Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.083308

ARTICLE

A Model-Driven Approach to Secure Device Onboarding Using a Device Security Passport

Sara Matheu[1,*], **Pedro Ruzafa**[1], **Ilias Kalouptsoglou**[2], **Antonio Skarmeta**[1] **and Dionysios Kehagias**[2]

[1]Department of Information and Communications Engineering, Faculty of Computer Science, University of Murcia, Murcia, Spain
[2]Centre for Research and Technology Hellas, Information Technologies Institute, Thessaloniki, Greece

*Corresponding Author: Sara Matheu. Email: saranieves.matheu@um.es

Received: 01 April 2026; Accepted: 31 May 2026; Published: 30 June 2026

ABSTRACT: The evolution of the Future Mobile Internet, driven by large-scale connectivity and heterogeneous device ecosystems, introduces significant challenges for securely integrating devices into operational environments. Existing onboarding mechanisms primarily focus on authentication and credential provisioning, while security policy enforcement is typically deferred, creating a temporal gap during which devices may operate without appropriate constraints. This paper addresses this limitation by enabling policy enforcement during onboarding. To this end, we propose a model-driven approach that integrates the Device Security Passport (DSP) with the FIDO Device Onboard (FDO) protocol. The DSP is a lifecycle-aware model that aggregates heterogeneous security descriptors, including component inventories, behavioral policies, and vulnerability information, into a structured and interoperable representation. The approach leverages the FDO onboarding channel to retrieve and process DSP data at bootstrap time, enabling automated policy translation and enforcement. The method is evaluated in a realistic Smart Home environment through phase-level performance analysis. Results show that, although the proposed approach introduces an additional onboarding overhead of around 5 s in the evaluated scenario, this cost is incurred only once during device provisioning. Compared to manual onboarding, the approach reduces deployment time from minutes to seconds while enabling immediate policy compliance. These findings provide evidence that the proposed approach effectively bridges the gap between provisioning and enforcement with a limited performance impact in the evaluated scenario.

KEYWORDS: Secure IoT onboarding; future mobile internet security; FIDO device onboard; device security passport; security policy enforcement; zero-touch provisioning; lifecycle security management

1 Introduction

The rapid evolution of the Future Mobile Internet, driven by paradigms such as 5G/6G, edge computing, and AI-enabled services, is enabling highly dynamic and large-scale interconnected ecosystems. This transformation is reflected in the accelerated growth of the Internet of Things (IoT) market, projected to increase from $546 billion in 2022 to nearly $991 billion by 2028 [1]. In these environments, billions of heterogeneous devices are expected to interact across distributed and often untrusted domains.

A critical requirement in such scenarios is secure device onboarding, i.e., the process through which devices are authenticated, provisioned, and integrated into operational environments. Existing onboarding mechanisms, such as the Device Provisioning Protocol (DPP) [2], the Agile Secure Onboarding Protocol (ASOP) [3], the Bootstrapping Remote Secure Key Infrastructure (BRSKI) [4], and FIDO Device Onboard (FDO) [5], provide secure identity establishment and automated provisioning capabilities. However, these

approaches focus primarily on authentication and credential management, while policy enforcement is typically addressed after onboarding. As a result, devices can temporarily operate without appropriate restrictions, in conflict with secure-by-default principles and emerging regulations such as the European Cyber Resilience Act (CRA) [6].

Enabling policy-aware onboarding requires structured and machine-readable security information describing device composition, behavior, and vulnerability status. Such information is already available through security descriptors such as Software Bills of Materials (SBOMs) [7], Manufacturer Usage Descriptions (MUDs) [8], and Vulnerability Exploitability eXchange (VEX) statements [9]. However, these descriptors are heterogeneous, independently managed, and not directly integrated into onboarding workflows, limiting interoperability and automated policy enforcement across the device lifecycle.

To address these limitations and building upon our previous work presented at MobiSec25 [10], this paper proposes the Device Security Passport (DSP), a machine-readable and lifecycle-aware model that integrates heterogeneous security descriptors through a unified and reference-based representation. Rather than duplicating existing artifacts, the DSP enables the referencing of external security information while preserving interoperability and supporting the gathering and fetching of device security data across manufacturing, deployment, and operation.

In addition, the DSP is integrated with the FDO protocol to make security descriptors available during the onboarding process. Rather than using the FDO *ServiceInfo* mechanism solely to exchange onboarding metadata, the proposed approach leverages it to retrieve and consume structured security information associated with the device. The retrieved descriptors are then processed to derive and enforce security policies before the device becomes fully operational. As a result, onboarding is extended beyond identity establishment and credential provisioning, enabling devices to enter the network in a controlled and policy-compliant state.

The main contributions of this work are:

- The formalization of the DSP as a structured and lifecycle-aware model derived from OSCAL, which enables the integration and referencing of heterogeneous security descriptors across the device lifecycle.
- A policy-aware onboarding architecture that extends FDO provisioning workflows with lifecycle-aware descriptor orchestration and onboarding-time policy enforcement using the DSP.
- The integration of automated policy enforcement directly into the onboarding process, removing the temporal gap between device provisioning and operational security.
- An experimental validation demonstrating the feasibility and performance impact of integrating DSP retrieval and policy enforcement into the FDO onboarding workflow.

The remainder of this paper is organized as follows. Section 2 reviews the state of the art in secure IoT onboarding and device security modeling. Section 3 presents the formal model of the DSP and its lifecycle design. Section 4 describes the proposed model-driven onboarding architecture and workflow, followed by a security analysis in Section 5. Section 6 reports the experimental validation and comparative analysis. Section 7 discusses the implications and limitations of the proposed approach, and finally, Section 8 concludes the paper and outlines directions for future work.

2 Related Work

Secure IoT deployment involves two closely related challenges: securely onboarding devices into operational environments and providing machine-readable security information to support risk assessment and policy enforcement throughout the device lifecycle. Although substantial progress has been made in both areas, onboarding mechanisms and security descriptor models have largely evolved independently.

This section reviews the state of the art in secure IoT onboarding and device security modelling, highlighting the persistent gap between device provisioning and security-context management.

2.1 Secure IoT Onboarding

The increasing scale and heterogeneity of IoT deployments have made secure device onboarding a critical requirement. Traditional approaches based on manual configuration, static credentials, or vendor-specific provisioning mechanisms are difficult to scale and prone to misconfiguration, increasing operational complexity and security risks [11].

Several onboarding approaches have been proposed to automate device commissioning across IoT environments. DPP [2] enables lightweight Wi-Fi onboarding using elliptic-curve cryptography, reducing manual configuration effort in local environments. Similarly, ASOP [3] provides a lightweight onboarding approach for constrained devices based on pre-shared credentials. However, both approaches primarily focus on network bootstrapping and provide limited support for structured security metadata integration or policy enforcement.

Other approaches emphasize stronger identity and trust establishment. BRSKI [4] relies on PKI-based onboarding using manufacturer-installed certificates (e.g., IDevID) [12], enabling automated onboarding in enterprise environments. Cloud-based solutions such as AWS IoT [13] provide scalable onboarding and integrated cloud-managed security controls, although they depend on proprietary infrastructures that may reduce portability and interoperability.

Among these solutions, FDO [5] has emerged as a suitable approach for large-scale and zero-touch provisioning scenarios. By supporting ownership transfer and late-binding, FDO enables devices to be securely associated with their operational domain at deployment time, even across distributed supply chains. In addition, FDO provides a secure mechanism for exchanging onboarding metadata through the ServiceInfo channel [14]. However, although FDO supports secure metadata exchange, policy enforcement remains external to the onboarding workflow. As a result, devices may temporarily operate without appropriate constraints until security policies are separately retrieved and deployed.

Overall, existing onboarding approaches focus on authentication and credential provisioning, while security metadata management and policy enforcement are typically performed as separate post-deployment activities. As a result, onboarding and security enforcement remain loosely coupled. This limitation has also been recognized by the NIST practice guide (SP 1800-36) [15], which emphasizes that onboarding should not be limited to credential provisioning but should support lifecycle protections such as policy enforcement and continuous security controls. The proposed DSP-FDO approach addresses this gap by integrating onboarding, security metadata retrieval, and policy enforcement within the same workflow. By leveraging the FDO *ServiceInfo* exchange, descriptors such as SBOMs, MUD profiles, and vulnerability information can be retrieved and processed during onboarding, enabling policies to be enforced before the device becomes fully operational. Table 1 summarizes the main strengths and limitations of representative onboarding approaches, including the proposed DSP-FDO model.

2.2 Security Frameworks for Modeling Device Security Information

Model-Driven Engineering (MDE) has been widely applied in cybersecurity to represent, exchange, and process security information through structured and machine-readable models [16,17]. In the IoT domain, this has led to the emergence of machine-readable security descriptors and frameworks designed to represent device composition, behavior, vulnerabilities, and compliance information [18,19]. These descriptors

support automated vulnerability assessment, policy enforcement, risk analysis, and regulatory compliance throughout the device lifecycle.

Table 1: Strengths and limitations of representative onboarding approaches.

Approach	Strengths	Limitations
DPP	Provides lightweight and automated Wi-Fi onboarding with secure credential establishment, reducing manual configuration effort in local deployments.	Primarily focused on network bootstrapping within Wi-Fi ecosystems. It does not support integrated security metadata management or policy enforcement during onboarding, limiting interoperability and lifecycle security integration.
BRSKI	Enables automated onboarding through strong PKI-based trust-establishment and manufacturer-issued device identities, making it suitable for enterprise deployments.	Strong dependence on pre-established PKI infrastructures and manufacturer trust anchors may reduce flexibility in heterogeneous or distributed environments. Security policies are enforced after onboarding rather than during bootstrap.
ASOP	Lightweight onboarding approach suitable for constrained IoT devices with low operational complexity.	Relies on pre-shared credentials, creating scalability and credential management challenges in large-scale deployments. Interoperability across vendors and domains is limited.
AWS IoT	Provides scalable cloud-native onboarding and integrated cloud-managed security controls for large deployments.	Highly dependent on proprietary cloud infrastructures and platform-specific mechanisms, reducing portability and increasing vendor lock-in. Security metadata integration remains platform-centric rather than interoperable.
FDO	Supports zero-touch onboarding, late-binding, and secure ownership transfer across distributed supply chains, enabling interoperable onboarding in heterogeneous environments.	Although FDO provides secure onboarding and metadata exchange capabilities, security policy enforcement remains external to the onboarding workflow. As a result, devices may temporarily operate without appropriate constraints until security policies are separately deployed and activated.

(Continued)

Table 1 (continued)

Approach	Strengths	Limitations
Proposed DSP-FDO	Combines secure onboarding with lifecycle-aware security metadata integration, enabling automated policy retrieval and enforcement during bootstrap. This reduces operational configuration effort while ensuring devices transition directly into a policy-compliant state.	Introduces additional onboarding overhead associated with DSP retrieval, verification, and policy translation, although this cost is incurred only during initial provisioning.

Composition Models. A first category of approaches focuses on device composition. Software Bills of Materials (SBOMs) [7] describe software components, versions, licenses, and provenance information, supporting transparency and vulnerability identification through correlation with databases such as CVE and NVD. Related efforts extend this concept to hardware and cryptographic assets. Hardware Bills of Materials (HBOMs) [20] improve hardware provenance and supply chain visibility, while Cryptographic Bills of Materials (CBOMs) [21] document cryptographic algorithms, libraries, and certificates to facilitate compliance verification and identification of weak or outdated cryptographic primitives.

Behavioral and Mitigation Models. Other approaches describe expected device behavior after deployment. IETF Manufacturer Usage Description (MUD) [8] specifies intended network communication patterns through machine-readable policies enforceable by network infrastructure, reducing the attack surface of constrained devices. NIST Threat MUD [22] extends this model by incorporating threat intelligence information to block communication with malicious endpoints and adapt to emerging risks.

Vulnerability Models. Additional formats focus on vulnerability disclosure and exploitability assessment. Vulnerability Disclosure Reports (VDRs) [23] enumerate known vulnerabilities and remediation status, while Vulnerability Exploitability eXchange (VEX) statements [9] indicate whether vulnerabilities are actually exploitable in a specific deployment context. Together, these descriptors help reduce false positives and provide more actionable information to operators and auditors.

Standardized Frameworks. Broader frameworks have also been proposed to structure and exchange security information. SCAP [24] supports automated vulnerability assessment and configuration checking through standards such as CVE, CPE, and XCCDF. More recently, OSCAL [25] introduces extensible machine-readable models for representing security controls, assessments, and compliance artifacts across organizations. OSCAL has gained attention in European regulatory and compliance automation initiatives, including EUROSCAL[1] and the ECSO OSCAL task force[2]. However, one of its main challenges is the initial learning curve and the need for organizations to adapt their existing processes to integrate OSCAL effectively. This transition can require significant time and resources, especially for organizations that are heavily reliant on traditional methods.

