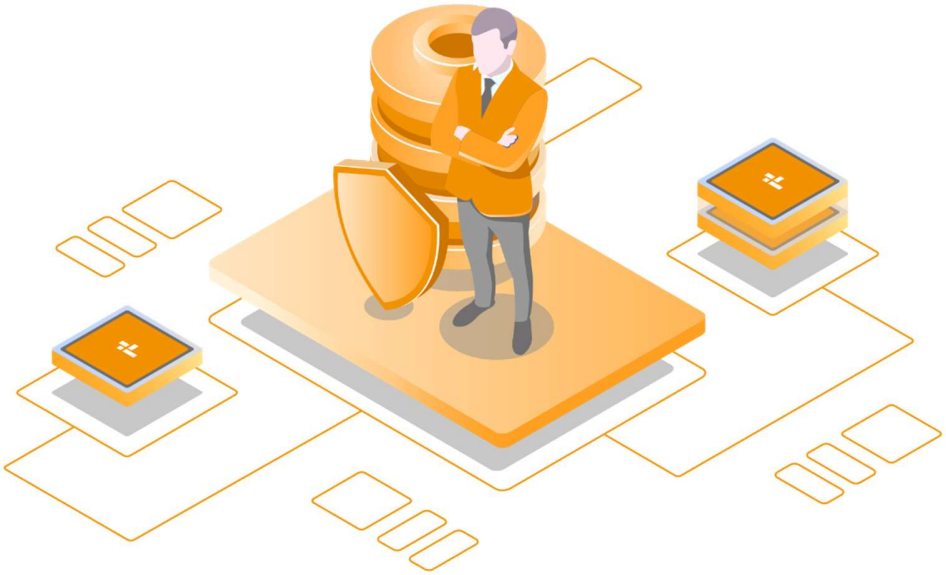
Fog Protect



D7.2 – Dynamic Risk Management Tools: Final Prototypes and Technical Report

Grant Agreement no: **871525** Project Acronym: **FogProtect** Lead Beneficiary: **IT Innovation**

Project title: **FogProtect – Protecting sensitive data in the computing continuum**

Work package: **WP7 - Dynamic Risk Management**

Document version: **1.0** Publication date: **October 2022**

Dissemination Level		
☒	PU:	Public
☐	PP:	Restricted to other programme participants (including the Commission Services)
☐	RE:	Restricted to a group specified by the consortium (including the Commission Services)
☐	CO:	Confidential, only for members of the consortium (including the Commission Services)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 871525.

Executive Summary

This deliverable has described final enhancements to the dynamic risk management components from FogProtect WP7 that detect events and threats and assess the risk levels due to events; and assess risks of adaptation proposals from the Adaptation Coordinator in WP5.

In Period 2, the upstream architecture to the Threat Diagnosis has changed significantly reflecting experiences from 2021. This has addressed the need for aggregation and filtering of 'raw' events became evident, not only to stop flooding the Managing Components of FogProtect, but also the need to differentiate between temporal (and perhaps accidental) behaviour and deliberate misbehaviour.

We have modelled populations needed for in scenarios fog-cloud enabling the modelling of a multiplicity of fog nodes efficiently. We have shown and highlighted two different cases of population behaviours, one corresponding to redundancy (e.g. relating to availability of redundant resources) and the other to where just a single member of a population causing an effect results in that effect being present in the population as a whole (e.g. where a data leak of a single copy of a data set results in a loss of confidentiality). From these, three different cases of population behaviour have been determined: the lowest, highest, and average likelihood cases, and these form the basis of population modelling. We have also shown how population sizes affect population affect likelihood, where each population level causes a specific shift in the likelihood level, which resulted in a lookup table to determine relating population size to changes in likelihood (e.g. of a threat). We have presented a worked example to illustrate the differences when moving from a singleton asset to an asset population.

We have provided an extension to enable recommendations of controls that address risks to be fed to the Adaptation Coordinator (or any other client). This provides suggestions for runtime controls based on examination of the risk model's "attack graph", which is a representation of the threat and risk propagation throughout the modelled system. The SSM can automatically suggest controls to block threats at each point in the attack graph, and the recommendations work has determined an algorithm that analyses the attack graph to determine the optimum controls and control strategies to block the threats. We have presented an example of a recommendation that lowers a risk level from "very high" to "medium".

Overall, this deliverable has described extensions to an integrated framework for dynamic (runtime) event detection, threat detection and risk analysis to specifically support situations commonly found in fog-cloud environments. This work is currently being evaluated in WP2 in the project's three use cases, and the results from this evaluation will be reported in the upcoming D2.4, due PM36, where updated risk models constructed to model the enhanced scenarios for each of the project's use cases will be demonstrated.

Disclaimer of warranties

This document has been prepared by the FogProtect project partners as an account of work carried out within the framework of the contract No 871525.

Neither the Project Coordinator, nor any signatory party of the FogProtect Project Consortium Agreement, nor any person acting on behalf of any of them:

- makes any warranty or representation whatsoever, express or implied, with respect to the use of any information, apparatus, method, process, or similar item disclosed in this document, including merchantability and fitness for a particular purpose, or
- that such use does not infringe on or interfere with privately owned rights, including any party's intellectual property or
- that this document is suitable for any particular user's circumstance; or
- assumes responsibility for any damage or other liability whatsoever (including any consequential damages, even if the Project Coordinator or any representative of a signatory party of the FogProtect Project Consortium Agreement, has been advised of the possibility of such damages) resulting from your selection or use of this document or any information, apparatus, method, process, or similar item disclosed in this document.

FogProtect has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 871525. The content of this deliverable does not reflect the official opinion of the European Union. Responsibility for the information and views expressed in the deliverable lies entirely with the author(s).

List of Contributors		
Participant	**Short Name**	**Contributor(s)**
University of Southampton IT Innovation Centre	IT Innov	Steve Taylor Panos Melas Ken Meacham Mike Surridge Samuel Senior Stephen C Phillips
Nokia Bell Labs	Nokia	Norbert Goetze

List of Reviewers		
Participant	**Short Name**	**Reviewer(s)**
Athens Technology Center	ATC	Anestis Sidiropoulos
Ubiwhere	UW	Sofia Rosas

Change History			
Version	Date	Short Name	Description
0.1	2022-06-20	UoS	Initial draft
0.2	2022-08-23	UoS	Add populations
0.3	2022-09-07	UoS	Update threat diagnosis API from D7.1 & add recommendations
0.4	2022-10-10	Nokia	Add SEIA
0.5	2022-10-21	UoS	Complete draft for review
1.0	2022-10-31	UoS	Address comments from internal review

Contents

Figures

Tables

1. Introduction

This deliverable describes the advances made to create the risk management framework in WP7 that enables design time cyber security risk assessment enhanced with dynamic event management, threat detection and runtime risk assessment.

In summary, the key advances made are as follows:

- The SIEA has been extended to accommodate input from vulnerability scanners such as Wazuh and to aggregate events coming from them into composite events that are used in risk assessment.
- Research has been undertaken to support populations of assets in the UoS System Security Modeller (SSM) tool. This is necessary to efficiently model multiplicities of elements (e.g. hundreds of local fog nodes) in risk models.
- Recommendations from the SSM are computed at the same time as risk assessment if the risk level is too high (e.g. above "Low"). Recommendations are included in outgoing messages from the WP7 tools and contain suggested security controls to lower the risk level.

The overall architecture within the WP7 components presented in D7.1 still applies, although it has been adjusted to accommodate the enhancements discussed above. A reprise of the WP7 architecture figure from D7.1 follows with the adjustments' positioning highlighted in bold italic.

