CSAF Writing Bootcamp

Justin Murphy (CISA, US)
Thomas Schmidt (BSI, DE)
Who are we?

Justin Murphy
- Vulnerability Disclosure Analyst @CISA
- Passion for
  - SBOM/VEX
  - CSAF
  - CVD
  - Service
  - International Cooperation

Thomas Schmidt
- Technical ICS Analyst @BSI (usually not into standardization)
- Passion for
  - ICS
  - International Cooperation
  - CVD
  - Capacity building
  - CSAF
Things covered in the workshop

- Introduction into CSAF
- How to write security advisories in CSAF
- Available tools for writing advisories
- VEX and its relation to CSAF
- Hands-on exercises
Introduction into CSAF

- Who has heard of CSAF?
- Who has used CSAF?
Transforming the Vulnerability Management Landscape

Released: November 10, 2022
Revised: November 14, 2022

Eric Goldstein, Executive Assistant Director for Cybersecurity

In the current risk environment, organizations of all sizes are challenged to manage the number and complexity of new vulnerabilities. Organizations with mature vulnerability management programs seek more efficient ways to triage and prioritize efforts. Smaller organizations struggle with understanding where to start and how to allocate limited resources. Fortunately, there is a path toward more efficient, automated, prioritized vulnerability management. Working with our partners across government and the private sector, we are excited to outline three critical steps to advance the vulnerability management ecosystem:

- **First**, we must introduce greater automation into vulnerability management, including by expanding use of the Common Security Advisory Framework (CSAF).
- **Second**, we must make it easier for organizations to understand whether a given product is impacted by a vulnerability through widespread adoption of Vulnerability Exploitability eXchange (VEX).
- **Third**, we must help organizations more effectively prioritize vulnerability management resources through use of Stakeholder Specific Vulnerability Categorization (SSVC), including prioritizing vulnerabilities on CISA’s Known Exploited Vulnerabilities (KEV) catalog.

What is CSAF?

Common Security Advisory Framework

- International, open and free OASIS Standard
- Machine-readable format for security advisories (JSON)
- Standardized way of distribution security advisories
- Build with automation in mind
- Standardized tool set
- Guidance to actionable information
- Successor of CSAF CVRF 1.2

Ready to use!
Organizations publishing CSAF

- Cisco
- Omicron
- TIBCO
- Siemens
- SICK
- Oracle
- Red Hat
- Festo
- Hitachi Energy
- Schneider Electric
- Arista
Two sides of the same coin – different maturity stages

Vendor / Supplier

Vendor (specific) internal tools and processes
- Continuous Security Advisory Release (CSAR)
- Unique Product IDs
- Content Management System
- Text editor / Writer

Quality of advisories
- Supply Chain routinely included
- Input data for automatic asset management
- Existence of machine-readable format
- Existence of human-readable format

Asset owner / User

Asset owner (specific) internal tools and processes
- Routinely patching
- Automatic processing of advisories
- (Semi-) Automated processing of advisories
- Manual processing of advisories

Requirements for tools and processes
- Security downtime accepted/ minimized/ mitigated
- Asset management system with unique Product IDs
- Asset management with smart search
- Web browser

Difficulty/maturity/automation capability
Next step: reach stage 2 across parties

Vendor / Supplier

Vendor (specific) internal tools and processes
- Continuous Security Advisory Release (CSAR)
- Unique Product IDs
- Content Management System
- Text editor / Writer

Quality of advisories
- Supply Chain routinely included
- Input data for automatic asset management
- Existence of machine-readable format
- Existence of human-readable format

Asset owner / User

Asset owner (specific) internal tools and processes
- Routinely patching
- Automatic processing of advisories
- (Semi-) Automated processing of advisories
- Manual processing of advisories

Requirements for tools and processes
- Security downtime accepted/ minimized/ mitigated
- Asset management system with unique Product IDs
- Asset management with smart search
- Web browser
Ecosystem

Vendor → CSAF Lister → CSAF Aggregator → User

- Pull from aggregator
- Pull directly from source
- Collect list of potential sources

BSI CSAF trusted provider

https://wid.cert-bund.de

https://aggregator.cert-bund.de
Structure of CSAF 2.0 Specification

- Introduction & Design Considerations
- Schema elements
- Profiles
- Additional Conventions
- Tests
  - Mandatory
  - Optional
  - Informative
- Distributing CSAF documents
- Safety, Security, and Data Protection Considerations
- Conformance
Where to find more information?

