivery
- Day 3: Preventing Exploitation
- Day 4: Avoiding Installation, foiling Command & Control & thwarting lateral movement
- Day 5: Exfiltration, Cyber Deception & Incident Response
- Day 6: APT Defender Capstone

SEC599.1

Course Outline & Lab Setup
- Course Overview & Objectives
- Attendee System Setup

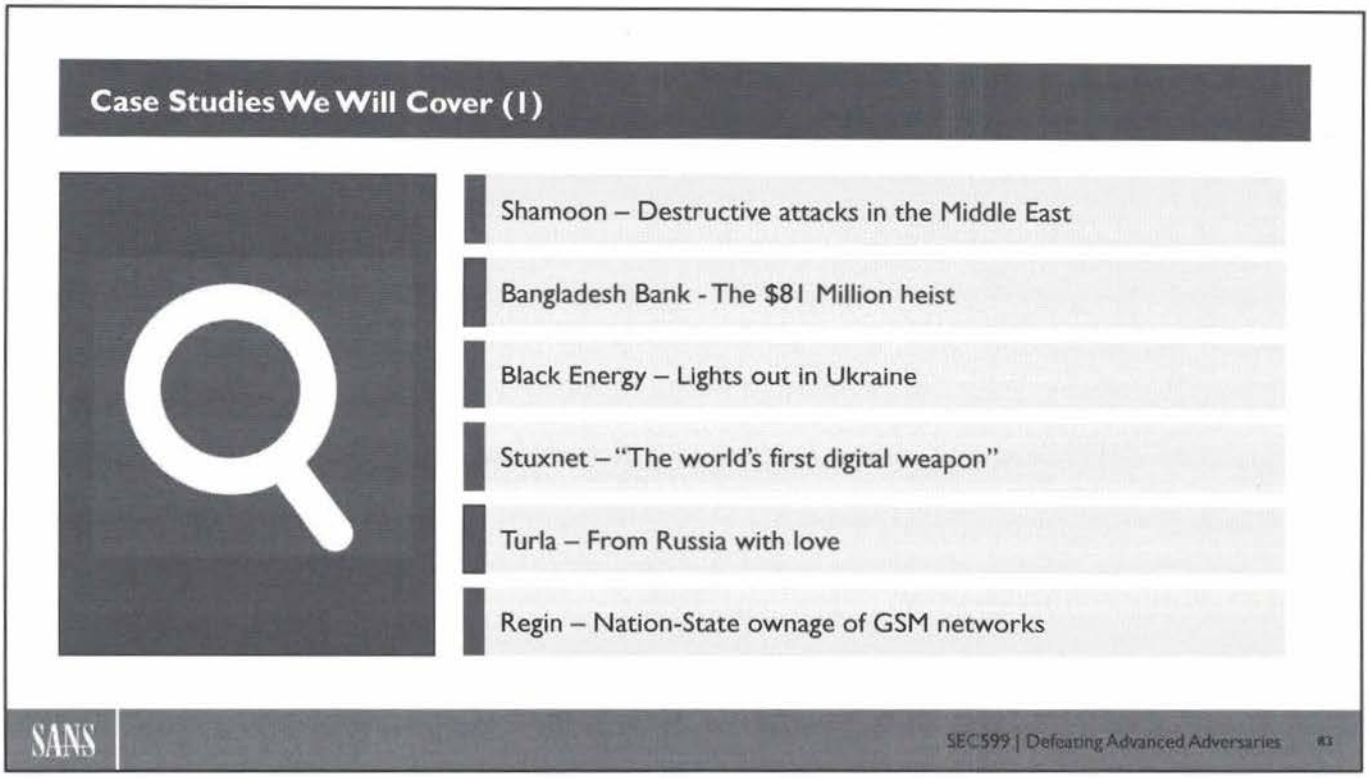
Current Threat / Attack Landscape
- Key Terminology
- What is happening out there?

Introducing the APT Attack Cycle
- Recent Case Studies – In-Depth
- Exercise: Analyzing The Behavior of Famous Malware
- Exercise: One Click Is All It Takes…

A Defensible Architecture & Environment

Preparation - Knowing Yourself
- Understanding Your Own Environment
- Determining What is "Normal"
- Understanding & Limiting Your Internet Footprint
- A Word on Vulnerability Management

This page intentionally left blank.



Case Studies We Will Cover (1)
In order to provide an insight in how advanced adversaries work, we've selected six interesting cases we will explore:

Shamoon
Shamoon is a virus that was first observed in 2012, attacking Middle Eastern oil companies. An espionage tool, it is used for the extraction of interesting data, but can also wipe data for sabotage purposes. Shamoon made a "comeback" in December 2016.

Bangladesh Bank - The $81 Million Heist
Malware was used to send counterfeited SWIFT messages to the Federal Reserve Bank of New York to transfer $81 million to the criminals' bank accounts in the Philippines, who quickly laundered the money through local casinos.

BlackEnergy
BlackEnergy is malware that was used to cause power outages in Ukraine in December 2015. By taking control over SCADA systems, adversaries were able to switch off substations of the power grid.

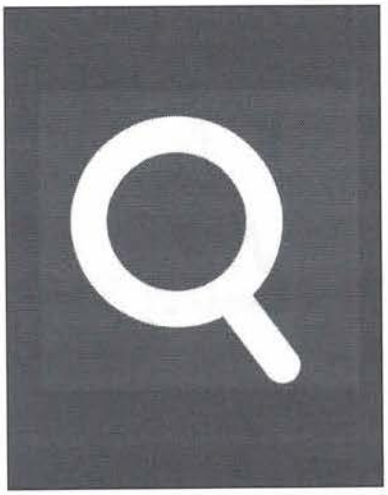
Stuxnet
Stuxnet is complex malware that was first detected in 2010. It attacks SCADA systems and disrupted Iran's nuclear program. This nuclear program included facilities for uranium enrichment with centrifuges, controlled by programmable logic controllers. Stuxnet reprogrammed these PLCs in order to damage the infrastructure.

Regin
Regin is a highly advanced malware that was first revealed "to the masses" in 2014, as it was used during a large-scale attack against Belgium's largest ISP Belgacom (now called Proximus). The goal of "Operation Socialist" was to obtain access to Belgacom's subsidiary BICS (Belgacom International Carrier Services), one of the world's largest roaming hubs, in order to intercept GSM communications.

Epic Turla

Epic Turla is espionage malware that was used to spy on diplomatic services. Infections were discovered at government institutions in Belgium, Ukraine, China, Jordan, Greece, Kazakhstan, Armenia, Poland, and Germany.

Case Studies We Will Cover (2)



Shamoon – Destructive attacks in the Middle East

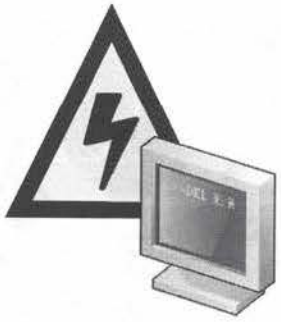
- Bangladesh Bank - The $81 Million heist
- Black Energy – Lights out in Ukraine
- Stuxnet – "The world's first digital weapon"
- Turla – From Russia with love
- Regin – Nation state ownage of GSM networks

This page intentionally left blank.

Shamoon – Destructive Attacks in the Middle East

An interesting example of corporate espionage is **Shamoon**:

- Initially identified in 2012, "comeback" in December 2016
- Mainly targets energy & oil companies in the Middle East
- Virus exfiltrates data and afterwards erases it, but had the potential to do much more
- Once a predefined date is reached, the Master Boot Record is overwritten
- Attack attributed to group called "Cutting Sword of Justice"
- Name derived from left-over PDB path in the Wiper Module



```
C:\Shamoon\ArabianGulf\wiper\release\wiper.pdb
```

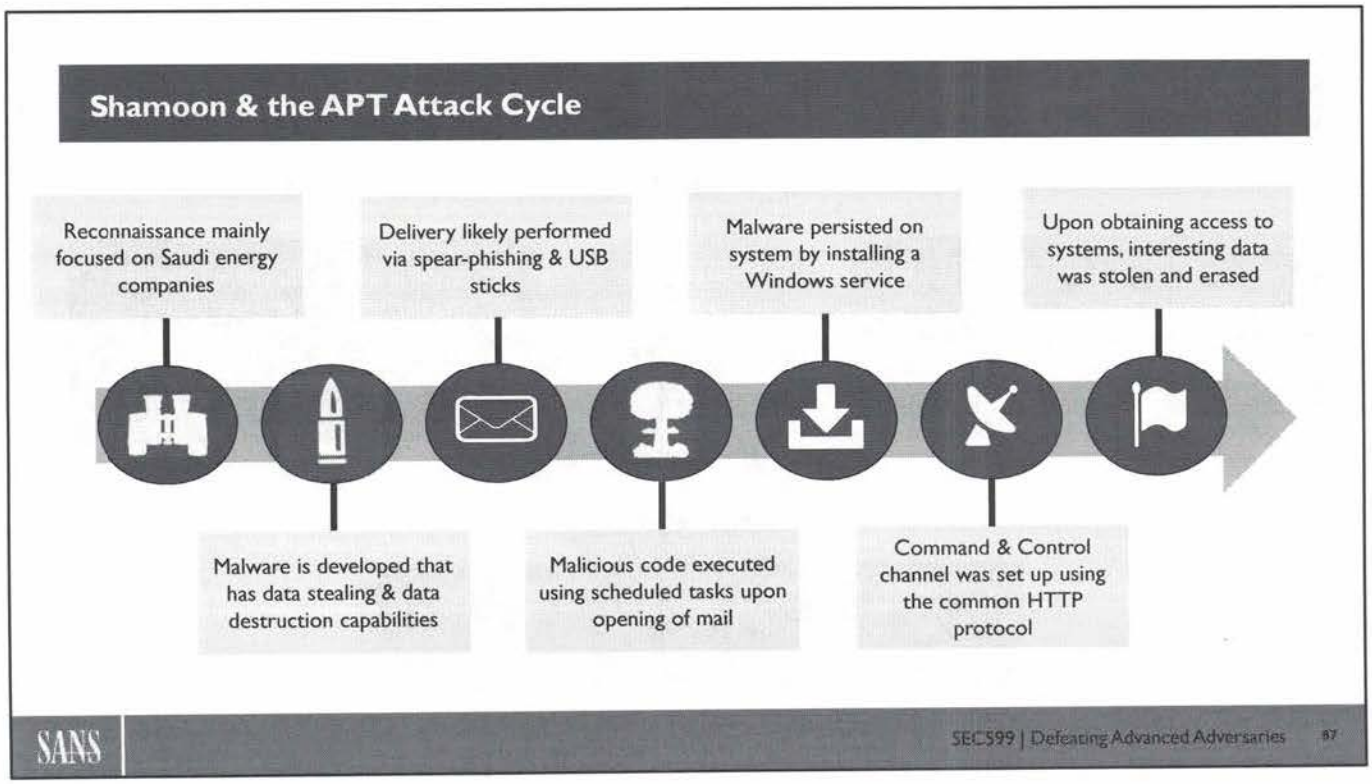

Shamoon – Destructive Attacks in the Middle East

Shamoon (sometimes also referred to Disttrack) was first discovered in 2012. It targets NT-kernel based versions of the Windows operating system and has mainly been used in cyber espionage / attacks against energy & oil companies in the Middle East. The virus operations in the following way:

- The initial intrusion / infection varies but uses commonly used techniques (e.g. spear-phishing, infected USBs…)
- Once infected, it compiles a list of interesting files which are uploaded to the adversary;
- The target files are overwritten with either random data or a pre-defined picture;
- Once a certain predefined date is reached, the virus will overwrite the Master Boot Record, rendering the system unbootable.

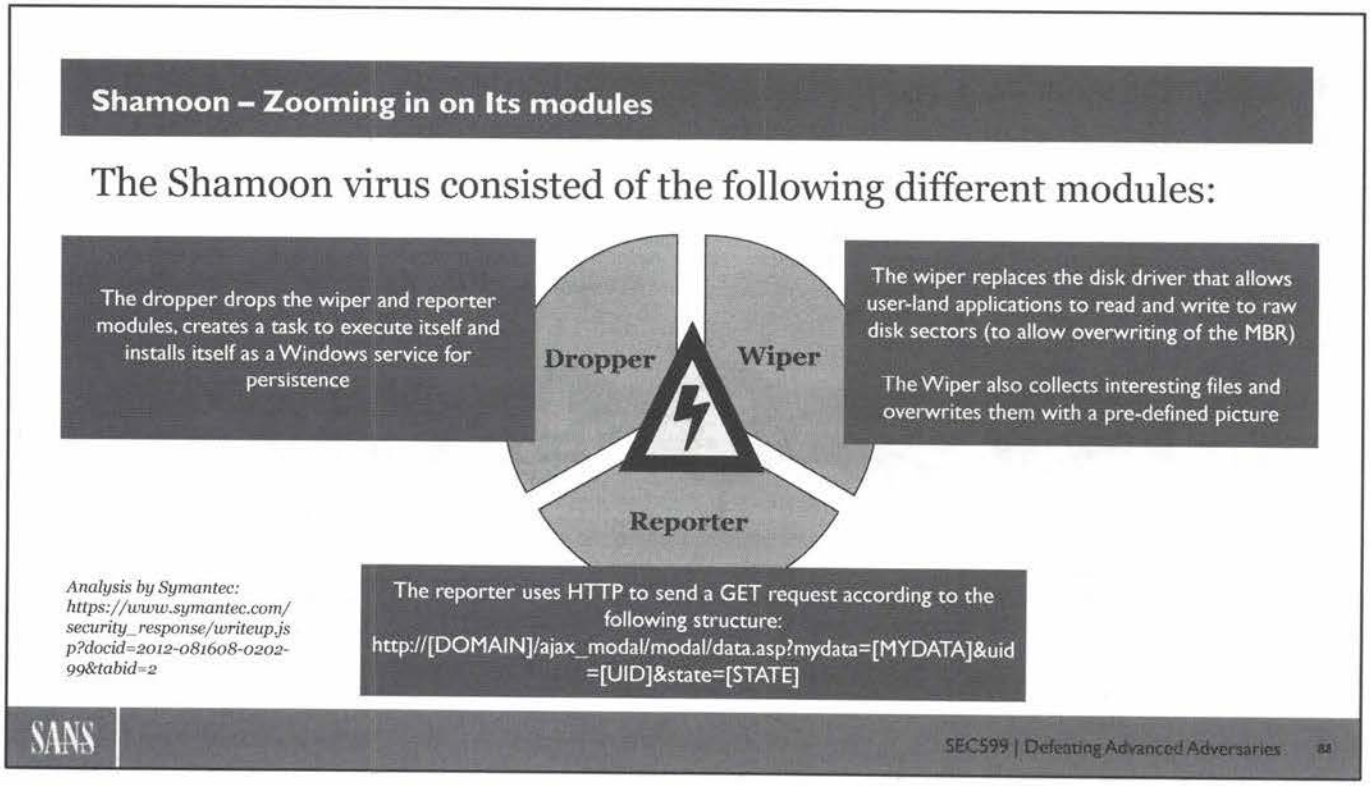
Shamoon has capabilities that allow it to spread from one infected machine to another.

References:
https://www.symantec.com/connect/blogs/shamoon-attacks-continue



Shamoon & the APT Attack Cycle
Let's first analyze Shamoon by using the APT attack cycle:

- Reconnaissance: Most likely, the adversaries performed some initial reconnaissance;
- Weaponization: A Windows executable was created (cfr. PDB-path identified during reverse engineering);
- Delivery: The malware was most likely delivered by spear-phishing and USB-sticks;
- Exploitation: Malicious code was executed using scheduled tasks;
- Installation: A Windows service was installed to make the malware persistent;
- Command & Control: The "Reporter" module used a standard HTTP command & control channel;
- Action on Objectives: Specific data was stolen and uploaded through the Reporter. Furthermore, data was erased using the Wiper module.



Shamoon – Zooming in on Its Modules
Shamoon consists of three different modules:

Dropper (Located in %System%\trksvr.exe)
The dropper is the initial infection which will drop the wiper and the reporter modules. Furthermore, it will create a task to execute itself. As a persistence mechanism, it will install a Windows service with the following details:

Service name: TrkSvr
Display name: Distributed Link Tracking Server
Image path: %System%\trksvr.exe

Wiper (Located in %System%\<NAME_FROM_LIST>.exe)
The wiper is installed in the %System% directory, using a name that is randomly selected from a list of possible names. The names appear to be legitimate files (e.g. dnslookup.exe, event.exe, drag.exe, msinit.exe…)

It is one of the core features of the virus, as it is responsible to perform the deletion of files and, eventually, the overwriting of the Master Boot Record. Note that it deletes an existing driver from the following location and overwrites it with another legitimate driver (digitally signed!):

%System%\drivers\drdisk.sys

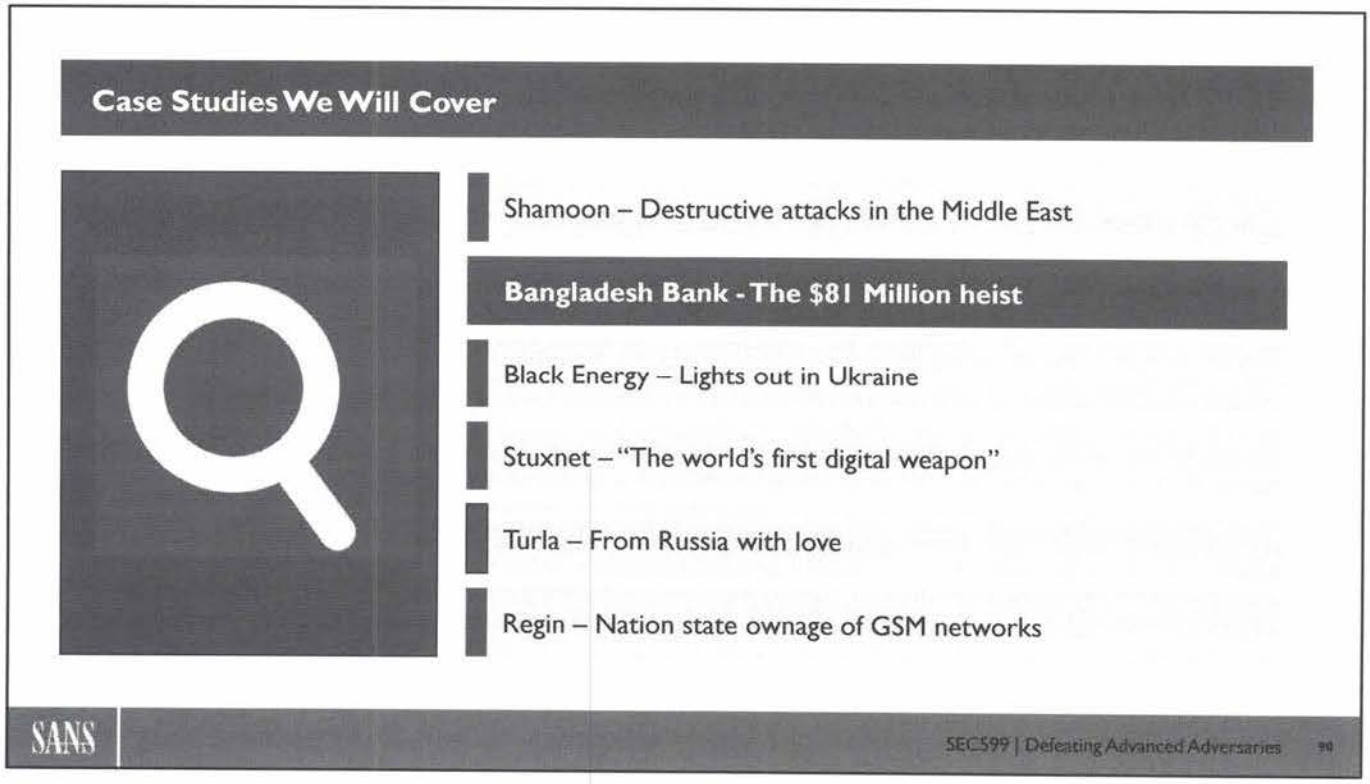
Reporter (Located in %System%\netinit.exe)
This component sends infection information (debugging) back to the adversary. The information is sent as a HTTP GET request with the following structure:

http://[DOMAIN]/ajax_modal/modal/data.asp?mydata=[MYDATA]&uid=[UID]&state=[STATE]

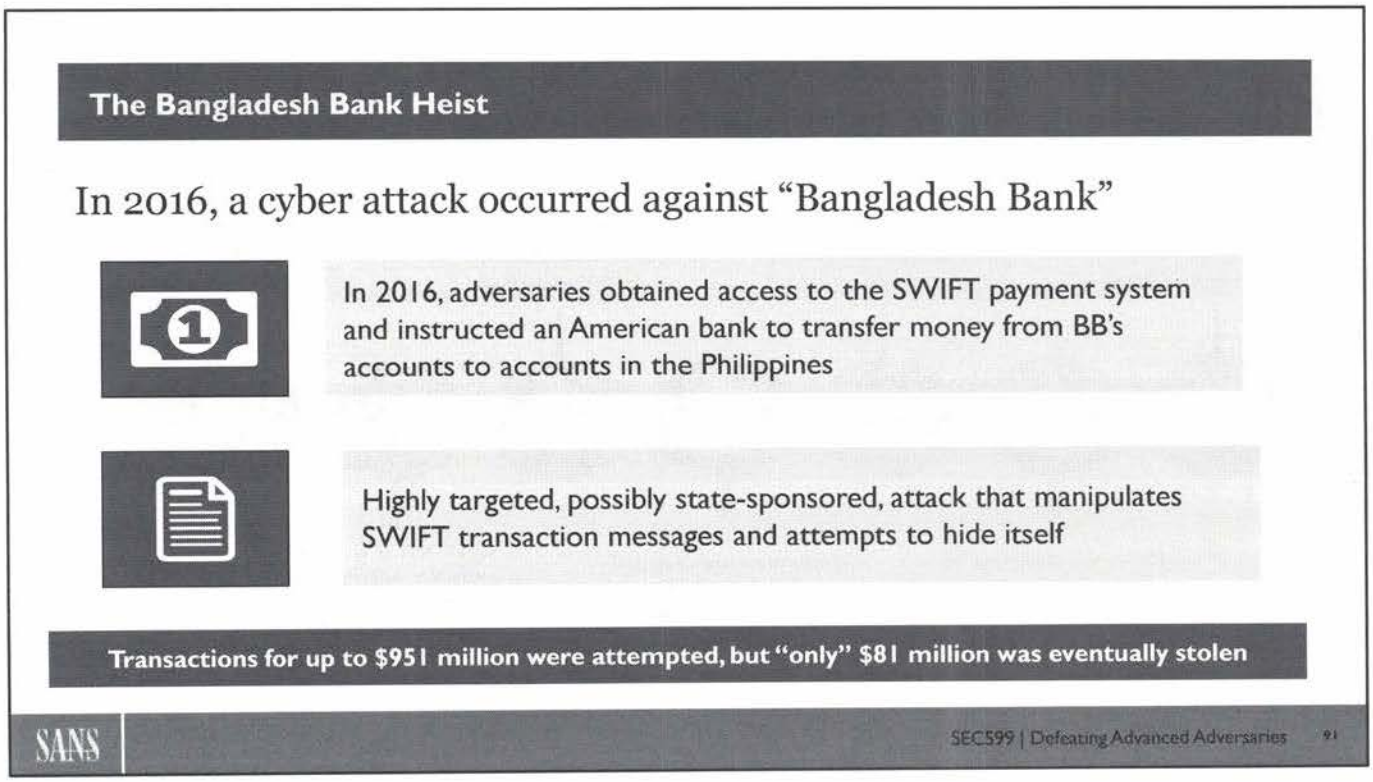
DOMAIN]—a domain name
[MYDATA]—a number that specifies how many files were overwritten
[UID]—the IP address of the compromised computer
[STATE]—a random number

For further information, please refer to the below write-up by Symantec:

https://www.symantec.com/security_response/writeup.jsp?docid=2012-081608-0202-99&tabid=2



This page intentionally left blank.



The Bangladesh Bank Heist

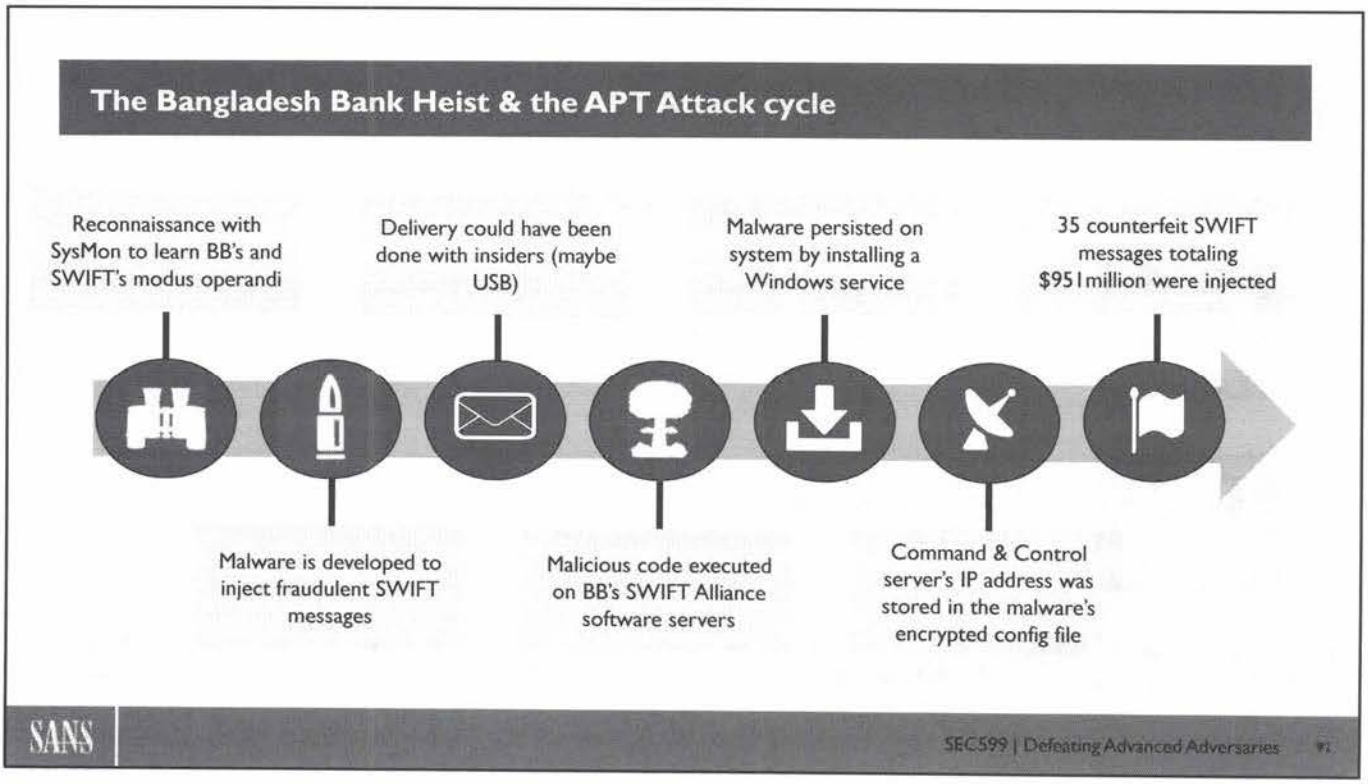
The Bangladesh bank heist of 2016 is a notorious digital attack via the SWIFT network on the Bangladesh Bank account at the Federal Reserve Bank of New York.

Adversaries outside of Bangladesh used the Dridex malware to compromise computer systems of the Bangladesh Bank, possibly with the help of insiders. Dridex gave them the capabilities to observe the operations of the bank regarding international payments and money transfers. Adversaries install SysMon on SWIFT systems as reconnaissance tool, helping them to understand how the SWIFT network operates and how the bank employees operated the SWIFT network to execute financial transactions.

The installed malware would manipulate SWIFT messages through PRT files and Printer Command Language, allowing the adversaries to generate 35 fraudulent SWIFT messages for a total of 951 million USD. The Federal Reserve Bank of New York blocked 30 suspicious SWIFT transactions, but let 5 of them through. These remaining five transactions resulted in the loss of 101 million USD of the Bangladesh Bank account at the Federal Reserve Bank of New York: 20 million USD were transferred to Sri Lanka and 81 million USD to the Philippines. The money transferred to Sri Lanka was later recovered, and 18 million USD from the Philippines were also recovered.

In total, the criminals managed to transfer and steal 61 million dollars. Most of this money was quickly laundered through casinos in the Philippines.

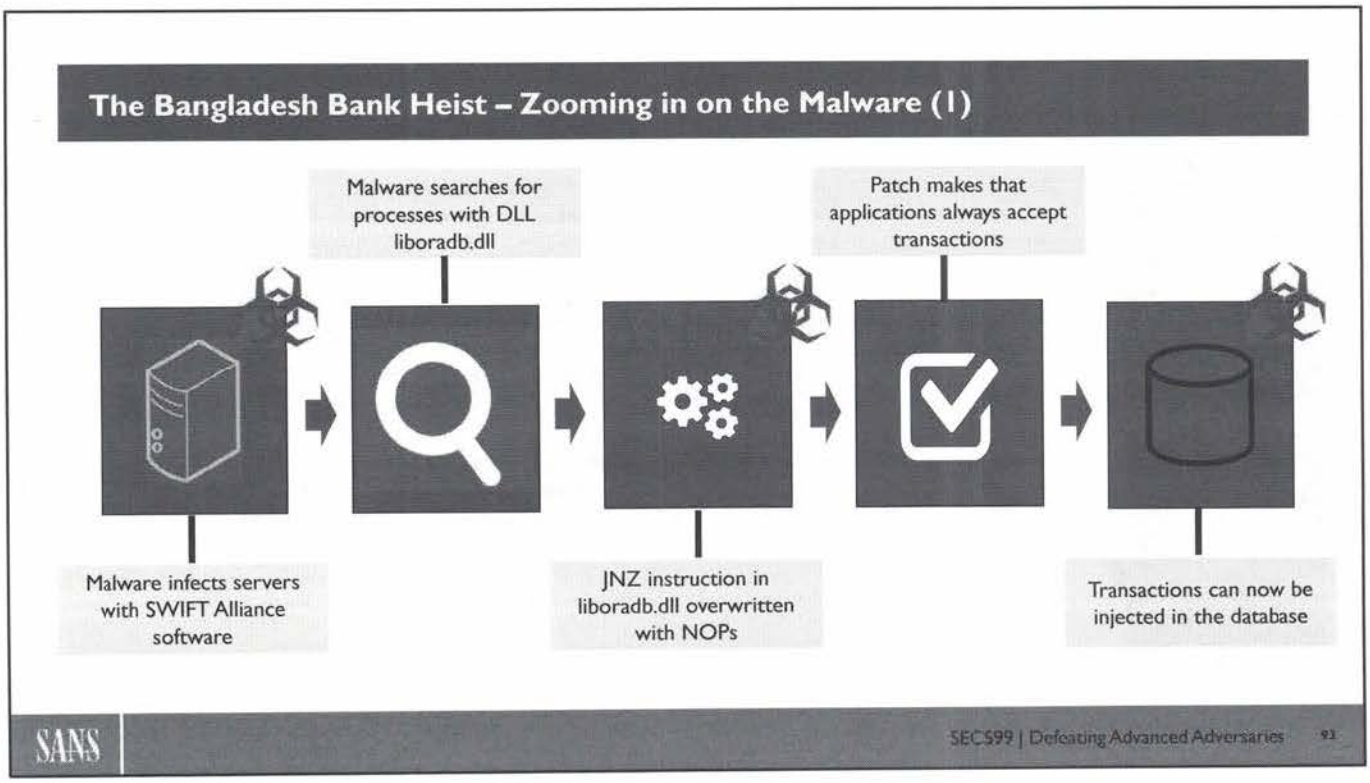
References:
https://www.nettitude.com/wp-content/uploads/2016/12/Nettitude-SWIFT-Threat-Advisory-Report-client.pdf
https://en.wikipedia.org/wiki/2016_Bangladesh_Bank_heist
http://baesystemsai.blogspot.be/2016/04/two-bytes-to-951m.html



The Bangladesh Bank Heist & the APT Attack Cycle

So, let's analyze The Bangladesh Bank Heist by using the APT attack cycle:

- Reconnaissance: The adversaries observed the operation of the Bangladesh Bank (with SysMon and maybe with the help of insiders);
- Weaponization: Windows malware was developed to patch SWIFT Alliance software to be able to inject counterfeit SWIFT messages;
- Delivery: It is not known how the malware was delivered, but reports have hinted at insiders;
- Exploitation: The malicious code was executed on Bangladesh Bank's SWIFT Alliance software servers;
- Installation: A Windows service was installed to make the malware persistent;
- Command & Control: the malware's encrypted config file contains an IPv4 address (196.202.103.174) to communicate with the C&C server;
- Action on Objectives: Counterfeit SWIFT messages were injected in Bangladesh Bank's SWIFT Alliance software servers to instruct the Federal Reserve Bank of New York to transfer $951 from the Bangladesh Bank account to foreign accounts.



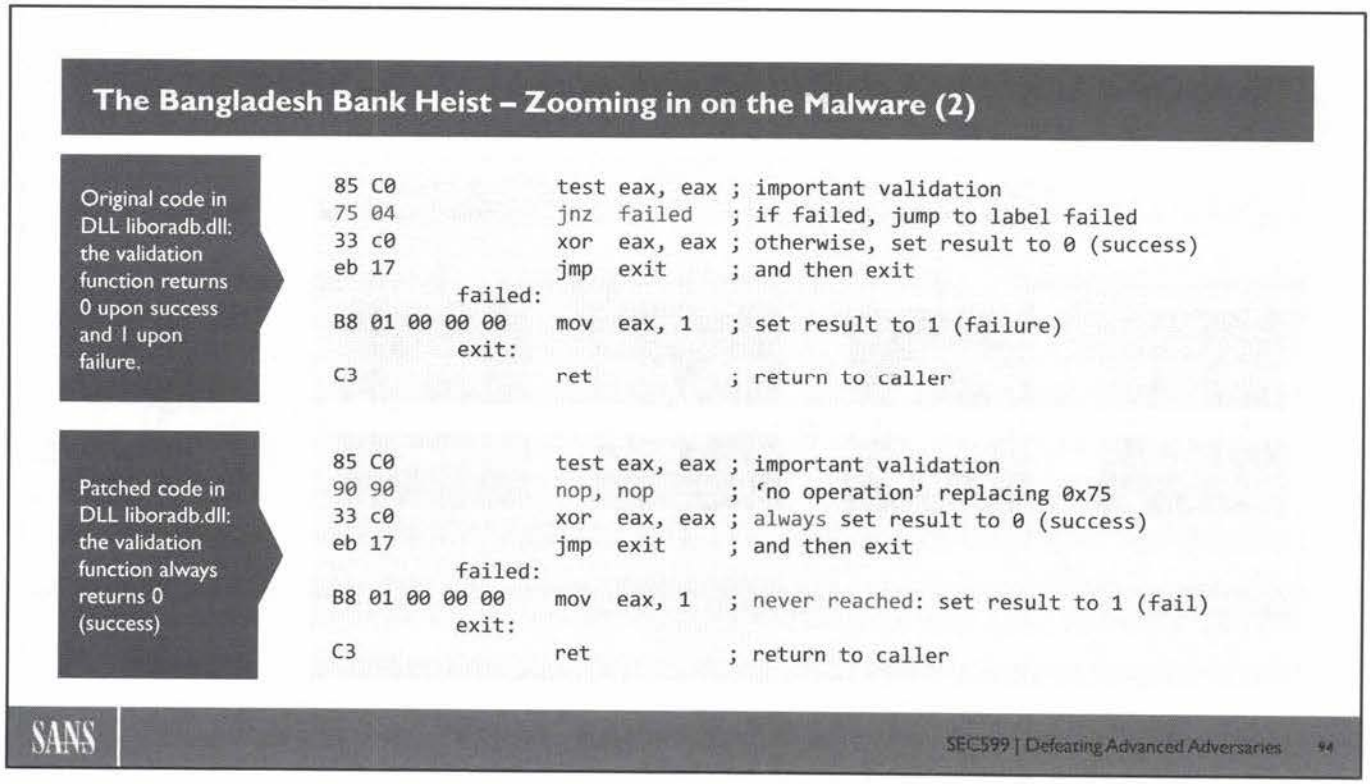
The Bangladesh Bank Heist – Zooming in on the Malware (1)
The malware is used to infect Bangladesh Bank's servers running SWIFT Alliance software. This software is responsible for the processing and managing of SWIFT messages. It is complex software that performs many checks to validate transactions. The malware will change the behavior of the transactions validations of the SWIFT software.

When executing on the server, the malware will check all processes running on the Windows OS and enumerate all modules loaded by processes. Modules are .exe files, .dll files and data files. The malware looks for a particular dll loaded inside a process: liboradb.dll. This DLL is part of SWIFT's Alliance software and performs the following tasks:

- Reading the Alliance database path from the registry
- Starting the database
- Performing database backup & restore functions

In each process that loads DLL liboradb.dll, the malware will patch the DLL in memory by replacing a particular JNZ instruction with 2 NOP instructions. Due to this change, the checks performed by the SWIFT software will always succeed: counterfeit transactions will now be accepted. Patching the DLL in memory has the advantage for the adversaries that it will not be detected by doing an integrity check of the software's files and that it does not invalidate SWIFT's digital signature of the DLL.

Once the SWIFT software has been patched in memory, the criminals can create counterfeit SWIFT messages and inject them into the database without having to get all the details and checks right.



The Bangladesh Bank Heist – Zooming in on the Malware (2)
In this slide, we see assembly code similar to code we would find in DLL liboradb.dll. On the top, we see the original code and at the bottom, the patched code.

The bytes we see at the left of the listings are the bytecodes of the assembly instructions. After that comes the assembly instructions themselves, followed by comment (everything starting with the semi-colon).

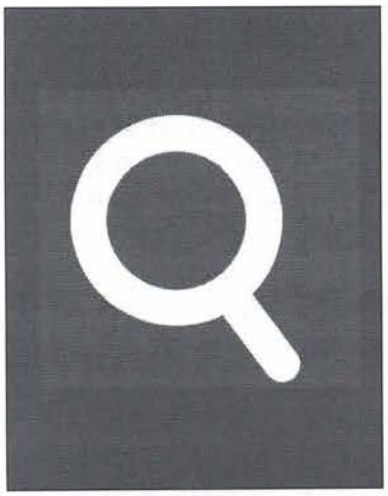
For example, on the first line, we have instruction "test eax, eax". This is an x86 instruction to perform a test on the value in register eax. This test instruction is encoded with 2 bytes: 0x85 and 0xC0.
On the second line, we have a conditional jump instruction: Jump if Not Zero. The Zero Flag is set by the previous test instruction (can be set to 0 or 1), and the jnz instruction will jump to label failed (4 bytes further) if the zero flag is not set, and will not jump (e.g. move on to the next instruction, xor) if the zero flag is set.

Instruction "xor eax, eax" is a trick to set register eax to 0 with a shorter instruction (xor) than a move instruction (mov). mov eax, 0 is valid too but takes 5 bytes instead of 2 for xor eax, eax. Compilers typically use xor when they have to set a register to 0, and not mov.

So, to make that this function always returns success (0), the malware authors have to remove the jnz instruction. But just deleting those 2 bytes is a problem, as this would imply that all subsequent bytes have to be shifted 2 positions, and this would also break jump locations. What is typically done to remove instructions in machine code without changing the position to the remaining instructions, is to replace the instructions with instructions that do nothing. The x86 instruction set has an operation just for that No OPeration, NOP. This instruction is one byte long (0x90). Hence to replace "jnz failed", with instructions that do nothing, we have to replace its 2 bytes (0x75 and 0x04) with 2 nop instructions (0x90 0x90).

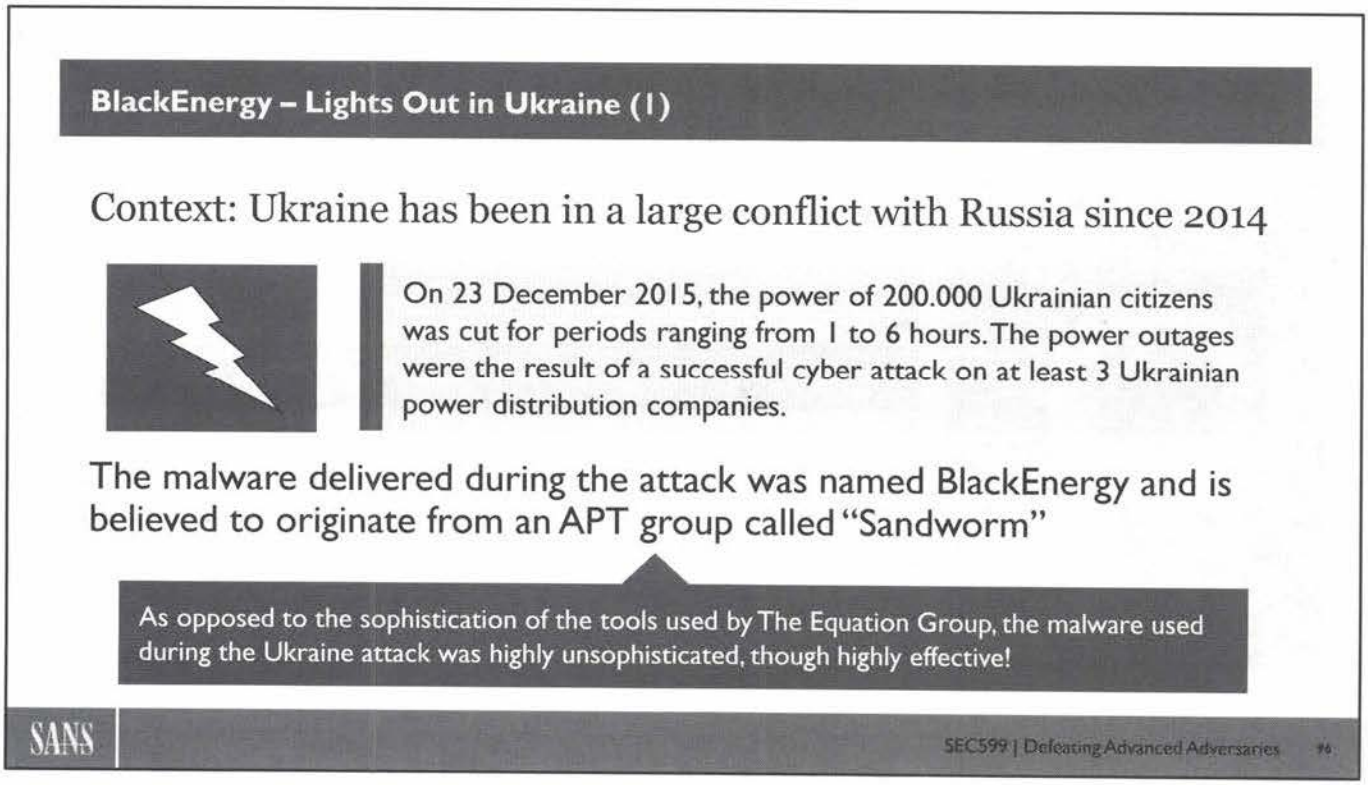
Patching machine code by replacing instructions with nop instructions is a popular technique.

Case Studies We Will Cover



- Shamoon – Destructive attacks in the Middle East
- Bangladesh Bank - The $81 Million heist
- **Black Energy – Lights out in Ukraine**
- Stuxnet – "The world's first digital weapon"
- Epic Turla – From Russia with love
- Regin – Nation state ownage of GSM networks

This page intentionally left blank.

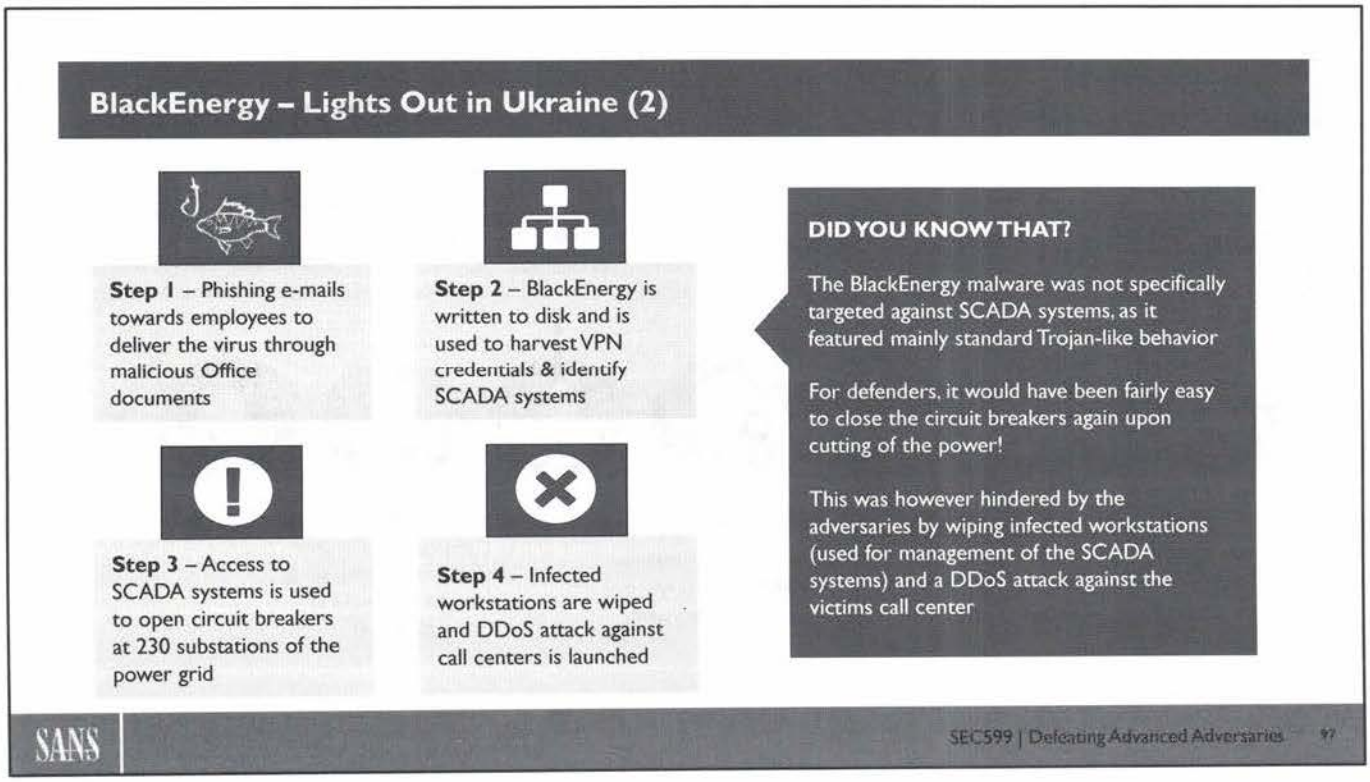


BlackEnergy – Lights Out in Ukraine (1)
On December 23th, 2015, an estimated two hundred thousand inhabitants of the Ukraine were left without electricity for periods varying between 1 and 6 hours. These power outages were the result of a successful digital attack on at least 3 Ukrainian power distribution companies. BlackEnergy is the name of the malware used in this attack.

