Moving Target Defense

Redefining the Power of Defense

POLYVERSE

Moving Target Defense

Redefining the Power of Defense

What is Moving Target Defense?

When it comes to programming, there is one fundamental and critical truth: every piece of software is hackable. Given enough time and resources, a vulnerability will be found, and an exploit crafted. Adversaries need only succeed once, while defenders must succeed continuously. Once a vulnerability is found and exploited, it can be run on millions of computers because systems are homogeneous. The effort-to-reward ratio is squarely in the hackers' favor. Today's traditional defenses, such as anti-virus, firewalls and even predictive security analytics are powerless against attacks, particularly when they have never been seen before.

Moving Target Defense (MTD) is different. MTD offers a solution to the cybersecurity problem that draws its inspiration from nature. Much like the human body's natural resiliency, a resilient system is one where vulnerabilities are assumed, and which has built-in defenses designed so that the system can continue to operate safely and reliably.

Genetic diversity is both a key to and a result of the survival and evolution of organisms. If every human was a clone, the first fatal disease that came along would affect each individual in the same way, wiping out the entire human race. Think of a malicious hack like a disease. It needs to interact with the host's defenses in a specific, replicable way to spread effectively. Yet fatal diseases do not wipe out the entire human race: thanks to genetic diversity, a disease that is deadly to one individual may not ail another with so much as a fever. MTD is the practical application to technology of nature's genetic diversity.

Moving Target Defense (MTD) is "the concept of controlling change across multiple system dimensions in order to increase uncertainty and apparent complexity for attackers, reduce their window of opportunity and increase the costs of their probing and attack efforts."[1] By introducing diversity into the system, we can limit the exposure to vulnerabilities and opportunities for attack, thereby increasing complexity and cost for attackers.[2]

Many believe that MTD is the only long-term viable technique to defend against the most insidious cyberattacks and Advanced Persistent Threats. Jason Yorty, formerly of the US Department of Defense's special-access program, even went so far as to say that "as an attacker, going after an application that has been transformed with Moving Target Defense is a massively disheartening effort."

Memory-exploiting zero-days

Some of the most insidious cyberattacks are "zero-day" attacks. What is so destructive about a zero-day attack is that it uses a vulnerability that is largely unknown and for which there is not yet a patch.

For example, the infamous Heartbleed vulnerability in 2014 was a zero-day in the OpenSSL protocol. Attackers exploited Heartbleed by sending a malformed heartbeat request over the Internet. The 2017 Microsoft .NET-framework vulnerability (CVE-2017-8759), which enables attackers to "take complete control of an affected system," is yet another example of a zero-day

1 Official definition from the Department of Homeland Security.

2 Trustworthy Cyberspace: Strategic Plan for the Federal Cybersecurity Research and Development Program published by the Executive Office of the President, National Science and Technology Council, December 2011.

vulnerability that was heavily exploited.

By definition, no existing signatures or firewall rules can catch a zero-day attack. Furthermore, a successful zero-day attack is often just the initial injection of an Advanced Persistent Threat, in which an attacker lurks undetected in the infected system for as long as he or she wishes, with free rein to compromise it at will.

Modern zero-days often leverage memory-exploiting techniques such as Return Oriented Programming (ROP)[3] or Jump Oriented Programming (JOP). ROP and JOP enable a remote attacker to hijack the control flow of the target program by smashing the call stack and using the hijacked program's own "gadgets" to execute specific instruction sequences.

Because ROP lets hackers bypass built-in defenses such as data-execution prevention, it has become one of the most popular techniques in software exploits. As such, by preventing ROP, we can eliminate nearly the entire class of memory-based software exploits.

Prior to Polyverse, the state-of-the-art defense against ROP/JOP attacks was address-space-layout randomization (ASLR), a technique found in almost all modern operating systems. Even though ASLR randomizes the base address of sections, other aspects of the memory structure are not randomized. Thus it is surprisingly easy for attackers to work around ASLR. By compromising just one number (e.g., a single memory address), it is possible to completely defeat ASLR defenses.

