Keeping Up With The Joneses (CVEs)

David Reyna
Wind River Systems

david.reyna@windriver.com

Security Response Management

Risk, Cost, and Best Practices in an Imperfect World

- Keeping our products secure is a requirement for survival
- Security data is available, but can be a flood of data with varying quality and completeness
- Managing security defects can be very inefficient, resulting in high costs
- We need to share best practices, knowledge, awareness, automation, and tools

Agenda

- Understanding CVE sources
- Understanding CVE quality
- Understanding CVE volume
- Managing your security response
- Costs, best practices, solutions
- New open source ‘Security Response Tool’ (SRTool)

General Security Patch Workflow

- Upstream CVE Sources
  - Gather data/fixes/info
  - Publish CVE Data
- You (OS Vendor/OEM/etc.)
  - Scan upstream CVEs
  - Manage CVE response
  - Fix CVEs
  - Create patches
- Customer
  - Receive patches
  - Test/deploy

(Covered Topics)

CVEs

- CVE (Common Vulnerability Enumerations)
  - The enumerations of the community tracked security vulnerabilities, separated by the year reported (e.g. CVE-2018-12345)
- CVE content
  - Description field
  - Estimated severity score (CCSV), Low to Critical, 0.0 to 10.0
  - Estimated impact and domain scores, e.g. “Attack Vector”, “Privileges Required”, “User Interaction”, “Scope”, “Confidentiality”, …
  - List of affected products and their version numbers (CPEs)
  - List of support links (published information, patches, reproducers, …)
  - Weakness categories (CWE), e.g. “buffer overflow”, “pointer issues”

Upstream CVE Sources

- MITRE
  - Manages the list of CVEs
- NIST (National Institute of Standards and Technology)
  - Manages the National Vulnerability Database (NVD) of CVEs
- Hardware Vendors, Software Maintainers, Distros
  - Many vendors track and share CVE's relevant to their product
  - Many CVE aggregators also available (e.g. cvedetails.com)
- Mailing lists, websites, and forums (public and private)
  - Preview of coming issues, place to discuss issues

CVE Workflow: Normal/Expected

Community	MITRE	SI Vendors / Maintainers	Vendors (NDA)	NIST	Vendors (Public)	Customers
Discover						
	(Private)	*(Work)*	*(Test)*	-		
	Public	Work		**Public**	Watch	Watch
		Fix – public			**Test**	“
					Patch	“
						Receive/ Fix

THE LINUX FOUNDATION

CVE Workflow: Out-of-order/Delayed

Community	MITRE	SI Vendors / Maintainers	Vendors (NDA)	NIST	Vendors (Public)	Customers
Discover						
	Reserved	Work	Test	-		
	“	Fix – Public		-	What?	What?
	“			-	Some Patch	Some fix
	Public			-	More Patch	More fix
				Public	**All Patch**	**All fix**

A High Profile CVE - Simplified

Community	MITRE	SI Vendors / Maintainers	Vendors (NDA)	NIST	Vendors (Public)	Customers
Discover	Reserve					
	“	Work 1,2,3,4	Test 3,4	-		
Public	“	Work 5,6	Test 5,6	-	What?	What?
	Public	Work 7,8	Test 7,8	Public	Sorry	OMG
		Patch A,B NDA	Test, Sorry!		Sorry!	OMG!
		Patch C,D NDA	Test, Sorry!!		Sorry!!	OMG!!
		Patch E,F Public	Test, Sorry!!!		Test, Sorry!!!	OMG!!!
		Patch G Public	Test, Sorry!!!!		Test, Sorry!!!!	OMG!!!!
		Patch H, I, J	Good Enough		Patch H,I,J,…	Fix H,I,J …

The focus here is not the vulnerability itself, but the process and cost in handling that vulnerability

Quality of CVEs: Issues

- CVEs may only have a brief or incomplete description
- CVE affected product list (CPEs) may have gaps, errors, unexpected version deviations, even be empty
- CVE content may be misleading, mentioning one package when it actually affects a different package
- CVEs may have few, inaccurate, or missing content links (discussion, reproducers, patches)
- CVE status changes continually as new information is discovered and shared
- Sometimes delays in content updates

Quality of CVEs: Issues (2)

- The most recently created CVEs (within the last few months) are particularly prone to the above issues, but unfortunately these are often all that organizations have to work with for their pending releases (i.e. there is often no CPE data to work with)

- Tools (e.g. CVE scanners) must insure that (a) they are flexible in processing the information, (b) that they can differentiate between strong and weak data, (c) that expectations are set as to what the tool is able conclude and act upon, and that (d) humans are appropriately included in the process.

