Zeus Cybersecurity Solution for Protecting Digital Infrastructure

Zeus is an innovative cybersecurity solution that implements patented[1] software hardening technology to protect current and future systems for digital infrastructure. These systems for digital infrastructure include cloud computing systems, data center systems, embedded systems, and internet of things (IoT) systems. Zeus enables systems for digital infrastructure to defend themselves from existing and emerging cyber attacks by hardening the software that runs these systems.

Zeus protects systems for digital infrastructure from critical cyber attacks[2] that exploit software vulnerabilities in your systems for digital infrastructure. A software vulnerability is a security flaw that can be exploited by an attacker to violate an exploit or implicit cyber security policy.[3] Cyber attackers frequently exploit memory corruption and buffer overflow vulnerabilities in systems for digital infrastructure.[4] According to the MITRE's Common Vulnerabilities and Exposures (CVE) reports, cyber attacks that exploit memory corruptions and buffer overflows vulnerabilities account for 20-40% of all reported cases.[5]

Zeus shields software developed in the C and C++ program languages[6] against cybersecurity attacks that exploit these software vulnerabilities by hardening the software. Zeus uses encryption based technique to protect code pointers that are used to control the flow of executions of software codes on memory. The CERT Division of SEI (Carnegie Mellon University Software Engineering Institute) is the world's leading cybersecurity research institution. According to the CERT C and C++ Coding Standards [7] and Secure Coding in C and C++ [8], code pointers must be protected in memory unsafe program languages such as C and C++ because these code pointers can cause software vulnerabilities, and thus, can be exploited for memory corruptions and buffer overflows attacks.

[1] Zeus patents US8583939, US10579806, US20200076593, and US17/145790
[2] These include the common and most damaging zero-day cyber attacks.
[3] This violation can cause the software or system to crash or produce invalid output or to behave in an unintended way.
[4] These attacks are the typical zero-day cyber attacks that are exploiting software vulnerabilities.
[5] Security Vulnerabilities By Type: https://www.cvedetails.com/vulnerabilities-by-types.php
[6] C and C++ are used for software that run most systems for digital infrastructure and ranked as the third and fourth top program languages by IEEE (Top Programming Languages 2020: https://spectrum.ieee.org/at-work/techcareers/top-programming-language-2020?referrer=%2F).
[7] https://wiki.sei.cmu.edu/confluence/display/seccode/SEI+CERT+Coding+Standards
[8] https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=54183

For Zeus details, please contact Alex G. Lee (alexglee@zeusswdefcom)

Zeus protects four types of code pointers that can be exploited for the memory corruptions and buffer overflows attacks [9]: The first and most common type is return addresses that support the nested execution of functions. Return addresses are used by the return instruction to return control to the calling function. The second type is the function pointer variables[10] appearing as literals and declared in source programs. They are used to hold the address of a function that can be called at runtime. The third type is the code pointers belong to compiler-generated pointertables used for invoking functions in shared libraries. Return instructions, indirect function calls, and invocations of shared library functions reference these code pointers correspondingly. The fourth type is function pointer tables attached for initialization or finalization by compilers and toolchains. These are structurally similar to compiler-generated pointer tables, and used in a similar fashion to function pointers.

Zeus modifies compilers and toolchains to harden executables against exploits for cybersecurity attack by encrypting these four categories of code pointers. Zeus reencrypts dynamically return addresses in the stack to reinforce the encryption by dynamic reencryption to renew encryption states during program execution. Zeus operates during program compilation, linking, loading, and early execution stage. Zeus produces executables that randomize code pointers to guard against information leaks.

[9] For Zeus technology details, please see Zeus whitepaper: https://github.com/alexgeunholee/zeus-software-security
[10] They are called function designators in the C language definition.

Zeus provides greater protection against cyber attacks with lower overhead than any other known solutions.[11] Zeus can successfully mitigate real world cybersecurity attacks that are reported in CVE related to buffer overflows and other memory corruption defects.[12] For example, Zeus can block the control-flow hijacking caused by stack buffer overflow vulnerability CVE-2018-18409 in the open source TCPFLOW project; CVE-2018-17439 and CVE-2018-15671 of the HDF5 library; and prevent the Nginx web server leaking a return address through Return Oriented Programming (ROP) and Blind ROP attacks as described by CVE-2013-2028.[13] Especially, Zeus' dynamic reencryption of return addresses can defend effectively against the brute-force type attacks to lick critical control information as in Blind ROP attacks, which is not possible in other competing solutions.

Zeus prevents cyber security breaches caused by buffer overflow attacks of computer memory and limits the impact of control information leak attacks with minimal performance overhead. Benchmarking of C programs with O3 optimization of the recent implementation shows execution overheads of 7.66% for CPU-intensive and 3.52% for IO-intensive workloads, respectively.[14] These execution overheads are lower than reported overheads in other competing cybersecurity solutions. [15]

Currently, Zeus is implemented by the LLVM C and C++ Compilers and related toolchains. Zeus protections can be applied to nearly any x86/Intel, ARM or PowerPC computing architecture for current and emerging digital infrastructure. Research and development are ongoing to improve Zeus' performance and coverage further.

[11] PointGuard™: Protecting Pointers from Buffer Overflow Vulnerabilities (https://www.usenix.org/legacy/events/sec03/tech/full_papers/cowan/cowan.pdf); Control Flow Integrity (https://en.wikipedia.org/wiki/Control-flow_integrity); Load-time Function Randomization (https://github.com/immunant/selfrando); Basic Block Randomization (https://patents.google.com/patent/US10140130B2)https://parthad.com/Papers-PDF/2010/ISSRE_memory.pdf); Address Space Layout Randomization (https://en.wikipedia.org/wiki/Address_space_layout_randomization)

[12] There were around 7600 CVE cases over the past 10 years (on average, around 9% of total attack accidents) base on the NIST NATIONAL VULNERABILITY DATABASE (https://nvd.nist.gov/vuln/search)

[13] Please see the demo videos for Zeus defenses against this CVE listed attacks: Nginx Web Server stack based buffer overflow case regarding arbitrary code execution through ROP (https://www.youtube.com/watch?v=FEKQMRrgvws&feature=youtu.be); Nginx Web Server stack based buffer overflow case regarding leaking critical information through Blind ROP (https://www.youtube.com/watch?v=1HL8bQ21_qs&feature=youtu.be)

[14] For Zeus benchmarking test details, please see Zeus whitepaper: https://github.com/alexgeunholee/zeus-softwaresecurity

[15] For example, the overheads are in the range of 7.6-16% for the Control Flow Integrity cases

For Zeus details, please contact Alex G. Lee (alexglee@zeusswdef.com)