Despite the availability of these descriptors and frameworks, security information is typically fragmented across independent formats and repositories, limiting automation and hindering its use during onboarding and operation. To overcome this limitation, the proposed DSP provides a unified and lifecycle-aware model that integrates heterogeneous security descriptors through structured references. By reusing

[1] https://euroscal.eu/.
[2] https://ecs-org.eu/?publications=actions-beyond-words-automating-audits-for-streamlined-cybersecurity-compliance-in-europe.

existing standardized artifacts, the DSP supports interoperable security information exchange, automated onboarding, policy enforcement, and continuous security management.

Table 2 summarizes the strengths and limitations of representative security descriptors and frameworks compared with the proposed DSP model.

Table 2: Strengths and limitations of representative security descriptors and frameworks.

Approach	Strengths	Limitations
SBOM/HBOM /CBOM	Provide structured inventories of software, hardware, and cryptographic components, supporting transparency, vulnerability management, and supply chain traceability.	Focus primarily on composition information and do not provide behavioral, lifecycle, or policy enforcement capabilities.
MUD/Threat MUD	Enable machine-readable specification of expected network behavior and automated network policy enforcement.	Restricted to communication behavior and network-level controls, without broader vulnerability or lifecycle context.
VDR/VEX	Provide actionable vulnerability and exploitability information, improving risk assessment and remediation prioritization.	Limited to vulnerability reporting and do not support broader device security modeling or onboarding integration.
SCAP	Provides standardized formats for automated vulnerability assessment and configuration compliance checking.	Primarily oriented to enterprise vulnerability management rather than device-centric lifecycle and onboarding workflows.
OSCAL	Offers extensible and interoperable models for representing security controls, assessments, and compliance information across organizations.	Remains system-centric and introduces considerable complexity for constrained IoT environments and onboarding integration.
Proposed DSP	Provides a lightweight, device-centric, and lifecycle-aware model that integrates different security descriptors through structured references, enabling onboarding integration, policy enforcement, and continuous security management.	Depends on the availability and quality of externally maintained security descriptors, which may vary across vendors and deployment environments.

3 The Device Security Passport as a Lifecycle Security Model

The DSP is inspired by OSCAL, which organizes security information into three complementary layers (Fig. 1): (i) the *Control Layer*, defining security controls and profiles; (ii) the *Implementation Layer*, describing how controls are realized within systems; and (iii) the *Assessment Layer*, capturing findings, risks, and mitigation actions.

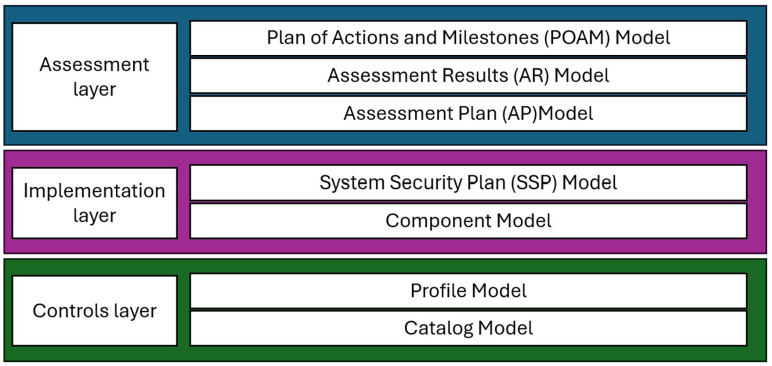

Figure 1: OSCAL layers and models.

Although OSCAL provides a comprehensive framework for structuring security information, OSCAL does not define how security artifacts are associated with individual devices, how they evolve throughout the device lifecycle, or how they can be directly consumed during provisioning. Moreover, the richness and flexibility of OSCAL models often introduce a level of complexity that may hinder their adoption. The DSP addresses these limitations through a lightweight, device-centric, and lifecycle-aware abstraction tailored to IoT environments. Instead of redefining existing security models, it selectively reuses relevant OSCAL concepts and extends them to support security descriptors and regulatory requirements such as the Cyber Resilience Act (CRA).

The following subsections formalize the DSP model, while Section 4 describes its integration into the onboarding workflow. Moreover, for the sake of readability and understandability, Table 3 summarizes the main mathematical symbols used throughout the DSP formalization. In addition, a reference JSON schema of the DSP model is publicly available on GitHub[3].

Table 3: Summary of the main symbols used in the DSP formalization.

Symbol	Description
D	Set of devices.
$d \in D$	Individual device belonging to the device set.
$DSP(d)$	Device Security Passport associated with device d.
U	Globally unique identifier (UUID) of the DSP.
M	Metadata section of the DSP, including versioning, roles, parties, and references.
S_{id}	Unique system identifier binding the DSP to a physical device.
L_d	Local security definitions associated with the device.
$C(d)$	Set of device components.
$A(d)$	Set of assessment assets associated with the device.
$O(d)$	Set of security observations obtained from assessments.
$R(d)$	Set of risks associated with the device and its components.
D_{sec}	Domain of security descriptors integrated into the DSP.
D_{BOM}	Set of composition descriptors (e.g., SBOM, HBOM, CBOM).
D_{BEH}	Set of behavioral descriptors (e.g., MUD profiles).
D_{VUL}	Set of vulnerability-related descriptors (e.g., VEX, VDR).

(Continued)

[3] https://github.com/saranieves92/DSP.

Table 3 (continued)

Symbol	Description
D_{MIT}	Set of mitigation descriptors (e.g., Threat MUD).
$\mathcal{L}_i$	Set of references linking a component to external security descriptors.
r_j	Individual risk associated with the device.
Rem_{r_j}	Set of remediation actions associated with risk r_j.
$genDSP$	Generic DSP created by the manufacturer.
$IDSP_1$	Instance-level DSP associated with a specific physical device.
$IDSP_2$	Operational DSP enriched during deployment and operation.
Δ_t	Incremental security updates incorporated during runtime in the $IDSP_2$.
$\oplus$	Merge operation used to update the $IDSP_2$.
$GUID_{FDO}$	FDO device identifier associated with the onboarding process.
c_{fdo}	DSP component representing onboarding-related information.
$\mathcal{P}_{fdo}$	Properties associated with the onboarding component.
$\mathcal{L}_{fdo}$	References to onboarding artifacts (e.g., ownership vouchers).

3.1 Structural Representation

From a structural perspective, the DSP is organized into several key sections derived from and aligned with existing OSCAL models, primarily the Plan of Actions and Milestones (POA&M), Component Definition, and Assessment models. This selective reduction addresses the inherent complexity of OSCAL and improves its practical applicability. In addition, the DSP avoids redundancy by not duplicating OSCAL information already provided by external security descriptors (e.g., SBOM, MUD, or VEX).

Formally, for a device $d \in D$, the DSP is defined as:

$$DSP(d) = \langle U, M, S_{id}, L_d \rangle \tag{1}$$

where U is a globally unique identifier (UUID) of the passport, M contains metadata, S_{id} is the device ID that binds the passport to a physical device, and L_d captures security-related local definitions.

These elements are derived from and aligned with OSCAL models but are selectively simplified to retain only the information required for the DSP. The *UUID* (U) and *metadata* (M) reuse common structures present across all OSCAL models. The UUID is intended to be a machine-oriented, globally unique identifier (e.g., RFC 4122 [26]) that can be used to reference this DSP elsewhere. The metadata element captures the contextual information required to identify, version, and track the evolution of the DSP. It includes descriptive attributes (e.g., title and version), temporal information (e.g., publication and last modification timestamps), and structured revision histories that record updates together with their associated context. To support accountability and governance, metadata also defines roles and associated parties (e.g., manufacturers, integrators, and operators), together with their responsibilities. This establishes a foundation for access control and coordinated lifecycle management across stakeholders. Finally, metadata supports references to external documentation through links, facilitating integration with complementary information or artifacts. The *System ID* (S_{id}) is derived from the POA&M model and establishes a verifiable binding between the logical passport and the physical device through globally unique identifiers and associated cryptographic material.

The local definitions section aggregates four main elements:

$$L_d = \{C, A, O, R\} \tag{2}$$

where C represents components, A represents assessment assets, O represents observations, and R represents risks. *Components* are aligned with both the POA&M and Component Definition models, *assessment assets* are derived from the POA&M and Assessment Plan models, while *observations* and *risks* follow the structure defined in the Assessment Results model. For clarity, a simplified excerpt of the DSP structure is shown in Listing 1, where only representative fields are included.

Listing 1: Simplified DSP structure.

```
{
  "uuid": "714210d2-f8df-448c-be3e-e2213816cf79",
  "metadata": {
    "title": "SmartBulb generic DSP",
    "version": "1.0",
    "last-modified": "2026-03-01T13:57:28",
    "dsp-model-version": "1.0",
    "links": [
     {"rel": "EU conformity", "href": "..."}
   ],
   "roles": [...],
   "parties": [...],
   "responsible-parties": [...]
  },
  "system-id": {
    "identifier-type": "UUID",
    "id": "123e4567-e89b-12d3-a456-426614174000"
  },
  "local-definitions": {
   "components": [...],
   "assessment-assets": {
     "components": [...],
     "assessment-platforms": [...]
   },
   "observations": [...],
   "risks": [...]
 }
}
```

As mentioned before, the DSP adopts a reference-based architecture in which security artifacts such as SBOM, MUD, or VEX documents are linked rather than embedded, preserving their original structure and ownership. Let $\mathbb{D}_{sec}$ denote the domain of security descriptors analyzed in Section 2:

$$\mathbb{D}_{sec} = \mathbb{D}_{BOM} \cup \mathbb{D}_{BEH} \cup \mathbb{D}_{VUL} \cup \mathbb{D}_{MIT} \tag{3}$$

where $\mathbb{D}_{BOM}$ includes inventory descriptors such as SBOM, HBOM, or CBOM, $\mathbb{D}_{BEH}$ includes behavioral descriptors such as MUD profiles, $\mathbb{D}_{VUL}$ includes vulnerability evidence descriptors such as VEX or VDR, and $\mathbb{D}_{MIT}$ includes mitigation descriptors. These artifacts are incorporated into the DSP through structured references within the local definitions $L_d = \{C, A, O, R\}$, each capturing a specific aspect of the device security state. The integration follows a mapping aligned with the semantics of each descriptor type. In particular, composition and behavioral descriptors (e.g., SBOM, HBOM, and MUD) are associated with *components*, vulnerability descriptors (e.g., VEX and VDR) are linked to *risks*, mitigation artifacts (e.g., Threat MUD) are incorporated in *remediations* and assessment-related outputs are captured through *observations*. We now describe how these elements are formally represented.

Let $C(d)$ denote the set of components associated with device d:

$$C(d) = \{c_1, c_2, \ldots, c_n\} \tag{4}$$

Each component c_i is described by a set of attributes and properties such as its UUID, type, title, description, purpose and status, and by a set of links $\mathcal{L}_i$. OSCAL information regarding ports and protocols has been removed, as it is already present in the MUD file.

The set $\mathcal{L}_i$ encodes references to external security descriptors in $\mathbb{D}_{BOM} \cup \mathbb{D}_{BEH} \subset \mathbb{D}_{sec}$, thereby associating each component with its composition and expected behavior. Formally, each reference is defined as $\mathcal{L}_i = \{(href_k, rel_k, media\_type_k)\}$, where $href_k$ identifies the external artifact, rel_k specifies the relationship between the component and the referenced descriptor, and $media\_type_k$ indicates the format of the referenced artifact, enabling its correct interpretation and processing. Listing 2 illustrates how SBOM and MUD descriptors are linked to a component.

Listing 2: Component linking SBOM and MUD descriptors

```
{
 "components": [{
   "uuid": "comp-1",
   "type": "software",
   "props": [{"name": "vendor", "value": "ACME"}],
   "links": [
    {"rel": "SBOM", "href": "...", "media-type":
      "application/vnd.spdx+json"},
    {"rel": "MUD", "href": "...", "media-type":
      "application/mud+json"}
   ]
 }]
}
```

The set $A(d)$ captures assessment assets, including tools and auxiliary components used during security evaluations or mitigation processes. The set $O(d)$ represents observations derived from these assessments, providing structured evidence and references to external reports. Together with the set of risks $R(d)$, these elements are aligned with the OSCAL assessment model, from which the DSP adopts its representation of assessment results and security findings.

The set $R(d)$ models security risks associated with the device and its components. Each risk is defined as $r_j = \langle \mathcal{P}_{r_j}, \mathcal{L}_{r_j}, Rem_{r_j} \rangle$, where $\mathcal{P}_{r_j}$ represents intrinsic properties of the risk (e.g., description, status, threat ID) following the OSCAL assessment model, and $\mathcal{L}_{r_j} \subset \mathbb{D}_{VUL}$ contains references to vulnerability evidence (e.g., VEX, VDR), linking each risk to externally maintained vulnerability information.

Each risk may include a set of remediation actions $Rem_{r_j} = \{rem_1, \ldots, rem_k\}$, which describe how the identified risk can be mitigated. Each remediation is defined as $rem_k = \langle \mathcal{P}_{rem_k}, \mathcal{L}_{rem_k} \rangle$, where $\mathcal{L}_{rem_k} \subset \mathbb{D}_{MIT}$ contains references to mitigation descriptors (e.g., Threat MUD profiles), enabling the association of risks with enforceable mitigation strategies.