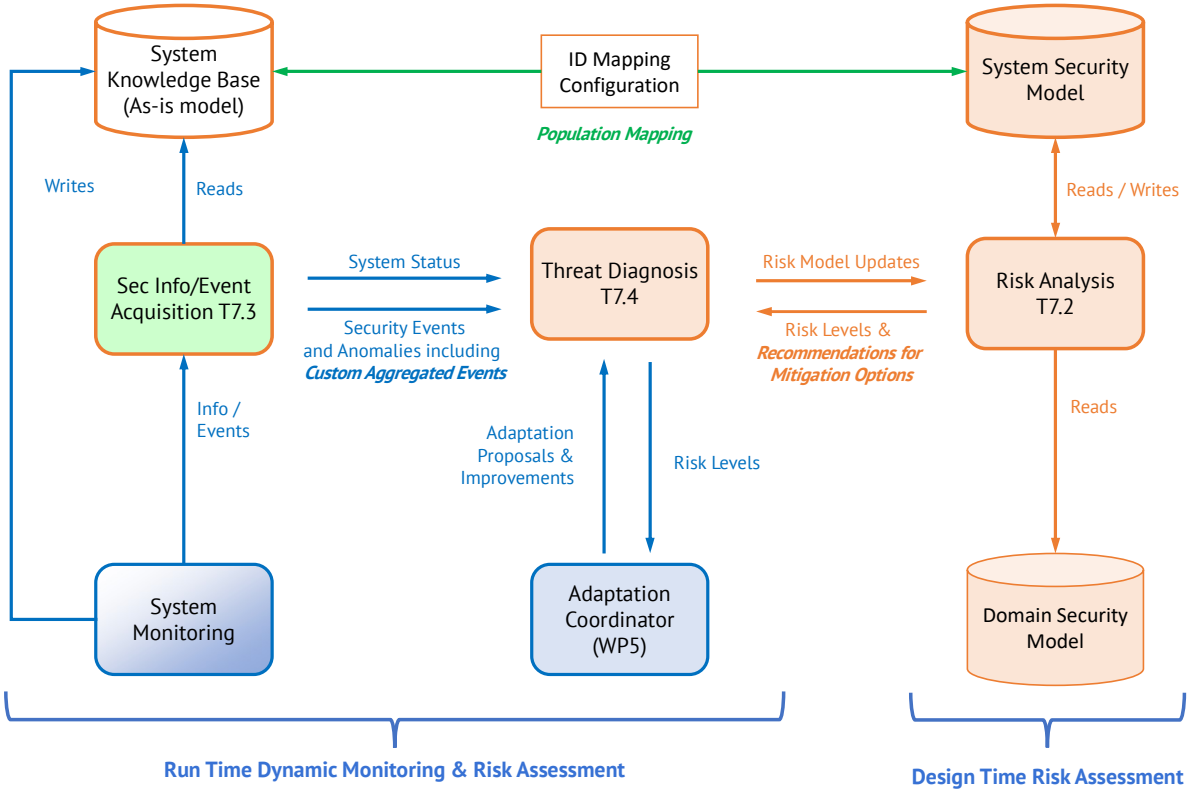


Figure 1: Enhancements to Threat Detection & Risk Modelling

The current status of this work is that these enhancements are being tested in deployments for the project's three use cases (smart cities, smart manufacturing and smart media). Some examples regarding these use cases are given in this

deliverable and full examples from the evaluations in the project's use case workpackage (WP2) will be reported in D2.4, due PM36.

This deliverable's structure follows the order of the advances as described above. Firstly Section 2 describes advances to the SIEA, the Populations modelling research is described in Section 3 and the Threat Detection, Risk Modelling & Recommendations work is described in Section 4.

2. Security Information and Event Acquisition Pipeline

The 'Security Information and Event Acquisition' (SIEA) is a microservice located at the front-end of the 'Run Time Dynamic Monitoring and Risk Assessment'. Its main purpose is to inform 'Threat Diagnosis' about security events and anomalies during operation. Its position and its communication partners in the overall FogProtect architecture are shown in *Figure 2*.

The SIEA (marked in red) relies on receiving active scanning events from the 'Vulnerability Detector Component' via the 'Monitoring Component' (both marked in orange) and other 'raw' events directly received by the 'Monitoring Component'. The SIEA 'pipeline' therefore contains not only the SIEA microservice, but also the Vulnerability Detector Component and the Monitoring Component.

In this high-level visualization, the most obvious change in the architecture during the second half (P2) of this project, is that the Vulnerability Detector now connects to the Monitoring Component instead of having direct access to the SIEA (compare with Figure 1 of D7.1). The advantage for this is that with this change, the Monitoring Component, as a single point of entry, can aggregate and filter all variants of detected 'raw' events. With this new enabler, we can differentiate between unintentional and deliberate (mis)behaviour. On top, the component can control the flow of tasks required to be processed by the SIEA based on the readiness of the SIEA and it can order the sequence of tasks based on the priorities of these tasks.

In P2, the Vulnerability Detector is deployed in the use-cases 'Smart Cities' (UC1) and 'Smart Manufacturing' (UC2). In 'Smart Media' (UC3), a new feature in the Monitoring Component has been implemented which aggregates granted and blocked access attempts of users enabling the enforcement of service level agreements (SLAs). In all use cases, the capability of scanning the hosts for cybersecurity vulnerabilities (see https://cve.mitre.org/) has not been used, but this functionality can easily be activated by configuring the system accordingly. Contrary to P1 of the project, the severity of active scanning events will be defined by the Vulnerability Detector itself in UC1 and UC2. The severity of all other events in which the source has no knowledge about the potential impact in the managed system will be defined by the Monitoring Component. This applies specifically in UC3 in which we aggregate the 'raw' events generated by Fybrik and check for recurrent misbehaviour of potentially untrustworthy users.

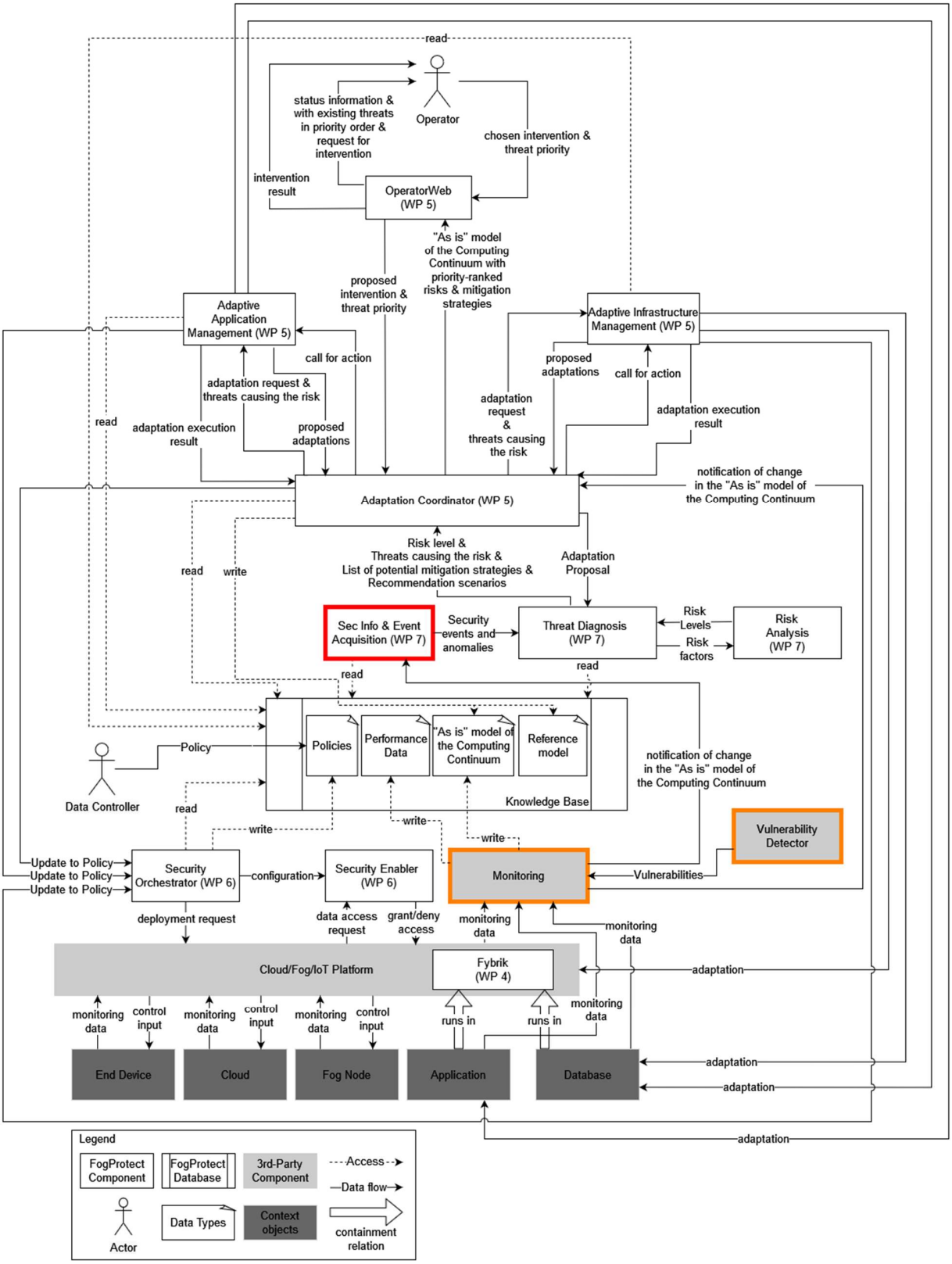


Figure 2: The Monitoring and the Vulnerability Detector Components are the Source of Events for the SIEA