Involved in a conflict with Russia since 2014, the digital attack on Ukraine is believed to have originated in Russia and security researchers have attributed such attacks to a Russian APT group with the name Sandstorm.

Although the Idaho National Laboratory demonstrated in 2007 that it was possible to physically destroy and electricity generator just using a program (the Aurora Generator Test), this attack on the Ukrainian power grid is believed to be the first successful digital attack on a power grid.

References:
https://en.wikipedia.org/wiki/December_2015_Ukraine_power_grid_cyber_attack
https://www.wired.com/2016/01/everything-we-know-about-ukraines-power-plant-hack/



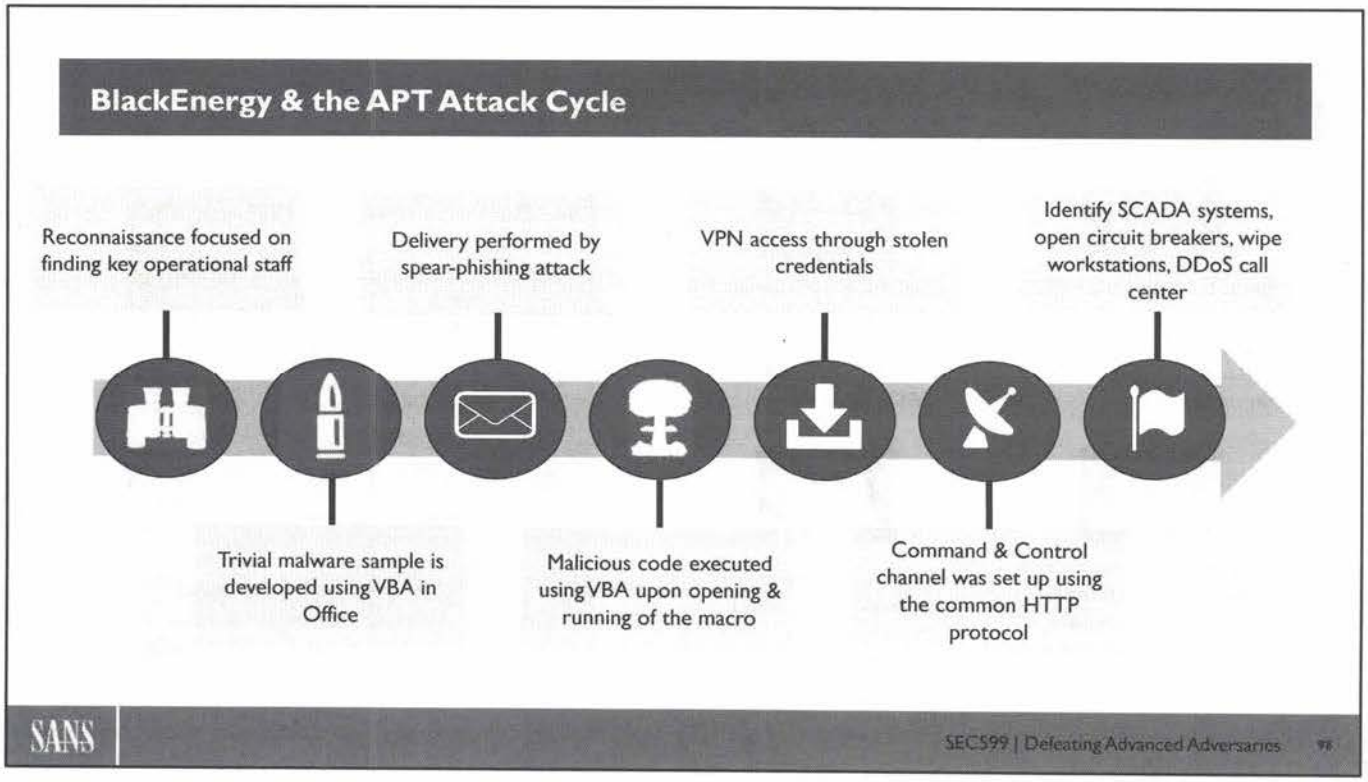
BlackEnergy – Lights Out in Ukraine (2)

Using spear-phishing, malicious documents were delivered to key staff in the power distribution companies. Spear-phishing is a form of phishing (using fake emails) were the recipients are a small, carefully selected group to maximize the success rate of the phishing campaign. Through reconnaissance, the adversaries identified key people in the target companies and obtained their email addresses. The malicious documents were Microsoft office documents using Visual Basic for Applications to deliver the BlackEnergy payload.

BlackEnergy is a modular, 32-bit Windows malware family that is not particularly designed to attack SCADA systems. It has been used for various purposes, like stealing information, remote access and compromising home banking transactions. Through the remote access feature, the adversaries seized control over Windows workstations connected to SCADA systems.

Via those SCADA systems, the adversaries managed to open circuit breakers at 230 substations of the power grid, thereby cutting electricity to 200,000 people. Once power was lost and the power companies were alerted, it would have been simple to restore power by simply closing the circuit breakers again. However, the adversaries took additional steps to prevent this simple operation and thereby prolonging the duration of the power cuts. By wiping the workstations infected with BlackEnergy with the KillDisk program, the adversaries prevented power grid company staff to remotely close the circuit breakers. Staff had to be dispatched to the different substations to manually close the circuit breakers.

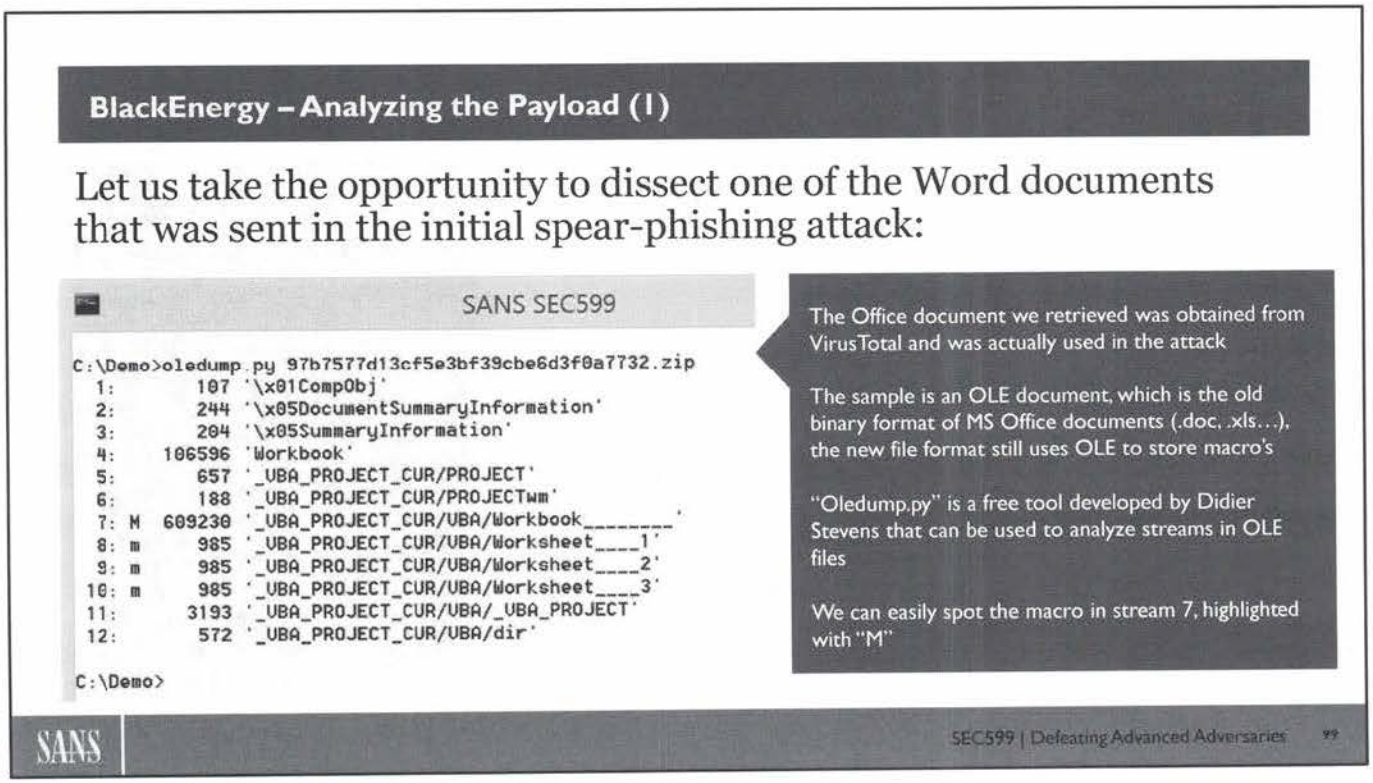
At the same time, a denial of service attack on the power grid company's call centers was executed. By overloading the exchanges with phone calls, adversaries prevented customers from calling in and reporting power cuts. Denying staff access to its control systems and information resulted in power cuts taking up to six hours.



BlackEnergy & the APT Attack Cycle

So, let's analyze BlackEnergy by using the APT attack cycle:

- Reconnaissance: Reconnaissance was performed to identify key operational staff (with access to SCADA systems);
- Weaponization: A trivial technique (using VBA in Office) is used for the initial exploit;
- Delivery: The malicious office document is delivered through a spear-phishing attack;
- Exploitation: Malicious code was executed upon opening of the document;
- Installation: VPN credentials were stolen to provide "persistent" access to the adversaries;
- Command & Control: The common HTTP protocol was used to set up a command & control channel;
- Action on Objectives: A number of actions were taken: identification of SCADA systems, opening of the circuit breakers and finally the wiping of workstations and a DDoS attack against the target's call center.



BlackEnergy – Analyzing the Payload (1)

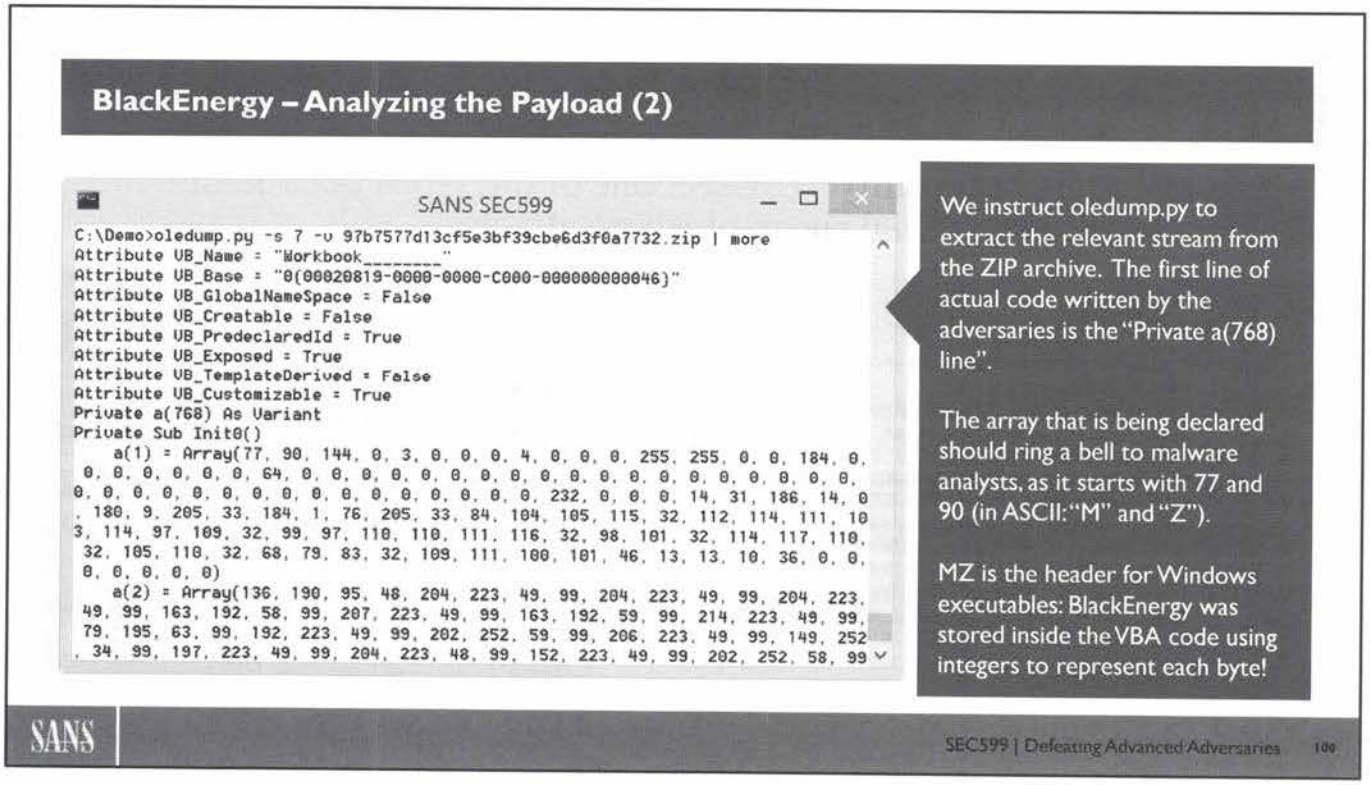
BlackEnergy malicious documents are surprisingly unsophisticated. To illustrate this, we will perform a static analysis of a BlackEnergy malicious document: MD5 97b7577d13cf5e3bf39cbe6d3f0a7732, similar to the one used in the Ukrainian power grid attacks.

We perform a static analysis because opening the document with MS Office involves risks, as it might execute unwantedly. Furthermore, the tool we will use (oledump.py by Didier Stevens) is open source Python, runs on many operating systems and does not require MS Office to be installed.

oledump.py is a command line tool. When oledump.py is launched with the name of the sample to analyze an argument (it may be contained inside a ZIP file), oledump will analyze the OLE files and list all streams found inside the OLE file. Streams with an M or m indicator contain VBA code (stream 7 in this example).

The OLE file format is an old Microsoft file format, officially called by Microsoft the Compound File Binary Format, implementing an embedded file system with files (called streams) and folders (called storage). The binary file formats used by MS Office (.doc, .xls, .ppt …) are actually OLE files. The new file formats introduced with MS Office 2007 (.docx, .docm, .xlsx, …) are ZIP containers with XML files inside, but VBA macros are still stored inside OLE files stored inside the ZIP container.

References:
https://blog.didierstevens.com/didier-stevens-suite/
https://blog.didierstevens.com/my-software/



BlackEnergy – Analyzing the Payload (2)

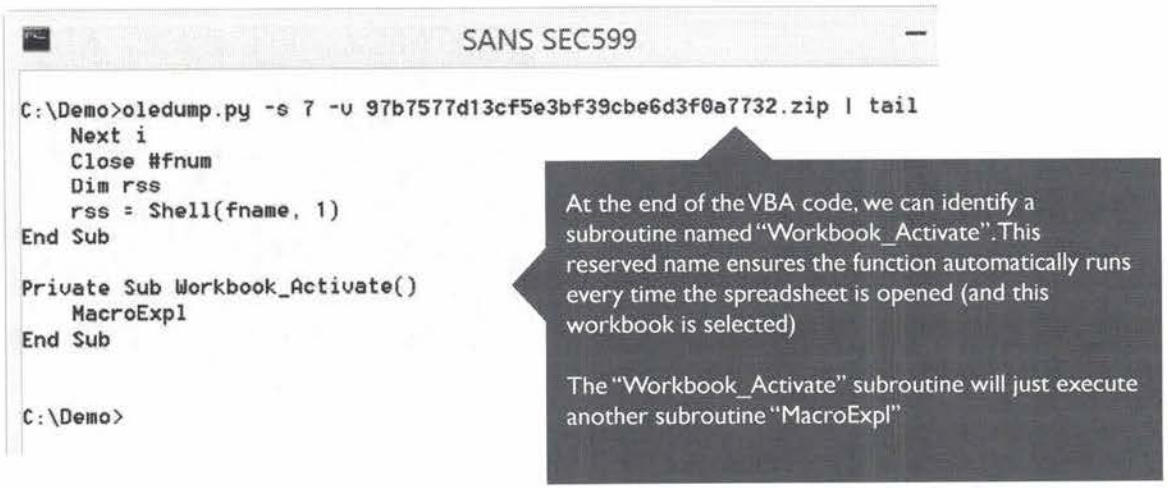
To extract the VBA macro code from stream 7, we have to instruct oledump.py to select stream 7 (with option –s 7) and to decompress the VBA source code (option –v). VBA source code is stored inside stream using a proprietary compression method. Although this method is proprietary, Microsoft has released documents explaining the decompression algorithm.

oledump.py will output the VBA source code, and this typically starts with attributes with names starting with VB_. These attributes are internal to VBA and are not displayed when visualized with the VBA editor inside MS Office.

The first line of code written by the adversaries in this malicious document starts with Private a(768)…

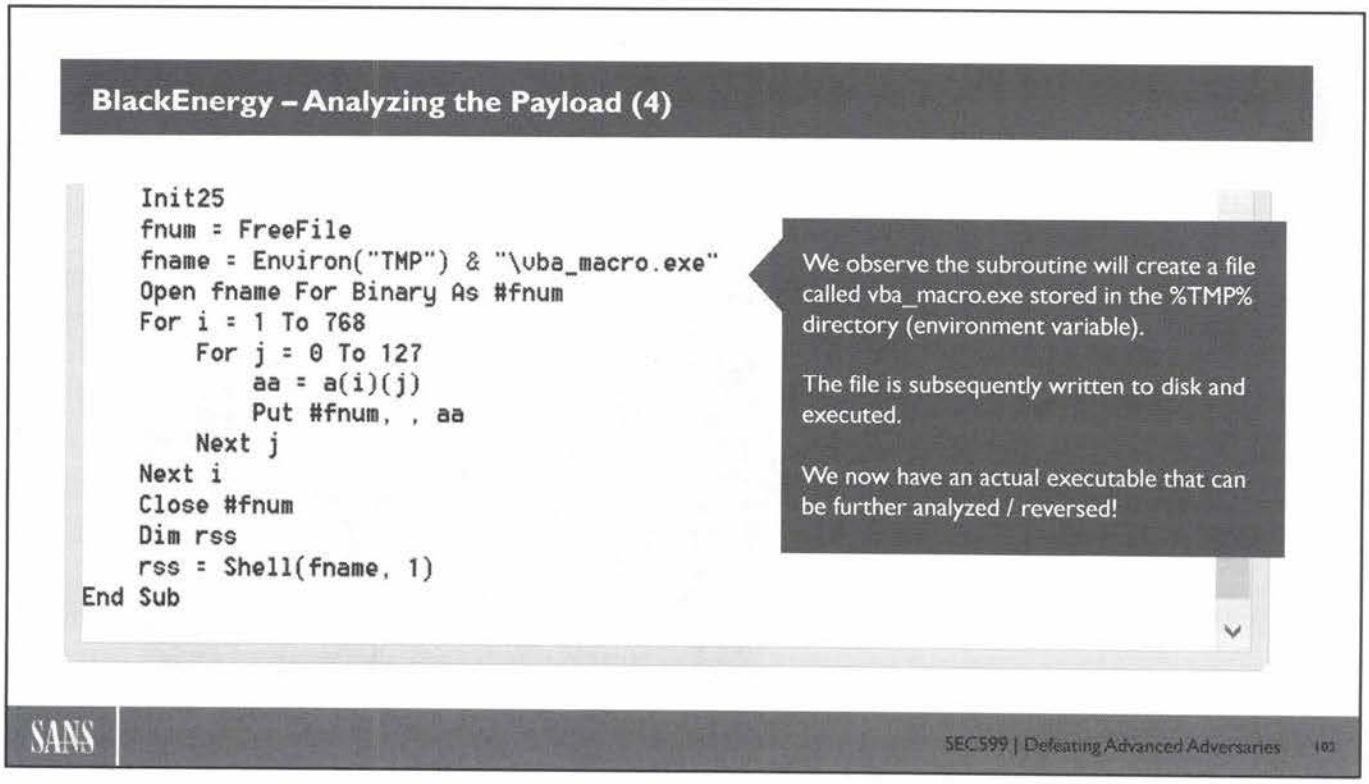
In the second line, we see a declaration of an array, starting with numbers 77, 90, 144, 0… These numbers will sound familiar to malware analysts: 77 is the numerical value of ASCII character M, and 90 is the numerical value of ASCII character Z. MZ is the header of a Windows program. So the payload, malware BlackEnergy, has been stored inside the VBA code using integers to represent each byte.

BlackEnergy – Analyzing the Payload (3)



BlackEnergy – Analyzing the Payload (3)
At the end of the VBA code, we see a subroutine named Workbook_Activate. This is a reserved name and makes that this function will execute automatically (autorun) when the malicious spreadsheet is opened.

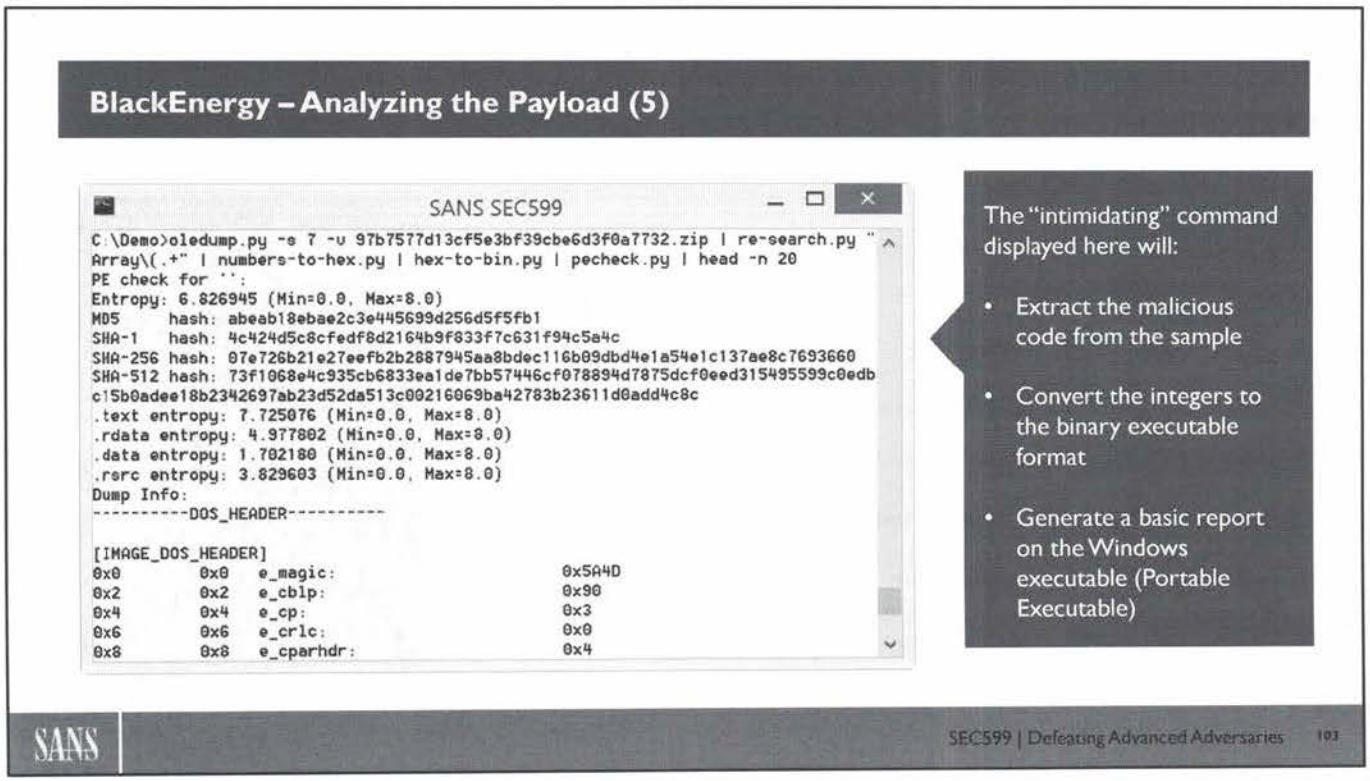
This subroutine will just execute another subroutine: MacroExpl. Let's investigate what MacroExpl is trying to do…



BlackEnergy – Analyzing the Payload (4)

MacroExpl is the subroutine take takes the embedded executable in the arrays and writes each byte of this array to disk. The name of the file dropped by the VBA code is %TMP%\vba_macro.exe. %TMP% is an environment variable pointing to a folder for temporary files inside the user's profile.

After the complete file is written to disk, the executable is launched with the Shell function.

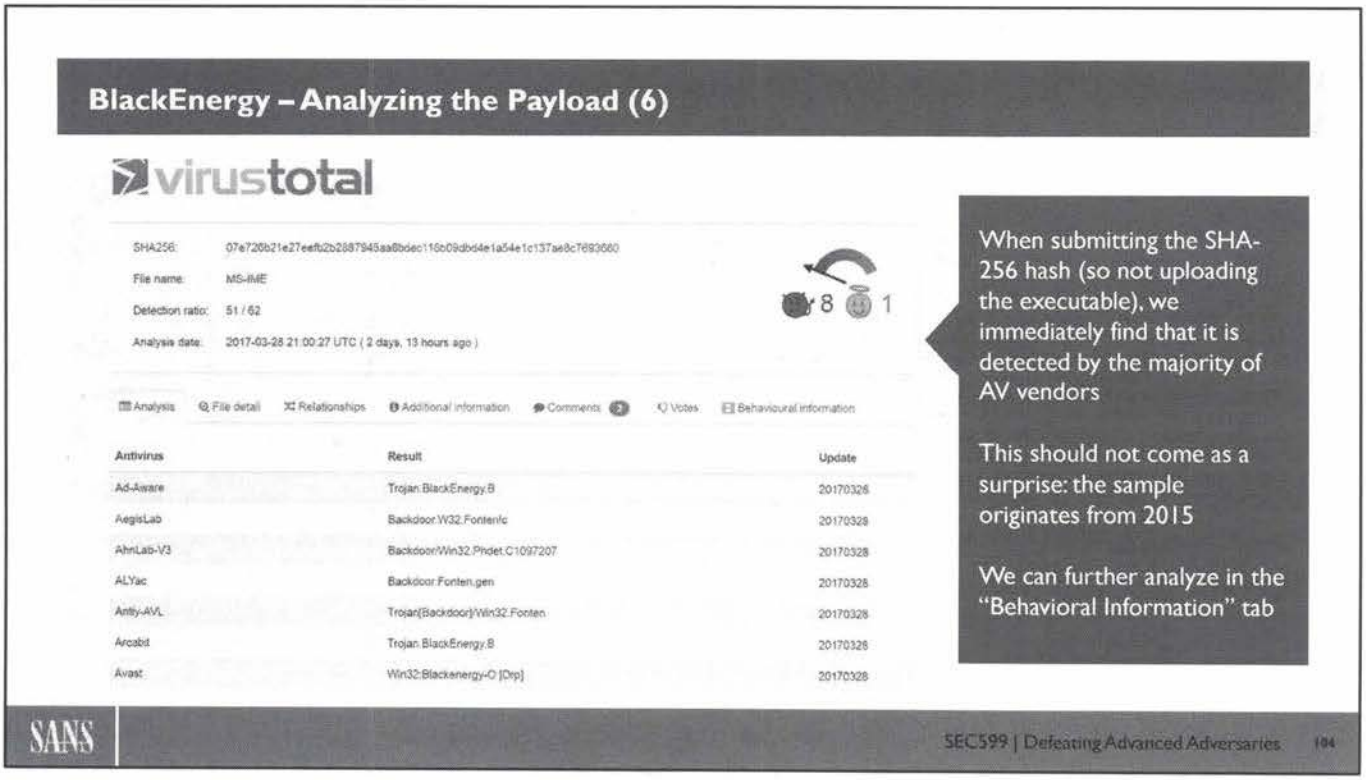


BlackEnergy – Analyzing the Payload (5)

To further understand what the malware does, we need to extract the executable from the VBA source code. This too can be done via static analysis, using small tools developed by Didier Stevens. By piping (|) the VBA code produced by oledump.py through tools to select and convert the integers that make up the binary executable, we can produce the embedded binary executable. The file format used to store Windows executables is called the Portable Executable file format. PE files start with MZ and have a header starting with PE a bit after MZ.

Pecheck.py is a tool developed by Didier Stevens, it's mainly a wrapper around the pefile Python module developed by Ero Carrera. Pecheck.py takes a PE file as input for analysis and produces an extensive report with metadata for the executable. For example, we can see that the report contains the different sections present in the PE file at the beginning of the report. Furthermore, the report starts with the MD5 and SHA checksum of the executable.

Analyzing this PE file is not as straight-forward as the malicious document analysis we just performed. As this is not a malware reverse engineering course, we will use another tool at our disposal: VirusTotal.



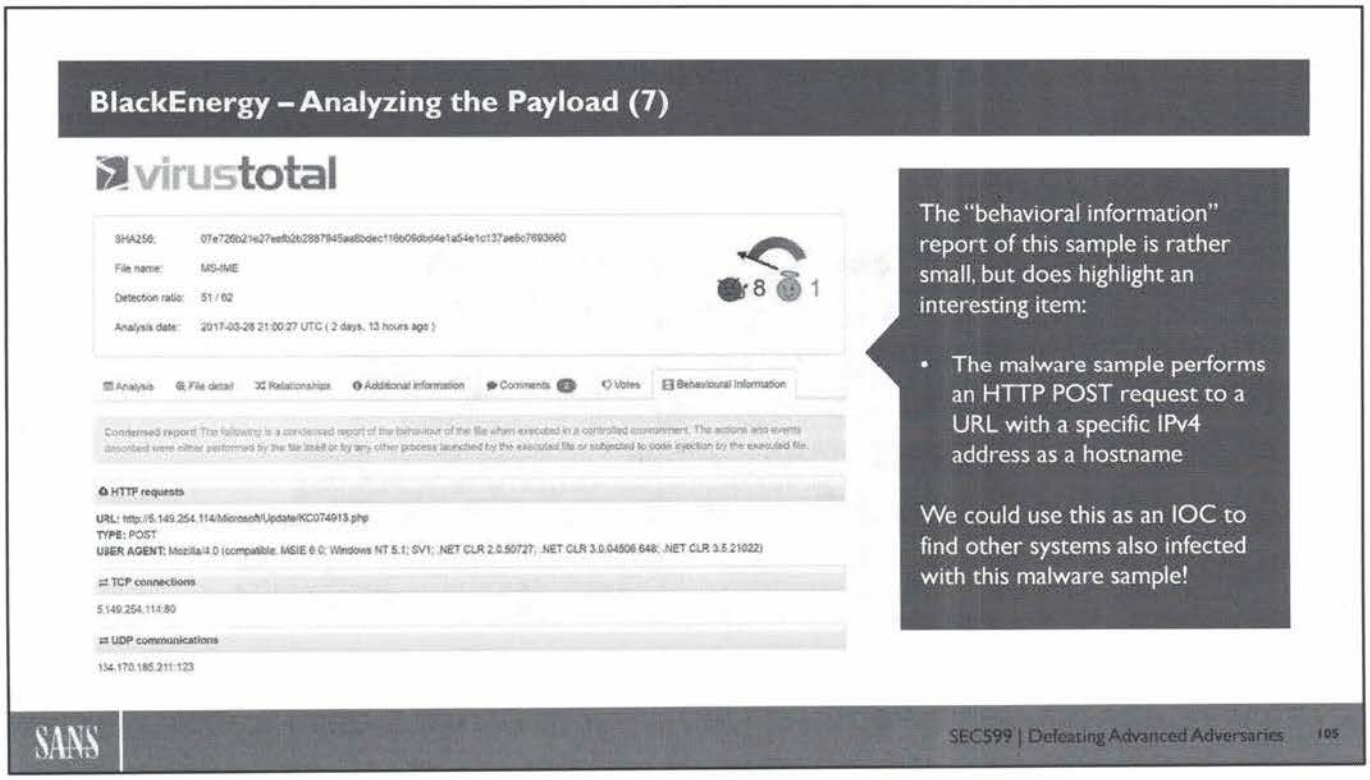
BlackEnergy – Analyzing the Payload (6)

Thanks to the cryptographic hashes of the executable, we can check if this executable has been submitted to VirusTotal. If the malicious executable is not known by VirusTotal, we could upload it for analysis to VirusTotal. But this is something that should be carefully considered.

The reports of samples that have been submitted to VirusTotal are available to everybody. Submitting a sample to VirusTotal, for the first time, can alert adversaries that they have been exposed: they just have to monitor VirusTotal for the hashes of the malware samples they created, and when it appears, they know their attack has been uncovered. An interesting (but commercial) feature of VirusTotal is to not only view reports but to also **download** samples. As an alternative, we could analyze the malware sample in a sandbox we set up and operate ourselves (e.g. Cuckoo).

In our case, we see that the sample has already been submitted to VirusTotal, so we can view the report and observe that several anti-virus programs detect the sample and identify it as BlackEnergy (MD5 abeab18ebae2c3e445699d256d5f5fb1). Notice the presence of a tab named "Behavioural information": this means that the sample has been submitted to a sandbox for execution and observation of its behavior and that a report is available.

Such a report can contain valuable IOCs…

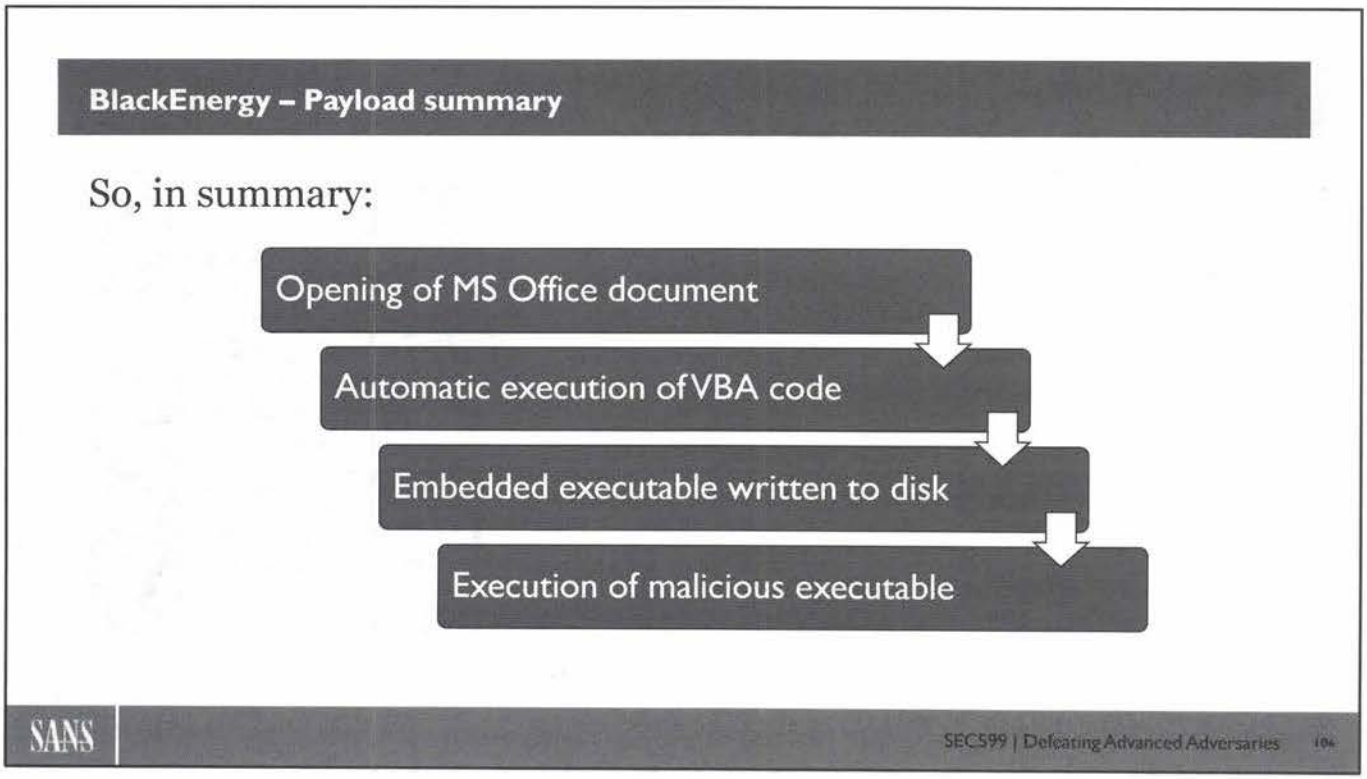


BlackEnergy – Analyzing the Payload (7)

The report with behavioral information for this sample is rather small; usually, these reports contain more information. The small size of the report could be an indication that the malware did not fully execute.

An important fact reported by the sandbox is that the sample did a HTTP POST request to a URL with an IPv4 address as hostname. This is an IOC that can be used to identify infected machines. If the infected organization has proxies with logging enabled, a simple grep through the proxy logs can quickly identify machines where the BlackEnergy sample was executed.

Note that in this case, the IPv4 address is assigned to a Dutch hosting provider, and could possibly have been a previously compromised web server.



BlackEnergy – Payload Summary
The malicious documents used in BlackEnergy attacks are Microsoft Office documents, typically Word or Excel documents. They are called malicious documents because they contain code and a payload that will activate when the document is opened with the Word or Excel application. This is not achieved via exploits or zero-days, but simply with VBA.

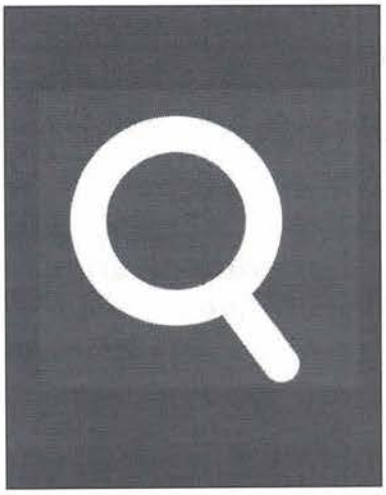
Microsoft Office supports a programming language: VBA. This is not only the case on Microsoft Office for Windows but also on Microsoft Office for Mac.
This is a full-featured programming language with access to the Windows API (unlike VBS) and is not restricted by a sandbox. VBA programs running inside Word have the same access rights to resources like files and registry entries as the user account using Word. This makes it a powerful language for adversaries.

VBA code inside a Word document will not execute automatically when the Word document is opened with MS Word. For this to happen, special VBA functions need to be declared (for example with the reserved function name AutoOpen). Microsoft Office contains protection mechanisms to prevent the automatic execution of VBA code in untrusted documents. It is not known if the power grid distribution staff members targeted by the Sandstorm APT group were socially engineered to bypass these protection mechanisms, or if these protection mechanisms were not in place, either because of (mis)configuration or because of old Microsoft Office versions without protection.

When executed, the VBA code will write an embedded executable to disk (inside the TMP folder) and execute it. This executable is the BlackEnergy malware.

One might wonder why the adversaries took the decision to send an Office document with embedded executable, instead of e-mailing the executable directly. The reason is that many mail servers no longer accept Windows executable (PE files) as attachments and that alternative vectors have to be found to deliver the payload.

Case Studies We Will Cover



- Shamoon – Destructive attacks in the Middle East
- Bangladesh Bank - The $81 Million heist
- Black Energy – Lights out in Ukraine
- **Stuxnet – "The world's first digital weapon"**
- Turla – From Russia with love
- Regin – Nation state ownage of GSM networks

This page intentionally left blank.

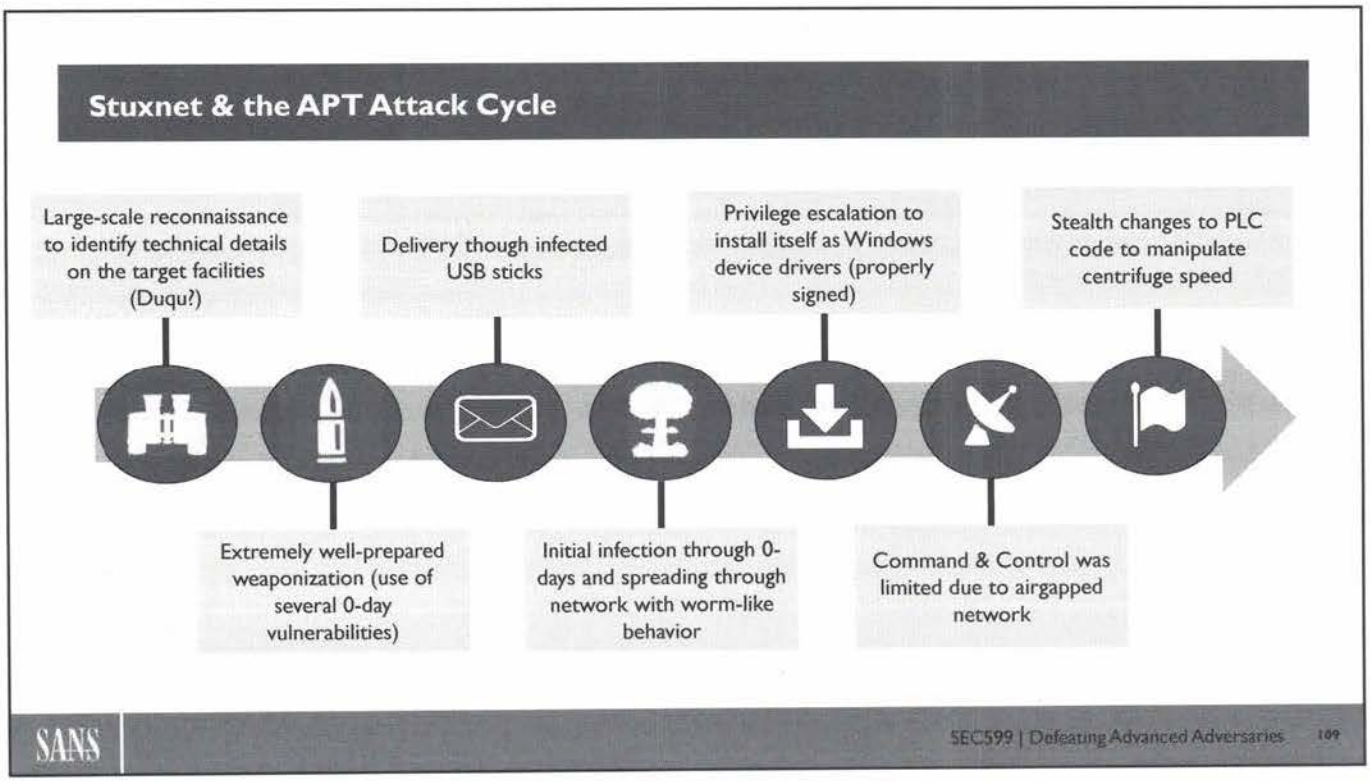
Stuxnet – The World's First Digital Weapon

Since the 1950's, Iran has pursued a nuclear program, with the support of Western countries such as the United States of America. The goal of this program is the production of electricity via nuclear energy. After the Iranian Revolution in 1979, Western countries started to express doubts that the Iranian nuclear program had solely peaceful goals and thus revoked its support.

The nuclear material used in power plants and in nuclear weapons is different but can be produced in similar nuclear facilities. In 1968 Iran signed the Nuclear Non-Proliferation Treaty, thereby accepting inspections from the International Atomic Energy Agency. As the same nuclear facilities can be used to produce weapons-grade nuclear material, the IAEA inspects these facilities to ascertain that they are not misused to produce illegal nuclear material, suitable for weapons. But in 2003 the IAEA launched an investigation after it received information of illegal activities in Iran's nuclear facilities. Iran opposed these inspections and has since then been in conflict with the IAEA and western countries.

Stuxnet is malware developed to disrupt Iran's nuclear program (59% of all Stuxnet infections were in Iran). It is generally agreed the virus' development was supported or even coordinated by nation states.

References:
https://en.wikipedia.org/wiki/Stuxnet
https://www.wired.com/2014/11/countdown-to-zero-day-stuxnet/



Stuxnet & the APT Attack Cycle
So, let's analyze Stuxnet by using the APT attack cycle:

Reconnaissance
As it is believed that not all information could be gathered by normal reconnaissance activities, other malware might have been involved in the planning phase.
Duqu is malware that was detected in 2011, a year after Stuxnet. Duqu is similar to Stuxnet and therefore believed to have been created by the same team of developers. But Duqu is not designed to interfere with PLCs, but to gather information via keylogging, screen capturing …

If it is true that Duqu was used for reconnaissance, then the defenders had a chance to detect it, unlike usual reconnaissance. The use of malware for reconnaissance made this step a complete digital kill chain in itself, with opportunities for detection and prevention.

Weaponization
With 4 zero-day exploits and 2 known exploits, Stuxnet was extremely weaponized. It is not that common that malware uses zero-day exploits, let alone 4 zero-day exploits! As a defender, patching and anti-virus will not help us protect against attacks like Stuxnet. There are no patches for zero-day exploits, and anti-virus started to detect Stuxnet long after it was used to attack. Only application white-listing technology (configured to include DLLs) might help us protect our systems and anti-exploit technology like EMET.

