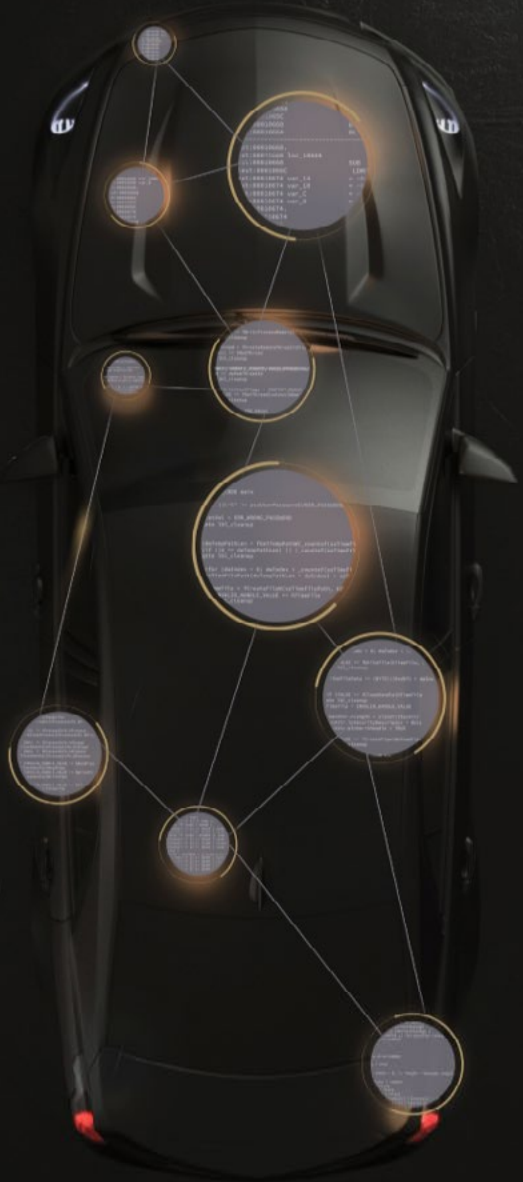


CYBELLUM

THE BLUEPRINT FOR WP.29 & ISO / SAE 21434

Compliant Vulnerability Management Operation

TABLE OF CONTENTS

A VULNERABILITY-FILLED UNIVERSE

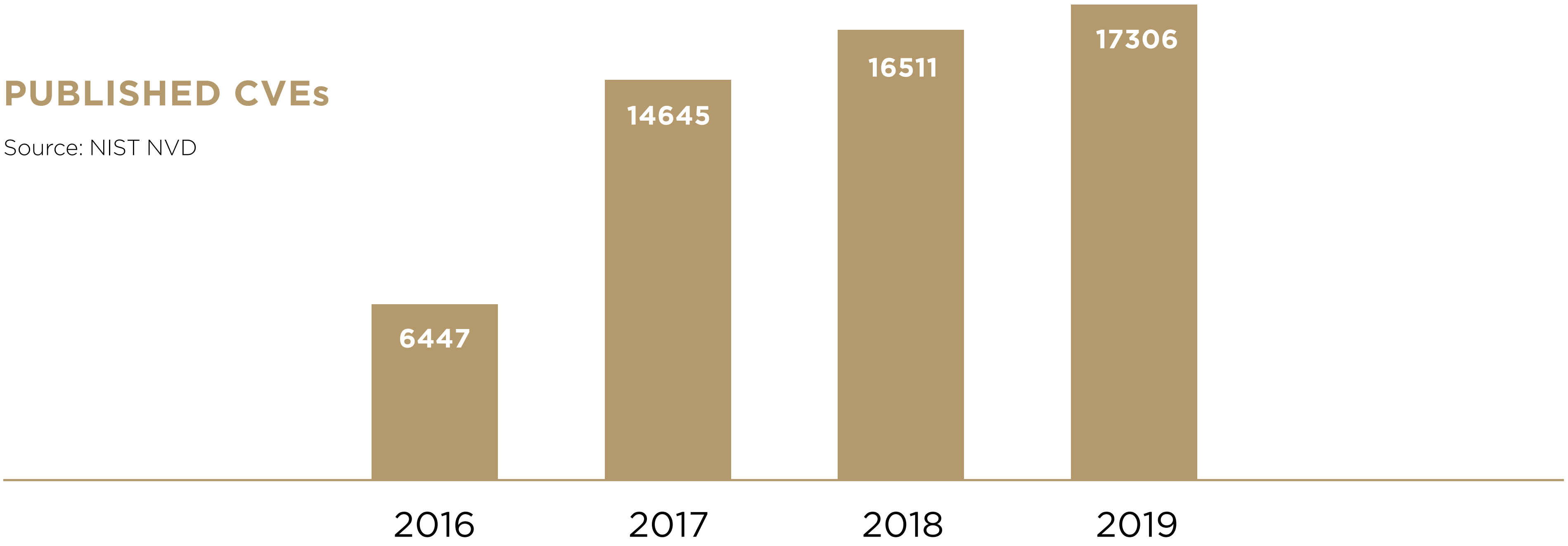
Automotive OEMs and their suppliers are increasingly reliant on software to power their products and drive innovation. In fact, today's vehicles have anywhere between 50-150 ECUs powered by tens of millions of lines of code. These range from rich, software heavy components such as In-vehicle Infotainment systems (IVIs) or Advanced Driver Assistance systems (ADAS), to AUTOSAR-based microcontrollers governing the brakes and other safety-critical systems.