Quality of CVEs: Examples 1

- CVE-2017-13220:
  - The CPE says “cpe:2.3:o:google:android:-:*:*:*:*:*:*:*” then talks about upstream kernel issues and refers to a kernel SHA.
- CVE-2014-2524:
  - Has a CPE which claims all releases of “readline” 6.3 and below are vulnerable, but the problem only exists in 6.0 onwards.

Quality of CVEs: Examples 2

- CVE-2017-8872:
  - Against “libxml” resulted in a bug and patch, but upstream ignored it. An almost-identical patch was merged recently but no mention of the CVE was made
- CVE-2018-10195:
  - A case study in 'dark CVEs'. Reserved in MITRE, Red Hat have their own notice and a patch. Since it is for software which is long-dead, this patch will never go upstream.

Volume of CVE Data: Issues

- Volume of CVEs is 1000+ per month and growing
- Every new CVE must be evaluated, even if only a percentage may be applicable
- Costly in sheer numbers and required analysis overhead given the quality limitations
- Incorrectly categorizing a vulnerability can be even more costly in customer escalations and trust

Volume of CVE Data: Example

Tools: CVE System Analysis

- Can be very valuable in targeting product specific review activities
- Tells you of known vulnerabilities, but not what you are NOT vulnerable to
- Scans almost exclusively in the category of 'needs investigation‘
- Depends on known data
- *Example: Nessus*

Tools: CVE Build/Source Analysis

- Can be more precise than system analysis
- Possible for something to trigger a vulnerable warning for components never used
- You still need to determine what you are not vulnerable to, understand the items that were reported, etc.
- Depends on known data
- *Examples: Black Duck, Yocto Project ‘cve-report’, Dependency Tracker*

Security Response Management

- While there is heighten awareness about device vulnerabilities, what is often missing is awareness about the process of managing the security response process itself
- Security response management is overhead, where costs need to be understood and reviewed
- Security response management does not make money, but it does protect money

Security Management: Issues

- The amount of work is growing, in the volume of CVEs and in the product support matrix
- The vulnerabilities often apply differently against different releases
- The data is often not well integration with other systems, for example defect managers, agile managers, compliance tools
- The data is often not aggregated in accessible ways
  - Difficult to share current data between development teams
  - Difficult to share current status between teams and management
  - Continual re-gathering status for reporting to management and customers
- Embargoed data requires special handling
  - Compliance tracking and reporting
  - Who knew what when

Security Management Services

- Some companies offload this process to external vendors
  - They can provide the missing expertise and/or resources
  - The pass-through can reduce customer response times
  - The external support can be expensive

Defect systems vs. Security Management

Defect systems are often poor security management systems

- Defects are per product, CVE's are across products
- An issue may need to be tracked before a CVE is created or published
- Hard to manage embargoed data in defect systems
  - Projects are normally public to entire product groups
  - Would require shadow projects
  - Would require a shadow project per authorized access list
- Awkward promoting private issues to public defects

Cost overview: Necessary costs

- Tracking upstream CVE's
- Creating and fixing defects
- Provide updates to customers, management
- Provide patches to customers

Cost overview: Unnecessary costs

- Repeated manual polling of upstream data, initial and all updates
- Repeated manual polling of defect status
- Manually un-assisted analysis of each CVE for vulnerability status, across products
- Manually re-analyzing each updated CVE for vulnerability status
- Manually tracking and sharing patches, reports, documents, ...
- Manually regathering status for customers, management
- Manually tracking private data, and "who knew what when"
- Manually repackaging data for public database

Best Practices

- Automate as much of the process as possible
  - CVE data gathering, updating, change notifications
  - Defect update polling, with filtered change notifications
  - Report tools for management and customers
  - History and audit tracking
- Use multiple sources
  - NIST, MITRE, distros, oss-security, linux-distros (private list), …
- Aggregate the data
  - Central database, central document store

Best Practices (2)

- Provide easy access to the data
  - GUI interfaces, command line scripts
- Be flexible with the data
  - Design for the imperfections of the upstream data
  - Defocus details (like version numbering) during analysis, to avoid big misses from small errors
- Provide tools for CVE inflow triage
  - Provide tools to help walk the volume of CVEs
  - Provide heuristics to help provide guidance given the gaps in the provided information
- Provide management for NDA information
  - Central but safe storage, user restrictions, easy promotion to public database