2.1. The Monitoring Component

The most prominent change in P2 of the project is reflected in the Monitoring Component which no longer contains the 'Kafka Latch'. The 'intelligence' of the 'Kafka Latch', which was mainly blocking an illegitimate sequence of events, is now located in the SIEA and in the 'As-Is Knowledge Base'. If e.g.: a threat is already mitigated by adaptations triggered by the Adaptation Coordinator, and the same event is detected again, the As-Is Knowledge Base can terminate the sequence at an early stage by responding to the SIEA with an empty list in the ChangesMadeToAsIsModel field (see comment in box after message #4 in *Figure 15*). Regardless of repetitions of events, the ChangesMadeToAsIsModel will also contain an empty list in case a new type of event is received, but the risk of this new event has already been mitigated via a previously executed adaptation.

In P2, the Monitoring component contains next to the Kafka Broker an Aggregator and a Distributor (see *Figure 3*).

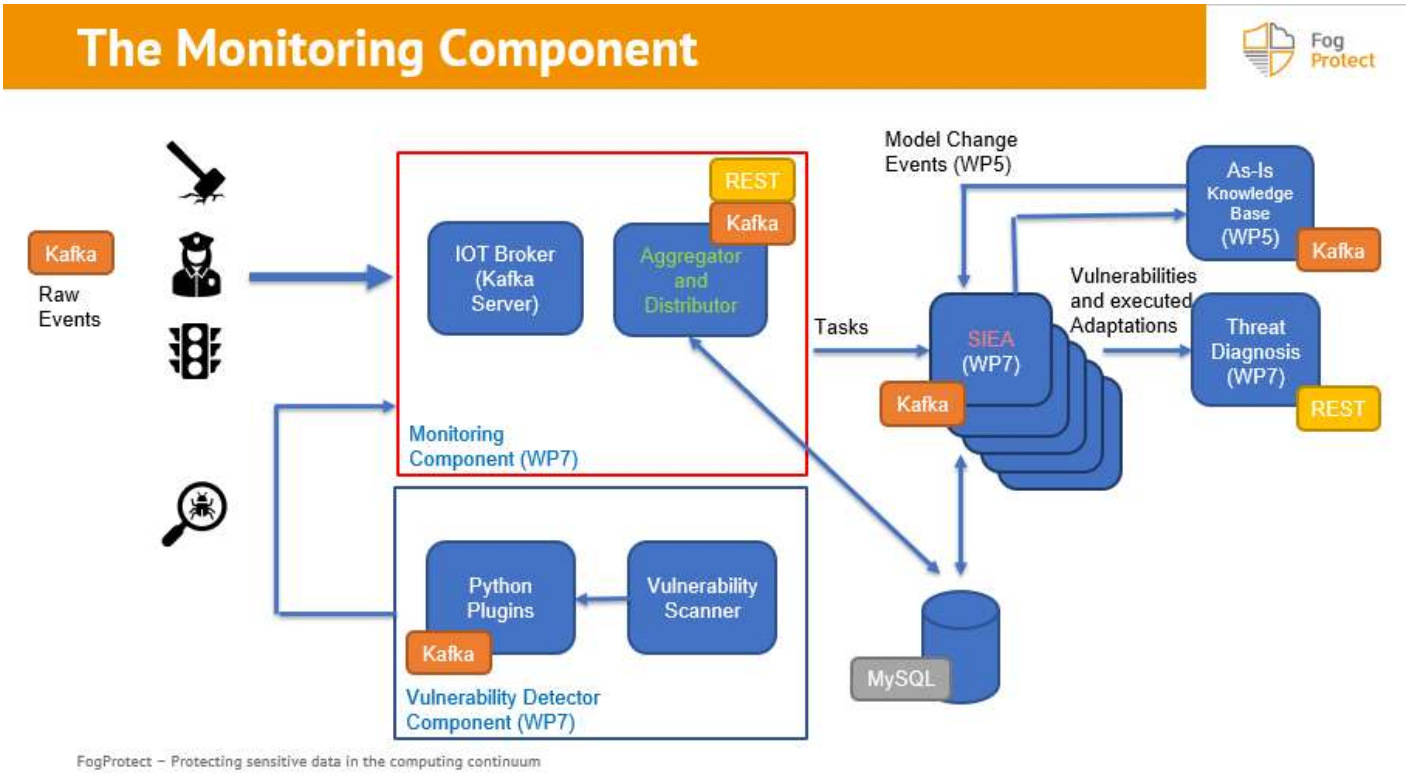


Figure 3: The Monitoring Component as single point of Entry for 'raw' Events

2.1.1. Kafka Broker

The Monitoring Component and the Vulnerability Detector will publish their notifications via a Kafka Broker (https://kafka.apache.org/intro). This has as advantage that the events can be stored as streams without corrupting their order and that services can easily be notified about such events by subscribing to them.

In FogProtect, we install Kafka based on https://github.com/wurstmeister/kafka-docker licensed under the Apache License 2.0. Kafka will be instantiated via docker-compose which activates both the Apache Kafka Broker and the Apache Zookeeper containers. The latter is required to store the metadata of the broker.

The Kafka Broker is configured to handle a set of topics that are derived from the requirements of the project's use cases, as well as general-purpose topics suitable for all use cases. The use cases (UCs) have changed significantly in P2 and are described in detail in D2.3. Briefly, UC1 covers smart cities where fog nodes in metal cabinets are attached to lamp posts in a city environment and gather and process CCTV footage, and UC2 covers smart manufacturing focusing on fog nodes located in a so-called 'Factory in a Box' (FiaB) – a complete smart manufacturing setup located within a shipping container. And UC3 focusses on enforcing SLAs on trusted users and blocking untrusted/malign users with different roles interacting with privacy-related video footage.

The following is a list of P2 Kafka topics:

a) ClearanceEvent
In all use-cases, the clearance event notifies that the system can go back to normal operation. Clearance means that one or more adaptations have been executed to mitigate the risk(s).

b) ResetEvent
Used for testing. All services listening to this topic can define and implement the appropriate actions. At this point in writing, it resets the Aggregator, the Distributor, the SIEA, the As-Is Knowledge Base and the Threat Diagnosis and brings them back to a normal operation. With the reset, all previously imposed adaptations are removed.

c) ModelChange
In P2, the SIEA uses this topic to inform the Knowledge Base that a security- or privacy-related event has been reported or to query for executed adaptations.

d) ModelChange4SIEA
When the status of a managed system changes, the Knowledge Base will send an 'As Is' json message targeted for the SIEA using this topic, so that the SIEA can notify the 'Threat Diagnosis' about the change in the security situation. There are two types of ModelChange4SIEA messages: one before the execution of adaptations and one after the adaptations have been made.

e) AdaptationChange4SIEA
An AdaptationChange4SIEA message can be generated by both the Adaptation Coordinator and the Monitoring Component. For the former, it signals that an adaptation has been executed or has been finalized. For the latter, it signals that a new security- and/or privacy-related event has been detected.

f) ConfidentialityAlert
This message is generated by the Vulnerability Detector to inform the SIEA via the Monitoring Component, that a case of data leakage has been detected. This event is used only in Smart Manufacturing (UC2).

g) IntegrityAlert
This more severe message is generated by the Vulnerability Detector to inform the SIEA via the Monitoring Component, that a case of data manipulation has been detected. This event is used in both the Smart Manufacturing use-case (UC2) and in the Smart Cities use-case (UC1).

h) granted-access
This event is triggered by Fybrik and reflects that a user was allowed to access specific endpoints. The Aggregator will monitor such granted raw events. If too many occur, an aggregated event will be sent to the SIEA via the Distributor.

i) blocked-access
This event is triggered by Fybrik and reflects that a user attempted to access specific sensitive data he was not authorized for. The Aggregator will monitor such blocked raw events. If too many occur, an aggregated event will be sent to the SIEA via the Distributor.

j) unblocked-access
The managing system will block users, roles and/or organizations based on the aggregated events from h) and i). With this event, generated by the Adaptation Coordinator, an unblocking will be initiated.

