Stuxnet, a Weapon of Cyber Warfare and an Opportunity to Learn About Security in DevOps

George Malki, Mayuri Salunke, Hasan Kalzi

May 2022

Geomal@kth.se, salunke@kth.se,
Kalzi@kth.se

Introduction

With the unrest that is going on in Ukraine in the last couple of months, we are reminded of the horrors that are caused by wars. Thanks to the technological breakthroughs in the last couple of decades, we are able to witness the war live on our smartphones. Some of us who are old enough get to relive the days of the cold war, now that the threat of using nuclear weapons is higher than ever. With that being said, there is another form of warfare that could cause as much havoc: cyber-warfare.

Cyber attacks according to can be defined as "An act in cyberspace that could reasonably be expected to cause harm" [3]. Cyberwarfare is when one nation does a cyber attack on another intending to create havoc on government and civil infrastructure. There are seven types of cyber warfare attacks: espionage, sabotage, denial-of-service attacks, electrical power grid, propaganda attacks, economic disruption, and surprise attacks. However, it's hard to find real-life examples of such attacks, since no direct loss of life can be witnessed and no nation will willingly take responsibility for an attack on a foreign country. And one such example that we will be discussing is a worm that has come to become one of the most infamous cyber warfare attacks that a nation has used against another, known as the Stuxnet. The main focus of the essay will be 'an analysis of the security breaches that Stuxnet exploited'. The essay shall be discussing and exploring the different DevSecOps concepts and how they were used to create one of the world's first cyberware weapons.

Background

The Stuxnet Worm

Stuxnet worm was first discovered by a researcher for a belarusian cyber security company VirusBlokAda who created a disscussion thread on a security forum in june of 2010[1]. The malware instantly drew a lot of attention, not only for its sophistication, but also for its use of four previously undiscovered zero-day exploits which was unheard of before [1]. Unlike many pieces of malware that were mainly designed and used for espionage nations used to spy on each other, the makers of stuxnet had another goal in mind. Stuxnet was

[1] https://www.wilderssecurity.com/threads/rootkit-tmphider.276994/post-1712134

designed and built with the intention to sabotage centrifuges in the power facilities of Natanz in Iran. It reportedly destroyed numerous centrifuges by causing them to burn themselves out and over time, the virus was modified to target other facilities like water treatment plans, power plants and gas lines [2].

Stuxnet was a multi-part worm which traveled through USB:s and was spread through Microsoft Windows computers. It would search the infected computers for signs of Siemens Step 7 software(an engineering tool used for configuring and programming controllers), and would update its code over the internet and send damage-inducing instructions to the PC controlled equipment. At the same time, the virus sent false feedback to the main controller and hence, anyone monitoring the equipment would not realize the presence of any problem until the equipment self-destructs.

