Case Study: Connecting Vulns to Products

Beverly Miller | bmiller2@lenovo.com
Scott Kelso | skelso@Lenovo.com
Your Presenters

Beverly Miller
PSIRT Principal Project Manager
• 20+ years PMI Certified
• Lean Six Sigma
• Member of FIRST.org & MITRE CVE Board
• FIRST PSIRT Framework Working Group

Scott Kelso
Manager, Product Security Office
• 30 years PC industry HW & SW engineering
• 3 years product security
• Master Inventor with 82 patents issued
• Member of FIRST.org
Lenovo’s Product Security Organization

- VP, Research & Technology
- Director/Mgr Security Office
- PSIRT
- Compliance
- WW Security Programs
- Supply Chain
- Technical Programs

- Contact: psirt@lenovo.com
- Advisories: https://support.lenovo.com/product_security/home
Introduction

• Problem Introduction
• Problem Solving Attempt
• Today’s Solution
• Lessons Learned
• Next Steps
The Problem

• Lenovo’s PSIRT supports:
  – 500+ hardware products (notebook, desktop, tablet, server, storage, etc…)
  – 20000+ components (drivers, firmware, apps, utilities)
  – ?? Attributes (3rd party/open source code included in components)

• In one year, PSIRT tracked 402 vulnerabilities, resulting in 5590 development tasks. Nearly half rejected as ‘not applicable’ = WASTED TIME because
  – We don’t know what components are affected by reported vulnerabilities
  – We don’t know what components belong to what hardware products
  – We don’t know what 3rd party/open source software is included in components

• Need to solve
  – How do we manage the complexity and volume we already have?
  – How do we scale for the future as more, faster, larger vulnerabilities are known?
First Attempt: Jira + Jira = Expensive Failure

• 2 linked Jira projects
  – Contracted out to ‘jira’ company who doesn’t fully understand our business
  – “Is this what you mean?”

• Project 1: Task assignment and workflow
  – Jira is GREAT at this!

• Project 2: ‘Database’ associating products, components and attributes
  – Jira is TERRIBLE at this!
  – Required use of spreadsheets for importing
  – Significant time to maintain spreadsheet
    - Adding new/removing end of life products
    - Adding new components and linking to products
    - Adding attributes and linking to components

RESULT: Did not resolve need for relational tracking of issues.
Second Attempt: Jira + Relational DB = Success!

• Bigger Picture: “What assets do we already have?”
  – Good Jira workflow and ticketing tool
  – eSupport knowledgebase containing products and component relationships
  – Relational database & web application coding skill
  – Composition analysis tools for identifying 3rd party code (Black Duck Binary Analysis/Protecode)

• eSupport Knowledge Management DB
  – Tells us what components are supported on hardware products and where they live (download URLs)

• Product Attribute Database (PAD) development
  – Leverages Knowledge Management DB
  – Relational; Connects products, components, and attributes

• Utilize composition analysis tools
  – Tells us what 3rd party code is included in each component

RESULT: Allows for simpler ‘one step’ opening/assignment of cases
Our Salvation: The eSupport Knowledge Base

They already model the product BOM

The development teams accept they have to populate it

And eSupport has an API. Woot!
Product Attribute Database (PAD) – Data Model

• **Product**
  – **Component** (PSL assigned)
    - Attribute
    - Attribute
  – **Component** (PSL assigned)
    - Attribute
    - Attribute
  – **Component** (PSL assigned)
    - Attribute
    - Attribute

• **ThinkPad T460**
  – **BIOS** (PSL 1)
    - BIOS
    - BIOS - Phoenix
    - EDKII
    - openssl
  – **Realtek Audio driver** (PSL 2)
    - Realtek Audio driver
  – **Lenovo System Update** (PSL 3)
    - Lenovo System Update
    - Antlr
  – **Synaptics Touchpad driver** (PSL 4)
    - libpng
    - zlib

