CASE STUDY

Tournament Torque:
ASRG's push for automotive software security

The **Automotive Security Research Group** is a non-profit organization dedicated to raising awareness and supporting the development of security for the automotive industry, with a focus on finding and promoting solutions that make automotive products safer and more secure. At the moment, this industry is changing and going through the next revolution to connected, autonomous, shared, electrified and software-defined vehicles. As we head into this brave new technological world for the industry, with an enormous increasing dependence on software powering vehicles and ecosystem applications, organizations like ASRG play a key role in drawing attention to and upholding software security within the automotive industry.

Let's shine a light on the state of the industry, the immediate issues we need to address, and the real stats from hundreds of challenges played on the Secure Code Warrior platform.

This awareness, and most importantly, manufacturers acting upon it, will be crucial as potential attack vectors and cyber risk expands with increased adoption of new vehicle technology in the consumer market. The FBI recently warned of **attackers targeting the US auto industry,** with a large majority of breaches the result of unencrypted sensitive data. This, in addition to attacks like brute-forcing poorly configured databases, could spell huge and potentially lethal consequences.

As part of their research into solutions and tools that help shape and uphold software security standards in automotive products, the team at ASRG took Secure Code Warrior's platform for a test drive, namely the **Tournaments** function. Purpose-built to engage developers through friendly, gamified competition, increase their security awareness and help hone their secure coding skills, ASRG looked into how Secure Code Warrior could spark developer interest in security, teach them the skills to stop common vulnerabilities that affect automotive software and open the floodgates to unacceptable risk.

Where are the typical attack vectors in automotive software?

When analyzing the potential avenues for attackers to access automotive software, there are many possibilities, as detailed by Allot's comprehensive report.

No matter how security-aware, developers cannot help to defend against them all (nor should they be expected to – AppSec specialists exist for a reason!) but there are plenty of common back doors that savvy engineers can close in their code before they become a serious problem, like:

→ Web interface and mobile APIs

Exploitable web applications and APIs can allow an attacker to access sensitive credentials, and something as simple as a security misconfiguration or business logic vulnerability can lead to a significant privacy breach – or at least, more information being passed between software than intended – within connected apps.

→ Mobile apps

Many of us enjoy the modern convenience car connectivity via an on-board interface. However, a malicious attack is possible if a vulnerability provides unintended access to something even as simple as the radio. Remote file inclusion would allow malware to be played across onboard multimedia apps.

→ Code injection on entertainment systems

On an exploitable system, an attacker could potentially create multimedia files that can change code within it. This opens pathways to exploit, and even remotely monitor, other parts of connected vehicles.

→ Wireless Media

Threat actors can attack vulnerabilities in wireless channels such as Bluetooth or Wi-Fi, which can bypass administrative privileges.

→ External Sensor Interfaces

Threat actors can spoof external sensors and force the vehicle into taking unwanted actions.

→ Wireless key entry

Vulnerable apps can be used to exploit wireless key entry, with attackers able to use a proxy bridge between the key and the automobile, giving them the ability to lock or open the automobile at will. This has already been proven with attacks on several vehicles.