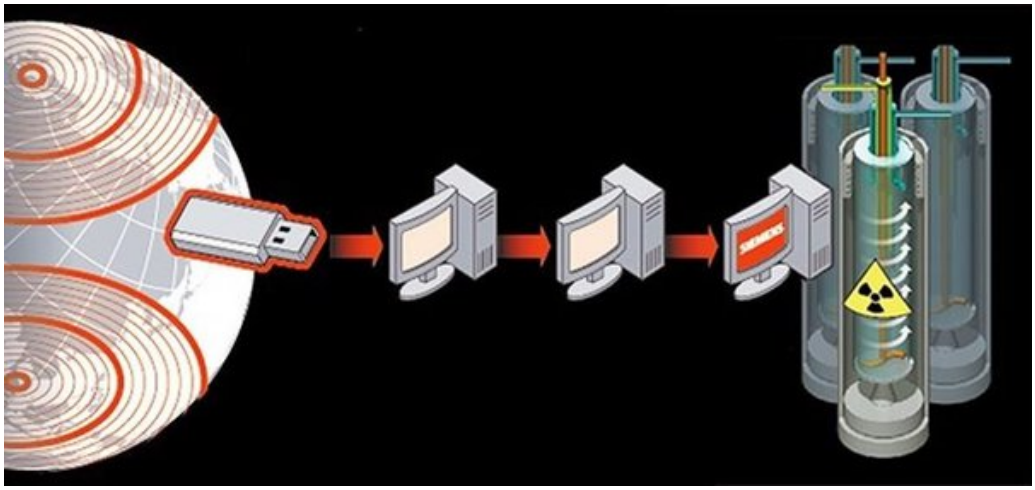

Figure 1: A visualisation of a Stuxnet attack

Stuxnet was one of the first cyberware weapons that was discovered which was able to target and destroy physical infrastructure. And by legitimizing such cyber attacks, an opportunity for a significant blow back has been created [3]. Instead of treating cyberspace as an opportunity for innovation and exchanging information, Stuxnet has now opened the realms of cyberspace for more cyberwars. It has now become a muse for cyber attackers to achieve in order to create a better virus. Hence, in the near future, the cyberwar race will be more difficult to ameliorate than the race of the nuclear weapons.

Further, no country has taken responsibility for Stuxnet but it is believed that it was a result of a joint effort from the USA and Israel to stop or at least delay the Iranian nuclear program.

DevSecOps

Deploying a piece of software, whether it being software changes or new additions, is more likely to contain vulnerabilities, some of which catastrophic, without adequate involvement of a security team. As a result, the concept of collaboration between the development teams inside an organization and security teams has gained popularity in the last couple of years. In a research paper titled "Software Security in DevOps: Synthesizing Practitioners' Perceptions and Practices" devOpsSec as "the concept of integrating security principles through increased collaboration between the development teams, operations teams, and security teams of a DevOps organization" [2].

[2] https://www.trellix.com/en-us/security-awareness/ransomware/what-is-stuxnet.html

[3] https://www.csmonitor.com/World/Passcode/Passcode-Voices/2015/1104/Opinion-The-troubling-Stuxnet-effect

DevSecOps is "the concept of integrating security principles through increased collaboration between the development teams, operations teams, and security teams of a DevOps organization". DevSecOps stands for development, security, and operations. It's strategy to automation, culture, and platform design to integrate security as a shared responsibility throughout the entire IT lifecycle. It is something like Agile but the main differences between them is that Agile is about flexibility in the development process while s about using security as a fundamental part of these transformations. Using DevSecOps has many advantages like: enhanced speed and agility for security teams, lowering costs and free time for continuous improvement.

