2026
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
UNITED STATES
SANTA CLARA, CA, USA
MARCH 2 - 5, 2026

Threat Modeling for SoC Security Design: IEEE P3164 SA-EDI Standardization

Wenzhen Li
IEEE P3164 Working Group

Outline of the presentation

- Introduction of IEEE P3164 standardization
  - What's SA-EDI?
  - Motivation
- Threat modeling for SoC security design
- Deep dive of SA-EDI data objects.
- Intel case study
- Future work
- Summary and Q&A

Motivation

- SoCs power mobile, automotive, IoT, AI, cloud systems
- Integration improves performance **but expands attack surface**
- Security is now a **primary design requirement**
- **Inconsistent threating modeling in hardware**
- Need **systematic, interoperable way for assets, weakness, and thread**
  - A **holistic and shift left methodology** is applied to SoC security design

Emerging Threats

- **Supply chain compromises**
  - Counterfeiting & IP theft
  - Hardware Trojans --- dormant, hard to detect
- **AI/ML accelerator vulnerabilities** – adversarial attacks, e.g., input data and training data poisoning
- **Quantum computer** -- Harvest now, decrypt later
- **Firmware/software exploits**
  - third-party and legacy component risk.
  - AI-powered attack
- **Architectural and Physical attacks**
  - memory row-hammer,
  - side-channel attack
  - Fault injection

These emerging threats demand a structured and repeatable threat modeling approach!!

Threat Modeling Process

1. Identify **critical assets & attack surface**
2. Define **threat actors & scenarios**
3. Apply frameworks
4. Perform **risk analysis**
5. Develop **countermeasures**

Framework Comparison for SoC

Framework	Core Focus	SoC-Specific Application	Best Used When
STRIDE	General application security: Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege.	Useful for modeling threats related to data flows and interfaces, such as firmware updates and communication protocols.	Starting the threat modeling process for a new feature or component.
PASTA	Risk-centric, attacker-focused methodology spanning seven stages. Integrates business objectives with technical analysis.	Provides a strategic, high-level view of how business and technical risks intersect in an SoC design, considering attacker motivation.	Integrating threat modeling into a broader risk management strategy.
MITRE ATT&CK	Real-world adversary tactics and techniques. Offers a detailed, behavior-focused view.	Mapped to the specific TTPs (Tactics, Techniques and Procedures) used to exploit firmware, devices, and systems controlled by a SoC.	Conducting a deep, intelligence-driven analysis of adversary behavior.
MITRE EMB3D	Critical infrastructure and embedded systems, building on ATT&CK.	Identifies threats specific to the unique hardware, software, and physical interfaces of embedded systems.	Designing an SoC for critical infrastructure, where the system itself is a high-value target.
Attack Trees	Visual, goal-oriented representation of attack paths.	Used to model multi-stage physical and logical attacks, such as combining a software vulnerability with a side-channel attack.	Analyzing specific, high-risk attack scenarios in detail.

TTP refers to the patterns of behavior and methods that threat actors use to carry out attacks. Mapping TTPs helps security teams understand the adversary's overall goals (tactics), the specific ways they achieve them (techniques), and the detailed steps they follow (procedures).

IEEE P3164 SA-EDI Standard

Goal:

Standardize security annotation for SoC design

Enables **interoperability** across IP providers, integrators, EDA vendors

JSON-based framework with **5 data object types** (machine-readable interoperable security collateral package: IP bundle):