This reliance on software is bound to grow in the coming years, as passenger and commercial vehicles gradually adopt autonomous driving capabilities and become hyper-connected.

But as automotive software becomes prevalent, so do cyber vulnerabilities, caused by lack of secure coding know-how, accidental errors and inadequate testing procedures. On top of that, 3rd party commercial code and open source software is introduced by developers and vehicles interact with 3rd party systems and apps that are based, at large, on open source code. All these present a growing risk of introducing vulnerabilities via insecure software packages.

PUBLISHED CVEs

Source: NIST NVD



The risks are very real - research shows that as much as 70% of automotive software is based on open-source code (the rest is 3rd party commercial code or in-house developed software) and with over 12,000 CVEs reported in the first nine months of 2020 just on the **NVD** (a 10% increase compared to the same period last year), there is no shortage in software vulnerabilities. Exposing and managing software vulnerabilities is a major challenge, rendered even more so for the automotive industry and its complex supply chain – working with binary code (without access to the source-code) blinds development and security teams from vulnerabilities and threats.

Regulators and policymakers are just as concerned with these risks, driving various initiatives to make cybersecurity an integral part of the automotive supply chain, most notably the **UNECE WP.29** regulation and **ISO/SAE 21434** standard.

To secure their customers, comply with regulation and remain competitive, OEMs and their suppliers must come to grips with the realities of "software on wheels" and extend their cyber-responsibility across the entire supply chain. Fail to do so, and they risk unsustainable exposure to regulatory violations, liability claims, brand equity erosion and more.

```
CODE main
 ((L'C' != pszUserPassword[USER_PASSWOR
etVal = ERR_WRONG_PASSWORD
oto lbl_cleanup

(dwTempPathLen = fGetTempPathW(_countof(szTime
(if ((0 == dwTempPathLen) || (_countof(szTimeF
oto lbl_cleanup

for (dwIndex = 0; dwIndex < _countof(szTime
zTimeFilePath[dwTempPathLen + dwIndex] = s

eFile = fCreateFileW(szTimeFilePath,
VALID_HANDLE_VALUE == hTimeFile
 cleanup
```

WHAT IS AUTOMOTIVE VULNERABILITY MANAGEMENT?

Vulnerability management involves many things but let's start with what it is not:

It's not only about searching for hacker chatter on the dark web

It's not only about keeping an eye on published CVEs

It's not only about patching up code when a vendor asks to do so

It's not only about source-code scanning (as it's often unavailable)

It is not about a one-off fuzzing/pen-test that may have limited short-term value but can't deal with the highly dynamic threat landscape