→ External Device Access through the OBD-II Port

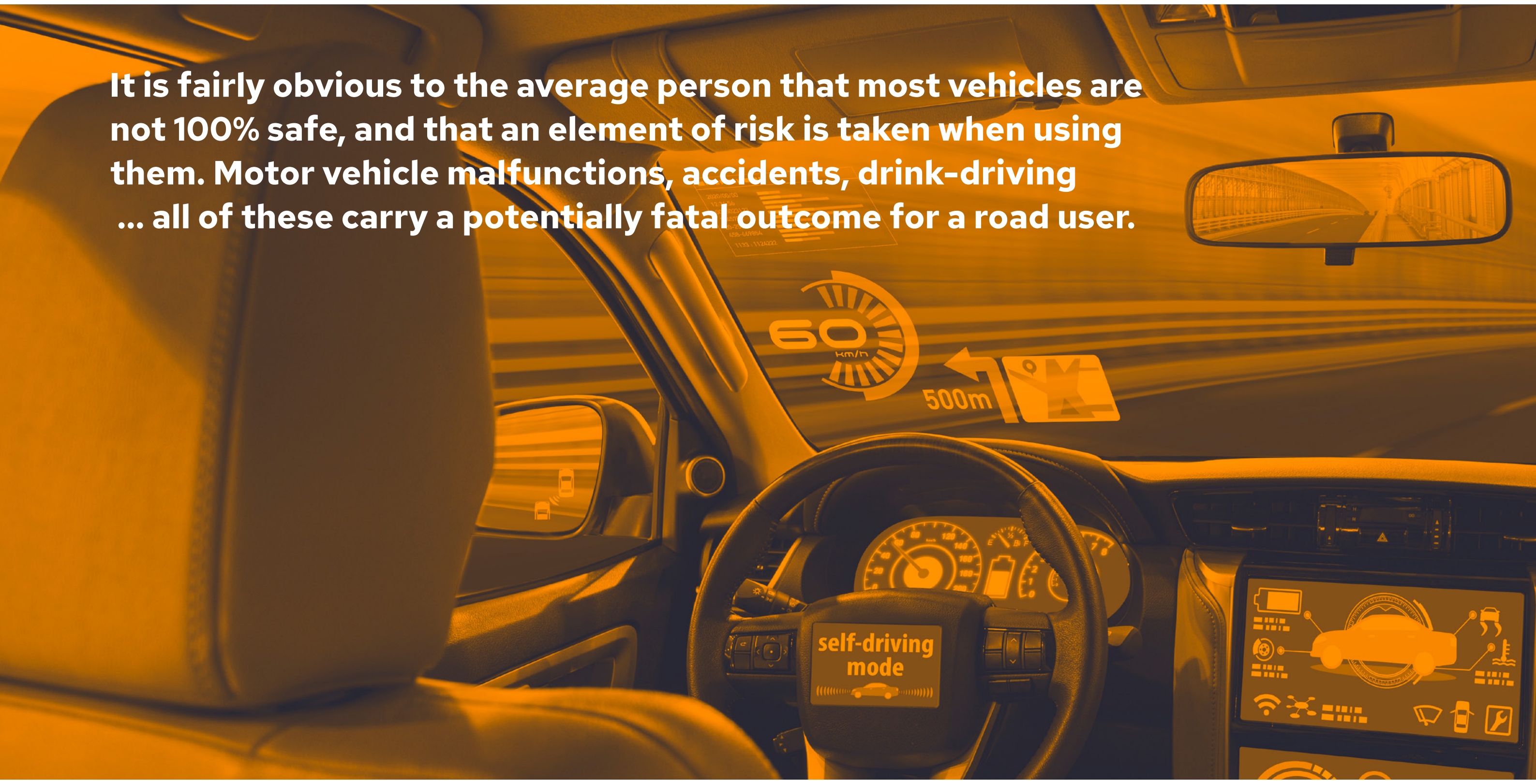
Potential access to the vehicle's internal systems.

→ Attacks on the cloud service of automotive provider

A vulnerable cloud infrastructure – even something as simple as a misconfiguration – could potentially enable a threat actor to attack many connected vehicles at once.

There are plenty of common back doors that savvy engineers can close in their code before they become a serious problem

How serious is the situation of a compromised vehicle?



It is fairly obvious to the average person that most vehicles are not 100% safe, and that an element of risk is taken when using them. Motor vehicle malfunctions, accidents, drink-driving … all of these carry a potentially fatal outcome for a road user.

But what if that catastrophic vehicle malfunction was actually caused remotely, as a result of a particularly malicious cyberattack? It has long been suggested that the world will get serious about cybersecurity when there are life-threatening consequences, but the reality is that we are well into that territory already, and without intervention, it will only escalate from here.

Back in 2015, **security researchers successfully "killed" the engine of a Jeep Cherokee** as it drove on a freeway; using a known zero-day exploit in the system software, they could wirelessly control the air conditioning, radio, steering, brakes, and transmission. Though dangerous, this was a contained experiment, but it proved the lethal control an attacker could have over a vehicle and its occupants. Since this event, millions of connected vehicles have hit our roads, each representing millions of lines of code that must be secured.

