Security Debt in EMEA

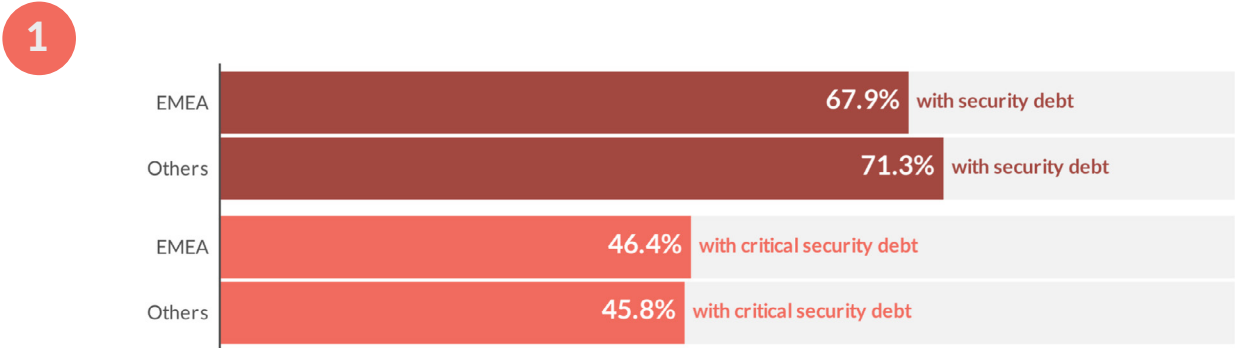**2024 EMEA Snapshot of the State of Software Security**

Veracode's State of Software Security 2024 spotlights the pressing issue of security debt in today's applications and provides a wake-up call to organizations worldwide. In particular, it seeks to answer questions such as "How risky is security debt really? How should it be tackled? And what's the best way to do it?" Whereas the main report analyzes security debt collectively across all organizations, this snapshot focuses exclusively on findings for Europe, the Middle East, and Africa (EMEA).

The demand for speed and innovation in software development has resulted in the accumulation of risk known as security debt. For the sake of consistency in this analysis, security debt is defined as all flaws that persist unremediated for longer than a year. Overall, we found flaws exceeding that threshold in 71% of organizations, making security debt far from a rare phenomenon.

The EMEA region is roughly on par with the overall ratio, with 68% of organizations exhibiting some level of security debt across their applications. Even more concerning, 46% of EMEA entities have high-severity persistent flaws that we'll classify as critical security debt. These flaws represent the highest risk to applications and thus warrant priority remediation. We'll return to that notion at the end of this snapshot.

Figure 1: Prevalence of security debt in EMEA

1

|	
EMEA	67.9% with security debt
Others	71.3% with security debt
EMEA	46.4% with critical security debt
Others	45.8% with critical security debt

It's also enlightening to break down security debt at the application level in Figure 2. Here we see that 36.1% of all applications across EMEA have debt, which is better than the overall rate of 42%. Furthermore, just 5.6% of applications are flaw-free, compared to 5.9% across other industries.

Figure 2: Prevalence of security debt across all applications greater than one year old[1]

2

Apps with no flaws (5.6%)   Apps with flaws but no debt (58.2%)   Apps with security debt (36.1%)
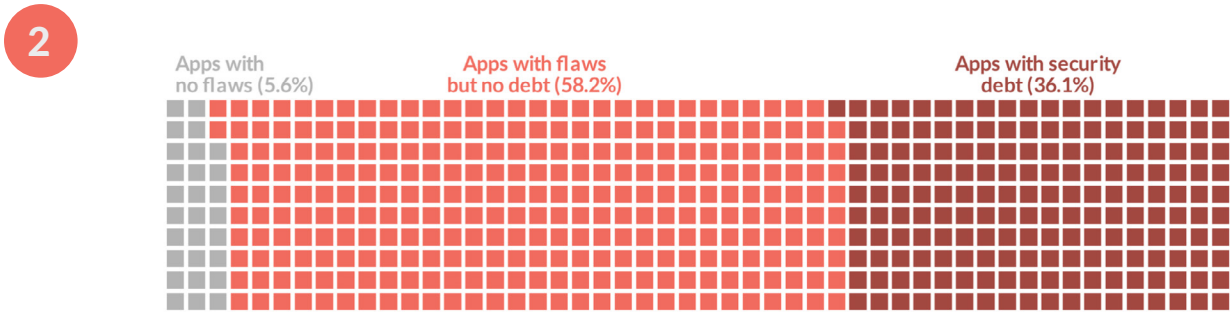
Does security debt exist in the code written by developers or in the third-party code they import into the codebase? The answer is a resounding "both." Specific to the EMEA region, in Figure 3, we observe that most (84%) of all security debt affects first-party code. But the majority of critical security debt comes from third-party dependencies. At almost 80%, it's higher than the the overall rate of 65%. Thus, focusing on both first and third-party code is necessary to drive down security debt within organizations.

