AGIL™

The Key to Secured Operating Systems On-Premise and Cloud: Polymorphing

ST Engineering

Table of Contents

Executive Summary

With an increasing proliferation of cyberattacks that undermine the security of critical operating systems, the need for advanced and innovative security solutions that prevent zero-day attacks is heightened.

To reinforce critical platforms with an additional sphere of security that is unique to each operating system, across the spectrum of hardware, software and operating system layers in both on-premise and cloud environments, ST Engineering works closely with Polyverse Corporation to provide leading-edge Polymorphing technology to global government and commercial customers.

Polymorphing is a game changer in the evolving cloud and Internet of Things (IoT)/edge computing landscape. Not only does it help to mitigate the problem of delayed patching of memory exploits and prevent zero-day attacks before they can even start moving to spread across line of business work areas, it can also be embedded in operating system-dependent devices such as firewall appliances, network devices, IoT/edge computing devices and sensor platforms to effectively protect against hackers.

The world runs on Linux. With 90% of servers running Linux on-premise and in the cloud, it has become the backbone of business infrastructure. Many critical systems such as SAP HANA and Oracle Database were developed to run on Linux due to its stability and security. Unfortunately, as the use of open source software has increased, hacking these platforms and operating systems has also increased. Of all successful cyberattacks, 80% leverage coding flaws such as buffer overflows, stack clashes and other malformed input validation logic and are referred to as memory-based attacks. According to MITRE's CVE database,[1] Linux and its variants have surpassed Windows as the operating systems (OS) with the most vulnerabilities.

This puts every organisation running Linux at risk.

The Open Web Application Security Project (OWASP) provides the following guidance to protect against buffer overflow vulnerabilities: "Keep up with the latest bug reports for your web and application server products and other products in your Internet infrastructure. Apply the latest patches to these products. Periodically scan your web site with one or more of the commonly available scanners that look for buffer overflow flaws in your server products and your custom web applications."[2]

Unfortunately, the approach of trying to keep up with the latest bug reports and rushing to patch vulnerable systems before attackers infiltrate them is a losing proposition.

The Patch Gap

All software contains bugs. These bugs can and will be exploited. Today, these cyber exploits work across all instances of the same operating system or application. The objective of a software or security patch is to fix bugs, in particular known exploits. However, software patches can themselves contain bugs, which can be exploited. Further, there is a delay between the identification of a cyber exploit and the patching of the cyber exploit.

A well-known example of an organisation that missed applying a needed security patch is Equifax. In this particular case, Equifax only scanned for vulnerabilities from a list of servers, instead of scanning all servers within their organisation. The server that was exploited was not on that list and, therefore, it was not scanned and patched. This resulted in massive damage to the company and left consumers vulnerable to identity theft. The breach ultimately cost Equifax more than $1.4 billion in damages, and loss of customer business since its disclosure in 2017.[3]

It only took a single unpatched server to breach the entire Equifax network. Organisations face great difficulty in tracking servers deployed in their networks. Deploying new servers, and decommissioning old servers, happens on a regular basis. Endpoint management is often a nightmare, especially if an employee deploys a server without following proper policy guidelines. Even a newly deployed server that is exposed externally for only a few minutes can be enough time for a hacker to exploit and compromise the network.

Even when an IT organisation manages to keep a perfect record of all of its deployed servers, there is still the problem of keeping up with patching those servers. 60% of breaches in 2019 were linked to a vulnerability where a patch was available, but not applied.[4] There is a long list of valid reasons why patching is logistically difficult to manage and apply even for the largest and most modern enterprises. This includes lack of staff, waiting for a window of time to take critical applications offline, prioritising which servers to patch first, and draining network traffic before the server can be taken out of rotation for the patch to be applied. In order to succeed, patching must be carefully coordinated to minimise the overall business impact. The disruption from taking down servers to apply patches to a company's business is a real and measurable cost.