Autonomous vehicle technology (and its adoption) is moving at a cracking pace, and this can have the consequence of immense strain on its creators, especially the teams responsible for shipping the code that powers the conveniences of tomorrow. There is an urgent need for software developers in the automotive industry to share the responsibility for security, and ASRG is the community hub many rely on for the latest security knowledge, tools, peer recommendations, and support. Their global Secure Code Warrior tournament sought to engage, assess, and inspire over 100 developers representing ASRG chapters all over the world. Participating in friendly competition and training, they sought to solve secure coding challenges that directly relate to the issues facing the software prevalent in their industry.

Autonomous vehicle technology (and its adoption) is moving at a cracking pace, and this can have the consequence of immense strain on its creators, especially the teams responsible for shipping the code that powers the conveniences of tomorrow.

The facts and figures from tournament and training trials

This is an indication of high engagement and a desire to keep playing – both immensely beneficial byproducts of gamification techniques in training and education.

Training and tournaments can be played in the languages and frameworks as desired by each individual developer, ensuring that challenges are hyper-relevant and using real-world code they would come across in day-to-day work. This contextual, bite-sized approach to learning ensures swift delivery of the content that matters most to solve the problems most prevalent in the organization's SDLC.

TIME SPENT

IN TRAINING:

3303 minutes

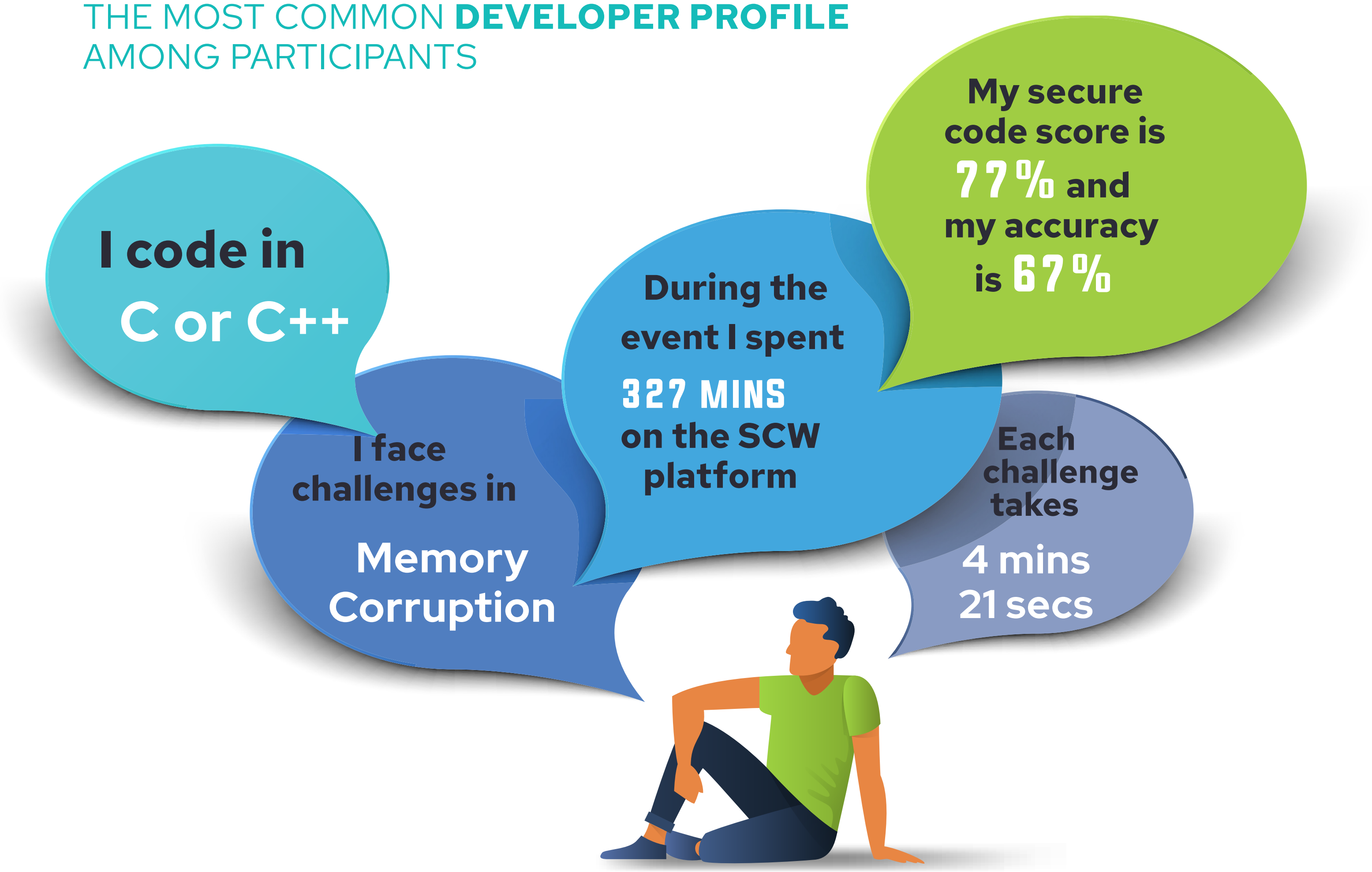
ACTIVE IN TOURNAMENTS:

3697 minutes

IN FIRST-TOUCH LEARNING

189 minutes

THE MOST COMMON **DEVELOPER PROFILE** AMONG PARTICIPANTS



I code in C or C++

I face challenges in Memory Corruption

During the event I spent 327 MINS on the SCW platform

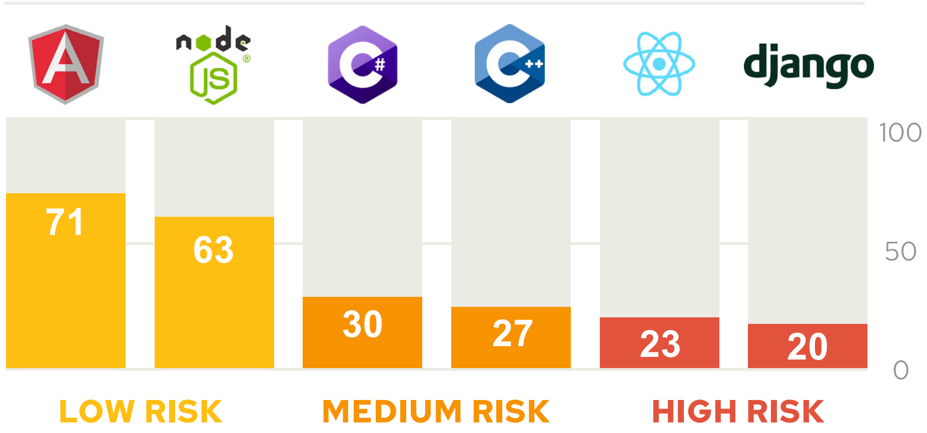
My secure code score is 77% and my accuracy is 67%