Listing 3 provides a simplified fragment of the blocks discussed (assessment assets, observations and risks), showing how to reference external information (VEX, threat MUD).

This reference-based structure allows the DSP to integrate different security artifacts within OSCAL, maintaining their original structure and semantics without duplicating their content.

Listing 3: Example of assessment assets, observations and risks

```
{
  "assessment-assets": {
   "components": [{}],
   "assessment-platforms": [
     {"uuid": "3f7a6c8e-1111-45b9-9ae8-4c6b1b7a9f8e", "title":
      "CT-SAST"}]
  },
  "observations": [{
    "uuid": "0c4de4fc-9bde-46af-b6fe-3b5e78194dcf",
    "title": "Buffer overflow detected",
    "description": "...",
    "methods": ["TEST"],
    "origins": [{...}],
    "subjects": [{...}],
    "relevant-evidence": [{...}],
    "collected": "2026-01-02T12:14:16",
    "expires": "2026-06-02T12:14:16"
  }],
   "risks": [{
    "uuid": "8b8bae66-b28c-4fa5-9a20-b79e7322fc00",
    "title": "CVE-2024-7392",
    "description": "...",
```

```
      "links": [
       {"rel": "VEX", "href": "...", "media-type":
         "application/vnd.cyclonedx+xml"}],
      "status": "remediating",
      "origin": [{...}],
      "threat-ids": [{...}],
      "characterizations": [{...}],
      "deadline": "2026-10-02T12:14:16",
      "remediations": [{
       "uuid": "d28873f7-...",
       "lifecycle": "planned",
       "title": "Vendor response",
       "description": "Mitigation via Threat MUD.",
       "props": [{"name": "type", "value": "accept"}],
       "links": [
        {"rel": "Threat MUD", "href": "...", "media-type":
        "application/mud+json"}
       ]
     }],
      "risk-log": {..},
      "related-observations": [...]
     }]
  }
```

3.2 Credential and Identity Representation

The DSP uses *System ID* element from the POA&M model to bind security information to individual device instances. The passport UUID uniquely identifies the DSP itself, while the System ID identifies the associated physical device, establishing a verifiable link between both entities.

Credential-related information is not embedded directly in the DSP but referenced through external descriptors. For example, cryptographic assets can be described through linked CBOM artifacts, while authentication credentials remain managed by the underlying onboarding mechanism.

In addition, the DSP structure allows representing onboarding-related information, including credentials or bootstrap data, through the components block when required. This provides the flexibility to incorporate provisioning-specific information without modifying the core model, while keeping such data explicitly separated from the generic security descriptors. A detailed example of how to embed FDO data is described in Section 4.

3.3 Lifecycle Model

Unlike static or monolithic security descriptors, the DSP is conceived as a dynamic and incremental model that evolves together with the device it represents. As devices progress from manufacturing to

integration and finally to operational deployment, their ownership context, configuration, and risk exposure naturally change. Consequently, security information must also evolve while preserving traceability and consistency. The DSP follows this progression by continuously enriching the device's security profile over time rather than replacing it.

The DSP distinguishes three types of instances of Eq. (1) corresponding to different stages of the device lifecycle: a manufacturer-defined generic passport (genDSP), an instance passport bound to a specific device ($IDSP_1$), and an operational passport maintained during deployment ($IDSP_2$). These instances are not independent documents but successive refinements of the same structure, where each layer adds contextual information to the previous one, as illustrated in Fig. 2.

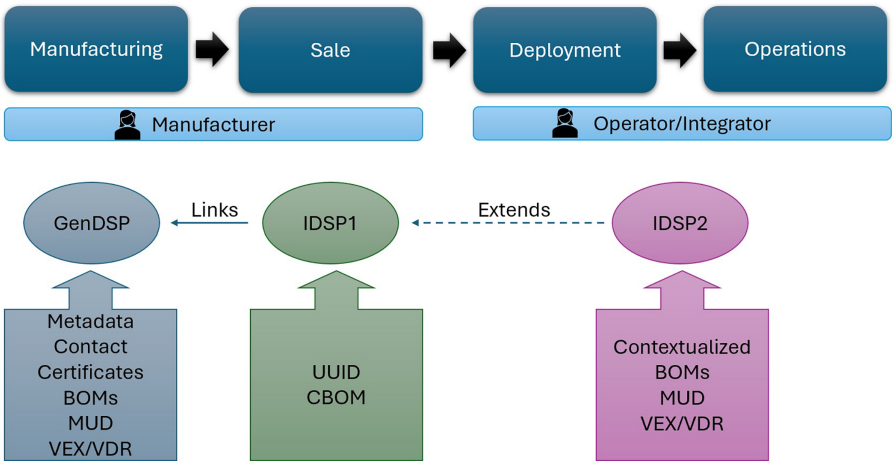

Figure 2: DSP lifecycle.

The **Generic DSP (genDSP)** acts as a baseline that describes static and general security information about a product. It is created by the manufacturer during the design and development phase and contains generic descriptors applicable to all devices of the same model. The genDSP is version-controlled to maintain historical traceability as vulnerabilities are disclosed or patches are released.

$$genDSP = \langle U_g, M_g, \varnothing, L_{d,g} \rangle \tag{5}$$

As devices move through the supply chain, the **Instance DSP ($IDSP_1$)** binds the generic passport to a specific physical device by introducing the system identifier S_{id}. This can be represented as a function:

$$\begin{aligned} instantiate &: genDSP \times S \rightarrow IDSP_1 \\ IDSP_1 &= instantiate(genDSP, S_{id}) \end{aligned} \tag{6}$$

so that $IDSP_1 = \langle U_{i1}, M_{i1}, S_{id}, L_{d,i1} \rangle$. This relation allows multiple instance passports to be derived from the same generic passport, each corresponding to a different device. To avoid duplication of information, the instance passport references the generic passport in the metadata (links) section, and only describes new modifications applicable to that device instance. In practice, $IDSP_1$ acts as the bridge between manufacturer-controlled information and deployment-specific contexts.

During operation, devices may diverge from their initial configuration due to updates, software extensions, hardware changes, or contextual security findings. The **Operational DSP ($\mathbf{IDSP_2}$)** captures this evolution by extending $IDSP_1$ with updated components, policies, observations, and vulnerabilities identified during runtime.

$$IDSP_2 = copy(IDSP_1) \quad IDSP_2 = \langle U_{i2}, M_{i2}, S_{id}, L_{d,i2} \rangle \tag{7}$$

Maintained by integrators or operators, this layer supports active security management and enforcement, for example by supplying policy information to network controls or contextual risk data to monitoring systems. At the same time, relevant findings can be fed back to the manufacturer to improve the generic baseline and provide mitigations, enabling a continuous feedback loop across the lifecycle. Operational updates are incorporated over time as $IDSP_2^{t+1} = IDSP_2^t \oplus \Delta_t$, where $\oplus$ denotes a merge operation that updates existing descriptors and integrates contextual information, including observations and security findings collected during device operation. This progressive enrichment of descriptors across the lifecycle leads to a hierarchical relationship between passport types, where each stage accumulates additional security information:

$$genDSP \subset IDSP_1 \subset IDSP_2 \tag{8}$$

As devices approach end-of-life, the DSP further serves as a comprehensive historical record of security-related events and actions. This accumulated evidence supports secure decommissioning procedures, such as revoking credentials and certificates, and facilitates audits, post-incident investigations, and compliance assessments. The passport also documents support periods and update availability, providing clear visibility into maintenance responsibilities and residual risks.

3.4 Storage and Retrieval of the DSP

DSP documents are managed through the Device Security Passport Platform (DSPP), which stores, updates, and provides controlled access to DSP instances across stakeholders. Manufacturers generate and maintain the generic ($genDSP$) and instance ($IDSP_1$) passports, while the operational passport ($IDSP_2$) is maintained by the operator and may reside outside the manufacturer domain. This separation reflects the different ownership and sensitivity of lifecycle information.

DSP retrieval is integrated into the onboarding workflow. During provisioning, the device provides a reference to its DSP, which is retrieved, validated, and processed by the operator infrastructure. This approach enables onboarding-time access to updated security information without embedding large artifacts within the device itself. DSPs may also be accessed independently by authorized stakeholders for auditing, inspection, or lifecycle management purposes. Access control is enforced through role-based mechanisms derived from DSP metadata (e.g., *roles* and *parties*) and external identity-management systems such as FEAM[4]. From a privacy and data protection perspective, the DSP follows a data-minimization approach in which security artifacts are referenced rather than embedded, allowing sensitive information to remain under the control of their original repositories.

Integrity and authenticity are ensured through the use of digital signatures applied at different stages of the DSP lifecycle. Each DSP instance is signed by the entity responsible for its creation or modification, enabling verification of both its origin and its content prior to use. In particular, the generic passport ($genDSP$) and the instance passport ($IDSP_1$) are signed by the manufacturer, while the operational passport

[4] https://iotac.eu/the-roadmap-of-feam-the-front-end-access-management-system/.

($IDSP_2$) is signed by the device operator after incorporating contextual updates. Formally, the integrity conditions for each passport type are defined as:

$$\begin{aligned} Integrity(genDSP) &\Leftrightarrow VerifySignature_{mfg}(genDSP) \\ Integrity(IDSP_1) &\Leftrightarrow VerifySignature_{mfg}(IDSP_1) \\ Integrity(IDSP_2) &\Leftrightarrow VerifySignature_{op}(IDSP_2) \end{aligned} \tag{9}$$

This mechanism enables DSP integrity and origin verification before the information is consumed during onboarding or operation.

4 Model-Driven Secure Onboarding

The proposed onboarding approach combines FDO with the DSP. FDO provides secure zero-touch provisioning, ownership transfer, and authenticated communication, while the DSP supplies the security descriptors required for policy enforcement during onboarding. As a result, policies are enforced immediately after onboarding, before the device obtains operational network access. The following subsections summarize the FDO protocol and describe the proposed DSP-based onboarding architecture.

4.1 FDO Foundations

As discussed in Section 2, FDO provides a standardized mechanism for secure and zero-touch device onboarding. In this work, FDO is used as the underlying protocol to establish a trusted communication channel and a verifiable ownership context between the device and the operator.

FDO enables devices to be provisioned with a uniform set of onboarding credentials during manufacturing and later commissioned into different operational environments through a late-binding process. The protocol relies on cryptographic identity and ownership evidence, allowing secure onboarding across distributed and potentially untrusted supply chains.

The onboarding process is organized into a manufacturing-time initialization phase (Device Initialization, DI) followed by a sequence of transfer-of-ownership protocols (TO0–TO2). During the DI phase, the *Manufacturer* provisions the *Device* with onboarding credentials and rendezvous information at manufacturing time. In the TO0 protocol, the *Owner* registers ownership metadata with the *Rendezvous Service*, enabling late-binding between devices and operational domains. Subsequently, in the TO1 protocol, the *Device* contacts the *Rendezvous Service* to discover the appropriate *Owner* endpoint. Finally, in the TO2 protocol, the *Device* and the *Owner* perform mutual authentication and establish a secure and encrypted communication channel, which is then used to deliver configuration data and provisioning artifacts.

A key capability of FDO for this work is the *ServiceInfo* exchange model, executed over the protected TO2 channel. ServiceInfo enables the structured and secure delivery of configuration data and external artifacts during onboarding. In the proposed approach, this mechanism is leveraged to retrieve and transport the Device Security Passport, enabling security descriptors to be available immediately after bootstrap.

4.2 FDO-DSP Integration

The integration of the DSP in FDO is implemented through the *ServiceInfo* exchange during TO2. Once the secure channel is established, the device provides a reference to its DSP. The *Owner* retrieves the corresponding DSP from the DSP platform, verifies its integrity, and processes the associated descriptors. Once retrieved, the DSP provides access to structured security descriptors $L_d = \{C, A, O, R\}$, including component inventories, behavioral policies, and security findings. These descriptors can be immediately

consumed by enforcement mechanisms within the operator domain, enabling policy-driven configuration at bootstrap time.

To ensure a consistent identity binding, the DSP system identifier is aligned with the FDO device identity:

$$S_{id} \equiv GUID_{FDO} \tag{10}$$

ensuring that the retrieved DSP corresponds unambiguously to the authenticated device.

The DSP can also represent onboarding-related information through a dedicated onboarding component:

$$c_{fdo} = \langle \mathcal{P}_{fdo}, \mathcal{L}_{fdo} \rangle \tag{11}$$

where $\mathcal{P}_{fdo}$ describes onboarding properties and $\mathcal{L}_{fdo}$ references onboarding artifacts such as Ownership Vouchers or discovery-related network policies (e.g., MUD profiles) that define how the device can reach the Rendezvous service during the discovery phase.

Listing 4 shows a simplified example of FDO-related information embedded in an $IDSP_1$.