Introducing the SRTool

- Wind River has developed a tool called the “Security Response Tool” based on its cumulative experience
- Its goal is to address the process pain points and inefficiencies, to scale with a limited staff, and to implement best practices
- Wind River has shared this with open source

Srtool Features for Best Practices

- Automation for multiple CVE source updates, defect status, report generation, history audit data
- Easy access via web interface and command line scripts
- Data aggregation in SQL database, download directory
- Data flexibility by design
- Tool for CVE inflow triage, with guidance heuristics
- NDA management via user model, deploy model

SRTool: Vulnerability Page Example

Affected Products

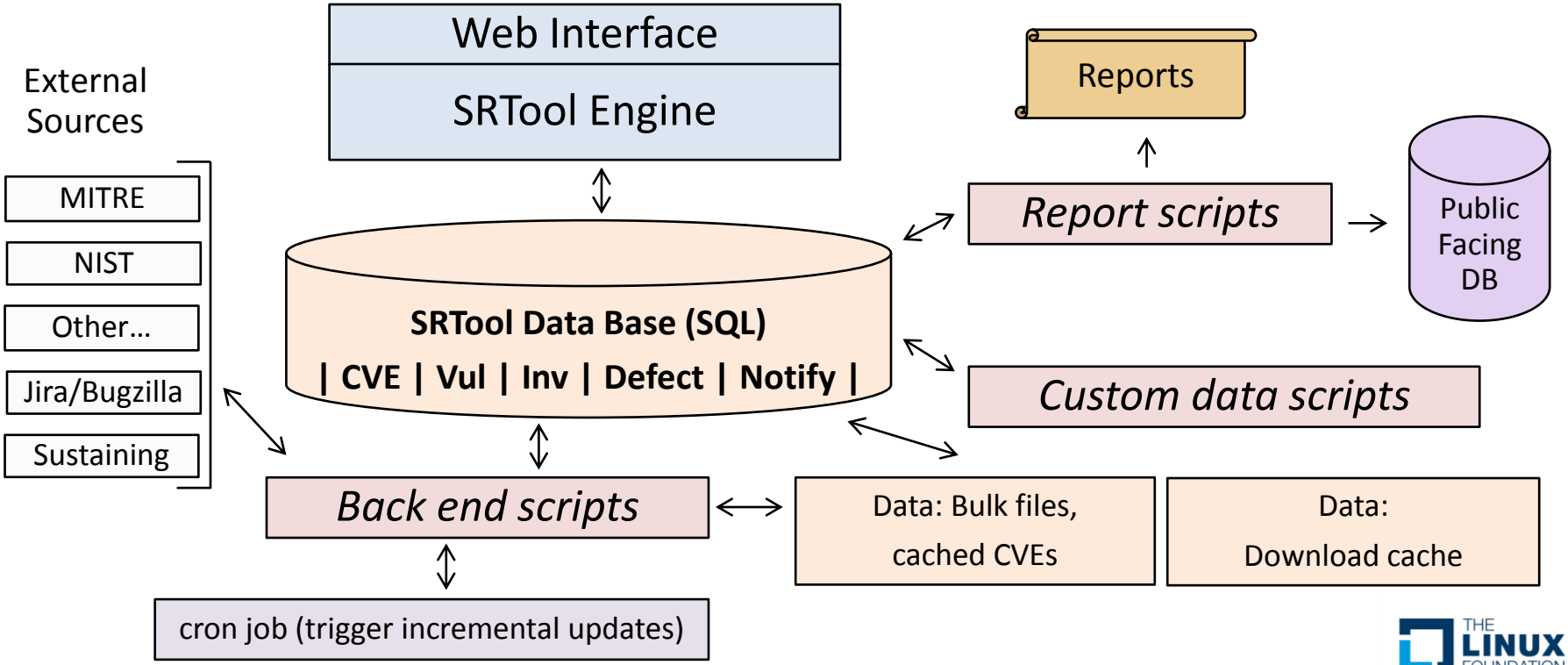
Add product ...