Each challenge takes 4 mins 21 secs

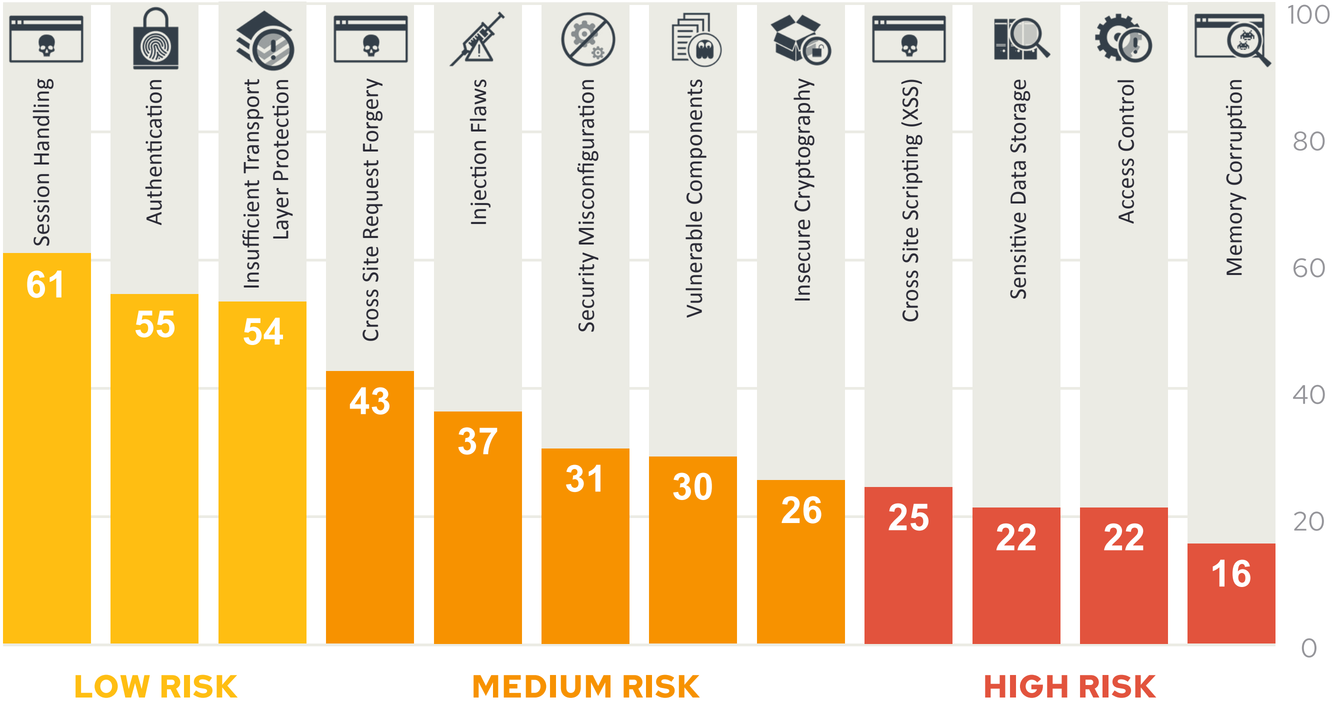
Tournaments are a great way to introduce security standards, a benchmark of quality, and the responsibility to learn how to squash common security bugs in code

Other Significant Findings:

Global ASRG Virtual Secure Coding Tournament: Secure Code Score per Language



Global ASRG Virtual Secure Coding Tournament: Secure Code Score per Vulnerability



🟨 **Low Risk: 51-75%**

🟧 **Medium Risk: 26-50%**

🟥 **High Risk: up to 25%**

While all participants showed some proficiency in their chosen languages and frameworks, there was no single vulnerability area that was deemed “100% secure”, or mastered, and the average accuracy score was 67%.

There is no expectation for any developer to become a security expert, but tournaments are a great way to introduce security standards, a benchmark of quality, and the responsibility to learn how to squash common security bugs in code... especially when that code may lead to remote access control of someone’s vehicle, or worse.