2.1.2. Aggregator

The Aggregator receives the 'raw' events from the managed system (Cloud/Fog/IoT Platform – this includes Fybrik – WP4 and physical sensors located in the managed system), from the Vulnerability Detector (Active Scanning Events), from the Adaptation Coordinator (Unblocking Events) and from the administrators (Clearance Events).

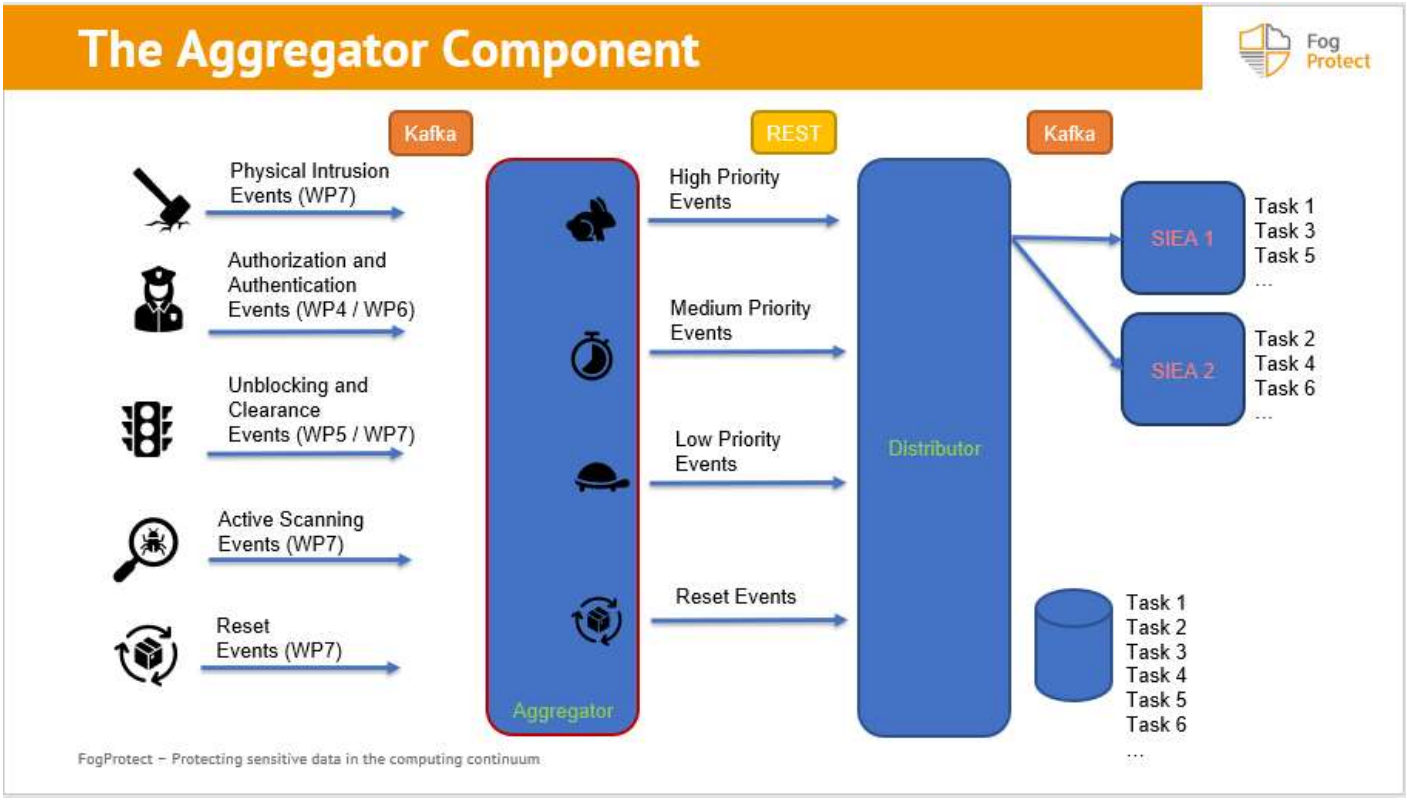


Figure 4: The Aggregator Component

The Aggregator is a Kafka Consumer (see *Figure 4*): it is only active during the processing time of incoming events and thereby handles the incoming events sequentially. As an example of the aggregator's use, for the Smart Media (UC3) use-

case, the Aggregator sums up the granted and blocked access events detected within a configurable timeslot, i.e.: a sliding window. If the number of events exceeds a specific threshold, a message will be sent to the Distributor via a RESTful POST message. For the data leakage (UC2) and data manipulation (UC1 and UC2) scenarios, the Aggregator can be made aware of planned maintenance periods. It would then suppress forwarding leakage (f) and manipulation (g) events to the SIEA via the Distributor if it is known that an authorized administrator is performing changes in the systems. All 'raw' events must pass through the Aggregator. Historic events outside of the sliding window will be discarded.

In the case of an ResetEvent (b), the Aggregator immediately forwards the RESET to the Distributor after clearing all in-memory data aggregated for high, medium, and low priority events.

Next to aggregation and filtering, the Aggregator forwards processed (and potentially aggregated) events to dedicated endpoints of the Distributor based on their priority. In the case of Smart Cities and Smart Manufacturing, the priority is mapped directly to the 'serviceLevel' reported by the Vulnerability Detector. In the case of Smart Media, the Aggregator sets the 'serviceLevel' typically to 'medium'. After pushing a processed event to the Distributor, the Aggregator will enforce a 'silence period' meant to suppress repetitions and to reduce unnecessary workload and could lead to benign denial of service due to overload. During this 'silence period', the aggregation of the same type of 'raw' events will still be active, but the reporting of the same type of 'processed' event towards the Distributor will be suppressed.

2.1.2.1 Configuration Examples

During power-up, the Aggregator checks for the presence of a configuration file and reads its content. Depicted in *Figure 5* is an excerpt of such a configuration file which shows how the Aggregator receives the information required to enforce the SLAs defined for Smart Media. So e.g. the activities of all editors from the organization ATC will be monitored using a sliding window of 60 secs. If the endpoint EP2 has been accessed more than 40 times within this window, an aggregated event will be posted to the SIEA via the Distributor. EP2 is a combination of a method and a URL provided by Fybrik from WP4. Any number of additional endpoints can be added as columns to this table.

```
25
26 # granted access groups (=keys/handles) with SLA details group1, group2, windowSize, all_endpoints, EP0, EP1, EP2, EP3, ...
27
28 # g1 = group1
29 # g2 = group2
30 # ws = windowSize
31 # all = all_enpoints
32 # EP0 = POST chatterbox-be.chatterbox.svc.cluster.local:4000/videos/
33 # EP1 = POST chatterbox-be.chatterbox.svc.cluster.local:4000/survey/answer
34 # EP2 = GET chatterbox-be-secure.fogprotect.com/videos/
35 # EP3 = GET chatterbox-be-secure.fogprotect.com/surveys/
36
37 #                  g1             g2  ws  all     EP0     EP1     EP2     EP3
38 #                  ------------------------------------------------------------
39 gaGroup gaUser_ATC  user           ATC 20  10      5       5       10      10
40 gaGroup gaUser_VRT  user           VRT 20  10      5       5       10      10
41 gaGroup gaEdit_ATC  editor         ATC 60  40      0       0       40      0
42 gaGroup gaEdit_VRT  editor         VRT 60  40      0       0       40      0
43 gaGroup gaOper_ATC  operator       ATC 60  30      15      15      0       30
44 gaGroup gaOper_VRT  operator       VRT 60  30      15      15      0       30
45 gaGroup gaOrg_ATC   organization   ATC 60  100     0       0       0       0
46 gaGroup gaOrg_VRT   organization   VRT 60  100     0       0       0       0
47
```

