OPEN SOURCE IN SECURITY-CRITICAL ENVIRONMENTS

James Zemlin
Executive Director
Linux Foundation
@jzemlin
Open Source is here to stay in security critical environments and every place software is used
Linux has grown into the most important open source project in the world.

Every market Linux has entered it eventually dominates.

- 100% Supercomputer Market
- 82% Smartphone Market Share
- 2nd To Windows in Enterprise
- 90% Mainframe Customers
- 90% Public Cloud Workload
- 62% Embedded Systems Market
- #1 Internet Client
Linux Evolves Faster Than Ever

4,300
Contributors From 450 Organizations

2,000
Lines of Code Modified Daily

10,000
Lines of Code Added Daily

2,500
Lines of Code Removed Daily

8.5
Changes Per Hour
Open Source Development is Accelerating

- **23M+** Open Source Developers
- **78M+** Repositories on Github
- **41B+** Lines of Code
- **1,100** New Projects a Day
- **10,000+** New Versions per day

Sources: SourceClear, Sonatype, Github
It’s actually open source software that’s eating the world.

- Venturebeat 2015
Creating Applications these days is like making a sandwich
Code Club (Sandwich)
Choose a Framework
Code Club (Sandwich)

- Choose a Framework
- Write Custom Code
Code Club (Sandwich)

- Use Open Source Libraries to Solve Problems
- Write Custom Code
- Choose a Framework
Open Source Code = ~ 90%

---

Use Open Source Libraries to Solve Problems
Open Source Code (~70%)

Write Custom Code
Custom Code (~10%)

Choose a Framework
Open Source Code (~20%)
So much code – so little time

23M+
Open Source Developers

78M+
Repositories on Github

41B+
Lines of Code

1,100
New Projects a Day

10,000+
New Versions per day

Sources: SourceClear, Sonatype, Github
Open source isn’t slowing down any time soon
All this abundance has created anxiety
The real question is which projects matter?

Criticality of software vs. Number of Open Source Projects
How do we make important projects sustainable?

Successful Projects depend on members, developers, standards and infrastructure to develop products that the market will adopt.
WHEN THIS CYCLE WORKS, IT WORKS WELL
Value of Individual Project

- How to accelerate cloud native computing: devops, containers, microservices
- How to create a portability layer for cloud

Collective Action

- 2015 Google created CNCF with The Linux Foundation
- Project seeded with Kubernetes
- CNCF founded with 28 members
- Kubernetes de facto standard for container management
- 179 members, including all major public clouds and enterprise software vendors
- Home to 14 additional projects beyond Kubernetes
- 49 Kubernetes certified vendors
- Kubernetes surpasses OpenStack on Google trends

Results - 2018

Number of Open Source Projects – Millions on Github
SOMETIMES THE SYSTEM DOESN’T WORK
Questions to ask

- What is the most important and security critical shared software in the world?
- Who is creating and maintaining that software?
- Why are the creating and maintaining that software?
- Is it secure, reliable, and healthy?
Core Infrastructure Initiative Census Project

Lists of Projects to Analyze

Projects Popularity

Analysis Program

Project Data From Debian

Analysis Results Ranked By Risk Index

Expert Selection from Highest-Risk Projects

Most Concerning Projects

Project From openhub.net

Project Recent CVE Vulnerability Counts
Core Infrastructure Initiative Census Project

Analysis Program

Lists of Projects to Analyze

Projects Popularity

Project from openhub.net

Project Data from Debian

Project Recent CVE Vulnerability Counts

Analysis Results Ranked by Risk Index

Expert Selection from Highest-Risk Projects

Most Concerning Projects
Current Algorithm

- Project has website (1 if no)
- Written in C or C++ (2 if yes)
- CVE vulnerability reports: 3 points if 4+, 2 points for 2-3, 1 point for 1.
- 12 month contributor count: 5 points for 0 contributors, 4 points for 1-3 contributors, 2 points if the number is unknown.
- Top 10% most popular Debian package: 1 if yes

- Exposure values: 2 points if directly exposed to the network (as server or client), 1 point if it is often used to process data provided by a network, and 1 point if it could be used for local privilege escalation.
- Application data only: Subtract 3 points if the Debian database reports that it is “Application Data” or “Standalone Data” (not an application)
Tremendous Systemic Risks to the Internet Still Unaddressed

<table>
<thead>
<tr>
<th>Binary Package Name</th>
<th>Source Package Name (If Different)</th>
<th>CII 2016 Census Risk Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>ftp</td>
<td>netkit-ftp</td>
<td>11</td>
</tr>
<tr>
<td>netcat</td>
<td>netcat</td>
<td>11</td>
</tr>
<tr>
<td>tcpd</td>
<td>tcp-wrappers</td>
<td>11</td>
</tr>
<tr>
<td>whois</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>at</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>libwrap0</td>
<td>tcp-wrappers</td>
<td>10</td>
</tr>
<tr>
<td>traceroute</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>xauth</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>bzip2</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>hostname</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>libacl1</td>
<td>acl</td>
<td>9</td>
</tr>
<tr>
<td>libaudit0</td>
<td>audit</td>
<td>9</td>
</tr>
<tr>
<td>libbz2.1.0</td>
<td>bzip2</td>
<td>9</td>
</tr>
<tr>
<td>libept.1.4.12</td>
<td>libept</td>
<td>9</td>
</tr>
<tr>
<td>libreadline6</td>
<td>readline6</td>
<td>9</td>
</tr>
<tr>
<td>libtasn1-3</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>linux-base</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>telnet</td>
<td>netkit-telnet</td>
<td>9</td>
</tr>
</tbody>
</table>

The Big Risk:

Commonly used open source code and libraries are among the most at risk to cyber attacks or other potential threats that could bring down the global Internet.