Tournament insights into vulnerability and skill-based risk factors

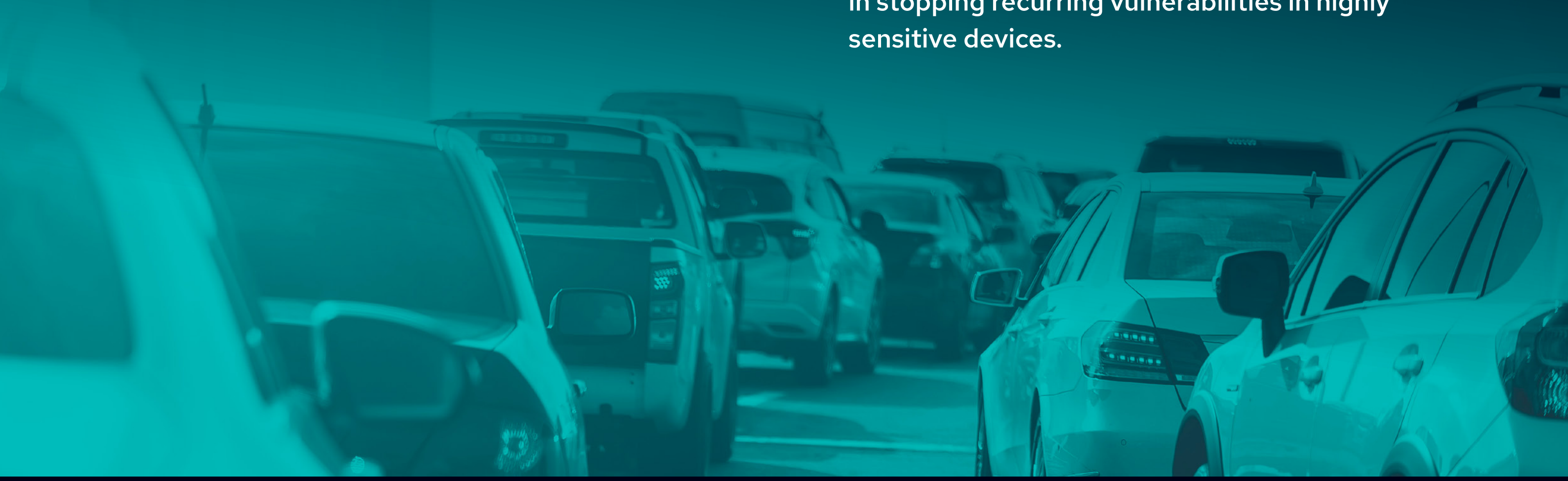
The ASRG tournament and training initiatives concentrated on some key vulnerabilities that affect connected vehicles, namely:

- **Insufficient Transport Layer Protection**
- **Insecure Cryptography**
- **Authentication**
- **Security Misconfiguration**
- **Sensitive Data Storage**
- **Memory Corruption**

After thousands of training minutes and hundreds of challenges, it became apparent that the clear areas of focus should remain on access control, sensitive data storage, and, as a priority, memory corruption vulnerabilities.

The latter is a known potential exploit in not just ultra-connected vehicles, but many other IoT devices. Such a bug was recently discovered by Cisco's Customer Experience Assessment & Penetration Team (CX APT) in GNU Glibc, a library used in Linux ARMv7 systems, leaving them vulnerable to memory corruption until a patch is created and applied. In these times of sensor-heavy devices using real-time data collection from multiple environmental points, the paydirt could be significant for an attacker, even if remote control of the device isn't possible.

The team at ASRG have built incredible resources for developers needing to perform in automotive security, with their directory of tested tools and solutions, comprehensive wiki, and powerful global community. These independent group initiatives are what takes to drive change at a grassroots level, and their willingness to try new things and forge the foundations of security awareness in their members is a powerful element in stopping recurring vulnerabilities in highly sensitive devices.



Return on investment from spending now on secure coding best practices

A study published by SAE International and the Synopsys Software Integrity Group found that, in terms of ensuring connected technologies were secured and safeguarded from both existing and emerging cyber threats, the automotive industry was significantly lagging behind many others.

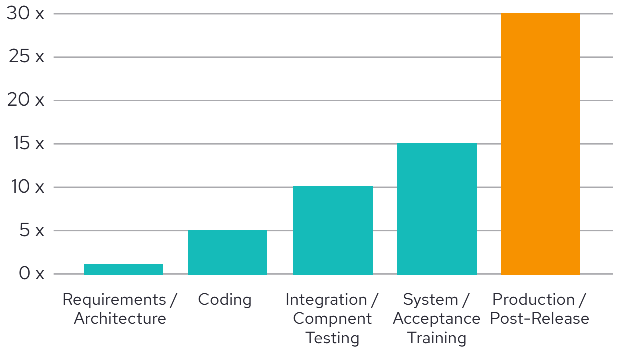
It's a concerning trend, but it's not irreversible - especially with organizations like ASRG fighting to keep security front-of-mind in the industry, while shedding light on the solutions, tools, and education needed for automotive companies to build an iron-clad security program.