Listing 4: Integration of FDO data in the $IDSP_1$

```
{
  "system-id": { "identifier-type": "urn:fdo:device-id",
           "id": "EDBA506B90E1420298485E91160CB16F" },

  "components": [
   {
    "type": "this-system",
    "title": "Home Assistant",
    "links": [
     { "rel": "CBOM", "media-type": "application/vnd.spdx+json",
        "href": "..." },
     { "rel": "MUD", "media-type": "application/mud+json", "href":
        "..." }
    ]
  },
  {
    "type": "service",
    "title": "FDO device onboard",
    "links": [
     { "rel": "ownership-voucher", "media-type": "application/cbor",
         "href": "..." }
    ]
   }
```

```
    ]
}
```

In this example, the *system-id* is aligned with the FDO device identity, ensuring a direct binding between the authenticated device and its corresponding DSP instance. The component definition includes references to external security artifacts (e.g., CBOM), while onboarding functionality is modeled as a dedicated component with references to FDO-specific artifacts such as the Ownership Voucher. The inclusion of a MUD descriptor further enables the specification of network-level policies, and should include rules to allow communication with the FDO RV service during the discovery phase.

4.3 Architecture

The proposed architecture is guided by a set of design principles. First, a clear *separation of concerns* is established between identity provisioning and secure communication (handled by FDO) and security policy representation (handled by the DSP). This separation allows both layers to evolve independently while remaining tightly coupled during onboarding. Second, the architecture follows the DSP *reference-based design*, enabling external descriptor retrieval without embedding large artifacts during onboarding. This improves scalability, avoids data duplication, and enables independent lifecycle management of security artifacts. Finally, the system follows a *secure-by-design onboarding paradigm*, in which policy enforcement is integrated into the onboarding workflow.

It is assumed that the FDO protocol provides a secure channel for device authentication and ownership transfer, and that DSP instances are stored in a trusted platform (DSPP) and protected through digital signatures, allowing their integrity and authenticity to be verified prior to use. The DSP platform is expected to be reachable during onboarding, as it is required to retrieve the security descriptors associated with the device. Finally, enforcement points within the operator domain are assumed to be capable of translating and applying the policies derived from DSP descriptors. The architecture is designed to fail securely. In case these assumptions are not met, such as DSP unavailability or verification failures, the system prevents the device from reaching a fully operational state, ensuring that no device is deployed without validated security descriptors.

The architecture defines explicit trust boundaries across its components to minimize the exposure of sensitive operations. The *operator domain* is considered the primary trusted environment, as it performs critical functions such as DSP retrieval, signature verification, descriptor processing, and policy enforcement. The *DSP platform* is trusted to provide authentic and integrity-protected DSP instances, although access to its content is controlled through authorization mechanisms. In contrast, the *rendezvous service* is treated as a minimally trusted component, as it only supports device discovery and redirection and does not participate in the secure communication channel or access sensitive data. The *device* is assumed to be trustworthy with respect to its hardware root of trust and stored credentials, but is not trusted to enforce global security policies beyond its local capabilities. The *manufacturer domain* is trusted to generate correct initial DSP instances and onboarding credentials, but is not involved in operational policy enforcement. This explicit separation confines sensitive operations to trusted domains and limits the role of external entities, thereby reducing the attack surface and strengthening the overall security posture.

As illustrated in Fig. 3, the architecture is structured into three main domains: the *Manufacturer domain*, the *Intermediary platform*, and the *Operator domain*, together with the *Device*, which acts as an edge entity interacting across these domains.

Manufacturer domain. The manufacturer domain hosts the *manufacturer repository* containing external security artifacts referenced by the $genDSP$, such as BOMs, MUD profiles, or vulnerability reports. These resources are not embedded in the DSP but are instead accessed through references, preserving their original structure and enabling independent lifecycle management.

The Intermediary platform acts as a trust broker between domains and hosts the DSP Platform (DSPP), which manages the storage, versioning, and controlled access to DSP instances.

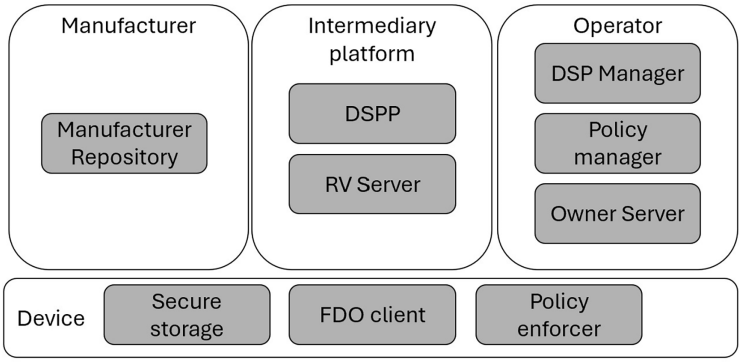

Figure 3: DSP-based onboarding architecture.

Intermediary platform. The intermediary platform hosts the DSP platform (*DSPP*), which manages the storage, versioning, and controlled access to $genDSP$ and $IDSP_1$. It also includes the FDO Rendezvous Server (*RV Server*), which supports device discovery by mapping device identities to operator endpoints during the TO1 phase. This platform does not participate in secure provisioning or policy enforcement, but facilitates coordination across domains.

Operator domain. The Operator domain represents the target deployment environment and concentrates the core orchestration logic of the system. It orchestrates the transition from device authentication to policy enforcement, ensuring that security controls are applied immediately after onboarding. It includes the following key components:

- The *Owner Server*, which executes the FDO TO2 protocol, authenticates the device, and establishes the secure communication channel.
- The *DSP Manager*, responsible for retrieving $IDSP_1$ from the DSPP using the reference received via ServiceInfo, verifying its integrity, and extracting relevant descriptors.
- The *Policy Manager*, which processes the extracted descriptors and translates them into enforceable configurations.

Device. The Device acts as an active participant in the onboarding process, executing the *FDO client* and exposing its onboarding capability as part of the DSP component model. It stores onboarding credentials and the DSP reference in hardware-backed *secure storage* (e.g., TPM or secure element), ensuring protection against extraction or tampering. During onboarding, the device autonomously performs discovery and ownership transfer, and provides the DSP reference via the ServiceInfo exchange. Policy enforcement may also be applied locally through a device-side *policy enforcer* as part of the overall system workflow.

4.4 Secure Onboarding Workflow

From a system perspective, the proposed approach follows a structured onboarding and enforcement pipeline spanning multiple domains. As illustrated in Fig. 4, the workflow captures the end-to-end data flow

across the Manufacturer, Intermediary, Device, and Operator domains, while Algorithm 1 formalizes the corresponding logic. The process begins with device provisioning and DSP association at manufacturing time, continues with device discovery and secure channel establishment through the FDO protocols (TO0–TO2), and culminates in the retrieval, verification, and processing of the DSP within the operator domain. Finally, the system supports continuous adaptation by retrieving updates of the DSP throughout the device lifecycle.

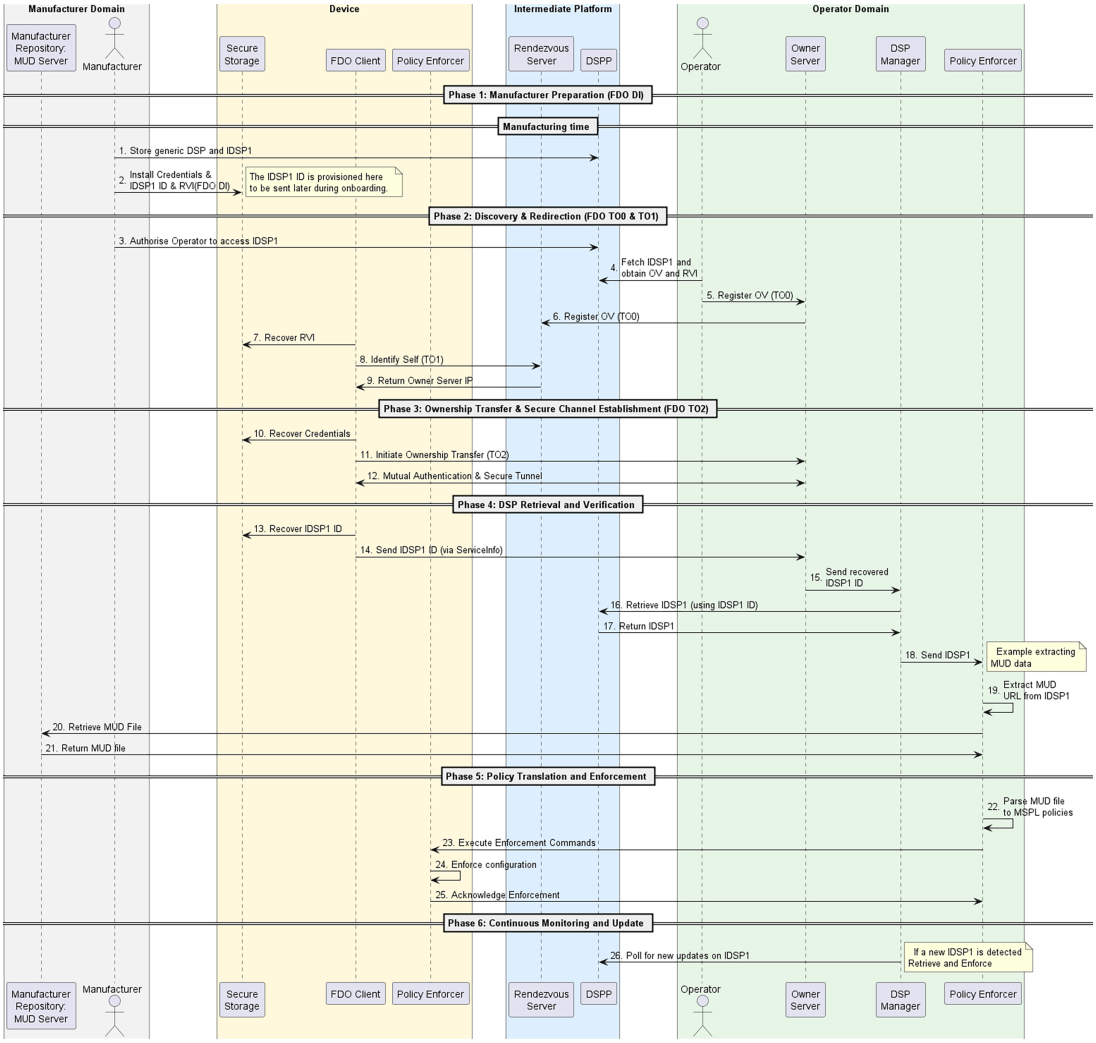

Figure 4: DSP-based data flow.

Algorithm 1: Model-driven secure onboarding with DSP integration

Require: Device d with unique identifier S_{id} and FDO credentials C_{bib5}, and an associated generic DSP model $genDSP$

1: **Phase 1: Device Initialization and DSP Association**
2: Generate $IDSP_1 \leftarrow instantiate(genDSP, S_{id})$
3: $Ref_{IDSP_1} \leftarrow Register(IDSP_1)$
4: Provision device with C_{bib5} and Ref_{IDSP_1}

(Continued)

Algorithm 1 (continued)

Make $IDSP_1$ accessible to the operator domain
Phase 2: Discovery (TO0-TO1)
// TO0: Owner-side registration
$OV \leftarrow ExtractVoucher(IDSP_1)$
$RV \leftarrow ExtractRVlocation(IDSP_1)$
TO0_Register(RV, OV, OwnerEndpoint)
// TO1: Device-side discovery
$OwnerAddr \leftarrow TO1\_Discover(d, RV)$
if $OwnerAddr = \varnothing$ **then**
    Abort (Discovery failed)
end if
Phase 3: Ownership Transfer (TO2)
$Session \leftarrow$ EstablishSecureChannel(d, $OwnerAddr$, C_{bib5})
if $Session$ not established **then**
    Abort
end if
Phase 4: DSP Retrieval and Verification
// Device-side: DSP reference delivery
Send Ref_{IDSP_1} via ServiceInfo over $Session$
// Owner-side: DSP retrieval and verification
$DSP_d \leftarrow Fetch(Ref_{IDSP_1})$
if VerifySignature(DSP_d) = False **then**
    Abort (Invalid DSP)
end if
$D_d \leftarrow ExtractDescriptors(DSP_d)$, $D_d \subseteq \mathbb{D}_{sec}$
Phase 5: Policy Translation and Enforcement
$Policies \leftarrow Translate(D_d)$
Enforce($Policies$)
Phase 6: Continuous Monitoring and Update
while d is active **do**
    if DSPUpdateAvailable(Ref_{IDSP_1}) **then**
        $DSP_d \leftarrow Fetch(Ref_{IDSP_1})$
        $D_d \leftarrow ExtractDescriptors(DSP_d)$, $D_d \subseteq \mathbb{D}_{sec}$
        $Policies \leftarrow Translate(D_d)$
        UpdateEnforcement($Policies$)
    end if
end while

To facilitate comprehension, we use throughout this section the running example of a Home Assistant device associated with the DSP fragment shown in Listing 4. The example illustrates how the DSP reference is exchanged during onboarding, how descriptors such as the MUD profile are retrieved, and how the resulting policies are enforced within the operator domain.

