Agenda

- Brief history of HardenedBSD
- State of infrastructure
- State of features
- Importance of exploit mitigations
- Current development efforts
- Future long-term research
- Interesting production use cases
HardenedBSD History

- **Goal**: clean-room reimplementation of the grsecurity patchset for the BSD community
  - Tangible focus: Navigate the nexus between infosec and human rights
- **Development started in 2013**
  - First features: PaX ASLR
  - First public in-development ASLR patch: Aug 2013
  - PaX SEGVGUARD in 2014
  - PaX NOEXEC (aka, strict W^X) in 2015
  - Full stack and VDSO randomization in 2015 (completion of ASLR)
- **Secure binary update mechanism in 2015 supports CURRENT and STABLE**
HardenedBSD History

- KLD syscall hardening in 2016
- RTLD hardening in 2016 (*ZERO* creation of executable per-thread stacks on amd64 and arm64)
- Base fully compiled as PIEs in 2016
- OPNsense integrates HardenedBSD’s PaX ASLR in 2016
- Network stack hardening over the years
- [lin]procfs hardening
- Brief LibreSSL experimentation
- Non-Cross-DSO CFI integration
- SafeStack integration
HardenedBSD History

• **Infrastructure**
  - Now have four package build servers: 12-STABLE/amd64, 13-STABLE/amd64, 14-CURRENT/amd64, 14-CURRENT/arm64
  - Two nightly build servers
  - One binary update build server
  - Self-hosted git server (GitLab Enterprise)
    - As of 06 Apr 2021, zero reliance on GitHub (thanks to FreeBSD’s git migration)
  - All public endpoints available via Tor onion service v3
    - Only enterprise OS that can go from dev to prod fully behind Tor
Exploit mitigations - ASLR

- Randomize application virtual memory layout
- Two variants: ASR and ASLR
  - HardenedBSD’s is a clean-room reimplementation of PaX ASLR
- Many distinct implementations
  - Vulnerabilities in one implementation aren’t necessarily applicable to another
- Help protect fully remote attacks
  - Nope: local attacks, MMU, running javascript in a browser, etc.
  - Yep: buffer overflows, integer overflows, format string vulns, etc. remotely
    - Except: infoleaks
Exploit mitigations – Strict W^X

- When a memory mapping is writable, it can never be executable
- When a memory mapping is executable, it can never be writable
- Non-strict W^X:
  - Only protects on mmap, can still transition between W and X with mprotect
- Strict W^X:
  - Protects both mmap and mprotect
  - Also protects across ptrace boundary
- OpenBSD uses non-strict W^X
- NetBSD and HardenedBSD follow the PaX NOEXEC model (strict)
Exploit mitigations – Control Flow Integrity (CFI)

• Ensure that the program’s control flow cannot deviate
• Cross-DSO versus Non-Cross-DSO
• Multiple implementations:
  – PaX RAP
  – Microsoft CFG
  – llvm CFI
• HardenedBSD’s focus is on llvm’s implementation
• CFI is divided into schemes
  – cfi-cast-strict, cfi-derived-cast, cfi-unrelated-cast, cfi-nvcall, cfi-vcall, cfi-icall, cfi-mfcall
Exploit mitigations – SafeStack

- Part of the Code-Pointer Integrity (CPI) project
- Separate the program stack into two distinct regions: the safe stack and the unsafe stack
- Safe stack stores return addresses, register spills, and local variables that are always accessed in a safe way
- Relies on ASLR and W^X for efficacy
- Currently does not cross DSO boundary
- Enabled by default in base, available for use in ports
Exploit mitigations – Security Hardening

- Zero-initialize uninitialized variables (-ftrivial-auto-var-init=zero)
- PIE/PIC and BIND_NOW everywhere
- Retpolines everywhere
- stack-protector-all everywhere
- Disable unprivileged process debugging
- Harden hardlinks
Exploit mitigations – Security Hardening

- Harden both Ptrace and [lin]procfs
- Harden “kernel infoleak as feature”
  - Example: KLD and gpart
- Harden bhyve
  - Run in a jailed context
  - Use MAP_GUARD pages
- Harden RTLD and libthr
  - Cannot map RWX stacks
  - Finer-grained MAP_STACK randomization
Exploit mitigations – Importance Of

- Vulnerability in bhyve fully mitigated by CFI
  - Author had to introduce a second vuln in order to exploit
  - Still need Cross-DSO CFI
- OpenSSL 1.1.1k: fix NULL ptr deref
- FreeBSD CVE-2020-25 577
  - rtsold vulnerability mitigated by SafeStack (turns RCE to data-only)
- FreeBSD CVE-2019-0053
  - telnet client stack overflow vulnerability mitigated by SafeStack
- Use of uninitialized data – nearly 100% mitigated
Exploit mitigations – Importance Of

- Mitigate and make worst case just a DoS
  - Example: NULL ptr deref
- Attackers pay close attention to projects commit-by-commit
- Raise the economic cost of successful attack
  - Attacks differ: pop a shell, flip a bit, exfiltrate data
  - Raise level of skill required
  - Require chaining multiple vulns/exploits
- Protect assets, systems, people
  - Be as OPSEC friendly as possible
Current Development Efforts

- Cross-DSO CFI
  - Significant progress in this effort
  - Issues with ZFS
- Cross-DSO SafeStack
- SafeStack port to arm64
- Fix individual ports entries
  - Some need special care with CFLAGs
  - Ship with exploit mitigations pre-toggled
- Exploit mitigation automated testing framework (Kyua)
- Documentation, documentation, documentation
Current Development Efforts

- **Infrastructure**
  - Need global public mirrors for both installation media and packages
  - Need more arm64 hardware
  - Integrate and experiment with new mixnets

- **Adoption**
  - Help the community deploy and maintain HardenedBSD in unique ways
  - Tor BSD Diversity Project
Current Development Efforts

• Need help with development
  – Lots of issues marked with “help wanted” and “good first issue”
  – Clean-room reimplementation of grsecurity RBAC
  – Clean-room ATF (Kyua) reimplementation of paxtest
  – Integrate Tor support with hbsdmon
  – Don’t write exploit mitigations? No worries! We need all sorts of contributions
  – Don’t write kernel code? No worries! Most of our work is focused on userland
  – Be proactive. No need to wait for me
HardenedBSD In Production

● Emerald Onion
  – Tax exempt, not-for-profit encrypted transit ISP with their own AS
  – Provides 26 (or more) high-bandwidth Tor relays and bridges
    • Traffic exiting from relays may go straight to destination thanks to peering agreements. No more hops to destination.

● OPNsense
  – Open source firewall distribution, forked from pfSense
  – Switched from FreeBSD to HardenedBSD years ago
  – Your Internet-facing firewall deserves exploit mitigations

● DEF CON Hacking Conference
  – Forums, mail, and database servers
HardenedBSD In Production

• The Blackhawk Security Appliance
• Available in 2U and 4U high-availability platforms (two blades, regardless of RU)
• All-flash storage. 4U can currently store a little over 1.5PB
• Store full pcap, tested at 10Gbps
• Place inline (IPS) or span (IDS)
• Correlate IDS/IPS, PCAP, and host audit data via HawkSense
  – Search stored pcaps and replay packets
  – Start endpoint malware scans
  – Orchestrate actions across the network with OpenC2
• Based on HardenedBSD
• ...And more!
exit(EXIT_SUCCESS);