Risks of Free Open Source Software and S-SDLC

Managing The Risk

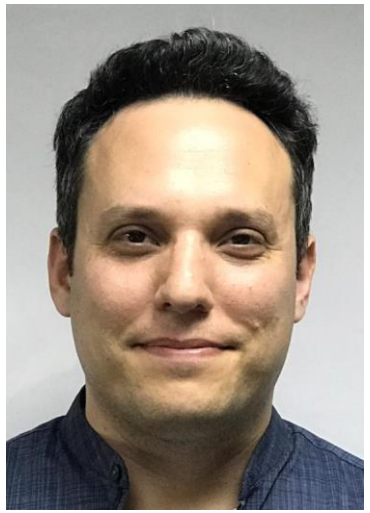
Lior Mazor
SecureDEV team Manager at Amdocs
CISM, CISSP
13/08/2020

amdocs

About Myself

Lior Mazor

CISSP, CISM

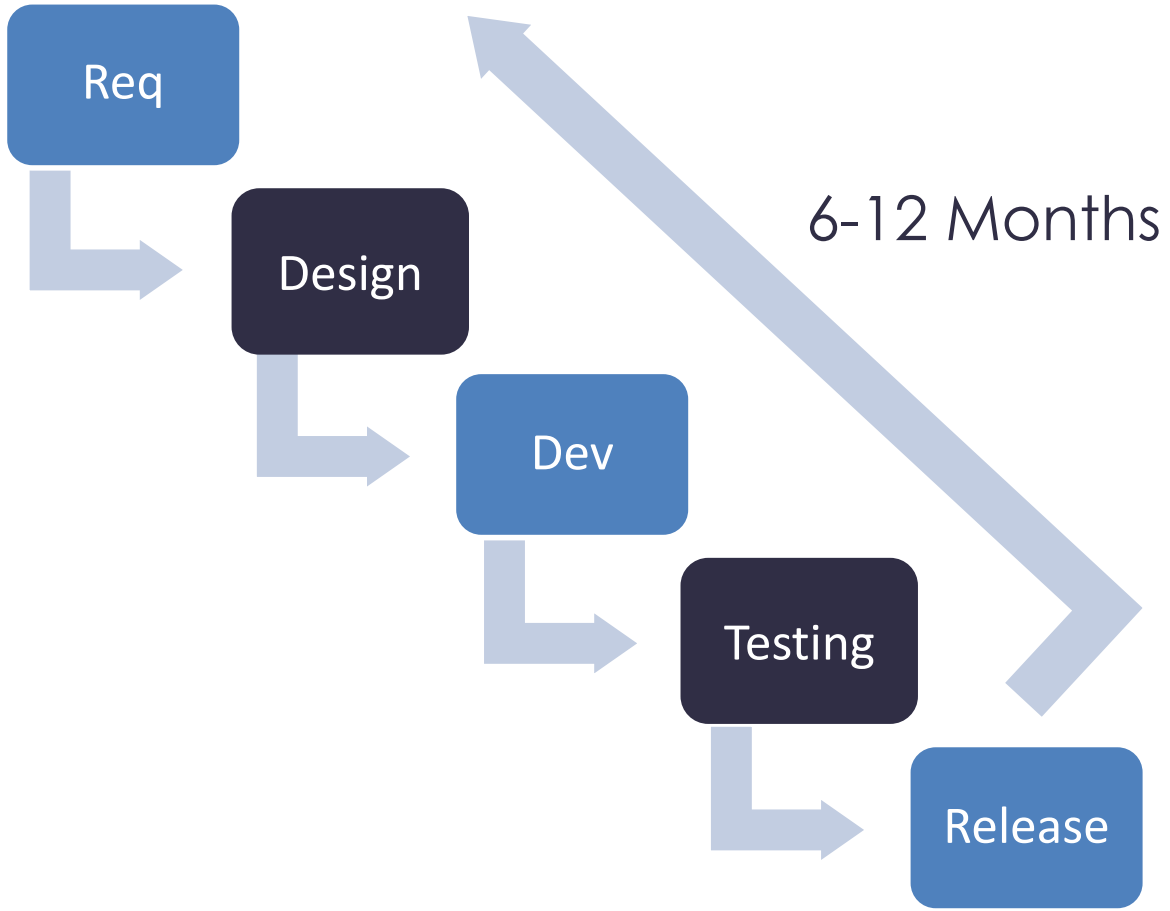


- Leading and Managing the S-SDLC and information product security at Amdocs
- B.Sc. Math & Computer Science
- Over 15 years of cybersecurity experience
- Implementing and promoting Information Security & Cyber Awareness in various enterprise organizations across the world
- Building Enterprise Cyber Defense Methodologies and ‘Execution Practices’
- Expertise in Secure Development, Code Review & Application Penetration Testing
- Certified as ISO 27001 Lead Auditor