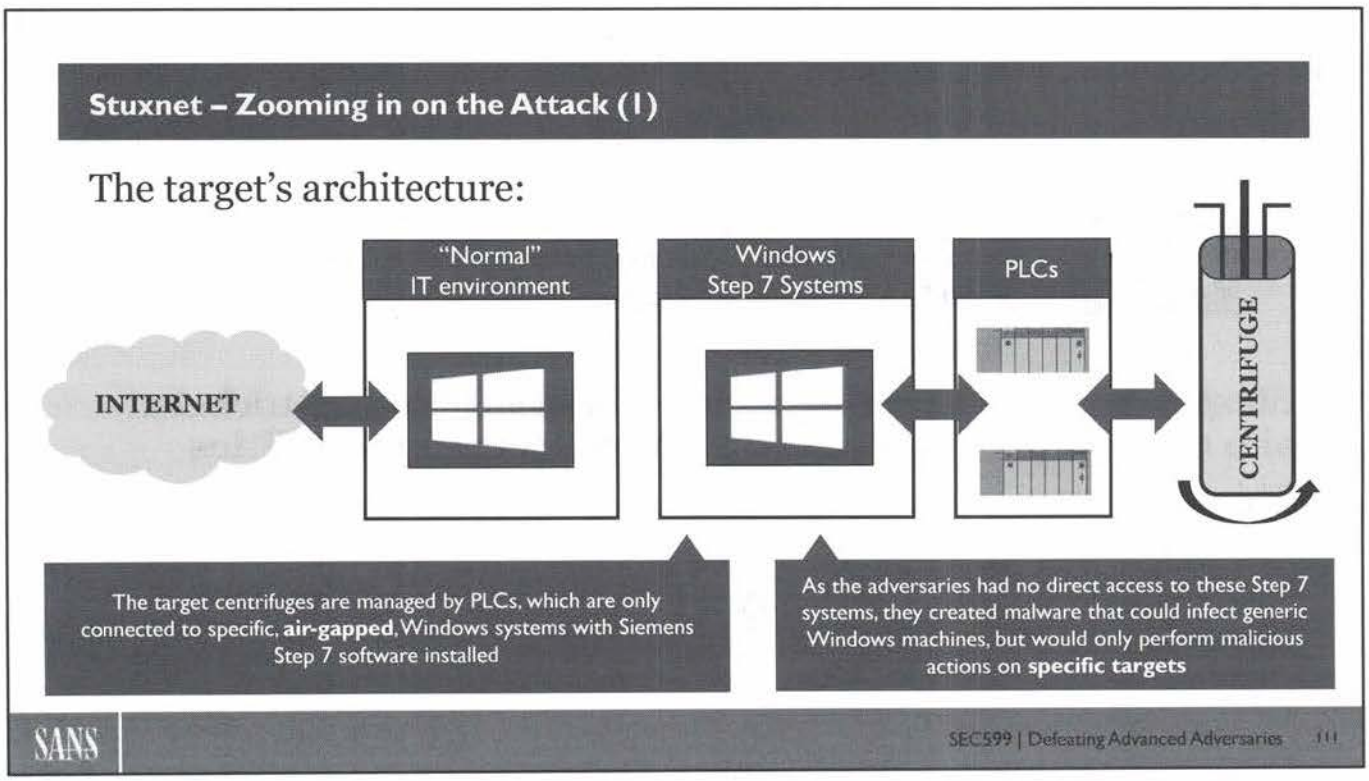
Delivery
Payloads were most likely delivered through infected USB sticks.

Exploitation
0-day vulnerabilities are used for initial exploitation, but also for local privilege escalation and further spreading in the network.

Installation
Uses encrypted filesystem and rootkit to hide its presence;

Command & Control
Although not essential for its correct functioning, Stuxnet seemed to have used Command & Control servers: websites hosted in Denmark and Malaysia. The C&C servers were used to update Stuxnet and to receive uploaded data for espionage. Furthermore, Stuxnet including a peer-to-peer C&C protocol to have interaction with air-gapped networks.

Action on Targets
Stuxnet performed many actions on its targets, all designed to be able to reach the final target: compromise of the S7-300 PLCs used in the Iranian uranium enrichment facilities.

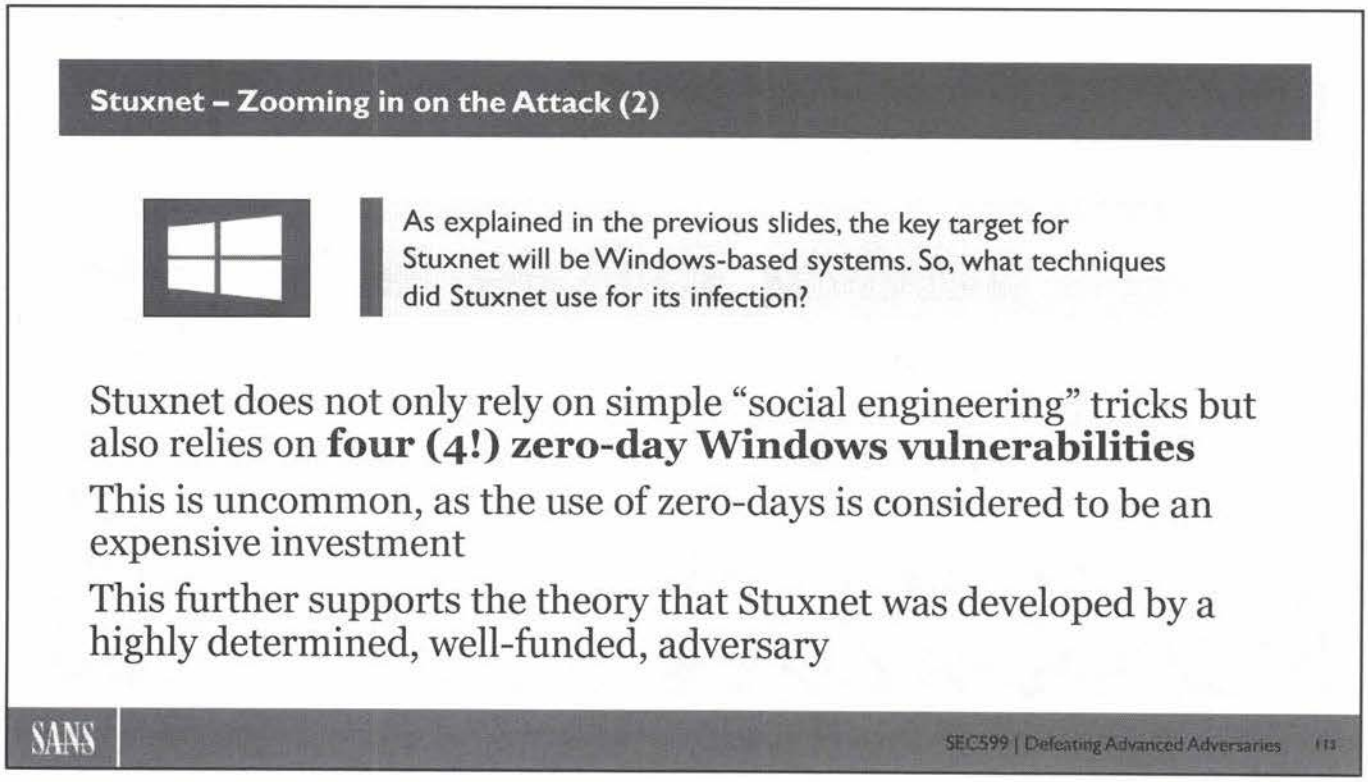


Stuxnet – Zooming in on the Attack (1)
In order to enrich / weaponized uranium, the nuclear centrifuges have to spin at the exact right frequency. In order to control or disrupt the uranium enrichment process, the adversaries would thus have to obtain control over the PLCs (Programmable Logic Controllers). These PLCs are not connected to the Internet, and cannot be attacked directly. The PLCs are programmed via Siemens' Step 7 software on Windows computers. For obvious reasons, these Windows computers also are not connected to the Internet.

Because the adversaries had no physical access or Internet connection to the Windows computers programming the PLCs, they decided to follow an elaborate plan to infect thousands of computers in the hope to reach the target computers.

It is not known how Stuxnet was initially propagated to Windows computers to start the chain of infections, but the Windows component has several attack vectors (including infection capabilities using USB drives, network connectivity,…). Stuxnet will achieve persistence on Windows computers but will remain dormant unless the infected Windows machine runs Siemens' Step 7 PLCs programming software.

If the Windows computer runs Step 7, Stuxnet will proceed to infect the attached PLCs. But again, does not do this indiscriminately, Stuxnet will only infect Siemens PLCs of a particular model and with particular modules attached to it.

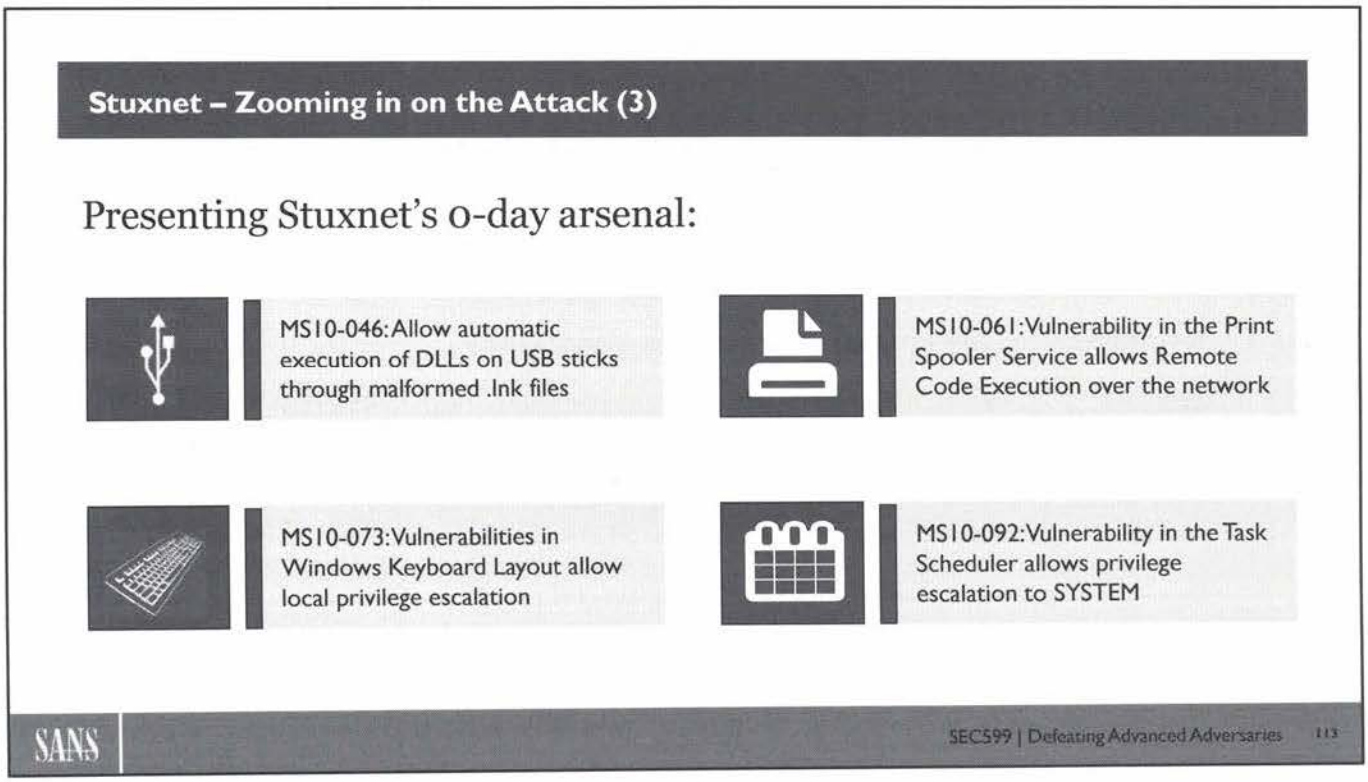


Stuxnet – Zooming in on the Attack (2)
To reach the target computers in Iran's nuclear enrichment facilities, Stuxnet infected computers worldwide. More than half of the infected computers were located in Iran, but many computers were infected in countries like Indonesia and India too.

Stuxnet used several exploits to infect computers, several of them were zero-days. A zero-day is an exploit that is not publicly known and for which there is no patch. Usually, Windows zero-day exploits that achieve code execution are valuable, and it is rare to see them used in common malware. As reliable zero-day exploits can command prices from ten thousands to hundreds of thousands of dollars, malware authors tend to use them sparingly. Using a zero-day in malware exposes it to discovery and ultimately patching, thereby significantly reducing its utility to the adversaries and thus its worth.

That is why the use of zero-days in malware is remarkable. Using 4 Windows zero-days, like Stuxnet did, is unprecedented. Many researchers believe that this indicates that the adversaries had vast resources at their disposal and were very determined to achieve their goal. This is another argument for attributing Stuxnet to a nation-state actor like the United States of America and Israel.

Two zero days are used as a propagation vector: the .lnk file vulnerability and the printer spooler vulnerability. Both achieve code execution without user interaction. The other zero-day exploits achieve privilege escalation, allowing the malware to run with the highest privilege and infect the Windows kernel.



Stuxnet – Zooming in on the Attack (3)
Presenting Stuxnet's four 0-days:

- MS10-046: Allow automatic execution of DLLs on USB sticks through malformed .lnk files
  Infecting computers via portable media like USB sticks is a common practice, but has become less practical since Microsoft started to change the autorun behavior of Windows. On old versions of Windows, removable storage could be configured to execute programs stored on the medium automatically, upon connection of the removable media with the computer. This behavior has changed in modern versions of Windows, and the user is always warned before programs autorun, with the option to prevent execution.

  Stuxnet exploits a vulnerability in the parsing of Windows Shortcut files (.lnk) to achieve code execution without user interaction. Due to a bug in Windows Explorer, DLLs present on the USB stick can be loaded and executed inside the Windows Explorer process when they are referenced in a particular way in the .lnk file. A DLL is a Windows library with executable code. By putting a malicious DLL file on a USB stick together with a malformed .lnk file exploiting this vulnerability, adversaries could achieve code execution on a Windows computer merely by inserting a USB stick and viewing the drive in Windows Explorer.

- MS10-061: Vulnerability in the Print Spooler Service allows Remote Code Execution
  Although the target Windows computers were air-gapped from the Internet, they were nevertheless connected via an IP/Ethernet network. When a Windows computer is connected to a trusted IP network, it will expose many services to be consumed by peer computers on the network. One of these services is the print spooler service, designed to share an attached printer with other Windows computers.

  The zero-day vulnerability inside the print spooler service allowed adversaries to have an infected computer connect to the print spooler of another Windows computer on the network and achieve remote code execution. Malware that self-propagates via networked computers without any user interaction is called a worm and is considered very potent malware that can spread blindingly fast.

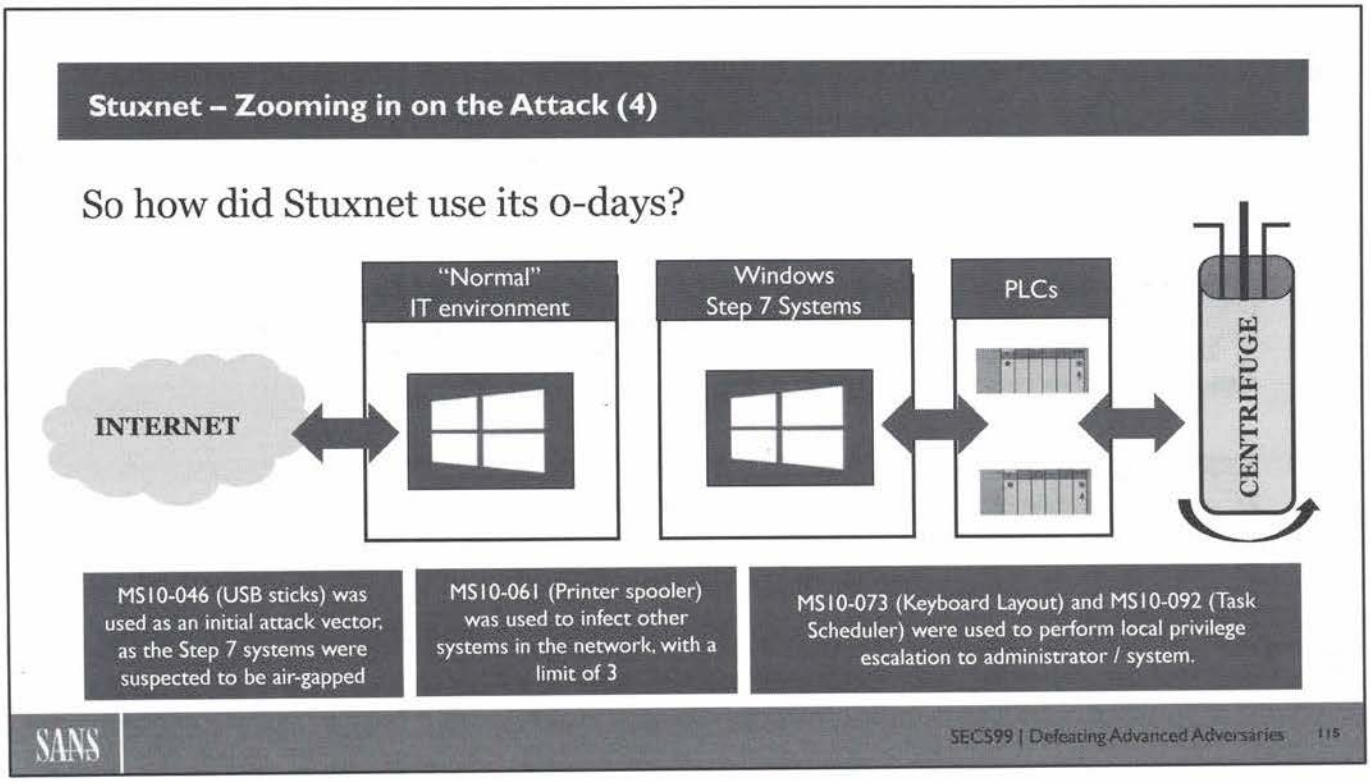
The 0-days described above would only provide limited, unprivileged, access to Windows systems. In order to reach its full potential, Stuxnet also included two 0-days that could help it escalate privileges:

- MS10-073: Vulnerabilities in Windows Keyboard Layout allow local privilege escalation
  A privilege escalation vulnerability existed due to the way that the Windows kernel-mode drivers maintain the reference count for an object. An attacker who successfully exploited this vulnerability could run arbitrary code in kernel mode. An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights.

  Stuxnet abused this flaw to escalate its privileges on infected systems.

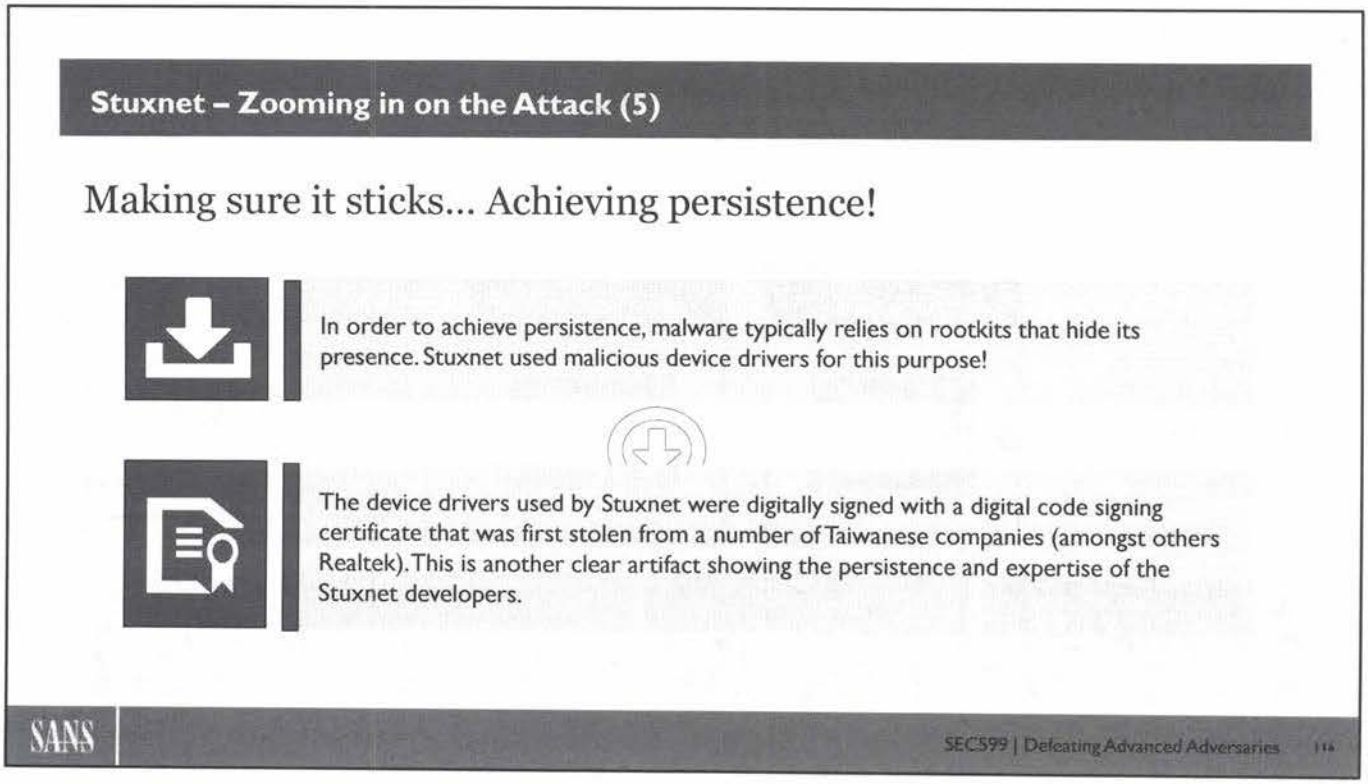
- MS10-092: Vulnerability in the task scheduler allows privilege escalation to SYSTEM
  When processing task files, the Windows Task Scheduler only used a CRC32 checksum to validate that the file has not been tampered with. Also, in a default configuration, normal users can read and write the task files that they have created. By modifying the task file and creating a CRC32 collision, an attacker can execute arbitrary commands with SYSTEM privileges.

  Stuxnet abused this flaw to escalate its privileges on infected systems.



Stuxnet – Zooming in on the Attack (4)
Stuxnet's 0-day arsenal was used in the following way:

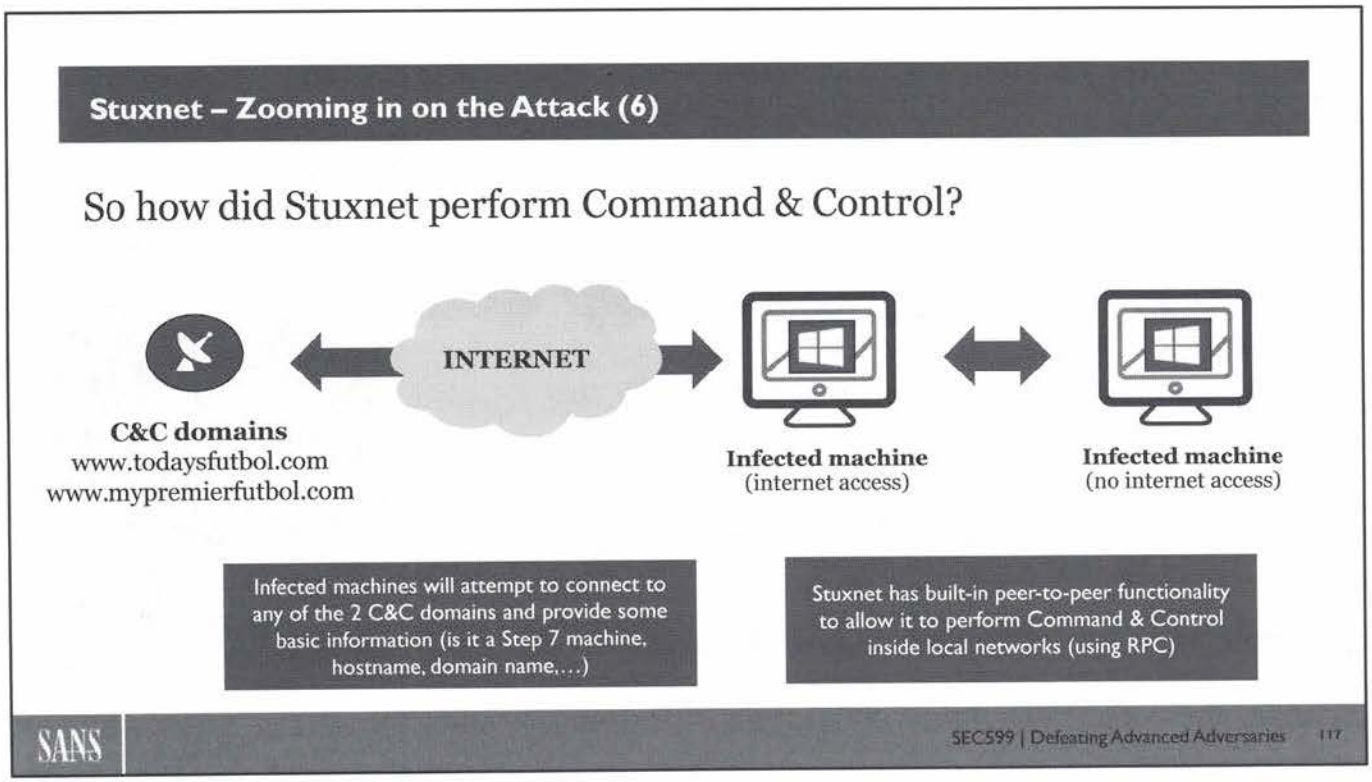
- In order to overcome the "airgapped network" problem, Stuxnet used MS10-046 in an attempt to infect target systems through infected USB sticks.
- In order to further infect other systems (e.g. in the airgapped network), MS10-061 was used to create a worm that would infect Windows systems through the typically exposed Printer Spooler Service. This worm-like behavior could spiral out of control, which is something the Stuxnet developers took into account: Stuxnet would not spread to more than 3 machines and erased itself after 24 June 2012.
- After achieving code execution on a Windows machine, Stuxnet needs to obtain full permissions on the Windows machine and achieve persistence. When code is executed via the .lnk vulnerability or the printer spooler vulnerability, it is not running with full permissions. The code is running in the context of a restricted user and needs to run in system context to fully compromise the host Windows machine. To obtain system-level access on infected hosts, Stuxnet used the MS10-073 and MS10-092 zero-days.



Stuxnet – Zooming in on the Attack (5)
Upon obtaining SYSTEM-level access to target machines, Stuxnet proceeds to install rootkits (user and kernel rootkits) so that it could hide its presence on the infected machine. A rootkit is malware designed to conceal the presence of malware. For example, a rootkit might hide certain files when a user executes a directory listing command.

The kernel rootkit was installed via device drivers. On Windows, device drivers need to be digitally signed before they can be installed. The authors of Stuxnet obtained 2 stolen digital code-signing certificates from Taiwanese companies (amongst others the rather well-known Realtek). These certificates were used by the Stuxnet developers to sign their own, malicious, device drivers.

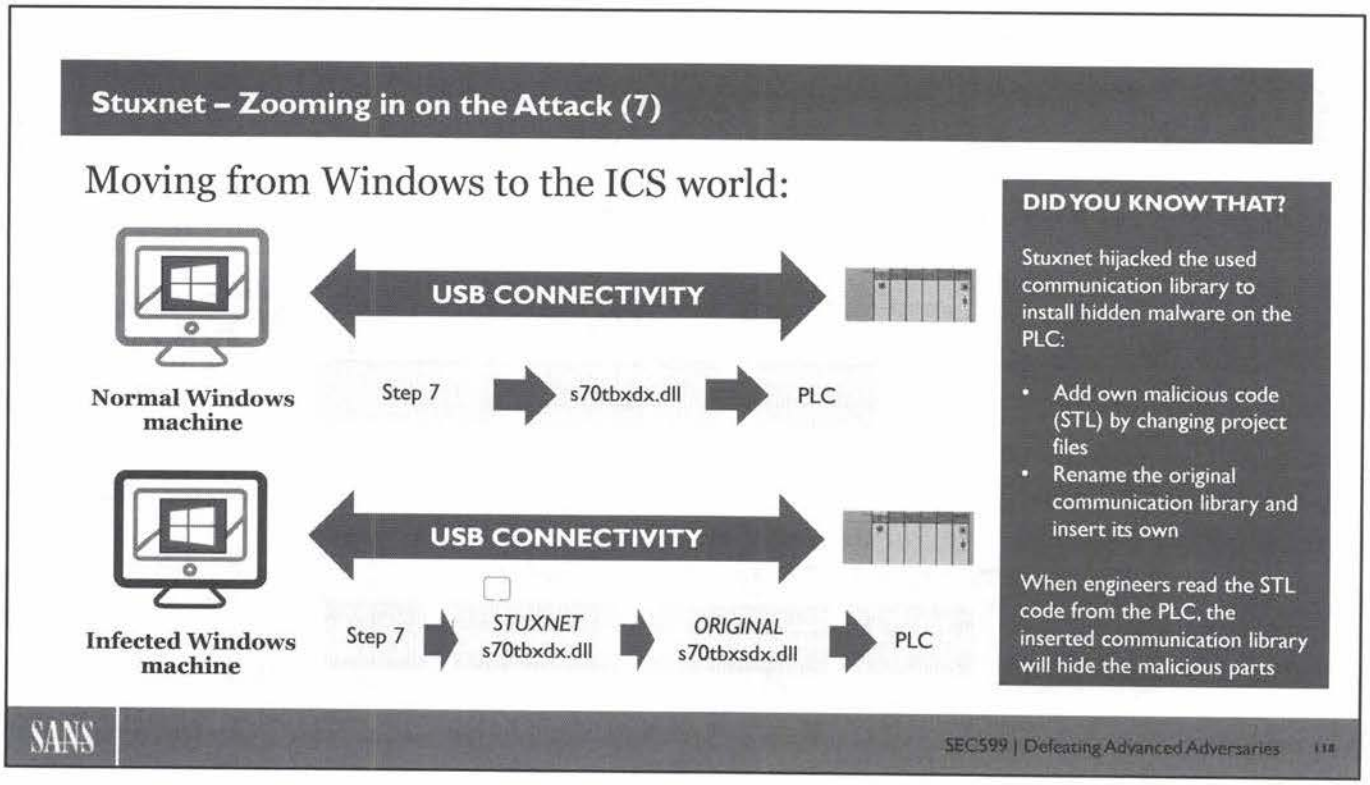
At this stage, Stuxnet looks for Step 7 software on the infected machine and the actual "attack" can start!



Stuxnet – Zooming in on the Attack (6)

Stuxnet was designed to operate without a fine-grained C&C infrastructure (as it needs to operate in air-gapped networks). That doesn't mean it has no C&C infrastructure:

- Two main C&C domains were used (www.todaysfutbol.com and www.mypremierfutbol.com), to which infected systems would send some initial information. This included, for example, the hostname & domain name of the system, but also a Boolean to indicate the system had Step 7 installed or not;
- Stuxnet-infected systems will also run an RPC server, which is used for peer-to-peer communications between infected hosts. This allows machines that are not connected to the Internet to receive updates and exfiltrate information (if it has peer-to-peer connectivity with other systems that are connected to the Internet);



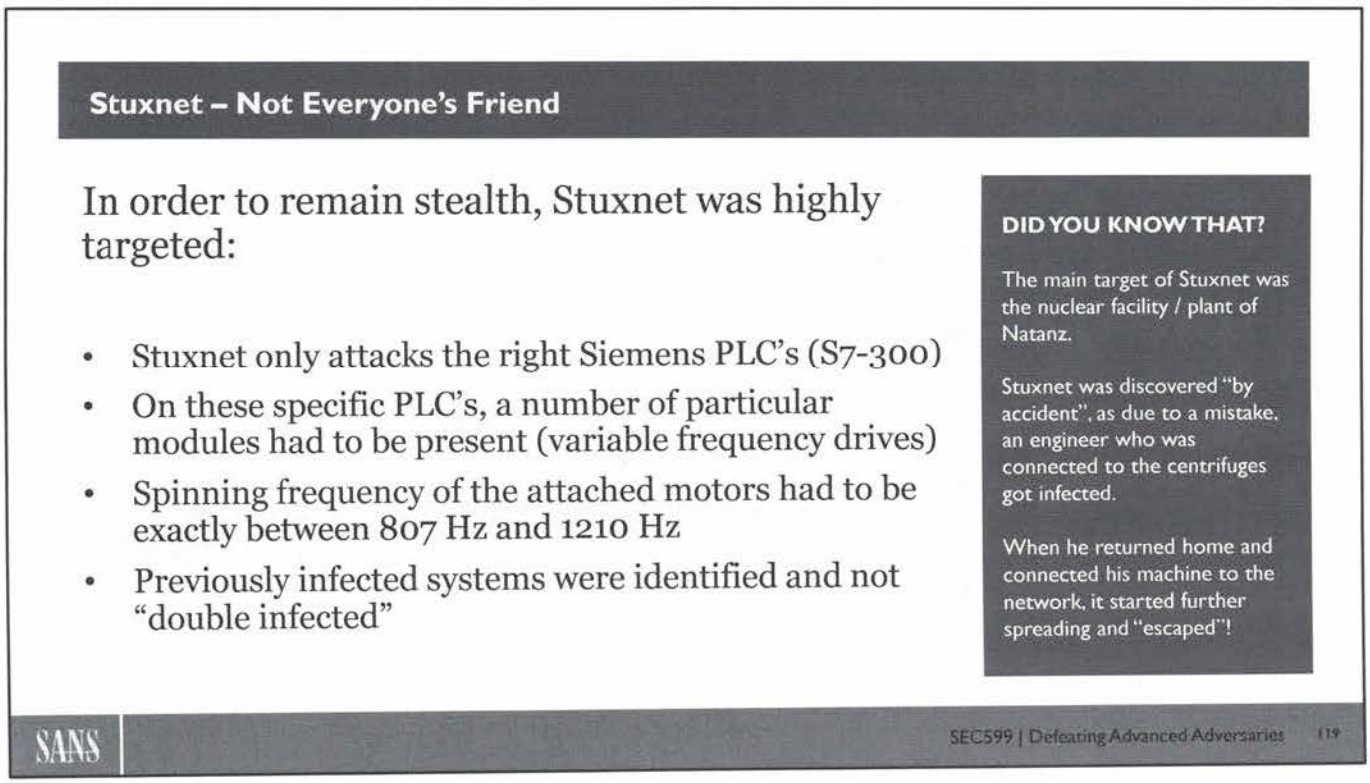
Stuxnet – Zooming in on the Attack (7)
Siemens' PLCs need to be connected via a data cable to a Windows machine running Step 7 software to be programmed. When connected, Step 7 will communicate with the PLC via a communication library (DLL s7otbxdx.dll).

On Windows computers with Step 7, Stuxnet will modify Step 7 project files to inject code and hijack the communication library to install hidden malware on the PLC:

- By modifying the project files, Stuxnet can inject its own STL code (Siemens' PLC programming language, Statement List) into the PLC.
- By renaming the original communication library s7otbxdx.dll to s7otbxsdx.dll, and inserting its own malicious communication library as s7otbxdx.dll, Stuxnet can interfere with the communication between Step 7 and the PLC.

Under normal circumstances, Step 7 can read an STL code block from the PLC by calling a function in the communication library s7otbxdx.dll to read a particular code block. This allows Step 7 to retrieve the program code of a PLC and have a programmer inspect and/or modify the code.

As this would potentially reveal malicious code installed on the PLC, the Stuxnet developers wanted to prevent this. Therefore, they inserted their own communication library between Step 7 and the original communication library. When Step 7 would want to retrieve a particular STL code block, it would call a function in communication library s7otbxdx.dll (the adversaries' library), which would pass it on to the original communication library which in turn would retrieve it from the PLC over the data cable. If the retrieve STL code block would contain malicious code (implanted by Stuxnet), the adversary's communication library would modify the STL code block to hide the malicious code before returning it to Step 7.



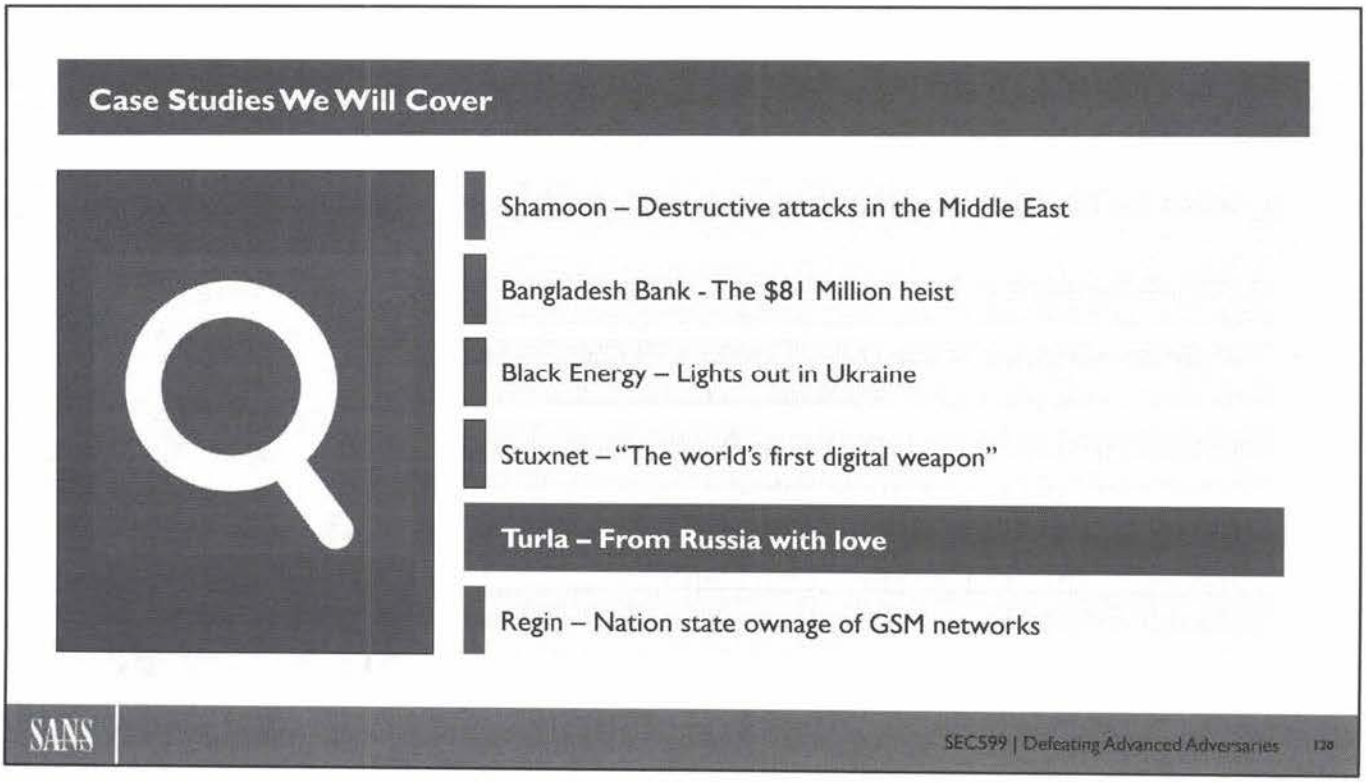
Stuxnet – Not Everyone's Friend

To achieve its goal while remaining stealth, Stuxnet would only infect specific PLCs that could possibly be used in Iran's uranium enrichment facilities to drive centrifuges. This is another strong indication that the adversaries were well prepared and disposed of specific information through reconnaissance.

Stuxnet would only infect Siemens' S7-300 PLCs. All other PLCs were left untouched. Targeted S7-300 PLCs would have to be configured with modules connected to variable-frequency drives of particular make and model (Iran and Finland). Variable-frequency drives control the speed of motors. These infected PLCs would be programmed with malicious code to monitor the speed used to drive the centrifuge motors, and only interfere with the operation if specific criteria are met. For example, the spinning frequency of the attached motors had to be between 807 Hz and 1210 Hz, all to avoid interfering with PLCs that are not used for the Iranian nuclear program.

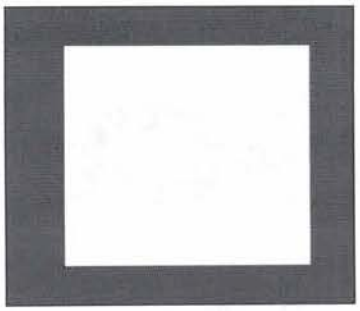
Furthermore, Stuxnet did not interfere with systems that were already infected.

When all conditions were met, Stuxnet would periodically modify the frequency of the drivers to alter the speed of the centrifuge motors, while reporting the original frequency back to the monitoring systems. This is the first documented case of a rootkit on a PLC.



This page intentionally left blank.

Epic Turla – From Russia, with Love



Turla, also known as Snake or Uroburos is one of the most sophisticated ongoing cyber-espionage campaigns (on a similar technical level as Regin)

Targets of the campaign include government entities (e.g. Ministry of Foreign Affairs, Intelligence Agencies), embassies, military, research and education organizations and pharmaceutical companies

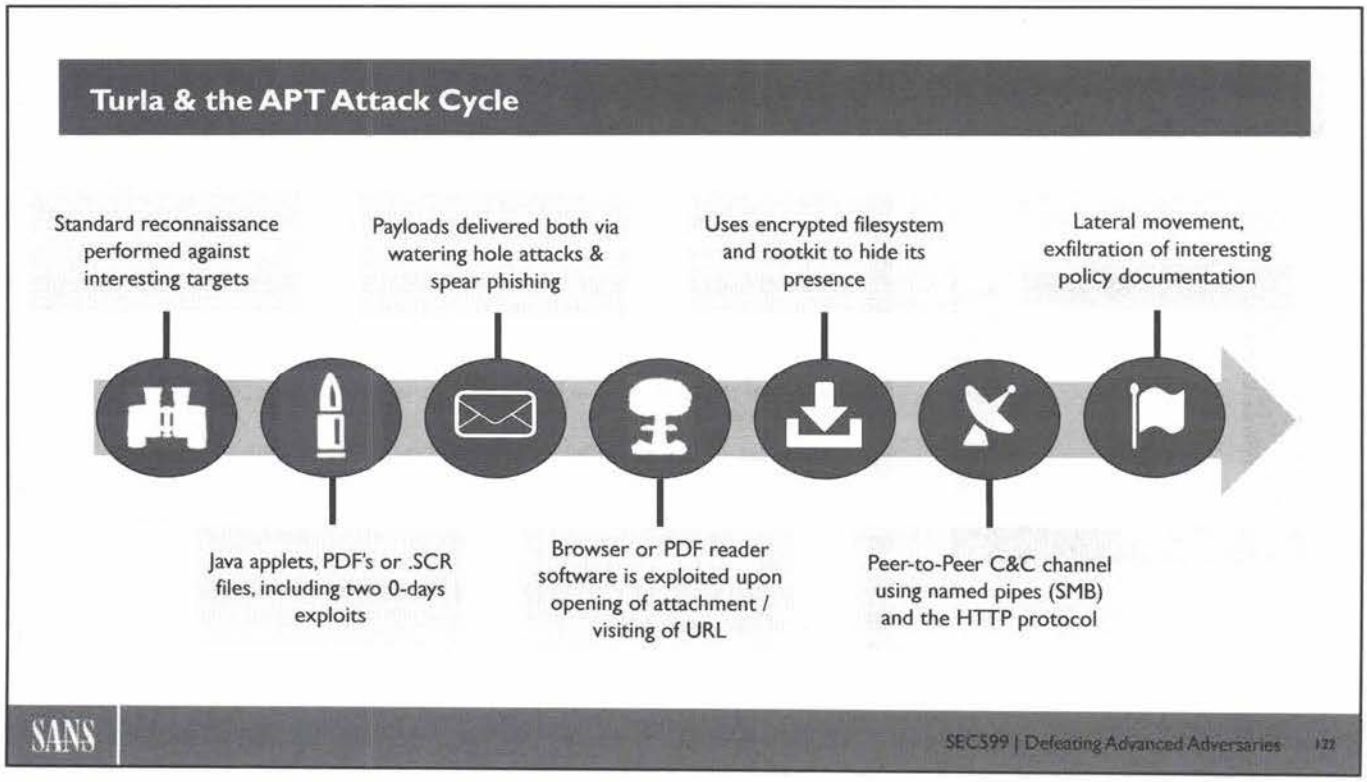
The name of the campaign refers to Ouroboros, which is the ancient symbol of a snake or dragon biting its own tail, it is widely accepted to be of Russian origin

Epic Turla – From Russia, with Love
Like Regin, Turla is very complex and modular. It uses an encrypted virtual file system too to store payloads and data. The difference with Regin is that Turla uses the NTFS file system in encrypted container files instead of a custom file system. Access to these encrypted container files is provided through kernel-mode device drivers. Turla installs just a couple of executables, everything else is stored in the encrypted virtual file system.

Turla is Windows malware (32-bit and 64-bit) first discovered in 2013 but has been targeting Western government and military organizations since at least 2008. Due to language and strings in the executables, encryption keys used and behavior, G Data attributes Turla to Russia. The malware seems to be related to malware Agent.BTZ that was used in 2008 during an attack on the United States of America. Turla checks for the presence of Agent.BTZ on a machine it tries to infect and remains inactive if found.

The Turla malware contains many references to snakes. Filenames containing the word snake and strings in the code like Ur0BuR()s. Ouroboros is an ancient symbol of a snake or dragon biting its own tail.