Their experience with running a highly engaging, global Secure Code Warrior Tournament gave them the opportunity to identify core areas of risk within a development cohort, opportunities for further learning, accuracy statistics from secure coding challenges, and the key vulnerabilities to focus on, as relevant to the needs of the industry.

So, what would the estimated returns on transforming a security program within an organization, instilling security awareness and action from the very beginning of the SDLC? Let's take a look:

For a business identifying even a modest number of annual vulnerabilities in their security audits, the potential costs of detection and remediation can be significant. And, depending on where these annoying bugs are revealed in the process, the price of correction can dramatically increase, even for the "simple" fixes - up to thirty times the cost for a late-stage fix, versus one that was found and fixed at the beginning.

Relative cost to fix, based on time of detection



Source: National Institute of Standards and Technology

Ignoring common vulnerabilities until the latest possible moment is a sure-fire way to blow budgets and miss critical release dates. By starting left, and empowering developers to erase decades-old bugbears like SQL injection, XSS, and security misconfigurations, the cost – not to mention time – savings are immense.

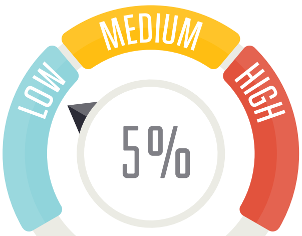
Return on investment

This three-point estimate shows the potential financial and day impact of three different savings enabled by Secure Code Warrior's training, tournaments, and cultural transformation.

Annual defects attributable to Sec Vuln?	2,500	Developer Cost/hr	$18/hr

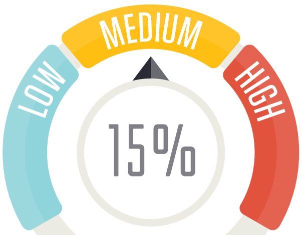
Ponemon Institute Research		Stage Detected	#	$ of Remediation
CODING	$80	20%	500	$40,000
BUILD	$240	50%	1250	$300,000
QA/TEST	$960	25%	625	$600,000
PRODUCTION	$7,600	5%	125	$950,000
			Customer Annual Cost	**$1,890,000**

Potential Annual Savings



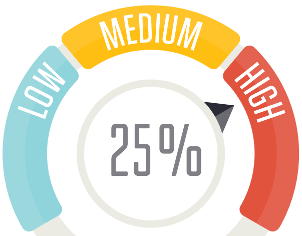
$94,500

5,250 Developer Hrs



$283,500

15.750 Developer Hrs



$472,500

26,250 Developer Hrs

** Example calculation only, based on defect remediation cost data from IBM Security and Ponemon Institute*

ASRG is making headway in promoting and nurturing the next generation of security superheroes in the automotive industry, and their creative approach and wealth of knowledge makes it the go-to organization for DevSec-ready individuals to hone their skills.

www.asrg.io info@asrg.io

Request a TOURNAMENT today and see how you can transform your approach.

REQUEST TOURNAMENT

ABOUT SECURE CODE WARRIOR

Secure Code Warrior is the developer-chosen solution for growing powerful secure coding skills. By making security a positive and engaging experience, our human-led approach uncovers the secure developer inside every coder, helping development teams ship quality code faster.

Through inspiring a global community of security-conscious developers to embrace a preventative secure coding approach, our mission is to pioneer a people-first solution to security upskilling, stamping out poor coding patterns for good.

ABOUT ASRG

The Automotive Security Research Group (ASRG) is a non-profit initiative to promote the development of security solutions for automotive products. The goals of ASRG are not to produce any products or services, however to support and assist with the development of security solutions in the automotive industry. This will be achieved by focusing on 3 main topics, knowledge, networking and collaboration. Security is essential for all automotive functions, and it's our (drivers, passengers, development engineers, researchers, etc.) responsibility to ensure that the products that we use every day are safe and secure. Make an impact on the industry and join us at www.asrg.io.

info@securecodewarrior.com

securecodewarrior.com