THE LANGUAGE OF SECURITY IS RISK

http://xkcd.com/795/

Security  Product Management

THE ANNUAL DEATH RATE AMONG PEOPLE WHO KNOW THAT STATISTIC IS ONE IN SIX.
WHAT IS RISK?
WHAT DOES THE SNAIL TELL US?

“...WE OWE A DUTY OF REASONABLE CARE TO OUR NEIGHBOR”

Lord Atkin: Donoghue v. Stevenson (1932)
You Built It, Your Responsible

“…a manufacturer of products, which he sells in such a form as to show that he intends them to reach the ultimate consumer in the form in which they left him with no reasonable possibility of intermediate examination, and with knowledge that the absence of reasonable care in the preparation or putting up of products will result in an injury to the consumer's life or property, owes a duty to the consumer to take that reasonable care.”
WHAT IS RISK?

*United States v. Carroll Towing Co.*
159 F.2d 169 (2d. Cir. 1947)
THE COST OF DOING NOTHING CAN’T BE IGNORED

“...IF THE PROBABILITY BE CALLED P; THE INJURY, L; AND THE BURDEN, B; LIABILITY DEPENDS UPON WHETHER B IS LESS THAN L MULTIPLIED BY P: I.E., WHETHER B < PL”.

Translation: If the Cost of Protecting Against Harm is less than the Cost of the Damage Multiplied by the Likelihood of the Damage, then there is negligence.

Risk = probability x impact
Modern software development

HAS CHANGED

Application security

HASN’T CHANGED ENOUGH
YOUR JOB DESCRIPTION HAS BEEN FUNDAMENTALLY RE-WRITTEN.

Applications are the new vector of attack. Development is going faster than security can keep up. Most source code has been replaced by open source components.

Did you know that 90% of a typical application is comprised of open source components which are assembled together like LEGO® building blocks?
 QUESTION:
 IS APPLICATION SECURITY BROKEN?

Security is bolted-on, not built-in.
Releases are monthly, weekly, or even daily. Security can’t keep up.
Software is assembled with components, yet we can’t really see what we’re using.
We build known vulnerabilities into our software, then spend even more time and resources to get them back out.
Open source usage is EXPLODING

Yesterday’s source code is today’s OPEN SOURCE
Creating today’s software SUPPLY CHAIN
Do you know who your SUPPLIERS ARE?

- COMPONENT SELECTION
- DEVELOPMENT
- BUILD AND DEPLOY
- PRODUCTION

Sonatype
CAN WE ALL AGREE?
THIS IS JUST NOT WORKING!

We scan source code.
We manually enforce whitelists and blacklists.
We (think we) have golden repositories.

All tickets on the things-we-think-we-should-do-to-be-competent train.

But your developers find work-arounds...
Cyber attacks are on the rise...
And software is still not secure...
THE FACTS: THIS IS NOT AN OPEN SOURCE PROBLEM.

This is productivity exceeding security.

Open Source Software (OSS) is essential in our world today. Without it, we couldn’t build our innovative, profit-making products or awesome new services quickly and reliably.

Trusting in open source is fine. But blind trust isn’t.
Components are like MOLECULES not atoms. There are massive dependencies.

**Complexity**
One component may rely on 100s of others

**Diversity**
- 40,000 Projects
- 200M Classes
- 400K Components

**Volume**
Typical enterprise consumes 1,000s of components monthly

**Change**
Typical component is updated 4X per year

Source: Sonatype, Inc. analysis of (Maven) Central Repository.
Typical component is updated 4X per year.

Change
Typical component is updated 4X per year

674,863 OSS COMPONENTS

11 MILLION OSS USERS

Source: Components: (Maven) Central Repository; Users: IDC
Typical component is updated 4X per year.

Unlike COTS, there is no clear, effective communication channel.

- Has a risk been identified?
- What type of risk?
- Is a better version available?
Manual processes DON’T WORK

Automation should ENFORCE POLICIES

Humans should manage exceptions
WHAT ME WORRY, TIS’ JUST A BIT OF FLOATING ICE
Components are a hidden risk
Spending and risk are **OUT OF SYNC**

### SECURITY SPENDING
- **People Security** $4B
- **Data Security** $5B
- **Host** $10B
- **Network Infrastructure** $20B

### APPLICATION MAKEUP
- **Presentation Layer**
  - Business Logic
- **Application Layer**
  - Database
  - Operating System
  - Firmware
  - Network

### RISK OF ATTACK
- **90%**

---

Risk of Attack Source: Ponemon Institute; Security spending numbers are approximate
If you’re not using secure COMPONENTS you’re not building secure APPLICATIONS
Today’s approaches AREN’T WORKING

- 46m vulnerable components downloaded
- 90% of repositories have 1+ critical vulnerability
- 71% of apps have 1+ critical or severe vulnerability

- COMPONENT SELECTION
- DEVELOPMENT
- BUILD AND DEPLOY
- PRODUCTION
LET’S GET MORE SPECIFIC.
EVEN AFTER SECURITY ALERTS WERE ISSUED AND FIXES PROVIDED…

there were 150,508 downloads of BOUNCY CASTLE

with an exploitability score of 10.

there were 70,665 downloads of APACHE STRUTS 2

with an exploitability score of 10.

there were 2,167,625 downloads of HTTP CLIENT

with an exploitability score of 8.6.

Hmmm…that’s a lot of sour components flowing into your applications. And fresher versions have been available for years!

Source: Sonatype analysis of downloads from (Maven) Central Repository and NIST CVE database
THE ANTI-PATTERNS
TURN OFF THE LIGHTS
LOCK THE DOORS
POINT FINGERS
THESE ARE NOT MY DROIDS
EVERYTHING IS A NAIL

To a man with a hammer, everything looks like a nail.
Applications don’t age, they rot like milk.
Time for a **FRESH APPROACH?**

**CURRENT METHODS**

- Problem discovery
  - “Scan and scold”
- Source code scanning
- Approval-centric workflow
- Scans after development
THE PROBLEM IS NOT PROBLEM
DISCOVERY
PROBLEM DISCOVERY WORKS FOR THIS
Time for a FRESH APPROACH?

To fix it, you must first find it. Time to make CVE information actionable.

Proprietary 4 Point CVE Curation Process

2 - 4 hours per component

1. Identify precise at-risk components
2. Understand hidden component dependencies
3. Expand research to find other impacted components
4. Include context* (when is the issue relevant?)

Enhanced, actionable component intelligence in CLM

* Coming soon
Got questions?

Get the ANSWERS.

? What production applications are at risk?

? What problems are most critical?

? What components are being used?
   Where are they?

? Which components have known security vulnerabilities?

? What are our license obligations?

? Do our applications comply with our policies?

? How can we choose the best components from the start?
BUILDING A BETTER BRIDGE BETWEEN DEV, OPS AND SECURITY

- Need to recognize that the priorities are different
- Tooling needs to adopt the practice of the practitioner not the other way around
- A Tool is not a process and a process is not a tool learn to leverage both.
Take the Open Source Impact CHALLENGE!

Get answers FAST! Contact us at www.sonatype.com/answers

What production applications are at risk?

What problems should you address first?

What components are being used?
  Where are they?

Which components have known security vulnerabilities?

What license obligations do you have?

Do your applications comply with your policies?

How can you choose better components from the start?