Polyverse's Moving Target Defense Technology

What if computer programs shared humanity's quality of having their own unique genetic makeup? Polyverse's Polymorphic Linux harnesses this Moving Target Defense strategy to create high levels of entropy in a software system in such a way that the entire memory structure is diversified.

Today, attackers assume that the gadgets they need are located in a certain address or with a specific offset from the absolute base address. This in-depth system knowledge is possible because the attacker has access to exactly the same code as everyone else, and has all the time in the world to craft a specific attack for the memory layout of that specific distribution (see Figure 1).

3 https://en.wikipedia.org/wiki/Return-oriented_programming

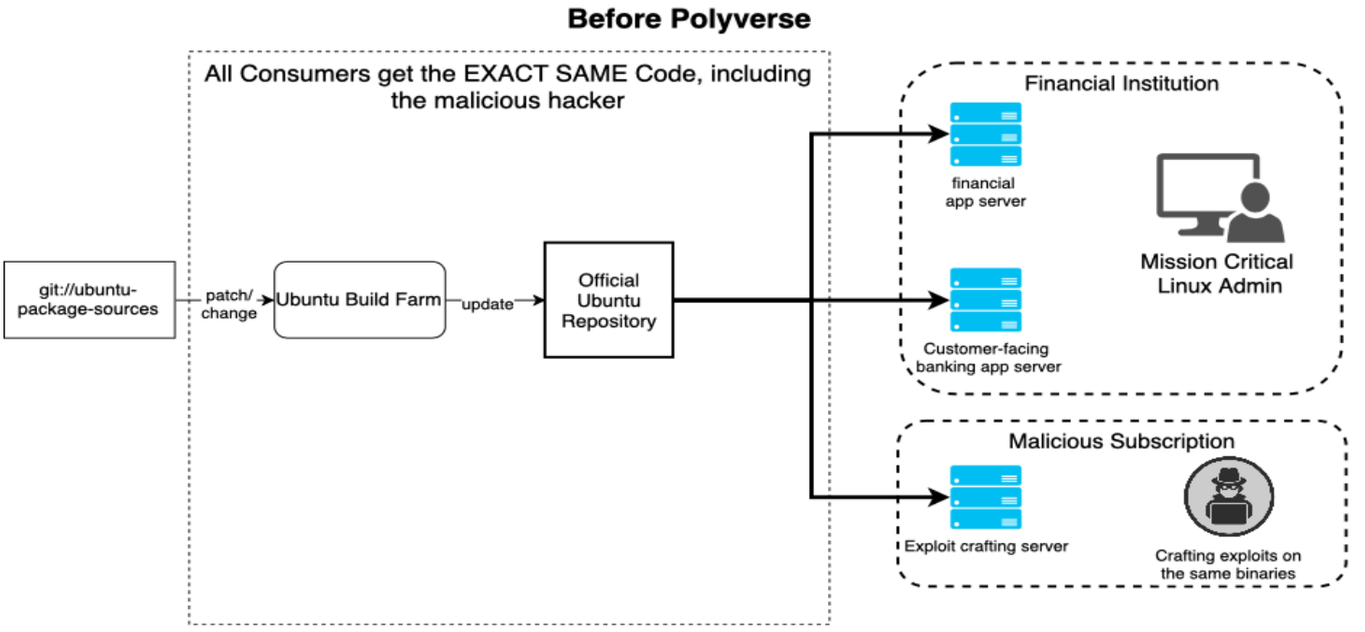


Figure 1: Open-source package distribution architecture. Attackers get exactly the same libraries as those that need to run mission-critical applications.

