manySECURED

ManySecured: What is it ?

© 2022 IoT Security Foundation

ManySecured Whitepaper

ManySecured: what is it ?

ManySecured is developing an open ecosystem to help routers and gateways better protect the network against IOT attacks.

This approach complements other initiatives which are enhancing IOT endpoint security. ManySecured accepts the fact that there are many insecure IOT devices already in the market, which are not going away any time soon. In addition, even the best IOT endpoint security can and will be compromised. Enhancing the role that the gateway can play in the detection and protection against IOT attacks is and always will be a valuable addition to a comprehensive security strategy.

This overarching objective breaks down into two primary problems we need to solve:

1. How to share security relevant data needed to "detect and protect" between multiple and shifting parties?
2. How do we implement the real time control loop to "detect and protect" at the gateway level?

We define a suite of interoperable specifications that directly solve these problems. These specifications are underpinned by open source implementations and some emerging open source databases.

ManySecured Architecture

The ManySecured Specifications are defined against a concrete deployable architecture which mirrors most real world deployments.

This deployable architecture has the following characteristics.

- IOT devices: the portfolio of endpoint devices which are arctic at any point in time. Each of which is connected to one or more gateways.
- Gateway device: this is the device through which the IOT device connects to the wider world. This could be simply a WIFI router, or could be a Zigbee, Zwave or LoRa router etc.
-

For the ManySecured Controller to make meaningful decisions it needs access to information about the devices it is "controlling". We term this security device metadata;

data about devices pertaining to security.

There are two coarse grained classifications of this security metadata.

- Device type data: this is data pertaining to a abstract device type (the class of device to which a particular device instance belongs)
- Device instance data: data relating to the history of this specific device instance.

Device Types and Instance

The distinction between device instances and device types is critical to the design of the ManySecured system and we would argue critical to meaningful implementation of real world IOT security systems. it is worth looking at this a little more closely.

Type data is the lynchpin of the D3 system. An assertion of type is a claim that an abstract physical type of device exists (commonly referred to as a SKU). When asserting a type we provide an immutable GUID and one or more URIs, which can be used to refer to this type.

The assertion of type provides a common language for making interoperable security statements: e.g.

- This device instances is of this **device type**
- This **device type** has this expected behaviour
- A vulnerability has been identified for this **device type**
- This **device type** has these available firmwares

There are many examples of device type metadata that already exists in the security ecosystem: SBOM, CVE, MUD statements.

Device instances, by contrast, are related to the specific physical instance of a device. Examples of device instance data could be MAC address, IP address, public key etc. These are ephemeral properties that help us identify a specific device instance.

But when making security assessments about devices, the history of the device is highly material. Hence, data which tells us about where the device came from, who brought it, who tested it, who configured it is all very important. We call these device instance lifecycle management events.