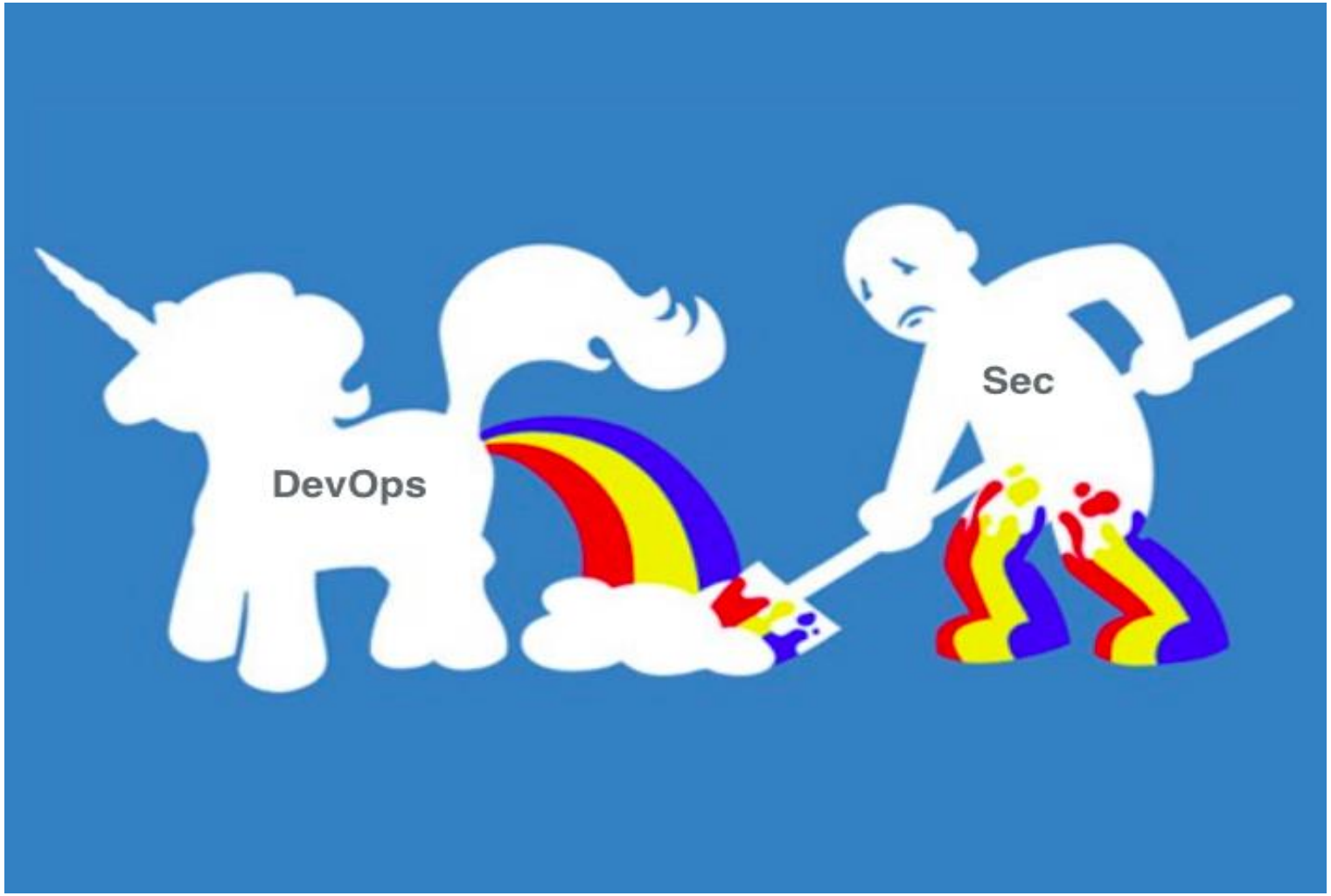
Agenda

- Waterfall VS Agile and DevOps development
- Secure Software Development Life Cycle (S-SDLC)
- The risk of Open Source
- How to manage the risk of Open Source

