SECURE SOLUTIONS

MITIGATING CYBERSECURITY RISKS IN DEVELOPMENT ENVIRONMENTS

White Paper

By Sara Spencer

WHITE PAPER

MITIGATING CYBERSECURITY RISKS IN DEVELOPMENT ENVIRONMENTS

Executive Summary

IT professionals and development business units have traditionally not worked very well together. This is because the IT professional is concerned with protecting the enterprise while the developer is pressured to develop meaningful products and needs more flexibility within the systems and environments which they work. Historically because the development teams ultimately produce revenue for the company and are under strict deadlines, they are put into self-managed labs, with the unspoken message to IT, “hands-off”. This builds a culture over time that there is no need to worry about the development teams’ infrastructure because they are in labs. A culture where IT does not want to be responsible for managing development environments because the relationship can become contentious.

Traditional IT configures compute devices to prevent end-users from running arbitrary code, packet capturing tools, and debugging tools. Often these tools are necessary for the software development teams in order for them to do their job effectively. These tools could result in a broad attack surface to would-be attackers.

The other issue between traditional IT and development teams, is the complexity of the systems that development teams are running. Systems that are not easily understood by the IT team. Therefore, patch management and device hardening becomes impossible from the IT teams perspective “Unfortunately, even the occurrence of seemingly routine events like load changes, node and link failures, software patches, or modifications of certain environmental parameters can cause such systems to behave in unexpected ways, often resulting in their failure to meet current objectives.” (Kumar, Cooper, Eisenhauer & Schwan, n.d.)

To close this gap there needs to be a flexible solution that is secure. Developers are intelligent and will find ways to work around hinderances to get their job done and this could potentially introduce more risk. Therefore, we need enterprise awareness about development and must reduce the complexity of security. Empowering development teams to provide self-managed security solutions that are simple to understand and implement. Most people will do the right thing, if they know what the right thing is. If security “folks” constantly tell the developer no, no, no you did it wrong citing results of a penetration test, vulnerability scan or some other action without telling the developer how they can do it right is not effective. “Matter of fact from my personal experience working in this domain and several years of managing people, I can tell you it will have the opposite effect” Sara Spencer.

This white paper will discuss ways to provide flexibility to development teams while securing and protecting propriety data, managed environments, and the issues with these techniques, and ways to effectively monitor the self-managed environments.

INTRODUCTION

Threats are becoming more sophisticated in today's world and have the ability to pivot, no longer contained in the adjacent network but exploiting one or more network hops away. "Malware often infects servers and systems without the knowledge of system users and owners. If the malware can establish persistence, it could move laterally through a victim's network and any connected networks to infect nodes beyond those identified in this alert." (CERT- TA18-149A, 2018) Therefore, culture in the enterprise needs to pivot and become security conscious in everything they do, realizing it is no longer an option. Security should begin at the "requirements definition" stage in development. Implementing security from inception, baking it in from the beginning and not spraying it on as an afterthought. The bad guys only have to get it right once, but the enterprise needs to get it right every time.

That means the first thing a development team should do is make sure their internal infrastructure and systems are as secure as possible. I often hear developers say, IT has implemented a secure backbone and I am in a lab isolated from the rest of the production environment, why do I need to worry this much about system and internal infrastructure security. Many years ago, that may have been a truer statement. The fact of the matter is it is no longer even a little true. Threats are becoming more capable and numerous. Also, with the cloud becoming more readily adopted our attack surface is growing at a faster rate than we can keep up with. Recently the VPNFilter attack proves that a hard outside with a squishy inside is ineffective at mitigating persistent threats. VPNFilter attacks routers and is capable of contacting a control and command (C&C) server to download further modules. Its payload is capable of file collection, command execution, and device management to name a few. It can snoop traffic and delete any traces of itself before bricking the device. (Symantec, 2018)

The hygiene and security posture of self-managed development spaces in regard to weak passwords, lack of patch management, system hardening, internal segmentation and deployment of anti-malware create an environment which is "high risk". Potentially affecting the confidentiality and integrity of not only some enterprises production environments but their proprietary intellectual data. Organizations that manufacture, and develop valuable, proprietary trade secrets and technologies must recognize that they are at risk from advanced persistent threats.

Research I conducted of approximately 300 development environments and that is out of the scope for this white paper reveals that legacy, unpatched systems, some containing vulnerable applications are interconnected with current technology environments such as on prem production, the cloud and the internet. This evidence further solidifies the fact that these development systems and environments are especially vulnerable. It also reveals that the integrity and confidentiality of an enterprises source code is at risk. Threats can enter into the internal network through a vulnerable device which is connected to either the external network or other infected hosts on the same network. This not only affects the confidentiality of proprietary information but the integrity of it, introducing their very own, "zero day".