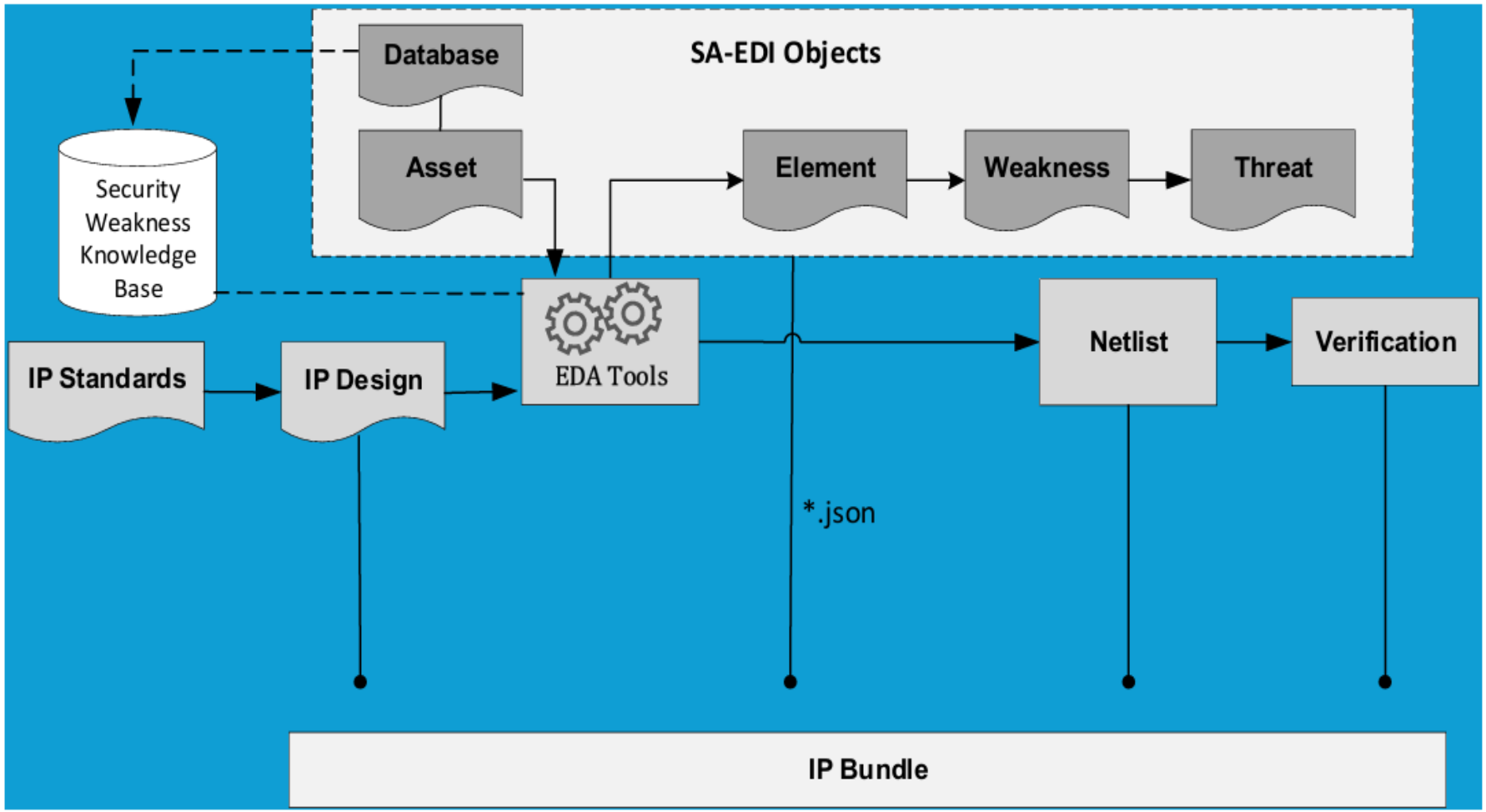
- Database,
- Asset,
- Element,
- Weakness,
- Threat

Enable **automation**, supports **AI/ML driven analysis, ensure the security intent preserved.**