Figure 5: A Granted Access Configuration Example

Figure 6 depicts the rules for aggregating and reporting blocked access attempts to endpoints. In case e.g.: an unauthorized (but authenticated) user from VRT tries to access forbidden endpoints, an event will be reported to the SIEA via the Distributor if the user tried it 31 times within 60 seconds. Contrary to SLAs, the method and the URL need not to be taken into account.

```
15 # blocked access groups (=keys/handles) with group1, group2, windowSize and all_endpoints
16
17 # g1 = group1
18 # g2 = group2
19 # ws = windowSize
20 # all = all_enpoints
21
22 #                  g1             g2  ws  all
23 #                  ---------------------------
24 baGroup baUser_ATC  user           ATC 60  10
25 baGroup baUser_VRT  user           VRT 60  10
26 baGroup baEdit_ATC  editor         ATC 60  30
27 baGroup baEdit_VRT  editor         VRT 60  30
28 baGroup baOper_ATC  operator       ATC 60  30
29 baGroup baOper_VRT  operator       VRT 60  30
30 baGroup baOrg_ATC   organization   ATC 60  50
31 baGroup baOrg_VRT   organization   VRT 60  50
```

Figure 6: A Blocked Access Configuration Example

2.1.3. Distributor

The Distributor receives the processed, potentially aggregated, and prioritized events from the Aggregator and forwards these events as tasks to the SIEA.

The Distributor is implemented as a CherryPy server and is provided with a set of endpoints which map to the priority of the processed events. Next to this, a browser can connect to this CherryPy server as it has been provided with a control functionality for triggering 'raw' events and displaying the status of the system (see *Figure 7*). The Distributor stores a list of tasks in-memory and offers these tasks sequentially to the SIEA instances based on their priority (highest priority task first).

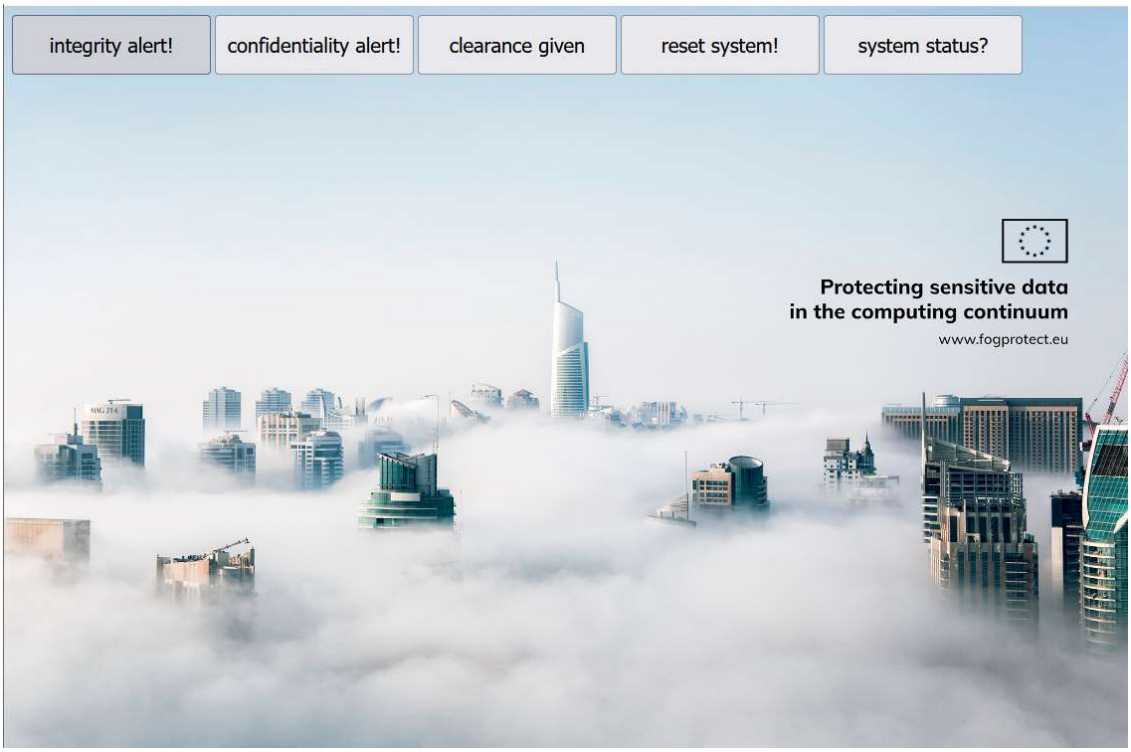


Figure 7: FogProtect GUI provided by the Distributor

New in P2 of this project is that the Distributor can connect to multiple SIEA instances and thereby keep track of the status' of these instances. This feature is enabled by a MySQL table called 'SIEA_TASK_STATUS'. Before launching a new task for the SIEA to process, the Distributor will add a new record into this table containing the task_id (see *Figure 8*). When the SIEA instance picks up this task, it will identify the relevant record in the table using the value of the 'SieaTaskId' parameter it received via Kafka, and update the task status of this record during its processing. As a side effect, this table also enables an administrator to check the sanity of all SIEA instances.

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

start_time	update_time	name	topic	message	task_id	task_status	idx
2022-09-09 15:22:57	2022-09-09 15:22:59	siea1	DataManipulationDetected	{"reason": "Sensitive data could have been r...	2-a1d91606-5c00-45a6-9...	jidMissing	14
2022-09-09 15:28:51	2022-09-09 15:33:00	siea1	DataManipulationDetected	{"reason": "Sensitive data could have been r...	1-475c3e4e-5e15-44ed-8...	reset	15
2022-09-09 15:30:34	2022-09-09 15:30:34	NULL	NULL	{"reason": "Sensitive data could have been r...	2-c8901608-cbc5-4ba3-81...	unassigned	16
2022-09-09 15:33:17	2022-09-09 15:34:46	siea1	DataManipulationDetected	{"reason": "Sensitive data could have been r...	1-10d4e4bb-3004-4256-a...	reset	17
2022-09-09 15:34:59	2022-09-09 15:35:38	siea1	DataManipulationDetected	{"reason": "Sensitive data could have been r...	2-37eab394-8227-43ec-a...	reset	18
2022-09-09 15:36:42	2022-09-09 15:36:43	siea1	DataLeakDetected	{"rule": "550", "agent": {"id": "001", "name":...	3-33974c96-41a0-474d-bf...	transientState1	19
2022-09-09 15:38:24	2022-09-09 15:38:24	NULL	NULL	{"rule": "550", "agent": {"id": "001", "name":...	4-f79084af-085d-462c-91...	unassigned	20
2022-09-09 15:39:07	2022-09-09 15:43:43	siea1	DataLeakDetected	{"rule": "550", "agent": {"id": "001", "name":...	1-602f9221-4680-452b-8d...	reset	21
2022-09-09 15:43:59	2022-09-09 15:44:00	siea1	DataLeakDetected	{"rule": "550", "agent": {"id": "001", "name":...	1-d656596a-f2a8-4865-9b...	transientState1	22
2022-09-09 15:48:25	2022-09-09 15:49:49	siea1	DataLeakDetected	{"rule": "550", "agent": {"id": "001", "name":...	1-01c223fb-cee1-44ba-8c...	reset	23
2022-09-09 15:54:28	2022-09-09 15:54:48	siea1	DataLeakDetected	{"rule": "550", "agent": {"id": "001", "name":...	1-ee2c005a-cf22-49cf-b5...	transientState2	24
2022-10-04 18:03:55	2022-10-04 18:04:07	siea1	DataManipulationDetected	{"reason": "Sensitive data could have been r...	1-f9b71a5e-12cc-4bbd-82...	transientState2	25
2022-10-05 08:47:37	2022-10-05 08:50:29	siea1	AdaptationExecuted	{"reason": "Sensitive data could have been r...	1-e0e15da5-6f8f-4db3-b9...	jobidMissing	26
2022-10-05 09:11:32	2022-10-05 09:11:34	siea1	TamperingResolved	{"agent": {"id": "001", "name": "green-ibis", ...	1-65551690-d007-47ed-b...	jobidMissing	27
2022-10-06 08:11:38	2022-10-06 08:12:00	siea1	DataLeakDetected	{"rule": "550", "agent": {"id": "001", "name":...	1-72ba2a86-9a29-4605-9...	transientState2	28
2022-10-06 08:36:33	2022-10-06 08:36:52	siea1	DataLeakDetected	{"rule": "550", "agent": {"id": "001", "name":...	1-63fd7cc3-8f1e-40f9-ba1...	transientState2	29
2022-10-06 08:41:42	2022-10-06 08:44:50	siea1	DataLeakDetected	{"rule": "550", "agent": {"id": "001", "name":...	1-ba3fab09-a507-46d6-9b...	jobidMissing	30
2022-10-06 09:06:18	2022-10-06 09:06:58	siea1	TamperingResolved	{"agent": {"id": "001", "name": "green-ibis", ...	2-70fe8adf-f8c9-4519-ac5...	reset	31
2022-10-06 09:09:27	2022-10-06 09:09:29	siea1	TamperingResolved	{"agent": {"id": "001", "name": "green-ibis", ...	1-13239b29-6c70-4c53-82...	jobidMissing	32
2022-10-06 09:43:46	2022-10-06 09:43:48	siea1	TamperingResolved	{"agent": {"id": "001", "name": "green-ibis", ...	1-6b10893e-2772-4d97-a...	noChange	33
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 8: SIEA_TASK_STATUS Table