The key to protecting critical enterprise environments including development spaces is to have a comprehensive security in depth plan and program by using a layered approach. That way if one of the layers fail, there are other layers that can catch a threat if one were to occur. "We must accept the fact that no barrier is impenetrable." (Verizon, 2015)

Some ways to create security in depth-

- Physical and Logical Segmentation
- Device Hardening
- Hardened images for deploying systems
- Anti-Virus/Anti-Malware
- Vulnerability Management
- Secure Coding
- Intrusion Prevention
- Network Monitoring
- Authentication, Authorization, Auditing
- Vulnerability Scanning/Penetration Testing
- Inventory
- Policies, Metrics, Standards, Processes
- User Education and Awareness

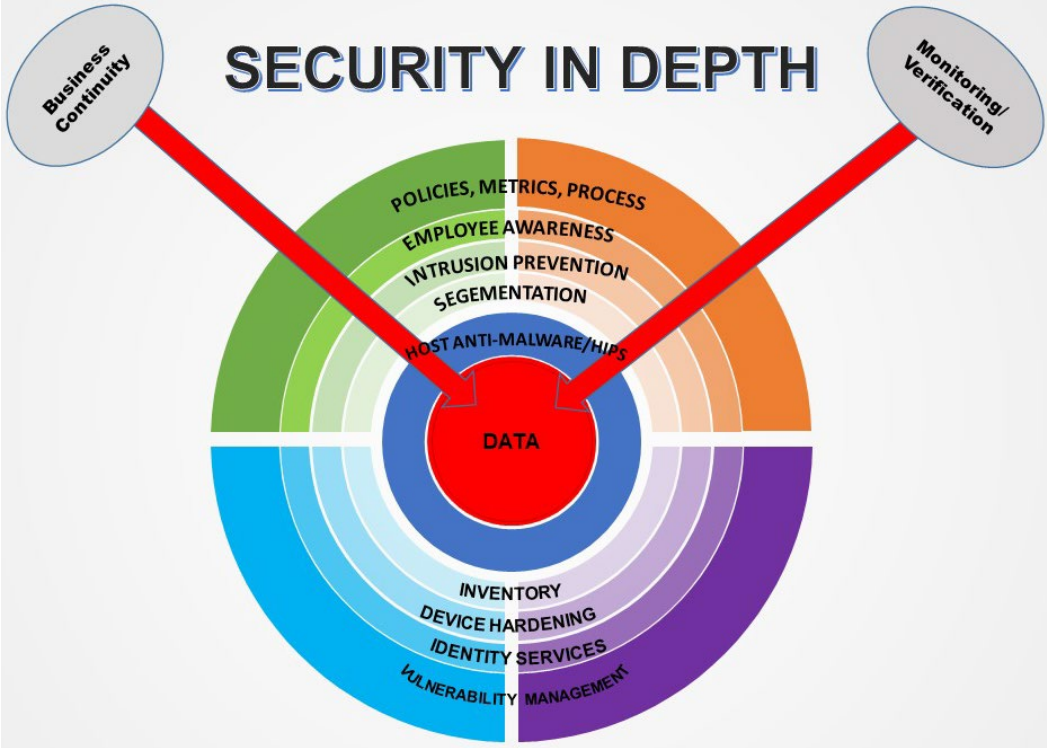


Figure 1.

This white paper will discuss ways to provide flexibility to development teams while securing and protecting propriety data, managed environments, and the issues with these techniques, and ways to effectively monitor the self-managed environments.

PREVIOUS APPROACHES

In the past, risk was determined based on the criticality of the legacy and vulnerable systems. Whether the system would be remediated or mitigated depended on the risk and cost to replace the system. If the cost was more than the risk, then organizations usually decide not to remediate or mitigate the vulnerability. That approach is no longer applicable, since threats are becoming more sophisticated and are able to escalate privileges, as well as, pivot and in many cases are no longer contained in the adjacent network but have the ability to pivot one or more network hops away.

The above mentality does not reduce the risk of an attacker. Developing software and embedded systems in environments and on systems that are not secure introduce the following potential threats.

- Stealing sensitive information (such as encryption and access keys, passwords, knowledge of security controls or intellectual property).
- Embedding malicious code in a developer's project without their knowledge.
- Using a compromised development device serves as a proxy to further attack in the build and deployment pipeline, through to production or your worse your customers.
- Understanding how your sensitive applications work is the first step in an attacker's reconnaissance in planning their attack.