Figure 3: Proportion of security debt and critical debt in first-party vs. third-party code

3

Third Party Code   First Party Code

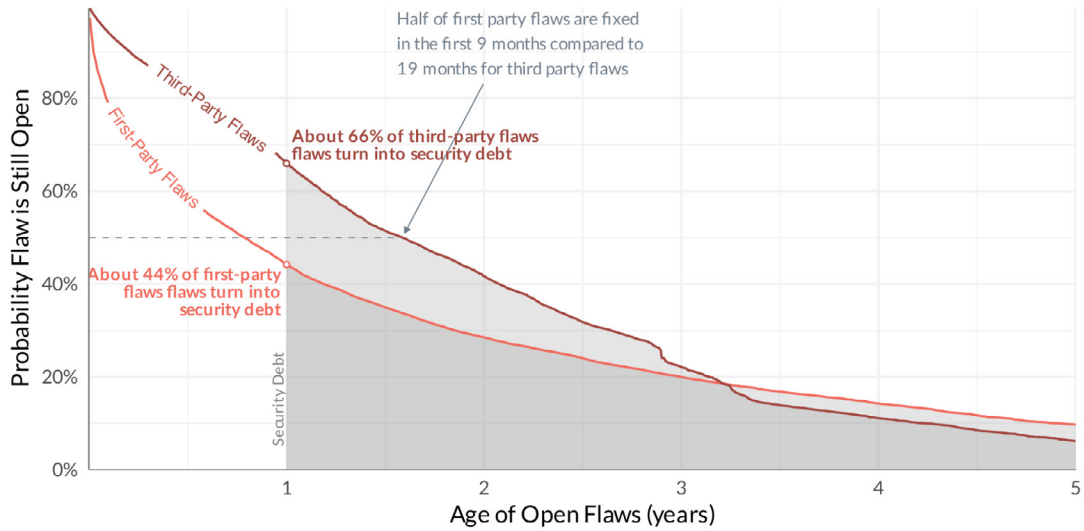|	Third Party	First Party
All Security Debt	16.0%	84.0%
Critical Security Debt	79.8%	20.2%

Percent of Flaws
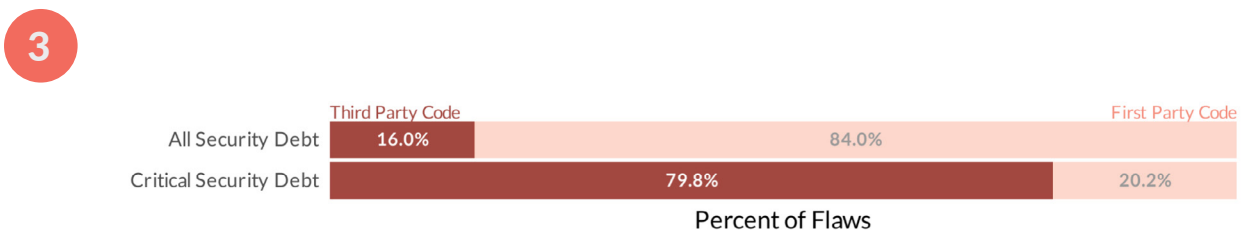
[1]This is based on applications that have been scanned for at least one year, since others haven't had the opportunity to accrue debt.

Now that we've looked at the prevalence of security debt in first and third-party code, let's see how long it takes to fix them. Analysis of remediation timelines in EMEA reveals that the half-life of flaws in third-party code is 19 months, compared to 9 months in first-party code. Ideally, we'd like to see the flaw survival curves in Figure 4 dive down sharply, with a very low percentage that becomes security debt at the one-year mark. Unfortunately, that's not the reality shown here.

