Security Vulnerability Assessment
Closing Summary

Coleman Kane
Coleman.Kane@ge.com

April 18, 2016

There are many legal & ethical concerns. Many of the questions around disclosure as well as vulnerability and exploit research.

- Legal exposure resulting from publishing
- Waiting periods, limited release
- Whether Proof-of-Concept (PoC) code may be adopted by adversaries
- Concerns with non-reporting: Whether those responsible are willing to patch

Vulnerability Assessment involves the process of determining what vulnerabilities exist in the environment, what don't, and how widespread they are. In this regard, it can be considered a vulnerability map of your attack surface.

- Classification
- OWASP process (one example of many, not a rule)
- Mitre CWE - Common Weakness Enumeration
- Microsoft STRIDE
- Lockheed Martin Cyber Kill Chain
- ISO-27002 Comprehensive Information Security Standard

As new software is introduced, or existing software updated, the probability of introducing new vulnerabilities along with it exists. OS-level vulnerabilities are typically of the software variety, but have the added consequence of exposing Supervisor-level access through these vulnerabilities.

- Mitre CVE - Common Vulnerabilities and Exposures
- Memory corruption (stack, heap, etc.)
- Race conditions
- Arbitrary code execution
- Control-flow hijacking
- Trojan Horses (masking intent)
- Covert/subliminal channels (hiding activity)

Software & OS Design and Implementation

Software is increasingly part of many fields of work. A proliferation of the "*Internet of Things*" is only going to amplify this trend. It becomes critical to deliver **secure-by-design** systems, rather than simply managing a vulnerable system.

- Privilege escalation / resource access control management
- Least-privileged access
- Ubuntu AppArmor
- GCC ProPolice / stack-smashing-protection
- SeLinux, HardenedBSD
- Multiple Independent Levels of Security (MILS)
- Auditing
- `https://www.owasp.org/index.php/Source_Code_Analysis_Tools`

Programming languages, and the design decisions within, can be the source of many vulnerabilities. This is especially the case where programmers skilled primarily in one language are instructed to develop in a new language.

- Type system, nuances, machine-representation of native types
- Control-flow, subroutines, argument-passing
- Standard library code, a.k.a. *Runtime*
- Default properties (public, private, etc.)

Network and Protocol Vulnerabiltiies

Networking introduces foreign data (data created outside the current system) streams into your code. Networking also frequently introduces real-time, interactive access to your software for remote and untrusted third-parties.

- Centralized authentication / authorization systems (KDC, ActiveDirectory, etc.)
- File servers (centralized file storage)
- Assets configured and managed by multiple parties
- Viruses & Worms
- Data hostage
- Confidentiality concerns (IPSec, etc.)

Many network-delivered attacks exist:

- Denial of Service (DoS, Distributed DoS)
- ARP Poisoning
- Man-in-the-Middle, Mailbox, Browser, etc.
- Remote Access Tools (RAT)
- Strategic Web Compromise (SWC) a.k.a. Watering-hole or Poison-Well attacks
- Data leakage via browser
- Java, Flash, JavaScript, ActiveX - Malicious executable delivery

Mitigation of vulnerability can be achieved through network and system monitoring, with detection and prevention of events considered to be undesirable.

- Firewalls
- Security Onion - Bro, Suricata, Snort, Network Intrusion Detection Systems (NIDS)
- Anti-Virus
- Host intrusion Detection/Prevention Systems (HIPS)
- Application-level logging & Monitoring

Configuration Vulnerabilities

Software that is otherwise relatively secure can be made insecure through mis-configuration or poorly-managed configuration.

- Scaling problems
- Configuration managment - Jenkins, Chef, Puppet (write once, deploy many)
- Manage configurations in revision-control (like Git, etc...)
- Disable unnecessary services (many OS's install "common" services)
- Reverse Turing Test (can computer identify who's human vs. bot?)

Social engineering continues to be a significant vector of attack. Humans are generally trusting, and also often overworked. These attributes can contribute to success of social engineering attacks.

- Phishing / Spear-phishing
- Pretexting
- Awareness
- Social Engineering aspect of the HBGary Federal hack
  `http://arstechnica.com/tech-policy/2011/02/anonymous-speaks-the-inside-story-of-the-hbgary-hack/`

- Multiple-reviewer mitigation strategies
- Verification