Note: A little more history of SA-EDI to tell

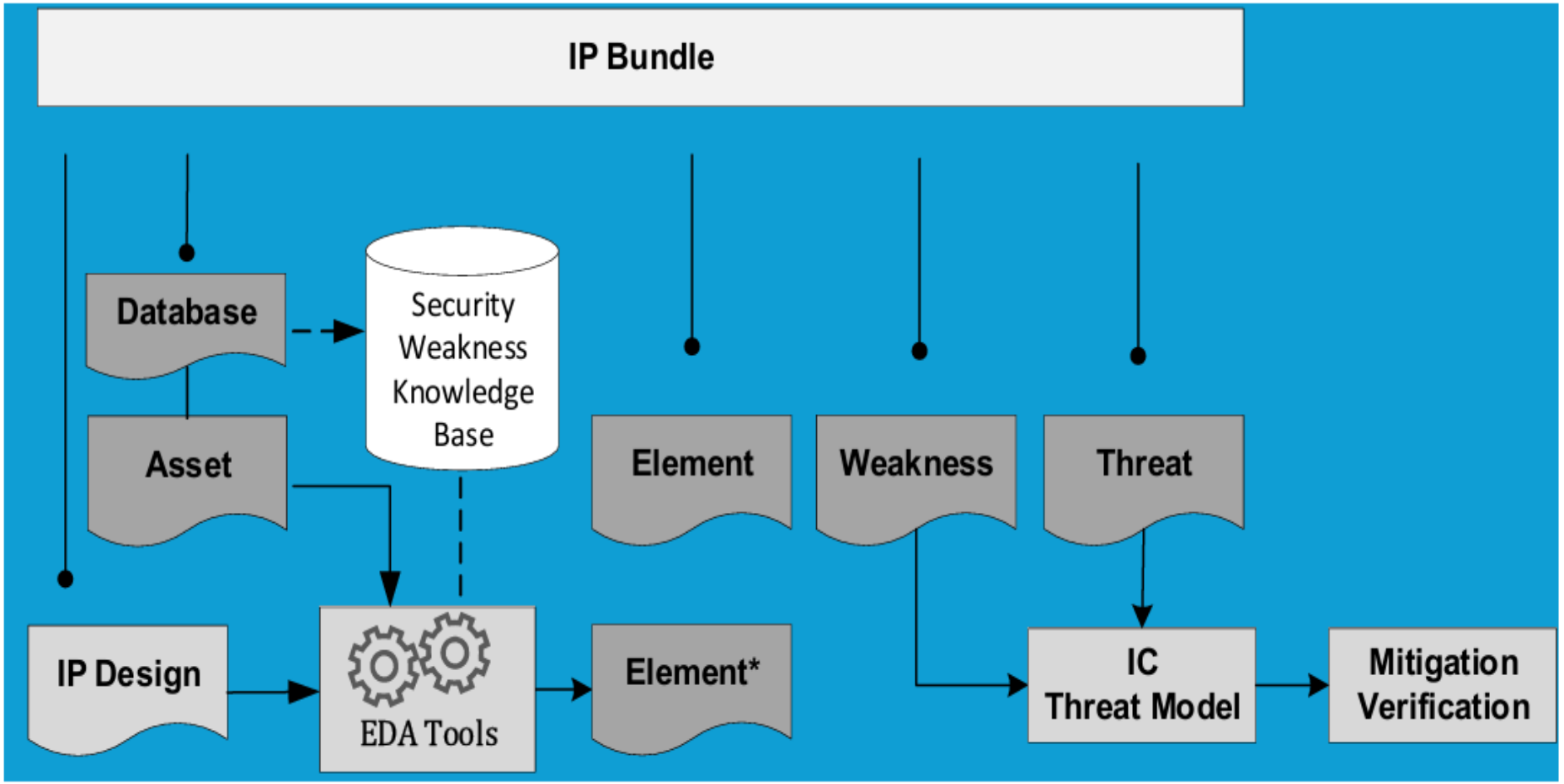
Threat modeling of SA-EDI: IP Provider's perspective

- **IP Provider** → creates security collateral (assets, weaknesses, threats)
- **Integrator** → validates & extends threat model
- **EDA Tools** → automate verification *(Insert simplified diagram of flow: IP → Integrator → EDA Tools)*



Threat modeling of SA-EDI: IP integrator's perspective

- **IP Provider** → creates security collateral (assets, weaknesses, threats)
- **Integrator** → validates & extends threat model
- **EDA Tools** → automate verification *(Insert simplified diagram of flow: IP → Integrator → EDA Tools)*