In order to prevent triggering multiple concurrent adaptations, the Distributor checks a MySQL table called 'SIEA_BUSY_INDICATION' before launching a new task. The Distributor will only add a new record into this table if the SIEA is idle. The Distributor will then clear the busy_bit of a new record to 0 before pushing a new task with topic 'AdaptationChange4SIEA' for the SIEA into Kafka. The SIEA will pick up this task, update the SIEA_TASK_STATUS table (see *Figure 8*) and set the busy_bit of the last entry in the 'SIEA_BUSY_INDICATION' to 1 (see *Figure 9*). Only when the SIEA completed this task, it will clear the busy_bit back to 0. During the time the busy bit is set to 1, the Distributor will delay pushing a new task into Kafka.

start_time	update_time	name	busy_bit	idx
2022-09-08 15:39:24	2022-09-08 15:44:36	siea1	0	5
2022-09-08 15:47:42	2022-09-08 15:52:13	siea1	0	6
2022-09-08 15:52:48	2022-09-08 17:13:52	siea1	0	7
2022-09-08 17:13:59	2022-09-08 17:36:59	siea1	0	8
2022-09-08 17:37:06	2022-09-08 18:01:30	siea1	0	9
2022-09-08 18:01:37	2022-09-08 18:10:35	siea1	0	10
2022-09-08 18:10:43	2022-09-08 18:21:20	siea1	0	11
2022-09-08 18:21:27	2022-09-09 15:20:57	siea1	0	12
2022-09-09 15:21:15	2022-09-09 15:21:17	siea1	0	13
2022-09-09 15:22:57	2022-09-09 15:28:44	siea1	0	14
2022-09-09 15:28:51	2022-09-09 15:28:52	siea1	1	15

Figure 9: SIEA_BUSY_INDICATION Table

2.2. The Vulnerability Detector Component

For the FogProtect project, we have deployed Wazuh (https://documentation.wazuh.com/current/) as a Vulnerability Detector (see *Figure 10*). In this figure, we have added some details which depict the components mapped to the Vulnerability Detector. In each fog node, a Wazuh Agent is configured to scan its host in a specific manner. In Smart Cities (UC1), each agent can detect the manipulation of video footage in the Smart Lampposts. In Smart Manufacturing (UC2), one agent can detect confidentiality and integrity attacks on sensitive data.

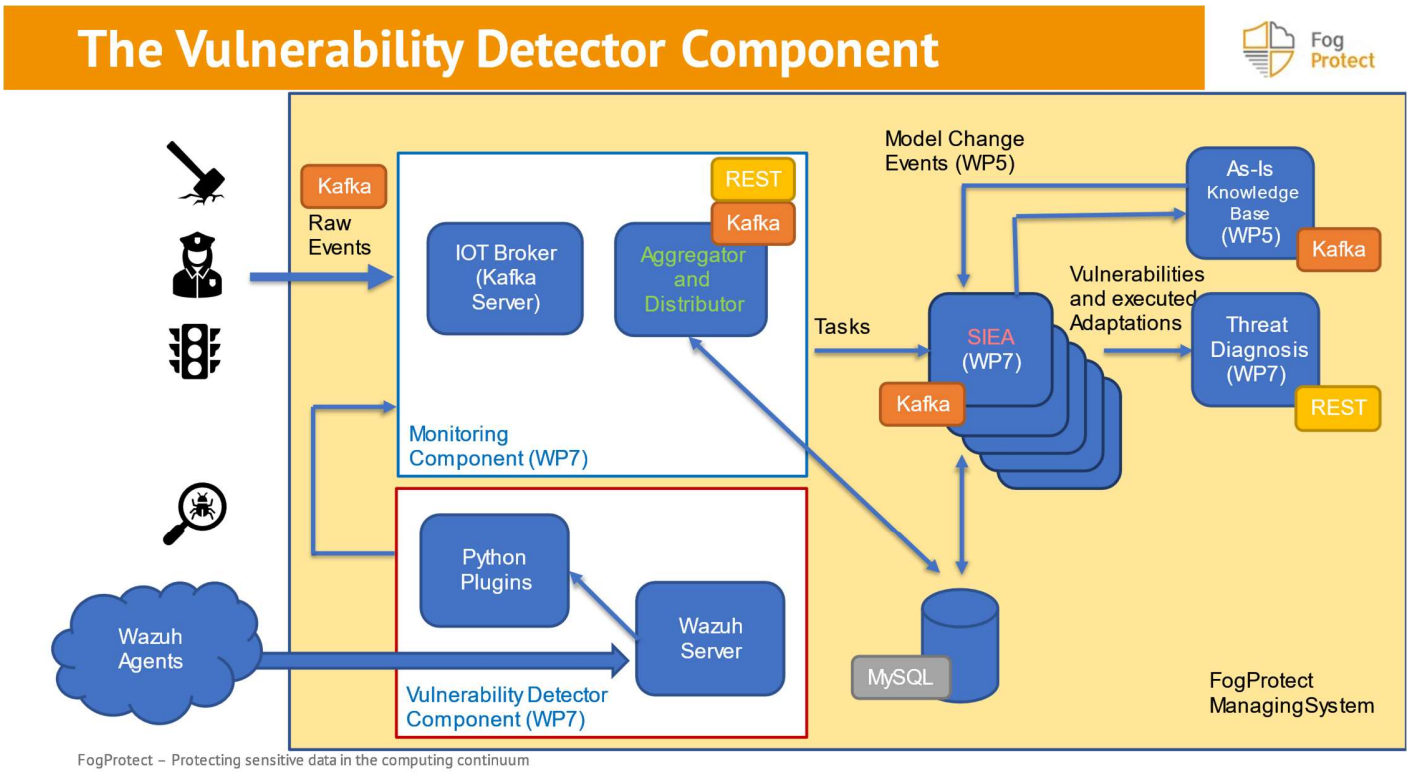


Figure 10: The Vulnerability Detector Component

The Wazuh Agents connect to the Wazuh Server via a proprietary protocol. The 'Vulnerability Scanner' functionality depicted in the *Figure 10* is therefore distributed between the fog and the cloud. Wazuh offers an 'integration' interface for python plugins. The python plugins will be called by the Wazuh Server if one or more configured rules have been triggered. In UC1 and in UC2 we check for rules 550 and 554 mapped to a system check of selected directories (see *Figure 11*). Rule 550 (file changed) is considered by Wazuh as more severe that rule 554 (file added), as it has a higher level.

```
226
227   <rule id="550" level="7">
228     <category>ossec</category>
229     <decoded_as>syscheck_integrity_changed</decoded_as>
230     <description>Integrity checksum changed.</description>
231     <mitre>
232       <id>T1565.001</id>
233     </mitre>
234     <group>syscheck,syscheck_entry_modified,syscheck_file,pci_dss_11.5,gpg13_4.11,gdpr_II_5.1.f,hipaa_164.312.c
235   </rule>
236
237   <rule id="553" level="7"> ...
246   </rule>
247
248   <rule id="554" level="5">
249     <category>ossec</category>
250     <decoded_as>syscheck_new_entry</decoded_as>
251     <description>File added to the system.</description>
252     <group>syscheck,syscheck_entry_added,syscheck_file,pci_dss_11.5,gpg13_4.11,gdpr_II_5.1.f,hipaa_164.312.c.1,
253   </rule>
254
```