https://cSAF.io

- CSAF GitHub: [https://github.com/oasis-tcs/csaf](https://github.com/oasis-tcs/csaf)
- CSAF 2.0 JSON Schema: [https://docs.oasis-open.org/csf/v2.0/csf_json_schema.json](https://docs.oasis-open.org/csf/v2.0/csf_json_schema.json)
- CSAF 2.0 Prose: [https://docs.oasis-open.org/csf/v2.0/csf-v2.0.html](https://docs.oasis-open.org/csf/v2.0/csf-v2.0.html)
- CSAF 2.0 Examples: [https://github.com/oasis-tcs/csaf/tree/master/csaf_2.0/examples](https://github.com/oasis-tcs/csaf/tree/master/csaf_2.0/examples)
- CSAF author guide: [https://secvisogram.github.io/secvisogram-documentation/](https://secvisogram.github.io/secvisogram-documentation/)
- Secvisogram online editor: [https://secvisogram.github.io](https://secvisogram.github.io)
Tools developed by the community

• CSAF producer: https://github.com/secvisogram/secvisogram or https://github.com/mfd2007/yace
• CSAF content management system: https://github.com/secvisogram/secvisogram + https://github.com/secvisogram/csaf-cms-backend (WIP)
• CSAF trusted provider: https://github.com/csaf-poc/csaf_distribution
• CSAF aggregator: https://github.com/csaf-poc/csaf_distribution (WIP)
• Provider checker: https://github.com/csaf-poc/csaf_distribution (WIP)
• CSAF management system: open for commercial and Open Source tools
• CSAF asset matching system: open for commercial and Open Source tools
• CSAF modifier: custom implementation
• CSAF downloader: https://github.com/csaf-poc/csaf_distribution
• CSAF full validator: https://github.com/secvisogram/csaf-validator-service
Prerequisites fulfilled? Set up the following

- Laptop with WLAN
- Connect to workshop WLAN
- Access webservers
- Familiar with Linux shell

Material at [https://files.test](https://files.test)
yace

- Simple online editor to create CSAF documents
- Limited functionality
Exercise: Simple CSAF advisory

- [https://yace.test](https://yace.test)
- [https://files.test/Exercise_01](https://files.test/Exercise_01)
- Instructions as Markdown file
Product Tree

- Hierarchical structure to convey products
- Usually: use branches with categories vendor, product_name, product_version
- product_identification_helper used for better comparison
Revision History

- Integer vs semantic version!
- New entry for each public revision
- `legacy_version` to link to “non-conforming” versions from human-readable advisories
- `current_release_date` set from date of last item in `revision_history`
- `version` must be updated as well
Document vs Vulnerabilities

- Different elements *(acknowledgments, notes, references)* exist in both
- Document: applies to the whole document
  - Legal disclaimer, Product descriptions,…
- Vulnerabilities: applies only to this specific vulnerability
  - Vulnerability description
Note categories

- **Summary**: two to three sentences, barely any technical detail
- **Description**: one to three paragraphs with (possibly) some technical detail
- **Details**: detailed description, exceeding the length of a description and go deeper into technical details
Secvisogram

- First open source tool conforming CSAF producer
- Structure based on the JSON of the standard
- Different views to reduce complexity
- Backend option available
- Templates possible
- https://github.com/secvisogram/secvisogram/
CSAF CMS Backend

- Automatically assign IDs
- Track revision history and version
- Track changes
- Use workflows
- Use company logos
- Use templates
- *Publish from there automatically (WIP)*
- https://github.com/secvisogram/csaf-cms-backend/
Exercise: Update advisory

- https://secvisogram.test
- https://files.test/Exercise_02
- Instructions as Markdown file
Product Tree

- Hierarchical structure to convey products
- Usually: use branches with categories vendor, product_name, product_version
- product_identification_helper used for better comparison
Exercise: Add product

- [https://secvisogram.test](https://secvisogram.test)
- [https://files.test/Exercise_03](https://files.test/Exercise_03)
- Instructions as Markdown file
Product version ranges

- vers vs vls
- Use only if
  - Only insufficient data is available
  - Enumeration is hard and vers is well-defined
- Can’t be used with CPE
- Careful with implications of vers:all/*
Exercise: Add product with version range

- https://secvisogram.test
- https://files.test/Exercise_04
- Instructions as Markdown file
Hardware and Software

- Separate hardware from software
- Bind them via relationships
Exercise: Describe hardware and software

- [https://secvisogram.test](https://secvisogram.test)
- [https://files.test/Exercise_05](https://files.test/Exercise_05)
- Instructions as Markdown file
Product status

- first_affected
- known_affected
- last_affected
- first_fixed
- fixed
- known_not_affected
- under_investigation
- recommended

contradicting groups

affected
fixed
not affected
under investigation
The Zephyr project received notification of this vulnerability through CERT before the publication date. We analyzed these vulnerabilities, and any affected code, and concluded that the Zephyr project is not impacted by any of these vulnerabilities, neither in the current releases, nor in any Long Term Support release.
Not affected but...