Product Name	Investigation	Status	Outcome	Defect	Release Version	Manage
Linux Customer Content Management	I9118	Not Vulnerable	Closed	LINCCM-2020 | LINCCM-2022 | LINCCM-2160 | LINCCM-2159 | LINCCM-2035 | LINCCM-2158 | LINCCM-2101 | LINCCM-2100 | LINCCM-2028	WRL 4.3 | WRL 4.3 | WRL 8.0 | WRL 8.0 | WRL 4.3 | WRL 5.0.1 | WRL 3.0.3 | WRL 3.0.3 | WRL 8.0	
Wind River Linux 5	I12769	Vulnerable	Fixed	LIN5-24077	5.0.1.42	
Wind River Linux LTS-17	I20518	Vulnerable	Open	LIN10-2989 | LIN10-3041	10.17.41.9 | 10.17.41.1	
Wind River Linux 9	I25978	Vulnerable	Open	LIN9-6155 | LIN9-6164	9.0.0.15 |	
Wind River Linux 8	I33082	Vulnerable	Open	LIN8-8498 | LIN8-8509	8.0.0.25 |	
Wind River Linux 7	I41608	Vulnerable	Open	LIN7-9344 | LIN7-9345	7.0.0.28 | 7.0.0.28	
Wind River Linux 7 SCP	I48600	Vulnerable	Open	SCP7-747	7.0.0.28	
Wind River Linux LTS-18	I49901	Not Vulnerable	Closed	LIN1018-313	unknown	
Wind River Linux 6	I53293	Vulnerable	Open	LIN6-14153 | LIN6-14156	6.0.0.37 | 6.0.0.31	
Wind River Linux 6 SCP	I62016	Vulnerable	Open	SCP6-1119	6.0.0.37	