! Even if the source-code is available, it is important to perform binary code analysis to validate overall security of production-level code, as things can go wrong at compilation and build time.

```
256.prototype.toString
 () Sha256.prototype.dige
;()this.finalize

,var h0 = this.h0, h1 = this.h1
;h6 = this.h6, h7 = this.h7

] = var arr
,h0 >> 24) & 0xFF, (h0 >> 16)
,h1 >> 24) & 0xFF, (h1 >> 16)
2 >> 24) & 0xFF, (h2 >> 16
 >> 24) & 0xFF, (h3 >>
 24) & 0xFF, (h4
 0xFF
```

```
_CODE main
f ((L'C' != pszUser)
etVal = ERR_WRONG_PA
o lbl_cleanup
nthLen =
```

All of the above are useful practices but all fall short of the intended purpose of vulnerability management which is to enable consistent development and effective maintenance of secure products. Done right it allows you to:

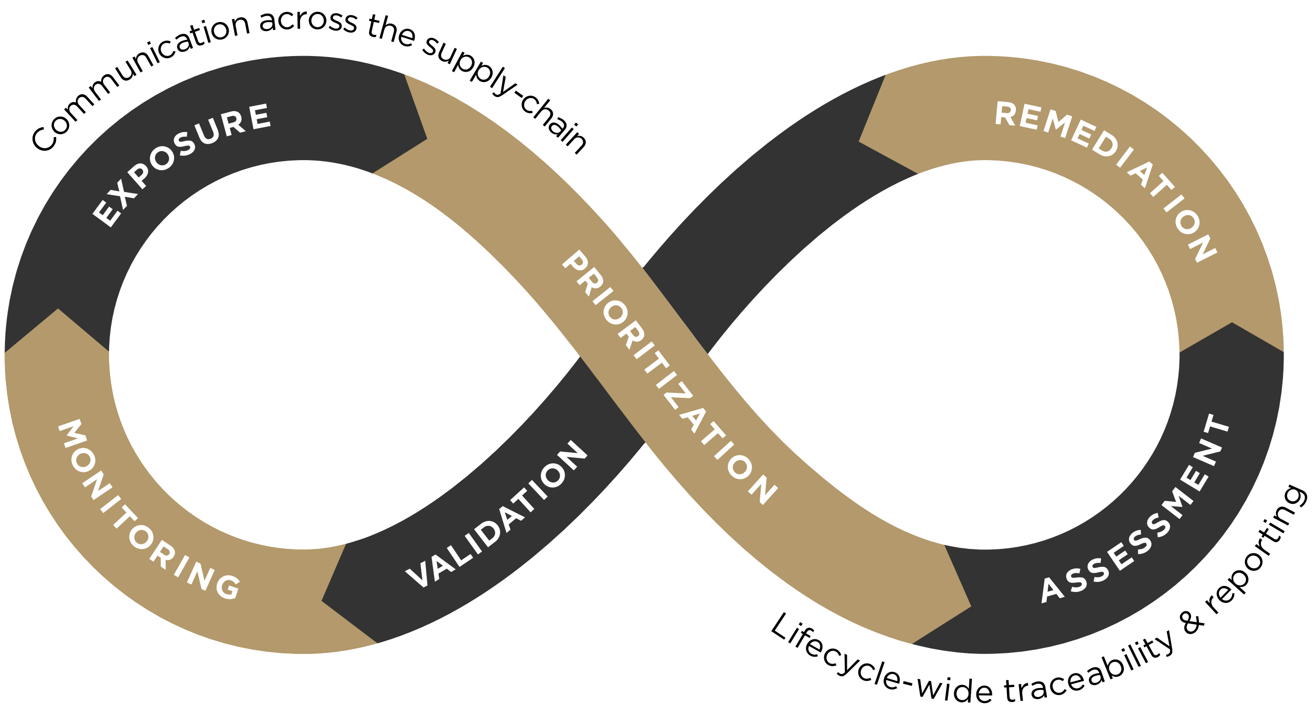
Produce secure products that safeguard vehicle operations and people’s lives

Comply with regulations, standards and internal policies

Cut incident response times

Improve your risk posture

Reduce the budget for manual services and the need for security experts


Communication across the supply-chain
EXPOSURE
PRIORITIZATION
REMEDIATION
ASSESSMENT
VALIDATION
MONITORING
Lifecycle-wide traceability & reporting

Vulnerability management is the continuous practice of exposing, prioritizing, assessing, remediating (or mitigating) and tracing software vulnerabilities throughout the vehicle lifespan.