Figure 11: Excerpt of Wazuh file 0015-ossec_rules.xml

The Wazuh Server is configured with a web-hook to which it can reporting findings detected by the python plugins. See *Figure 12* in which the plugin 'custom-uc2-checker' is mapped to the web-hook 'relay_event'.

```
381
382   <integration>
383     <name>custom-uc2-checker</name>
384     <hook_url>http://0.0.0.0:5000/relay_event</hook_url>
385     <level>4</level>
386     <rule_id>550,554</rule_id>
387     <group>ossec</group>
388     <alert_format>json</alert_format>
389   </integration>
390
```

Figure 12: Web-hook for UC2

Since we need to translate the active scanning event sent to the web-hook into a Kafka message, we have added some minor functionality to the Distributor, i.e.: the CherryPy server. So, in effect, the http events triggered by the python plugins will be received by the 'relay_event' endpoint of the Distributor which then translates the events into Kafka messages targeted for the Aggregator.

Figure 13 shows an example of the content of a 'ConfidentialityAlert' message recorded by the plugin 'custom-uc2-checker.py'.

```
Tue Oct 11 16:06:28 CEST 2022 /tmp/custom-uc2-checker-1665497186-329695922.alert  http://0.0.0.0:5000/relay_event  > /dev/null 2>&1
Tue Oct 11 16:06:28 CEST 2022: # Starting
Tue Oct 11 16:06:28 CEST 2022: # Webhook
Tue Oct 11 16:06:28 CEST 2022: http://0.0.0.0:5000/relay_event
Tue Oct 11 16:06:28 CEST 2022: # args
Tue Oct 11 16:06:28 CEST 2022: /var/ossec/integrations/custom-uc2-checker.py
Tue Oct 11 16:06:28 CEST 2022: /tmp/custom-uc2-checker-1665497186-329695922.alert
Tue Oct 11 16:06:28 CEST 2022:
Tue Oct 11 16:06:28 CEST 2022: http://0.0.0.0:5000/relay_event
Tue Oct 11 16:06:28 CEST 2022:  > /dev/null 2>&1
Tue Oct 11 16:06:28 CEST 2022: # File location
Tue Oct 11 16:06:28 CEST 2022: /tmp/custom-uc2-checker-1665497186-329695922.alert
Tue Oct 11 16:06:28 CEST 2022: # Processing alert
Tue Oct 11 16:06:28 CEST 2022: {'timestamp': '2022-10-11T16:06:26.780+0200', 'rule': {'level': 7, 'description': 'Integrity checksum changed.', ...
Tue Oct 11 16:06:28 CEST 2022: # Generating message
Tue Oct 11 16:06:28 CEST 2022: {"topic": "ConfidentialityAlert", "attachments": {"reason": "Integrity checksum changed.", "id": "550",...
Tue Oct 11 16:06:28 CEST 2022: # Sending message
Tue Oct 11 16:06:28 CEST 2022: <Response [200]>
Tue Oct 11 16:06:28 CEST 2022: # Sent message
```

Figure 13: Excerpt of integartions.log

2.3. The SIEA

The SIEA will detect changes in vulnerabilities and threats on the cloud/fog/edge system under evaluation (i.e.: the managed system) and report these changes via RESTful POST messages to the 'Threat Diagnosis'.

To perform this task, the SIEA will subscribe to the topics 'AdaptationChange4SIEA' and 'ModelChange4SIEA'. The former is generated by the Distributor and the Adaptation Coordinator. The latter is generated by the As-Is Knowledge Base.

Next to this, the SIEA will also subscribe to the topic ResetEvent which is mainly meant to facilitate debugging.

New for the SIEA implementation in P2 is that:

- it connects to a MySQL database used for synchronizing its tasks with the Distributor and for visualization of the status of the SIEA
- multiple instances of the SIEA can be instantiated
- the SIEA now only has 4 transient states next to a single 'steady' state (i.e.: the idle/stable state)

Next to a Kafka interface for receiving messages from the Distributor, the Adaptation Coordinator, and the As-Is Knowledge Base, the SIEA is provided with a RESTful client interface for connecting to the Threat Diagnosis. On top, the SIEA can connect to a MySQL database to access the SIEA_BUSY_INDICATION and SIEA_TASK_STATUS tables mentioned previously.

High priority tasks received from the Distributor will be pushed to the 'notify-immediate-action-event' endpoint of the Threat Diagnosis. Low and medium tasks will be forwarded to the 'evaluate-event-risks' endpoint. When the SIEA receives a Kafka message with topic ResetEvent, it will send to a message to the newly implemented 'reset' endpoint of the Threat Diagnosis. For the sake of being aware of the processing status of the Threat Diagnosis, the SIEA can poll the 'jobs' endpoint by offering the jobid it is interested in. Upon reception of 'FINISHED', the SIEA can transit into the next state. In case the SIEA receives a 'FAILED', the SIEA will report this issue, break further processing, and return to the 'steady' state.

2.3.1. The SIEA State Machine

SIEA State Machine is now agnostic to the state of the managed system and is implemented with only 5 states. The difference to P1 is, that these 5 states are independent to reported events from the Monitoring Component and to performed adaptations. We have also introduced a 'SieaTaskId' as a session indicator which will be defined by the Monitoring Component. This session indicator will be stored in the SIEA instance which has taken over the processing this task and will be forwarded in every message it transmits. The progress of the task is reflected by the state the SIEA itself is in (compare *Figure 8* and *Figure 14*). With the SieaTaskId, the SIEA can ignore all message responses circulating in the FogProtect system which do not map to its current task.