4.4.1 Phase 1—Device initialization and DSP Association

Before deployment, each device is provisioned with onboarding credentials and bound to its corresponding security passport instance. As shown in Fig. 4, during the Device Initialization (DI) phase, cryptographic credentials and rendezvous information are securely stored within the device, establishing its hardware root of trust.

In parallel, the manufacturer derives an instance-level passport ($IDSP_1$) from the generic model ($genDSP$) by binding it to the unique system identifier S_{id} of the device, as defined in Section 3. The resulting $IDSP_1$ is registered in the DSP platform, which returns a unique reference (Ref_{IDSP_1}). This reference is embedded in the device and is securely stored together with the onboarding credentials, ensuring a one-to-one association between the physical device and its corresponding passport instance since manufacturing.

Upon device acquisition, the operator is granted access to the corresponding $IDSP_1$ through the DSP platform. This enables the operator domain to retrieve the security passport associated with the device, establishing the initial linkage between the manufacturing context and the operational environment.

In the running example, the device is provisioned during manufacturing with its FDO credentials and a device-specific DSP instance ($IDSP_1$) partially shown in Listing 4. The $IDSP_1$ is bound to the device identifier (e.g., `EDBA506B90E1420298485E91160CB16F`) and contains references to the security descriptors associated with that device (e.g., MUD and CBOM).

4.4.2 Phase 2—Discovery and Redirection (TO0–TO1)

When first powered on in the field, the device has no prior knowledge of its target operational environment. This phase ensures that devices can be dynamically bound to an operational domain without prior configuration, supporting late-binding. FDO addresses this through a discovery process mediated by the rendezvous service.

Before device activation, the operator registers its Owner endpoint within the rendezvous infrastructure through the TO0 protocol. For this, he leverages onboarding-related information exposed through the $IDSP_1$, including references to FDO trust artifacts such as the Ownership Voucher. This design allows onboarding metadata to be accessed through a unified DSP interface while preserving compatibility with the FDO protocol. In the running example, we can see on the DSP shown in Listing 4 references to the ownership voucher required for FDO onboarding and external descriptors such as the MUD file that should allow the communication between the Home assistant device and the RV server.

Upon boot, the device initiates the discovery process by contacting the rendezvous service using its onboard credentials and rendezvous information. During the TO1 protocol, the rendezvous service assists in device redirection by returning the address of the appropriate Owner endpoint.

This interaction, depicted in Fig. 4, highlights the separation between discovery and trust-establishment, where the rendezvous service only facilitates redirection without participating in secure provisioning.

4.4.3 Phase 3—Ownership Transfer and Secure Channel Establishment (TO2)

The device connects to the Owner server and executes the TO2 protocol. During this phase, mutual authentication is performed based on device credentials and ownership evidence, resulting in the establishment of a secure and encrypted communication channel between the device and the operator domain. As illustrated in Fig. 4, this phase establishes the trust boundary between the device and the operator domain,

enabling subsequent secure metadata exchange. This secure channel also guarantees confidentiality and integrity for all subsequent exchanges, including the transmission of the DSP reference.

4.4.4 Phase 4—DSP Retrieval and Verification

Once the secure channel is established, the device provides the DSP reference Ref_{IDSP_1} via the ServiceInfo exchange. As illustrated in Fig. 4, this reference is received by the Owner Server and forwarded to the DSP Manager, which retrieves the corresponding passport from the DSP platform.

The retrieved DSP is then validated (Eq. (9)) to ensure its integrity and authenticity. If verification fails, the onboarding process is aborted, preventing the device from reaching an operational state (line 27 of Algorithm 1). After successful verification, the DSP Manager may extract a set of security descriptors from external repositories referenced within the *DSP*, including component inventories, behavioral policies, or vulnerability information. As shown in Fig. 4, this may trigger additional interactions with manufacturer-hosted repositories (e.g., MUD servers) to retrieve authoritative policy definitions associated with the device.

This behavior is enabled by the reference-based design of the DSP, where only a lightweight identifier to $IDSP_1$ is exchanged during onboarding, while the retrieval and processing of potentially large descriptors (e.g., MUD, SBOM) are delegated to operator-side components. This design avoids overloading resource-constrained devices and ensures that descriptor size does not impact the onboarding protocol.

In the Home Assistant running example, the DSP shown in Listing 4 references external descriptors such as the CBOM and MUD profile associated with the device, which are retrieved and processed by the DSP Manager during onboarding. For instance, the retrieved MUD profile may contain rules such as the one shown in Listing 5, allowing communication only with devices from the same manufacturer:

Listing 5: Example MUD Access Control List (ACL) retrieved during onboarding

```
{
  "name": "frace2553",
  "matches": {
    "ietf-mud:mud": {
     "same-manufacturer": [ null ]
    }
  },
  "actions": {
    "forwarding": "accept"
  }
}
```

4.4.5 Phase 5—Policy Translation and Enforcement

The extracted descriptors are processed within the operator domain by the Policy Manager, as depicted in Fig. 4. This component acts as the central decision point, aggregating security inputs and translating high-level security intents into enforceable configurations.

In particular, policy translation follows a multi-stage process in which abstract descriptors (e.g., MUD rules or security constraints) are first normalized into an intermediate representation, Medium-level Security Policy Language (MSPL), and then converted into infrastructure-specific or device-specific configurations.

These may include network-level rules (e.g., ACLs, SDN flow rules) or device-level actions (e.g., configuration updates).

In the running example, the retrieved MUD rule shown in Listing 5 is normalized into an intermediate MSPL representation before being converted into an infrastructure-specific enforcement rule, as illustrated in Listing 6.

Listing 6: Example MUD ACL translated to MSPL

```
</mud-abstraction-local_networks-toace44539-60926bd1>
<mud-abstraction-same_manufacturer-toace2553-e2e3c259 type=
< "dict" >
  <external-data type= "dict" >
  <priority type= "int" >1</priority>
 </external-data>
 <configuration-action type= "dict" >
  <filtering-action type= "dict" >
   <filtering-action-type type= "str" >allow</filtering-action-
    type>
 </filtering-action>
</configuration-action>
<configuration-condition type= "dict" >
 <filtering-configuration-condition type= "dict" >
  <packet-filter-condition type= "dict" >
    <direction type= "str" >INBOUND</direction>
  </packet-filter-condition>
 </filtering-configuration-condition>
</configuration-condition>
<name type= "str" >mud-abstraction-same_manufacturer-
  toace2553-e2e3c259</name>
</mud-abstraction-same_manufacturer-toace2553-e2e3c259>
```

The resulting policies are enforced through dedicated enforcement components, which may operate at the infrastructure level (e.g., gateways, switches) or at the device level via a local policy enforcer. This distributed enforcement model ensures that the device transitions directly into a controlled, least-privilege operational state immediately after onboarding. In the running example, the resulting MSPL policy is then mapped to a webhook invocation toward the Home Assistant platform, triggering the corresponding automation or enforcement action within the operator environment.

4.4.6 Phase 6—Continuous Monitoring and Update

After onboarding, the DSP Manager periodically interacts with the DSP platform to detect updates to the associated $IDSP_1$, as shown in Fig. 4. When updates are identified, the corresponding descriptors are retrieved, reprocessed, and the resulting policies are updated accordingly.

This continuous interaction enables the system to maintain an up-to-date security posture throughout the device lifecycle. In particular, changes in the DSP may reflect updated configurations, newly available security information, or mitigation actions for emerging threats. In such cases, previously established policies can be dynamically revised or revoked.

Furthermore, if a device is reassigned or transferred to a different operational domain, the FDO protocol inherently supports re-onboarding through a new ownership transfer process, allowing the device to be securely provisioned under a new trust context.

Overall, this phase extends onboarding into a continuous lifecycle management process, ensuring alignment between device behavior and evolving security requirements.

5 Security Analysis

This section analyzes the security properties of the proposed FDO–DSP onboarding approach, focusing on the threat model and the resistance to common attacks across IoT layers. The analysis is grounded on the architecture and workflow defined in Sections 4.3 and 4.4.

This analysis does not include formal verification of the protocol composition between FDO and DSP. Instead, it builds on the established security properties of FDO and complements them with descriptor validation and policy enforcement mechanisms.

5.1 Threat Model and Assumptions

The threat model considers a distributed IoT deployment in which devices, networks, and intermediary services may be exposed to adversarial actions. The attacker is assumed to have network-level capabilities, including eavesdropping, message injection, and replay, and may attempt to impersonate legitimate entities or tamper with exchanged data during the onboarding process. The security analysis is based on the following assumptions:

- devices are equipped with a hardware root of trust and securely store onboarding credentials;
- the FDO protocol correctly provides mutual authentication and secure channel establishment;
- DSP instances are integrity-protected through digital signatures and can be verified upon retrieval; however, the model assumes that signing authorities and descriptor distribution services are not compromised;
- the operator domain correctly enforces the derived security policies.

5.2 Resistance to Common Attacks

The proposed approach mitigates relevant security threats across different layers of the IoT system by combining FDO-based secure onboarding with DSP-driven verification and policy enforcement.

Device layer. At the device level, the main threat is device spoofing or identity impersonation. This is mitigated through FDO mutual authentication during the TO2 phase, where only devices possessing valid credentials and ownership evidence can establish a secure session with the Owner. The use of hardware-backed secure storage further protects device credentials against extraction and misuse. This also mitigates credential misuse scenarios, as sensitive material is protected against extraction and unauthorized replication.

Network layer. At the communication level, threats include man-in-the-middle (MITM), replay, and eavesdropping attacks. The TO2 protocol establishes an authenticated and encrypted channel, ensuring the confidentiality and integrity of all exchanged data, including the DSP reference transmitted via ServiceInfo. Replay attacks are prevented through freshness guarantees, such as nonces and session-based

exchanges defined in the FDO protocol. Additionally, the secure channel prevents message injection and manipulation attacks.

Application and service layer. At the management level, risks include tampering with security descriptors and unauthorized modification of device configuration. DSP instances are digitally signed and verified upon retrieval (Phase 4). Any modification of the DSP or its metadata results in verification failure and aborts the onboarding process. This prevents attackers from exploiting the onboarding phase to introduce malicious or altered security metadata. The model also mitigates descriptor poisoning scenarios involving manipulated MUD profiles, forged vulnerability descriptors, or malicious references distributed through external repositories. Since both DSP instances and referenced artifacts are validated before processing, unauthorized or altered descriptors cannot be enforced without passing integrity verification. In addition, the use of authenticated retrieval channels and trust anchoring in manufacturer- or operator-controlled repositories reduces the risk of unauthorized descriptor injection during distribution or transport. Additionally, security policies derived from the DSP are enforced immediately after onboarding, preventing devices from operating with excessive privileges.

Intermediary services. The rendezvous service is treated as a minimally trusted component. It only performs discovery and redirection functions and does not participate in authentication or access sensitive data. As a result, even if compromised, it cannot impersonate the Owner or gain control over the onboarding process.

Cross-layer. Security policies derived from the DSP are enforced immediately after onboarding, eliminating the common exposure window in which devices operate without restrictions. This ensures that devices transition directly into a controlled, least-privilege state. Furthermore, the reference-based design of the DSP enables continuous policy updates during operation, allowing the system to adapt to newly discovered vulnerabilities and evolving security requirements throughout the device lifecycle.

Table 4 shows a summary of the threats and mitigation mechanisms identified in each layer.

Table 4: Overview of threats and mitigation mechanisms.

Layer	Threat	Mitigation
Device	Device spoofing/impersonation	FDO mutual authentication (TO2), hardware-backed credential storage
Network	Man-in-the-Middle (MITM) Replay attacks Eavesdropping Message injection/manipulation	Authenticated and encrypted channel (TO2) Nonces and session-based freshness guarantees in FDO Encrypted communication channel (TO2) Integrity protection via authenticated secure channel (TO2)
Application/Service	Descriptor tampering Unauthorized access/over-privilege	DSP signature verification before processing Immediate policy enforcement after onboarding
Intermediary services	Compromised rendezvous	Minimal trust design, no role in authentication or provisioning
Cross-layer	Post-onboarding temporal gap Vulnerability exploitation/evolving threats	Immediate policy enforcement at onboarding Dynamic updates via DSP (e.g., MUD, Threat MUD)

5.3 Limitations of the Security Model

While the proposed approach provides strong guarantees for secure onboarding and policy enforcement, several limitations must be acknowledged.