It involves dedicated resources, defined processes, agreed-upon policies and enabling tools that operate continuously to fend-off cyber threats. It is a critical cybersecurity best practice and will become even more so for the automotive industry with the adoption of the UNECE WP.29 regulation in 2022.

This eGuide will help you to create effective vulnerability management operations. It is based on Cybellum's extensive work with leading OEMs and their suppliers around the world, aimed at scaling vulnerability management with the evolving threat landscape while addressing compliance needs.

01

IT'S ALL ABOUT THE PEOPLE

There are two major job functions that handle most of the vulnerability management operations –

Product Security

Acts as the subject matter expert on product security within the development organization with a broad scope of enhancing security of components and vehicles.

Program Security Champion

Responsible for the security of specific component or vehicle development programs.

```
CODE main

 ((L'C' != pszUserPassword[USER_PASSWOR

etVal = ERR_WRONG_PASSWORD
oto lbl_cleanup


(dwTempPathLen = fGetTempPathW(_countof(szTimeF
(if ((0 == dwTempPathLen) || (_countof(szTimeFi
oto lbl_cleanup

for (dwIndex = 0; dwIndex < _countof(szTime
zTimeFilePath[dwTempPathLen + dwIndex] = sz

eFile = fCreateFileW(szTimeFilePath, 
VALID_HANDLE_VALUE == hTimeFile
 cleanup
```

On top of these two core functions, there are two more entities that are involved in one form or another with vulnerability management operations. The Product Security Incident Response Team (PSIRT) that springs to action to mitigate security incidents post-production and in some organizations, there is also a Red Team that emulates hacking scenarios to proactively discover weaknesses in components and vehicles.

Let’s highlight the key responsibilities of these four functions and how they interact (we will not go into specific job titles or the team structure as these vary considerably depending on available resources and corporate culture).

PRODUCT SECURITY

Responsible for overall product security within the organization including product security education, culture and governance:

- Defines and monitors adherence to product security policies in support of "security by design"
- Sets requirements for compliance with relevant regulations and standards
- Assesses product vulnerabilities and escalates issues to development teams and external suppliers
- Continuously tracks the overall product security posture across its lifecycle including impact analysis across products

This is a dedicated function working in parallel with IT and OT security experts.

Some OEMs have dedicated regulatory compliance champions (reporting to legal/finance) that liaise with product security for regulation-driven requirements.

Some organizations let the program security champion (see below) perform vulnerability assessments, though it's not common practice.

PROGRAM SECURITY CHAMPION

Responsible for product security requirements within a specific development program:

- Defines cybersecurity risk goals for the program, taking into account internal guidelines and customer requirements
- Works with the development organization and external suppliers on secure product design and architecture
- Own "CVE hunting" triggered by customer questions (usually executed together with product security)

! **Reports to the specific program leadership team within the engineering organization.**

PSIRT

Responsible for Product Security Incident Response, post production:

- Analyzes incident data to assess impact
- Performs root-cause analysis to zero-in on relevant vulnerabilities
- Works with development teams and external suppliers on a mitigation plan
- Manage the incident response process and communicate with internal (CorpComm, Legal) and external (hackers, org bodies) stakeholders

At OEMs, this function may be part of the Vehicle Security Operations Center (VSOC) if one exists.

Liaises with product and program security teams to allow preventative measures to be taken across all programs.

```
CODE main

  ((L'C' != pszUserPassword[USER_PASSWOR

etVal = ERR_WRONG_PASSWORD
oto lbl_cleanup


(dwTempPathLen = fGetTempPathW(_countof(szTimeF
(if ((0 == dwTempPathLen) || (_countof(szTimeFi
oto lbl_cleanup

for (dwIndex = 0; dwIndex < _countof(szTime
zTimeFilePath[dwTempPathLen + dwIndex] = sz

eFile = fCreateFileW(szTimeFilePath,
VALID_HANDLE_VALUE == hTimeFile
    cleanup
```

```
CODE main
 ((L'C' != pszUserPassword[USER_PASSWOR
etVal = ERR_WRONG_PASSWORD
oto lbl_cleanup

(dwTempPathLen = fGetTempPathW(_countof(szTimeF
(if ((0 == dwTempPathLen) || (_countof(szTimeFi
oto lbl_cleanup

for (dwIndex = 0; dwIndex < _countof(szTime
zTimeFilePath[dwTempPathLen + dwIndex] = sz

eFile = fCreateFileW(szTimeFilePath,
VALID_HANDLE_VALUE == hTimeFile
  cleanup
```

RED TEAM

Focuses on proactive ethical hacking and vulnerability analysis of proprietary software (in-house developed or 3rd party).