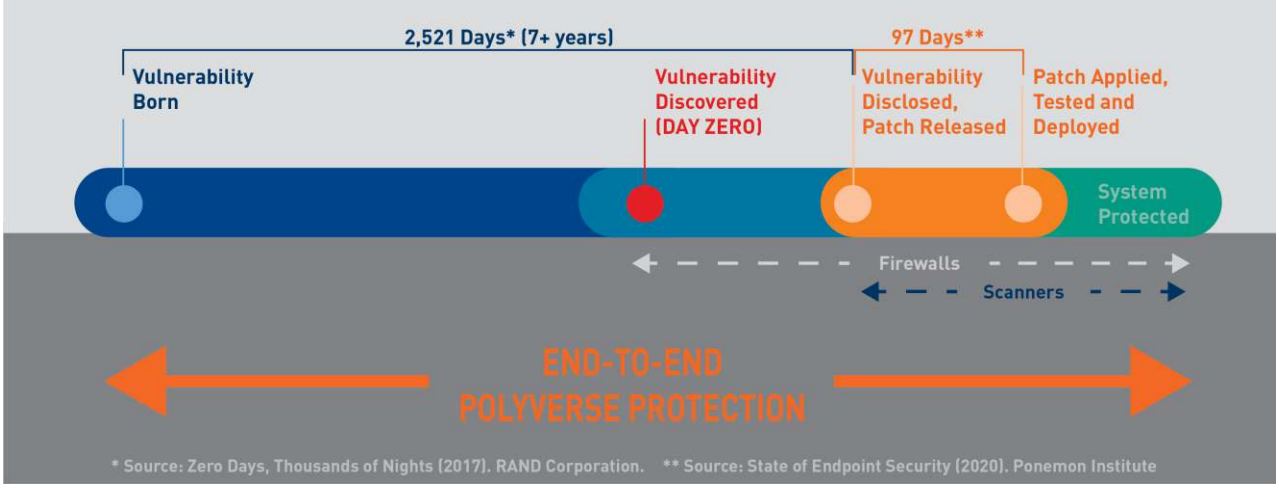

Figure 1: The Patch Gap

According to the Ponemon Institute, it takes an average of 102 days to patch a vulnerable system.[5] This is a wide window of opportunity for a hacker to exploit any unpatched servers that are left vulnerable. In some cases, it only takes 20 minutes for an attacker to exploit an unpatched system. Hackers are constantly scanning the Internet to target vulnerable servers with known exploits. What about attacks from unknown exploits using vulnerabilities that have not yet been announced and reported to the vendors whose software we rely on? Organisations are left unaware, not realising their systems have been compromised for days, months, or even years. This is known as the Patch Gap (see Figure 1).

Polymorphing

So how can you close the patch gap? The only way to close the patch gap, is to do away with the need for patching as being the primary defense against the attacker. Polymorphing™ for Linux is a ground-breaking technology that was previously only an academic theory and is the only technology capable of closing the gap. Hackers rely on every instance of an OS being the same. For example, if you have Red Hat Enterprise Linux 7, it will be the version of Red Hat Enterprise Linux used everywhere else – even by the hacker. Therefore, if a hacker manages to develop an exploit from one of the vulnerabilities found in Enterprise Linux 7 (which currently has 762,978 weaknesses), they can use it to exploit systems and applications that run on that same OS elsewhere.

The only way to prevent a hacker from utilising and then scaling a single attack to millions of systems is to make every operating system instance unique. Polymorphing takes this approach by scrambling the instruction set at the binary level, making each Linux instance unique.

This approach takes a cue from nature's innate diversity. If every human was a clone, the first fatal disease that

comes along would wipe out the entire human race. Yet thanks to genetic diversity, a highly contagious disease such as COVID-19 that is deadly to one individual may only ail another with a fever. Much like humanity's inherent diversity, a resilient software system must have binary diversity to prevent an attack that previously worked on one system from successfully working on another. This technique is called "moving target defense".

Polymorphing is a ground-breaking technology that hardens open source Linux distributions. The source code is run through a polymorphic compiler to effectively scramble the low-level machine code, without affecting the original source code, interoperability, or functionality. By substituting the OSs original instruction sets with unique, equivalent instruction sets for each installation, a hacker cannot rely on prior knowledge to craft a memory-based attack. This creates an instance with completely randomised and unique resource mapping, making it exceptionally difficult to reverse engineer. Each diverse Linux instance is effectively immunised against the entire category of memory-based attacks including everything from code-execution, buffer overflow, and memory corruption attacks.

This means that crafted exploits targeting a buffer overrun vulnerability will not work, even when the application is left unpatched. Polymorphing reduces the pressure to urgently patch all systems as quickly as possible, freeing up your IT organisation to patch in a controlled cadence and focus on more high-value projects.

For environments with continuous deployment pipelines (CI/CD), new Linux containers can be generated and deployed with a new, uniquely scrambled OS every 24 hours. Even if a hacker is able to reverse-engineer a Polymorphed system's scrambled instruction set, and craft an attack specifically for that Linux system, they would only have a 24-hour window to successful exploit it. Once the Linux system is updated with new scrambled binaries, the hacker would need to start over to discover the layout of the new instruction set. These are high hurdles to overcome for compromising a single target. Due to this shift in effort where the defender is now favoured, most attackers move on to easier targets.