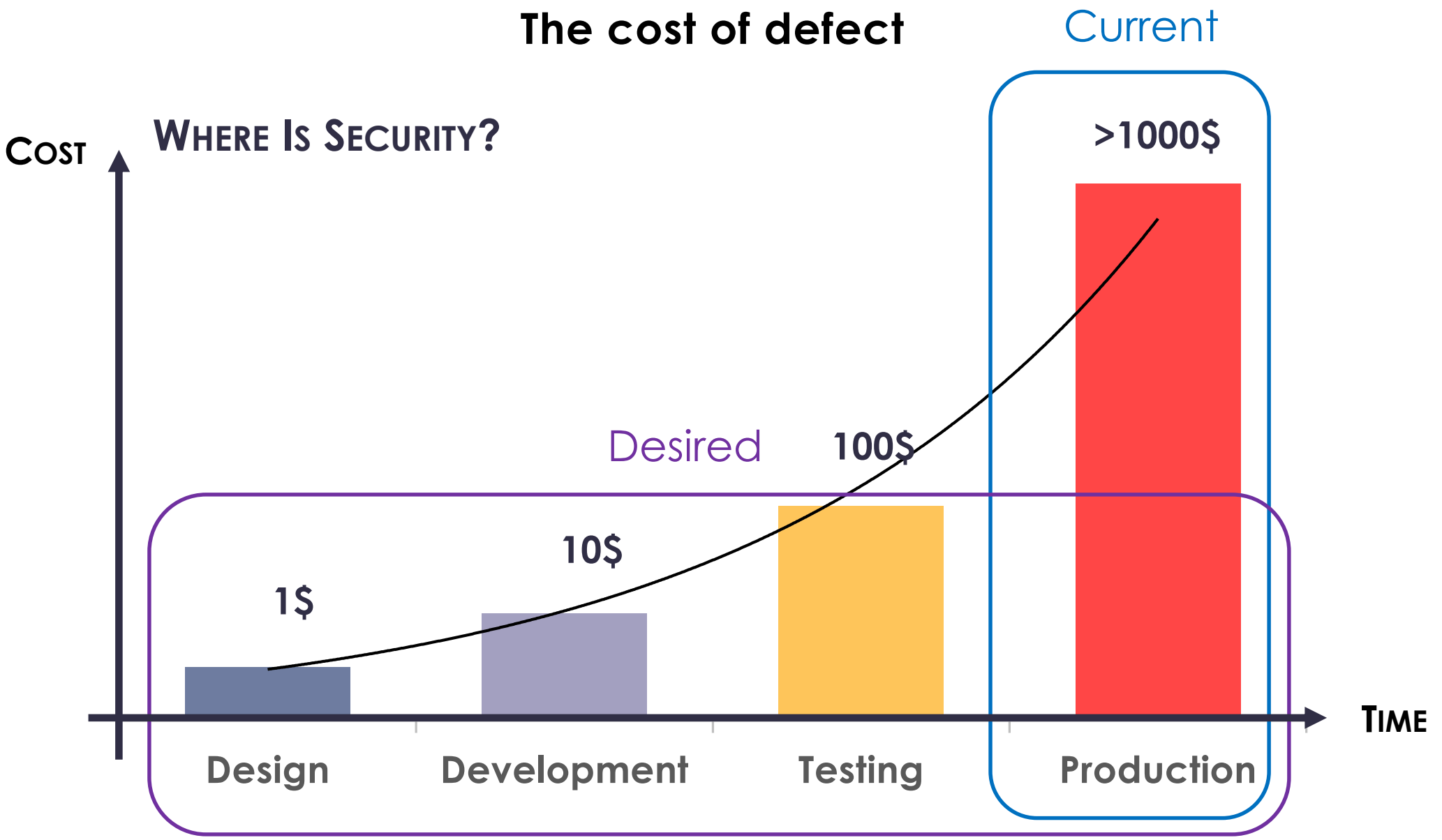
Waterfall Development



We Want DevOps (AGILE)!!!



The cost of defect

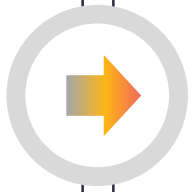


Secure Software Development Life Cycle (S-SDLC)

What is S-SDLC?