Source: CII 2016 Census
A little love goes a long way

2014 - OpenSSL was maintained by two people and moribund
2016 – Recorded more activity than in the entire previous history of the project, including:

- Three new releases
- 3889 commits
- 481 GitHub users
- Thousands of forks.
- 1052 pull requests closed
- 47 CVEs reported and handled
How to create secure code?
We must secure the most critical open source software projects that power the world’s infrastructure, and to promote a culture of secure coding.
100 Projects Granted CII Best Practice Badge

- Initiative launched in May 2016 to raise awareness of development processes and governance steps for better security outcomes
- The badge makes it easier for users of open source projects to see which projects take security seriously, it isn’t a “rubber stamp” process
- 1,000 projects registered for the badge
Education

- One of the largest causes of security vulnerabilities is developers being unaware of security best practices.
- We need courses for open source developers for Security and Auditing.
- Organizations like SAFEC ode provide curriculum and training but we need more.
We need to be able to pass information about software bill of materials across the tech value chain in a simple and reliable way. You can’t fix bugs for code you don’t even know you have.
Software Tracking: The Challenge

companies combine Open Source Software with other software

Creating an accurate bill of materials and notices requires effort & research

Your code
OSS Package
OSS Package
Outsource SW
3rd party SW

Software Bill of Materials (BOM)
Software BOM: The Challenge

The effort is repeated at each step in the supply chain

Supplier 1
Supplier 2
Customers
“Open Source”-scape

Upstream Projects

Useful “Collections” of Open Source

Added-value Software

Products
Software Package Data eXchange

Open Standard:
- A standard format for communicating the licenses and copyrights and identity associated with software packages

Vision:
- To help reduce redundant work in determining software BOM information and facilitate compliance

Guiding principles:
- Human and machine readable
- Focus on capturing facts; avoid interpretations
What makes up an SPDX Document?

SPDX v2.1 Document contains:

- Document Creation Information
- Package Information
- File Information
- Snippet Information
- Other Licensing Information
- Relationships
- Annotations
Emerging “Between Organization” Trust Models

**Software Parts Ledger** - utilizes Blockchain to manage open source across the supply chain. Utilizes Hyperledger Sawtooth Platform & SPDX based BOM to conform to OpenChain best practices.

See: [https://github.com/Wind-River/sparts](https://github.com/Wind-River/sparts)

Accepted 2018/3 into Hyperledger Labs - [https://github.com/hyperledger-labs/hyperledger-labs.github.io/blob/master/labs/SParts.md](https://github.com/hyperledger-labs/hyperledger-labs.github.io/blob/master/labs/SParts.md)

**ClearlyDefined** - Announced 2018/3 - calls for participation in curating the metadata to summarize projects. See [ClearlyDefined.io](https://ClearlyDefined.io) for more information.
Sharing software bill of materials is critical part of security process

- OpenChain builds trust in open source by making sharing of software BOM simpler and more consistent
- Adobe, Arm, Cisco, Harmen, Hitachi, HPE, GitHub, Qualcomm, Siemens, Toyota, Wind River and Western Digital
Learn how open source software flows

Incorporation

A developer may copy portions of a FOSS component into your software product.

Relevant terms include:
- Integrating
- Merging
- Pasting
- Adapting
- Inserting
Get a process in place

Working through the FOSS Review

The FOSS Review process crosses disciplines, including engineering, business, and legal teams. It should be interactive to ensure all those groups correctly understand the issues and can create clear, shared guidance.
We need to invest in tools that test upstream code
Frama-C False-Positive-Free Checking

- Frama-C is a highly respected static checker
- When used with test cases and modified Unix standard functions, it is able to detect bugs without false positives
- Proposal is to modify several standard Unix functions to support false-positive-free operation on OpenSSL
- In addition, the proposal is to use the American Fuzzy Lop fuzzer to automatically generate test cases from which Frama-C can detect bugs
Fuzzing

- [Link](https://fuzzing-project.org/) is Hanno Böck’s project
  - Uses zzuf, Address Sanitizer and American Fuzzy Lop to find bugs in open source projects
  - Discovered numerous GnuPG bugs in Feb 2015
  - He and others have found numerous bugs in many projects: [http://lcamtuf.coredump.cx/afl/#bugs](http://lcamtuf.coredump.cx/afl/#bugs)

- His main activity is to convert the fuzzer output into reproducible test cases and file bugs for them
- He is also doing great work training new developers to become expert fuzzers
- CII is also reaching out to fuzzing toolkit authors
Reproducible Builds

- Debian and Fedora rely on package maintainers to compile source code from the upstream authors.

- Because the resulting binaries depend on machine configuration (like timestamps and file ordering), these binaries are not reproducible.

- That makes it impossible to independently verify that the binaries have not been tampered with.

- Binary reproducibility should become an expected attribute of free software distros.
We need to invest in audit of upstream open source code for critical shared infrastructure
Auditing: Many critical open source projects do not have resources to audit

- Auditing finds critical bugs that won’t be found any other way
- Auditing is expensive, time consuming and only finds a subset of the bugs so it can’t be the only tool
- OpenSSL audit underway
How to get involved?
Follow up material

- See Linux Foundation-sponsored Institute for Defense Analysis (IDA report, "Open Source Software Projects Needing Security Investments")
- Some of the projects we're most concerned about (because they are ubiquitously deployed and could result in Heartbleed-style vulnerabilities) include compression libraries (bzip2, gzip, unzip, zlib) and format libraries (libjpeg, libpng, and expat)
- Unlike before Heartbleed, there is actually a group focused on these issues. Two major programs we’re undertaking with IDA:
  - CII is not only reactively looking for broken projects (i.e., fighting fires) through our Census Project
  - We are also developing the building codes (in terms of security best practices) to avoid fires in the future