ManySecured Defensive Control Loop

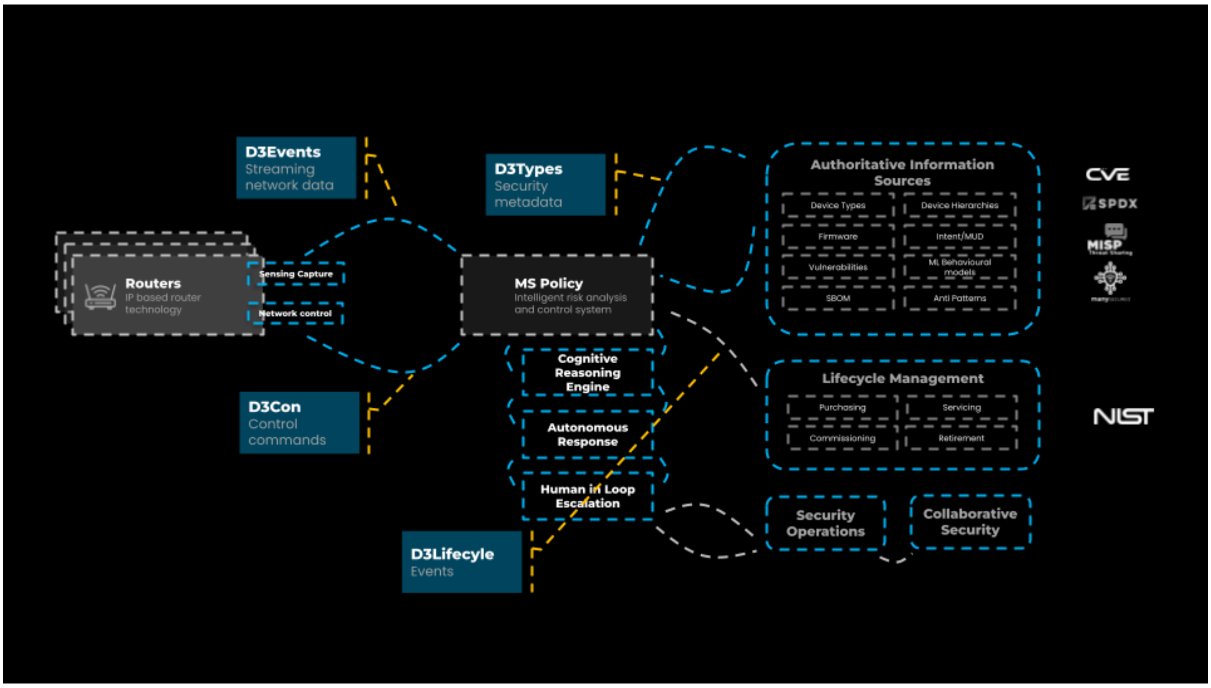
The vision of ManySecured is encapsulated as an intelligent defensive control loop. This control loop consists of four stages.

1. **Sensing**: retrieve all real time information pertaining to a device instance. Monitor device activity and feed it to the MS defensive Controller.
2. **Inform and learn:** feed the MS defensive Controller with all "useful knowledge" pertaining to the observed devices. These could be lifecycle device events

pertaining to device instances and security meta-data relating to relevant device types.

3. **Reason:** based on all the data available to the MS defensive controller, reason about device instances. Attempt to classify device instances by device type. Make security assessments of risk based on device type information and observed behaviours. Recommend an action.
4. **Action**: finally, do something about the current perceived risk. Make a real world change to the system.

ManySecured Standards



By mapping these abstract notions onto concrete architectural elements, we get the following

Sensing: the D3Events API defines a precise interoperable method by which the MS controller can observe critical network based events from the gateway. The D3EVents API defines both the mechanism of transport and the data structure (schema) of the messages passed for each critical event.

Inform/Learn: the MS controller receives all the pertinent security metadata from authoritative sources, by receiving updates of D3 Claims. Each D3Claim is a predefined data structure conveying critical information needed for security decision making. Each D3 Claim (a Verifiable Credential under the hood) has in-built provenance information; the signatory of the claim. Each MS controller determines which authoritative resources it uses for its data capture. D3 Claims can be received through straight HTTP downloads, synchronisation protocols (eg. Git), or indeed events can be streamed using the D3Event API. These “learned " D3 Claims come in two principle forms.

a) Type information: where new information relating to device types is published;
b) Instance lifecycle information, where new information regarding device instance lifecycle events are provided.

This knowledge is provided by external systems and many MS controllers may have access to similar information.

Reasoning: the MS controller is responsible for reasoning about the information provided to it. This is the aggregate of all realtime D3 Events, D3 Types and D3 Workflow information provided to it. The method by which the MS controller reasons and the logic or process it uses is left as an undefined implementation detail. Most D3 Controllers will implement a form of memory (database/knowledge base) to store information long term.

The result of this black box reasoning process will in most cases be some form of external action on the network.