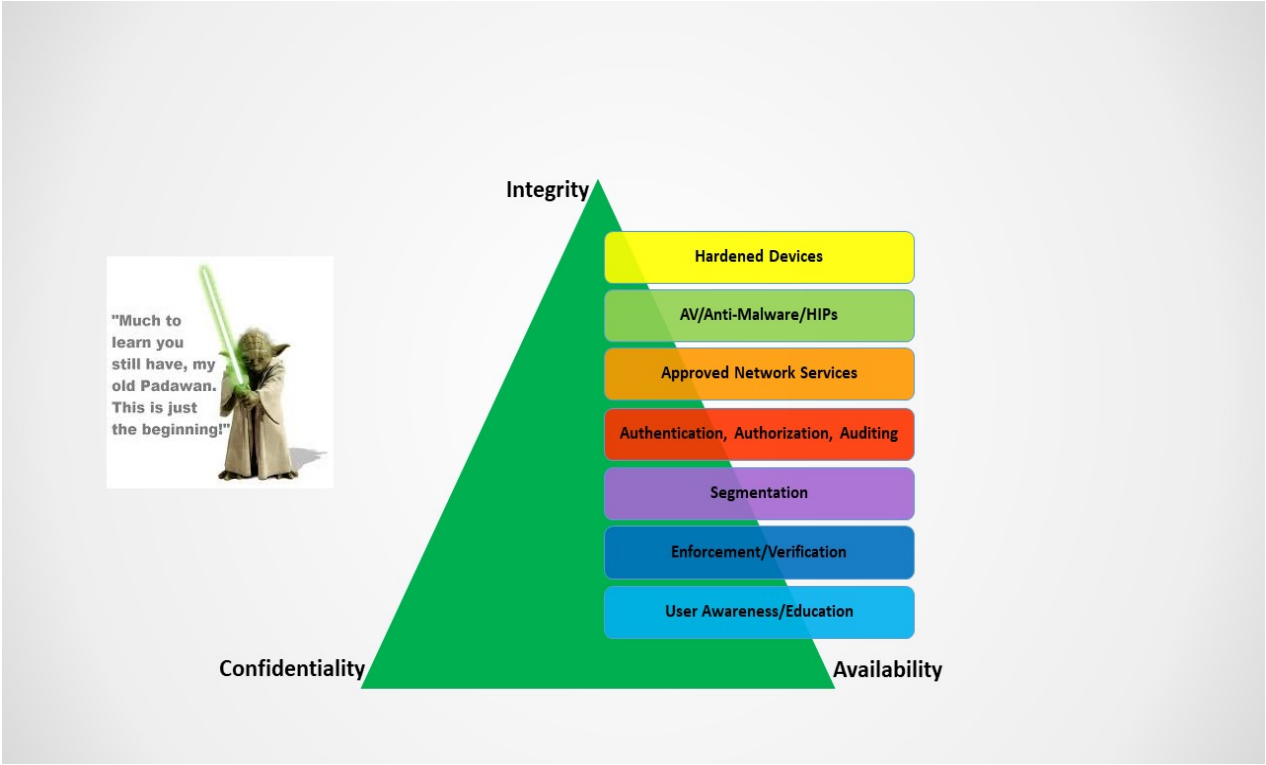
Furthermore, developers are not cybersecurity professionals nor are they experts in infrastructure management. They are concerned with doing their job effectively. They often need administration or root level permissions in their working environment. This type of access can lead to huge impacts if malware compromises their device. This can happen by spear-phishing, drive-by downloads, or accidental malware installation.

This culture of a lack of security knowledge in development teams has been driven by decades of management not fully understanding the why it is important as pointed out by Carol Woody in an SEI Podcast produced by Carnegie Mellon University Software Engineering Institute in the following quote, "I am continually told that we just have to get the systems built. We need all that functionality. It has to be cheap, and it has to be fast. It has got to replace people, and it has got to get things done quickly. We do not have time for the security stuff because that takes extra effort and costs extra. Besides, we do not have the people that understand how to do that."

Figure 2. Flickr, Pixabay, (n.d.) *Time to Rethink*

NEW FINDINGS

Stakeholders depend on the various business units to protect the enterprises source code, customer and intellectual data. Our customers also depend on the enterprise to provide secure products. One of the most important findings I discover while working directly with enterprise level development business units, is that developer awareness, education and providing simple solutions has gone a long way in strengthening the security posture in the development space.