The S-SDLC process integrates into the overall development process and its objectives are:

- Reduce the number of vulnerabilities and weaknesses
- Improve the level of information security
- Identify information security weaknesses at earlier stages in the development process
- Define the construction, guidelines, knowledge, tools, policies and testing requirements



The results of these goals should be:

- Reducing / preventing damage caused by cyber attacks
- Reducing the costs of remediating information security weaknesses
- Fewer remaining vulnerabilities before production, therefore reducing delay due to security

S-SDLC Framework

The S-SDLC activities are organized into a framework and categorized to 5 domains:

Governance

Manage and measure the S-SDLC Process

Construction

Build Software/ System security **knowledge**

Verification

Test artifacts produced throughout development

Deployment

Manage the **release and the maintenance**

Supporting tools & Utilities

Security services that provide assistance and support in the S-SDLC Process to development departments

Secure Development vs Secure By Design

Secure Development

Goal:

Ensuring that product development will apply security from early stage to release, allowing the product posture to be secure from potential vulnerabilities that can be exploited given wrong development practices.

How to Obtain it?

- S-SDLC
- Secure code Automation (Static Code Analysis)
- FOSS Scanning
- Pen Testing
- Bug Bounty

Secure by Design

Goal:

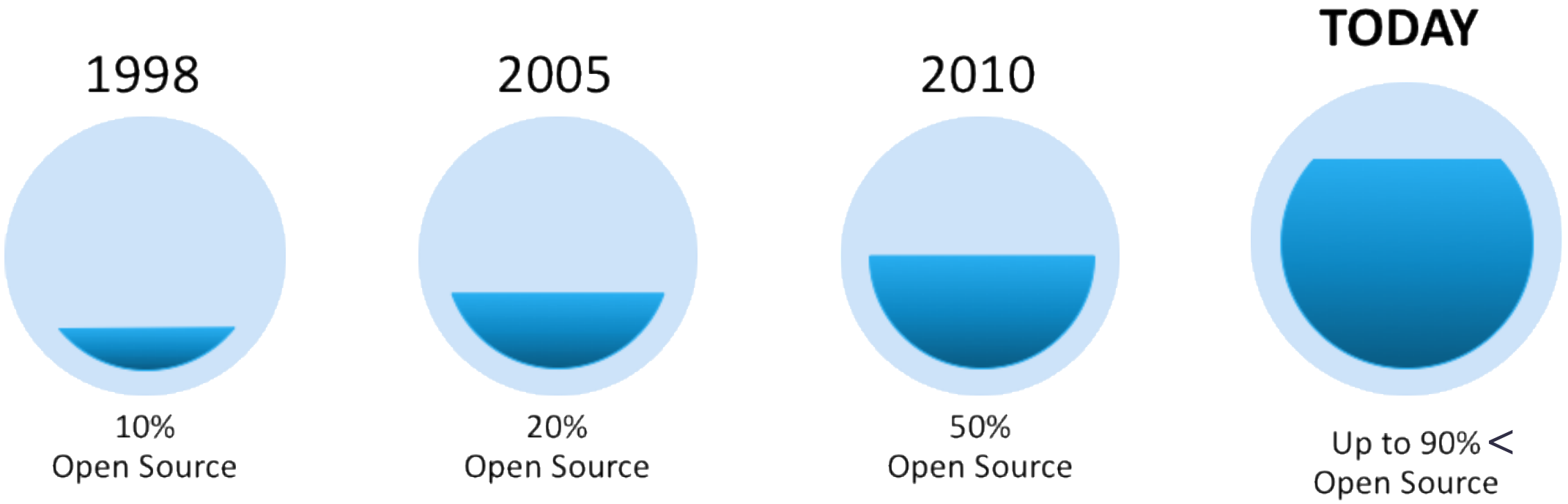
Embedding OOTB security practices to enable core security controls to minimize gaps from failed secure development practices & reduction of potential security threats and stronger compliance with customer needs

Examples:

- Secure file upload
- Cryptography
- Enhanced Logging
- OOTBH Input validation
- ABAC, RBAC…
- FOSS Policy