With Polymorphic Linux, the memory layout of the entire Linux software stack is thoroughly randomized. The resulting program is semantically identical to the original—they function exactly the same way and operate with a similar performance. However, nearly every machine instruction is changed. Entropy is introduced throughout so that no ROP attack can anticipate the random structure, and thus fails to work.

Polymorphic Linux produces a uniquely randomized set of binaries for the full Linux stack, from the kernel to libc to programming-language frameworks to middleware components such as Apache—and can be updated every 24 hours. Even if an attacker was able to compromise an instance of Polymorphic Linux through some other means, such as physically gaining control of the computer, they will have to do the work all over again and create unique exploits tuned for each instance of the software every day.

By default, Polymorphic Linux is delivered through standard package repositories for Linux. Thus, you deploy and manage Polymorphic Linux in the same way as you would any standard Linux. In addition, with Polymorphic Linux installed, whenever a patch or update is implemented you are simultaneously provision with a unique set of binaries. This continuous diversity entirely defeats memory-exploiting attacks, because the attacker will never have *a priori* knowledge of your system's configuration (see Figure 2).

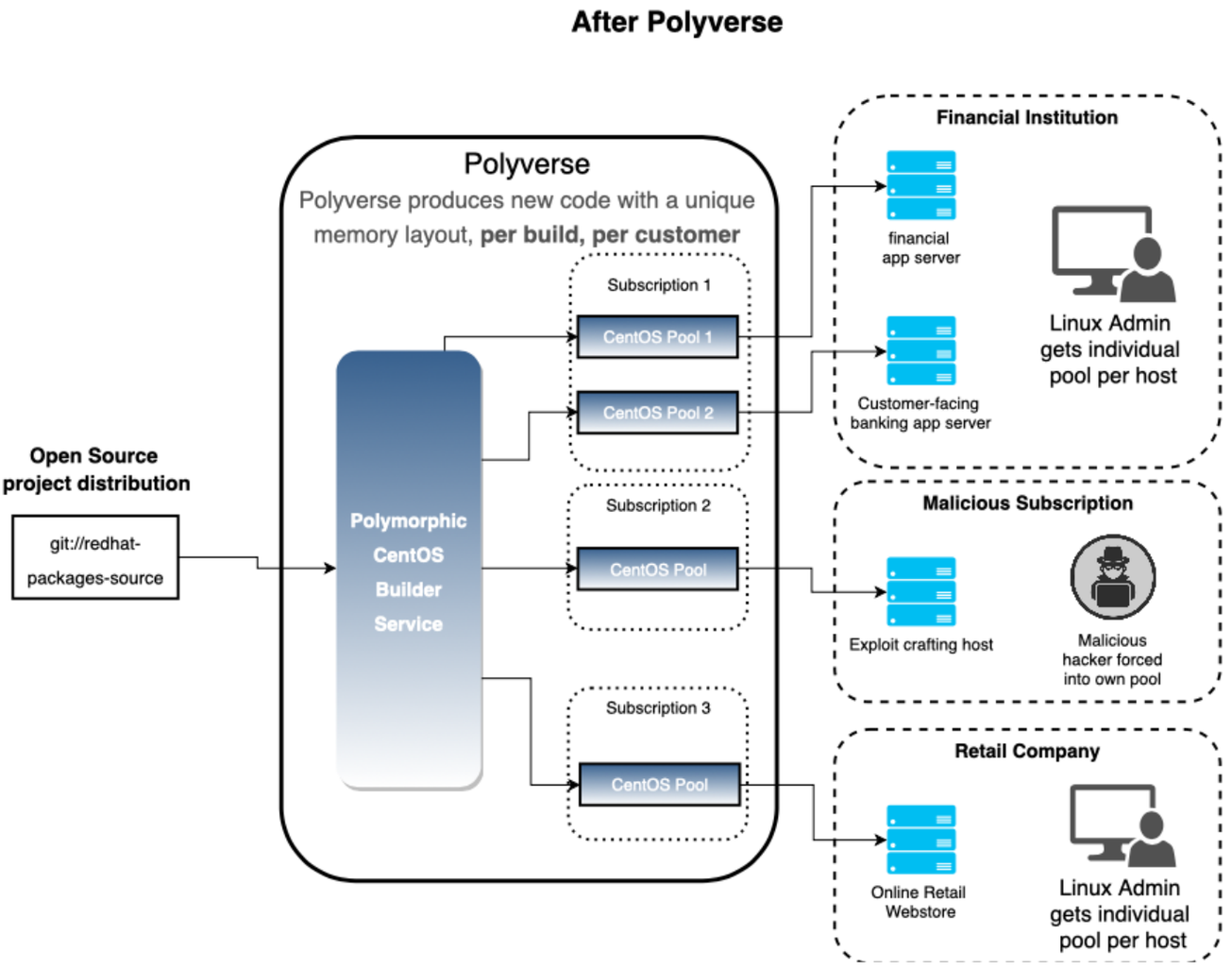


Figure 2: With Polyverse, every subscription and every host gets its own pool of unique repos at every build. You can build as many times as desired.

How does it work?

In the software compilation process, the compiler makes many predictable choices. If an attacker already knows how the compiler makes those choices, they have the necessary level of knowledge that could lead to a successful memory-exploiting attack.

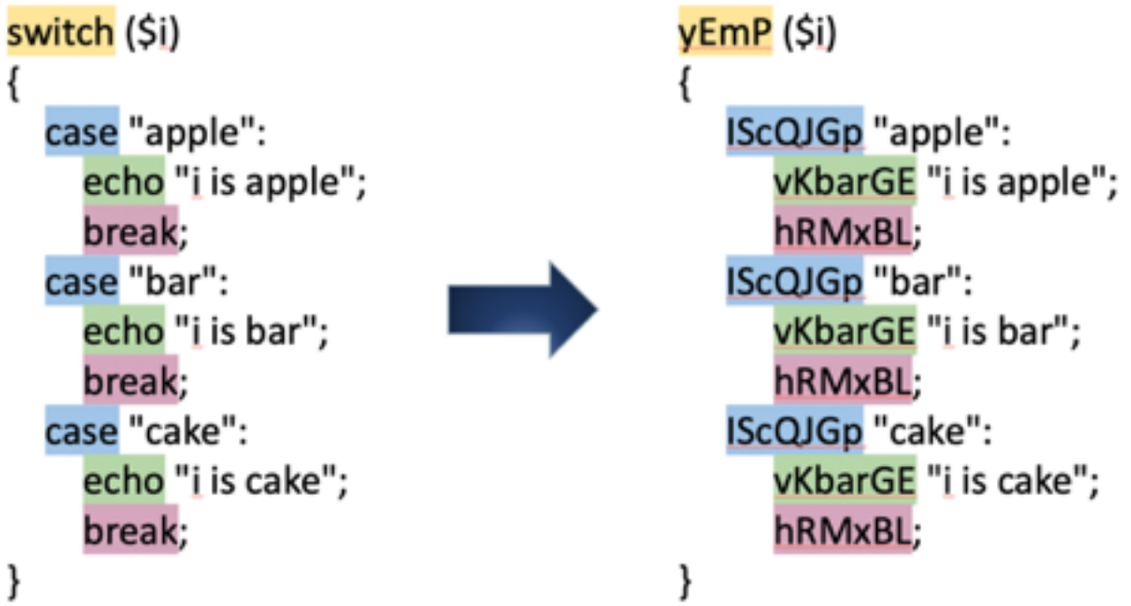
For instance, during optimization the compiler allocates variables to registers. Register allocation often happens by a predictable algorithm, such as first come, first serve. But if we instruct the compiler to randomly choose a register, we would have introduced some entropy into the system that the attacker would not expect.