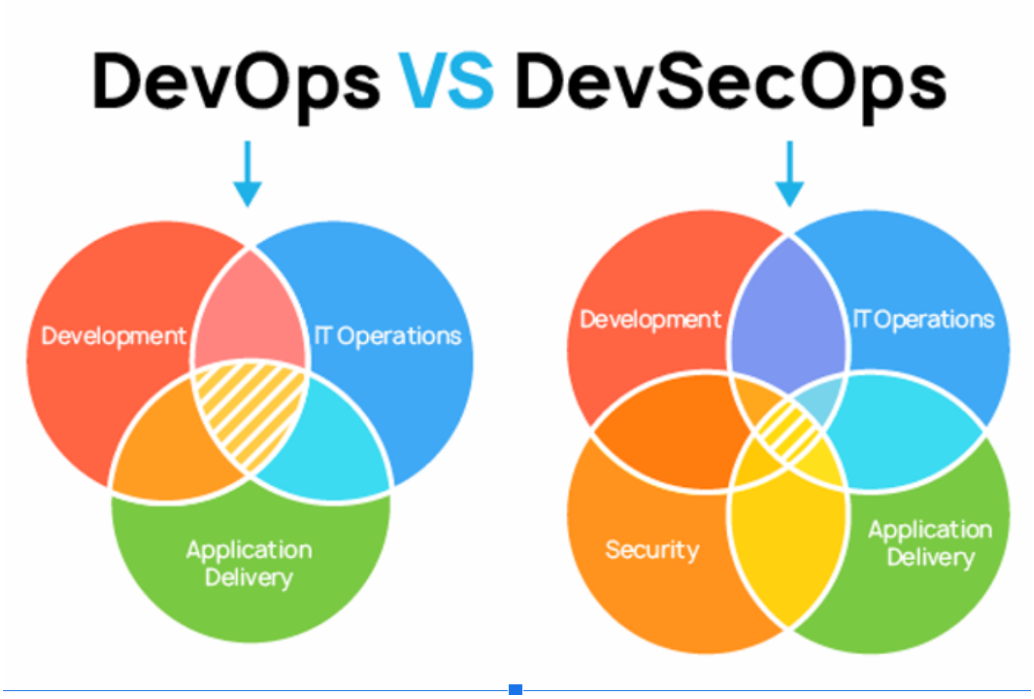

Figure 2: visualisation of a comparison between DevOps and DevSecOps

Privileged Access Management (PAM)

We can define Privileged Access Management (PAM) as a primary element that holds the keys to an IT company. When employed correctly, it allows complete control of data, infrastructure and assets. What happens if those keys fall into the wrong hands? They will be able to use the access to steal sensitive data and cause huge damage to the company. PAM solutions secure, manage and monitor privileged access to prevent attackers from reaching their simple goal, which is gaining access to your most valuable systems and assets they need to successfully carry out attacks[4].

Privileged Access is not isolated, and it exists on all workstations and servers in applications, DevOps pipelines and across hybrid and multi-cloud environments. Whenever we look at an environment, we come across two categories of privilege access. The first is permissions, which allows elevation of other accounts to privileged status. It enables someone to make somebody else privileged if the first one has the privileged permission. The second is permissions allow denial of access to other accounts. This includes any account or privilege that might be destructive (denial of access). So even if the account can't make somebody else privileged, it may stop the whole system from working in some

[4]https://docs.microsoft.com/en-us/microsoft-identity-manager/pam/privileged-identity-management-for-active-directory-domain-services

Figure 3: An Illustration of PAM (Privileged Access Management)

way [5].

Zero-day Exploits

A zero-day vulnerability in simple terms is a flaw [6]. It is an unknown vulnerability in the software or hardware of the system which when exploited can allow attackers to gain unauthorized access allowing them to damage or exploit the system, before someone realizes that something went wrong. Hence, zero-day exploit is a type of cyber attack exploiting the zero-day vulnerability which was unknown to the software and antivirus vendors. It is called 'zero-day' as the attack happens once the attacker detects the flaw and exploits it by releasing a malware, thereby giving no chance for the developer to create a patch to fix the vulnerability. Hence, zero-day exploits are a serious security threat as the success rate of these attacks are very high as there are no defenses placed to prevent it.

Once the attackers have identified the vulnerability, they have to use a delivery mech-

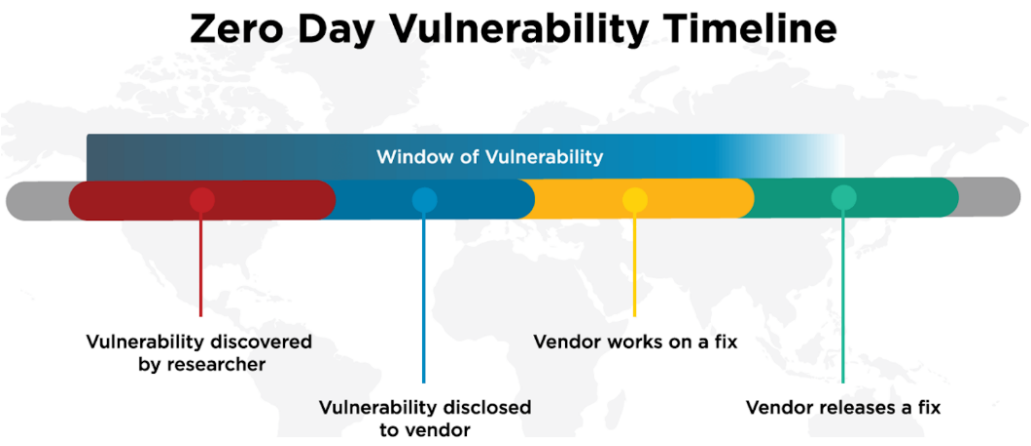

Figure 4: A visualisation of Zero day vulnerability timeline

[5] https://en.wikipedia.org/wiki/Privilege_(*computing*)
[6] https://www.fireeye.com/current-threats/what-is-a-zero-day-exploit.html