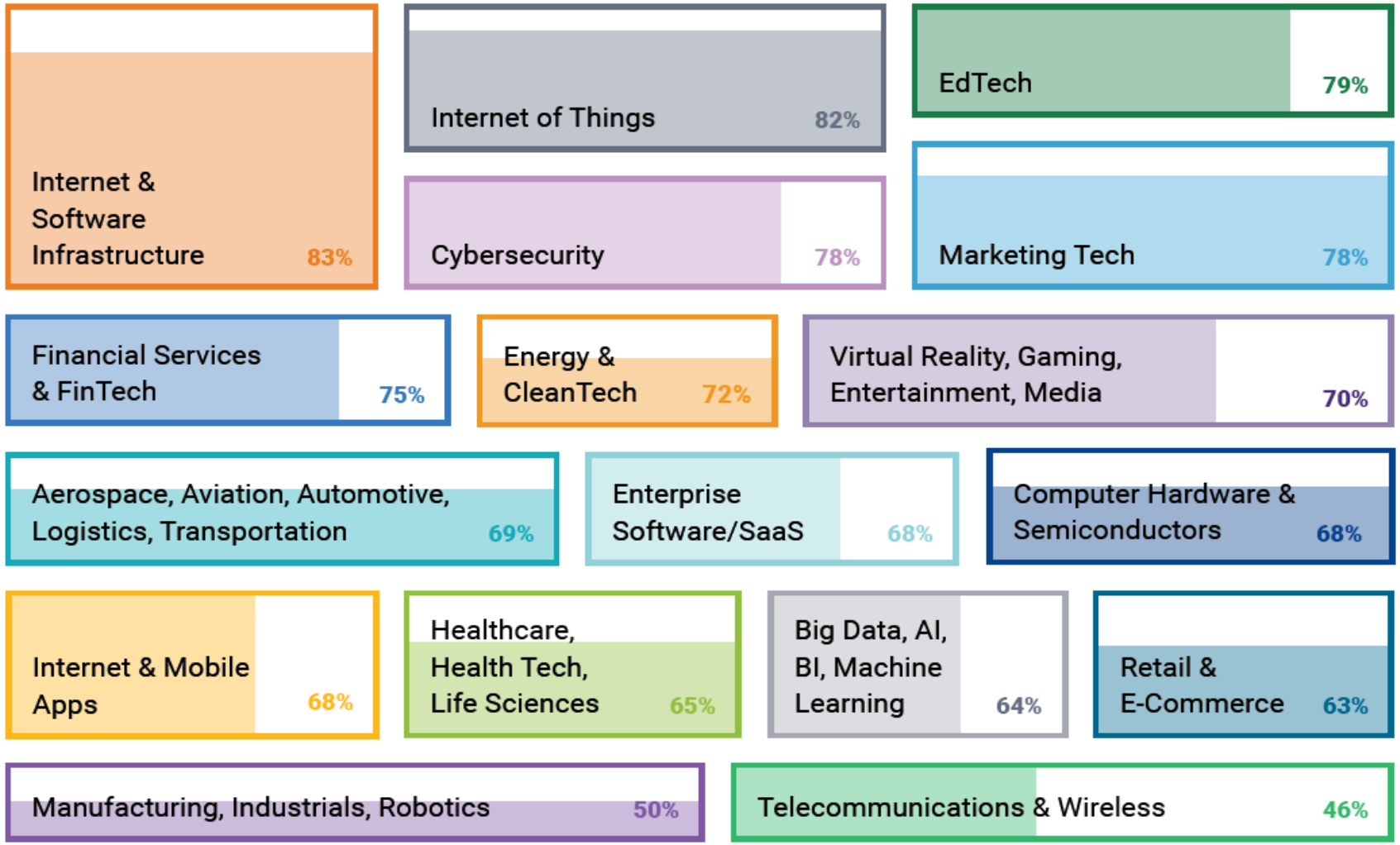
The risk of Open Source?