Asset data object

Attribute	Required	Type	Definition
Asset Name	Yes	String (case-sensitive)	Full hierarchical path name of the asset as defined in the RTL source
Description	No	String	Brief description about the asset (e.g., what makes it an asset, its purpose, etc.). This is not a required field however it is strongly recommended since it provides useful information to the IP integrator.
Condition	No	String (case-sensitive[4])	A condition expressed using SystemVerilog syntax that captures configuration values which cause an RTL object to be an asset. (e.g., PARAM_A == 256 && PARAM_B == 32)
Security Objective	No	Array of Strings	One or more security objectives the asset shall achieve
Family	Yes	Array of Strings	Describes the IP type or family. The values are listed in Table 1. The value may be the numeric string or string name. There may be more than one type that is applicable.
Classification	Yes	Array of Strings	Describes the asset type. The values are listed in Table 2. The value may be the numeric string or string name. There may be more than one type that is applicable.
Database_ID	No	Array of Strings	Reference to the attribute Database_ID as defined in Table 3

Databease data object

Attribute	Required	Type	Definition
Database_ID	Yes	String	A unique identifier that is associated with a SWKB. This may be the name of the database.
Description	No	String	Brief description of the database (e.g. how to use it, types of weaknesses, etc.)
URI	Yes	String	URI locator of the security weakness database
Version	Yes	String	Version identifier of the security weakness database

Element data object

Attribute	Required	Type	Definition
Asset Name	Yes	String	Reference to the attribute Asset Name as defined in Table 4
Direction	Yes	Enumeration	Defines the direction of the Ports attribute: 1. Input 2. Output 3. Side-channel
Security Objective	No	Array of Strings	Reference to the attribute Security Objective as defined in Table 4.
Ports	Yes	Array of Strings (case-sensitive[4])	Ports exposed at the integration level that influence or observe the behavior of an asset
Condition	No	String (case-sensitive[4])	Reference to the attribute Condition as defined in Table 4
Security Weakness Reference	No	Array of Strings	Security weakness reference(s) from the SWKB. The format is dependent on the format of the entries in the database.