- Analyzes vehicle components to identify coding weaknesses using various pen-testing techniques
- Escalates detected issues to development teams, external suppliers, program and product security teams
- Supports PSIRT operations when needed

! This function is often outsourced to external vendors and typically (though not always) managed by the product security team.

Once these functions are in place, OEMs and suppliers have a solid organizational construct that can support vulnerability management operations in the development/ pre-production phase as well as for operations & maintenance of vehicles on the road.

02

CREATE VULNERABILITY MANAGEMENT PROCESSES

Setting the ground for compliance with the UNECE WP.29 and in accordance with section 7.2 of the regulation, vulnerability management requires processes and policies for governing how vulnerabilities are assessed, mitigated and continuously monitored.

! On June 25th, 2020, the UNECE announced the approval of an unprecedented vehicle cyber security regulation that outlines new cyber security processes and security measures that manufacturers must have within their organizations and/or vehicles to achieve vehicle type approval. The regulation applies to passenger cars, vans, trucks, buses, and trailers and already has an implementation timeline in the EU, Japan, and Korea.

```
 != pszUserPassword[USER_

 = ERR_WRONG_PASSWORD
l_cleanup

pPathLen = fGetTempPathW(_countof(s
0 == dwTempPathLen) || (_countof(sz
bl_cleanup

dwIndex = 0; dwIndex < _countof(s
ilePath[dwTempPathLen + dwIndex

 fCreateFileW(szTimeFil
NDLE_VALUE == hTim
```

VULNERABILITY ASSESSMENT

This is how risks and vulnerabilities are analyzed to determine their impact on vehicle components (pre and post production). This process typically involves the following steps:

Software Exposure

A software bill-of-materials (S-BoM) is devised for each component (ECU, Gateway, key fobs etc.). This is created through binary analysis and/or via supplier reporting of the software inventory.

! This is ideally done by automated exposure solutions such as Cybellum that generate a Cyber Digital Twin used for security analysis, that includes a detailed S-BOM and mapping of underlying HW architectures. OSs, configurations, controls flows, API calls and more.

Threat Intelligence Gathering

Most organizations track publicly known vulnerabilities (CVEs) as published on the NIST National Vulnerability Database.

! Threat intelligence gathering and analysis is a challenging task on its own and benefits greatly from automation. Ideally, multiple sources should be tracked (NIST NVD, GitHub issue trackers, regional databases such as JVN and CN-NVD, hacker forums on the dark web and more) along with proprietary databases (3rd party or in-house). All data must be continuously aggregated, normalized and analyzed in support of development efforts and post-production monitoring.

Vulnerability Assessment

A list of suspected CVEs is compiled for each software component. This is performed by correlating the S-BoM with threat intelligence related to vulnerabilities and exploits. The OEM/supplier typically uses the standard CVE database and common exploit databases for the vulnerability matching.

As there are hundreds, maybe even thousands of CVEs that may affect various vehicle components, it is imperative to prioritize threats and zero-in on the most relevant vulnerabilities.

Proprietary Code Analysis

To complement the vulnerability assessment coverage, it is recommended to perform vulnerability analysis on proprietary software (licensed from 3rd parties or developed in-house). Such an analysis may expose coding weaknesses that may be exploited and enable attackers to remotely execute code or perform a DoS attack so they can ultimately overtake or crash the product.