Action

Finally the MS Controller will take action to contain any perceived security risk. This is implemented by an API between the MS controller and the IOT gateway or gateways in question. This API is defined in ManySecured by the DCON API. The DCON API provides the practical methods to allow the gateways network to be reconfigured or for individual IOT endpoint devices to be disabled or constrained.

D3DB: Many Secured Type Database

The D3 type specifications are by design distributed and will support many sources of D3 type information and the creation of many aggregators or curators of D3 information. There is nothing in the specification that requires or forces a single source of truth. This explicit design decision reflects the reality of trust relationships in the real world. Not all companies trust each other, not all governments trust each other. And in a world where a physical IOT device might outlive the company that manufactured it we need a trust and information model built for legacy and built for the long term.

There is clear value, however, in publishing a “best known” database as a primary source of truth. Something we will call D3DB. This provides stable URI hosts of a master domain that server the D3 Type information for the most commonly used devices. This information could be served directly from the D3DB host or the D3DB host could proxy to the originating claim issuer.

Clearly for this to work there needs to be an active curation function to determine which claims to accept, which to reject and which to give priority to when conflicts arise.

We envision an open source like governance system for the management of this curation process

Foundational assumptions

Behind the design of the ManySecured system are certain foundational assumptions, or requirements which have shaped the nature of the solution. These assumptions are formally documented in the specifications but the key ones can be quickly summarised.

- Legacy: manysecured is designed to support legacy devices. By this we mean populations of IOT devices already deployed and for which no updates are ever likely in the future. But these devices do present a real and continued challenge.
- Flexible trust: trust in the D3 ecosystem is "subjective". It is up to the consumer of the D3 Claims to determine their level of trust in the data presented. Practically, this is most easily expressed by asking the question: do you always trust the originating manufacturer to present the best and most up to date security information on their devices.
- Operational continuity: following on from the above point, physical devices will often outlive the company that manufactured them. The information economy that underpins the security must be able to handle data flows and operational systems when originating manufactures are no longer in business.
- Crowdsourcing: and to underpin the above two concepts, the operational D3DB system must support many sources of information and knowledge, including crowdsourcing.

Continuous assurance for IOT devices

The intelligent control loop that the ManySecured system defines, encourages you to subtly reframe some standard device security processes. Traditional data flows often address device authentication and device authorisation and that becomes the bedrock of the security processes. In ManySecured we substitute these processes with the notion of **continuous assurance.** For the device to securely operate we need to implement notions of authentication and authorisation, but these are not one off, single shot tests to be performed at device onboarding time. They are continuous processes that must be tested on first contact, but must be continually reevaluated. The question of whether the network owner trusts the device is based on many factors, and all of these factors change continuously. This continuous trust assessment is based on, for example: the behaviour of the device, the behaviour of other devices of this type, disclosed vulnerabilities on this device type, the current software/upgrade status of this device, etc etc. This continuous assurance process is in many senses merely a practical application of zero trust principles, to device trustworthiness assessment.

Non binary trust

A second underlying principle of ManySecured is the nature of the trustworthiness assessment. Many trust assessments based on PKI cryptography produce a binary assessment of whether this device is trusted or not. But the obvious reality of the situation is trust is not binary; security is not black and white. The design of Many Secured reflects this. The underlying technology on which D3 is based is Verifiable Credentials. Verifiable Credentials was originally designed to “reboot the web of trust” model, which rely less on immutable trust anchors delivering black and white assessments and more on multiple, group source trust anchors delivering a variable trust assessment. We believe this approach is essential to practically address the legacy and forward looking IOT security problem

Engagement

ManySecured is a fully fledged working group of hte IOT Security Foundation. The underlying specification and open source reference implementation are well established. We are now in the phase of community building, database building and of course iterative improvement and evolution. To get engaged please use the links below

Resources and Links

Website: https://manysecured.net/

Specifications: https://specs.manysecured.net/

Specification source: https://github.com/TechWorksHub/ManySecured-WGs

Open source gateway reference: https://github.com/nqminds/edgesec/issues

Email: info@manysecured.net