anism to exploit the vulnerability. Usually this is done by sending a mail or message that is made to look like it is from someone legitimate. The message usually entails some instructions to convince the user to perform an action like downloading / opening a file or visiting a website, upon which the attacker is able to deploy its malware to exploit the vulnerability. And once the company has been attacked, it needs to first identify where the vulnerability is, which might take a lot of time and then fix the patch as well. Hence, these attacks are very dangerous, and the financial, operational and legal impact is very devastating as these attacks can go undetected for weeks, months or even years. According to Verizon's 2021 Data Breach Investigation Report [7] 95% of the targeted companies lost between $148 to $1,594,647 in Computer Data Breach incidents.

Air-gapped Networks

Air gapping or air walling in its simplest form is the act of physically isolating a single, or multiple computers from unsecured networks and is one of the most common security measures that is employed around the world. These unsecured networks could for example be the internet or an unsecure local network. Essentially, no network connection can be established with the device, whether wired or over WiFi and in order to move data into or from the air-gapped system a physical data medium is required, such as a USB-stick or a hard drive. Some form of authentication is usually also required to ensure that the USB drive might be compromised which would result in a vulnerability in the system.

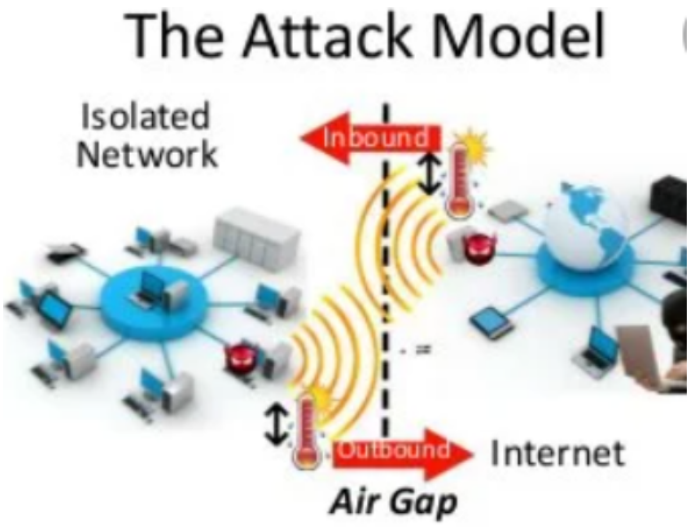

Figure 5: A visualisation of an Air-Gapped network

Air gapped systems can be found everywhere, for example electronic sprinkler controls for watering of lawns, Financial computer systems, such as stock exchanges, industrial control systems, such as SCADA in Oil & Gas fields, military/governmental computer networks/systems as well as life-critical systems such as controls of nuclear power plants[8].

The first question that comes to mind when discussing air gapped environments as a developer is: how can we deploy an application into an air gapped environment whether it being a software update or an entirely new functionality? There are a couple of services (some of which open-source) that allow us to deploy our code to air-gapped systems, such

[7] https://www.verizon.com/business/resources/reports/dbir/2021/masters-guide/summary-of-findings/

[8] https://en.wikipedia.org/wiki/Air$_gap_(networking)$

as Teleport[9] and jfrog[10]. The main idea is that you can create Cluster Images and/or Kubernetes to package and ship an application's code and dependencies from one computing environment to another. You then copy the created image onto a USB-stick and transfer it to the air gapped system. The aforementioned services can then install the Cluster images onto the destination server without needing any internet access.