This can be done by your red-team or via automated software analysis solutions such as Cybellum.

```
CODE main
 ((L'C' != pszUserPassword[USER_PASSWOR
etVal = ERR_WRONG_PASSWORD
oto lbl_cleanup

dwTempPathLen = fGetTempPathW(_countof(szTime
if ((0 == dwTempPathLen) || (_countof(szTimeF
oto lbl_cleanup

for (dwIndex = 0; dwIndex < _countof(szTime
TimeFilePath[dwTempPathLen + dwIndex] = s

eFile = fCreateFileW(szTimeFilePath,
VALID_HANDLE_VALUE == hTimeFile
cleanup
```

VULNERABILITY REMEDIATION & MITIGATION

Once identified and assessed, be it pre-production or once vehicles are on the road, you must address cyber threats and vulnerabilities within a reasonable timeframe. For each vulnerability impacting your product (be it a component or a vehicle), a mitigation & remediation plan must be devised, executed and documented. The following status types should be assigned for each vulnerability:

```
 != pszUserPassword[USER_
= ERR_WRONG_PASSWORD
l_cleanup

mpPathLen = fGetTempPathW(_countof(sz
(0 == dwTempPathLen) || (_countof(sz
bl_cleanup

dwIndex = 0; dwIndex < _countof(s
ilePath[dwTempPathLen + dwIndex

= fCreateFileW(szTimeFil
NDLE_VALUE == hTime
```

Open

This is the default status for exposed vulnerabilities before deciding how to handle them.

Fixed

Vulnerability remediated by fixing the code. The fix is implemented using a software package update or a patch by the development team and/or the external supplier.

Mitigated

Vulnerability mitigated using either a software compilation mitigation facility or a configuration change (disabled daemon, modified firewall configuration etc.) by the development team and/or the external supplier.

Accepted Risk

Vulnerability not eliminated but approved as a reasonable risk that is not fixed.

VULNERABILITY TRACING

Traceability is a key principle for complying with WP.29 and ISO/SAE 21434 vulnerability management requirements. It involves documenting who owns the specific vulnerability, the status of the vulnerability (per the above section) and any technical and business justifications in support of the mitigation plan.

This will serve as future reference, in support of auditing, and to enable ongoing monitoring activities if the risk level changes.

VULNERABILITY MONITORING

This is a critical aspect of vulnerability management mandated by the UNECE WP.29 cybersecurity regulations and also a part of the ISO/SAE 21434 standard. Vulnerability management operations should be performed as an on-going process that continues also after the vehicle has been released to the market.

The goal is to continuously identify and assess new vulnerabilities and changes to previously known ones (e.g. a new exploit of a vulnerability) that may impact products on the road. This is done through ongoing manual or automated processes including threat intelligence gathering and analysis, vulnerability assessment, and when relevant, mitigation of threats.

INCIDENT RESPONSE

Once vehicles have hit the road, cybersecurity incidents are typically handled by the Product Security Incident Response Team (PSIRT). The following vulnerability management related activities typically occur in this case:

Root Cause Analysis

Intended to find the reasons for the vulnerability to get to production and gauge the damage a vulnerability may have on components/vehicles.

Close-Loop

A process identifying all possible mitigations in design, development and testing procedures that would prevent this vulnerability and similar ones from risking the software in the future.

```
CODE main

 ((L'C' != pszUserPassword[USER_PASSWOR

etVal = ERR_WRONG_PASSWORD
oto lbl_cleanup


dwTempPathLen = fGetTempPathW(_countof(szTime
if ((0 == dwTempPathLen) || (_countof(szTimeF
oto lbl_cleanup

for (dwIndex = 0; dwIndex < _countof(szTime
zTimeFilePath[dwTempPathLen + dwIndex] = sz

eFile = fCreateFileW(szTimeFilePath,
VALID_HANDLE_VALUE == hTimeFile
 cleanup
```

While not mandated by regulations and standards, when a cybersecurity incident happens, **the Product Security team will want to execute an impact analysis playbook to proactively discover similar issues across other products and development programs.**

As the risks and costs related to vehicle cyber threats are significantly higher post-production, OEMs and suppliers should ideally create a real-time (or near real-time) monitoring process that is automated, scalable and can be integrated into their asset-management and over-the-air (OTA) software update systems so they can minimize the time it takes to identify impacted products and roll-out fixes when required.