The security model relies on the assumption of a trusted computing base, including the correctness of the FDO protocol implementation, the integrity of the DSP platform, and the proper enforcement of policies within the operator domain. Compromise of any of these components may impact the overall security guarantees. In particular, while DSP integrity verification prevents unauthorized modification of descriptors during distribution, the model assumes that descriptor signing authorities and backend management services behave correctly. A compromised DSP management infrastructure could distribute outdated, incorrect, or maliciously signed descriptors, potentially affecting the resulting policy generation process. Similarly, compromise of manufacturer-hosted repositories containing referenced descriptors (e.g., MUD, SBOM, or VEX artifacts) could affect the quality or trustworthiness of the retrieved security information. Although integrity verification mechanisms prevent unauthorized descriptor modification, malicious but correctly signed descriptors remain a potential supply-chain risk. Mitigating such scenarios may require additional mechanisms such as descriptor revocation, certificate revocation lists (CRLs), short-lived signing credentials, trusted timestamping, repository replication, or transparency and auditing mechanisms. These mechanisms would allow outdated, compromised, or deprecated descriptors to be invalidated before subsequent onboarding or policy-update operations. Also, the approach assumes the presence of hardware-backed secure storage on the device. Devices lacking such capabilities may be more vulnerable to credential extraction or tampering. The model does not explicitly address physical attacks against devices beyond the protection offered by secure elements. Advanced physical tampering or side-channel attacks remain out of scope. Finally, while the system enables rapid mitigation through dynamic policy updates, its effectiveness depends on the timely availability and distribution of updated descriptors (e.g., MUD or Threat MUD). Delays in update propagation may temporarily expose devices to known vulnerabilities.

6 Experimental Validation

This section presents the experimental validation of the proposed model-driven secure onboarding approach. The evaluation considers a realistic Smart Home scenario to assess both the feasibility of integrating DSP within the FDO onboarding process and the impact of this integration on performance and policy enforcement. The validation focuses on two main aspects: (i) onboarding performance and associated overhead, and (ii) responsiveness to security updates during operation.

6.1 Scenario Description

The experimental validation is performed in a heterogeneous Smart Home environment designed to reflect realistic IoT deployment conditions. The setup includes multiple platforms and device configurations to evaluate the proposed model-driven onboarding approach under diverse operational settings. The environment comprises typical IoT components such as sensors, actuators, and a centralized control platform.

Home Assistant [27] is used as the main orchestration platform, acting as the central point for device management and policy enforcement. It exposes a Representational State Transfer (REST)-based interface through which security policies derived from DSP descriptors are enforced, enabling integration without requiring modifications to device firmware. The FDO device functionality is integrated into Home Assistant through its add-on mechanism [28]. Home Assistant applications (add-ons) are containerized services managed by the underlying operating system. In this work, the FDO client is packaged as a custom add-on available in [29], allowing it to run natively within the platform and interact with other system components. This approach enables seamless integration of the onboarding process within an existing IoT ecosystem without requiring external deployment or manual configuration.

The implemented prototype follows the architecture defined in Section 4.3. FDO components (Rendezvous and Owner servers) are deployed as modular services, while the DSP platform is implemented as a

microservices-based system exposing REST-based APIs for DSP storage, retrieval, and lifecycle management. The DSPP acts as a central trust broker, supporting both generic and instance-level DSP management, and enabling controlled access to DSP documents through authentication and access control mechanisms. Its architecture separates API mediation, business logic, and data access into independently deployable services, facilitating scalability and modularity.

The prototype combines open-source components and custom microservices. The FDO protocol relies on the reference implementation provided by the FIDO Alliance [30], while the DSP Manager and Policy Manager are implemented as lightweight Python-based services. All components are deployed as containerized services using Docker, enabling modularity and isolation. The evaluation environment simulates the different architectural domains through isolated virtual networks, allowing reproducible execution of the FDO phases (DI, TO1, TO2) and DSP-based interactions. During manufacturing, devices are provisioned with FDO credentials and associated with a device-specific passport ($IDSP_1$), embedding a reference (Ref_{IDSP_1}) for retrieval during onboarding. The DSPs used are available in [31]. To assess the impact of descriptor retrieval on the onboarding process, two deployment configurations for the MUD server are considered. In the first configuration, MUD profiles are hosted on a public cloud storage service, representing a realistic external deployment scenario. In the second configuration, MUD profiles are served from a local Python-based server. This allows isolating the effect of external infrastructure latency and evaluating the intrinsic overhead of the proposed approach.

The evaluation is conducted across multiple deployments, all running Home Assistant as the control platform:

- a Smart Home pilot deployment at CERTH [32] using Home Assistant Green (Raspberry Pi Compute Module 4)
- a local deployment on Raspberry Pi 4B
- a virtualized deployment using Proxmox
- a generic Linux-based device (AMD Ryzen, 64 GB)

This heterogeneous setup allows assessing the behavior of the system across both resource-constrained edge platforms and more powerful virtualized environments. Finally, the evaluation includes a comparison with a manual onboarding and configuration process, providing insight into the practical benefits of the proposed approach in terms of automation and reduction of operational effort.

6.2 Phase-Level Analysis

For the evaluation, we consider the total onboarding time and a phase-level analysis of the onboarding workflow. In particular, we measure the latency of the main FDO phases (DI, TO1, TO2), as well as the additional steps introduced by the proposed approach, including DSP retrieval, descriptor processing, policy generation, and enforcement. We can decompose the onboarding process as follows:

$$T_{onboarding} = T_{DI} + T_{TO1} + T_{TO2} + T_{DSP} + T_{MUD} + T_{policygen} + T_{policyenf}$$

where T_{DI}, T_{TO1}, T_{TO2} correspond to the FDO phases, T_{DSP} captures DSP retrieval and validation, T_{MUD} includes descriptor resolution and retrieval, and $T_{policygen}, T_{policyenf}$ represent policy generation and enforcement, respectively. In addition, we measure the update detection latency T_{update} during operation, defined as the time required to obtain and apply updates to the DSP once they become available. This decomposition allows isolating the contribution of each phase and analyzing the impact of DSP integration.

Fig. 5 presents a detailed breakdown of the onboarding process across all evaluated deployments, distinguishing between configurations where descriptors are retrieved from a cloud-based repository and from a local server.

The results show that most of the onboarding latency is attributable to the FDO phases (DI, TO1, and TO2). These phases remain consistent across deployments, although slightly lower execution times are observed in more powerful environments (e.g., Proxmox and Linux-based platforms) compared to resource-constrained devices such as Raspberry Pi or Home Assistant Green. This behavior is expected, as these phases involve cryptographic operations and protocol exchanges that benefit from increased computational resources.

The additional phases introduced by the proposed approach remain lightweight and stable. DSP retrieval is highly consistent across all deployments, with an average latency of approximately 0.67 s, confirming that the integration of DSP does not introduce significant overhead. Similarly, policy generation and enforcement phases are negligible, as they involve lightweight processing and simple configuration actions (e.g., triggering Home Assistant automations via webhook).

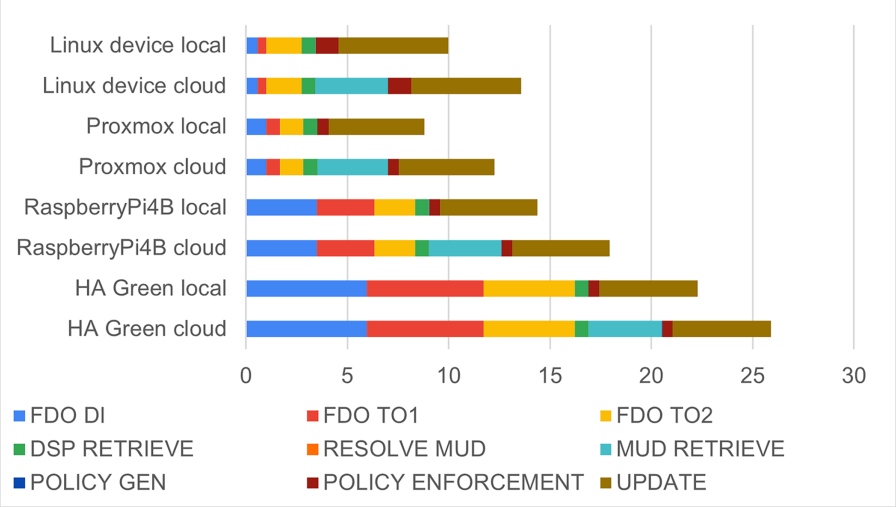

Figure 5: Phase level analysis of FDO-DSP in seconds.

The main source of variability is observed in the descriptor-related phases, particularly MUD resolution and retrieval. When descriptors are hosted in a cloud environment, these phases introduce a significant delay (approximately 3–4 s), which substantially impacts the overall onboarding time. In contrast, when a REST MUD server is used, this latency is reduced to the order of milliseconds, resulting in a much more compact onboarding profile. This clearly indicates that the dominant overhead is not intrinsic to the proposed FDO–DSP integration, but rather depends on external infrastructure and network conditions. Across all deployments, the relative behavior of the phases remains consistent, demonstrating that the approach scales across heterogeneous environments while preserving predictable performance characteristics.

Finally, the update phase (T_{update}) reflects the time required to retrieve, process, and re-enforce updated DSP descriptors once they become available. The measured latency ranges between approximately 4.71 and 5.41 s across deployments. Although the system relies on a polling mechanism to detect updates, the polling interval is a configurable parameter and is therefore excluded from the reported measurements. As such, T_{update} captures only the effective processing time, confirming that dynamic updates can be applied efficiently without introducing significant overhead during operation.

6.3 Comparative Evaluation

To provide a comprehensive evaluation, we combine a quantitative comparison against baseline and manual approaches with a qualitative assessment of existing onboarding solutions.

6.3.1 Experimental Comparison

A direct experimental comparison with alternative onboarding approaches is challenging due to differences in implementation models, infrastructure dependencies, and the lack of publicly available and comparable implementations. Therefore, the evaluation focuses on a controlled comparison between baseline FDO, the proposed FDO-DSP integration, and a manual onboarding approach, allowing us to isolate the overhead introduced by structured metadata processing and policy enforcement, as well as to assess the benefits of automation compared to manual onboarding approaches.

Fig. 6 summarizes the total onboarding time across all evaluated deployments. The results show that the overhead introduced by FDO-DSP remains consistent across platforms in absolute terms, ranging between 4.72 and 5.41 s when descriptors are retrieved from a cloud-based repository. This indicates that the additional latency is largely independent of the underlying hardware and primarily associated with descriptor retrieval and processing.

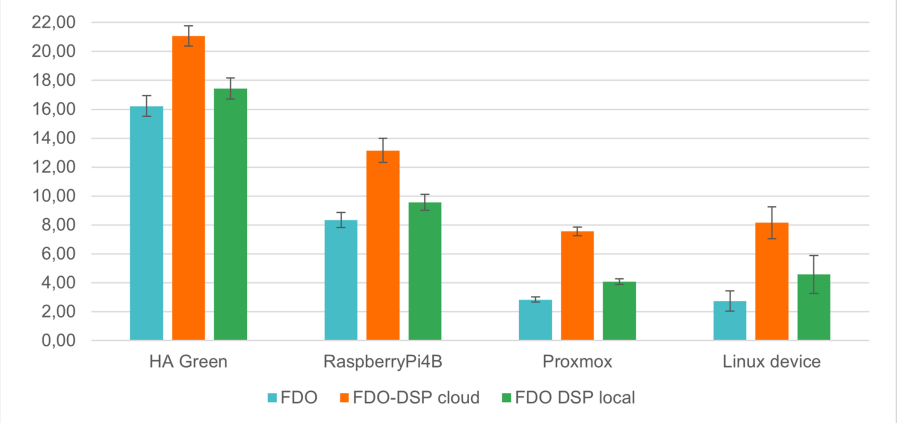

Figure 6: FDO vs. FDO-DSP time comparison (seconds).

In relative terms, the overhead varies from approximately 29.90% in resource-constrained environments (e.g., Home Assistant Green) up to 197.35% in high-performance platforms. However, this variation is mainly due to the significantly lower baseline onboarding time in more powerful systems, rather than an increase in the intrinsic cost of the proposed approach.

It is worth noting that when descriptors are retrieved from a local REST-based server, the overhead is reduced to less than 1 s (approximately 0.67–0.69 s), corresponding to less than 25% even in optimized environments. This indicates that the intrinsic overhead of DSP integration is minimal, and that the dominant cost observed in cloud-based deployments is driven by external infrastructure rather than by the onboarding mechanism itself.

From a system perspective, this overhead is incurred only once during the onboarding phase. In return, the proposed approach enables immediate enforcement of security policies at bootstrap time, eliminating the need for post-deployment configuration. This significantly reduces the exposure window in which devices operate without restrictions, a well-known security risk in conventional onboarding approaches.

Therefore, the overhead should be interpreted as a bounded and largely network-dependent cost that enables secure-by-design deployment. The trade-off is favorable: a small increase in onboarding time results in stronger security guarantees, automated policy enforcement, and elimination of manual configuration steps. These findings are consistent with prior work in IoT onboarding and provisioning, where the integration of additional security mechanisms introduces moderate but acceptable latency in exchange for improved security and manageability [11,14].

Finally, manual onboarding provides a useful reference point to contextualize the benefits of automation. In the evaluated scenario, manual onboarding and policy configuration require approximately 2330 s, which is substantially higher than the automated approaches. Manual results are not included in Fig. 6, as their magnitude would distort the scale and limit the readability of the comparison. Nevertheless, this gap clearly illustrates the substantial reduction in deployment time and operational effort achieved through the proposed approach, while simultaneously improving security guarantees.