References:
https://en.wikipedia.org/wiki/Ouroboros
https://public.gdatasoftware.com/Web/Content/INT/Blog/2014/02_2014/documents/GData_Uroburos_RedPaper_EN_v1.pdf
https://en.wikipedia.org/wiki/Turla_(malware)

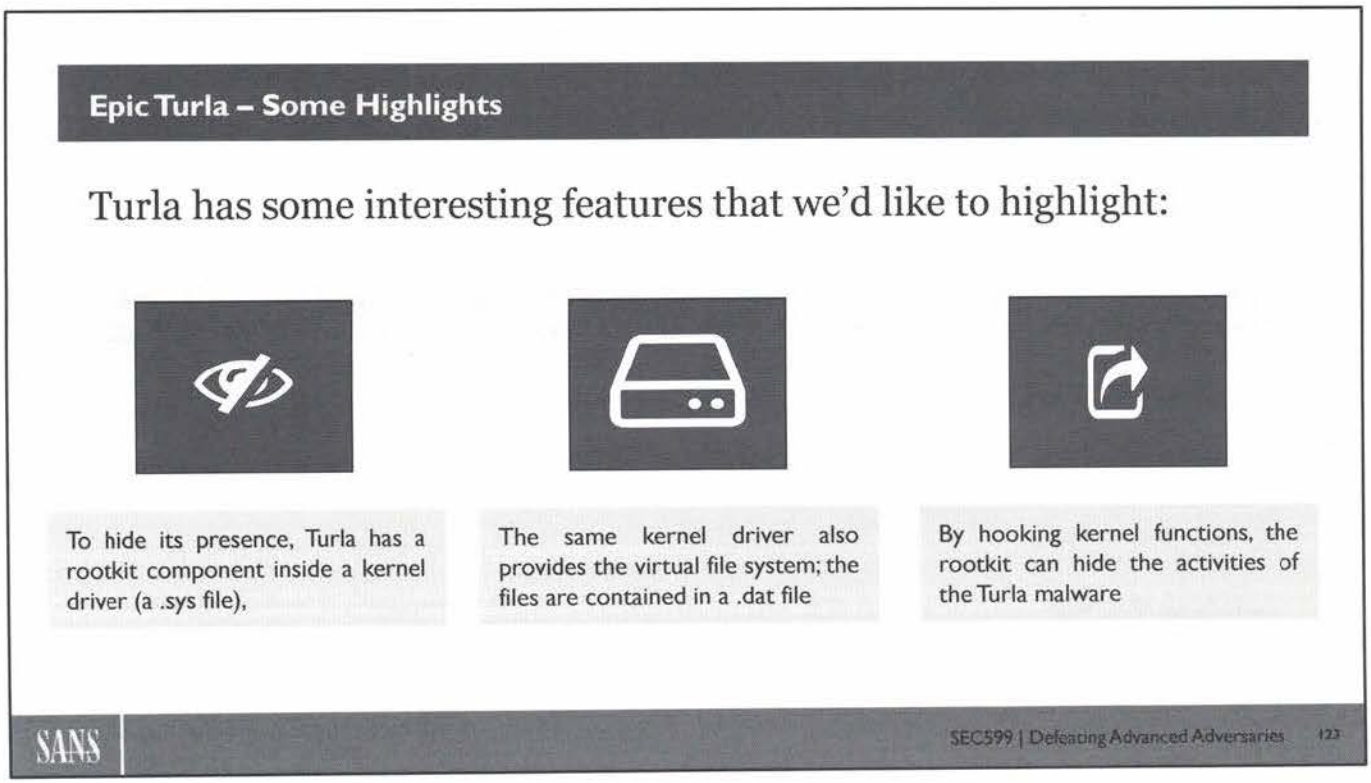


Turla & the APT Attack Cycle

So, let's analyze Turla by using the APT attack cycle:

- Reconnaissance: Standard reconnaissance performed against interesting targets (government agencies);
- Weaponization: Uses PDF's or .SCR files, at least two 0-days were used during the weaponization phase;
- Delivery: Payloads delivered both via watering hole attacks (Java applets) & spear phishing (PDF, SCR…);
- Exploitation: Browser or PDF software is exploited upon opening of attachment or visiting of URL;
- Installation: Uses encrypted filesystem and rootkit to hide its presence;
- Command & Control: The C&C channel was set up using Peer-to-Peer connectivity (for air-gapped hosts, using SMB named pipes) and the HTTP protocol
- Action on Objectives: Lateral movement in the environment looking for interesting policy documentation. Keywords the adversaries were looking for include "NATO", "EU", "Budapest"…

References:
https://securelist.com/analysis/publications/65545/the-epic-turla-operation/



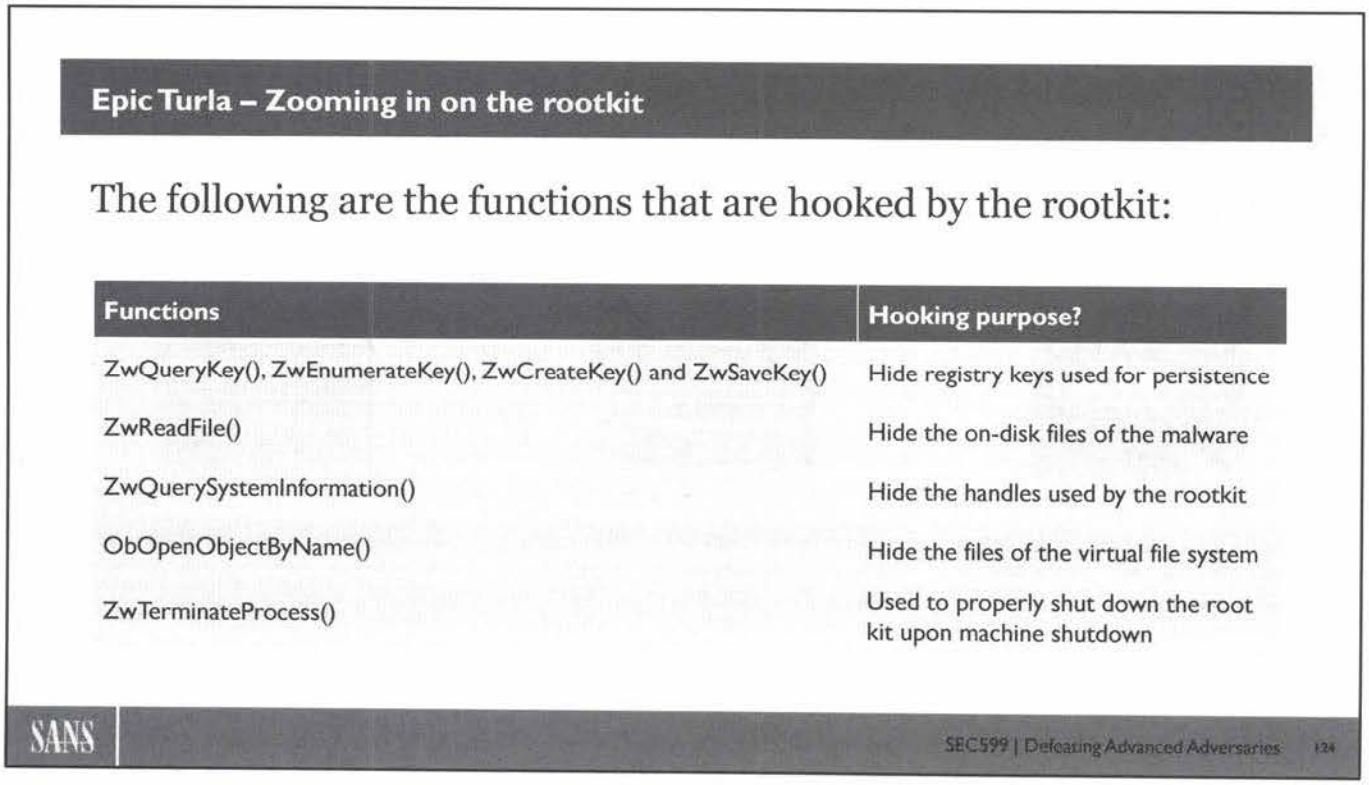
Epic Turla – Some Highlights

The Turla malware has a rootkit component, implemented inside a kernel driver. This kernel driver also implements the API for the encrypted virtual file system, the content of this VFS is stored in a .dat file located in the same directory as the .sys file. At system startup, the rootkit is started via a service named Ultra3.

A rootkit hides the activity of malware by intercepting calls to API functions that provide access to system resources, like reading files, and changes the data returned by these functions to hide resources that indicate the presence of malware (like the Ultra3 registry key used to achieve persistence through a service).

The kernel driver also injects DLLs into processes in user-land.

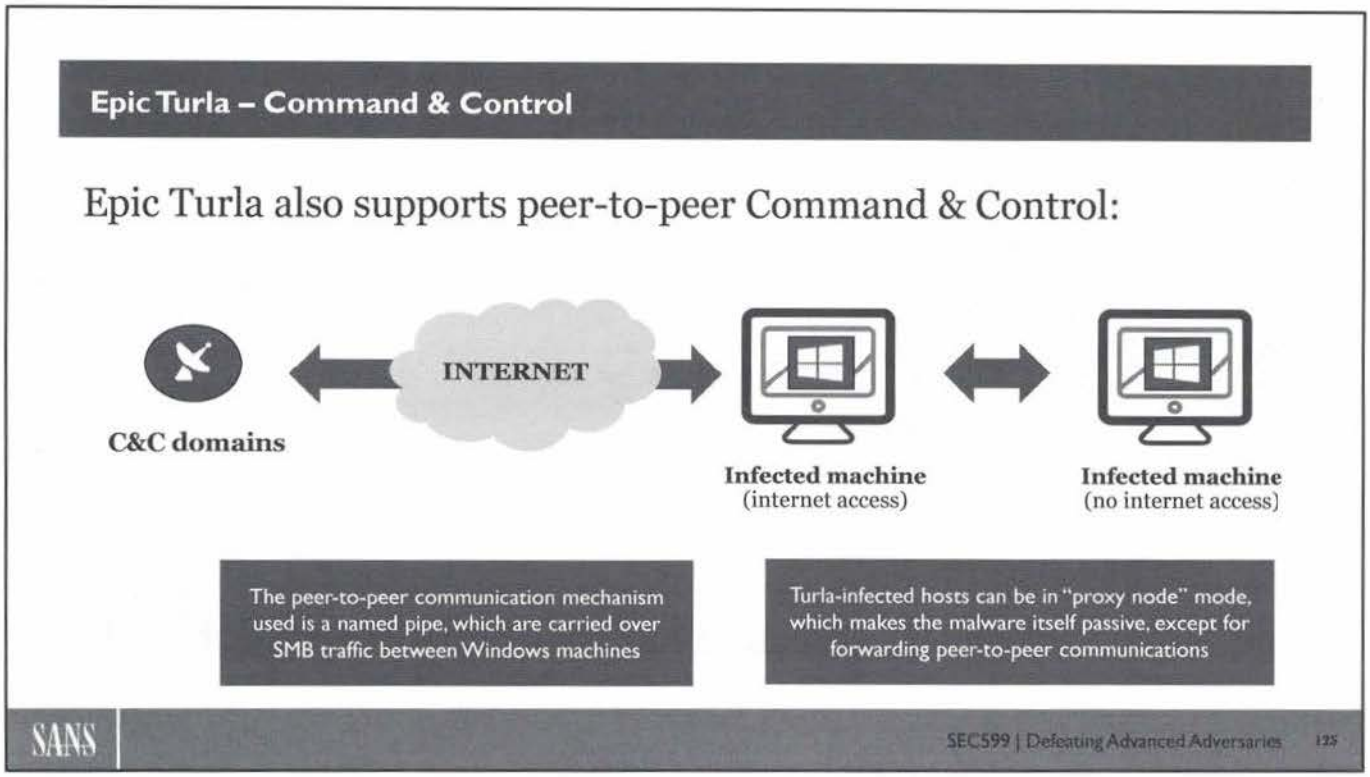
Turla hooks the API functions by patching the functions inline: a jump is executed to the malicious function which does the filtering (hiding) and then returns to the legitimate function.



Epic Turla – Zooming in on the rootkit

By hooking the above API functions, Turla's rootkit component can hide the presence of Turla on the infected Windows machine:

- Hooking registry API functions ZwQueryKey(), ZwEnumerateKey(), ZwCreateKey() and ZwSaveKey() allows the rootkit to hide the registry keys used for persistence (like Ultra3).
- File system API function ZwReadFile() is hooked to hide the files of the Turla malware: the device driver and the virtual file system container files. These are volumes \\.\Hd1 and \\.\Hd2.
- By hooking API function ZwQuerySystemInformation(), Turla can hide the handles used by the rootkit.
- ObOpenObjectByName() is another API function that is hooked to hide the files of the virtual file system.
- API function ZwTerminateProcess() is hooked for another purpose than to hide resources of the rootkit: this function is hooked to be able to shut down the rootkit cleanly (like closing the files of the virtual file system) when the system is shut down.

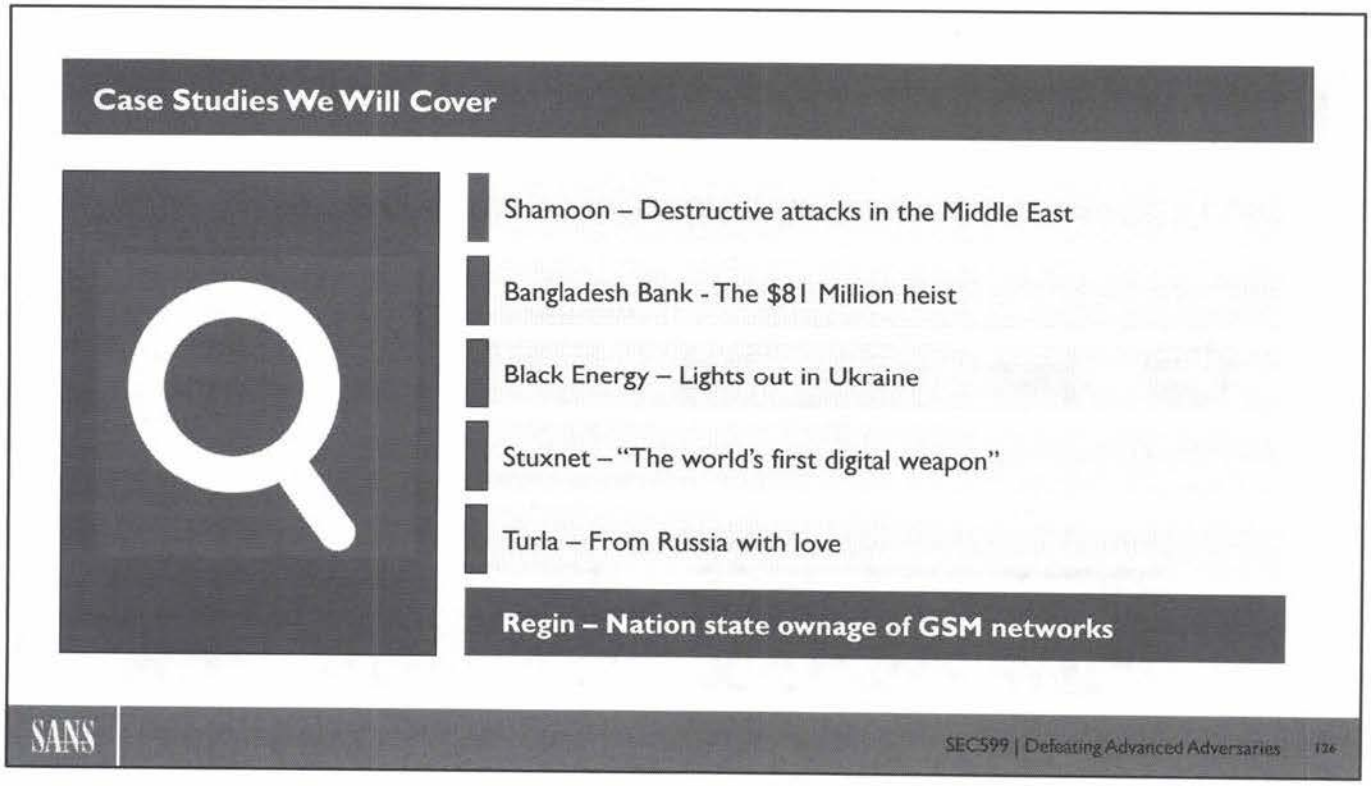


Epic Turla – Command & Control
Similar to Stuxnet, Turla is capable of peer-to-peer networking: this feature can be used to exfiltrate data stolen from a machine without Internet connection to a command and control server of the adversaries via another Turla infected machine with Internet connection.

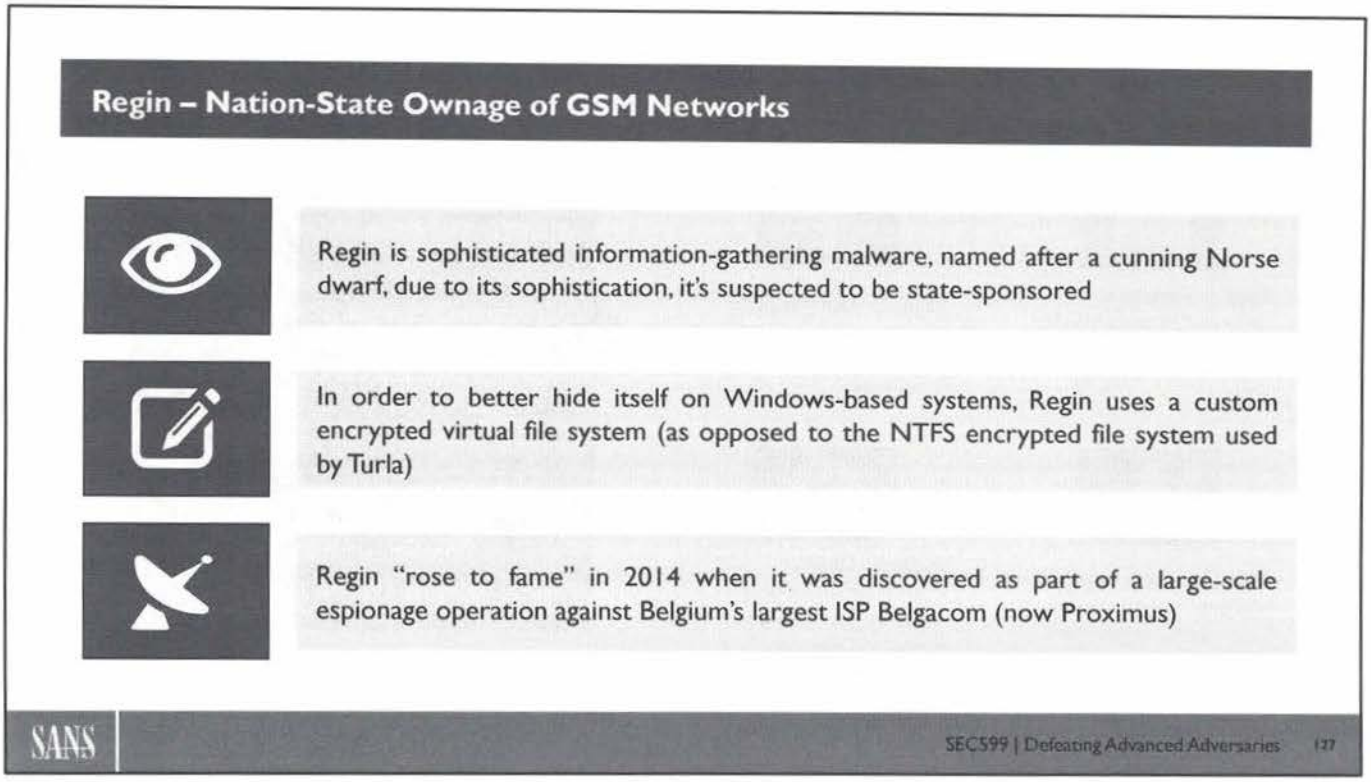
The peer-to-peer communication mechanism used is a named pipe. A named pipe is a Windows inter-process communication medium. Named pipes can be used to transmit data, received data or both receiving and transmitting. It can be used between two or more Windows processes. These processes can be on the same machine, or on different machines.

The machine without Internet connection is the "spied-on node", and the machine with Internet connection is the "proxy node". In "proxy node" mode Turla is passive: it will not actively spy or conduct other malicious activities, just relay data from its peers to the command and control server. This passive e-behavior makes it harder to detect "proxy nodes".

The advantage to the adversaries of using a named pipe as a communication mechanism between infected machines is that named pipes traffic is carried over SMB between Windows machines. This makes that the malicious peer-to-peer traffic blends in with legitimate traffic between Windows machines.



This page intentionally left blank.



Regin – Nation-State Ownage of GSM Networks

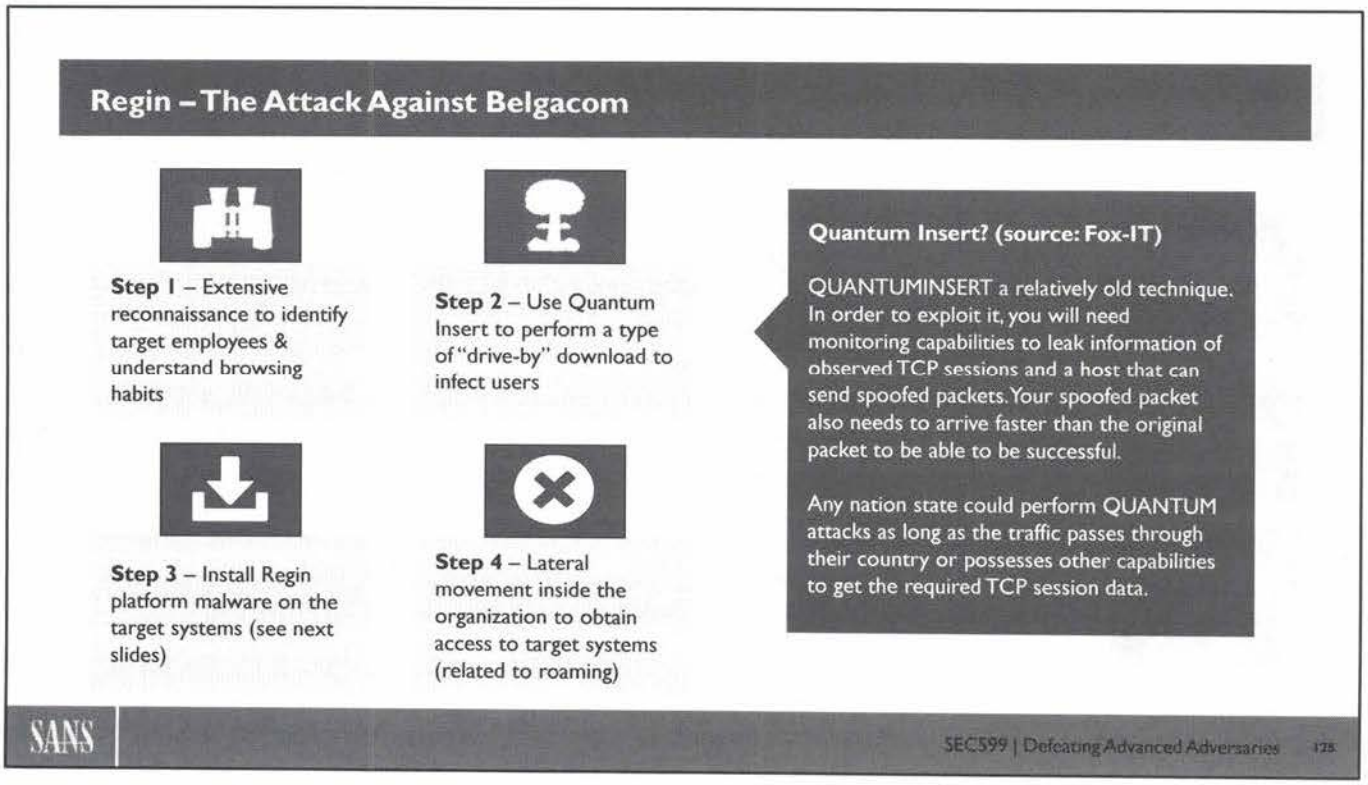
Regin is a highly sophisticated type of information gathering malware. It was named after a cunning Norse dwarf. Furthermore, Regin is IN REGistry switched around. As Regin hides its different stages in registry and extended attributes, detection of the malware is highly difficult. The delivery method of the first stage is not known, but no exploits seem to have been involved. As the first stage does not exhibit particular malicious behavior, traditional file-based detection methods fail to detect it.

A particular feature of Regin is the use of a custom encrypted virtual file system. Apart from the stage 1 loaders, Regin does not write directly to the Windows file system. Instead, it creates single files that are containers for a virtual file system. The content is encrypted, and the files inside the VFS are identified with numbers, not names.

It was first discovered in 2012 and believed to have samples as early as 2003. The goals of Regin are intelligence gathering and facilitating attacks. It has been found to target telecom operators, government institutions, political, financial and research institutions and cryptographers. A well-known case of Regin is the attack on Jean-Jacques Quisquater. Quisquater, a well-known Belgian cryptographer, disclosed in February 2014 that he was the victim of a sophisticated cyber attack. Kaspersky confirmed that the samples were of the Regin platform. Another case that generated a lot of exposure is the attack on systems of Belgacom, Belgium's largest telecom operator.

Victims of Regin have been found in 14 countries: Russia, European countries and Asian countries. Due to its sophistication and its targets, it is widely believed to be developed by state-sponsored attacking groups.

References:
https://securelist.com/files/2014/11/Kaspersky_Lab_whitepaper_Regin_platform_eng.pdf
https://en.wikipedia.org/wiki/Regin_(malware)

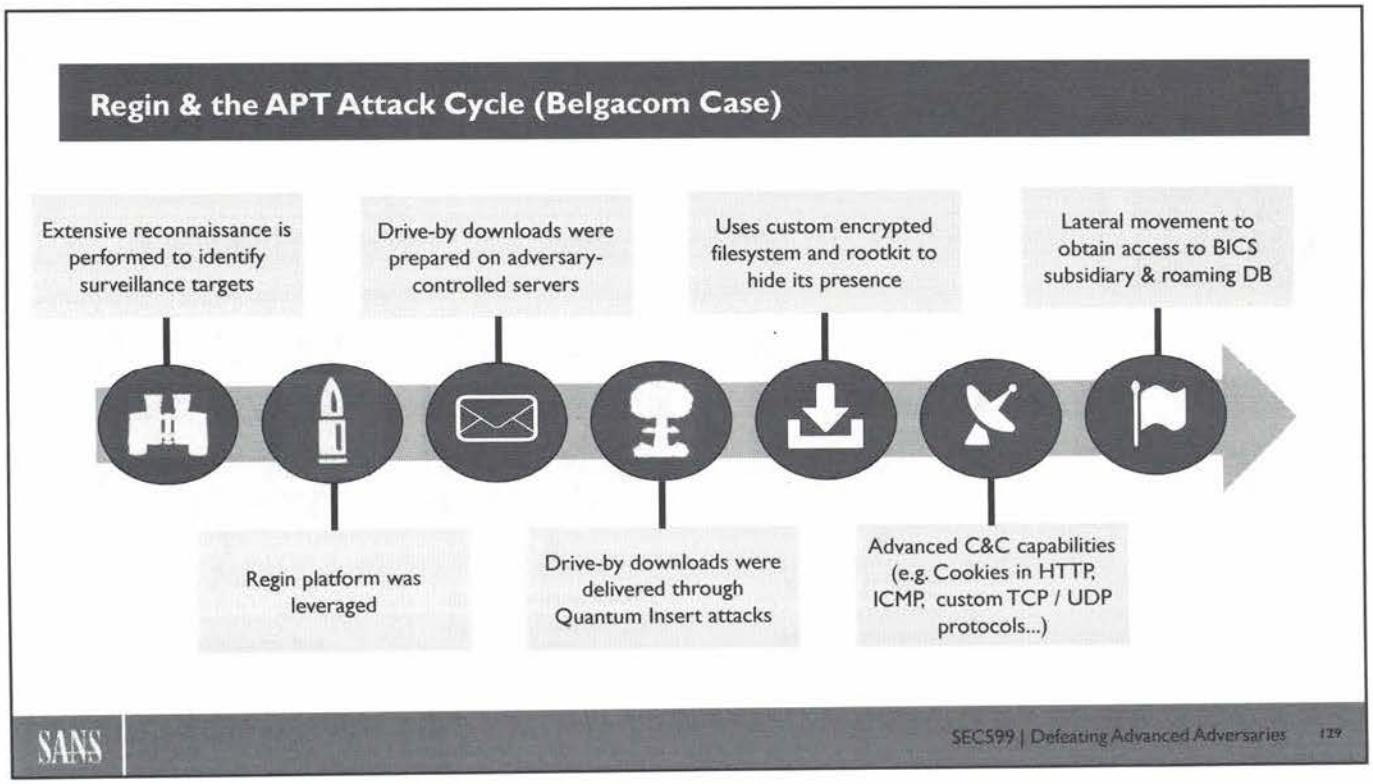


Regin – The Attack Against Belgacom
Let's analyze how the attack against Belgacom took place:

Step 1 – Extensive reconnaissance to identify target employees & understand browsing habits
Step 2 – Use Quantum Insert to perform a type of "drive-by" download to infect users
Step 3 – Install Regin platform malware on the target systems (see next slides)
Step 4 – Lateral movement inside the organization to obtain access to target systems (related to roaming)

So, what is a Quantum Insert? A "QUANTUMINSERT" is a relatively old technique. In order to exploit it, you will need monitoring capabilities to leak information of observed TCP sessions and a host that can send spoofed packets. Your spoofed packet also needs to arrive faster than the original packet to be able to be successful. Any nation state could perform QUANTUM attacks as long as the traffic passes through their country or possesses other capabilities to get the required TCP session data.

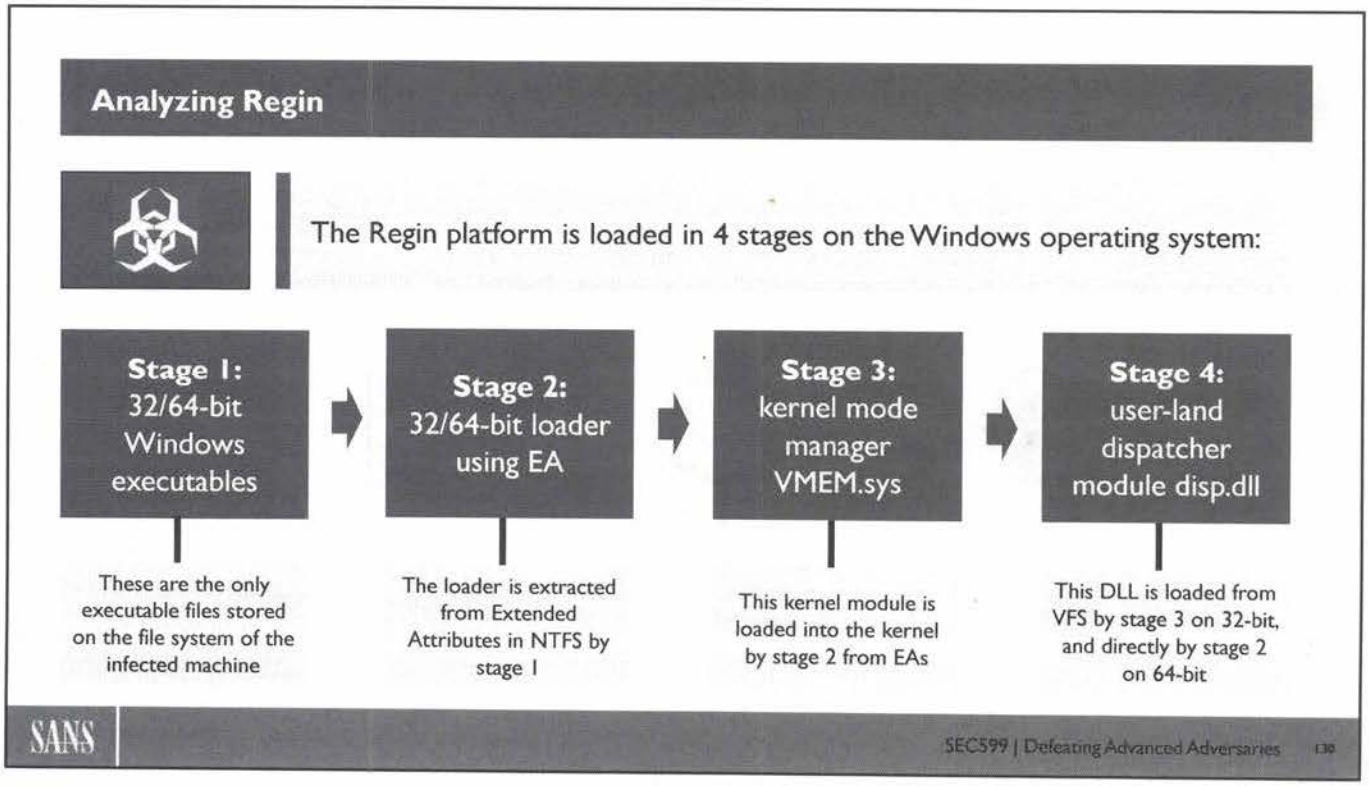
In the next few slides, we will analyze the Regin platform malware in-depth!



Regin & the APT Attack Cycle (Belgacom Case)
So, let's analyze Regin by using the APT attack cycle:

- Reconnaissance: Extensive reconnaissance is performed to identify surveillance targets (key employees in areas such as maintenance & security);
- Weaponization: Regin platform was leveraged;
- Delivery: Drive-by downloads were prepared hosted on adversary-controlled servers;
- Exploitation: Drive-by downloads were delivered through Quantum Insert attacks;
- Installation: Uses custom encrypted filesystem and rootkit to hide its presence
- Command & Control: Advanced C&C capabilities (e.g. Cookies in HTTP, ICMP, custom TCP / UDP protocols...)
- Action on Objectives: Lateral movement to obtain access to BICS subsidiary & roaming DB

References:
https://theintercept.com/2014/12/13/belgacom-hack-gchq-inside-story/
https://securelist.com/files/2014/11/Kaspersky_Lab_whitepaper_Regin_platform_eng.pdf

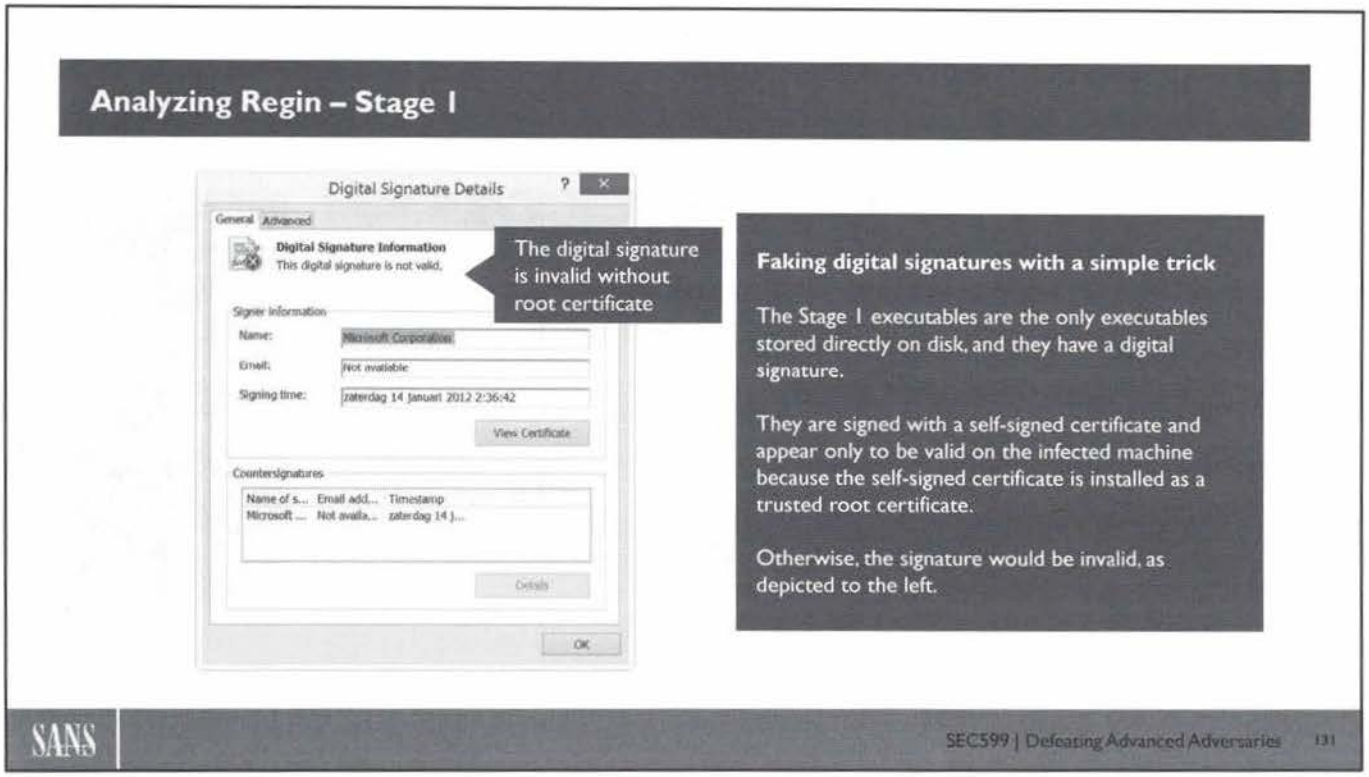


Analyzing Regin
The Regin platform operates in 4 stages:

Stage 1: 32/64-bit Windows executables
Stage 2: 32/64-bit loader
Stage 3: kernel mode manager VMEM.sys
Stage 4: dispatcher module disp.dll and encrypted virtual file system

The Windows executables in the first stage, try to load the executables for the second stage and next stages from Extended Attributes in Windows' NTFS file system. Extended attributes were originally implemented in Windows to provide compatibility with OS/2 applications (similar to Alternate Data Streams that were implemented to provide compatibility with Apple's file system). Storing the next stages in EA makes them harder to detect: the executables, split over several EA blocks, are joined, decrypted and executed in memory.

Until 2012, this method of operation was unique to Regin.



Analyzing Regin – Stage 1

Another particular feature of the stage 1 executables of the Regin platform is the use of fake digital signatures. The executables are signed with a self-signed digital certificate claiming to be from well-known software manufacturers like Microsoft or Broadcom. Self-signed certificates are not recognized by Windows, and a check of the signature would reveal that the signature is not valid. However, the adversaries installed the self-signed root certificate in the trusted root certificates store of the infected Windows machines, thereby configuring Windows to accept all executables signed with this certificate as valid.

As defenders, this offers us an opportunity to detect intrusions in our Windows systems. The Windows trusted root certificates stored should be monitored and alerts generated when new root certificates appear.

The second stage loads the VMEM.sys kernel module.

Analyzing Regin – Stage 3 (vmem.sys kernel module)

Id	Module Description
1	Core framework functionality
7	API for manipulating the encrypted virtual file system (VFS)
13	UCL library for compression and decompression using the nrv2 family of algorithms
15	RC5 encryption and decryption facilities
61	API for manipulating the encrypted virtual file system (VFS)
50111	Utilities
50215	System information
50225	Module notification routines
50223	API for code injection and kernel-mode hooking

The kernel driver vmem.sys

Regin is a highly modular malware platform. In the table to the left, we see a nonexhaustive lists of modules (or plugins) available for installation in the kernel.

Analyzing Regin – Stage 3

Stage 3 is loaded into memory by stage 2 from the EA in the NTFS file system.

VMEM.sys is a kernel driver: this code runs in the Windows kernel. It offers different basic functionalities for the Regin platform via modules with different APIs. Modules are identified by a number.

Modules 7 and 61 offer APIs for the encrypted virtual file system. The encryption is done with APIs in module 15: a modified RC5 algorithm.

After initialization, it can load additional plugins from the encrypted virtual file system. One of the modules it will load from VFS is module 50221. This is the dispatcher module, disp.dll. The dispatcher module is phase 4 of the loading and is the user-land presence of the Regin platform.

Analyzing Regin – Stage 4 (Dispatcher disp.dll)

Id	Module Description
1	Core framework functionality
7	API for manipulating the encrypted virtual file system (VFS)
11	File writer
13	UCL library for compression and decompression using the nrv2 family of algorithms
15	RC5 encryption and decryption facilities
17	In-memory storage object
19	Configuration storage object
25	Network transport using packet filters
61	API for manipulating the encrypted virtual file system (VFS)

The user-land dispatcher

The user-land dispatcher (stage 4) is also modular. The modules are listed in the table to the left (non-exhaustive list).

They provide an interesting insight in Regin's capabilities.

Analyzing Regin – Stage 4

Stage 4 is the dispatcher loader into user-land, on 32-bit systems, it is loaded from the VFS by stage 3, and on 64-bit systems, it is directly loaded by stage 2.

We encounter similar modules as found in the kernel driver vmem.sys, like modules 7, 15 and 61. This is because this functionality needs to be available both to code running in user-land and code running in kernel-land. Code running in user-land cannot directly run code (using APIs) in kernel-land, and vice-versa. It also illustrates the sophistication and attention to secure design of the developers of the Regin platform.

The dispatcher is, in essence, the "kernel" of the entire Regin platform.

Analyzing Regin – Encrypted File System

The Regin malware platform uses an encrypted virtual file system, stored inside regular Windows files on the NTFS file system

Position	Size in bytes	Description
0x0000	2	Sector size
0x0002	2	Maximum number of sectors
0x0004	2	Maximum number of files
0x0006	1	Unknown
0x0007	4	CRC32 of first seven bytes of the header with seed 0x45
0x000B		
...		

Header of Regin's VFS container

The header of a Windows file containing Regin's VFS (shown left) is not encrypted (only the virtual file content is).

Because of the unique 4-byte CRC32 code at position 7, Regin's VFS header is quite unique.

Detection rules for the first 11 bytes of the header have been developed, allowing for the scanning of infected machines.

Analyzing Regin – Encrypted File System
In its 4th stage, Regin uses an encrypted virtual file system.

The file system is contained in regular Windows files, often stored in folder c:\windows\system32 or subfolders, and with innocuous names like cdata.dat, dnscache.dat, …

One particular file discovered during the analysis of Regin infected computers, pertains to GSM communications. It is a log of a Base Station Controller, appearing to contain Ericsson OSS MML commands, hinting at a capability of Regin to monitor GSM systems.

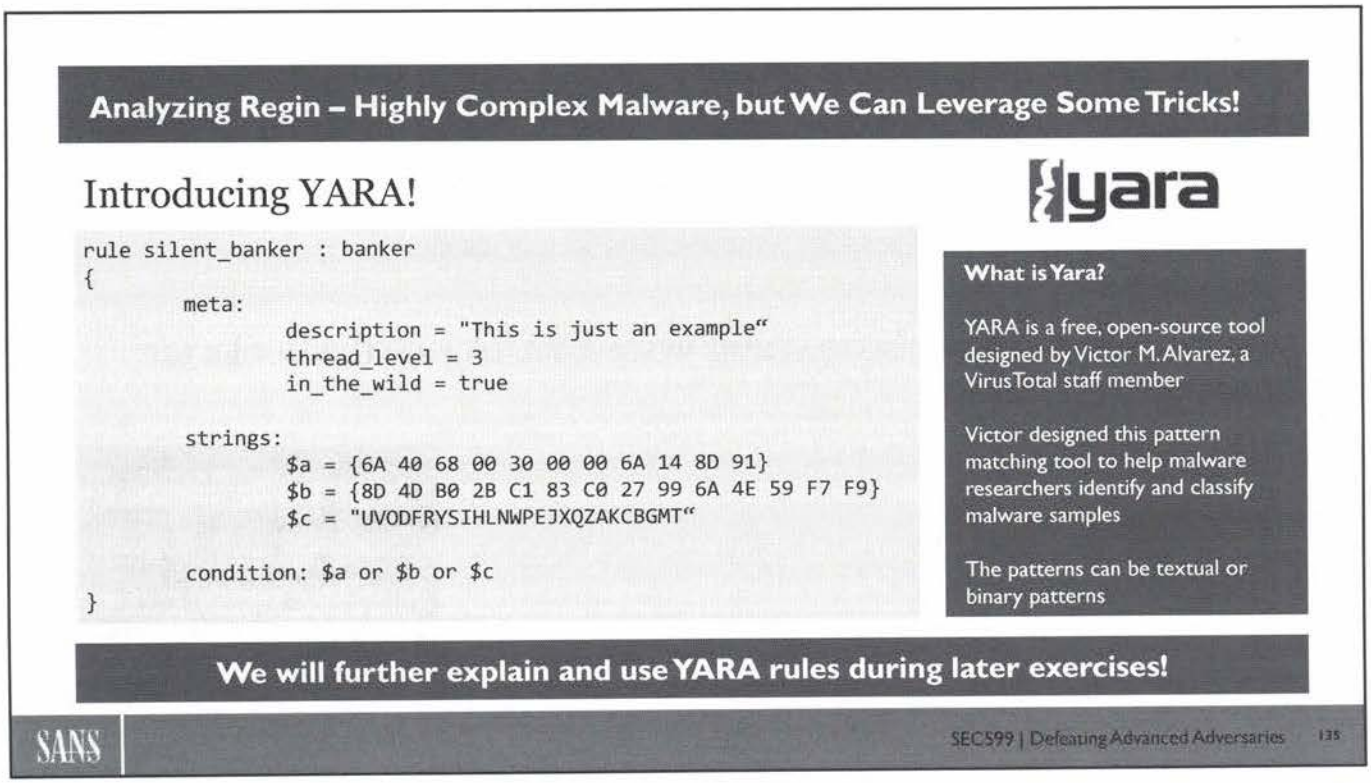
The VFS uses an internal structure very similar to the FAT file system. The VFS starts with a header providing information necessary to parse and process the virtual file system, like:

- Sector size
- Maximum number of sectors
- Maximum number of files
- A CRC32 field with seed 0x45 calculated over the first 7 bytes of the header
- …

This header is followed by a sector allocation table and a file allocation table. The VFS contains encrypted files, the structures of the VFS itself are actually not encrypted. RC5 is the encryption algorithm used by the Regin platform, and often files are first compressed with the nrv2e algorithm of the UCL library (found in module 13).

Kaspersky discovered that most files in the VFS found on machines it examined were encrypted with the same RC5 key: 73 23 1F 43 93 E1 9F 2F 99 0C 17 81 5C FF B4 01. This key can also be found in the kernel module VMEM.sys.

Such a key is a pretty unique byte sequence and a good IOC to detect Regin on compromised systems. We will illustrate this with the YARA tool.