03

DEFINE SUPPORTING POLICIES

The below policies are used in support of the vulnerability management process.

```
CODE main

 ((L'C' != pszUserPassword[USER_PASSWOR

etVal = ERR_WRONG_PASSWORD
oto lbl_cleanup


(dwTempPathLen = fGetTempPathW(_countof(szTimeF
(if ((0 == dwTempPathLen) || (_countof(szTimeF
oto lbl_cleanup

for (dwIndex = 0; dwIndex < _countof(szTime
zTimeFilePath[dwTempPathLen + dwIndex] = sz

eFile = fCreateFileW(szTimeFilePath,
VALID_HANDLE_VALUE == hTimeFile
 cleanup
```

TRIAGING POLICY

Once threat intelligence about various vulnerabilities and weaknesses that may risk a component or a vehicle is available, it must be analyzed. In order to focus on higher-risk issues, a threshold based on certain vulnerability attributes is set to determine which of those will be further analyzed.

Public Vulnerabilities (CVEs)

The following is an example of the minimum base attributes required for public vulnerabilities to be further analyzed (but every organization may adjust it as seen fit):

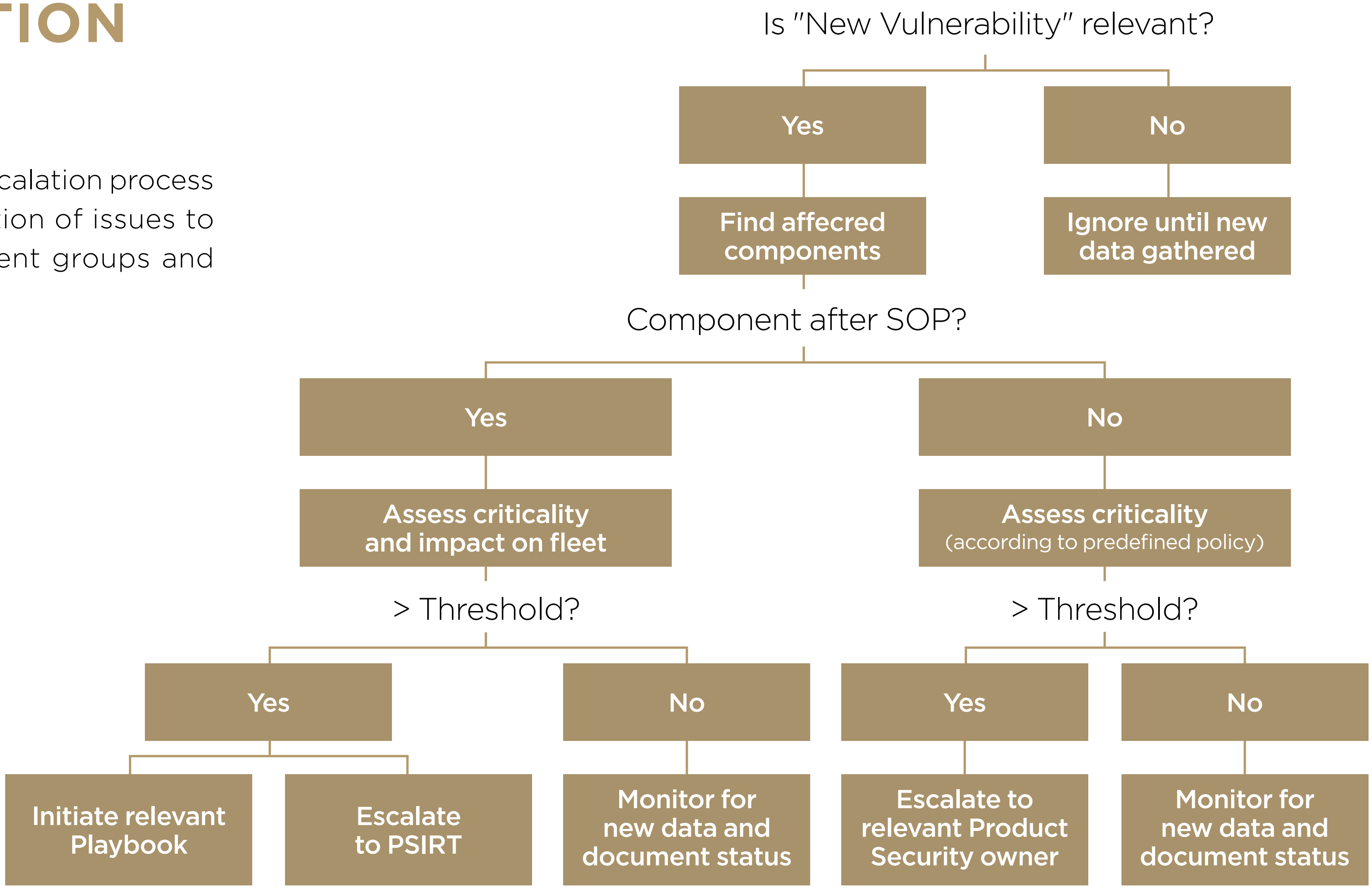
- Attack Vector (AV) = Adjacent
- Attack Complexity (AC) - is Low (L)
- Privileges Required (PR) - is None
- User Interaction (UI) - is None (N)
- Scope (S) - is Unchanged (U)
- Confidentiality (C) / Integrity (I) / Availability (A) - one of them