6.3.2 Comparison with Existing Onboarding Approaches

Table 5 complements the experimental evaluation with a qualitative comparison of representative IoT onboarding approaches, highlighting trade-offs between automation, interoperability, metadata integration, and policy enforcement.

Table 5: Comparison of onboarding approaches with FDO-DSP.

Feature	Manual	Wi-Fi DPP	ASOP	BRSKI	AWS IoT	FDO	FDO-DSP
Automation /Scalability	None	Medium (user-assisted)	Medium (user-assisted)	High (zero-touch)	High (automated)	High (zero-touch)	High (zero-touch)
Identity and Credential Management	Static/none	Public key + QR	PSK	PKI (X.509, IDevID)	Cloud managed PKI (X.509)	PKI-like + OV	PKI-like + OV
Late-binding	No	No	No	Limited	No	Yes	Yes
Metadata exchange	No	Network parameters	Partial	Limited	Yes	Yes	Yes (structured and updatable)
Policy enforcement	Manual (post)	No	No	No	Automated (post)	No	Automated in onboarding
Ownership tracking	No	No	No	MASA based	Partial (cloud registry)	OV chain	OV chain
Platform dependency	Tooling dependent	Open (Wi-Fi Alliance)	Low (open+Google)	Open (IETF)	High (cloud)	Open (multivendor)	Open (multivendor)

In terms of automation and scalability, manual onboarding approaches require significant human intervention, whereas solutions such as Wi-Fi DPP and ASOP reduce operational effort but still depend on user involvement. In contrast, BRSKI, AWS IoT, and FDO support automated or zero-touch provisioning, making them suitable for large-scale deployments [33]. FDO-DSP preserves these scalability properties by relying on lightweight DSP references instead of embedding security descriptors directly in the onboarding exchange. As a result, resource-constrained devices are not required to process large security artifacts during onboarding.

Identity and credential management also differ across approaches. Manual provisioning relies on static credentials, whereas DPP and ASOP use pre-shared or public-key bootstrapping. BRSKI and AWS IoT adopt PKI-based trust models with stronger identity guarantees but increased infrastructure requirements [12].

FDO enhances flexibility through ownership vouchers and late binding, allowing devices to be associated with an operational domain at deployment time rather than during manufacturing [33]. FDO-DSP inherits this capability, making it suitable for dynamic multi-stakeholder supply chains.

The differences become more evident in metadata integration and policy enforcement. Manual onboarding does not support metadata exchange, while DPP, ASOP, and BRSKI provide only limited support focused on onboarding or identity-related information. AWS IoT enables richer metadata management through cloud-specific mechanisms [34]. Although FDO supports secure onboarding data transfer, it does not define a structured representation for security information or associated enforcement mechanisms. FDO-DSP extends this capability by introducing structured metadata, enabling consistent representation and interpretation of security-relevant information across systems. This distinction has direct implications for policy enforcement. As shown in Table 5, most approaches provide limited or no support for enforcing policies during onboarding. Even in BRSKI, enforcement is indirect and typically tied to PKI-based mechanisms, while AWS IoT provides strong policy enforcement through tightly integrated cloud services. FDO, despite supporting data transfer, does not define how policies should be derived or enforced. In contrast, FDO-DSP enables strong policy enforcement by linking structured onboarding information with enforcement mechanisms, allowing policies to be applied as part of the onboarding process rather than as a separate step.

Ownership and lifecycle management present a similar pattern. Manual, DPP, and ASOP approaches do not provide ownership tracking, while BRSKI and AWS IoT offer only partial support tied to their trust infrastructures. FDO introduces ownership transfer through ownership vouchers, but without explicit lifecycle-aware security context management. FDO-DSP extends this model by enabling explicit tracking of ownership and associated security information across the device lifecycle.

From an infrastructure perspective, manual approaches require minimal support, whereas DPP, ASOP, BRSKI, and AWS IoT depend on configurators, PKI infrastructures, or cloud services. FDO adopts a modular architecture based on rendezvous and ownership services, avoiding tight coupling to specific platforms. FDO-DSP preserves this deployment model while adding descriptor processing capabilities without significant architectural changes.

Finally, platform dependency remains low for most approaches except AWS IoT, which is tightly coupled to its cloud ecosystem. Both FDO and FDO-DSP maintain low dependency, preserving interoperability across heterogeneous environments.

Overall, the table shows that existing onboarding solutions typically optimize specific dimensions, leading to trade-offs between scalability, flexibility, interoperability, and policy enforcement. FDO provides a balanced approach through automation, scalability, and late-binding support, but does not natively address structured security metadata integration or onboarding-time policy enforcement. FDO-DSP extends these capabilities by incorporating structured metadata, automated policy enforcement, and lifecycle-aware ownership tracking while maintaining the scalability and interoperability properties of FDO.

6.3.3 Comparison with Existing Security Descriptors

Beyond onboarding mechanisms, it is also important to compare how existing security descriptor models represent and integrate security-related information. Table 6 summarizes representative approaches according to their expressive capabilities and integration potential. Current models are generally domain-specific: SBOM focuses on software components, VEX/VDR on vulnerability disclosure, MUD on network behavior, Threat MUD on mitigation actions, and OSCAL on compliance and assessment. In

contrast, DSP adopts a unified device-centric model that combines these perspectives within a single security representation.

For *vulnerability description*, VEX/VDR and OSCAL provide explicit support, while SBOM formats may include vulnerability information through extensions such as CycloneDX [35]. Threat MUD offers only partial support through references to vulnerabilities and severity metrics (e.g., CVE and CVSS identifiers) without modeling vulnerabilities as structured entities [22]. DSP integrates vulnerability information through structured references to external artifacts such as VEX/VDR documents, enabling consistent integration within a broader security context.

Regarding *behavior specification*, MUD and Threat MUD natively define network communication policies, whereas OSCAL provides only descriptive and non-enforceable representations. DSP supports policy-based behavioral specifications linked directly to enforcement mechanisms, while SBOM and VEX/VDR do not model device behavior.

Table 6: Comparison of security descriptor models.

Feature	SBOM	VEX/VDR	MUD	Threat MUD	OSCAL	DSP
Primary focus	Components	Vulnerabilities	Network behavior	Mitigation	Compliance	Unified device security
Vulnerability description	Partial	Explicit	None	Partial	Explicit	Explicit (via VEX/VDR)
Behavior specification	None	None	Native	Native	Descriptive	Policy-based
Mitigation definition	None	Partial	Implicit	Explicit	Explicit	Explicit
External referencing	Limited	Supported	Supported	Supported	Supported	Supported
Multi-descriptor integration	Limited	None	Limited	None	Limited	Native
Policy enforcement support	None	None	Network-level	Network-level	None	Fully supported

For *mitigation definition*, Threat MUD explicitly represents mitigation actions enforceable at the network level, and OSCAL supports remediation planning through structured compliance constructs [25]. VEX/VDR provide partial support, since some formats include remediation guidance (e.g., CSAF-based VEX [36]) while others focus only on vulnerability status (e.g., OpenVEX [37]). MUD provides only indirect mitigation support through traffic restrictions. DSP extends these capabilities by explicitly representing mitigation strategies and linking them to vulnerabilities and risks through structured descriptors.

Most approaches support some form of *external referencing*. VEX/VDR, MUD, Threat MUD, OSCAL, and DSP can reference external resources such as documentation or vulnerability databases, whereas SBOM formats typically embed information locally, although some specifications, such as CycloneDX, support external references [35].

A major differentiating factor is *multi-descriptor integration*. Existing approaches typically focus on a specific type of security information and, when external descriptors are supported, rely on loosely coupled references rather than a unified representation. DSP instead provides a consistent model capable

of integrating heterogeneous security descriptors while preserving their original semantics. Similarly, *policy enforcement support* remains limited in most existing approaches. MUD and Threat MUD support network-level enforcement [38], whereas SBOM, VEX/VDR, and OSCAL remain primarily descriptive. DSP extends this capability by enabling policy enforcement directly driven by integrated security descriptors.

These capabilities align with emerging regulatory frameworks such as the EU CRA [6], which emphasizes secure-by-default operation, vulnerability management, and continuous security updates. A detailed mapping between DSP capabilities and CRA requirements is presented in [10].

7 Discussion

The results presented in Section 6 demonstrate the feasibility of integrating DSP-driven security descriptors within a secure onboarding workflow based on FDO. Beyond functional validation, the proposed approach reframes onboarding as a control point for enforcing security-by-default, rather than solely as a mechanism for identity establishment.

A key observation is that the proposed approach reduces the temporal gap between device provisioning and policy enforcement identified in Section 2. In conventional workflows, policies are typically applied after connectivity is established, leaving devices temporarily over-privileged. In contrast, the proposed model enables policy enforcement immediately after onboarding, allowing devices to transition directly into a policy-compliant operational state. From a performance perspective, the integration introduces a bounded and predictable latency, ranging between approximately 4.72 and 5.41 s in cloud-based deployments. The phase-level analysis shows that this overhead is dominated by descriptor retrieval, particularly MUD resolution, rather than by the DSP model itself. When descriptors are served locally, the overhead drops below one second (approximately 0.67 s), confirming that the intrinsic cost of DSP integration is low. This indicates that the overhead is primarily driven by infrastructure locality and deployment conditions. Moreover, this cost is incurred only once during onboarding and does not affect runtime operation.

The results also indicate that the proposed approach preserves the scalability properties of FDO. Since onboarding exchanges only transfer lightweight DSP references rather than complete descriptors, device-side processing remains minimal even in constrained environments. Descriptor retrieval, DSP processing, and policy generation are delegated to backend management services, reducing the computational requirements imposed on IoT devices. Nevertheless, large-scale deployments involving hundreds or thousands of simultaneous onboarding operations may increase the load on backend components such as the DSPP or policy management services, particularly under centralized deployments. In such scenarios, scalability is expected to depend primarily on the capacity and distribution of the management infrastructure rather than on the DSP-FDO protocol itself. Mechanisms such as distributed deployments, load balancing, caching, or edge-based processing may therefore be required to maintain predictable onboarding latency.

From an operational perspective, the comparison with manual onboarding highlights a substantial reduction in deployment effort. While the automated approach introduces a moderate latency overhead with respect to FDO, it reduces provisioning time from minutes to seconds and eliminates manual policy configuration. This may be relevant in large-scale deployments, where manual processes introduce scalability limitations and increase the risk of configuration errors. The proposed approach also enables a lifecycle-aware security management model. Since descriptors remain externalized and maintainable independently, updates can be dynamically retrieved, processed, and enforced during operation without requiring device reprovisioning. The evaluation shows that updated descriptors can be processed and re-enforced in approximately 5 s, enabling continuous adaptation of security policies throughout the device lifecycle. From a scalability perspective, the evaluated overhead suggests that concurrent onboarding operations can be efficiently parallelized across multiple backend instances. Assuming the cost of approximately 5 s per device,

a deployment scenario involving 100 simultaneous onboarding requests would still be expected to complete within tens of seconds under near-linear resource scaling. In this context, scalability is expected to be primarily constrained by descriptor retrieval latency and policy translation throughput rather than by the FDO protocol itself. These observations suggest that the proposed architecture can support larger-scale deployments, particularly when complemented with mechanisms such as distributed deployments, load balancing, descriptor caching, or edge-assisted processing.

At the same time, the findings highlight several limitations that may affect the practical adoption of the approach. First, reliance on external descriptor repositories introduces dependencies on network availability and service responsiveness. As observed in the evaluation, descriptor retrieval may become the dominant source of latency, particularly in cloud-based deployments. Moreover, the results show that the intrinsic overhead introduced by DSP integration remains low, while most of the observed latency is associated with external descriptor access and deployment conditions. Second, the effectiveness of the proposed model depends on the quality, completeness, and freshness of the underlying security descriptors. The DSP assumes that artifacts such as SBOMs, MUD profiles, or VEX statements are accurate and consistently maintained. In practice, inconsistencies, missing information, or delayed updates may affect the reliability of automated policy generation and enforcement. While this challenge is not specific to the proposed approach, it becomes more critical when security enforcement is tightly coupled to descriptor content. Finally, the current implementation relies on predefined mappings between abstract policy representations and concrete enforcement mechanisms. Although sufficient for the evaluated environments, this may limit flexibility in heterogeneous or highly dynamic infrastructures where enforcement capabilities differ across platforms and devices. Extending the approach toward more adaptive and context-aware policy translation therefore remains an open challenge.

Overall, the results indicate that integrating structured security models such as the DSP with secure onboarding protocols enables a more consistent and integrated approach to device provisioning and policy enforcement, reducing the gap between these processes while maintaining practical performance.

