Relative Vulnerability: An Empirical Assurance Metric

Crispin Cowan, Ph.D.

Chief Scientist, ~~WireX~~

Immunix, Inc.

Assessing the Assurance of Retro-Fit Security

- Commodity systems (UNIX, Linux, Windows) are all highly vulnerable
  - Have to retrofit them to enhance security
- But there are lots of retrofit solutions
  - Are any of them effective?
  - Which one is best?
  - For my situation?

What New Capability Would Result?

- Instead of “How much security is enough for this purpose?”
- We get “Among the systems I can *actually* deploy, which is most secure?”
  - Tech transfer experience: customer says “We are only considering solutions on FooOS and BarOS”
- Relative figure of merit helps customer make informed, realistic choice

Why Now?

Old

- Stove pipe systems, made to order
- Orange book/Common Criteria lets customer order a custom system that is “this” secure
- The question is “Is this secure enough?”

New

- Reliance on COTS
- Customer must choose among an available/viable array of COTS systems
  - And possibly an array of security enhancements
- The question is “Which is best?”

State of the Art

Common Criteria

- High barrier to entry:
  - ~ $1M for initial assessment
- Hard to interpret result
  - Only a particular configuration is certified, and it may not relate to real deployments
- 3-bit answer: EAL0-7
  - Several of which are meaningless (0-2 useless)
  - Others are infeasible (6 & 7 are too hard for most systems)
  - Really 2-bits: none, 3, 4, 5

ICSA

- Lower barrier to entry
  - But still high enough that most retrofit mechanisms are not certified
- Hard to interpret result
  - ICSA certifies that whatever claims the vendor makes are true
  - *Not* whether those claims are meaningful
- 1-bit answer: certified/not

Proposed Benchmark: Relative Vulnerability

- Compare a “base” system against a system protected with retrofits
  - E.g. Red Hat enhanced with Immunix, SELinux, etc.
  - Windows enhanced with Entercept, Okena, etc.
- Count the number of known vulnerabilities stopped by the technology
- “Relative Invulnerability”: % of vulnerabilities stopped

Can You Test Security?

- Traditionally: no
  - Trying to test the negative proposition that “this software won’t do anything funny under arbitrary input”, I.e. no surprising “something else’s”
- Relative Vulnerability transforms this into a positive proposition:
  - Candidate security enhancing software stops at least foo% of unanticipated vulnerabilities over time

Immunix Relative Vulnerability

- Immunix OS 7.0:
  - Based on Red Hat 7.0
  - Compare Immunix vulnerability to Red Hat’s Errata page (plus a few they don’t talk about :-)
- Data analyzed so far: 10/2/2000 - 12/31/2002
  - 135 vulnerabilities total

Vulnerability Categories

Local/remote: whether the attacker can attack from the network, or has to have a login shell first

Impact: using classic integrity/privacy/availability

- **Penetration**: raise privilege, or obtain a shell from the network
- **Disclosure**: reveal information that should not be revealed
- **DoS**: degrade or destroy service

Immunix 7.0
Relative Vulnerability

	Not Stopped	Stack Guard	Format Guard	Race Guard	Totals
Local Penetration	38	12	6	3	(21/59) 35.6%
Remote Penetration	17	8	4	0	(12/29) 41.4%
Local Disclosure	11	0	0	0	(0/11) 0%
Remote Disclosure	7	0	0	0	(0/7) 0%
Local DoS	11	0	0	6	(6/17) 35.3%
Remote Dos	5	0	0	0	(0/5) 0%
Totals	89	20	10	9	39/135 28.9%

Version Churn

- Previous data compared Red Hat 7.0 to Immunix 7.0
  - 2 year old technology
  - Notably did *not* include SubDomain
- Defcon 2002 system: Immunix 7+
  - Mutant love child of Red Hat 7.0 and 7.3
  - No valid basis for RV comparison
- Next up: Red Hat 7.3 vs. Immunix 7.3

Immunix 7.3 Relative Vulnerability

	Not Stopped	Stack Guard	Format Guard	Race Guard	Sub Domain	Totals
Local Penetration	2	5	0	0	8	(10/12) 83.3%
Remote Penetration	2	4	0	0	6	(7/9) 77,8%
Local Disclosure	0	0	0	0	0	(0/0) 0%
Remote Disclosure	1	0	0	0	0	(0/1) 0%
Local DoS	1	1	1	0	3	(3/4) 75%
Remote Dos	1	1	1	0	1	(3/4) 75%
Totals	7	11	2	0	18	23/30 76.7%

Validation: Does early RV predict later values?