Polymorphic Linux relies on custom compilers that generate unique binaries that allow for the constant rearrangement of all the specific details of a program binary, from function addresses to instruction sequences to register allocation. By "scrambling" these technical details, the protected software programs and systems effectively become immune to any exploit that relies on prior knowledge of these details. For example, if an attack was expecting to find credit-card data in register "eax," that data would no longer be in that location.

Every time you randomize a particular component, you introduce entropy that reduces the likelihood of successful memory-exploiting attacks. In other words, the application's memory landscape is a constantly shifting moving target, making exploitation practically impossible, resource-intensive and time-consuming. Polyverse's technology can thoroughly randomize the entire program layout, such that the chances of a successful attack is roughly 8x10-15. In other words, with Polyverse in place, attackers are a billion times more likely to win the lottery than break through Polyverse protections.

What's Next: Polyscripting

Beside memory-based attacks, another large category of attacks that has been devastating to the digital world is code injection. Polyverse is currently working on an R&D project that applies its Moving Target Defense strategy directly to the programming language itself. Polyscripting takes a programming language and scrambles the syntax and grammar within the source for that language before the interpreter is compiled. The output is a dictionary that is used to transform all necessary source code before it runs in production. This results in an application that has its own unique implementation of a language. The new interpreter no longer understands the original syntax and grammar of the original language. It will only execute the source code that matches the newly generated unique interpreter. Additionally, this process can be repeated on demand, adding additional layers of defense, making time an ally to a system's defenses through the use of regular intervals at which the interpreter and source code undergo new rounds of Polyscripting.



The Polyscripting process emulates a moving target, remapping the application's address space so frequently that proper enumeration, crafting and execution of an exploit becomes impractically difficult. This skews the effort-to-reward ratio so that it is no longer in a hacker's favor. Currently, Polyscripting can be deployed on WordPress sites.

Polyverse deployment and performance

Polyverse's Moving Target Defense technology has been deployed in many military and Department of Defense environments, protecting some of the government's most valuable strategic assets. For the first time, Polyverse has now made the same Polymorphic Linux product available to the commercial market.

Today, Polyverse Polymorphic Linux provides protection for the Linux open-source programming stack. We have successfully randomized (and tested) the top 70,000 open-source packages, including Java, Ruby, PHP and others. When you deploy a scrambled open-source package from Polyverse, as opposed to the original open-source distribution, you are protected against attacks that target widespread open-source vulnerabilities such as those used in Spectre, Equifax, WannaCry and SystemD.

Polymorphic Linux is easy to use and seamlessly integrates into your development and operations process. To use, simply point your package repository to Polyverse repositories instead of a community run mirror. For standard installations, this is done through a simple shell script:

```
curl https://repo.polyverse.io/install.sh | sh -s <auth key>
```

From then on, Polymorphic Linux operates identically to standard Linux. Installing packages and patching are done through Linux standard tools like yum and apt. Furthermore, the installed packages are both protected by Polyverse and continue to run with identical performance and semantics to their unprotected counterparts.

The Polyverse functionality has been thoroughly tested, by us, by our government installations, and by third-parties.[4] The scrambled library easily defeats memory-exploiting attack attempts at near-zero performance cost, and it remains compatible with any other components with which the software interacts.

Conclusion

Moving Target Defense is the practical application of nature's genetic diversity to technology. It creates a program that, while identical in function, is entirely unique from any previous version. MTD is a game changer in cybersecurity. It redefines the power balance between defenders and attackers. Polyverse is the leading innovator of MTD technology. And our products, Polymorphic Linux and Polyscripting, render system compromise via memory exploits both prohibitively expensive and in practice impossible for attackers.

For more information

Contact support@polyverse.io or visit our website https://polyverse.io/.

4 For instance, ISACA did a penetration testing report on Polyverse. The report can be downloaded at https://www.isaca.org/Knowledge-Center/Research/ResearchDeliverables/Pages/Polyverse-Case-Study.aspx.