Open Source in Numbers



* source: 2020 OSSRA Report, Black Duck by Synopsys

Open Source by industry



* source: 2020 OSSRA Report, Black Duck by Synopsys

Open Source in Numbers

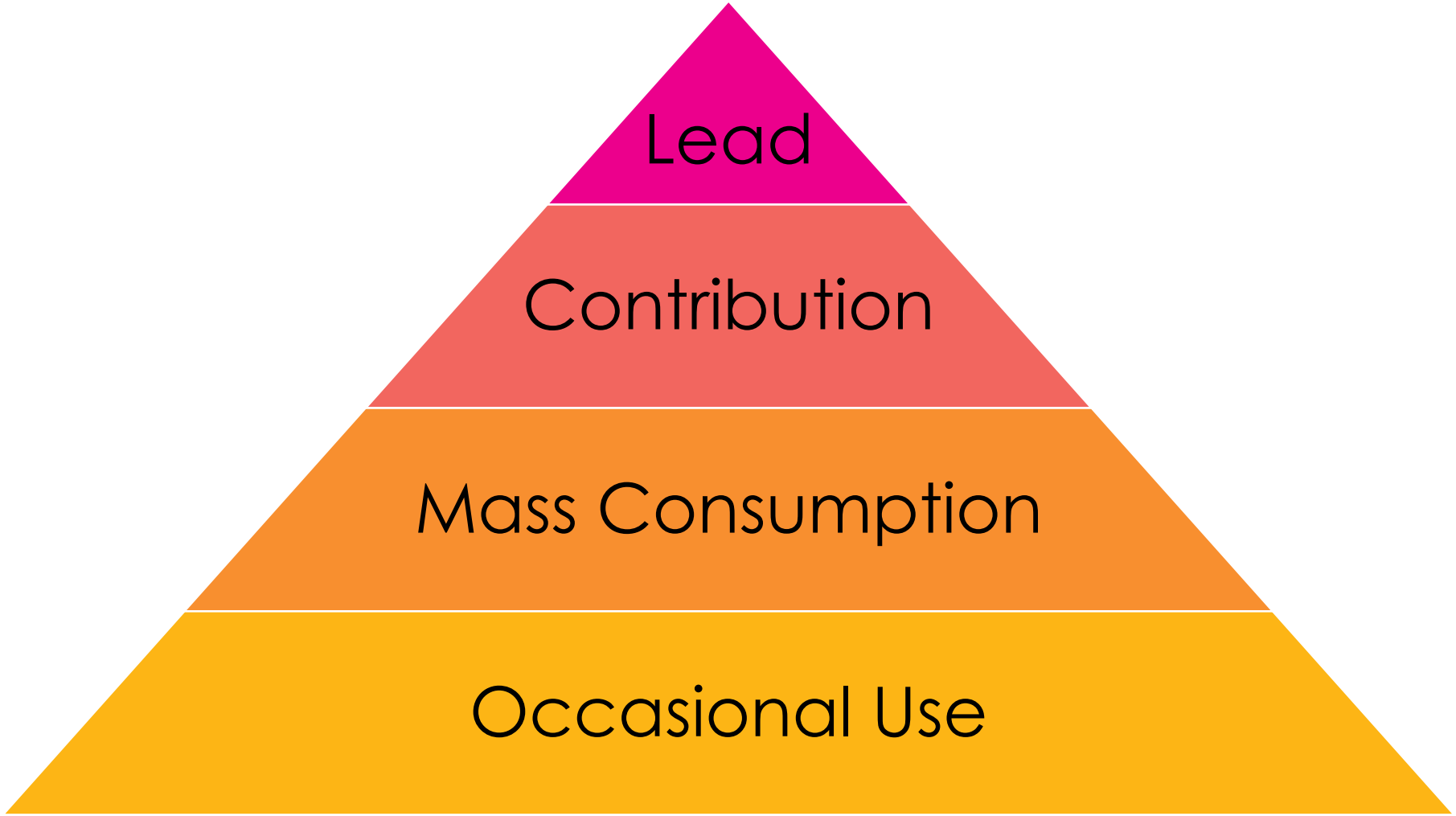
96% of applications scanned include open source components. Average is 445 components per application

Average percentage of codebase that is open source is 70% vs. 60% last year

78% of codebases contain at least one vulnerability with an average of 82 vulnerabilities per codebase (49% of the audited codebases contained high-risk vulnerabilities)

