Using protocol fuzzing to harden storage systems and to protect them from 0-day attacks

Mikko Varpiola
Codenomicon Ltd.
http://www.codenomicon.com/
INTRODUCTIONS
About the Speaker

- Codenomicon Co-Founder (test suite developer No. 1)
- Been fuzzing software since 1998
  - Specialized in protocol fuzzing
- Key member in PROTOS research and of OUSPG research group
  - First practical implementations of model based fuzzers
  - Industry wide ASN.1 flaws
- Speaker in various conferences such as Blackhat, CanSecWest, and Techno Security
"Fuzzing is the process of sending intentionally invalid data to a product in the hopes of triggering an error condition or fault. These error conditions can lead to exploitable vulnerabilities."

-- HD Moore
UNKNOWN VULNERABILITY MANAGEMENT
Software *is* Infrastructure
Software products have among the highest levels of defects of any products sold today, and there is very little accountability on the part of the producers of software products. Symptomatic of these defects are product patches, one of the major business and security impacts affecting the [financial] sector. These patches are released to address vulnerabilities that arise from defects in design and/or implementation and are, at best, analgesic.

Programmers are NOT known for writing secure, or bug free code!!!!

- Homeland Security Strategy For Critical Infrastructure Protection in the Financial Services Sector, May 2004
So say we all!

From 1995 to Q3 of 2008 Computer Emergency Response Team (CERT) had 44,074 known vulnerabilities cataloged based on reports from public sources and submitted data.

This means the potential of unknowns is most likely greater and the severity has escalated to current day

http://www.cert.org/stats/

That’s 9.28 new known vulnerabilities per day over past 13 years.
Unknown vs. Known vulnerabilities

**KNOWN VULNERABILITIES**
What you can find with your current security tests.

**UNKNOWN VULNERABILITIES**
What Codenomicon can help you reveal.
Short history of [notable] exploitation of unknown vulnerabilities

- **Morris Worm*** (1988)
  - Phone masters (1994-1995)
  - Citibank (1994)

- **Melissa Virus** (1999)
  - Amazon, Ebay,…

- **CodeRED***
  - PROTOS SNMP Multi-vendor Test suite (2000-2001)

- **SQL Slammer***
  - Xbox GoldenEye exploit for homebrew games*
  - JPEG (and other static content exploits. Around 2004)

- **Stuxnet***
  - Wikileaks (operation Payback)
  - Google.cn (operation Aurora)
  - Morgan Stanley?
  - Indian Defense (Apr, 2010)
  - British Foreign Ministry
  - Gawker Media (+1.5M records stolen…)
  - Kaspersky AV

- **Attacks on Sony**
  - Anonymous
  - LulZsec
  - EMC/RSA SecureID
  - Lockheed Martin (and who others?)
  - South Korea DoD

- **First CyberWar. Between Russia and Estonia (2008)**

- **Era of SQL injection, DDoS, spam and malware**

- **2011**

We are not keeping up...

An estimate of 4/5th of our security spending is on *reactive* technologies and less than 1/5 on *proactive* measures!

Does that sound right?

( and by the way – reactive measures offer little or no defense against attacks based on unknown vulnerabilities )
Big picture – evolution of cyber threat

197x-1995: Early hacktivism and computer crime

1995-2000: Organized cyber- and computer crime

2000-2009: State sponsored hacking

2009-2010: Kinetic damage and kinetic response

2011-2011: Pseudo-Random target selection

1995-: Vandalism / 2nd generation “hacktivism”

2010-2011: iHacktivism

Careful target selection

http://www.whitehouse.gov
Anatomy of a hack

Swordfish / http://www.imdb.com/title/tt0244244/

Sorry….Only in Hollywood!
Anatomy of a hack

- Choose your target
- Study, analyse, identify
- Plan
- Implement
- Execute!
Anatomy of a hack

- Choose your target
- Study, analyse, identify
- Plan
- Implement
- Execute!
Anatomy of a hack

- Choose your target
- Study, analyse, identify
- Plan
- Implement
- Execute!
Anatomy of a hack

- Choose your target
- Study, analyse, identify
- Plan
- Implement & Test
- Execute!
Anatomy of a hack

- Choose your target
- Study, analyse, identify
- Plan
- Implement & Test
- Execute!
Or just use a perfect backdoor

NOTE: Skillsets of exploiting and finding the bug is @crossroads or has already diverged.
Are you compliant, and... 

...ARE YOU RUNNING FAITH BASED SECURITY INITIATIVE? OR FACT BASED ONE?
UVM spells Unknown Vulnerability Management

But HOW? (and what)
We need to start building better SW

Everyone needs to start doing fuzzing and to manage their unknown vulnerabilities!