Figure 4: Comparison of remediation timelines for first versus third-party flaws

Half of first party flaws are fixed in the first 9 months compared to 19 months for third party flaws

About 66% of third-party flaws flaws turn into security debt

About 44% of first-party flaws flaws turn into security debt

Third-Party Flaws

First-Party Flaws

Security Debt
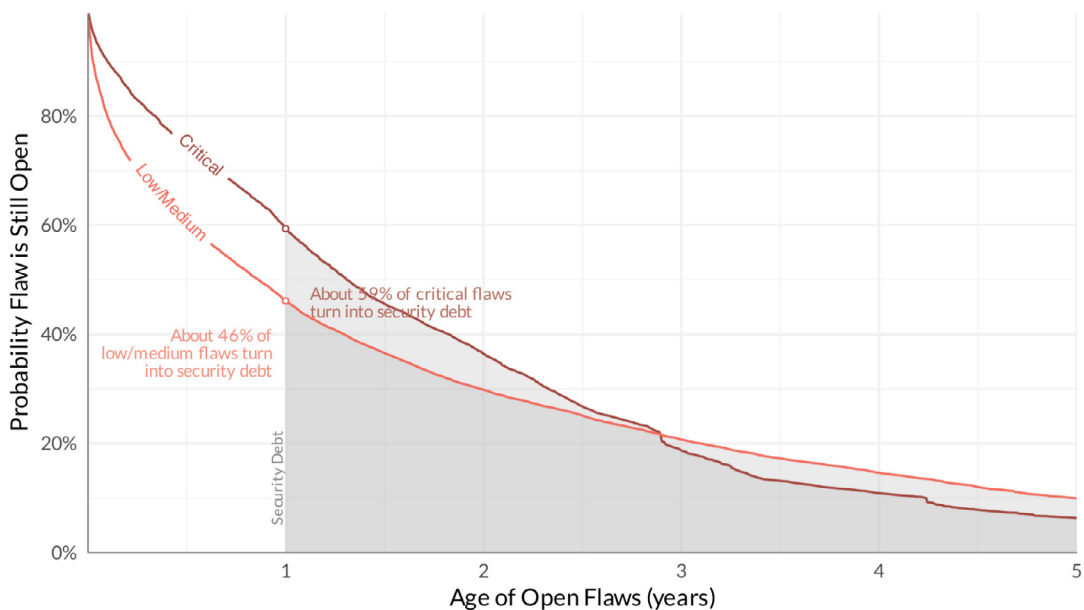
Probability Flaw is Still Open

Age of Open Flaws (years)

Speaking of how long it takes to fix things, let's focus on the first-party flaws and the criticality of the findings. Figure 5 looks at the rate of remediation for those findings and the proportion of them expected to reach our definition of security debt within the EMEA region.
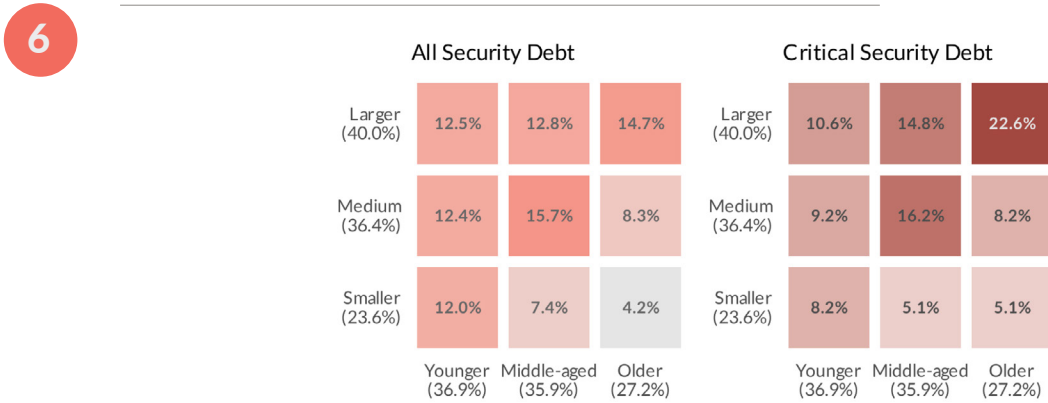
Figure 5: Comparison of remediation timelines for lower-severity vs. critical flaws

Critical

Low/Medium

About 59% of critical flaws turn into security debt

About 46% of low/medium flaws turn into security debt

Security Debt

Probability Flaw is Still Open

Age of Open Flaws (years)

The codebase of most applications grows over time, so it's logical that a correlation might exist between age and the accumulation of security debt. To test this, we grouped applications into three roughly equal bins according to their relative age and size. Per Figure 6, security debt in the EMEA region tends to concentrate away from smaller applications, and towards older applications. This is especially true for critical security debt. That's not terribly surprising, but it does confirm the correlation and point remediation efforts in the right direction.