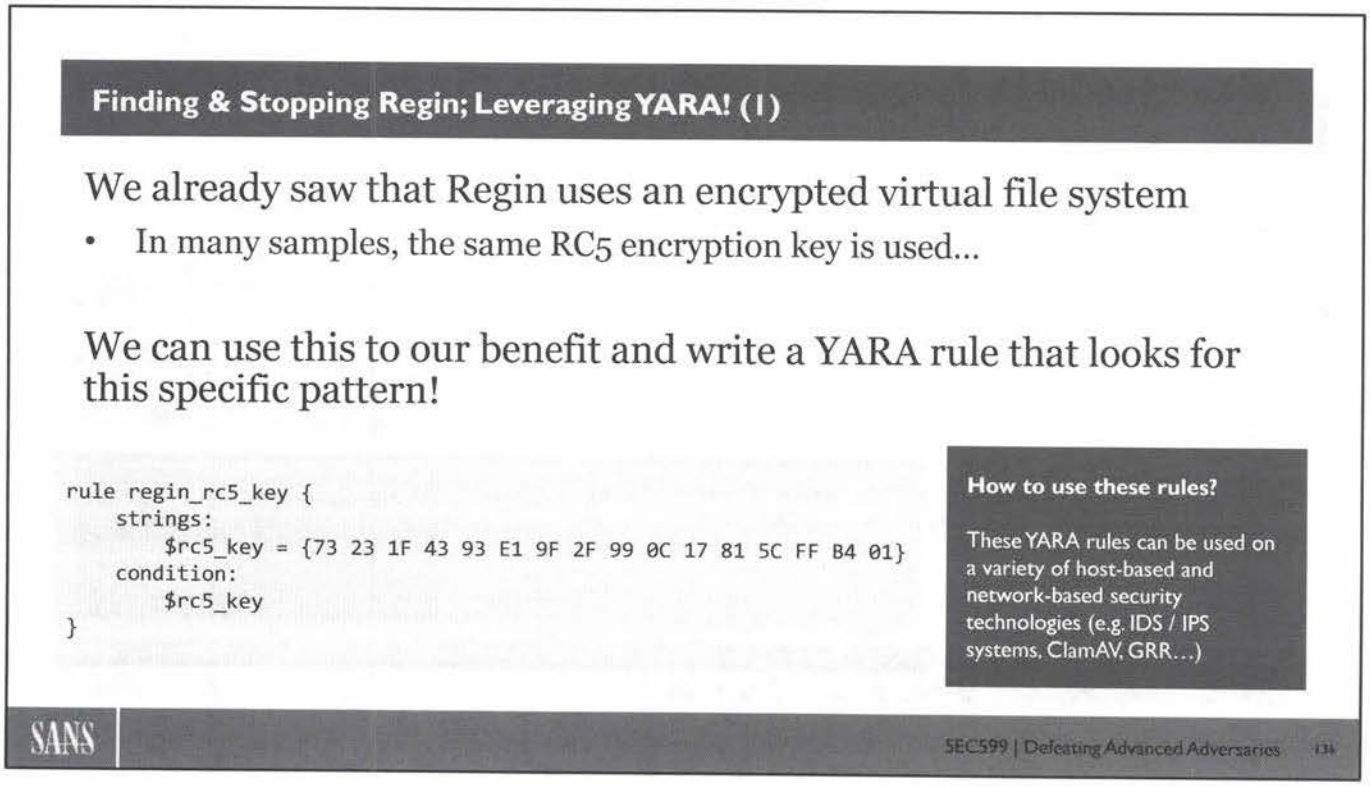
Analyzing Regin – Highly Complex Malware, but We Can Leverage Some Tricks!
YARA is a free, open source tool designed by Victor M. Alvarez, a VirusTotal staff member. Victor designed this pattern matching tool to help malware researchers identify and classify malware samples.

YARA works with rules, written by the user or third parties like malware researchers. Rules are contained in text files according to the YARA rule language. It has grown to a flexible language with many features, but it is essentially a language that allows for the specification of strings and/or byte sequences that have to be found in the sample for the rule to trigger.

YARA runs on different operating systems (Windows, Linux, OSX …), and can scan files and the memory of running processes. It has also been integrated into other tools like the open source anti-virus program ClamAV. The advantage of using a tool like ClamAV for hunting with YARA rules is that ClamAV offers many file decomposition features. For example, ClamAV will decompress and scan inside archive files like ZIP, while the original YARA tool will not.

We will now illustrate how to use YARA to hunt for Regin.

References:
http://virustotal.github.io/yara/
https://blog.nviso.be/2017/02/14/hunting-with-yara-rules-and-clamav/
https://github.com/Yara-Rules/rules



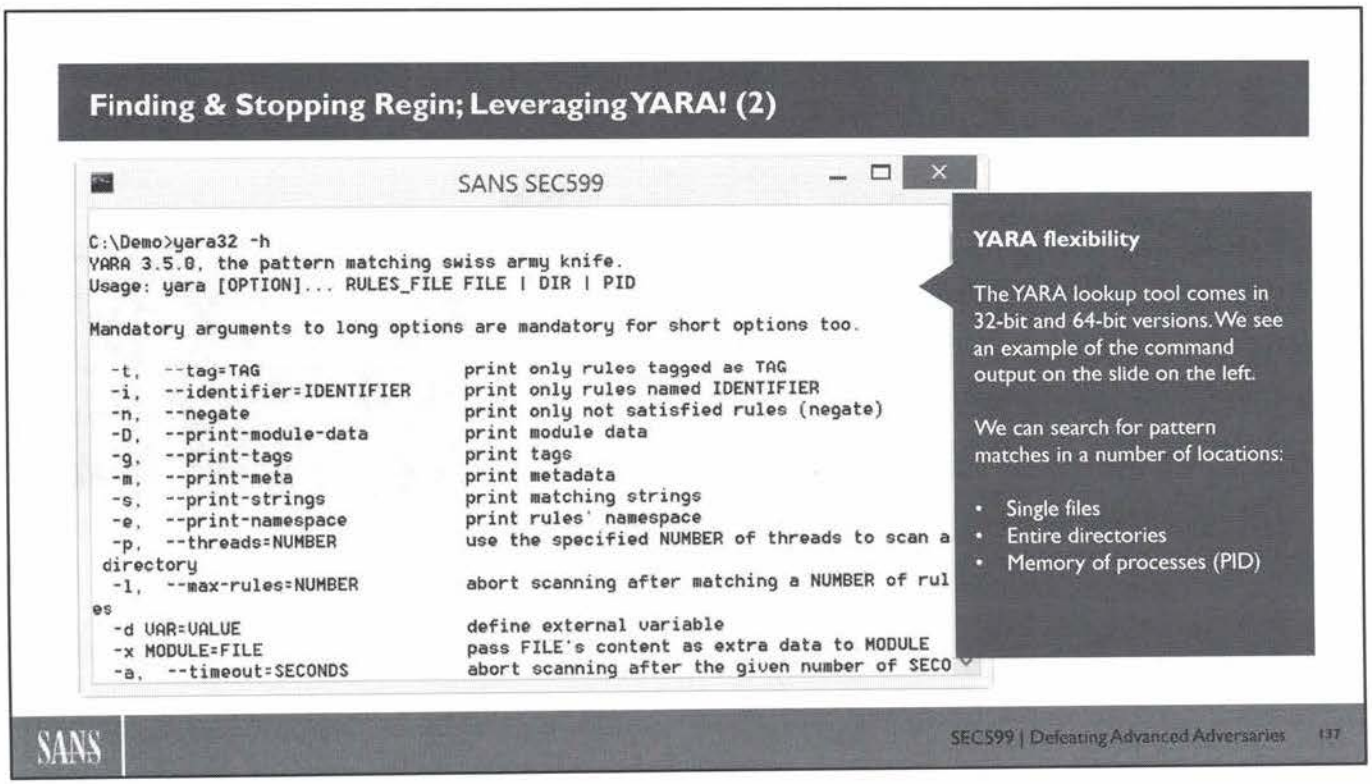
Finding & Stopping Regin; Leveraging YARA! (1)

We already saw that Regin uses an encrypted virtual file system, as the same RC5 encryption key is used in several different samples! We can use this to our benefit and write a YARA rule that looks for this encryption key pattern!

Several YARA rules can be stored inside a text file. In this example, we have one rule. To identify YARA rules, we give them names. A typical YARA rule starts with the keyword *rule* followed by the name of the rule and the body of the rule. In our example, the name is regin_rc5_key. The body of a YARA rule is delimited by curly braces: {}

The body of a rule will mention strings to search for (under the header *strings:*) and the condition that needs to be fulfilled for the rule to trigger (under the header *condition:*).

In our rule for the Regin RC5 encryption key, the "string" to search for is not an ASCII string, but a sequence of bytes. A sequence of bytes can be represented by hexadecimal values enclosed in curly braces. The rule will trigger if the $rc5_key string is found at least once.

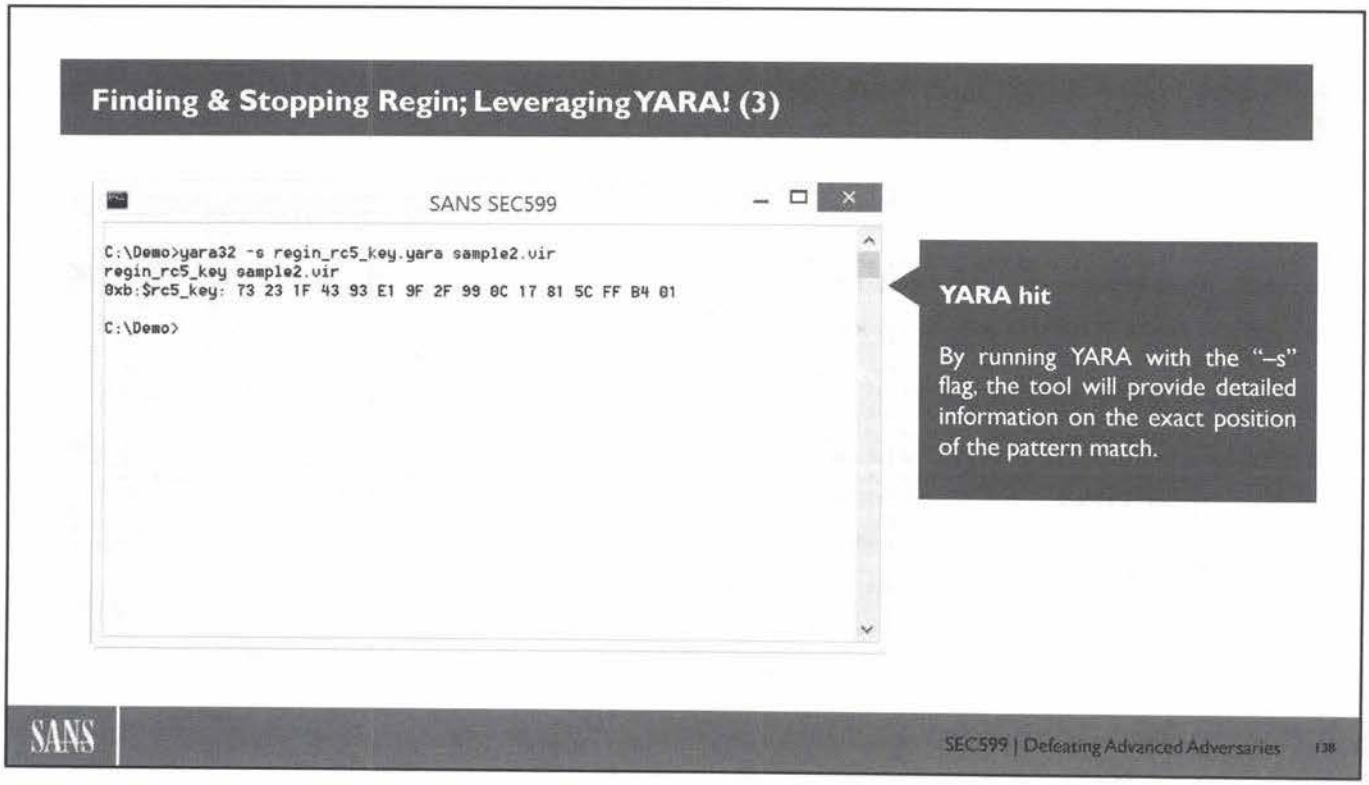


Finding & Stopping Regin; Leveraging YARA! (2)

The YARA tool comes in a 32-bit and 64-bit version (on Windows: yara32.exe and yara64.exe). The version used here is 3.5.0. YARA takes options and 2 arguments: the name of the text file with the YARA rules, and the name of the target to scan. This can be a file, a directory or a process.

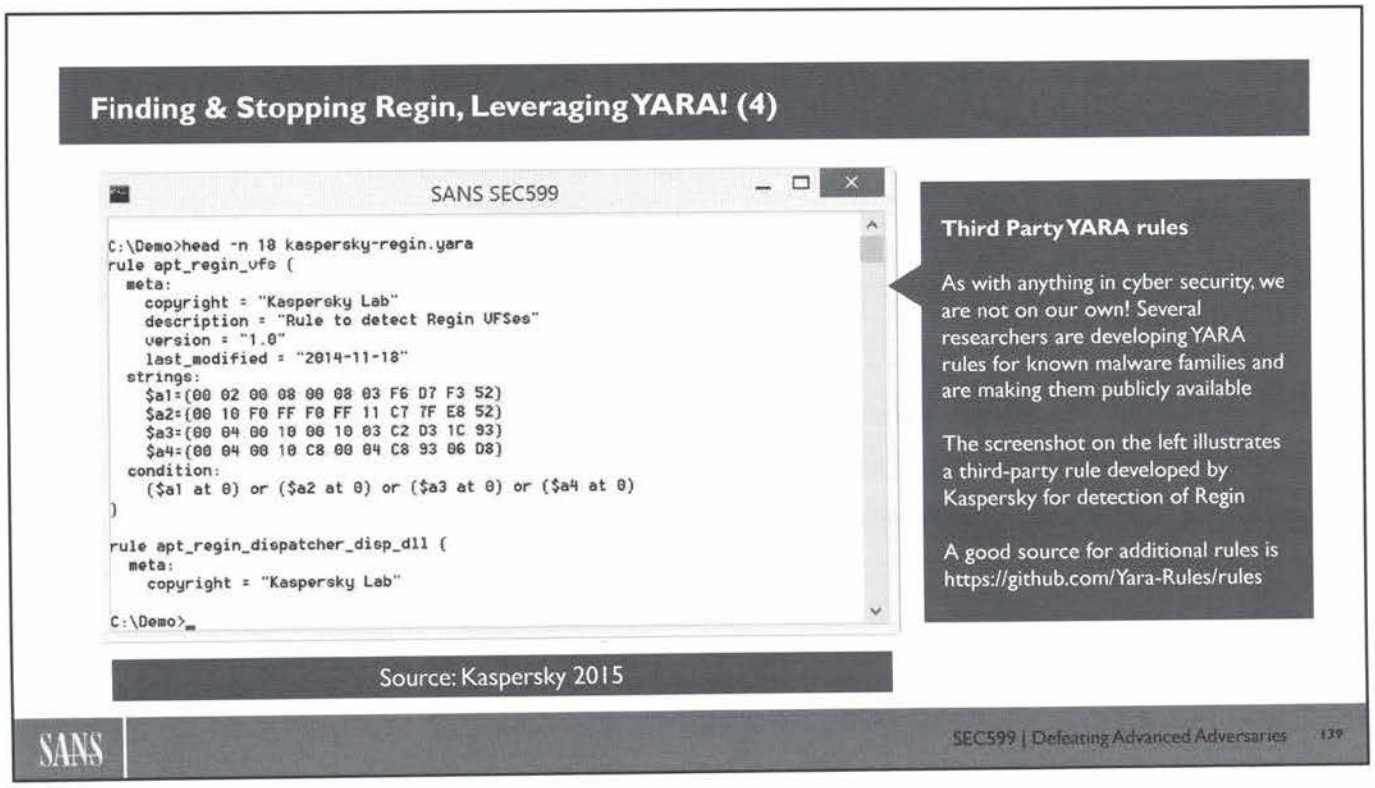
Usually, when we want to scan a machine, we will not scan individual files, but scan the full disk. YARA supports this with option –r: recursive. When we provide a directory to scan and use option –r, YARA will scan all files in that directory and all subdirectories.

Remember that the RC5 key is stored in vmem.sys, which is hidden in Extended Attributes of the NTFS file system. It's unlikely that we will discover this key when simply scanning files, that's why we also scan the memory of processes: to be able to use the RC5 key, the RC5 key must be in memory. The YARA tool can also be used to scan the memory of processes (running programs).



Finding & Stopping Regin; Leveraging YARA! (3)
With option –s, we can instruct yara32 to provide details when a rule triggers.

After the line with the rule and file name, YARA outputs the position, variable name and value of all strings/byte sequences found. Since there is only one line of extra output compared to the previous example, we know that the RC5 key was only found once in the sample. The key is found at position 0xB, and YARA prints out the byte sequence found. This is useful if we search for more than one string or byte sequence with the same rule.



Finding & stopping Regin; Leveraging YARA! (4)
Finally, we want to illustrate third party YARA rules.

Kaspersky is one of the companies providing good YARA rules for malware samples they analyze extensively. Besides including rules to detect the malware itself, they will try to create rules to detect unique artifacts created by the malware.

For Regin, Kaspersky discovered that the headers of files containing the encrypted virtual file systems are quite unique. Remember that the structures used in the virtual file system are not encrypted and that after the first 7 bytes comes a 4-byte long CRC32 checksum. This checksum is a good way to detect these VFS containers.

When we look at the rule apt_regin_vfs, we see that 4 strings are defined. 4 byte sequences, each 11 bytes long (7 data bytes + 4 CRC32 bytes).

The condition will trigger if one (or more) of these byte sequences is found at the beginning of the container file. Kaspersky identified 4 different header values used by Regin for the encrypted virtual file system.

A very good source for YARA rules from a variety of authors is https://github.com/Yara-Rules/rules

References:
https://securelist.com/files/2014/11/Kaspersky_Lab_whitepaper_Regin_platform_eng.pdf
https://github.com/Yara-Rules/rules

Course Roadmap

- **Day 1: Knowing the adversary, knowing yourself**
- Day 2: Averting Payload Delivery
- Day 3: Preventing Exploitation
- Day 4: Avoiding Installation, foiling Command & Control & thwarting lateral movement
- Day 5: Exfiltration, Cyber Deception & Incident Response
- Day 6: APT Defender Capstone

SEC599.1

Course Outline & Lab Setup
- Course Overview & Objectives
- Attendee System Setup

Current Threat / Attack Landscape
- Key Terminology
- What is happening out there?

Introducing the APT Attack Cycle
- Recent Case Studies – In-Depth
- Exercise: Analyzing The Behavior of Famous Malware
- Exercise: One Click Is All It Takes…

A Defensible Architecture & Environment

Preparation - Knowing Yourself
- Understanding Your Own Environment
- Determining What is "Normal"
- Understanding & Limiting Your Internet Footprint
- A Word on Vulnerability Management

This page intentionally left blank.

Exercise – Analyzing the Behavior of Famous Malware

The objective of the lab is to analyze a number of known malware samples from the campaigns we briefly addressed above. This will be your first interaction with the LODS environment, so the lab is designed to "get going" in a comfortable fashion!

High-level exercise steps:

1. Authenticate to the Windows02 machine
2. Mount the ISO containing the malware samples (CAREFUL! - PW "infected")
3. Upload the samples to Cuckoo sandbox
4. Analyze & review the results

Exercise – Analyzing the Behavior of Famous Malware
The objective of the lab is to analyze a number of known malware samples from the campaigns we briefly addressed above (including Regin, WannaCry, Notpetya, Shamoon…). This will be your first interaction with the LODS environment, so the lab is designed to "get going" in a comfortable fashion!

Throughout the lab, we will rely on the open-source Cuckoo sandbox, which we will further discuss in section 2 of this course. Cuckoo sandbox can be used to upload suspicious files, after which it will perform both a static and dynamic analysis of the file. After its analysis, Cuckoo will provide you with a report that includes both the result of a static analysis (e.g. including the results of strings), but also a dynamic analysis (including a memory dump, file system access, the network connections opened…)

The high-level exercise steps are the following:

- Authenticate to the Windows02 machine
- Mount the ISO containing the malware samples (CAREFUL – password is "infected"!)
- Upload the samples to Cuckoo sandbox
- Analyze & review the results

For additional guidance & details on the lab, please refer to the LODS workbook.

Exercise – Analyzing the Behavior of Famous Malware - Conclusions

During this lab, we analyzed a number of famous / known malware samples, including WannaCry, Notpetya, Regin, Shamoon...

We focused on getting hands-on with the lab environment by using different machines that are available in our environment.

We used Cuckoo as a sandbox to perform dynamic analysis to analyze different samples

Exercise – Analyzing the Behavior of Famous Malware – Conclusions
During this lab, we analyzed a number of famous / known malware samples, including WannaCry, Notpetya, Regin, Shamoon… Throughout the lab, you most likely noticed the different results that can be obtained with Cuckoo. Our Cuckoo instance has been set up with a Windows 7 32-bit guest, without any "advanced" tailoring or configuration, it uses the standard Cuckoo configuration.

During our labs on day 2, we will further work with Cuckoo and analyze its configuration in-depth. We will also assess how malware samples are trying to detect sandboxing.

Please note that this is not a dedicated malware reverse engineering class, so we only touch upon this subject as an introduction. Should you be interested to learn more about this subject, SANS has a course specifically dedicated to malware analysis, namely "FOR610 – Reverse Engineering Malware".

Course Roadmap

- **Day 1: Knowing the adversary, knowing yourself**
- Day 2: Averting Payload Delivery
- Day 3: Preventing Exploitation
- Day 4: Avoiding Installation, foiling Command & Control & thwarting lateral movement
- Day 5: Exfiltration, Cyber Deception & Incident Response
- Day 6: APT Defender Capstone

SEC599.1

Course Outline & Lab Setup
- Course Overview & Objectives
- Attendee System Setup

Current Threat / Attack Landscape
- Key Terminology
- What is happening out there?

Introducing the APT Attack Cycle
- Recent Case Studies – In-Depth
- Exercise: Analyzing The Behavior of Famous Malware
- Exercise: One Click Is All It Takes…

A Defensible Architecture & Environment

Preparation - Knowing Yourself
- Understanding Your Own Environment
- Determining What is "Normal"
- Understanding & Limiting Your Internet Footprint
- A Word on Vulnerability Management

This page intentionally left blank.

Exercise – One Click Is All It Takes... (1)

During this first exercise, we will provide you with hands-on experience on how adversaries typically compromise target environments. You will play the role of adversary and attempt to compromise a fictive organization "synctechlabs.com".

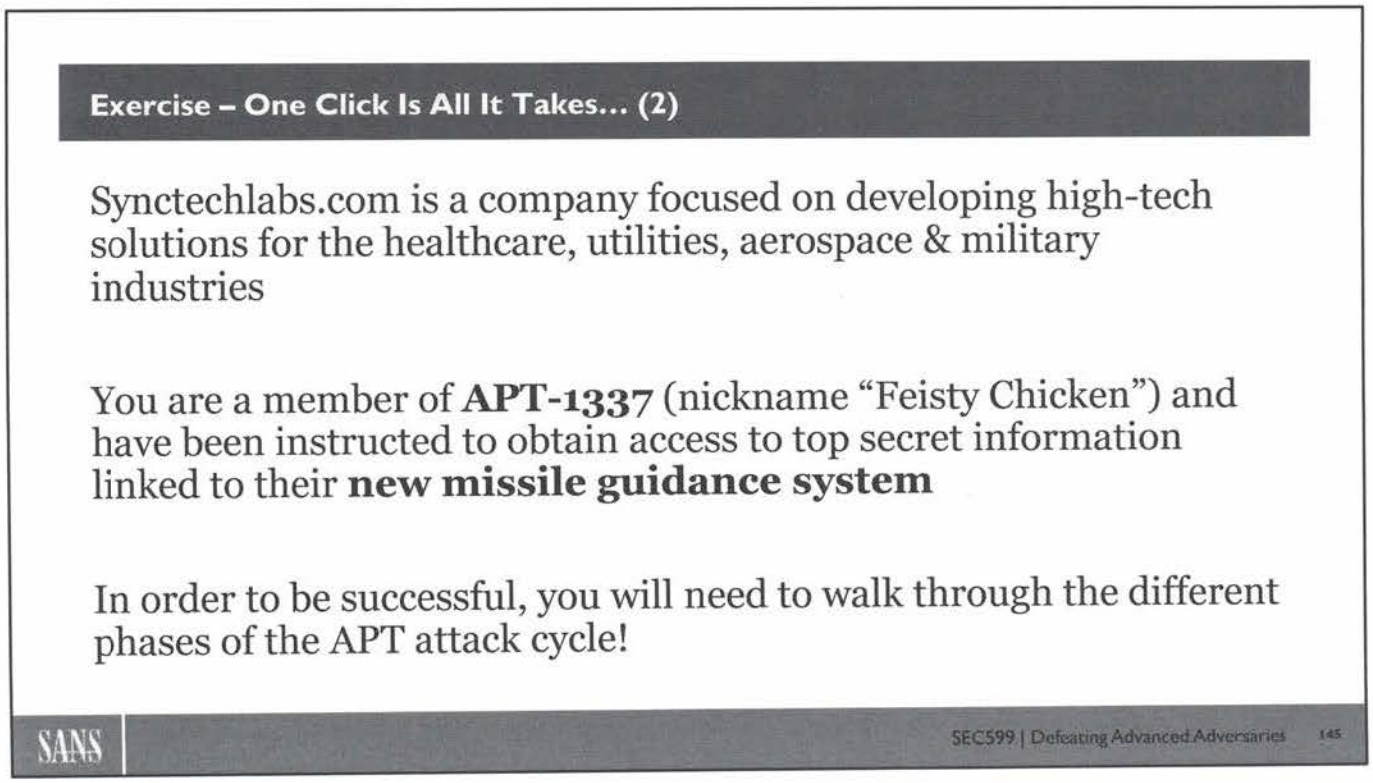
High-level exercise steps:

1. Authenticate to the Kali machine provided to you
2. Walk through the various phases of the Attack Cycle to perform our attack
3. Review controls in place and assess how the attack could have been stopped

Exercise – One Click Is All It Takes... (1)
During this first exercise, we will provide you with hands-on experience on how adversaries typically compromise target environments.
This is of tremendous value if we want to better defend our own organizations.

For the purpose of this exercise, we have developed the online presence of a fictive organization called "Syntechlabs.com. Synctechlabs.com is a company focused on developing high-tech solutions for the healthcare, utilities, aerospace & military industries.

As this is not an offensive course, we will walk you through the different attack steps one by one and guide you on the way… This will be the only **offensive-focused** exercise in the course.

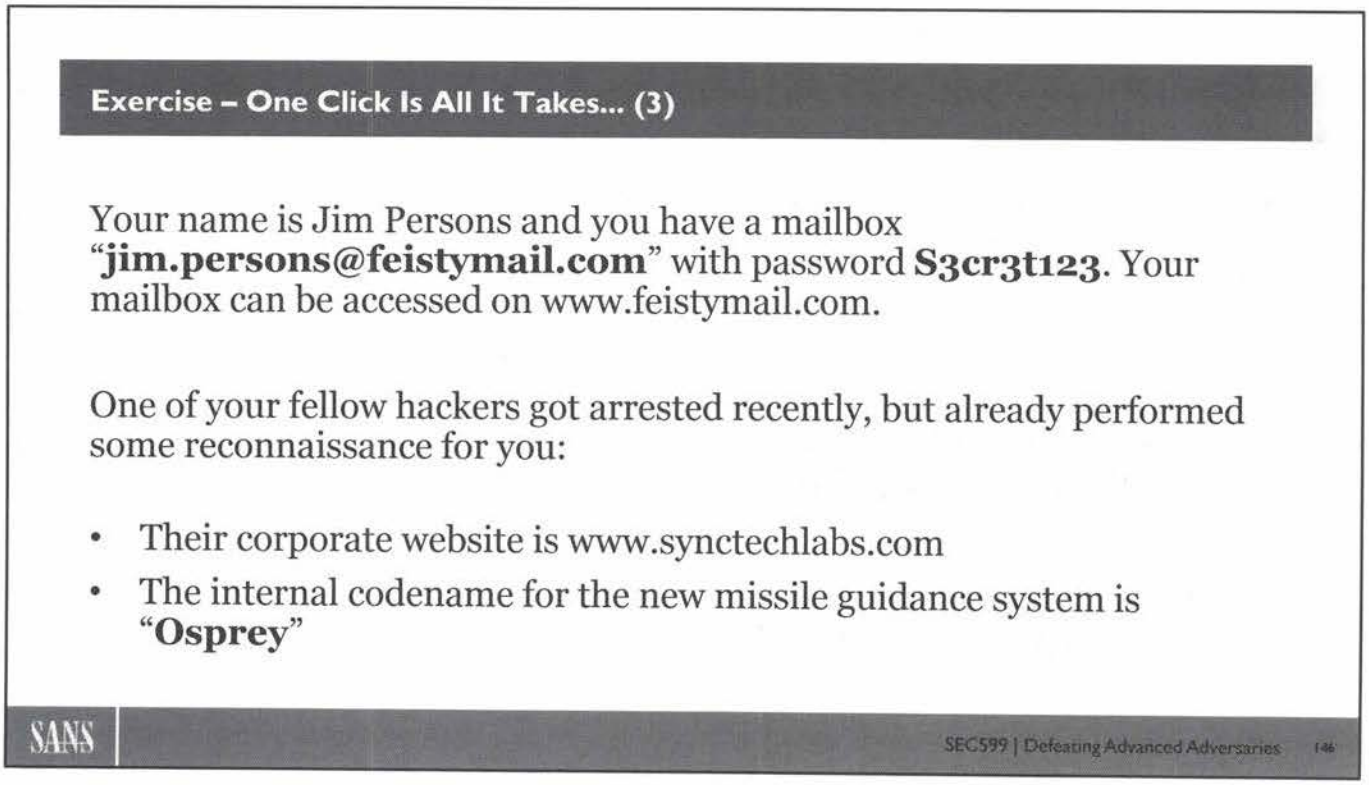


Exercise – One Click Is All It Takes... (2)

Synctechlabs is a typical target for espionage: they focus on developing high-tech solutions for the healthcare, utilities, aerospace & military industries. They have thus attracted some unwanted attention from an APT-group called APT-1337 (nicknamed "Feisty Chicken").

In this lab, you will play the role of one of APT-1337's most skilled hackers and you have been tasked to steal top secret information linked to the new missile guidance system. In order to be successful, you will need to plan and execute your attack carefully… Throughout the lab, you will notice the different steps of the APT attack cycle as you play the adversary role.

Good luck!

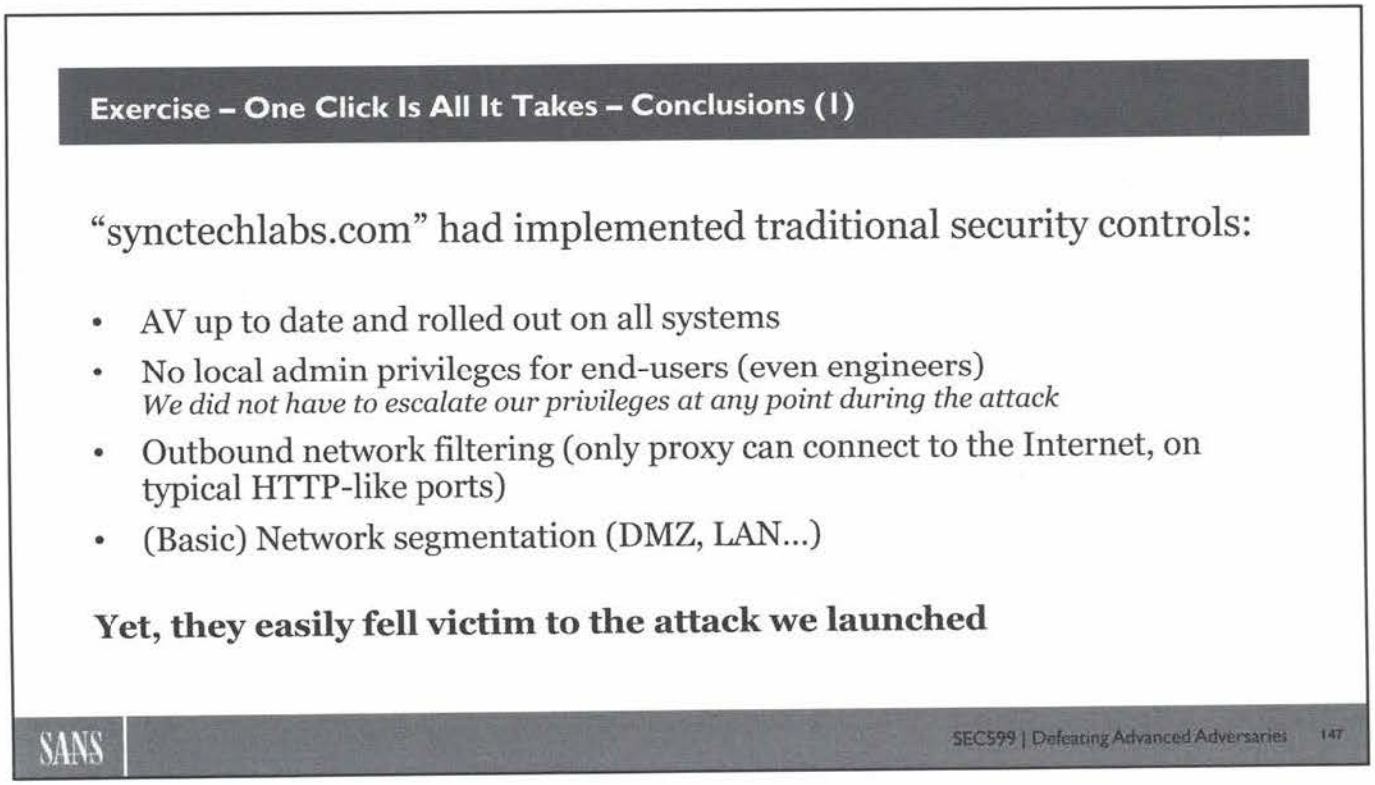


Exercise – One Click Is All It Takes… (3)
Your name is Jim Persons and you have a mailbox "jim.persons@feistymail.com" with password S3cr3t123. Your mailbox can be accessed on www.feistymail.com. As Synctechlabs is a crucial target of your hacking groups, you are not working on this alone! One of your fellow hackers got arrested recently, but already performed some reconnaissance and has obtained the following results

- The corporate website is www.synctechlabs.com
- The internal codename for the new missile guidance system is "Osprey".

The next few slides will walk through the different attack steps in detail. You are free to choose whether you want to peek ahead and follow the attack steps one-by-one or try for yourself how creative you can be throughout the lab. For students without any experience with offensive security techniques, we recommend keeping the materials close-by!

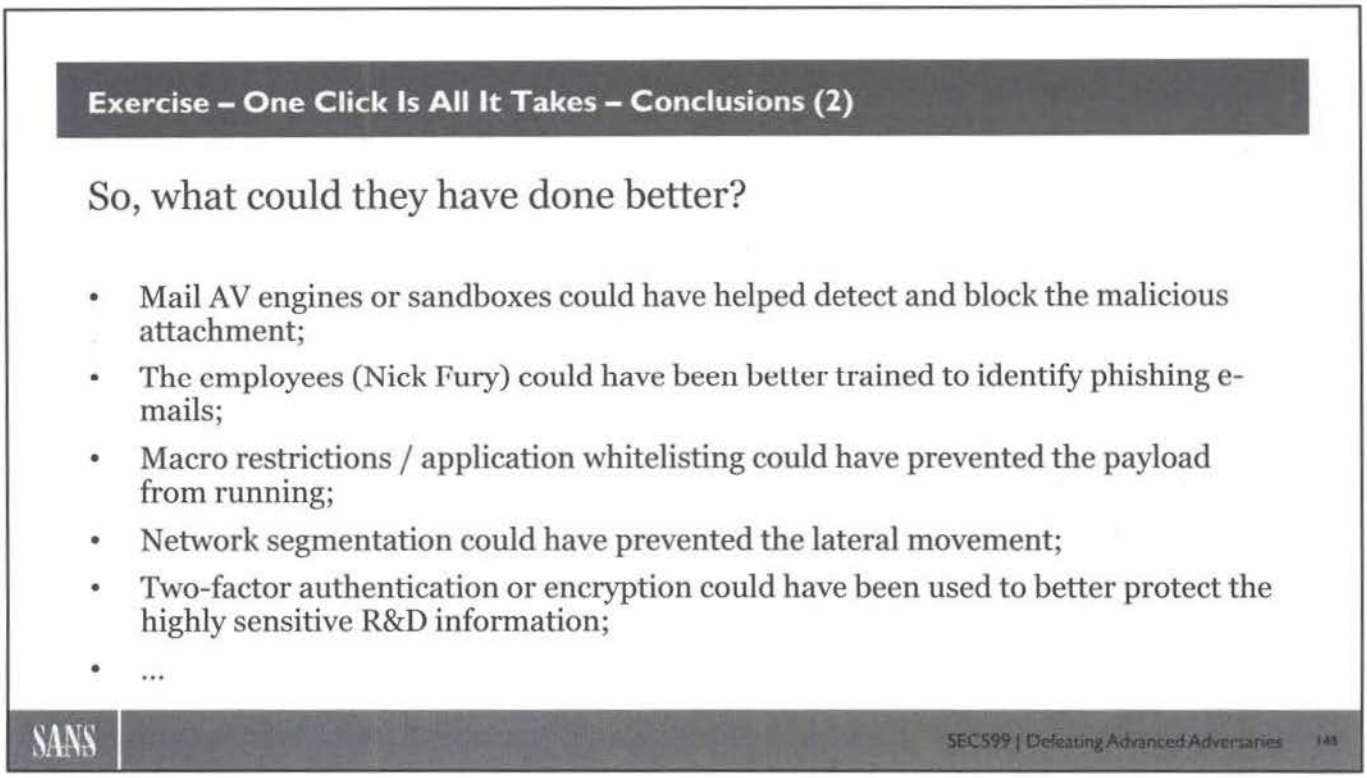
Should you have any questions, please don't hesitate to ask your Instructor or TA for help!



Exercise – One Click Is All It Takes – Conclusion (1)
It's impressive (and somewhat scary) to see how easily the target organization succumbed to our attack, although several traditional security controls were correctly implemented:

- All workstations (including the workstation of Stephanie Jones) is running an up-to-date AV software;
- The employees in the organization have limited access to their systems (no local administrator privileges). Note that we did not have to escalate our privileges at any point during the attack;
- There is outbound network filtering, as workstations and servers are required to pass over the proxy in order to access the Internet;
- There is some basic network segmentation;

We will use this initial lab as a basis to define security controls for the upcoming days.



Exercise – One Click Is All It Takes – Conclusion (2)
So what could they have done better? There's a number of points for improvement that we can highlight here:

- The malicious Word document was not stopped by any mail AV engines or sandboxes;
- The employees (Nick Fury) did not recognize the phishing e-mail that was sent;
- Additional restrictions on macro's or application whitelisting could have prevented the payload from running;
- Network segmentation could have prevented lateral movement from employee workstations to servers storing highly sensitive R&D information;
- The R&D information itself was not properly protected (no strong authentication, encryption…)

For the remainder of this course, we will deep-dive into securty controls that can help us prevent & detect this type of attack!

Course Roadmap

- **Day 1: Knowing the adversary, knowing yourself**
- Day 2: Averting Payload Delivery
- Day 3: Preventing Exploitation
- Day 4: Avoiding Installation, foiling Command & Control & thwarting lateral movement
- Day 5: Exfiltration, Cyber Deception & Incident Response
- Day 6: APT Defender Capstone

SEC599.1

Course Outline & Lab Setup
- Course Overview & Objectives
- Attendee System Setup

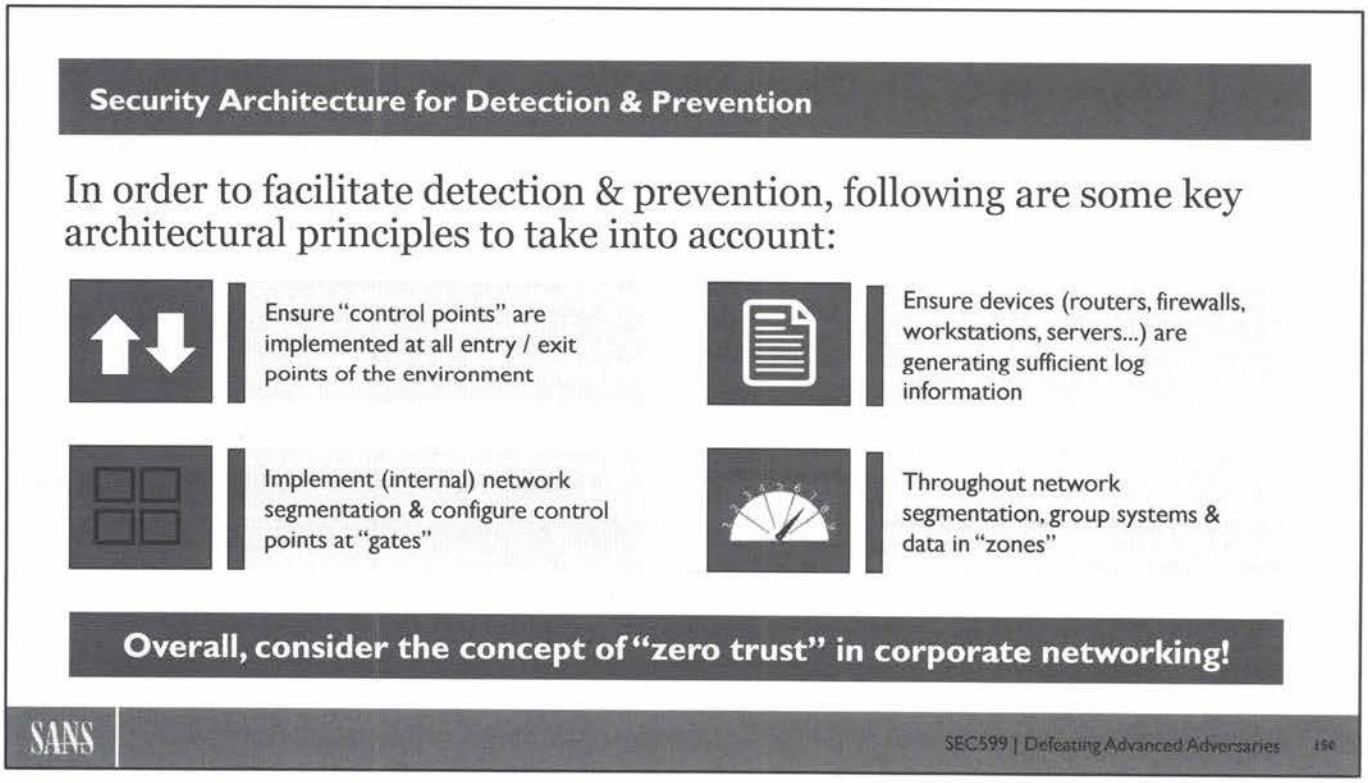
Current Threat / Attack Landscape
- Key Terminology
- What is happening out there?

Introducing the APT Attack Cycle
- Recent Case Studies – In-Depth
- Exercise: Analyzing The Behavior of Famous Malware
- Exercise: One Click Is All It Takes…

A Defensible Architecture & Environment

Preparation - Knowing Yourself
- Understanding Your Own Environment
- Determining What is "Normal"
- Understanding & Limiting Your Internet Footprint
- A Word on Vulnerability Management

This page intentionally left blank.



Security Architecture for Detection & Prevention

Network architecture is only one element of the overall design of your enterprise IT infrastructure. We take network architecture as an example to illustrate the impact it can have on prevention, detection, and logging, but be aware that other elements of your IT infrastructure architecture can have an impact too.

All entry and exit points of your environment should be control points where traffic is monitored. These points are your first line of defense when it comes to keeping adversaries out.

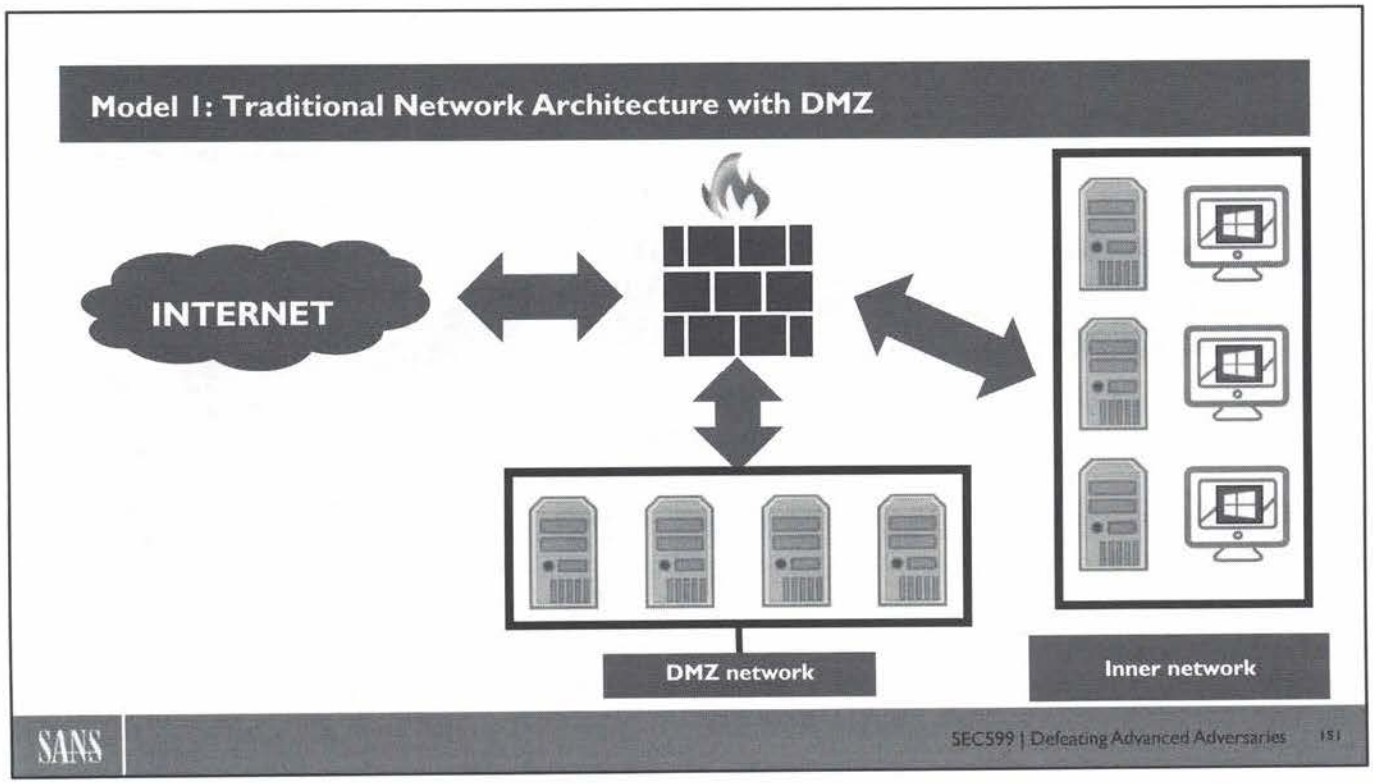
Not a single large enterprise has a flat network architecture, where all systems are connected to each other without any limitations. Complex networks need to be structured in a way to make them efficient, manageable, and secure. There are several network architectures, and most of them include a perimeter segregating the enterprise systems from the systems on the Internet. The "gates" separating these different network segments should be configured as control points similar to your environment's entry and exit points. They can be seen as the entry/exit points for your different network segments, where certain segments might have stricter security requirements than others.

To allow detection and monitoring, all devices located in the network, such as routers, firewalls, servers… should generate log information. Some of these devices will be serving as control points in your network, which makes logging especially important on these devices. As we have seen in the previous slides, having adequate logs available will facilitate investigations in case of a breach.

In addition to network segmentation, systems and data can be grouped into zones as well, for example indicating similar requirements for data confidentiality and integrity. These zones could implement the need for certain communication patterns as well, allowing systems in zoned X to communicate with zone Y, but not with zone Z.