1 Discussion

With the digitalisation of every aspect of our everyday life in the last couple of decades, the topic of security and software vulnerability is becoming more and more important. This doesn't only include our data but also highly confidential data that is meant to be stored securely and outside the influence of any person/organization. Network breaches and data leaks are nightmare scenarios that a lot of organizations have to think about when designing and developing a new platform. It's therefore not uncommon to have a security team that works hand in hand with the developers of the platform to ensure that no vulnerability can be found.

Though, the probability of writing bug-free/vulnerability-free code diminishes exponentially the bigger the product is. A clear case of this is the Zero-day exploits that were found in Microsoft operative system Windows. Stuxnet did not exploit one, but four different zero-day vulnerabilities to delve and spread into the Microsoft Windows operating system. It used a Windows rootkit (collection of softwares which provides unauthorized access to a computer or an area of the software), antivirus evasion techniques, stolen certificates from trusted CAs and peer-to-peer updates [11]. The stolen certificates from the trusted CAs were used for the malware to appear legitimate and to avoid detection by the traditional intrusion detection systems (IDS). While the peer-to-peer communication enabled the malware to self-update even when the compromised device had no direct access to the internet [12]. And by using these four zero-day vulnerabilities, Stuxnet was able to reportedly ruin Iran's numerous centrifuges.

But how was Stuxnet able to install itself on an Air-gapped system anyways? While it's common knowledge that you shouldn't connect a USB-thumb drive to any of your computers, let alone the computer that is used to control the centrifuges that enrich and create military grade Uranium, at least one Iranian engineer didn't get the memo. The engineer bridged the local and secure network with the outside world for a brief moment allowing the worm to install itself on the device by inserting the USB-drive into the computer.

Another question that comes to mind is: how was the worm able to install itself without notifying the user and how was it able to remain undetected for such a long period? It's because it has its drivers digitally signed with the private keys of two public key certificates from separate well-known companies, JMicron and Realtek. It is unknown how the makers of Stuxnet got a hold of these companies private keys but most speculate that the keys were stolen directly from the companies or bought on the black-market. Since the drivers were digitally signed the worm was able to install kernel mode rootkit drivers successfully without users being notified, bypassing User Account Control and other Windows measures designed to prevent malicious code from being installed and thus it remained undetected for a relatively long period of time.

[9] https://goteleport.com/blog/airgap-deployment/
[10] https://jfrog.com/blog/air-gap-distribution-delivers/
[11] https://www.sciencedirect.com/topics/computer-science/stuxnet
[12] https://www.sciencedirect.com/topics/computer-science/stuxnet

Conclusion

With the advancements that mankind has been able to achieve in the last couple of decades, we have entered a generation of warfare, Cyber warfare. Though on paper it doesn't sound as deadly as the typical warfare, it has the same catastrophic potential as oldschool warfare without the need to shed a single drop of blood. Cyber warfare can come in many shapes and forms but the underlying goal is the same, disrupt the infrastructure of the other nation to cause damage. Through this essay, we talked about the various DecSecOps concepts and how they were exploited to one of the world's first cyber-war weapon which destroyed numerous centrifuges in Iram's Nataz uranium enrichment facility. At the same time, it also illustrates how we can never be 100 percent secure. It makes us realize that our defenses can fall short and that we need to keep finding weakness and faults in our system to prevent an attack such as Stuxnet from occurring again.

References

[1] Marie Baezner and Patrice Robin. Stuxnet. Technical report, ETH Zurich, 2017.

[2] Akond Ashfaque Ur Rahman and Laurie Williams. Software security in devops: Synthesizing practitioners' perceptions and practices. In *2016 IEEE/ACM International Workshop on Continuous Software Evolution and Delivery (CSED)*, pages 70–76. IEEE, 2016.

[3] Michael Robinson, Kevin Jones, and Helge Janicke. Cyber warfare: Issues and challenges. *Computers Security*, 49:70–94, 2015.