- ...still costumers call / write tickets
- ...nobody reads the press release
- Huge effort for hotlines / support
Already investigating but…

- …still costumers call / write tickets
- …nobody reads the forum entry
- Huge effort for hotlines / support
VEX

- Vulnerability Exploitability eXchange
- Communicate product status explicit
- Machine-readable to address scalability

- No a specification but implementations in different standards
  → CSAF profile
“Modality” of VEX – Use Cases
Benefits

▪ Save money!
▪ Reduce human workload!
▪ Spread the word about the good work your PSIRT does!
▪ Use established tools and distribution
Minimum fields – VEX document metadata

```json
{
    "category": "csaf_vex",
    "csaf_version": "2.0",
    "publisher": {
        "category": "vendor",
        "name": "Example Company ProductCERT",
        "namespace": "https://psirt.example.com"
    },
    "title": "Example VEX Document Use Case 001 - Not Affected",
    "tracking": {
        "current release date": "2023-03-03T11:00:00.000Z",
        "id": "EVD-2023-001",
        "initial release date": "2023-03-03T11:00:00.000Z",
        "generator": {
            "engine": {
                "name": "Secvisogram",
                "version": "2.1.0"
            }
        },
        "revision_history": [
            {
                "date": "2023-03-03T11:00:00.000Z",
                "number": "1",
                "summary": "Initial version."
            }
        ],
        "status": "final",
        "version": "1"
    }
}
```

VEX format identifier

Author [author] and Author role [author_role]

Timestamp last updated [doc_time_last_updated]
Document ID [doc_id]
Timestamp first issued [doc_time_first_issued]

Tooling [tooling]

Document version [doc_version]
Minimum fields – VEX statement metadata

- Statement ID [statement_id]: given through CSAF structure (e.g. /vulnerabilities[2]/product_status/known_affected)
- Inherited from document:
  - Statement version [statement_version]
  - Timestamp first issued [statement_time_first_issued]
  - Timestamp last updated [statement_time_last_updated]
Minimum fields – Product details

```
"product_tree": {
  "branches": [
    {
      "branches": [
        {
          "category": "product_version",
          "name": "4.2",
          "product": {
            "name": "Example Company ABC 4.2",
            "product id": "CSAPPID-0001",
            "product_identification_helper": {
              "cpe": "cpe:/example:ABC:4.2",
              "hashes": [
                "purl": "pkg:x-example/ABC@4.2",
                "sbom urls": [
                  ...
                ]
              ]
            }  
          }
        },
        {
          "category": "product_name",
          "name": "ABC"
        }
      
      "category": "vendor",
      "name": "Example Company"
    }
  
  "category": "supplier",
  "name": "Example Company"
}
```

Product details with Product identifier [product_id]

Human-readable product name
Product Identifier for reference within document

Different identification methods

Supplier [supplier]
Product statuses

- Not Affected $\rightarrow$ impact statement
- Affected $\rightarrow$ action statement
- Fixed
- Under Investigation
Minimum fields – Vulnerability and Product status

Vulnerability identifier [vul_id]
Justification [justification]

Description [vul_description]

Status notes [status_notes]
Minimum fields – Vulnerability and Product status

```
"product_status": {
  "known not affected": [
    "CSAFFID-0001"
  ],
"threats": [
  {
    "category": "impact",
    "date": "2023-03-03T11:00:00.000Z",
    "details": "Class with vulnerable code was removed before shipping."
  },
  "product_ids": [
    "CSAFFID-0001"
  ]
}
```

Status [status]

Impact statement [impact_statement] with Timestamp of impact statement [impact_statement_time]
How to link to an SBOM?

Product identification helpers:
- Retrievable SBOM

```json
"sbom_urls": {
  //...
  "items": {
    "https://example.com/location-to-sbom"
  }
}
```
How to link to an SBOM component?

- CycloneDX

```json
"x_generic_uris": [
  {
    "namespace": "https://cyclonedx.org/capabilities/bomlink/",
    "uri": "urn:cdx:411dafd2-c29f-491a-97d7-e97de5bc289/1#pkg:maven/org.jboss.logging/jboss-logging@3.4.1.Final?type=jar"
  }
]
```

- SPDX

```json
"x_generic_uris": [
  {
    "uri": "https://swinslow.net/spdx-examples/example4/main-bin-v2#SPDXRef-libc"
  }
]
```
Status Justifications

- Component_not_present
- Vulnerable_code_not_present
- Vulnerable_code_cannot_be_controlled_by_adversary
- Vulnerable_code_not_in_execute_path
- Inline_mitigations_already_exist
Exercise: Create a VEX

- https://secvisogram.test
- https://files.test/Exercise_06
- Instructions as Markdown file
Next steps

- Publish CSAF on your website
- Integrate it into your workflow
- Become a CSAF trusted provider
- Submit yourself to a CSAF lister / aggregator: [https://wid.cert-bund.de/portal/wid/csaf/submit](https://wid.cert-bund.de/portal/wid/csaf/submit)
- Spread the word! #oCSAF
Thank you!

https://csaf.io