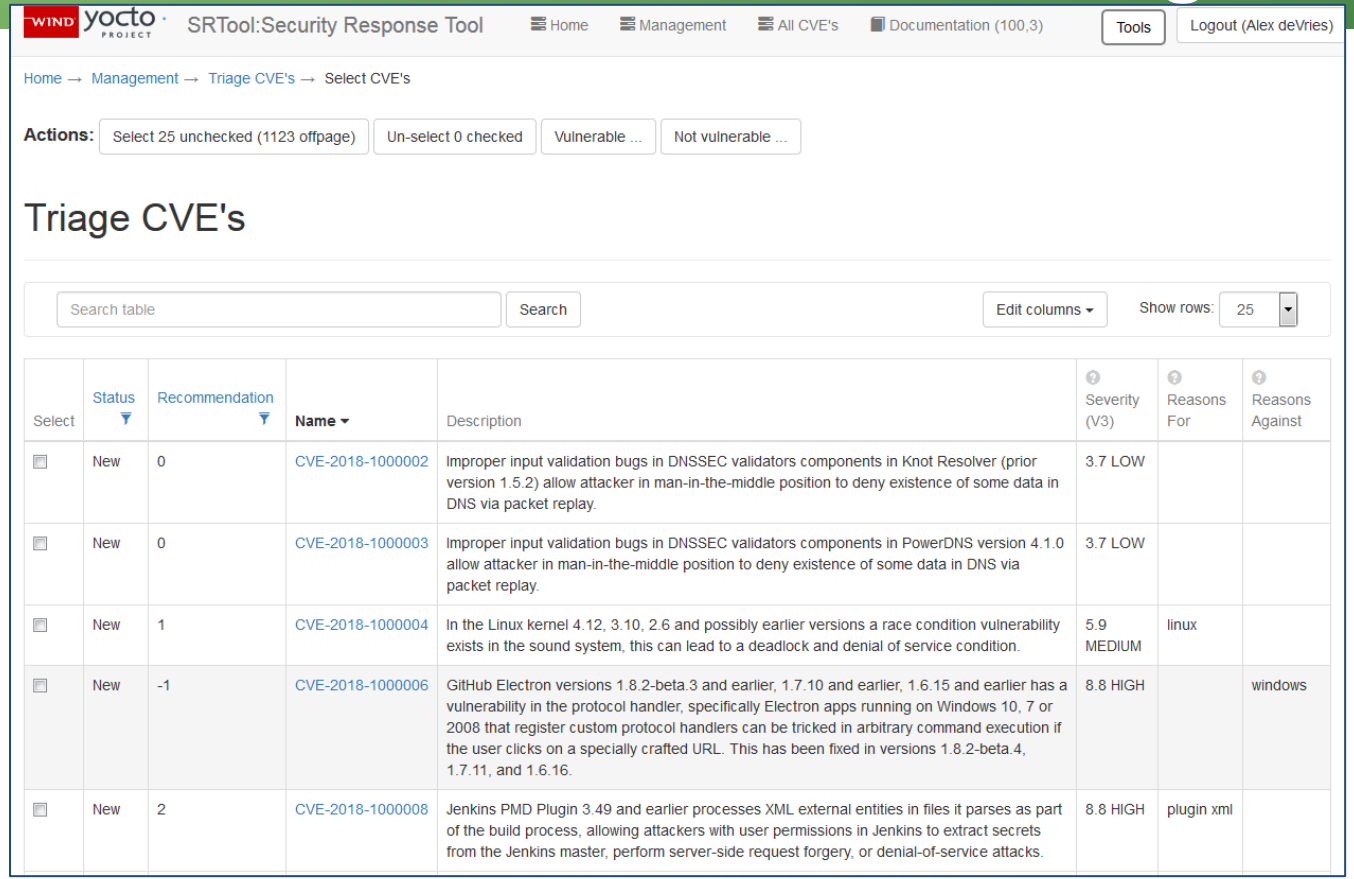
SRTool: Object Model

- **Data source**: represents external content, like CVE data providers, defect system, and sustaining team
- **CVE**: the representation of the upstream CVEs
- **Vulnerability**: the mapping of CVE(s) issues across the products
- **Investigation**: the mapping of a vulnerability to specific product/defects
- **Defect**: the mapping to the organization's defects (Jira, Bugzilla) to CVE's via Investigations
- **Notifications**: automatic messaging of changed upstream CVE and internal defect status

SRTool: Functional Layout



SRTool: Guided Incoming CVE Triage



Select	Status	Recommendation	Name	Description	Severity (V3)	Reasons For	Reasons Against
☐	New	0	CVE-2018-1000002	Improper input validation bugs in DNSSEC validators components in Knot Resolver (prior version 1.5.2) allow attacker in man-in-the-middle position to deny existence of some data in DNS via packet replay.	3.7 LOW		
☐	New	0	CVE-2018-1000003	Improper input validation bugs in DNSSEC validators components in PowerDNS version 4.1.0 allow attacker in man-in-the-middle position to deny existence of some data in DNS via packet replay.	3.7 LOW		
☐	New	1	CVE-2018-1000004	In the Linux kernel 4.12, 3.10, 2.6 and possibly earlier versions a race condition vulnerability exists in the sound system, this can lead to a deadlock and denial of service condition.	5.9 MEDIUM	linux	
☐	New	-1	CVE-2018-1000006	GitHub Electron versions 1.8.2-beta.3 and earlier, 1.7.10 and earlier, 1.6.15 and earlier has a vulnerability in the protocol handler, specifically Electron apps running on Windows 10, 7 or 2008 that register custom protocol handlers can be tricked in arbitrary command execution if the user clicks on a specially crafted URL. This has been fixed in versions 1.8.2-beta.4, 1.7.11, and 1.6.16.	8.8 HIGH		windows
☐	New	2	CVE-2018-1000008	Jenkins PMD Plugin 3.49 and earlier processes XML external entities in files it parses as part of the build process, allowing attackers with user permissions in Jenkins to extract secrets from the Jenkins master, perform server-side request forgery, or denial-of-service attacks.	8.8 HIGH	plugin xml	

- CVE incoming rate 1000+ a month
- View for fast review and triage
- Heuristics from the previous defects to help guide the filtering process

THE LINUX FOUNDATION

SRTool: Next Steps

- The SRTool is under active open source development, so come join us!
- The design is modular, so it is easy to add your data sources and implement your business rules
- The SRTool is intended at this time to be an internal tool, with scripts to export clean data to the organization's public CVE site
- The community page is hosted here:
  - https://wiki.yoctoproject.org/wiki/Contribute_to_SRTool

Conclusion

- There is quite a wealth of vulnerability information available.
- With knowledge, awareness, adaptability, and automation, we can manage this struggle.
- We need to spend people's time on the actual problems, not the process

- Use these links to learn more:
  - https://lists.yoctoproject.org/listinfo/yocto-security
  - david.reyna@windriver.com (SRTool maintainer)

See a live SRTool demo at the Yocto Project Booth!

Embedded Linux Conference Europe

OpenIoTSummit Europe

THE LINUX FOUNDATION