• **System x 3950 X5**
  – **BIOS** (PSL 5)
    - BIOS
    - BIOS - Insyde
    - EDKII
    - openssl
  – **IMM2** (PSL 6)
    - IMM2
    - glibc
    - ntp
  – **Lenovo GSS** (PSL 7)
    - Lenovo GSS
    - ntp
    - openldap
Integrating Issue Tracking & Product Structure Tools

Product Security Leads

ALIRT 2 Jira Application

ScriptRunner

Product Attribute Database (PAD)

PSIRT Administrator

Jira Workflow Tool
Integrating Issue Tracking & Product Structure Tools

Why go outside Jira?
- Can implement a data model suitable for product structure
- Performance – Jira chokes on large datasets

What problems result?
- Now you’re a developer too
- IT administration is harder
  - Two skillsets
  - Two user directories
- More middleware to license
Making eSupport Data Work for PSIRT

• Starts with synced copy of Knowledge Management DB
  – Houses all products and components, except JV and China-unique
  – Used to publish tips/KB articles and component updates/code to Lenovo Support site

• Modifications necessary
  – Remove unsupported products from view
  – Add product information such as code names, lifecycle dates
  – Assigned priorities at product/component levels (SLA)
  – Add and associate ‘attributes’
  – PSL assignment
  – nFeed fields
Demo
What We’ve Learned (so far)

• You probably can’t describe everything you do in one go, and developers hate this
  – Learned: If possible, find developers with whom you can have a agile, long-term relationship
  – Learned: Contracted resource too inflexible…and thus will become costly

• Writing the application is more complicated than everyone thinks it will be
  – Learned: Don’t try to be perfect – make something, demo, listen, adapt and make something more

• Tomorrow’s problem will be different from today’s problem
  – Learned: Extensibility – scale and complexity of vulnerabilities will grow (side-channel again!)
  – Learned: Speed – customers want answers NOW
  – Learned: Automation – design to integrate with other tools: Vulnogram, MITRE CVE git, …

• Your internal customers don’t use your tools the way you think they do
  – Learned: For some teams, product volumes (and locked-in processes) still require…spreadsheets
  – Learned: Early User Acceptance Testing is critical! (earlier than we did it)
Next Steps

• More and more and more automation
  – Import & associate 3rd party components using composition analysis
  – Integrate with Vulnogram, CVE publishing (git), CERT/CC & other subscription-based info

• Enhance relational structures
  – More data attached to relationships between things
  – Expand the severity-risk model

• Improve user interface
  – Web 2.0 technologies – get UI in to the 2000s

• Give PSLs ownership of products in PAD

• Incorporate Threat Intelligence in to the tooling
  – Knowing what’s in our products allows more targeted TI
Definitions

• **Product**: The thing Lenovo sells; has a SKU, Part Number or Machine Type

• **Component**: The building blocks that make up the Product
  – Firmware (BIOS, Chipset, etc)
  – Drivers (graphics, audio, etc)
  – Applications (Lenovo System Update, xClarity products, etc)

• **Attribute**: Code that makes up the Component
  – 3rd party libraries
  – Open source libraries
  – Lenovo’s special sauce
# SLA: Service Level Adherence Proposal

<table>
<thead>
<tr>
<th>RISK: (Asset Criticality)</th>
<th>Vulnerability Severity (CVSS3.0)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical/Code Red</td>
</tr>
<tr>
<td>CVSS 9.0-10</td>
<td></td>
</tr>
<tr>
<td>CVSS 7-8.9</td>
<td></td>
</tr>
<tr>
<td>CVSS 4-6.9</td>
<td></td>
</tr>
<tr>
<td>CVSS .1-3.9</td>
<td></td>
</tr>
<tr>
<td>High</td>
<td>Priority 1</td>
</tr>
<tr>
<td>Medium</td>
<td>Priority 2</td>
</tr>
<tr>
<td>Low</td>
<td>Priority 3</td>
</tr>
</tbody>
</table>

Asset Criticality should be defined based on VOC and Lenovo reputational risk

Requirements:
- Support from ALIRT 2.0 PAD (SLA metrics)
- Brands/PSO(define asset criticality)
thanks.

Different is better