We need Energy Star for Software fuzzing!

[Unknown] Vulnerability Management

1995 - 2000
SATAN/SAINT
Around 1998-
Nessus,
ISS
2000
Qualys, HP, IBM,
Symantec,…

Known Vulnerability Management

Known vulnerability scanners and policy compliance tools

Unknown Vulnerability Management (UVM)

SAST APPROACH
(1980-)
PC Lint, OSS,
Coverity, Fortify, IBM Ounce
Labs, Microsoft,…

DAST APPROACH
(2000-)
Fuzzing
(OSS, Codenomicon,
Peach, Sulley,…)

Whitebox testing

Blackbox testing

Total
Vulnerability
Management
So I found UV – Now what? Remediate!

- Fix the problem
  - Update version of software
  - Fix defect if you own the IP
  - Report problem to affected vendor
- Isolate the device or subnet
- Write a rule/signature/policy for the IDS/IPS/Gateway/etc
- Turn the service off until issue resolved

---

**DAST APPROACH**

(2000-)

Fuzzing

(OSS, Codenomicon, Peach, Sulley, …)

- No source code needed
- Implementation language independent
- No false positives (e.g. crash is a crash)
- Meaningful remediation paths
- Modern tools easy to use / operate

---

**Unknown Vulnerability Management (UVM)**

**Total Vulnerability Management**
UVM Lifecycle

Test/Scan

Analyze/Identify

Protect/Remediate

Report
We are getting there…

- Federal/Mil. requirements:
  - DoD DIACAP,
  - NIST SP.800,
  - (Common Criteria)
  - …

- Private infrastructure:
  - All major US carriers are fuzzing
  - Verizon now requires it as lab entry criteria

- Enterprise world following suit
  - Amphion Forum

- Industry forums
  - SNIA
  - MSFT
  - Blutooth SIG

DoD DIACAP
http://iase.disa.mil/diacap/

NIST SP.800
http://csrc.nist.gov/publications/PubsSPs.html
...and we need to build it right!

Modern SDLs such as Microsoft SDLC, new Cisco SDLC and BSIMM all recognize that fuzzing has a key role in creating a secure and rugged software.

Fuzzing in the Verification phase of the MS SDL
Unknown (0-day) vulnerability

Difference between security- and QA-bug is just who finds it!

http://www.google.com/googlebooks/chrome/big_08.html

When 0-day exploit is seen in public, then flaw its exploiting becomes a known vulnerability and it is then added to known vulnerability tools and databases.
Picture from http://blogs.abiss.gr/mgogoulos/entry/fuzzing_finding_vulnerabilities_in_software
What is fuzzing

---

THERE ARE SEVERAL WAYS WE TEST EACH CHECK-IN, FROM UNIT TESTS OF INDIVIDUAL PIECES OF CODE --

-- TO AUTOMATED UI TESTING OF SCRIPTED USER ACTIONS LIKE "CLICKED BACK BUTTON... WENT TO PAGE..." --

-- TO FUZZ TESTING: SENDING YOUR APPLICATION RANDOM INPUT.

Pam Greene, Software Engineer

http://www.google.com/googlebooks/chrome/big_10.html
What can be fuzzed, what are the failures?
Case study - filer

Applications:
Anti Virus, ...

Access:
SMB/CIFS, SMB2, SunRPC, NFSv2, NFSv3, NFSv4, iSCSI, FCoE, ...

Management:
SNMPv1, SNMPv2c, SNMPv3, SSH, Telnet, HTTP
TLS/SSL, IPMI,...

Operating system:
IPv4 (ARP, IP, UDP, TCP, ICMP, IGMP), IPv6 (IP, ND, ICMP6, MLD, TCP, UDP), IPSEC, DNS, DHCP, NTP, ....
Fuzzing is a method to test software
Fuzzing is fuzzy

- There are several types of fuzzing!
  - All find problems, others more effective than others
  - Environment should dictate types used
- Generational fuzzing
  - Model based (RFC, specification)
  - (fake model based via traffic capture)
- Template (or mutation) based fuzzing
  - Traffic capture / replay fuzzing
  - Adaptive sample based fuzzing
  - Random fuzzing

DEFENSICS test suites perform model based fuzzing
DEFENSICS traffic capture test suite (TCF) Performs traffic replay fuzzing
DEFENSICS Universal Fuzzer (DUF)
Finally – let's not forget...

Rugged Software Manifesto – part #4

(http://www.ruggedsoftware.org/)

I recognize that my code will be used in ways I cannot anticipate, in ways it was not designed, and for longer than it was ever intended.
Thank You!

Go FuZZing!!! 😊

Oh – Questions?