Defensive Controls Mapping

Introduction

The following will be a defensive mapping based on five Cyber Security principles that ensure security in more than a theoretical sense, but a practical, realistic sense:

1. Defense in Depth
2. Principle of Least Privilege
3. Separation of Duties
4. Security By Design
5. K.I.S.S Keep It Simple Stupid

Never Should have happened (Map 1)

This NEVER should have happened. Since 2022, Marcos has been disabled by default in Microsoft Office. Something as *simple* (Principle 5) as having the latest update can save your nuclear plant from ruin. Furthermore, a GPO can prevent the likes of Homer from ever activating it manually. Other than that, you can enroll your employees in a Security Awareness Class along with various forms of email security measures, such as spam and junk filters, as well as attachment filters like those provided by SpamTitan or Barracuda (I prefer SpanTitan).

Programs like DMARC do a good job with email security and allow you to set up authentication and policies for measures servers can take in times of trouble.

Host-based firewalls can be effective in preventing suspicious files. Some even provide “sandboxing” which will isolate and analyze those files. All of these systems will be integrated into SIEM and our professionals will use the aggregate data for detection.

We're Here Now (Map 2)

Using the *Separation of Duties* (Principle 3), we could prevent Homer and his entire department from using terminals like Powershell. That could make it difficult for Marcos to do damage; however, there are ways around it. Let's assume that's not the case, and an employee opens that malicious attachment with Powershell intact. There are several points of security that now come into play, starting with anti-virus. Anti-virus software scans for malware by using known malware signatures and analyzing behaviors.

Furthermore, there are software solutions that can help with the detection of the establishment of command and control C2 channels. Effective programs include Microsoft Defender for Endpoint or Threat Feed Services, the latter of which can track various Command and Control frameworks. Analysts should also be auditing. While automated systems are effective, you cannot write a script for something you haven't seen before. Many attacks get through by being the "Black Swan".

Privileges (Map 3)

At this point, the attacker has established command and control, and is escalating Homer's privileges to the admin level.

The defense against this starts with Principle 2, *Principle of Least Privilege*. Homer shouldn't have admin privileges to begin with, as he is **not** an administrator. Also the administrator level, as well as all others, should have multi-factor authentication, making it even more complicated for an attacker to enter. All passwords should be long and strong, and privileges should be audited and updated. If an employee gets promoted to a different position, his privileges should be checked. Different departments should have **different** privileges. One should couple all of these methods with regular patches and updates to keep the system strong.

That wasn't the case here, and our attacker gained access to admin privileges. He used those to empower Mimikatz, a powerful program that requires the admin level to use its commands. He used MimiKatz to grab a "Shared local administrator password".

This is exactly why these shared passwords are frowned upon. With proper security, the attacker would never have access to this. ACLs on firewalls and routers can also prevent traffic from different VLANS from accessing one another, and can even specify certain protocols from being used like Telnet or SSH. Homer's computer and Smither's computer are in two separate networks, and network security can help here, however it can ALSO impact legitimate traffic. Mimikatz itself isn't undetectable, however it has been rewritten in both powershell and GoLang, so finding it can be troublesome, especially if you don't even know you're being attacked. The average company doesn't know it's been attacked until a whopping 200 days after the attack has happened.

UNPROTECTED (Map 4)

Once on Smither's computer, our attacker found his **unprotected** SSH key. This particular SSH key grants access to a jump box that's connected to a SCADA systems network, which is notoriously vulnerable for several reasons, ranging from lack of anti-virus to weak authentication. SSH keys should be protected with multi-factor authentication and they should be regularly changed.

Using the key, the attacker made it to the jump box. Then he used Nmap to find open ports. This could be prevented if the jump box didn't have access to Nmap. Jump boxes are simple and don't require many tools. For security purposes, only having what's **needed** is the best practice for jump boxes (among other things).

The attacker scans for TCP/666, finds it, and connects to it via Telnet. Telnet, from a security perspective, has been outdated for decades. You should not be able to Telnet your way into a Nuclear Reactor. SSH should be used, along with keys and passphrases.

Prevention is the Best Medicine (Map 4)

Lastly, the attacker placed malware on the reactor that will act in 30 days. SCADA systems should be constantly checking for malware and constantly audited by administrators. The truth is that the attack should never have gotten this far. Security is never **one** thing, it’s Defense in Depth (Principle 1). Security includes PREVENTION, DETECTION and RESPONSE. You monitor data with SIEM, analysts investigate, network security and firewalls limit the possibilities of what can enter your space. You must educate your employees on what to look for as well as limit their capacity to do harm (both purposefully and accidentally).

Like I stated earlier, it takes 200 days to know you’ve been attacked, and another 70 days to fully eliminate the threat. Like the doctor says, prevention is the best medicine.