Working under the assumption that breaches will happen, it is possible to design your enterprise systems to facilitate detection and monitoring.

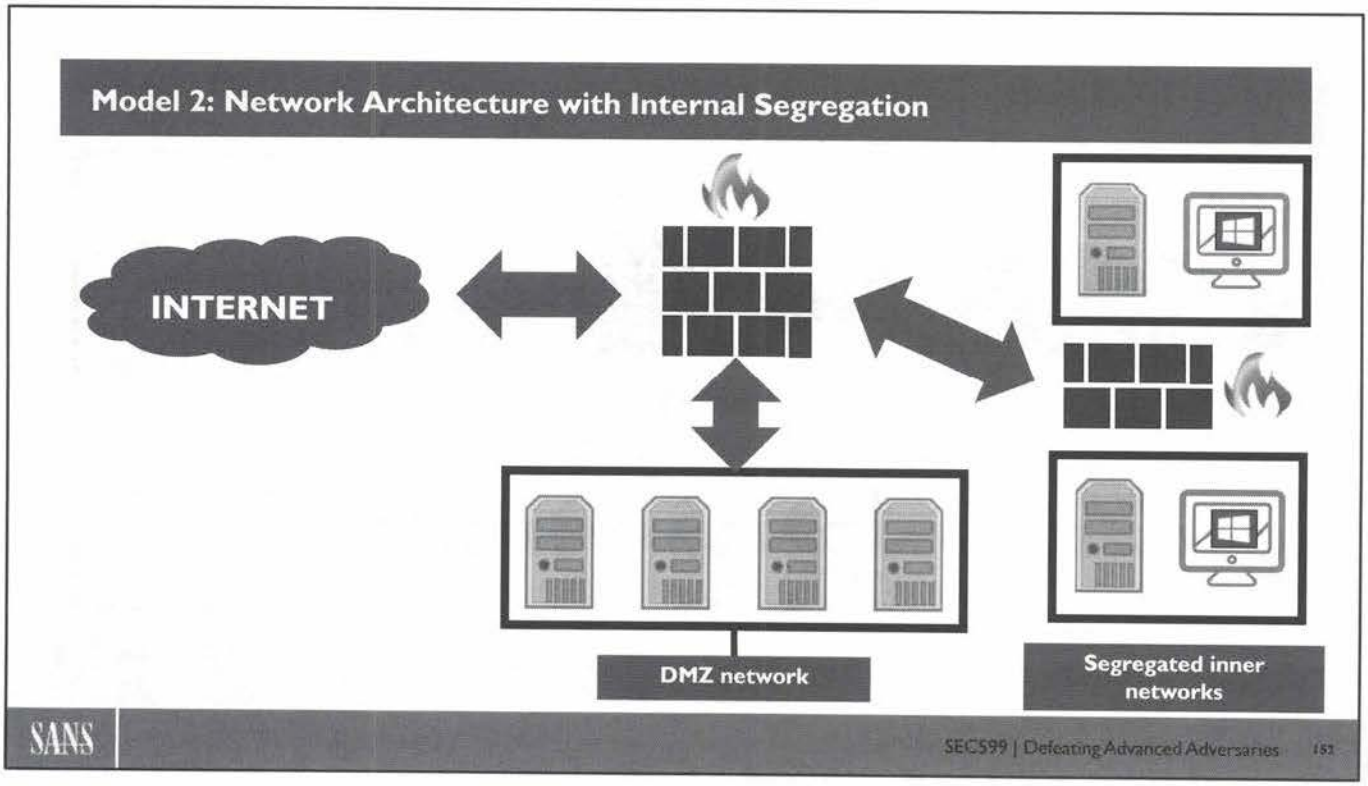
In the next slides, we will illustrate a couple of different network architectures, and the impact they have on prevention, detection, and monitoring.



Model 1: Traditional Network Architecture with DMZ
A popular network design includes a Demilitarized Zone: a DMZ is a network zone where servers that require direct Internet access are placed into. For example, e-mail servers, http proxies… The DMZ zone has access to the Internet (often filtered through a firewall, but this is not illustrated here to keep the diagram simple). The Inner network contains servers and workstations that don't have direct Internet access. They can freely communicate with each other, or with the DMZ servers through firewalls (and possibly other network devices, likes IDS/IPS).

The presence of firewalls between the DMZ zone and Inner zone allows us to perform prevention, detection, and logging. Systems in the Inner network just need to access a couple of TCP ports on the servers in the DMZ, thus TCP connections to other ports can all be dropped by the firewalls between the DMZ and the Inner network. This prevents attacks from the Internet or DMZ that use protocols that require other ports, like SMB. Dropped packets can be logged and the logs monitored, giving us the capability to detect attacks.

In a DMZ architecture, often connections between the DMZ and Inner network have to be initiated from the Inner network.



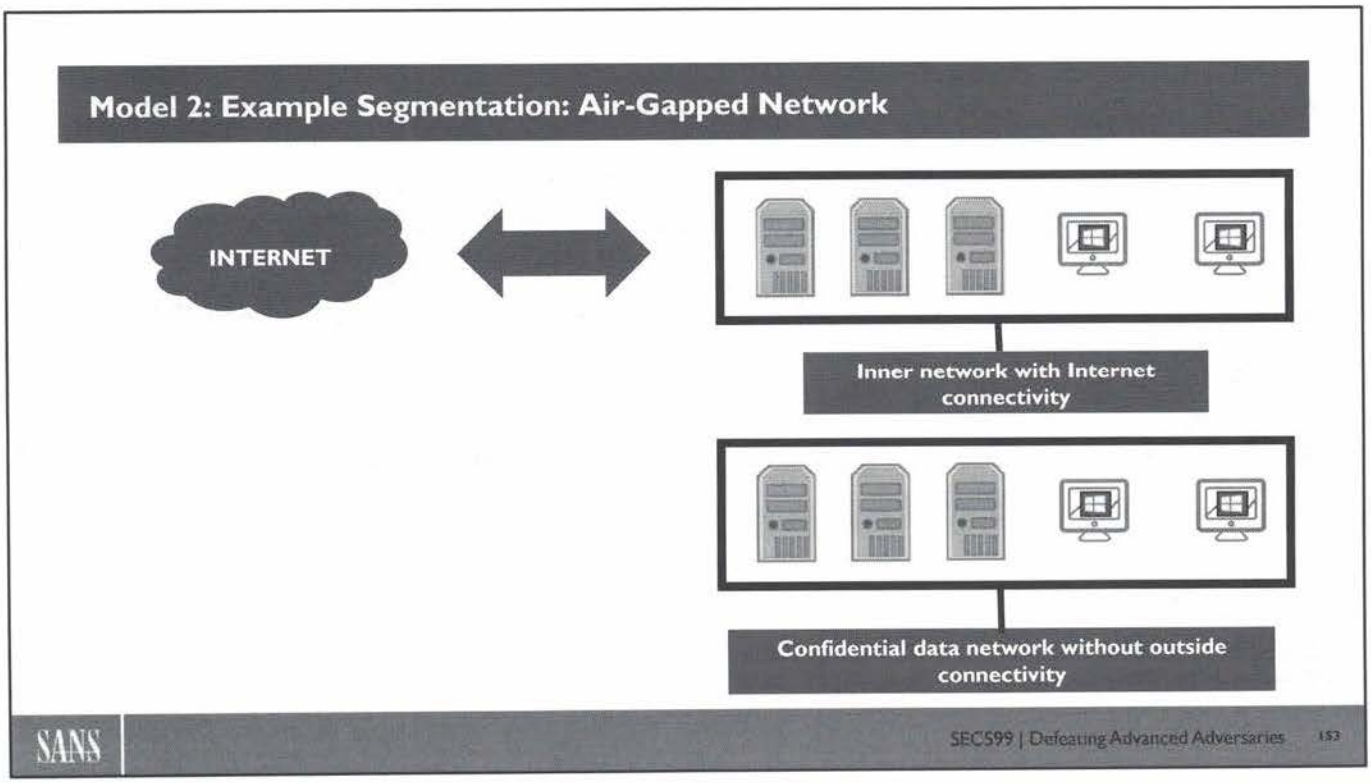
Model 2: Network Architecture with Internal Segregation
Although the DMZ / Inner zone design is popular and relatively easy to implement, it has little to offer when it comes to prevention and detection inside the Inner network.

A solution that is often implemented, is to further divide the Inner network in separate networks and put firewalls in place between the different Inner networks. This network segregation allows putting prevention and detection in place at the firewalls between the Inner networks. Firewalls are not the only network devices that can be put between zones: IDS/IPS can also be put in place.

Inner network segregation can be done according to different criteria, depending on the type of enterprise and the security needs.

Examples of segregation:

- Segregate the networks in production, testing and development networks
- Segregate the type of systems: servers in one zone, workstations in another
- Segregate the type of systems according to business unit: IT, HR, accounting, …
- Segregate geographically
- Segregate the systems according to security level: systems with confidential data, with secret data, with unclassified data, …
- …



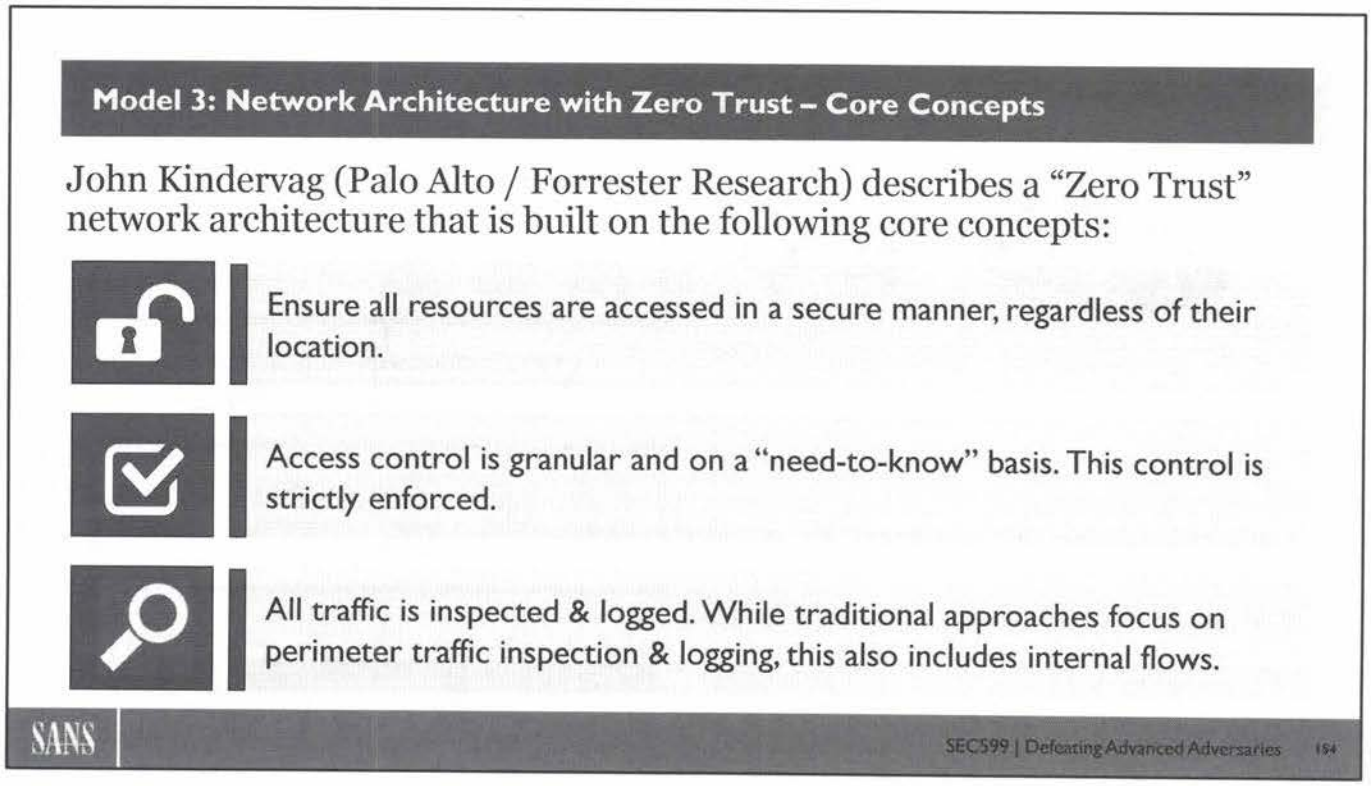
Model 2: Example Segmentation: Air-Gapped Network
An interesting variant of segmented networks is the use of "air-gapped" networks, which are often used to handle classified data.

While all enterprises require Internet access, many enterprises have also highly confidential data that must never leave the enterprise. Some enterprises will create a network zone that has no physical connections to the Internet, or to other zones that have Internet connections. This segregated network zone contains highly confidential data, the systems on that network are the only systems with access to that data, and the data never leaves the confidential data network zone. It is a network architecture that can be found in intelligence agencies and military organizations and is often referred to as an airgapped network.

These networks are not implemented with Software Defined Networks or Virtual LANs because these technologies do not offer physical segregation.

Building a separate, physical network carries a significant cost, hence this is only done for highly confidential data. Organizations that put this network architecture in place also have a strict, formalized data classification scheme. Unless you can clearly identify and manage highly confidential data, it is useless to implement a separate network.

While we focused on networks and their segregation, one must not forget that networking is only one access path to systems. Physical security is as important as network security. Unattended computer systems must be placed in an environment where only authorized and trusted staff has access to the systems.

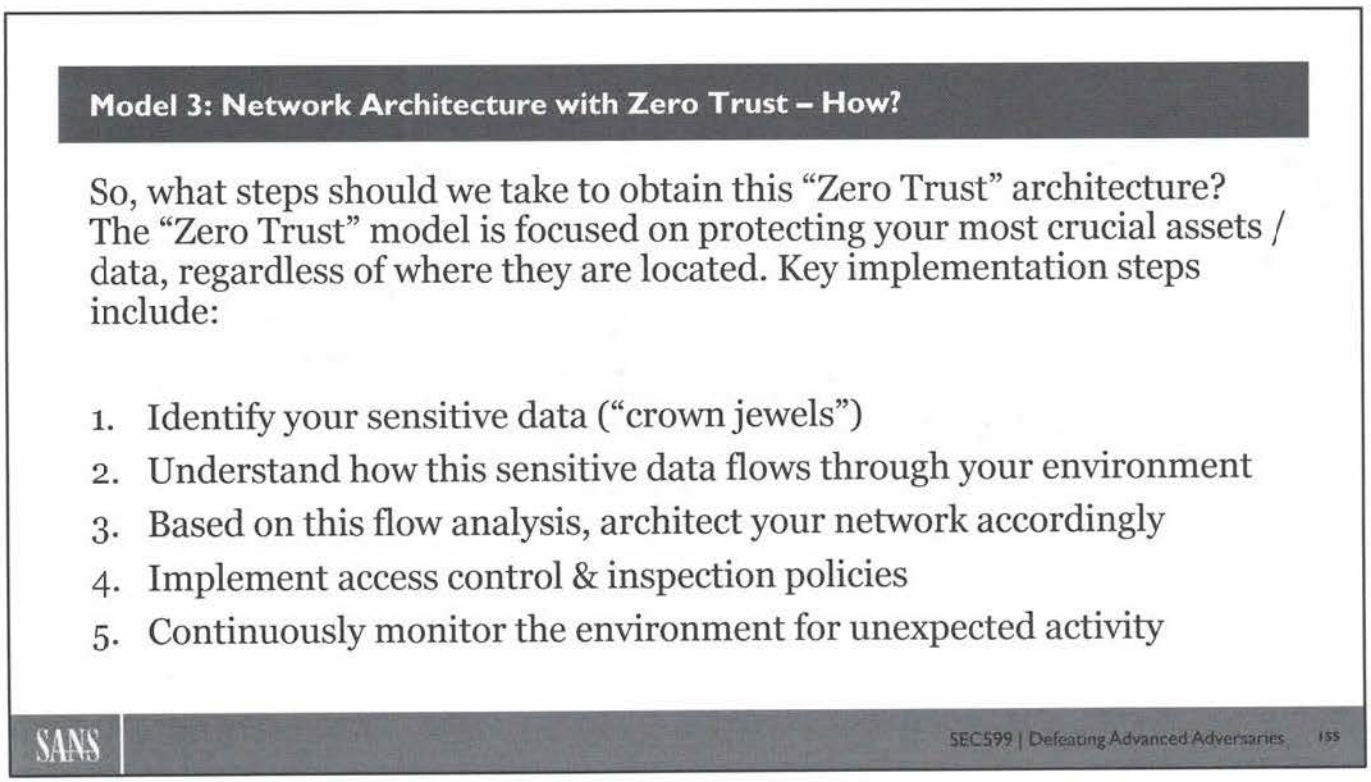


Model 3: Network Architecture with Zero Trust – Core Concepts
In a large, flexible enterprise, putting systems in dedicated zones with specific security levels proves to be a problem. Staff is also working off-premise (for example at home), uses mobile devices, … Mobile devices can be owned by the organization, or privately owned (Bring Your Own Device schemes).

John Kindervag (Palo Alto / Forrester Research) defined the term of a "Zero Trust" network architecture, where trust as a concept in architecture is eliminated. The idea is that this would simplify overall security architecture, as abused trust relationships are often at the basis of security incidents & breaches. As a result of the Zero Trust model, a single compromise will be isolated in a very specific segment of the network. Because of the Zero Trust, it will be hard(er) for an adversary to move laterally in the network and compromise further devices.

The "Zero Trust" model is built on the following core concepts:

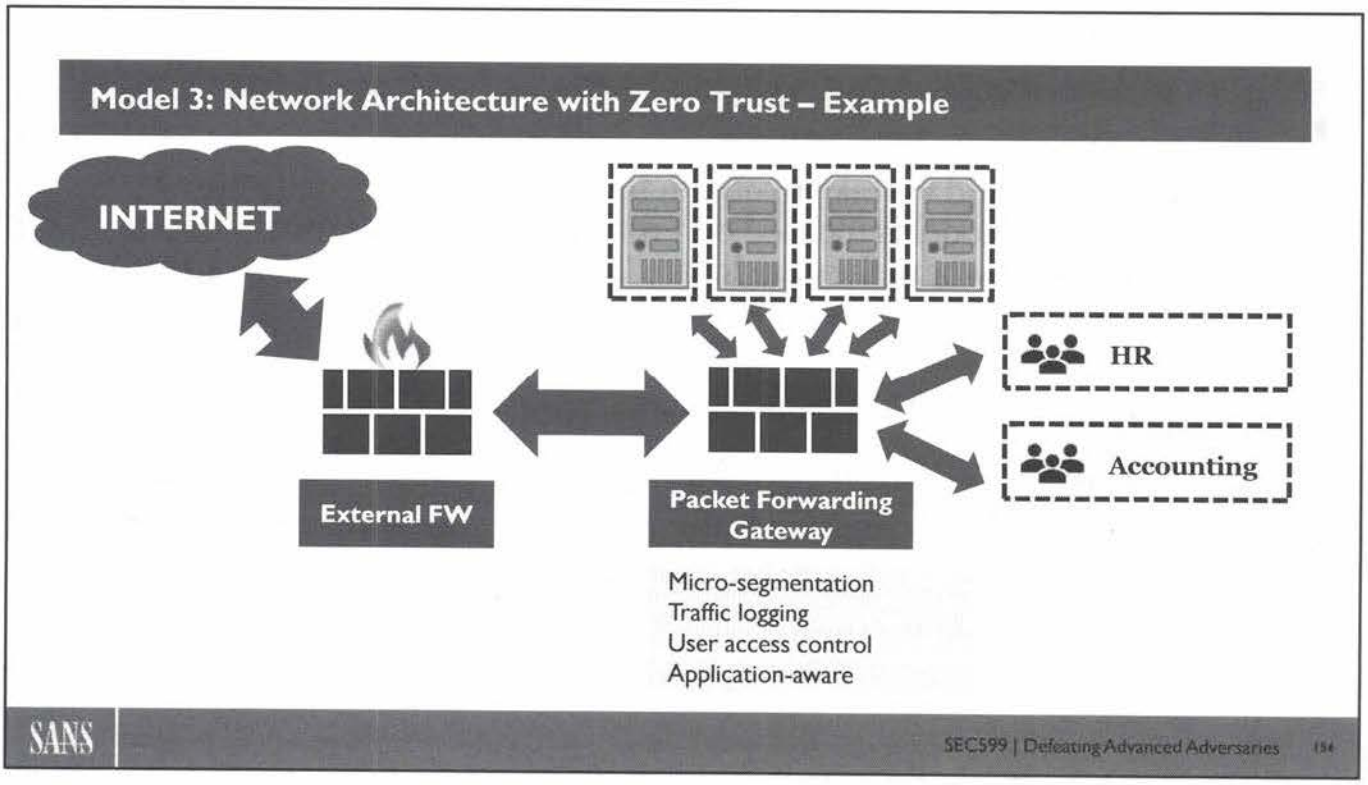
- All resources are accessed in a secure manner, regardless of their location;
- Access control is granular and on a "need-to-know" basis and is strictly enforced;
- All traffic is inspected & logged. While traditional approaches focus on perimeter traffic inspection & logging, this also includes internal flows.



Model 3: Network Architecture with Zero Trust – How?
So what steps should we take to obtain this "Zero Trust" architecture… The "Zero Trust" model is focused on protecting your most crucial assets / data, regardless of where they are located… Key implementation steps include:

- Identify your sensitive data ("crown jewels")
- Understand how this sensitive data flows through your environment
- Based on this flow analysis, architect your network accordingly
- Implement access control & inspection policies
- Continuously monitor the environment for unexpected activity

So, let's have a look at what that could practically look like on the next page…



Model 3: Network Architecture with Zero Trust – Example
In a typical Zero Trust setup, at the heart of the environment could be a so-called "Packet Forwarding Gateway", that includes some of the following functions:

- Micro-segmentation
- Centralized traffic logging & monitoring
- User access control
- Application-level awareness

Thanks to its location at the heart of the environment & its application-level awareness, the "Packet Forwarding Gateway" can make fine-grained security decisions & be a central point for logging & monitoring. This is something that wasn't possible before, as network appliances lacked the computing power to perform such fine-grained decision-making / analysis. Instead, firewalls were used that would make "simpler" security decisions based lower layers of the OSI stack using rules, zones & network segments.

An interesting implementation of this "Zero Trust" principle is Google's BeyondCorp, which they now also offer as a cloud-based service. More information can be found here: https://cloud.google.com/beyondcorp/. BeyondCorp's principles are very much in line with the core concepts in Zero Trust:

High-level Components of BeyondCorp
Single sign-on, access proxy, access control engine, user inventory, device inventory, security policy, trust repository

BeyondCorp Principles
Connecting from a particular network must not determine which services you can access.
Access to services is granted based on what we know about you and your device.
All access to services must be authenticated, authorized and encrypted.

A Defensible Architecture – Additional Resources

Some additional resources that can prove to be useful for security architectures include:

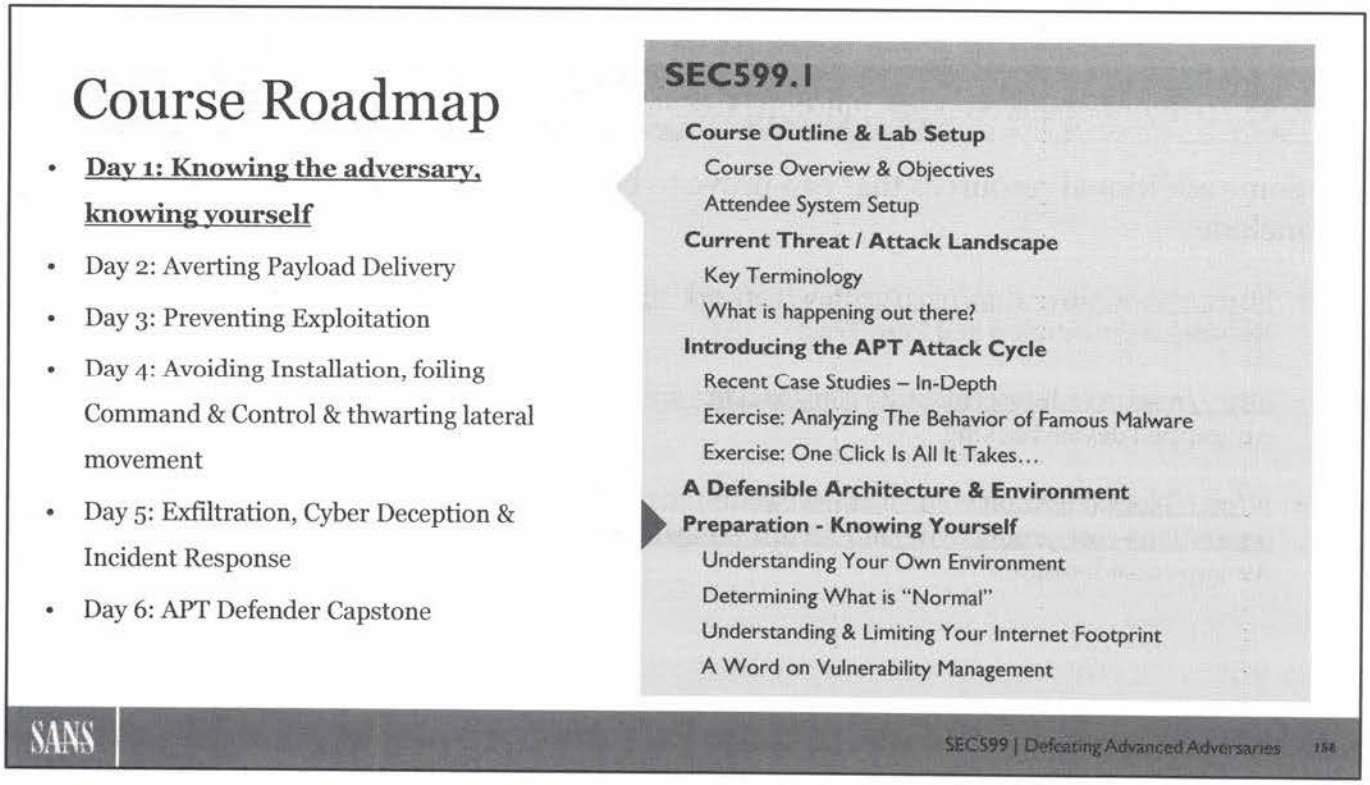
- https://www.plixer.com/blog/netflow/network-segmentation-zero-trust/
  Network segmentation and Zero Trust
- http://resources.infosecinstitute.com/hacking-air-gapped-networks/#gref
  Air-gapped device hacking
- https://blogs.microsoft.com/microsoftsecure/2017/05/01/mind-the-air-gap-network-separations-cost-productivity-and-security-drawbacks/
  Air gap considerations

A Defensible Architecture – Additional Resources
Some additional resources that can prove to be useful for security architectures include:

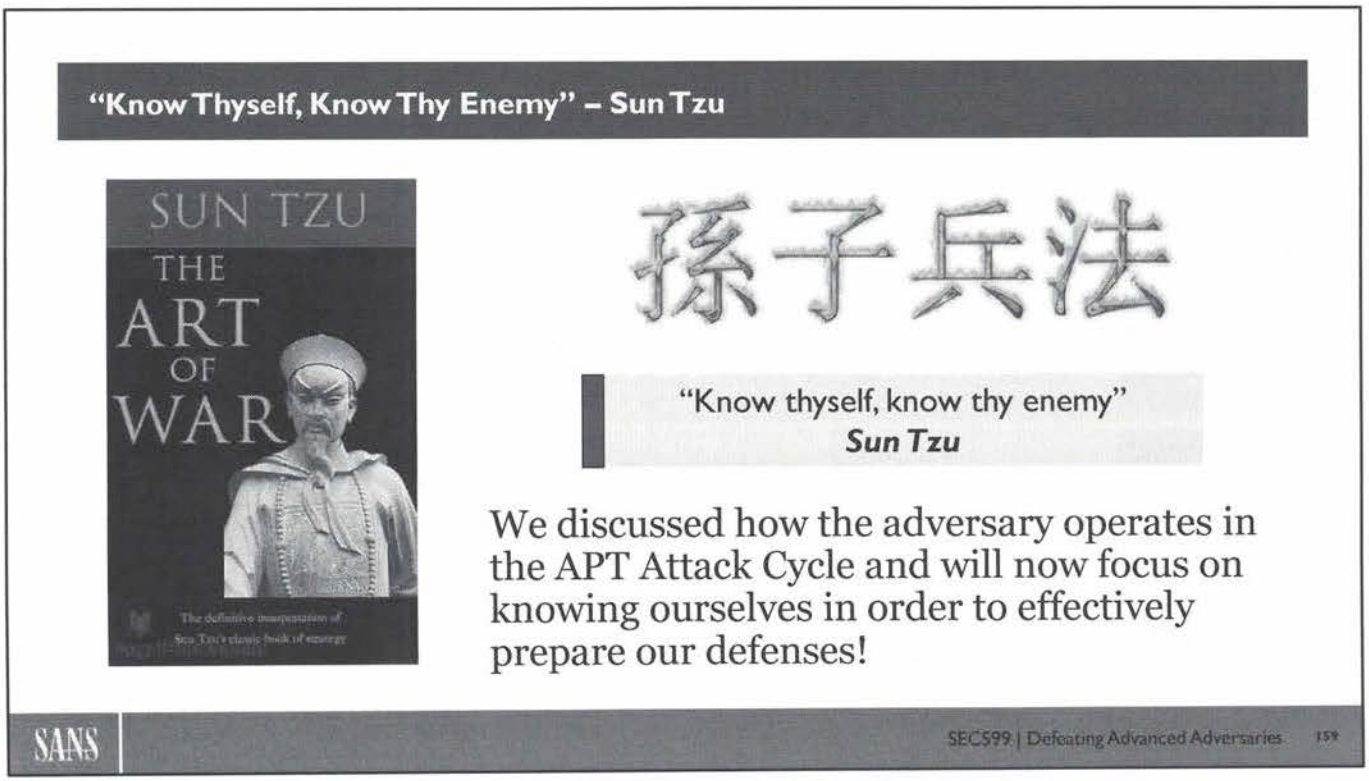
https://www.plixer.com/blog/netflow/network-segmentation-zero-trust/
Network segmentation and Zero Trust

http://resources.infosecinstitute.com/hacking-air-gapped-networks/#gref
Air-gapped device hacking

https://blogs.microsoft.com/microsoftsecure/2017/05/01/mind-the-air-gap-network-separations-cost-productivity-and-security-drawbacks/
Air gap considerations



This page intentionally left blank.

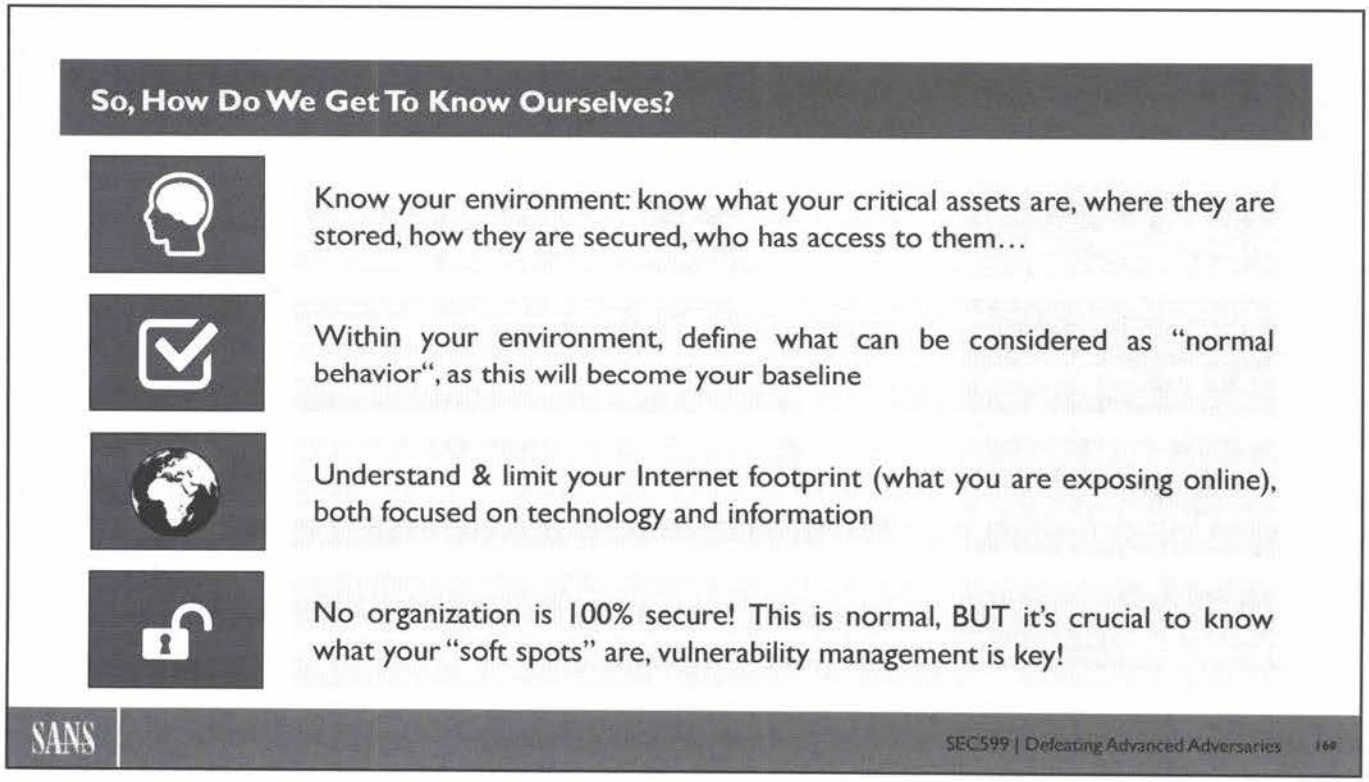


"Know Thyself, Know Thy Enemy" – Sun Tzu
As an IT security professional working for your company or organization, you have assets to protect. The adversary covets (some of) your assets.

To protect these assets, it is important that you know what your assets are. As it is not economically viable to give all assets the same level of protection, you will have to make a selection. Select important assets (crown jewels) to protect first. Crown jewels are important to the company, and coveted by your adversaries. Know your enemy, and why the enemy is interested in your assets.

Know which skills you and your team have. Know which skills are important to protect your assets. Making a matrix of your skillset and required skillset enables you to identify skills you might lack. A lack of skills can be resolved through various means: training, recruiting, outsourcing… Estimate the number of resources you need to protect your assets: FTEs, software, hardware…

To know more about your enemies, stay up-to-date with the current attacks in your industry. Read articles, go to conferences, speak with your industry peers.

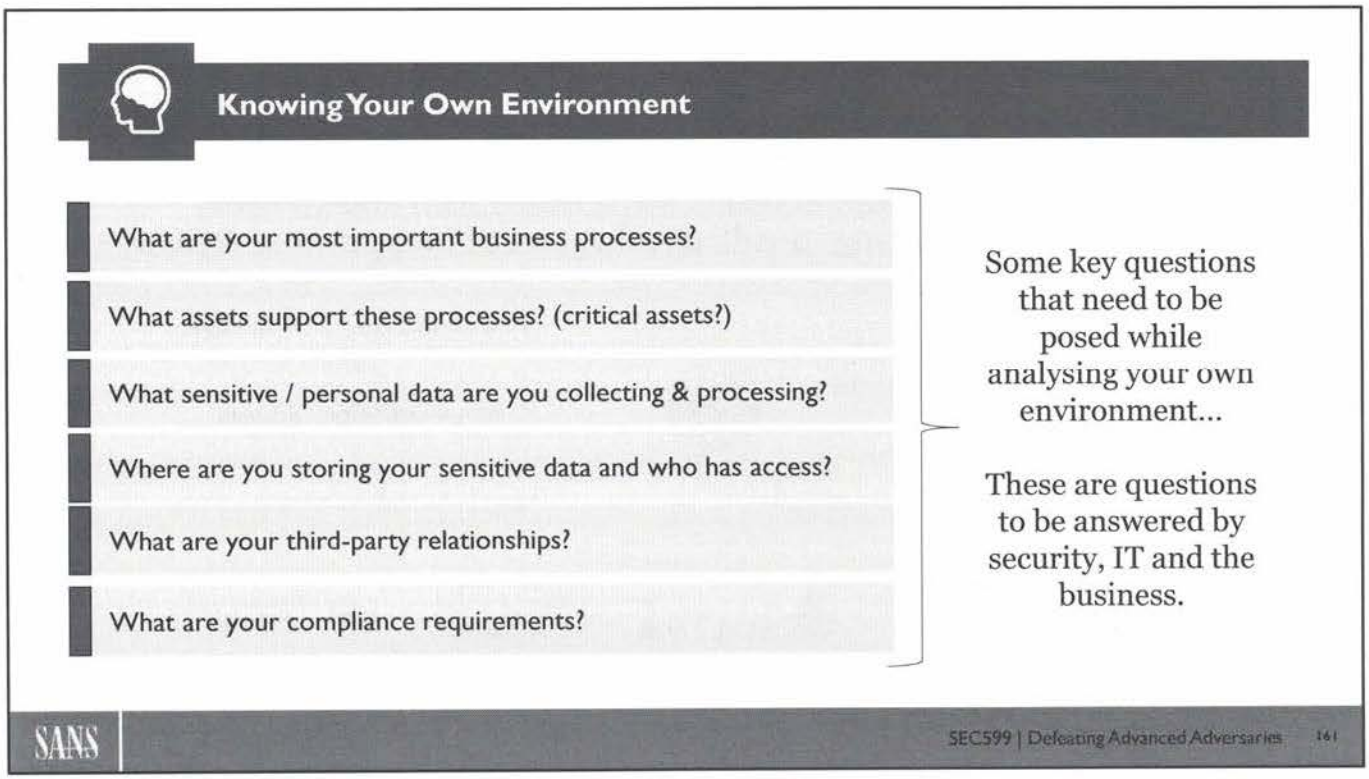


So, How Do We Get to Know Ourselves?
An incredibly big part of defending against advanced adversaries is knowing your own environment. This is something often overlooked or underestimated by organizations. Your environment includes IT people, hardware, software, networks, … your company hires, rents or owns. It includes third-party services you use. Especially with the rise of cloud technology, it becomes more common to use hardware and services your company does not own. To protect this environment, you need to understand it. What are the organization's business objectives and how does IT enable that?

Furthermore, it's important to know what is considered to be "normal" for your organization. Although certain "generic" baselines exist, every organization is different. What is normal for your organization could be highly suspicious to another organization. It is thus important to spend some time "getting to know" your environment.

Every organization has an Internet footprint (what you are exposing online). It's important to know what your Internet footprint is. What are you exposing online? What kind of systems are directly connected to the Internet? What are the different departments of the firm publishing online?

Finally, we have to understand and accept that no organization is 100% secure. This is normal, as it's all about risk management. What organizations do however have to understand is what their vulnerabilities are and what the soft spots are. This information will be crucial when adversaries are probing your systems (or have already compromised parts of your environment).

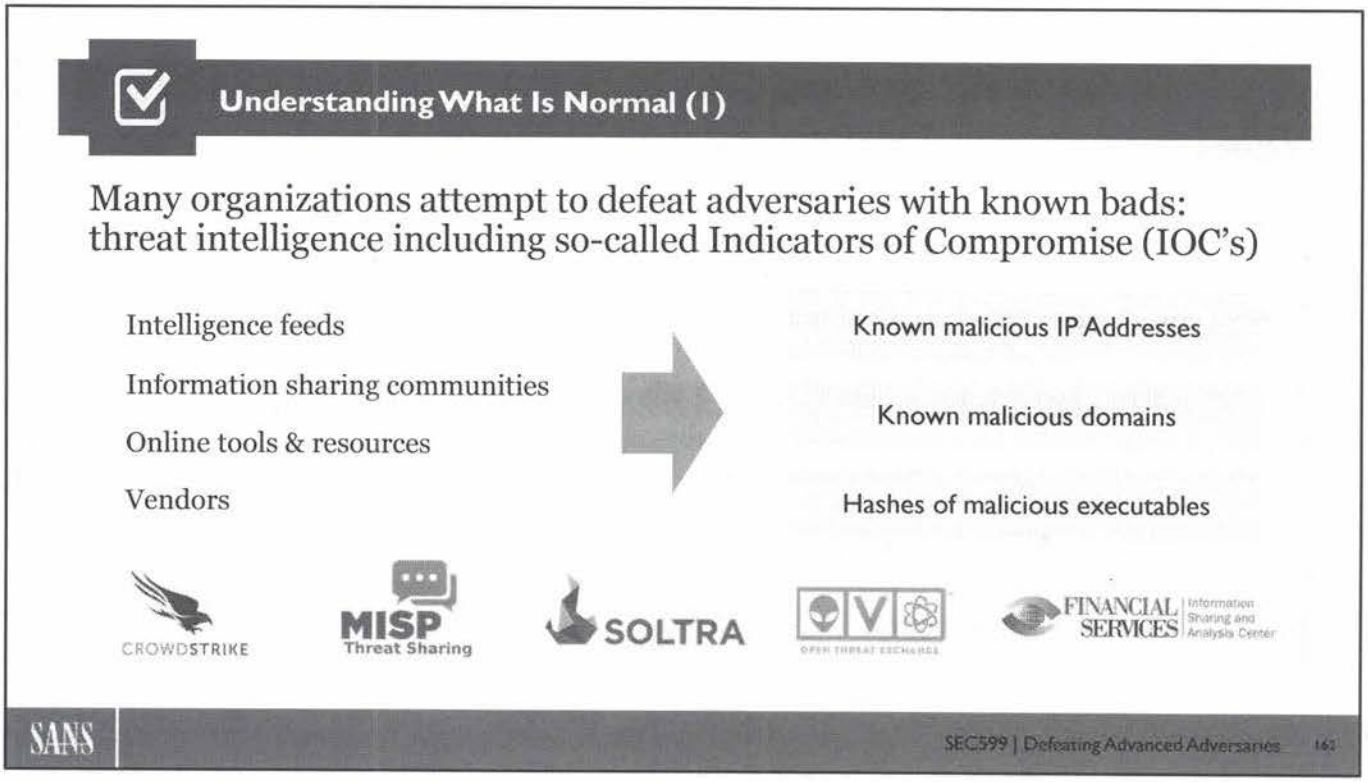


Knowing Your Own Environment

The slide above lists some interesting questions when we want to get to know our own environment. The list should not be considered exhaustive, but it's a good start to get the discussion going:

- What are your most important business processes? This is what it's all about… What does your organization do? Are you an eCommerce organization? Do you perform Research & Development? This is not a question for your IT department, this is a question for the business!
- What assets support these processes? The assets that support your core processes will be your critical assets. They are considered to be vital for the organization to reach its objectives.
- What sensitive data or personal data are you collecting & processing? Even if some information is not considered critical to the organization's objectives, it could, however, have a serious impact. Imagine that you are accidentally storing personal information about your customers, without actual business need. Should this information be stolen, you'll face serious reputation damage and possibly legal ramifications or regulatory issues.
- In this time and age, where is this data stored? To what extent are you already leveraging cloud-based services?
- Who controls access to your critical assets and data?
- What are compliance requirements / regulations you need to adhere to? This will be different for different industries.

Although these questions may appear to be straightforward, the responses can be rather complex! Responses to these questions should not only be drafted by security personnel, it should be discussed amongst IT, security and business personnel.



Understanding What Is Normal (1)

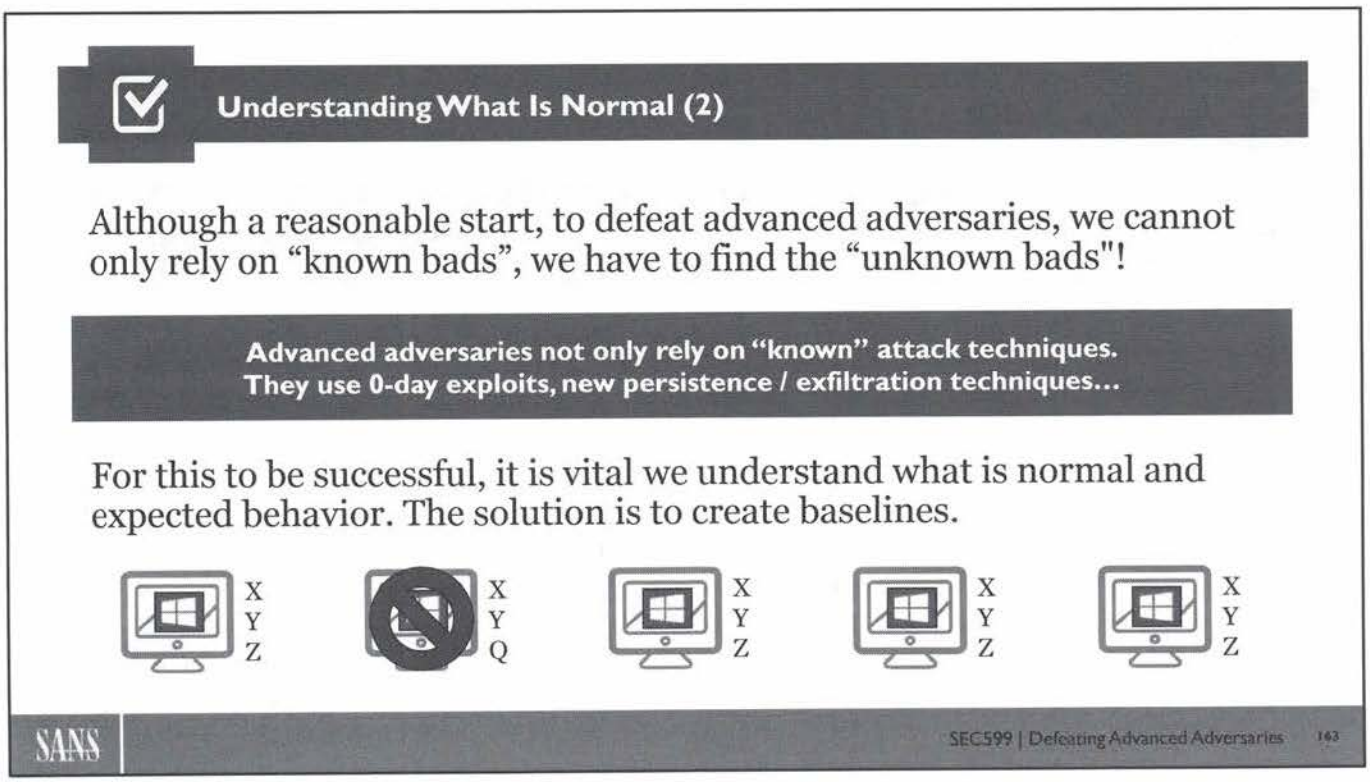
Historically, many organizations have attempted to defeat adversaries with known bads. For example, they have used "threat intelligence" including so-called Indicators of Compromise (IoC's). Sources of this type of information include:

- Intelligence feeds (open-source or commercial);
- Information sharing communities;
- Online tools & resources;
- Vendors.