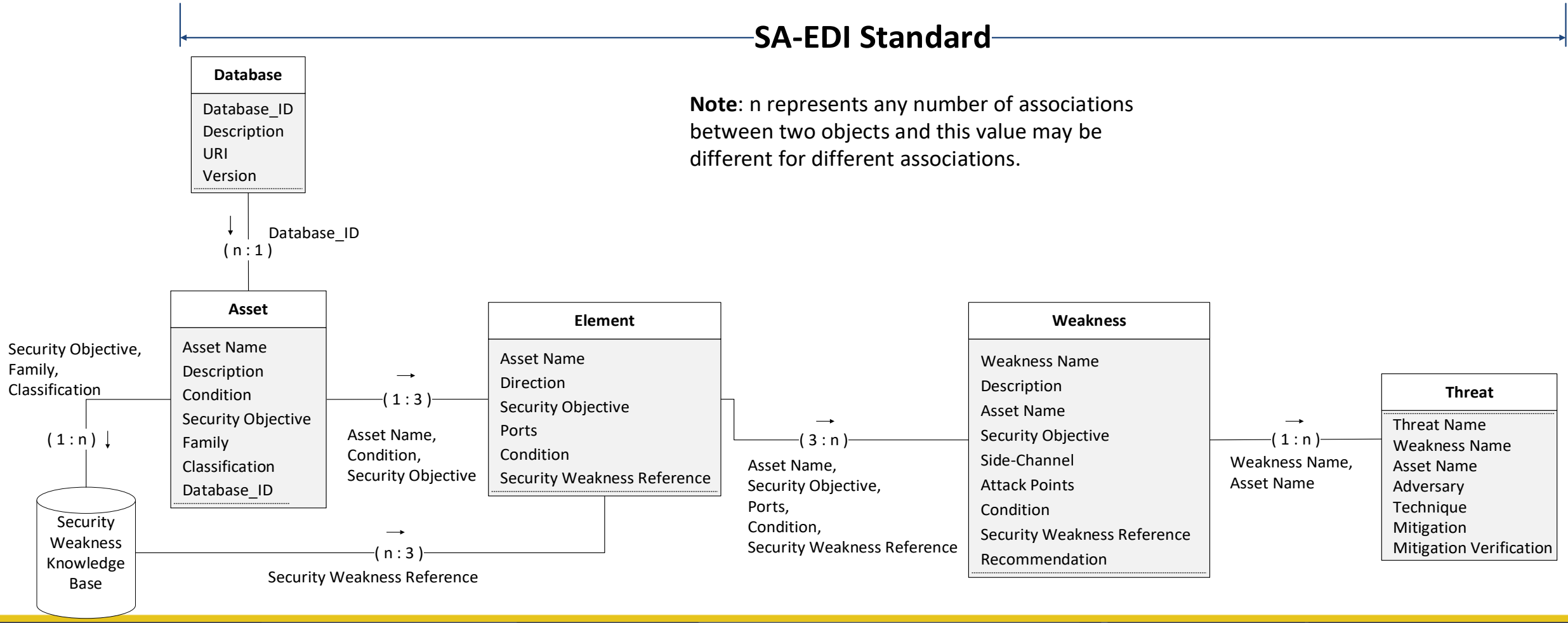
Weakness data object

Attribute	Required	Type	Definition
Weakness Name	Yes	String	Unique identifier of the weakness for Asset Name. The Weakness Name need not be unique across multiple assets.
Description	Yes	String	Details about the weakness and how the security objective may be compromised
Asset Name	Yes	String	Reference to the attribute Asset Name as defined in Table 5
Security Objective	Yes	Array of Strings	Reference to attribute Security Objective as defined in Table 5.
Side-Channel	Yes	Enumeration	Indicates if this data object captures a side-channel weakness: 1. Yes 2. No
Attack Points	No	Array of Strings	Ports listed in Table 5 that are associated with this security objective and the security weakness of this object
Condition	No	String	Reference to the attribute Condition as defined in Table 5
Security Weakness Reference	No	Array of Strings	Reference to attribute Security Weaknesses Reference as defined in Table 5.
Recommendation	No	Array of Strings	Additional information that the integrator needs to be aware of such as mitigations, configuration settings, etc.

Threat data object

Attribute	Required	Type	Definition
Threat Name	Yes	String	Unique identifier of the threat
Weakness Name	Yes	String	Reference to the attribute Weakness Name as defined in Table 6
Asset Name	Yes	String	Reference to the attribute Asset Name as defined in Table 6
Adversary	Yes	Array of Strings	The agent/attacker that can perform the technique detailed in the Technique attribute
Technique	Yes	String	Details about the attack itself such as actions required, equipment needed, etc.
Mitigation	No	Array of Strings	Details the IC protection mechanism(s) that defends against the threat
Mitigation Verification	No	Array of Strings	Details how to verify the mitigation(s) described in the Mitigation attribute

Deep dive of data objects in SA-EDI



Note: n represents any number of associations between two objects and this value may be different for different associations.

SA-EDI Workflow steps

Step#	Owner	Required	Details	Output
1	IP provider	No	Identify a database of known weaknesses and create a Database object(s).	Database Data Object
2	IP provider	Yes	Identify asset(s) in the IP and create an Asset object(s) for each asset.	Asset Data Object
3	IP provider	Yes	Using the output of steps #1 and #2, generate the Element object(s) using an EDA tool. This step may also be done manually.	Element Data Object
4	IP provider	Yes	Using the output of step #3, create a Weakness object(s).	Table 3. Weakness Data Object
5	IP provider	No	Using the output of step #4, create a Threat object(s).	Table 4. Threat Data Object
6	IP provider	Yes	Bundle all the data objects created in steps #1-5 into the IP delivery package (i.e., IP bundle).	IP Bundle
7	Integrator	Yes	Using the output of step #6, determine which Weakness objects are in scope for the IC's threat model.	IC Threat Model
8	Integrator	Yes	Using the IC Threat Model, verify the security objectives are met by the mitigations in the design	Mitigation Verification

Diagrams and tables are preliminary and subject to change. We are targeting a public review release of SA-EDI standard on Q2'26.