Polymorphing your instance of Linux does not change the source code of your operating system, only the binary layout of the compiled packages. This does not affect the functionality, debuggability, performance, or interoperability of your Linux instance. Since there's no shim in between the application layer and operating system, Polymorphing also does not incur any runtime overhead. Polymorphing can be applied to a Linux installation by pointing its package repository to Polyverse's® Polymorphic repositories, instead of a community-run mirror, and updating its packages. The system will download uniquely scrambled binaries from Polyverse. Every time the package manager is run to pull down new updates, a different set of scrambled binaries are downloaded. Polyverse scrambles more than 70,000 of the top open-source packages, including Java, Ruby, and PHP. It currently supports Red Hat, Ubuntu, CentOS SUSE Linux Enterprise Server, and Alpine Linux in public clouds and private data-center deployments, as well as ARM and Intel architectures for embedded devices.

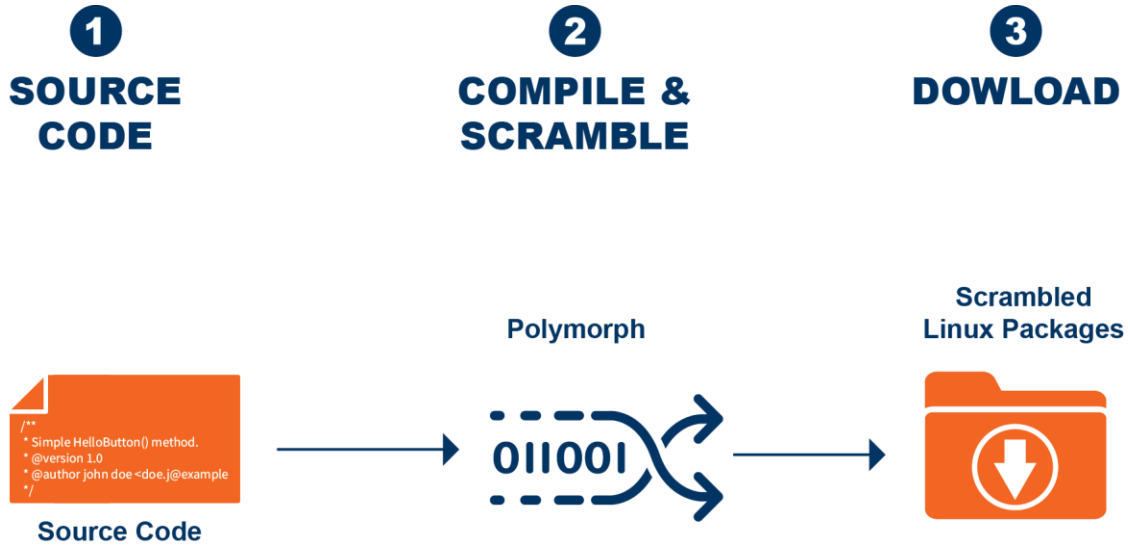

Figure 2: Diagram of Polymorphing

Mirror

Polymorphing for Linux is licensed per instance or can be licensed for a distribution to best suit your infrastructure needs. A Polyverse Mirror is a deployment model that allows an organisation to license an unlimited number of Polymorphing for Linux instances of a specific distribution (e.g. Red Hat, CentOS, Alpine). Instead of licensing a fixed number of Polymorphing instances for a specific version, a Mirror provides a simpler, more cost-effective licensing model that saves an organisation from
having to track and manage their instances.

This licensing model is beneficial to organisations that spawn and recycle many Linux instances (i.e. Linux VMs and containers) or have a large Linux installation.

Conclusion

Today, all versions of a given operating system are the same, making it possible for a hacker to easily exploit the vulnerabilities commonly found in that operating system. The most successful form of cyberattacks are memory-based and in today's world the only real way to defend against them is via patching. Unfortunately, patching is fraught with problems and is the least sure way to protect your critical data, applications and systems. The only way to truly protect against such attacks is by doing away with the reliance on patching for security, by making

each of your Linux operating system unique via Polymorphing. Polymorphing for Linux configures your existing system with uniquely scrambled binaries without affecting the functionality of the system or its performance. Scrambled binaries can be updated every 24 hours, resetting attackers from trying to map your system's unique instruction set. This means that the hackers can no longer exploit the vulnerabilities in your Linux OS and you can now patch for hygiene, not for security, stopping attackers before they start.

This paper is jointly produced with POLYVERSE

ST Engineering is in partnership with Polyverse Corporation to provide leading-edge, secure operating systems solutions for critical platforms via AGIL Cloud Management Platform and other suite of solutions

Contact Us

Email: polyverse@stengg.com

Website: https://www.stengg.com/cloud

References

[1] https://www.cvedetails.com/top-50-products.php

[2] https://owasp.org/www-community/vulnerabilities/Buffer_Overflow

[3] https://blog.bitdiscovery.com/2020/03/equifaxs-lack-of-asset-management-wasthe-cause-of-their-breach , https://securityboulevard.com/2019/10/60-of-breaches-in-2019-involved-unpatched-vulnerabilities

[4] https://www.helpnetsecurity.com/2019/10/30/unapplied-security-patches

[5] State of Endpoint Security, 2018, Ponemon Institute