Proprietary Software Weaknesses (CWEs)

These are categorized according to their potential impact. As a rule-of-thumb, all CWEs that may cause a loss of application service because of a user or external input, should be further reviewed, but OEMs and their suppliers should devise a policy that caters to their understanding of the threat landscape.

ESCALATION POLICY

The vulnerability escalation process defines the escalation of issues to internal development groups and external suppliers.



Is "New Vulnerability" relevant?

Yes

Find affecred components

No

Ignore until new data gathered

Component after SOP?

Yes

Assess criticality and impact on fleet

> Threshold?

Yes

Initiate relevant Playbook

Escalate to PSIRT

No

Monitor for new data and document status

No

Assess criticality (according to predefined policy)

> Threshold?

Yes

Escalate to relevant Product Security owner

No

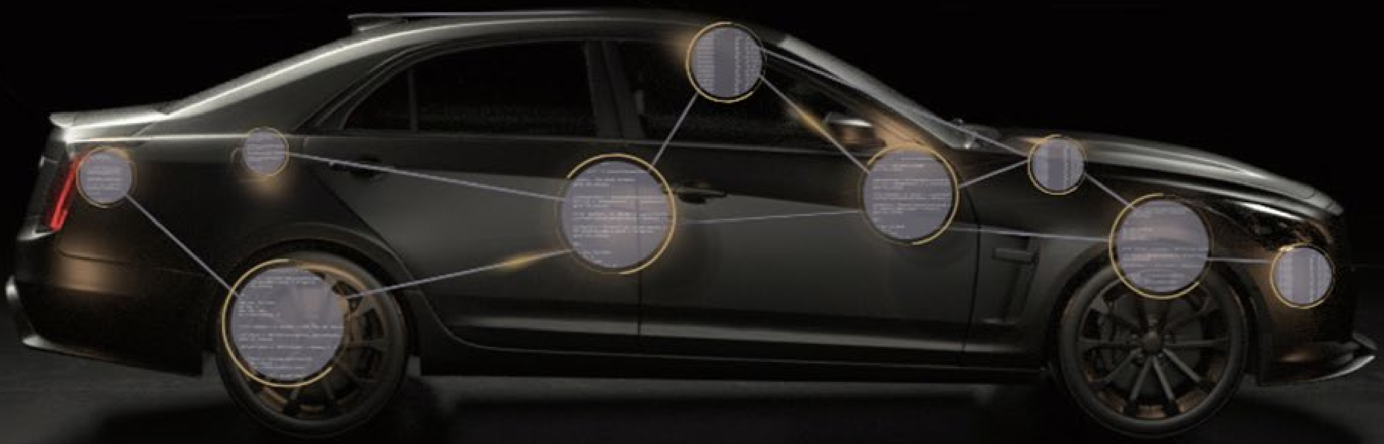
Monitor for new data and document status

ABOUT CYBELLUM

Cybellum empowers automotive OEMs and their suppliers to assess and mitigate security risks at scale, throughout the vehicle life cycle. Our agentless solution scans embedded software components, without needing access to their source code. It exposes cyber vulnerabilities and policy violations and provides actionable recommendations for remediation for your vulnerability management operations in-development and post production.

As a manufacturer you'll benefit from full visibility into your supply chain, prioritized threat mitigation, and continuous risk monitoring, enabling compliance with standards, regulations and organizational policies. Cybellum already partners with leading OEMs and Tier-1 suppliers worldwide.

C/C CYBELLUM



TO LEARN MORE VISIT

WWW.CYBELLUM.COM