Network Segmentation-

Core business services such as mail, Active Directory, Document Management, DNS, DHCP and other Tier 1 servers that contain critical information should be blocked by a firewall or routing, this makes pivoting more difficult for an attacker. Separation can also be achieved by running virtual machines on hardware that meets the proper security posture, creating another lay of security. The implementation of cross connects between development DMZs and production should never, and I repeat never be a solution.

If an attacker is able to compromise a developer's system, they can potentially inherit the same levels of permissions as the developer. Separating developing environments from critical infrastructure can mitigate the impact of a security breach on a developer's system. Another method that can mitigate damage is requiring multi-factor authentication on critical production systems, as well as, secure admin controls, such as accessing management interfaces from a secure subnet, single IP address and secure system like a jump host.

Educate and Trust your Developers-

An educated developer will be able to detect and prevent an attack more efficiently than the technical controls themselves.

Provide Developers with Tools-

Providing Operating System (OS) images that are hardened with a current version of Layer 7 applications like SSH, SSL, and turning off unnecessary services like SMB, and closing unnecessary ports won't introduce common vulnerabilities into the development environment when they spin up devices or VMs. These images could have agents like OSQuery for monitoring, as well as anti-malware. This is a number one request I get from development labs. Using demo images and unhardened images introduces thousands of vulnerabilities when the devices are brought up for the first time and are rarely ever resolved.

Vulnerability Management-

Scan development labs to make sure there are no critical vulnerabilities and to verify the controls are being implemented. When a vulnerability is found make sure the development team understands the risk to their project and help them in resolving the issue if the developer needs technical guidance. Too often I see the security teams hand over a long list of vulnerabilities, and when the team responsible for fixing the problem technically, the security folks have no systems experience and are unable to help the teams resolve the issues. This results in contention.

Least Privilege-

If root access to a system is necessary, make sure developers understand they should run in least privilege mode when escalated privileges are not needed. This should not be an option. Privileged accounts should be monitored to see what accounts are being used to access what applications.

CONCLUSION

"EVERY DAY SEEMS to bring news of another dramatic and high-profile security incident, whether it is the discovery of longstanding vulnerabilities in widely used software such as OpenSSL or Bash, or celebrity photographs stolen and publicized. There seems to be an infinite supply of zero-day vulnerabilities and powerful state-sponsored attackers." (Sampemane, 2015)

In order to implement a defense-in-depth strategy there may have to be a cultural shift that occurs. This culture shift will come with education and continued practice. Introducing security before the project begins and during the System Development Life Cycle is more effective than finding out later you have wasted several hundred thousand dollars on the development of a product that will need to be trashed. There is nothing worse than having your customers, who are becoming more savvy find these vulnerabilities for you.

Trust that if your development team knows what to do, they will do it. Monitoring and verification is effective in verifying that the controls are actually being implemented by the various teams.

Developers are not thinking like CyberSecurity experts or even IT admins, it is important to train them. Tell them why and what, but more importantly give them the resources and knowledge to do the how.

References

(Verizon, 2015). *Data Breach Investigations Report (DBIR)* Retrieved from

https://www.verizonenterprise.com/resources/reports/rp_data-breach-investigation-report_2015_en_xg.pdf

(Kumar, V., Cooper, B.f., Eisenhaur, G., Schwan, K., n.d.). *iManage: Policy-Driven Self-management for Enterprise Scale Systems* Retrieved from https://link.springer.com/content/pdf/10.1007%2F978-3-540-76778-7_15.pdf

(CERT- TA18-149A, 2018). *Hidden Cobra – Joanap Backdoor Trojan and Brambul Server Message Block Work* Retrieved from https://www.us-cert.gov/ncas/alerts/TA18-149A

(Symantec, 2018) *New Router Malware with Destructive Capabilities* Retrieved from https://www.symantec.com/blogs/threat-intelligence/vpnfilter-iot-malware

(Flickr, Pixabay, n.d.) *Time to Rethink* Retrieved from

https://www.teskalabs.com/blog/snap-to-it-mobile-secure-gateway-is-your-future

(Sampemane, 2015) *Internal Access Controls*, Communications of the ACM, January 2015, Vol. 58 No. 1, Pages 62-65 Retrieved from http://web.a.ebscohost.com.ezproxy.umuc.edu/ehost/pdfviewer/pdfviewer?vid=5&sid=9767d1cd-5d8e-4c65-9cf9-ca9f3de59116%40sessionmgr4009