Intel Case Study (materials are courtesy of Intel)

- Intel developed **9 adversary models** for SoC design, including software, hardware and network
- Intel's security design combines **open-source + in-house tools**
  - Kernel Fuzzer for Xen : an open source resource designed to fuzz some of the most complex projects: Kernel internals and drivers.
  - Spectre and Meltdown Checker : an open source tool used by many end users and companies to scan their systems for transient execution side channel vulnerabilities. While Intel contributes to the Linux kernel and regularly creates reporting mechanism for end users to check the status of known issues, this tool provides more detailed information about those issues.
  - CVE Binary Tool helps users improve their supply chain security by listing known vulnerabilities associated with third-party components.
  - Penetration testing is akin to "ethical hacking" where engineers are organized into a "red team" with the purpose of attacking or breaking a product.
  - TPM2 (Trusted Platform Module 2.0) software stack for Linux, working with the hardware, software, and security ecosystems to support this technology.
  - AI-driven threat detection: Memory scanning involves analyzing a computer's active RAM for malicious code or patterns. This technique can identify known threats using signatures or detect unknown threats by spotting anomalous behavior, with technologies like Intel Thread Detection Technology (TDT) offloading the process to a GPU to improve performance

Intel Case Study (Cont')

- Intel also developed a new MITRE ATT&CK mapping: vPRO for its AI PC security design,
  - MITRE ATT&CK®
  - Intel vPro Landing - Mappings Explorer
- AI PC security design demonstrates **structured threat modeling adoption**

Key Trends in Hardware Security

- **Chiplet architectures** → new trust boundaries
- **PUFs** → unique device identity, resistance to cloning
- **Confidential computing** → protect data in use
- **AI/ML in security** → defense + new attack vectors
- **Quantum cryptography**

Future Work

Our goal: tell a compelling story about SA-EDI standard

- The one shows how to enable threat modeling and security verification automation.
- We welcome collaboration from IP vendors, integrators, and EDA tool providers.
  Your contributions will help shape the next generation of secure SoC design.

Some thoughts on Future work

- There are known and unknown threats for SoC security design.
- Known threats are relatively straightforward:
  - pull in SA-EDI collateral, use open-source or proprietary tools, and verify mitigations.
- Unknown threats are the real challenge.
  - We need ways to emulate white-hat adversaries, build hierarchical threat models, and leverage AI agents for attack simulation.
- This is where SA-EDI can become a foundation for future innovation.

Hierarchical threat mitigation and verification

We envision a tiered approach:

Tier1: A lot of security design issues are due to repeated developmental flaws that happen over and over again. These CWE (common weakness enumeration) can be easily exploited by hackers.

- Use of a security coding standard.
- Static analysis, and CWE-based checks.

Tier2:

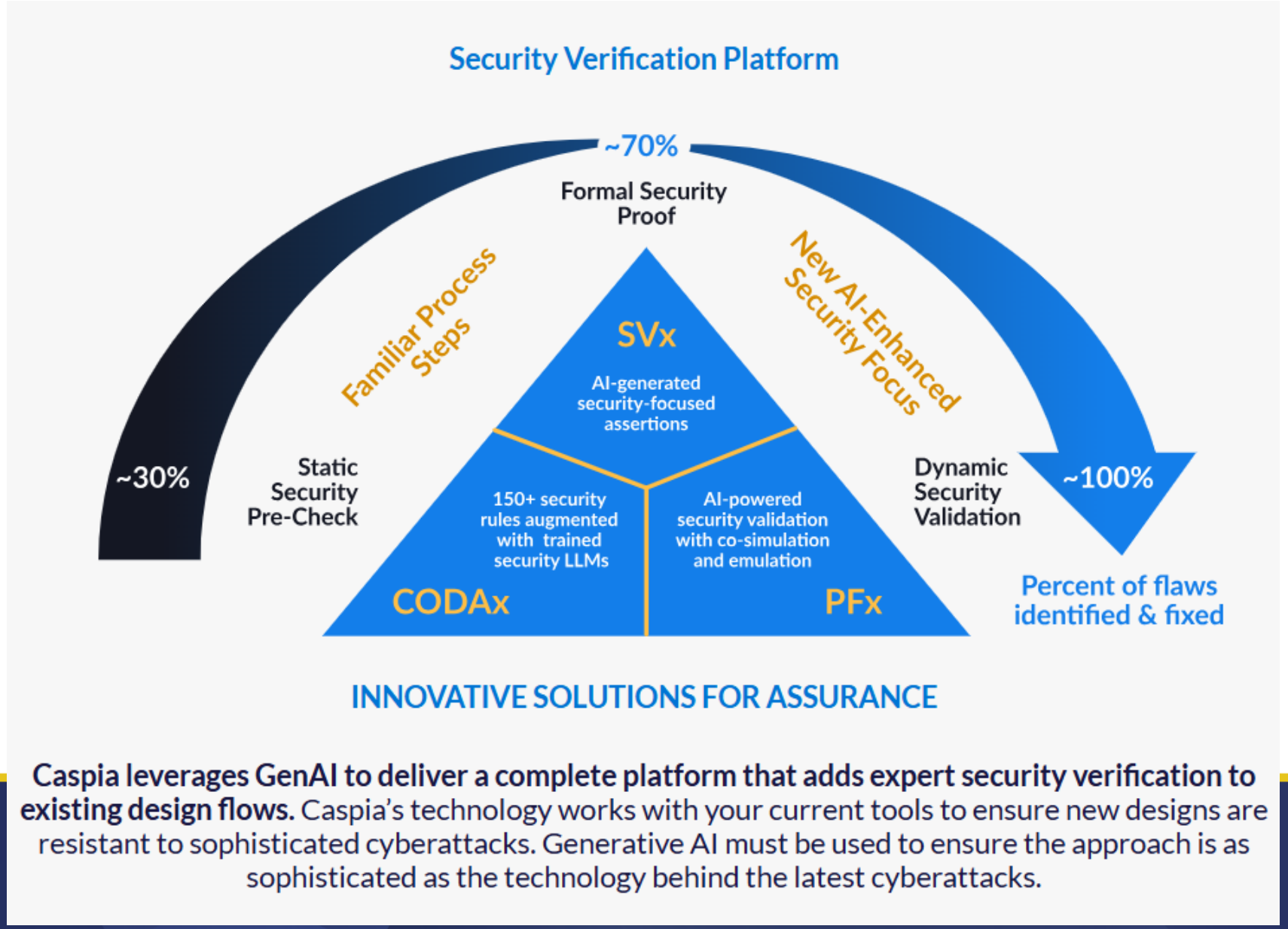
- Formal verification enhanced with SA-EDI annotations.
- Coverage and equivalence checking tools compliant with SA-EDI standard.

Tier3:

- Fuzzing (randomize the test vectors and test patterns) and AI-driven threat simulation leveraging SA-EDI data objects.

This hierarchy supports **both breadth and depth in security verification**.

A security verification example from Caspia (courtesy)



Caspia leverages GenAI to deliver a complete platform that adds expert security verification to existing design flows. Caspia's technology works with your current tools to ensure new designs are resistant to sophisticated cyberattacks. Generative AI must be used to ensure the approach is as sophisticated as the technology behind the latest cyberattacks.

Summary

- Threat modeling is **essential for resilient SoC design**
- IEEE P3164 SA-EDI → **uniform, interoperable framework**
- Aligns industry stakeholders (IP providers, integrators, EDA vendors)
- Prepares SoC security for **AI, quantum, chip-let architecure** challenges

Q&A

Thank you!

Contact:

Chair of IEEE P3164: Brent M Sherman, brent.m.sherman@intel.com

or Wenzhen Li, lwzh@hotmail.com

IEEE P3164 Working Group