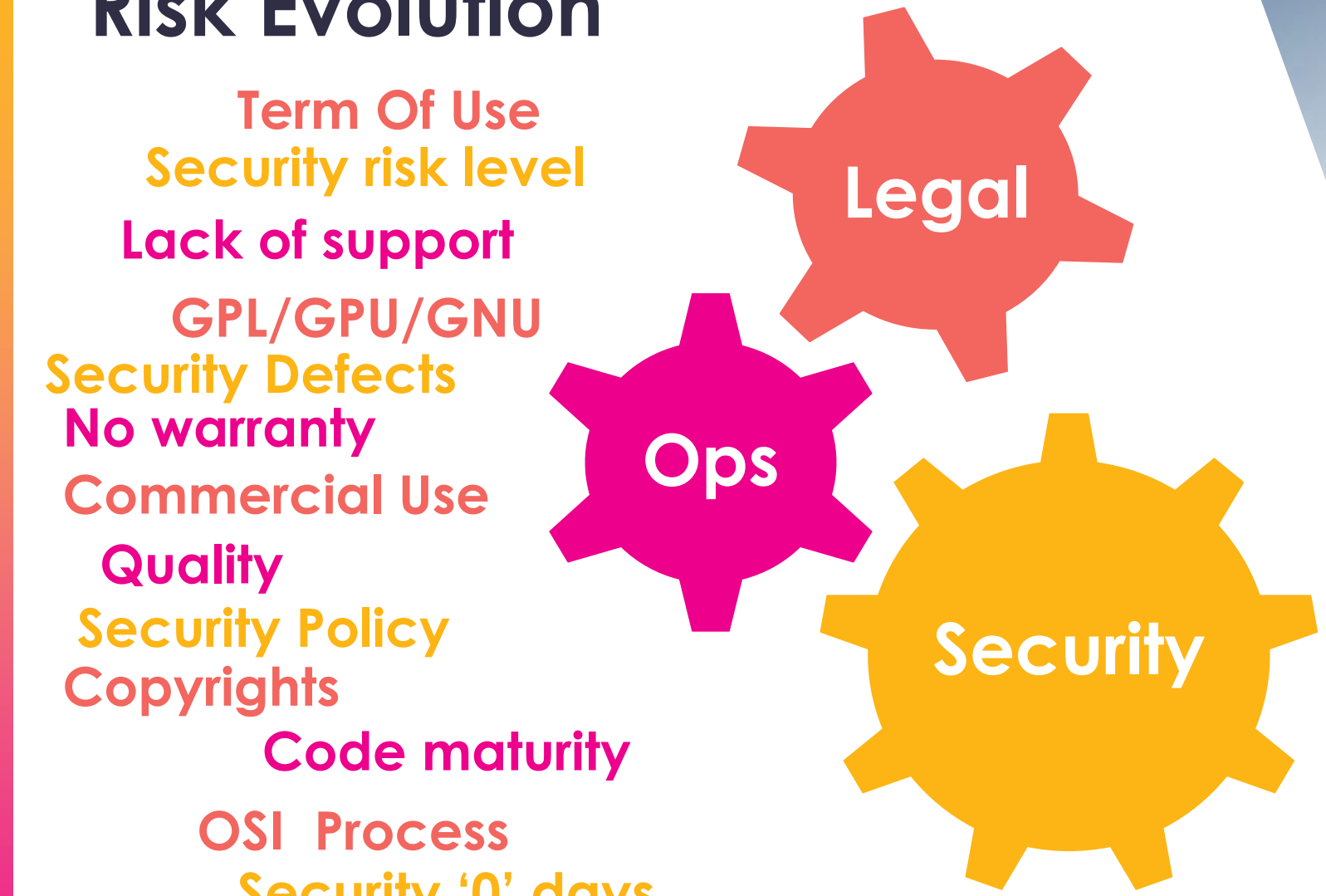
* source: 2020 OSSRA Report, Black Duck by Synopsys

Evolution of Open Source Adoption in Enterprises



Risk Evolution

Term Of Use
Security risk level
Lack of support
GPL/GPU/GNU
Security Defects
No warranty
Commercial Use
Quality
Security Policy
Copyrights
Code maturity
OSI Process
Security '0' days
Common use of FOSS



How to manage the risk of Open Source?

Open Source Management

The need to automate open source management

- **MANAGE** all projects in a systematic way
- **GUIDE** the development teams on policy and resolutions
- **MONITOR** compliance and security, automatically

Building the Right Processes

Clear policies for license compliance (Legal) and security vulnerabilities

Ensure all products/distributions are scanned

Scanning as a service across the organization

Ensure all identified issues are handled before distribution

- Open defects for FOSS security vulnerabilities
- FOSS Governance: '0' Critical/High defects before release

Choosing the Right Tool

Accuracy of detection

Minimize analysis and false positives (Runtime scanning)

Libraries vs. snippets

Suggest remediation options for security vulnerabilities

Provide alerts for new vulnerabilities

Integration with CI/CD


Thank you