Typical information that is obtained through these channels could be the following:

- Known malicious IP addresses;
- Known malicious domains;
- Hashes of malicious executables;
- …

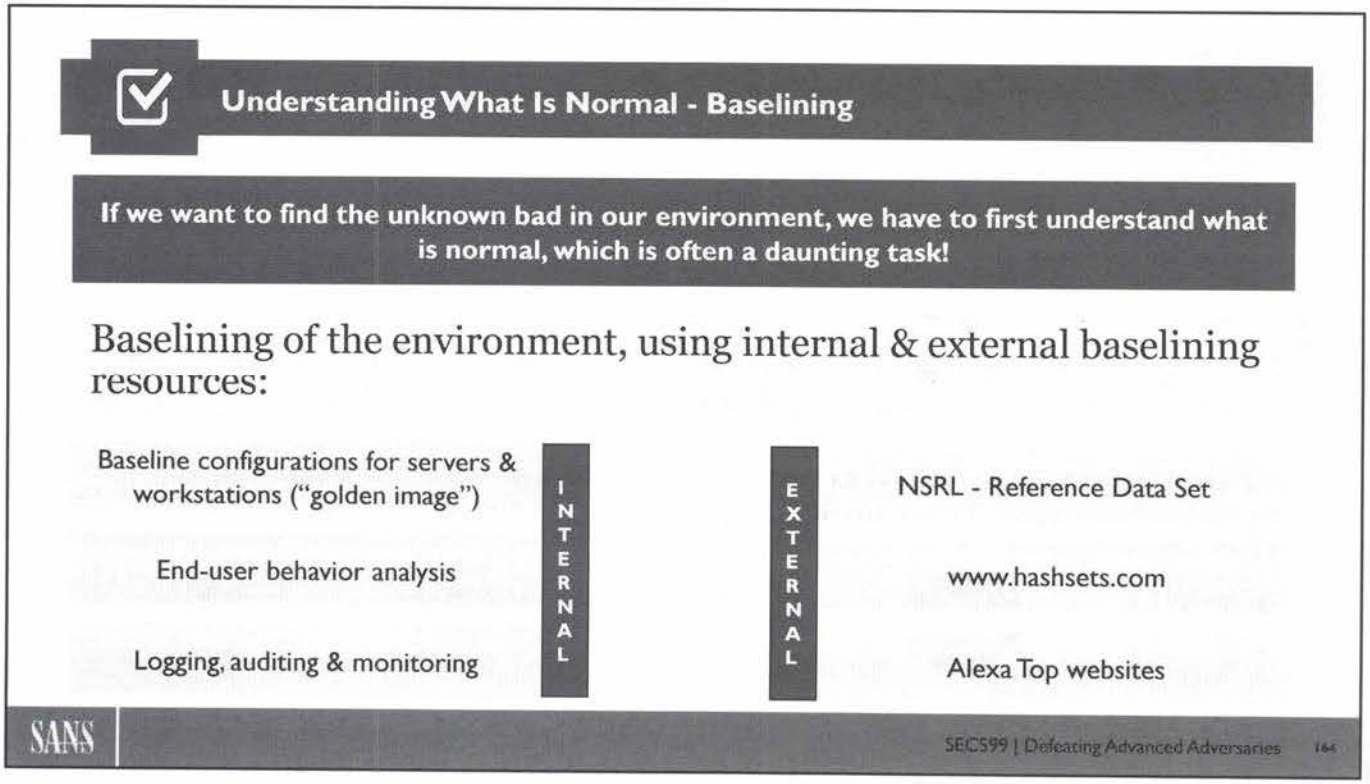
Although this is a reasonable start, it's important to note that it is not enough to defeat advanced adversaries.



Understanding What Is Normal (2)

To defeat advanced adversaries, it's not enough to rely on "known bads". We have to be able to detect the "unknown bads" as well. Advanced adversaries do not only rely on known attack techniques, they are constantly looking for new ways to compromise networks and systems. They want to stay under the radar and avoid detection, that's why they use 0-day exploits and new persistence or exfiltration techniques.

If we want to be able to successfully detect unknown bad, we should know what the known good is, so we need to know what kind of configuration, status, or behavior is expected and normal. The solution is to create baselines containing a situation under normal operations.



Understanding What Is Normal – Baselining

A baseline is a set of data that tells you what is normal. There are many ways to produce or obtain baselines. For the purposes of our course, we will make a distinction between internal and external baselines:

Internal baselines

Internal baselines are generated based upon information that is collected in your own organization. Some examples include

- On the host-level, you could have a baseline configuration for servers & workstations, where all executables (with their hash) are registered;
- On the host-level, you could use Windows event logs to monitor end-user behavior;
- On the network-level, you could run a full packet capture to baseline what type of network activity is normal;

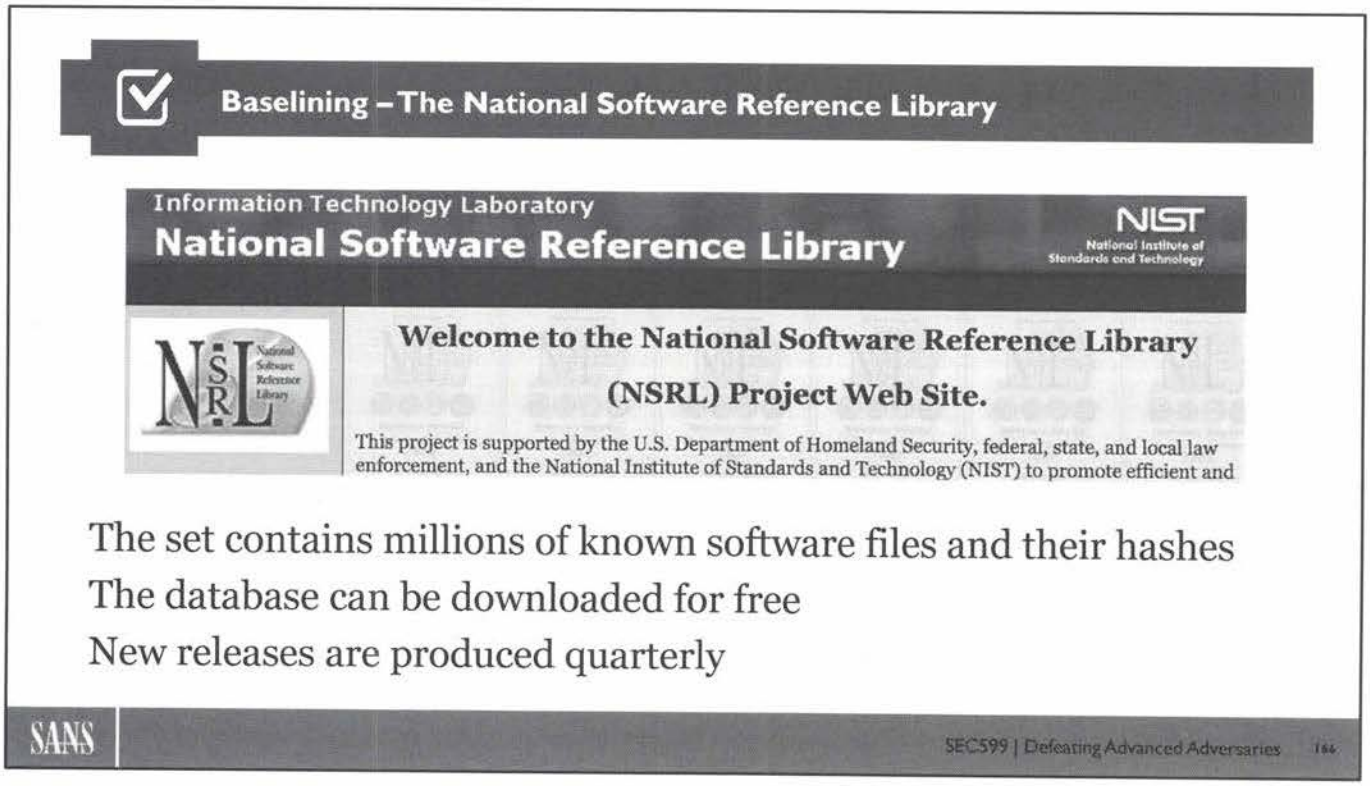
The creation of internal baselines takes time and is not a one-time effort: the organization continuously evolves and so will the baseline!

There are applications and devices that automate this. They observe the behavior of a running Windows machine, or for example, the network traffic going through your router, and in a first phase make a baseline. After the baselining period, where they consider everything that happened during that period as normal, those applications and devices will start monitoring behavior and report deviations from the baseline. For example, there are anti-virus applications that use behavior (with baselines) instead of signatures to detect malware.

Machine learning is new technology that can be classified under artificial intelligence. Machine learning algorithms can be fed data of normal and abnormal behavior, and will then be able to classify new behavior accordingly. Broadly speaking, there are 2 major learning techniques: supervised and unsupervised. In supervised mode, humans will supervise the learning process and correct classifications. In unsupervised learning, there is no human correction. Supervised machine learning tends to generate less false positives and negatives than unsupervised learning.

External baselines

In order to "hit the ground running" and start baselining quickly, you can rely on some external data sources such as the NSRL Reference Data Set (see next slide for details), www.hashsets.com and the Alexa Top lists for most commonly used websites.



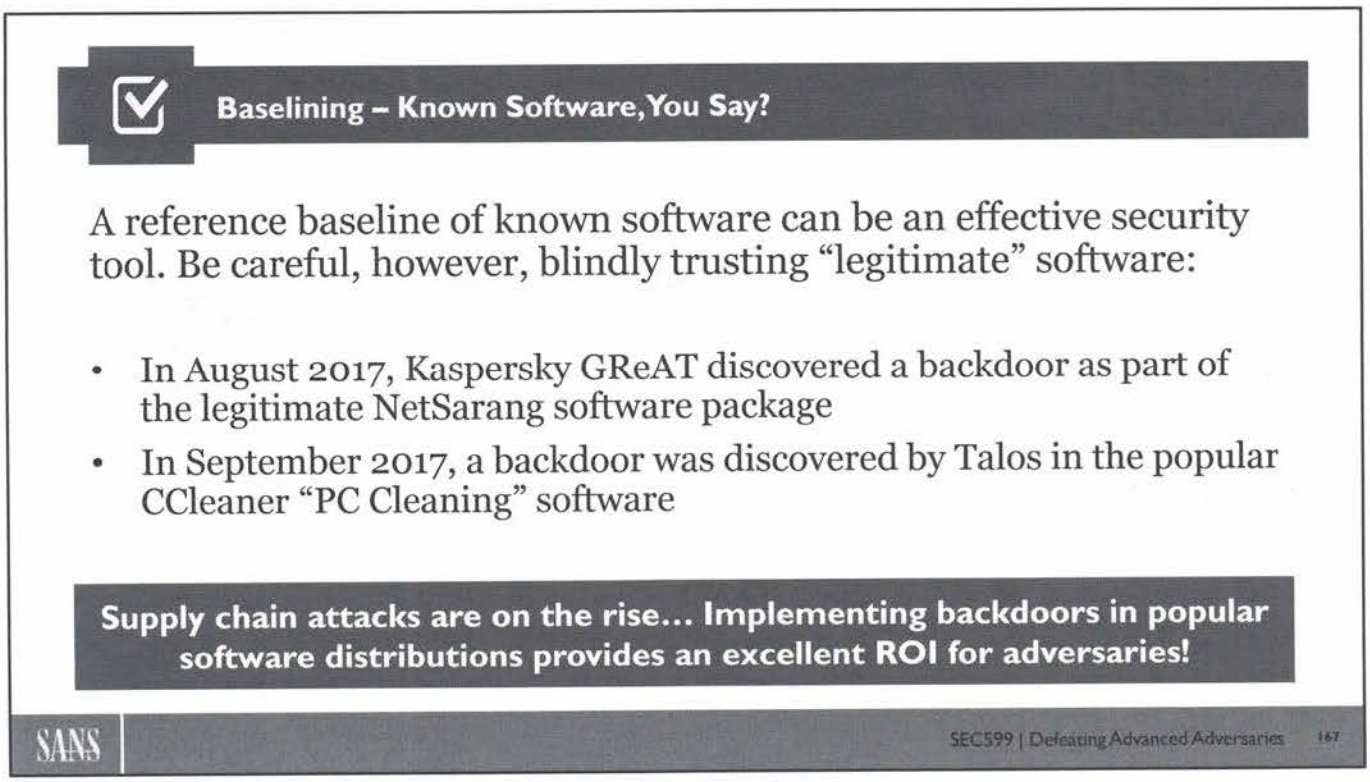
Baselining – The National Software Reference Library

The National Software Reference Library (NSRL) is designed to collect software from various sources and incorporate file profiles computed from this software into a Reference Data Set (RDS) of information. The RDS can be used by law enforcement, government, and industry organizations to review files on a computer by matching file profiles in the RDS. This will help alleviate much of the effort involved in determining which files are important as evidence on computers or file systems that have been seized as part of criminal investigations. The RDS can be obtained from NSRL's website: https://www.nsrl.nist.gov/

It is a free resource: the set can be downloaded for free, without registration. There are different sets available for download. The complete set will include multiple entries for many executables. As the set not only includes the file name and different cryptographic hashes of the file but also other metadata like the classification and the source, multiple entries are necessary. As many executables are present in different applications. Take for example Windows' system DLLs. The same DLL can be found on different SKUs of the same version of Windows.

To reduce the size of a set, and speed up correlation, there is also a set where each executable is only listed once. This set is good to make the distinction between normal and abnormal but is less useful to retrieve the source of an executable. The NSRL produces a new set each quarter.

There are commercial and free tools, mostly forensic tools, that support the RDS.



Baselining – Known Software, You Say?

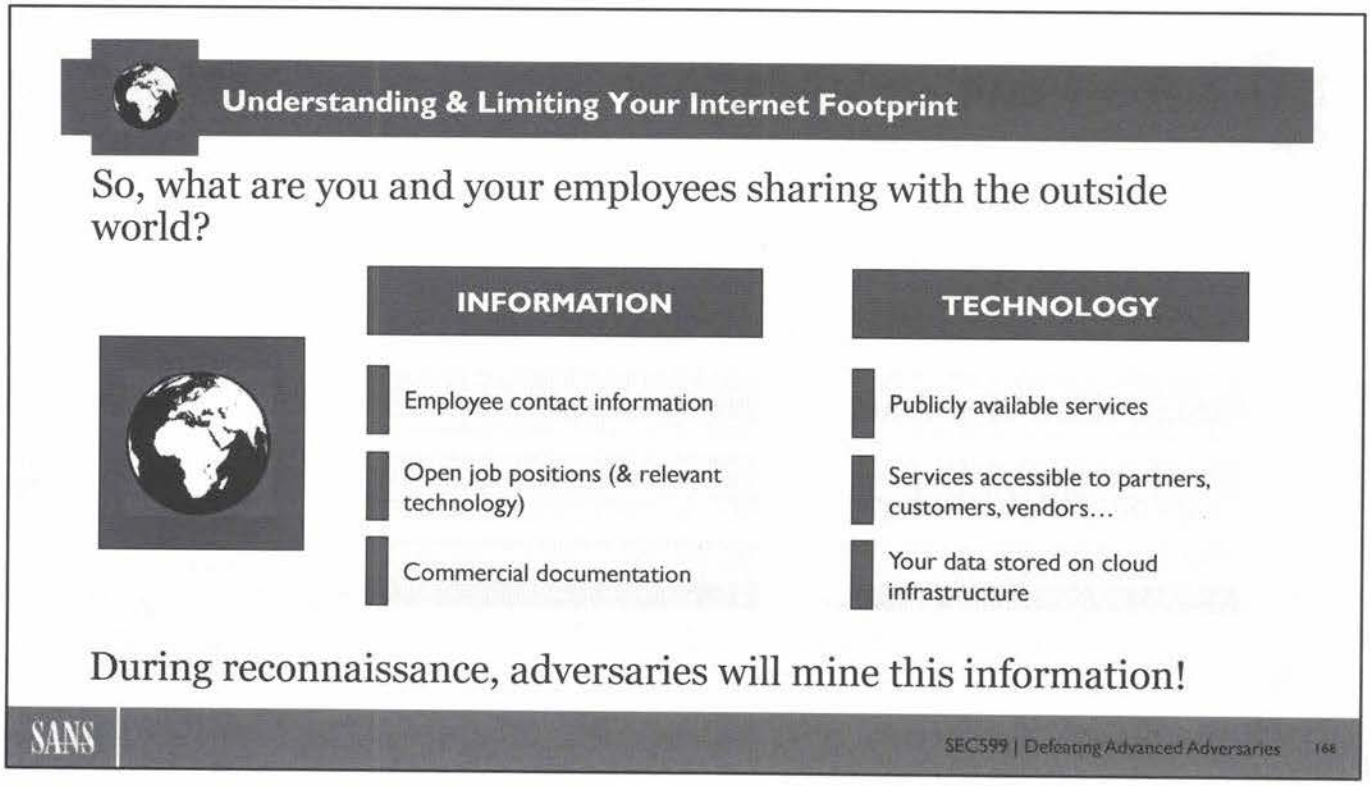
While we introduce baselining of "known trusted software" as a good idea, this also comes with inherent risks. Over the last few months & years, we've been seeing more and more "Supply Chain Attacks", where adversaries attack an organization in the supply chain of their actual targets. The idea is that by first compromising an organization that is already a "partner" of your actual target, you are no longer attacking the target from a purely external perspective: As an example, the supplier the adversary compromised might have network connectivity towards trusted segments of the target network.

Software companies are a prime target for these types of attacks: As an adversary, a highly effective attack technique is to implement a backdoor in an official "software package" released by a trusted vendor. As part of the normal software update process, your backdoor will subsequently be installed on target machines. There is no need to perform any noisy attacks against your actual target to have your backdoor installed: it just enters through the front door ☺

In 2017, we saw two rather large examples of such attacks:

- In August 2017, Kaspersky GReAT (Global Research & Analysis Team) discovered a backdoor ("ShadowPad") as part of the legitimate NetSarang Xshell software package. Founded in 1997, NetSarang Computer, Inc. develops, markets and supports secure connectivity solutions and specializes in the development of server management tools for large corporate networks. As you can imagine, this makes them a prime target for adversaries looking to infiltrate as many corporate organizations as possible!
- In September 2017, a backdoor was discovered by Talos in the popular CCleaner "PC Cleaning" software. Upon additional analysis, it appeared that amongst others, Sony, Samsung, Intel, Microsoft & Google were targets of this campaign.

For this type of attack, the most interesting targets are obviously "small" software development companies that are creating highly popular software with a large user base.

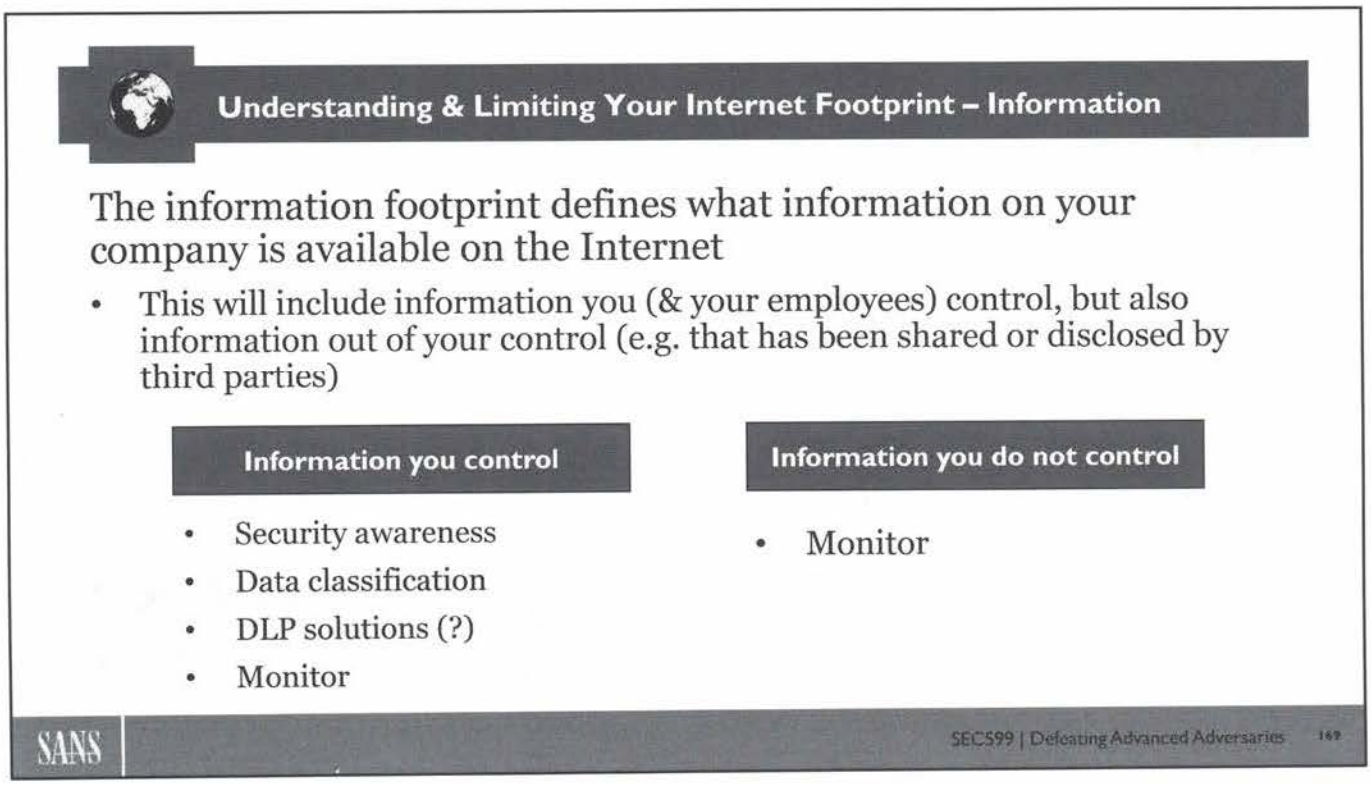


Understanding & Limiting Your Internet Footprint
An "Internet footprint" can have more than one interpretation.

All the information available on the Internet about your company or organization defines the "information" footprint. It should be obvious that more information available means a bigger footprint. Information about your company or organization, available on the Internet, is not always under your control. You have your company's resources sharing information (like your websites), or even exposing information that you would rather not. And you have third parties sharing information. For example, social media, where employees, clients, providers… share information.

All your resources facing the Internet define the "technical" Internet footprint of your company or organization. These resources are web servers serving your websites, file servers, VPN concentrators, email servers, DNS servers… An individual service can have a small or a large footprint, depending on the type and amount of resources it makes available on the Internet. For example, the number of ports, the number of web pages, the number of files…

Your Internet footprint is discussed in this course because it defines your exposure. Simply put, your footprint is a measure of your exposure. Another term you might be more familiar with is the "attack surface". The bigger your footprint is, the bigger your exposure and the bigger your attack surface.



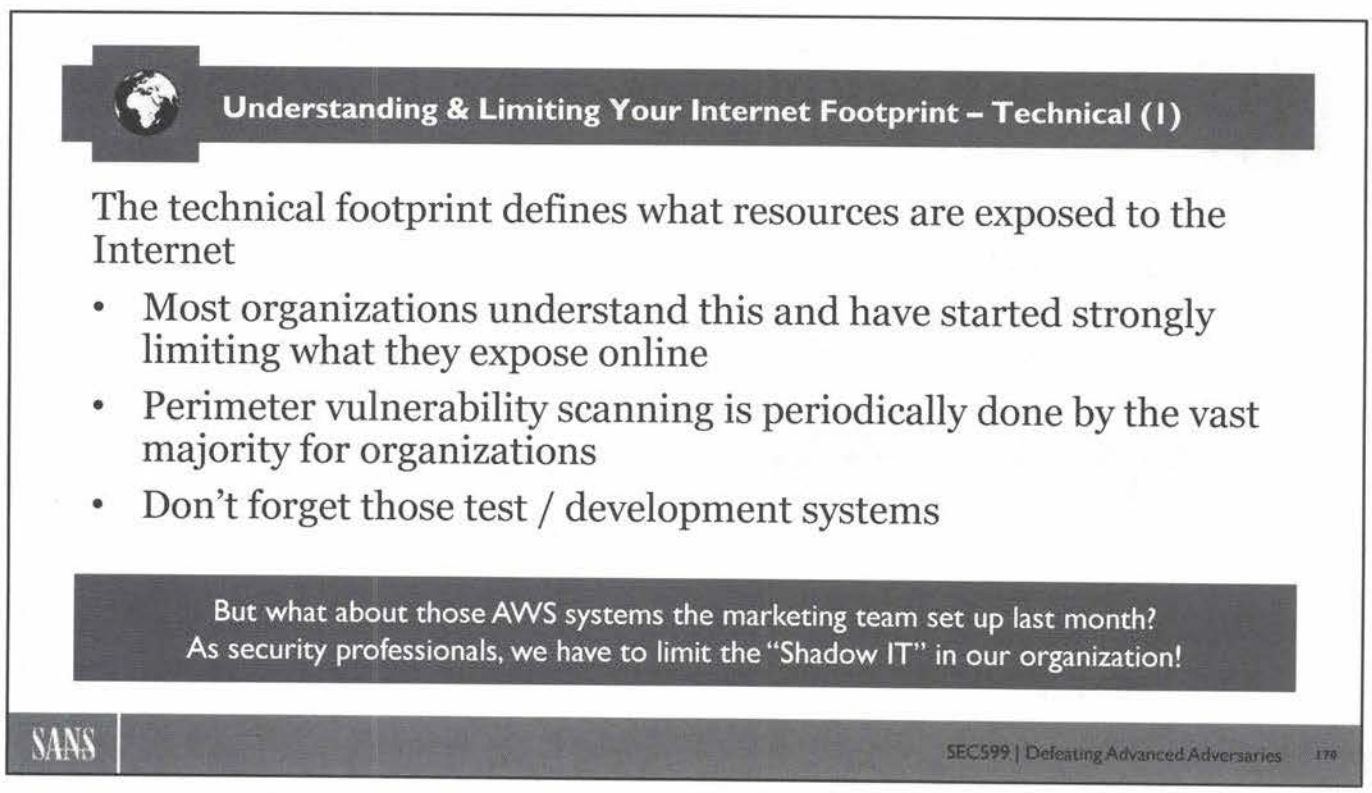
Understanding & Limiting Your Internet Footprint – Information
The information footprint defines what information on your company is available on the Internet. To do business, your company has to share information. Most of that information is publicly available on the Internet, but not everything!

For our purposes, we can distinguish two main types of online information: information you (& your employees) control, but also information out of your control (e.g. that has been shared or disclosed by third parties).

For information you control, you can focus on a number of solutions, including:

- Increasing security awareness of your staff: They should be made aware and trained about information classification guidelines and publishing policies. Knowledge about scammers should be shared with your staff.
- Data classification: It's difficult for staff to understand what they can share if they don't know how data is classified. Ensure all data in the organization is classified and clear rules exist on what these classifications mean. Top Secret information is most likely not intended to be shared on social media.
- You could consider implementing DLP solutions, that attempt to stop classified information from leaving the Internet;
- Monitor the Internet to see what type of information is available on your corporate web page, social media accounts, partner websites …

For information you do not control, there's not much we can do except for monitoring the Internet and responding to information that is exposed.



Understanding & Limiting Your Internet Footprint – Technical (1)

The technical footprint of your company on the Internet encompasses all devices connected to the Internet. Servers, routers, network devices… all connected to the Internet to make your company reachable to customers and potential customers over the Internet, are also reachable to adversaries. An open network port on your server is not in itself a risk, but the server application that opened the port to listen to incoming connections can be a problem. adversaries will connect to the server application via that open port, and interact with it to look for vulnerabilities. Vulnerabilities are bugs in the application, or misconfigurations, that allow adversaries to take over control over the application, for example with remote code execution.

An inventory of all Internet-facing resources will help you to understand your company's technical footprint. Your company probably already has a list of all its assets used in production, and how they are connected to the Internet. But this list must also include the services running on those assets: server applications, open ports, protocols supported… Test, development and staging servers should also be included in this list. It won't be the first time nor the last time that a company has its IT infrastructure compromised because of an Internet-facing development server that was "forgotten" and not protected like production servers.

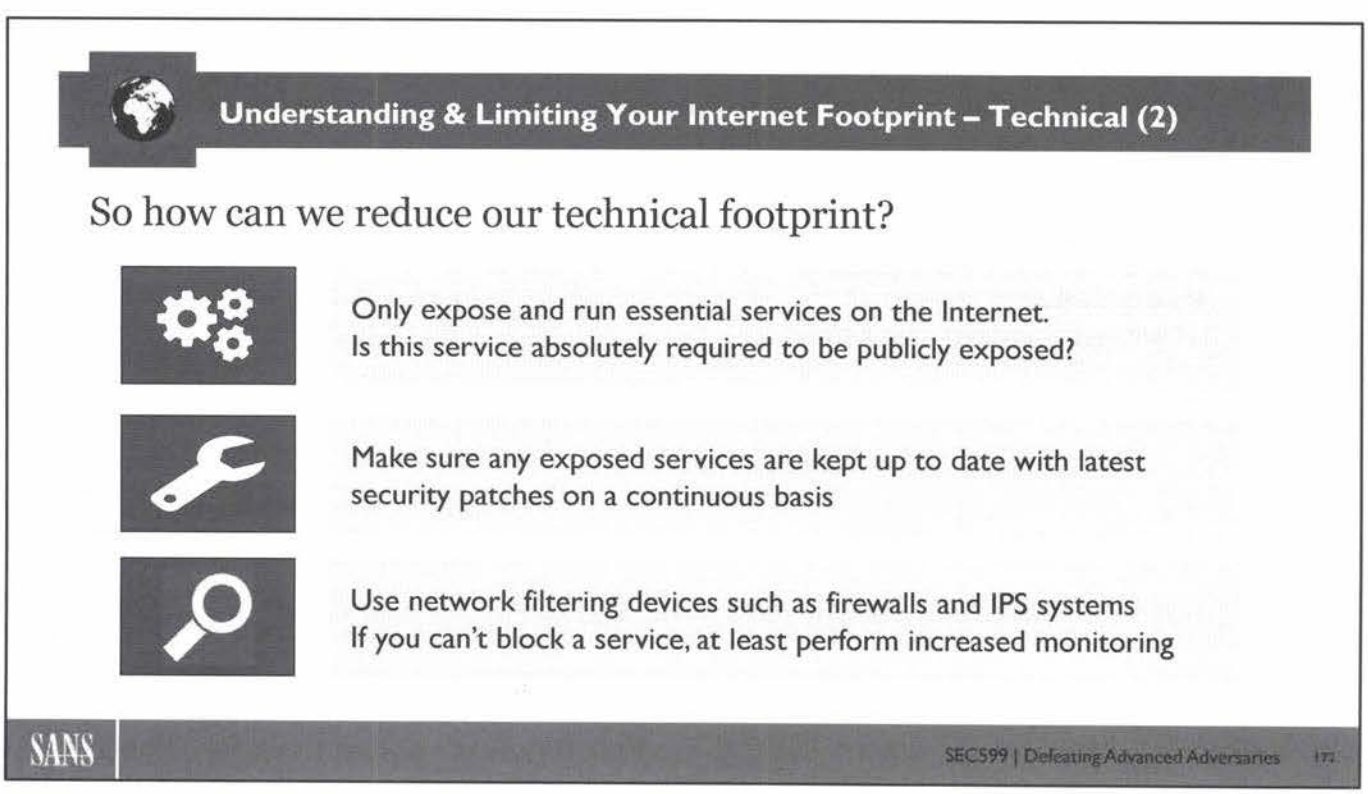
Producing this list is not just an "accounting" exercise: besides compiling a list by collecting information from the different IT teams, it is important to also approach this problem from a practical point. Your opponents will scan all your IP addresses to map your Internet footprint; this is something that is good to do too. It will allow you to discover services that escaped control from your IT teams. These scanning exercises should be conducted on a regular basis, as your Internet footprint is dynamic. It changes over time. Exercises like these can be performed by staff or contracted to third parties specialized in scanning services.

The attack surface is all the resources and services you expose to the Internet and that adversaries can use to try to enter your systems and compromise your operations.

An increasing problem here is that cloud-based services allow virtually anyone to easily launch / host online services. We have seen many organizations where different departments are setting up IT systems in the cloud outside of the IT department's control. This is a serious risk we call "Shadow IT": No one controls these systems:

- Are they properly configured / hardened?
- What data is stored on these platforms?
- Are they patched?
- Is there logging & monitoring in place?

As security professionals, we have to limit this as much as possible!

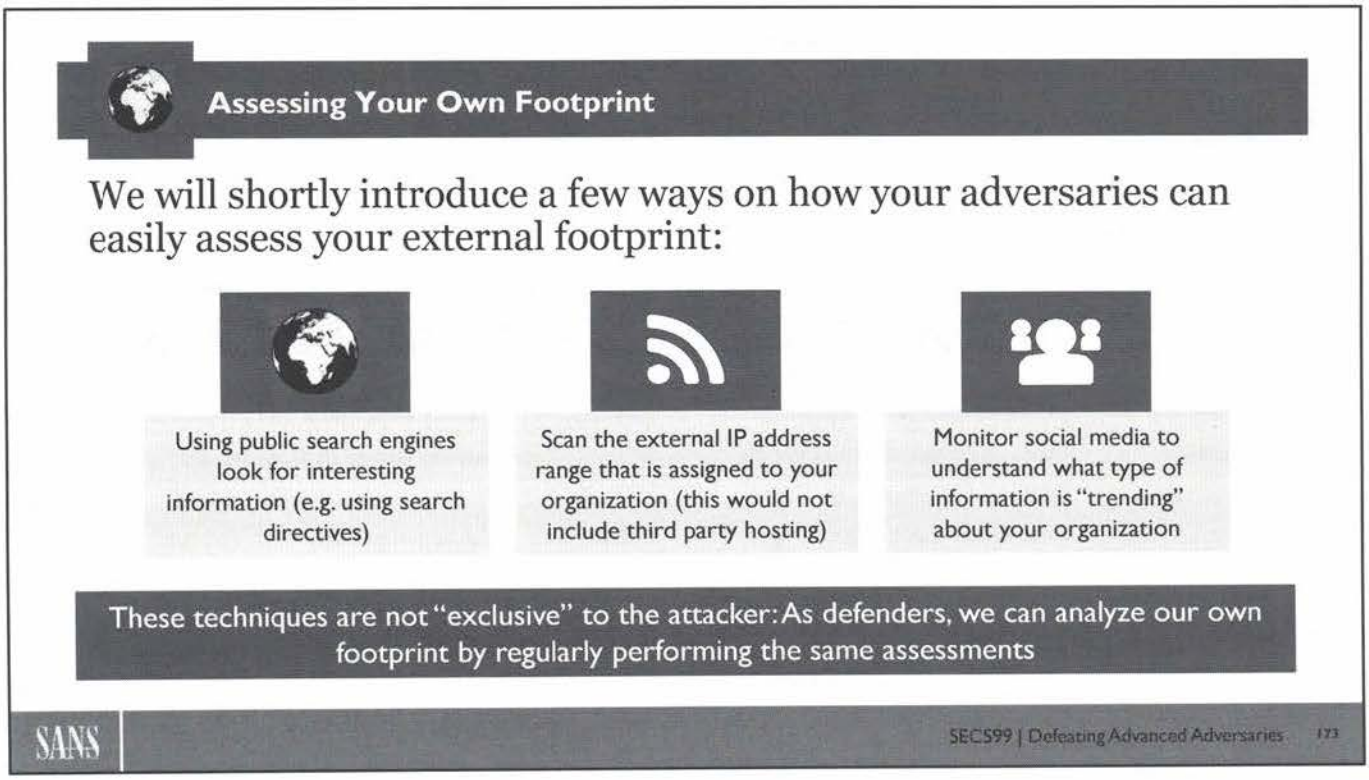


Understanding & Limiting Your Internet Footprint – Technical (2)

As every service exposed to the Internet will be attacked, it stands to reason to disable unnecessary services. All Internet exposed devices are permanently scanned by countless scanners under control of criminals and computers infected with malware. Disabling unnecessary services is not just closing ports, but also configure services to limit the features they offer to features that are required for the operation of your company's Internet presence. For example, if you need a file server just to enable clients to upload documents, don't enable all features of the file server. Only enable uploading of files, don't enable listing of files or downloading of files. If the file server application contains unknown vulnerabilities in its file listing functions, for example, it wouldn't be exploitable by an Internet attacker.

All software contains bugs, and many bugs lead to vulnerabilities that can be exploited. Software vendors and open source projects that maintain their projects will fix bugs they discover or are reported to them. Keeping your software up-to-date makes sure that known vulnerabilities are removed. Applications are not maintained forever. Besides patching, you need to keep up with major releases because old software that is end-of-life is no longer maintained.

Besides disabling features and services, you can also protect from the Internet by filtering the network traffic directed to them with firewalls, web application firewalls, intrusion prevention services… Furthermore, if you cannot block a service from being accessed from the Internet (for valid business reasons), consider implementing increased logging & monitoring on these systems.

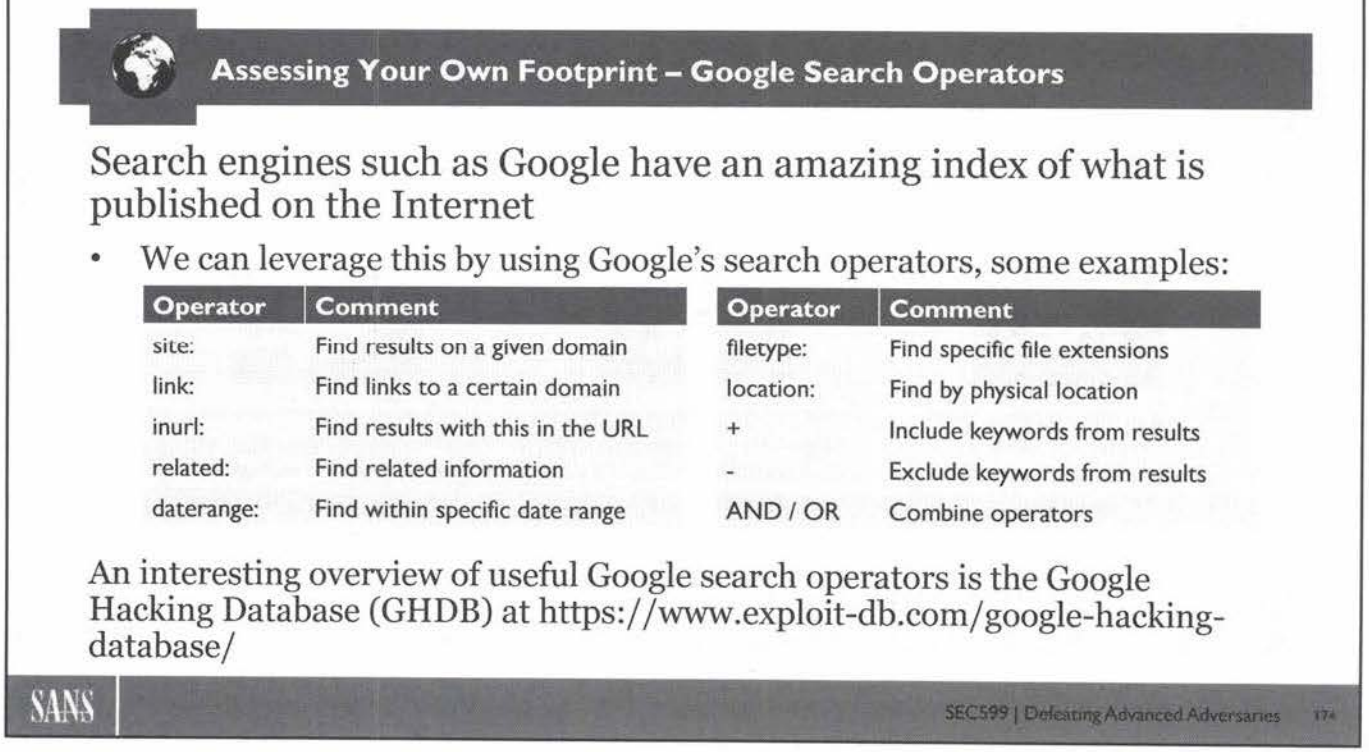


Assessing Your Own Footprint

As every service exposed to the Internet will be attacked, it stands to reason to disable unnecessary services. All Internet exposed devices are permanently scanned by countless scanners under control of criminals and computers infected with malware. Disabling unnecessary services is not just closing ports, but also configure services to limit the features they offer to features that are required for the operation of your company's Internet presence. For example, if you need a file server just to enable clients to upload documents, don't enable all features of the file server. Only enable uploading of files, don't enable listing of files or downloading of files. If the file server application contains unknown vulnerabilities in its file listing functions for example, then these vulnerabilities cannot be exploited because the feature cannot be accessed from the Internet.

All software contains bugs, and many bugs lead to vulnerabilities that can be exploited. Software vendors and open source projects that maintain their projects will fix bugs they discover or are reported to them. Keeping your software up-to-date makes sure that known vulnerabilities are removed. Applications are not maintained forever. Besides patching, you need to keep up with major releases because old software that is end-of-life is no longer maintained.

Besides disabling features and services, you can also protect from the Internet by filtering the network traffic directed to them with firewalls, web application firewalls, intrusion prevention services, …



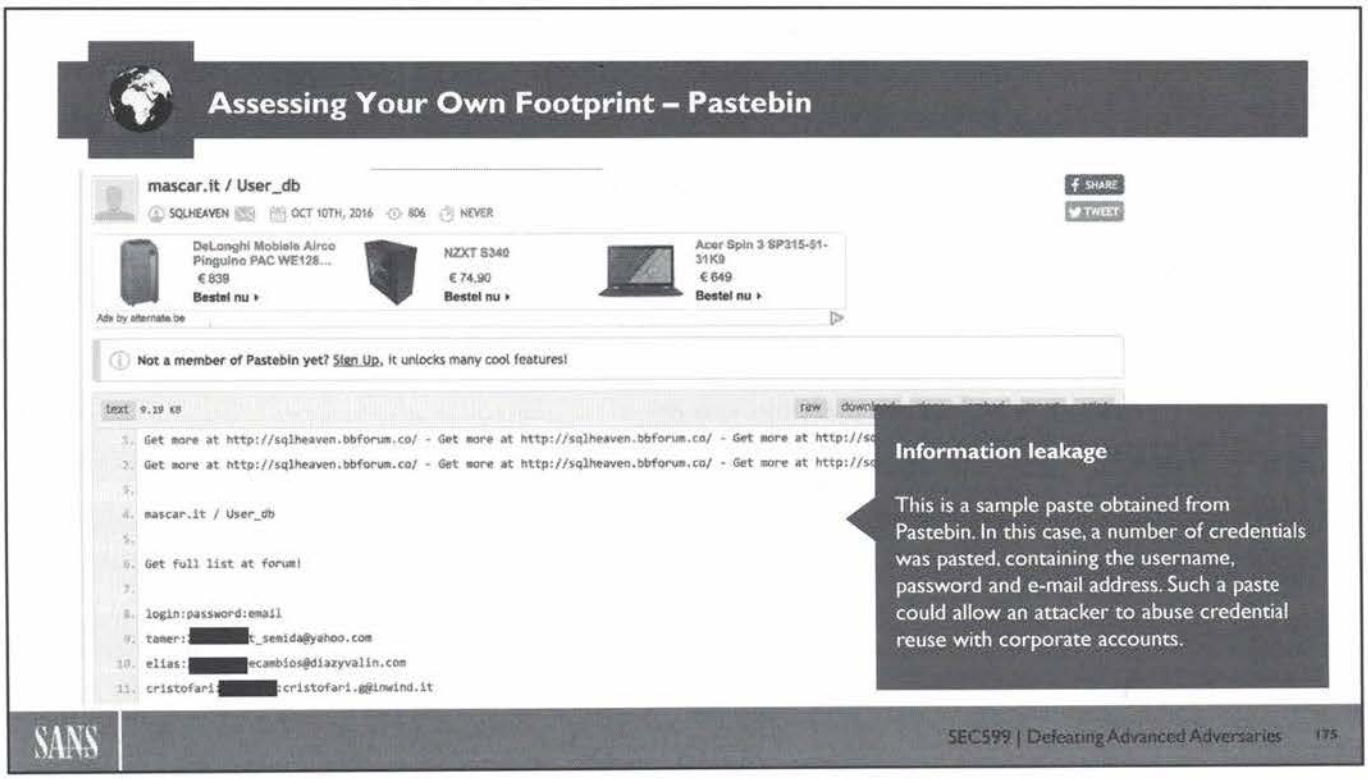
Assessing Your Own Footprint – Google Search Operators

Another method to scope your Internet footprint is to look up what information about your Internet-facing devices other actors have collected. There are numerous indexing services on the Internet that constantly spider the Internet to index information. Well-known ones like Google and Bing index the content of web servers.