8 Conclusion and Future Work

This work presented a model-driven approach to secure IoT onboarding by integrating the DSP with the FDO protocol. The proposed solution bridges the gap between onboarding and policy enforcement. It enables the retrieval and activation of security descriptors during the onboarding phase, ensuring that policies are enforced immediately after onboarding. The DSP was formally defined as a lifecycle-aware model inspired by OSCAL, capable of aggregating heterogeneous security descriptors (e.g., SBOM, MUD, VEX/VDR) through a reference-based structure. Its integration with FDO enables a secure binding between device identity and its associated security metadata, supporting both initial provisioning and continuous adaptation during operation. Experimental results demonstrate the feasibility of the approach, showing that DSP integration introduces a moderate and bounded overhead while enabling immediate policy enforcement and contributing to improved security at deployment time.

Future work will focus on extending the evaluation to large-scale and heterogeneous IoT environments, further assessing scalability and performance under diverse conditions. In addition, future research will include adversarial validation scenarios to evaluate the effectiveness of onboarding-time policy enforcement against policy violations and malicious device behavior. The potential use of Large Language Models (LLMs) will be explored to support automated analysis and interpretation of security requirements. In particular, future research will investigate how LLMs can be leveraged to map regulatory or standard-based requirements to DSP representations, enabling automated compliance assessment and the generation of validation tests based on device security descriptors. Finally, future work will explore the integration of

additional security descriptors and contextual information to support more adaptive and context-aware policy enforcement during onboarding.

Acknowledgement: The authors would like to thank Miguel Fernández Llamas for his support in the development of the graphical interface used to facilitate timing measurements and policy enforcement management.

Funding Statement: The work has been produced with the support of a 2024 Leonardo Grant for Scientific Research and Cultural Creation, BBVA Foundation. The foundation takes no responsibility for the opinions, statements and contents of this project, which are entirely the responsibility of its authors. Research also partially supported by the EU through the Horizon Research and Innovation Program DOSS (grant agreement No. 101120270) and COBALT (grant agreement No. 101119602).

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Sara Matheu, Pedro Ruzafa, Antonio Skarmeta and Dionysios Kehagias; methodology, Sara Matheu, Pedro Ruzafa; software, Pedro Ruzafa and Ilias Kalouptsoglou; validation, Pedro Ruzafa and Ilias Kalouptsoglou; formal analysis, Sara Matheu; investigation, Sara Matheu, Pedro Ruzafa; resources, Pedro Ruzafa and Ilias Kalouptsoglou; data curation, Pedro Ruzafa and Ilias Kalouptsoglou; writing—original draft preparation, Sara Matheu, Pedro Ruzafa and Ilias Kalouptsoglou; writing—review and editing, Sara Matheu; visualization, Sara Matheu; supervision, Sara Matheu and Antonio Skarmeta; project administration, Sara Matheu, Antonio Skarmeta and Dionysios Kehagias; funding acquisition, Sara Matheu, Antonio Skarmeta and Dionysios Kehagias. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Data openly available in https://github.com/saranieves92/DSP.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript

AI	Artificial Intelligence
API	Application Programming Interface
ASOP	Agile Secure Onboarding Protocol
AWS	Amazon Web Services
BOM	Bill of Materials
BRSKI	Bootstrapping Remote Secure Key Infrastructure
CBOM	Cryptographic Bill of Materials
CRA	Cyber Resilience Act
CPE	Common Platform Enumeration
CVE	Common Vulnerabilities and Exposures
DI	Device Initialization
DPP	Device Provisioning Protocol
DSP	Device Security Passport
DSPP	Device Security Passport Platform
ECC	Elliptic Curve Cryptography
ECSO	European Cyber Security Organisation
EU	European Union
FDO	FIDO Device Onboard
FEAM	Front-end Access Management
FIDO	Fast IDentity Online
GUID	Globally Unique Identifier

HBOM	Hardware Bill of Materials
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IoT	Internet of Things
MUD	Manufacturer Usage Description
NIST	National Institute of Standards and Technology
NVD	National Vulnerability Database
LLM	Large Language Model
OSCAL	Open Security Controls Assessment Language
OV	Ownership Voucher
PKI	Public Key Infrastructure
RV	Rendezvous
SCAP	Security Content Automation Protocol
SBOM	Software Bill of Materials
TPM	Trusted Platform Module
UUID	Universally Unique Identifier
VDR	Vulnerability Disclosure Report
VEX	Vulnerability Exploitability eXchange
XCCDF	eXtensible Configuration Checklist Description Format

References

1. Gartner. Forecast: IoT market opportunity by technology segment, 2022–2028; 2024 [cited 2026 Mar 31]. Available from: https://www.gartner.com/en/documents/5499495.
2. Espressif Systems. Wi-Fi easy connect (DPP)—ESP32-S3 ESP-IDF programming guide v6.0; 2026 [cited 2026 May 4]. Available from: https://docs.espressif.com/projects/esp-idf/en/stable/esp32s3/api-reference/network/esp_dpp.html.
3. Reaz K, Wunder G. ASOP: a sovereign and secure device onboarding protocol for cloud-based IoT services. arXiv:2403.13020. 2024.
4. Pritikin M, Richardson M, Eckert T, Behringer MH, Watsen K. RFC 8995: bootstrapping remote secure key infrastructure (BRSKI); 2021 [cited 2026 May 4]. Available from: https://datatracker.ietf.org/doc/rfc8995/.
5. Cooper G, Behm B, Chakraborty A, Kommalapati H, Mandyam G, Tschofenig H, et al. FIDO device onboard specification 1.1; 2022 [cited 2026 Mar 27]. Available from: https://fidoalliance.org/specs/FDO/FIDO-Device-Onboard-PS-v1.1-20220419/FIDO-Device-Onboard-PS-v1.1-20220419.html.
6. European Commission. Cyber resilience act; 2025 [cited 2025 Apr 23]. Available from: https://digital-strategy.ec.europa.eu/en/policies/cyber-resilience-act.
7. The United States Department of Commerce, NTIA SBOM Minimum Elements Working Group. Minimum elements for a software bill of materials (SBOM): preliminary report. national telecommunications and information administration (NTIA); 2023 [cited 2025 Apr 23]. Available from: https://www.ntia.doc.gov/files/ntia/publications/sbom_minimum_elements_report.pdf.
8. Lear E, Romascanu D, Droms R. RFC 8520: manufacturer usage description specification; 2019 [cited 2025 Apr 23]. Available from: https://datatracker.ietf.org/doc/rfc8520/.
9. National Telecommunications and Information Administration (NTIA). Vulnerability-exploitability eXchange (VEX): an overview; 2021 [cited 2025 Sep 5]. Available from: https://www.ntia.gov/files/ntia/publications/vex_one-page_summary.pdf.
10. Matheu Garca SN, Skarmeta Gómez A. One passport to govern them all: bringing order to IOT security and compliance. Zenodo; 2025. doi:10.5281/zenodo.18120755.
11. Maksuti S, Bicaku A, Zsilak M, Ivkic I, Peceli B, Singler G, et al. Automated and secure onboarding for system of systems. IEEE Access. 2021;9:111095–113. doi:10.1109/ACCESS.2021.3102280.

12. Hoglund J, Raza S, Furuhed M. Towards automated PKI trust transfer for IoT. In: 2022 IEEE International Conference on Public Key Infrastructure and its Applications (PKIA). Piscataway, NJ, USA: IEEE; 2022. p. 1–8. doi:10.1109/PKIA56009.2022.9952223.
13. Amazon Web Services. Device manufacturing and provisioning with X.509 certificates in AWS IoT core—device manufacturing and provisioning with X.509 certificates in AWS IoT core; 2026 [cited 2026 May 2]. Available from: https://docs.aws.amazon.com/whitepapers/latest/device-manufacturing-provisioning/device-manufacturing-provisioning.html.
14. FIDO Alliance. FIDO alliance input to NIST trusted internet of things (IoT) device network-layer onboarding and lifecycle management (Draft); 2020 [cited 2026 May 2]. Available from: https://fidoalliance.org/wp-content/uploads/2020/10/FIDO-ALLIANCE-Response-to-NIST-IOT-Onboarding-Draft-October-2020.pdf.
15. National Institute of Standards and Technology (NIST). NIST special publication 1800-36: trusted internet of things (IoT) device network-layer onboarding and lifecycle management: enhancing internet protocol-based IoT device and network security; 2025 [cited 2026 May 2]. Available from: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1800-36.pdf.
16. Geismann J, Bodden E. A systematic literature review of model-driven security engineering for cyber-physical systems. J Syst Softw. 2020;169(1):110697. doi:10.1016/j.jss.2020.110697.
17. Mashkoor A, Egyed A, Wille R, Stock S. Model-driven engineering of safety and security software systems: a systematic mapping study and future research directions. J Softw Evol Process. 2023;35(7):e2457. doi:10.1002/smr.2457.
18. Thramboulidis K, Kontou I, Vachtsevanou DC. Towards an IoT-based framework for evolvable assembly systems. IFAC-PapersOnLine. 2018;51(11):182–7. doi:10.1016/j.ifacol.2018.08.255.
19. Hernández-Ramos JL, Matheu SN, Feraudo A, Baldini G, Bernabe J, Yadav P, et al. Defining the behavior of IoT devices through the MUD standard: review, challenges, and research directions. IEEE Access. 2021;9:126265–85. doi:10.1109/ACCESS.2021.3113678.
20. Stalnaker T, Wintersgill N, Chaparro O, Di Penta M, German DM, Poshyvanyk D. BOMs away! inside the minds of stakeholders: a comprehensive study of bills of materials for software systems. In: Proceedings of the IEEE/ACM 46th International Conference on Software Engineering (ICSE). New York, NY, USA: Association for Computing Machinery; 2024. p. 1–13. doi:10.1145/3597503.3623347.
21. OWASP. Authoritative guide to CBOM; 2024 [cited 2025 Sep 5]. Available from: https://cyclonedx.org/guides/OWASP_CycloneDX-Authoritative-Guide-to-CBOM-en.pdf.
22. Matheu García SN, Skarmeta Gómez A. Defining the threat manufacturer usage description model for sharing mitigation actions. In: 2022 1st International Conference on 6G Networking (6GNet). Piscataway, NJ, USA: IEEE; 2022. p. 1–4. doi:10.1109/6GNet54646.2022.9830415.
23. Boyens J, Smith A, Bartol N, Winkler K, Holbrook A, Fallon M. Cybersecurity supply chain risk management for systems and organizations. NIST special publication 800-161 revision 1; 2022. doi:10.6028/NIST.SP.800-161r1.
24. Waltermire D, Quinn S, Booth H, Scarfone K, Prisaca D. The technical specification for the security content automation protocol (SCAP) version 1.3. Gaithersburg, MD, USA: National Institute of Standards and Technology (NIST); 2018. doi:10.6028/NIST.SP.800-126r3.
25. National Institute of Standards and Technology (NIST). OSCAL model documentation; 2024 [cited 2026 May 15]. Available from: https://pages.nist.gov/OSCAL-Reference/models/.
26. IETF. A universally unique identifier (UUID) URN namespace; 2005 [cited 2026 May 15]. Available from: https://www.ietf.org/rfc/rfc4122.txt.
27. Open Home Foundation. Home assistant; 2026 [cited 2026 May 15]. Available from: https://www.home-assistant.io/.
28. Home Assistant. Developing an app; 2025 [cited 2026 May 4]. Available from: https://developers.home-assistant.io/docs/apps/.
29. Ruzafa P. ha-fdo-addons; 2025 [cited 2026 May 4]. Available from: https://github.com/peri324/ha-fdo-addons.
30. FIDO Device Onboard. pri-fidoiot; 2025 [cited 2026 May 4]. Available from: https://github.com/fido-device-onboard/pri-fidoiot.

31. Matheu S. DSP repository; 2025 [cited 2026 May 4]. Available from: https://github.com/saranieves92/DSP.
32. CERTH/ITI. ITI smart home. [cited 2026 May 4]. Available from: https://smarthome.iti.gr/.
33. Angelogianni A, Politis I, Xenakis C. How many FIDO protocols are needed? Surveying the design, security and market perspectives. arXiv:2107.00577. 2021.
34. Fries S, Falk R. Device onboarding transparency—supporting initial trust establishment; 2024 [cited 2026 Apr 24]. Available from: https://personales.upv.es/thinkmind/dl/conferences/securware/securware_2024/securware_2024_2_40_30026.pdf.
35. CycloneDX. CycloneDX v1.7 JSON reference; 2024 [cited 2026 Apr 24]. Available from: https://cyclonedx.org/docs/1.7/json/.
36. OASIS. Common security advisory framework version 2.0; 2022 [cited 2026 Mar 27]. Available from: https://docs.oasis-open.org/csaf/csaf/v2.0/os/csaf-v2.0-os.html.
37. OpenVEX. 2023 [cited 2026 Mar 27]. Available from: https://github.com/openvex.
38. Matheu Garca SN, Molina Zarca A, Hernndez-Ramos JL, Bernab J, Skarmeta Gmez A. Enforcing behavioral profiles through software-defined networks in the industrial internet of things. Appl Sci. 2019;9(21):4576. doi:10.3390/app9214576.