- No:
  - added a technology layer (SubDomain) and numbers changed drastically
  - Fashion: bug hunts come in waves, and there has not been a recent wave of race or format bugs
- Yes:
  - StackGuard continues to be the #1 intrusion prevention layer in Immunix

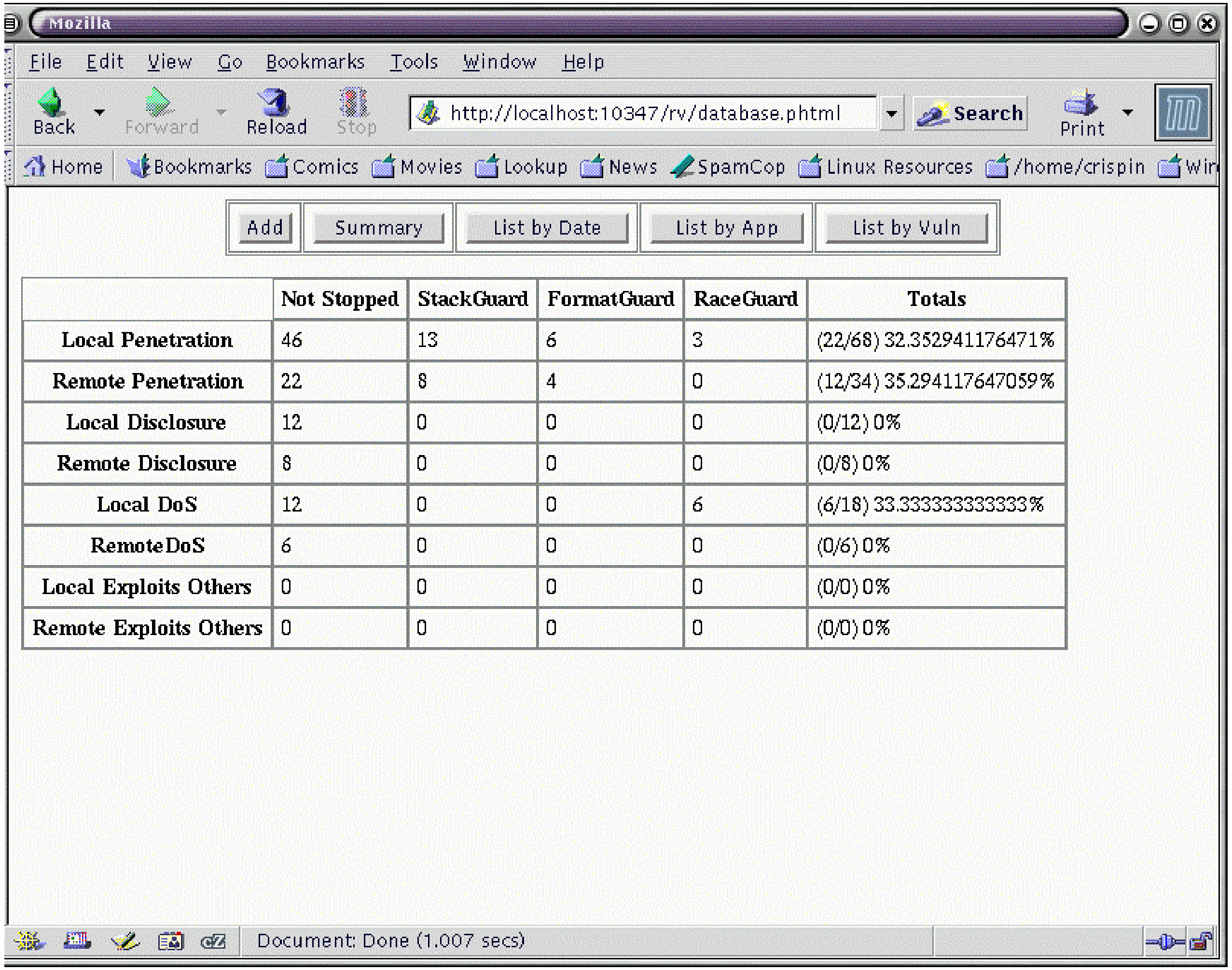
Impact

- Lower barriers to entry
  - Anyone can play -> more systems certified
- Real-valued result
  - Instead of boolean certified/not-certified
- Easy to interpret
  - Can partially or totally order systems

RV Database

- Built a PostgreSQL database of RV findings
- Allows relational queries to answer statistical questions, e.g. “RV for StackGuard vs. Remote Penetration?” or “How many bugs were stopped by more than one technology?”

RV Summary


Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://localhost:10347/rv/database.phtml
Search
Print
Home Bookmarks Comics Movies Lookup News SpamCop Linux Resources /home/crispin Win
Add
Summary
List by Date
List by App
List by Vuln
Document: Done (1.007 secs)

	Not Stopped	StackGuard	FormatGuard	RaceGuard	Totals
Local Penetration	46	13	6	3	(22/68) 32.352941176471%
Remote Penetration	22	8	4	0	(12/34) 35.294117647059%
Local Disclosure	12	0	0	0	(0/12) 0%
Remote Disclosure	8	0	0	0	(0/8) 0%
Local DoS	12	0	0	6	(6/18) 33.333333333333%
RemoteDoS	6	0	0	0	(0/6) 0%
Local Exploits Others	0	0	0	0	(0/0) 0%
Remote Exploits Others	0	0	0	0	(0/0) 0%

Impact

- Empirical measurement
  - Measure *results* instead of *adherence to process*
- Implementation measurement
  - CC **can't** measure most of the Immunix defenses (StackGuard, FormatGuard, RaceGuard)
  - RV can measure their efficacy

Issues

- Does not measure vulnerabilities *introduced* by the enhancing technology
  - Actually happened to Sun/Cobalt when they applied StackGuard *poorly*
- Counting vulnerabilities:
  - When l33t d00d reports “th1s proggie has zilli0ns of bugs” and supplies a patch, is that one vulnerability, or many?