This is a tool that is also available to you. Why wouldn't you spend some time to use Google and see what information it has gathered about your company or organization? You can refine your searches with "search operators". We have listed a sample of search operators above, but there's a few more.

site: Find results on a given domain
link: Find links to a certain domain
inurl: Find results with this in the URL
related: Find related information
daterange: Find within specific date range
filetype: Find by specific file extensions
location: Find by physical location
+ Include keywords from results
- Exclude keywords from results
AND / OR Combine operators

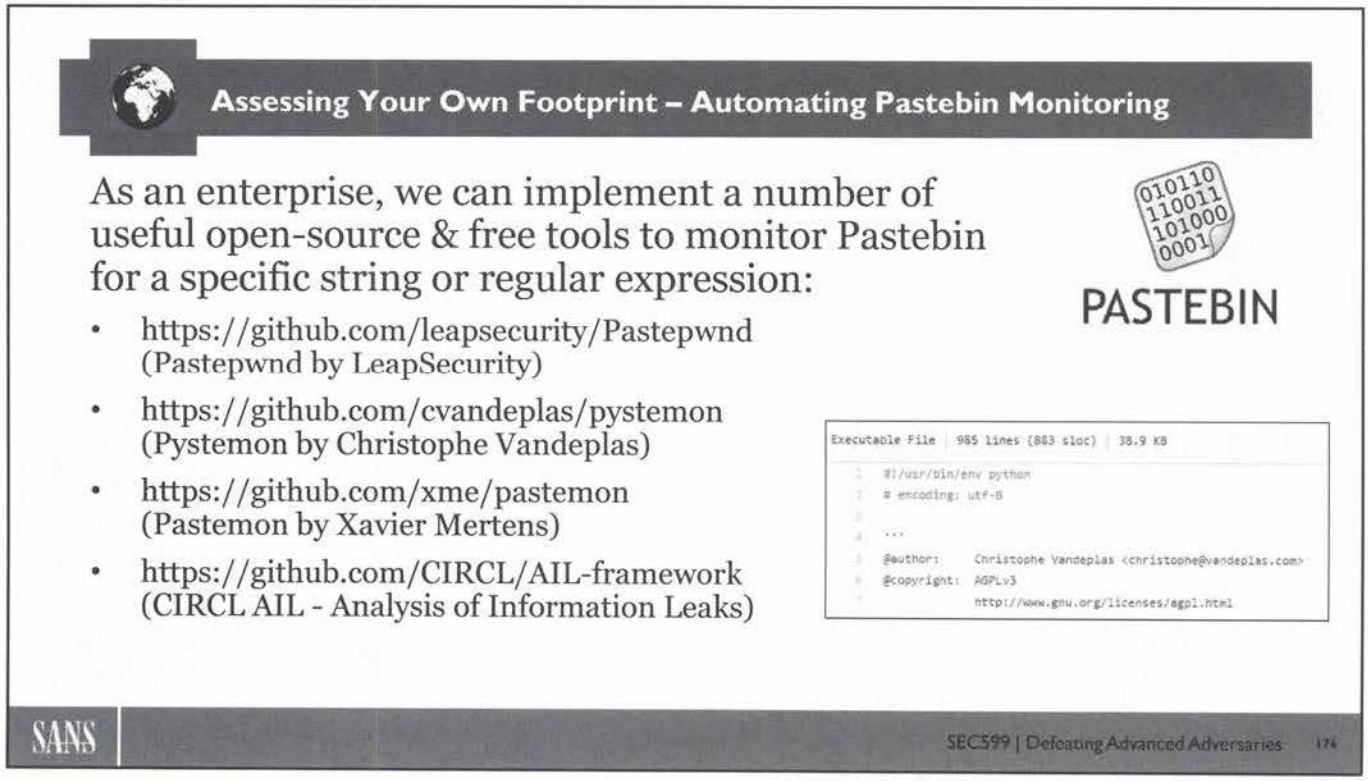
An interesting overview of useful Google search directives is the Google Hacking Database (GHDB) at https://www.exploit-db.com/google-hacking-database/



Assessing Your Own Footprint – Pastebin

Another tool that often causes an (unintentional) increase in your internet footprint is Pastebin (pastebin.com). People use this tool to paste pieces of text and share them with others or just keep them in a pastebin for themselves. These pastes, however, are publicly available and can be viewed by anyone with internet access. Pastes often contain source code, error logs, configuration scripts, etc. Employees, such as developers, may unintentionally leak information relating to an application under development or system configurations. If the paste can be traced back to your company, this information might be abused by an attacker.

On the other hand, this service is used by attackers as well. User information, such as e-mail addresses, names, or credentials are often spread through Pastebin. This information is usually obtained through a hack or data leak against a certain company. By monitoring Pastebin for mentions of your company or employees, it's possible to determine what kind of data is being leaked, whether intentional or by accident. Even though it's possible to scrape data from Pastebin pages, your IP address may be rate-limited or even banned. Using a lifetime Pastebin Pro account allows you to whitelist your IP address and use the API to retrieve pastes.

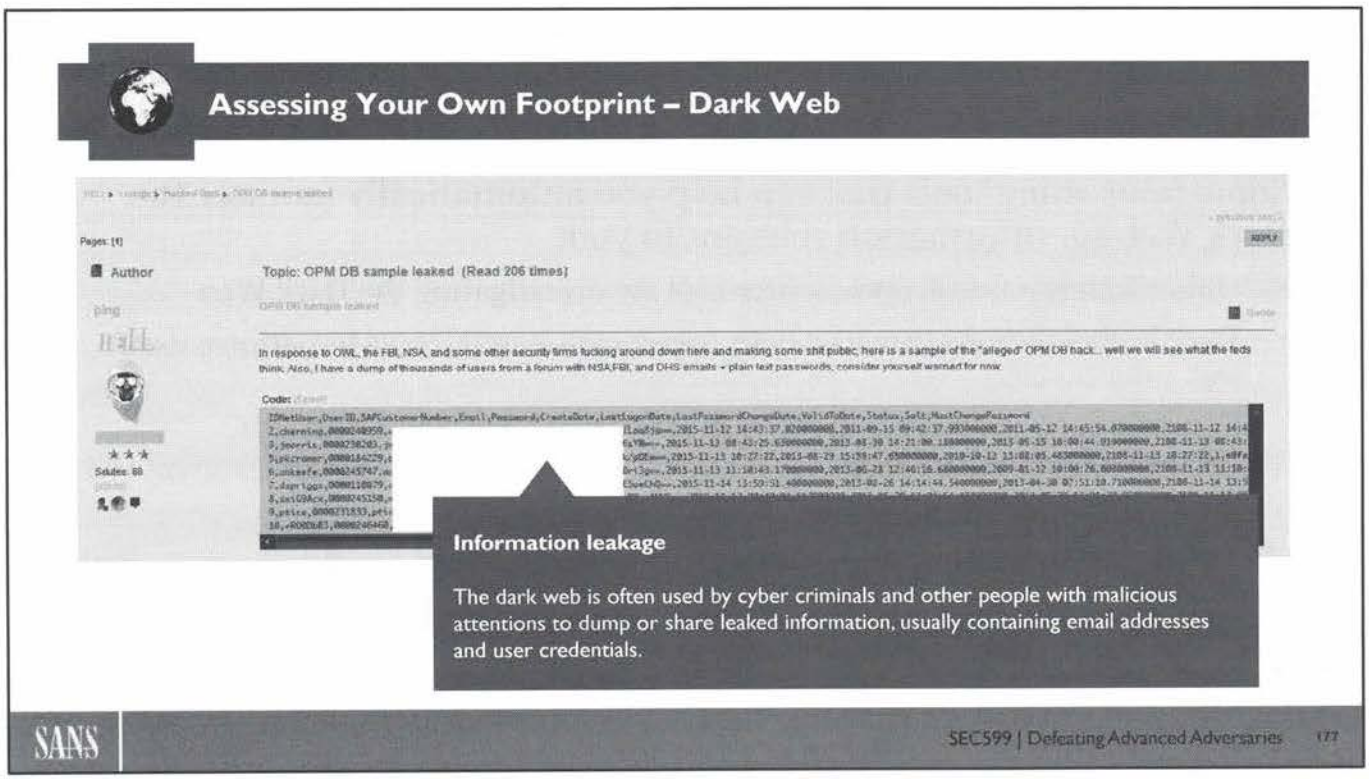


Assessing Your Own Footprint – Automating Pastebin Monitoring

As an enterprise, we are looking for solutions that are automated and do not require too much manual user interaction. It would a bit overkill to have someone check Pastebin manually on a periodic basis. Several security experts have created some useful scripts where you can define regular expressions or strings that need to be matched on Pastebin:

- Pastepwnd by LeapSecurity (available at https://github.com/leapsecurity/Pastepwnd)
- Pystemon by Christope Vandeplas (available at https://github.com/cvandeplas/pystemon)
- Pastemon by Xavier Mertens (available at https://github.com/xme/pastemon)
- CIRCL AIL-framework (available at https://github.com/CIRCL/AIL-framework)

Depending on whether you have a Pastebin Pro account, these scripts can scrape the website or directly query the Pastebin API.



Assessing Your Own Footprint – Dark Web

We just mentioned that attackers themselves sometimes use Pastebin to distribute leaked data. As opposed to Pastebin, the dark web is highly unlikely to contain accidentally leaked information and will thus be a source of information that is put there by people with malicious intentions. It can be a good idea to perform monitoring on dark websites for mentions of your company or employees.

A tool that is aimed at performing this task is called OnionScan (https://github.com/s-rah/onionscan). OnionScan wants to help researchers and investigators monitor and track dark websites and is able to scan the dark web for hidden services. The following blog contains a series on OSINT automation using OnionScan: http://www.automatingosint.com/blog/2016/07/dark-web-osint-with-python-and-onionscan-part-one/

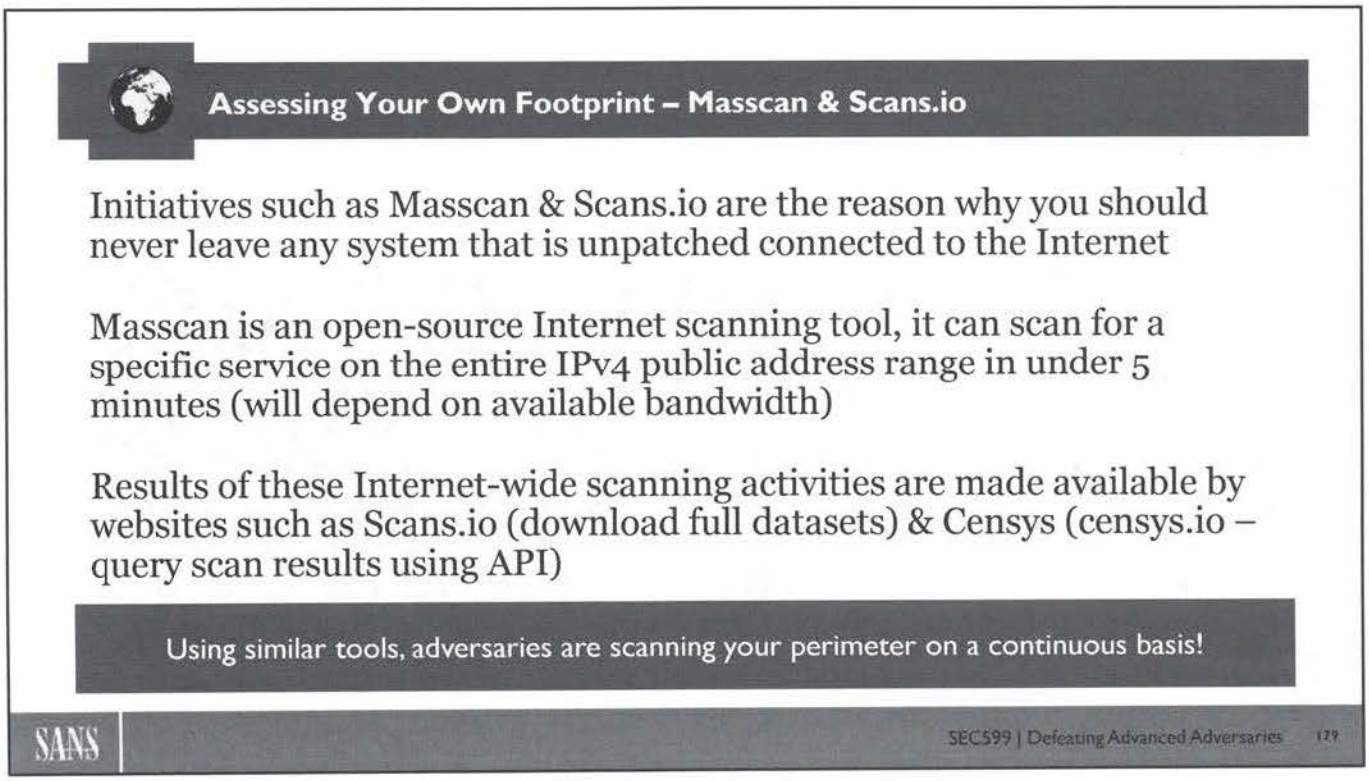
Assessing Your Own Footprint – Automating Dark Web Monitoring

As with pastebin monitoring, several tools are available that can help you automate your monitoring efforts on the dark web. We would like to list a few available solutions that are free / open source:

- OnionScan is a free & open source tool for investigating the Dark Web;
- Dark Web Solutions is an online website offering both free & commercial tools to perform dark web monitoring.

There are of course a wide variety of commercial players that offer similar features. Typically, they provide trial accounts to test & assess the value of their service offering. Some examples include:

- DarkWebID (by idagent.com)
- Strixus (by Massive Alliance)
- DarkWeb (by RecordedFuture)
- Spycloud (mainly focused on credential stealing)
- Intel471
- …

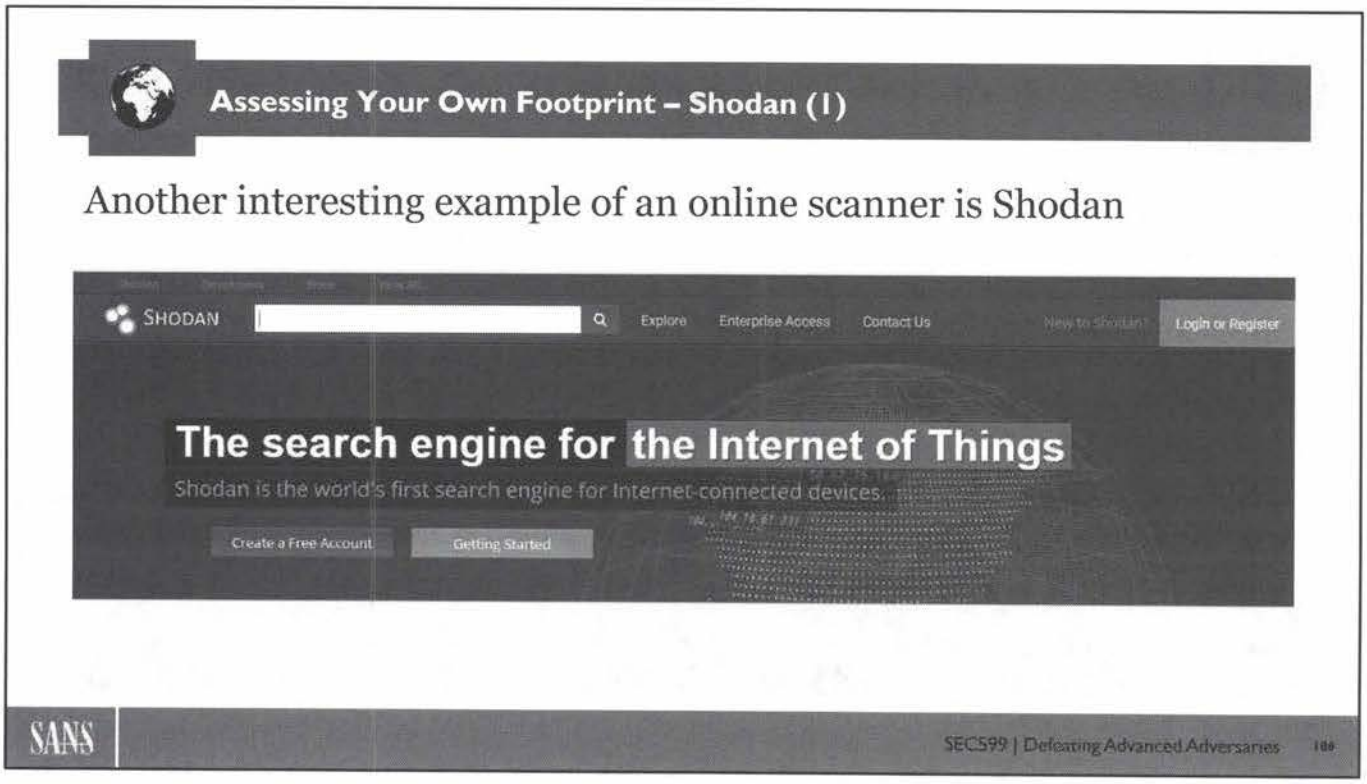


Assessing Your Own Footprint – Masscan & Scans.io
Initiatives such as Masscan & Scans.io are the reason why you should never leave any system that is unpatched connected to the Internet.

Masscan is an open-source Internet scanning tool, it can scan for a specific service on the entire IPv4 public address range in under 5 minutes (will depend on available bandwidth). It is commonly used to scan the Internet to understand the impact of newly identified vulnerabilities (e.g. what systems on the Internet are running this service). Although it appears to offer the same functionality as port scanning tools like NMAP, it uses a highly optimized scanning algorithm and its own custom TCP/IP stack. You could download and install Masscan (or even NMAP) to scan your own external IP ranges.

Instead of scanning your perimeter yourself, you could also leverage public websites that expose this type of information: Results of these Internet-wide scanning activities are made available by websites such as Scans.io (download full datasets) & Censys (censys.io – query scan results using API).

Using similar tools, adversaries are scanning your perimeter on a continuous basis!



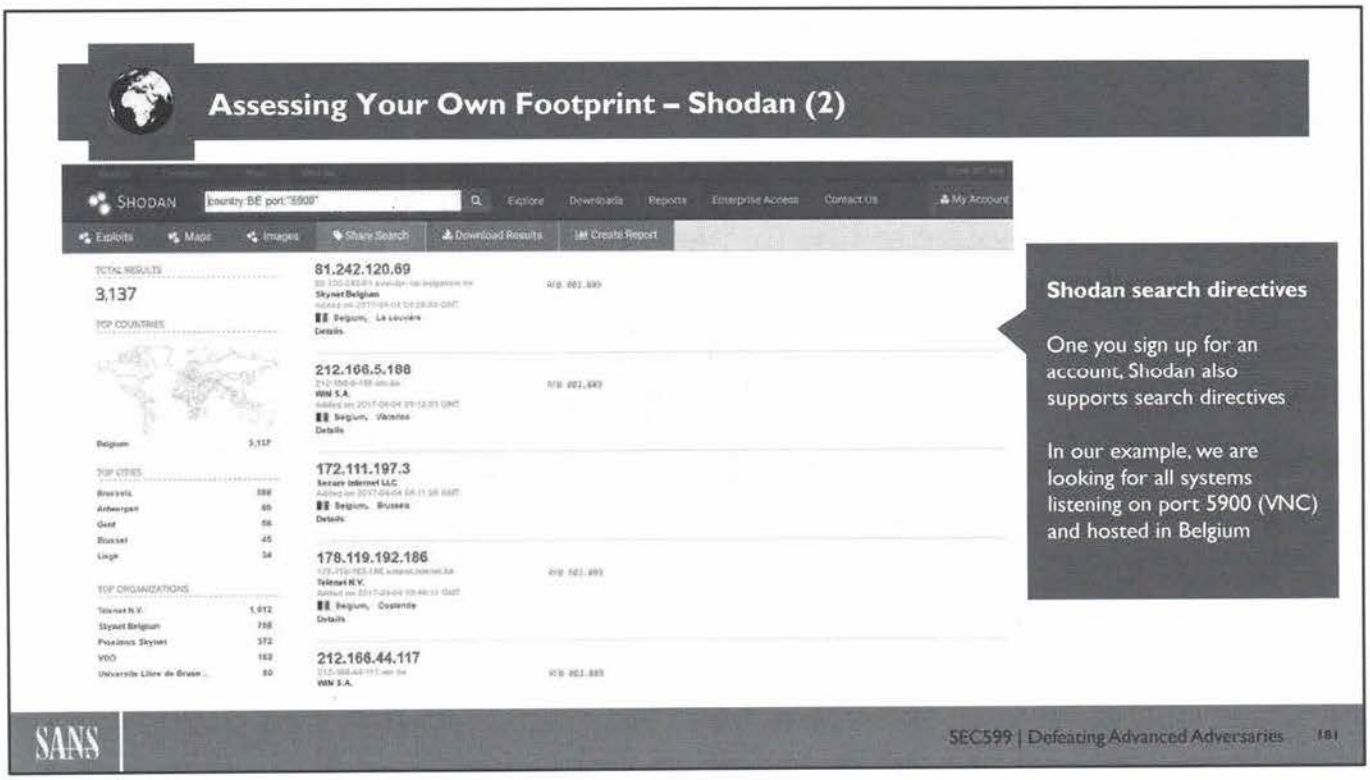
Assessing Your Own Footprint – Shodan (1)
Besides well-known search engines like Google and Bing, there are specialized search engines that index Internet-facing resources like webcams, routers, ICS devices... Shodan (www.shodan.io) is the most popular scanner for these devices.

While classic indexing services like Google and Bing will index the content of web servers, Shodan will capture information about the services (metadata). Shodan will scan ports of web servers, ssh servers, ftp servers, telnet servers… establish a connection, and index the meta data shared by the service. These are called "service banners", and typically announce the implementation and version of the service. This is, for example, a service banner returned by OpenSSH running on a server:

SSH-2.0-OpenSSH_5.3

By indexing this information, Shodan offers its users the capability to look for particular services on the Internet. For example, it can be used to search for older, vulnerable version of OpenSSH present on the Internet.

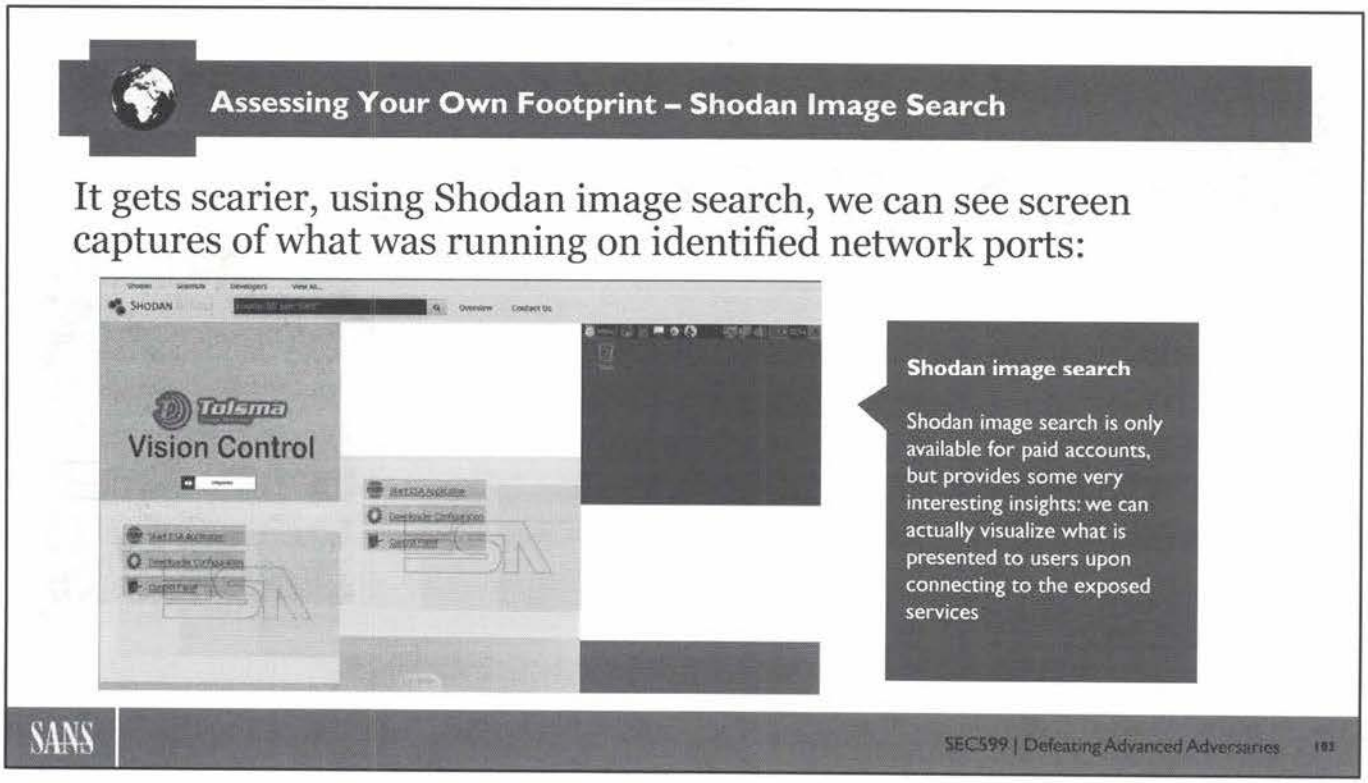
Shodan is a free service. It returns up to 10 results for searches performed without registration, up to 50 results for searches performed with a free account, and more with paying accounts. It also has a Windows GUI application: Shodan Diggity.



Assessing Your Own Footprint – Shodan (2)

In the above example, we are running a Shodan search directive "country:BE port:5900", resulting in 3137 results! We are looking for all systems listening on port 5900, which are hosted in Belgium. We can of course further tailor this to include specific IP ranges of your organization (or ISP).

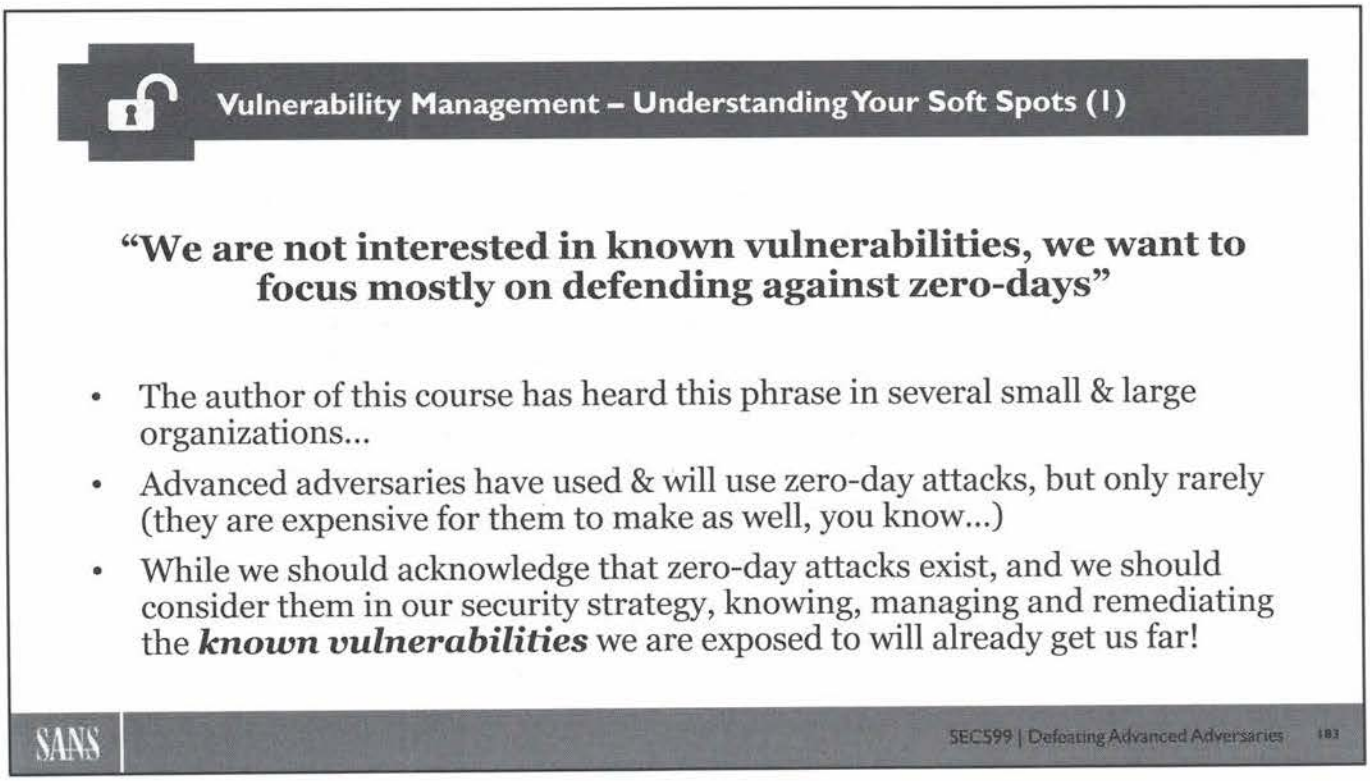
VNC is an often insecurely configured remote administration protocol, so it's likely something we'd need to further investigate…



Assessing Your Own Footprint – Shodan Image Search

It gets scarier, using Shodan image search, we can see screen captures of what was running on identified network ports! Note that Shodan image search is only available for paid accounts, but provides some very interesting insights: we can actually visualize what is presented to users upon connecting to the exposed services!

This is an excellent tool if we want to show people with less technical expertise what Shodan is doing and what you are exposing as an organization!

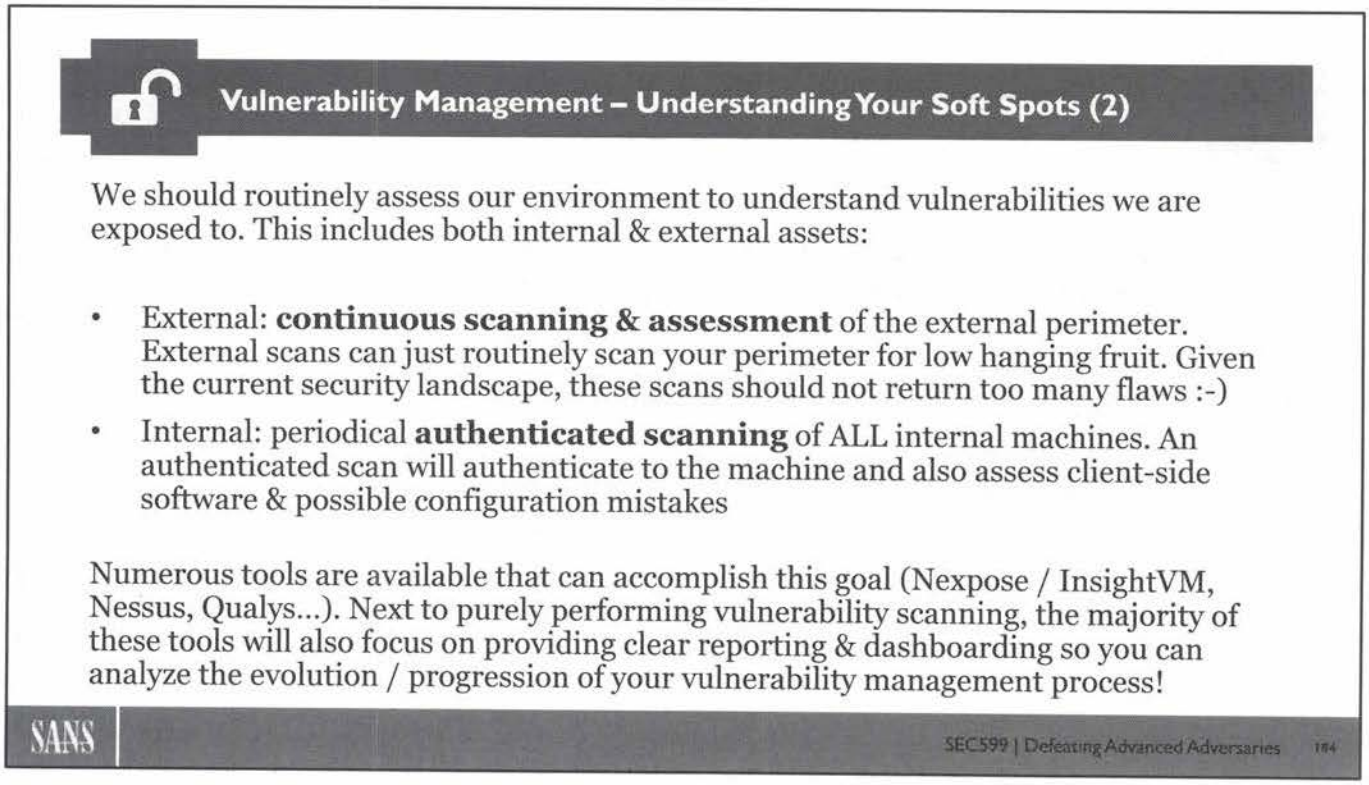


Vulnerability Management – Understanding Your Soft Spots
A quote we hear all too often is:

"We are not interested in known vulnerabilities, we want to focus mostly on defending against zero-days".

Zero-days exist and we should consider them in our security strategy and the defenses we put in place. They are however expensive to develop as well. Adversaries need to also consider their ROI (Return On Investment). They will not hesitate to deploy zero-days against prized assets, but only if they really believe the "juice is worth the squeeze".

We should start by building an effective vulnerability management program, where we can already make sure known vulnerabilities are mitigated. There are few organizations out there that can say they have mitigated every single vulnerability reported by an automated vulnerability scanners…

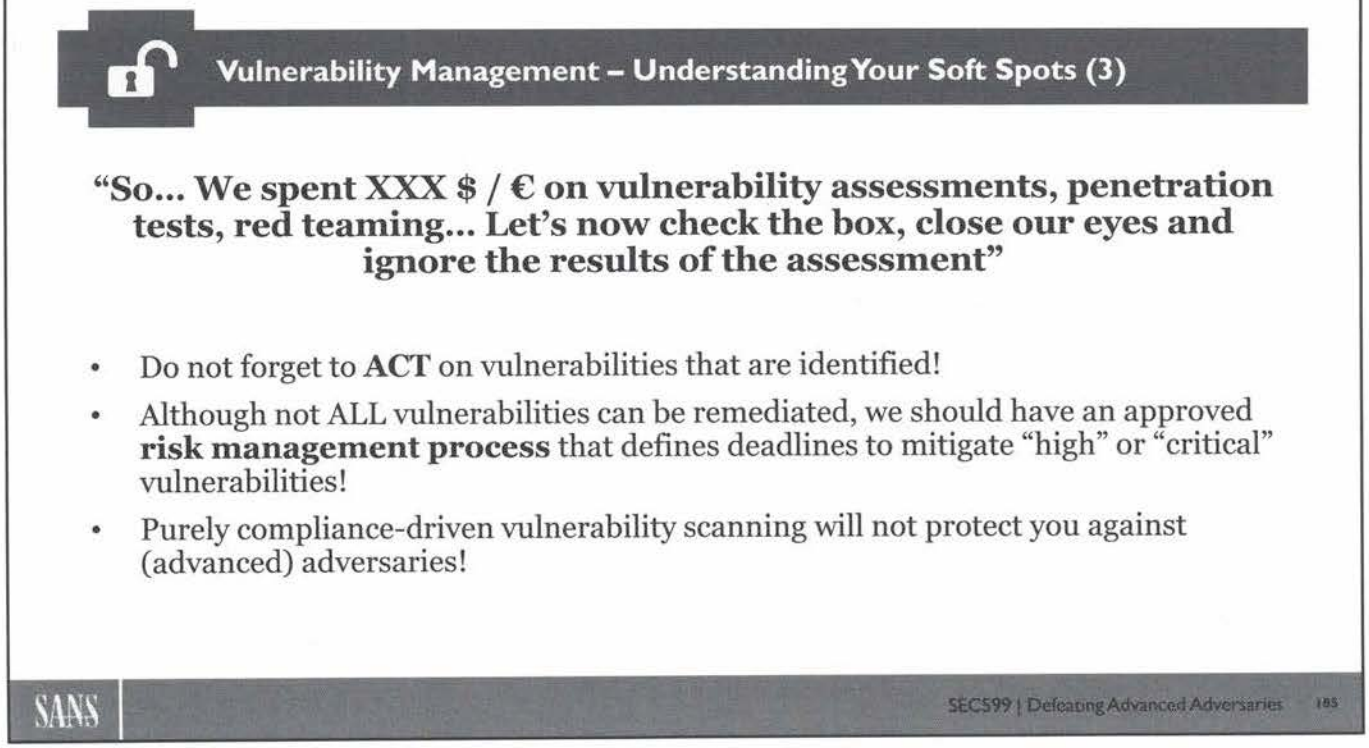


Vulnerability Management – Understanding Your Soft Spots (2)
We should routinely assess our environment to understand vulnerabilities we are exposed to. This includes both internal & external assets:

- External: continuous scanning & assessment of the external perimeter. External scans can just routinely scan your perimeter for low hanging fruit. Given the current security landscape, these scans should not return too many flaws :-)
- Internal: periodical authenticated scanning of ALL internal machines. An authenticated scan will authenticate to the machine and also assess client-side software & possible configuration mistakes

Numerous tools are available that can accomplish this goal (Nexpose / InsightVM, Nessus, Qualys…). Next to purely performing vulnerability scanning, the majority of these tools will also focus on providing clear reporting & dashboarding so you can analyze the evolution / progression of your vulnerability management process!

We will further discuss vulnerability scanning on day 3, when we will also perform an authenticated vulnerability scan against our target environment.



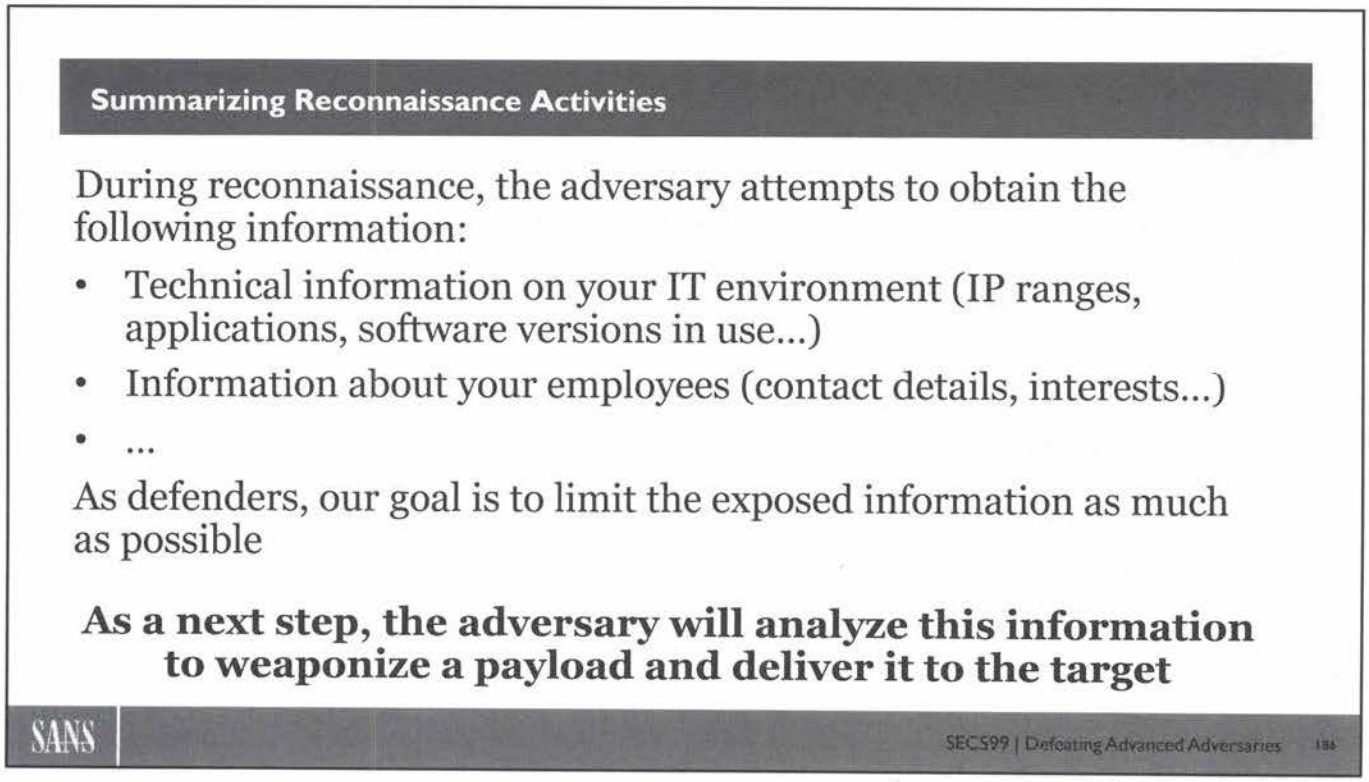
Vulnerability Management – Understanding Your Soft Spots (3)
As another part of our quote box, here is another fan favorite:

"So… We spent XXX $ / € on vulnerability assessments, penetration tests, red teaming… Let's now check the box, close our eyes and ignore the results of the assessment"

To be fair, this is not really a quote, but sometimes a mindset we observe at some organizations. Vulnerability scanning / security assessments are useless if we do not act upon the risks / vulnerabilities that are identified.

We recognize not all vulnerabilities are to be remediated (as some might only have a low impact and hinder / slow down the business), every organization should have an approved risk management process that defines deadlines to mitigate "high" or "critical" vulnerabilities. Non-mitigation of such risks should require a formal acceptance by executives who can be held accountable.

Purely compliance-drive vulnerability scanning will not protect us against (advanced) adversaries.



Summarizing Reconnaissance Activities

Reconnaissance is an attacker's first step towards an attack on your organization. Attackers will try to obtain information on your environment, containing both technical information and information about the people that are part of your operations:

- Technical information on your IT environment contains, for example, IP ranges, applications, and software versions in use.
- Information about your employees, such as contact details and interests can be abused by an attacker during phishing or social engineering campaigns. This type of information could even be linked to your technical environment. Job descriptions can provide knowledge on the technologies that are used in your environment and where your organization might be understaffed.

Even though it is hard to prevent reconnaissance activities, steps can be taken in order to limit an attacker's possibilities. As defenders, our goal is to limit and understand what we expose online.

Once an attacker has finished the reconnaissance phase, the next step will be to make use of the obtained information by selecting a target attack vector and weaponizing a payload.

Course Roadmap

- **Day 1: Knowing the adversary, knowing yourself**
- Day 2: Averting Payload Delivery
- Day 3: Preventing Exploitation
- Day 4: Avoiding Installation, foiling Command & Control & thwarting lateral movement
- Day 5: Exfiltration, Cyber Deception & Incident Response
- Day 6: APT Defender Capstone

SEC599.1

Course Outline & Lab Setup
- Course Overview & Objectives
- Attendee System Setup

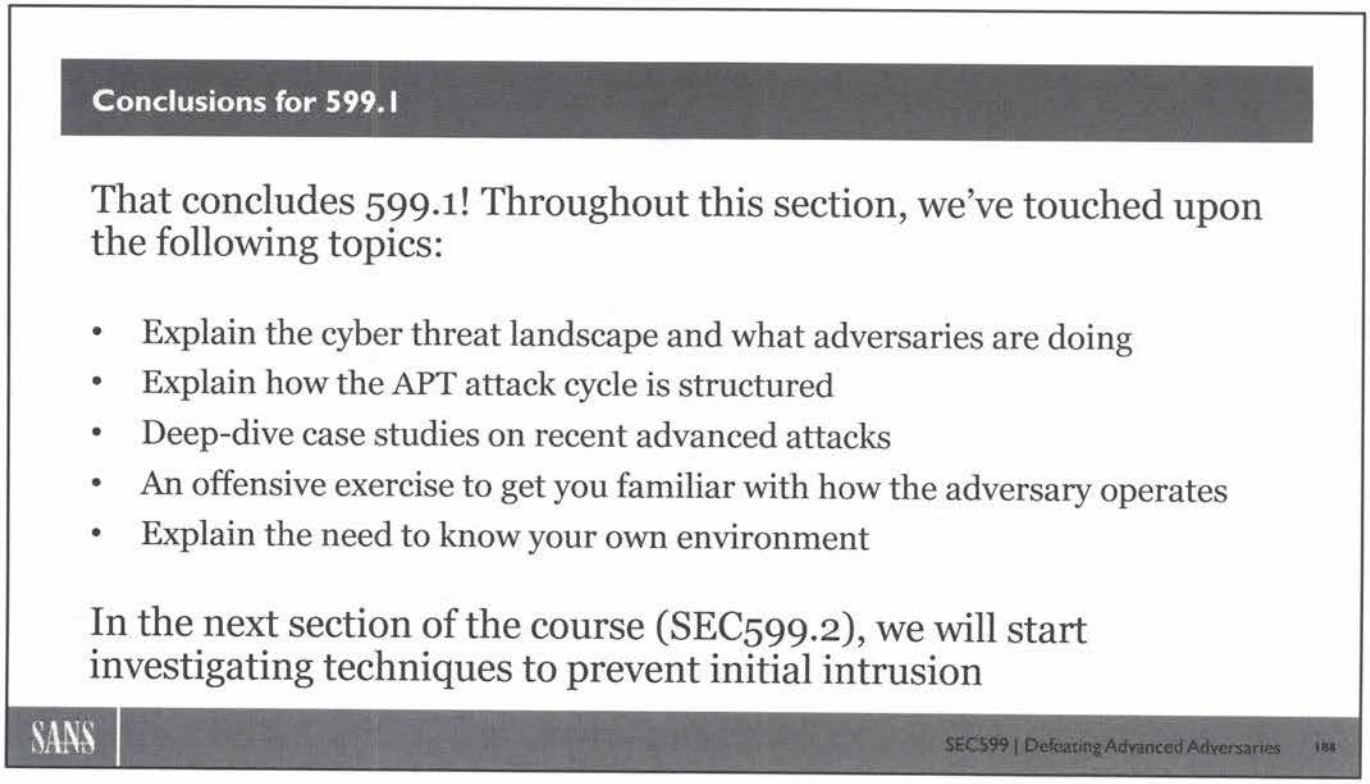
Current Threat / Attack Landscape
- Key Terminology
- What is happening out there?

Introducing the APT Attack Cycle
- Recent Case Studies – In-Depth
- Exercise: Analyzing The Behavior of Famous Malware
- Exercise: One Click Is All It Takes…

A Defensible Architecture & Environment

Preparation - Knowing Yourself
- Understanding Your Own Environment
- Determining What is "Normal"
- Understanding & Limiting Your Internet Footprint
- A Word on Vulnerability Management

This page intentionally left blank.



Conclusions for 599.1

So, that the first day of SEC599 (599.1)! Throughout this section, we've attempted to illustrate both how you work yourself, but also how your adversaries operate.
More specifically, we've touched upon the following topics:

- Explain the cyber threat landscape and what adversaries are doing
- Explain how the APT attack cycle is structured
- Deep-dive case studies in recent advanced attacks
- An offensive exercise to get you familiar with how the adversary operates
- Explain the need to know & understand your own environment

In the next section of the course (SEC599.2), we will start investigating techniques to prevent initial intrusion.

Course Resources and Contact Information

AUTHOR CONTACT
Erik Van Buggenhout
evanbuggenhout@nviso.be
Stephen Sims
ssims@sans.org

SANS INSTITUTE
8120 Woodmont Ave., Suite 310
Bethesda, MD 20814
301.654.SANS (7267)

CYBER DEFENSE CONTACT
Stephen Sims
ssims@sans.org

SANS EMAIL
GENERAL INQUIRIES: info@sans.org
REGISTRATION: registration@sans.org
TUITION: tuition@sans.org
PRESS/PR: press@sans.org

This page intentionally left blank.