Figure 6: Distribution of security debt across application age and size

6

All Security Debt				Critical Security Debt		
Larger (40.0%)	12.5%	12.8%	14.7%	Larger (40.0%)	10.6%	14.8%	22.6%
Medium (36.4%)	12.4%	15.7%	8.3%	Medium (36.4%)	9.2%	16.2%	8.2%
Smaller (23.6%)	12.0%	7.4%	4.2%	Smaller (23.6%)	8.2%	5.1%	5.1%
	Younger (36.9%)	Middle-aged (35.9%)	Older (27.2%)		Younger (36.9%)	Middle-aged (35.9%)	Older (27.2%)

In prior SOSS versions, we've shown that the language used to develop applications has a strong bearing on security outcomes across the board (prevalence, flaw types, remediation times, etc.). It seems like a foregone conclusion, then, that languages would have a lot to do with an application's debt density, too.

Figure 7 compares the security debt profile for different development languages based on the most recent scans in EMEA organizations. It does this by plotting the proportion of applications (x-axis) and flaws (y-axis) that exhibit any security debt (left) and critical debt (right). The size of the dot corresponds to the number of applications for each language. From this, we see a mix of languages being the source of debt in EMEA.

Figure 7: Prevalence of security debt by application development language

7
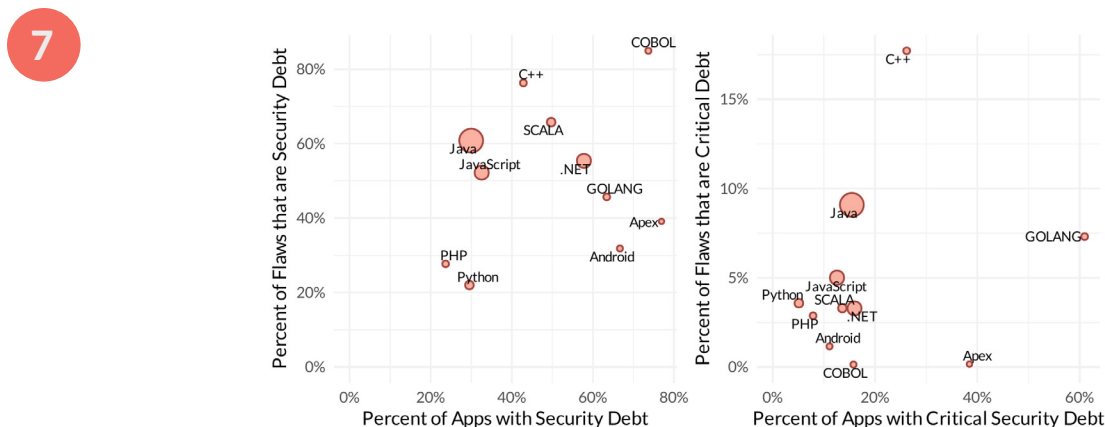
Finally, let's establish a baseline target for how many flaws organizations would need to remediate to keep critical security debt from piling up across applications. Figure 8 carves up all flaws identified across the entire codebase of the typical organization in EMEA into four quadrants. The horizontal separation is between critical and non-critical flaws. The vertical axis divides flaws that are currently debt vs. those that aren't (yet).

Three-fifths (60%) of all flaws in EMEA organizations are neither debt nor critical in severity. We're not saying ignore them altogether (or they'll eventually become debt), but remediation of those flaws can be deferred in favor of those that represent greater risk. Instead, focus development teams on fixing the 4.3% of flaws that constitute critical debt. Once that's done, organizations can tackle critical flaws or non-critical debt based on their risk tolerance and capabilities.

Figure 8: Distribution of all flaws based on severity rating and security debt status

8
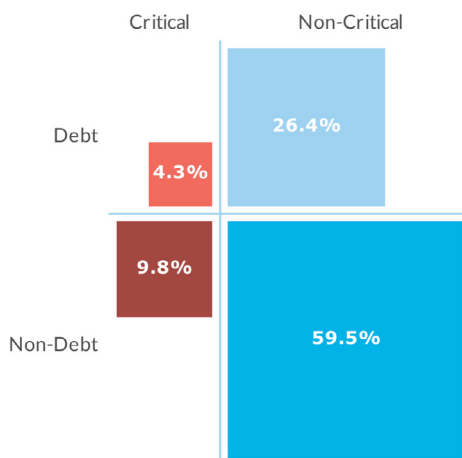
The good news is that the debt reduction strategy we've described here is realistically achievable. Most organizations have the capacity to remediate all critical debt and start driving down risk in the other quadrants in Figure 8. We hope this snapshot has provided useful information to help your organization accomplish that goal.

Download the full State of Software Security 2024 here:

https://www.veracode.com/state-software-security-2024-report