Issues

- Dependence on exploits
  - Many vulnerabilities are revealed *without* exploits
    - Should the RV test lab *create* exploits?
    - Should the RV test lab *fix* broken exploits?
  - Probably ***yes***
- Exploit success criteria
  - Depends on the test model
  - Defcon “capture the flag” would *not* regard Slammer as a successful exploit because payload was not very malicious

Issues

- What is the goal?
  - Access control can keep an attacker from exploiting a bad web app to control the machine
  - But *cannot* prevent the attacker from exploiting a bad application to corrupt that application's data
- Idea: RV for applications
  - Consider the RV of an application vs. that application defended by an enhancement
  - E.g. web site defended by in-line intrusion prevention
  - The *Guard technologies offer some application RV, while SubDomain mostly does not

Work-factor View

- Assume that well-funded attacker can penetrate almost any system eventually
- The question is “How long can these defensive measures resist?”
- RV may probabilistically approximate the work factor to crack a system
  - foo% of native vulnerabilities are not actually exploitable
  - Therefore foo% of the time a well-funded attacker can’t get in that way
  - Attacker takes foo% longer to get in???

Lessons Learned the *Hard* Way

- Security advisories lie
  - often incomplete, or wrong
- Published exploits are mostly broken, deliberately
- Compiled-in intrusion prevention like StackGuard makes it *expensive* to determine whether the defense is really working, or if it is just an incompatibility
  - Also true of diversity defenses

Technology Transfer

- ICSA Labs
  - traditionally certify security products (firewalls, AV, IDS, etc.)
  - no history of certifying secure operating systems
  - interested in RV for evaluating OS security
- ICSA issues
  - ICSA needs a pass/fail criteria
  - ICSA will not create exploits