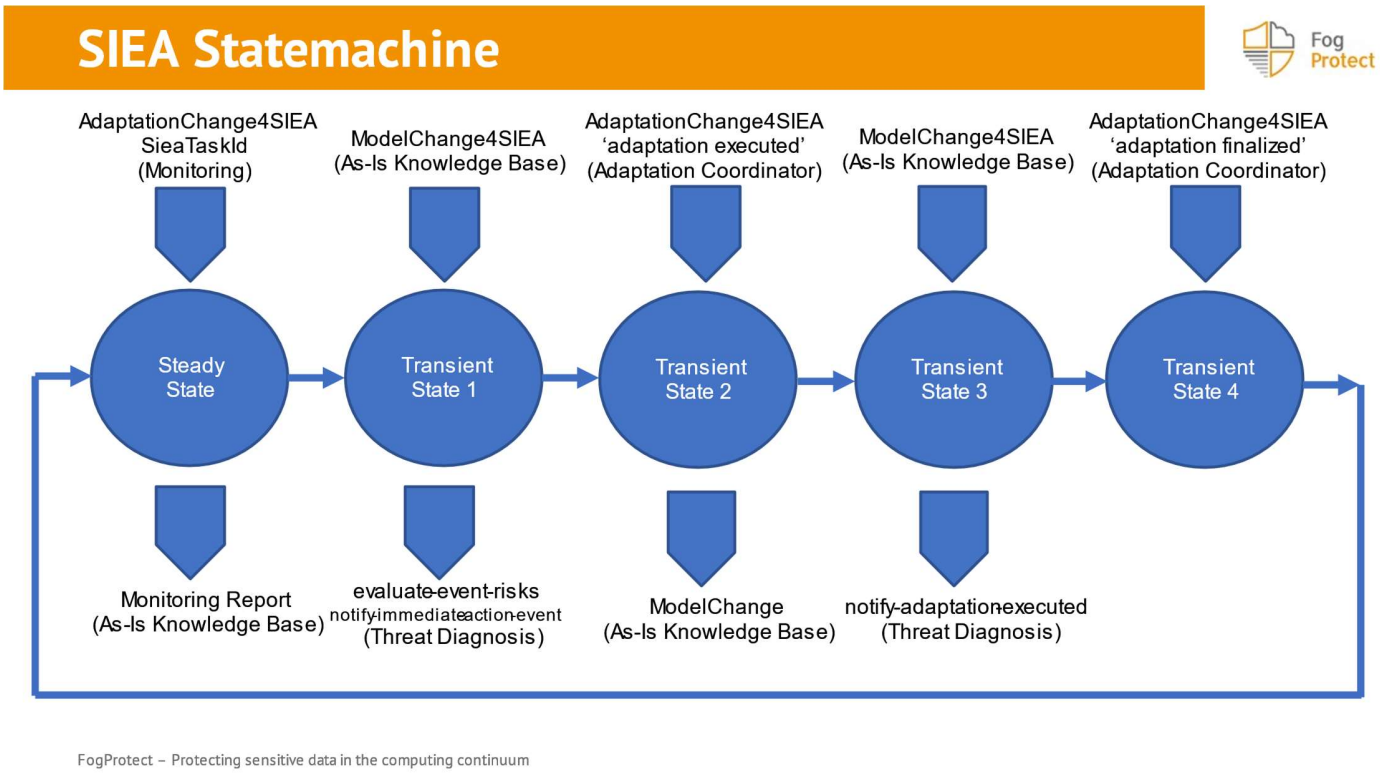


Figure 14: SIEA Statemachine

In the following description, we will map the 5 SIEA states to the sequence diagram depicted in *Figure 15*. This sequence diagram shows the 'happy path' message flow in case of a detected data leakage. Please note that the upstream messages after WP5 have been truncated to simplify this figure.

Upon reception of an AdaptationChange4SIEA sent by the Monitoring Component (Message #2), the SIEA will store the SieaTaskId '001' and use it to verify if following messages are relevant for this task. On top, it will transit from the steadyState to the transientState1 after sending a Kafka message with topic 'DataLeakDetected' to the Knowledge Base (Message #3). Upon reception of a 'ModelChange4SIEA' from the Knowledge Base (#4), the SIEA will send a RESTful POST message to the 'evaluate-event-risks' endpoint of the Threat Diagnosis (#5, see also *Figure 16*). If the Threat Diagnosis reports that the jobid has 'FINISHED' processing (which means that the handshake between WP5 and WP7 came to an end), the SIEA will transit into the transientState2. Upon reception of an AdaptationChange4SIEA sent by the Adaptation Coordinator (#6), the SIEA will query the Knowledge Base for executed adaptations (#7) and enter transientState3. When now the SIEA receives an ModelChange4SIEA (#8), it will pass the entire content to the Threat Diagnosis using its endpoint 'notify-adaptation-executed' (#9, see also Figure 17). After receiving the 'FINISHED' indication from Threat Diagnosis for this jobid, the SIEA will enter the final transientState4. If now WP5 sends an AdaptationChange4SIEA with content 'finalized' (#10), the SieaTaskId '001' has been completely and successfully processed. With this message, the SIEA will be idle and go back to the steadyState again.

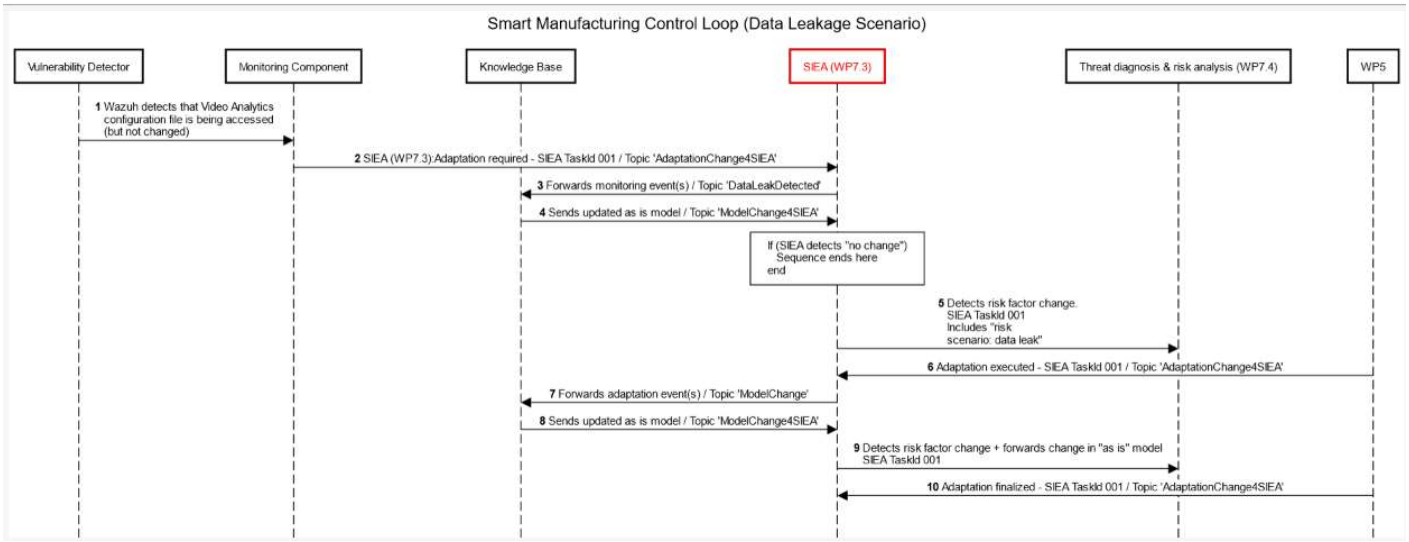


Figure 15: Communication Partners of the SIEA with an Excerpt of a Message Flow for Smart Manufacturing

Example message formats are shown below as illustrations.


```
{
  "SieaTaskId": "1-33aa0b70-3dad-4c5c-969a-2907c27ce45e",
  "ChangesMadeToAsIsModel": [
    {
      "ObjectToIdentify": {
        "name": "Attacker_ReadDataFlow",
        "type": "fogprotect.adaptation.ComputingContinuumModel.ReadDataFlow",
        "atid": "//@tosca_nodes_root.53"
      },
      "Changes": [
        {
          "ChangeType": "CHANGE",
          "AttributeChanged": "disab",
          "AttributeType": "EBoolean",
          "AttributeOldValue": "true",
          "AttributeNewValue": "false"
        }
      ]
    }
  ],
  "EventName": "DataLeakDetected",
  "Timestamp": 1658914511590,
  "Vulnerabilities": [
    {
      "rule": "550",
      "agent": {
        "id": "001",
        "name": "green-ibis",
        "ip": "192.168.1.250"
      },
      "reason": "Sensitive data could have been read by an unauthorized person",
      "filename": "app_config.py",
      "source": "wazuh",
      "serviceLevel": "medium"
    }
  ]
}
```


Figure 16: Evaluate Risks Message

```
{
  "SieaTaskId": "1-33aa0b70-3dad-4c5c-969a-2907c27ce45e",
  "AsIs": {
    "AsIsModel": {
      "type": "CloudEnvironment",
      "@id": "/",
      "tosca_nodes_root": [ ...
      ]
    },
    "OldAsIsModel": { ...
    },
    "ChangesMadeToAsIsModel": [
      {
        "ObjectToIdentify": {
          "name": "ReadDataFlow (User-NormalDB)",
          "type": "fogprotect.adaptation.ComputingContinuumModel.ReadDataFlow",
          "atid": "//@tosca_nodes_root.60"
        },
        "Changes": [
          {
            "ChangeType": "CHANGE",
            "AttributeChanged": "disab",
            "AttributeType": "EBoolean",
            "AttributeOldValue": "false",
            "AttributeNewValue": "true"
          }
        ]
      }
    ],
    "Risks": []
  },
  "Adaptations": []
}
```


Figure 17: Adaptation Executed Message

2.4. Summary

In Period 2, the upstream architecture to the Threat Diagnosis has changed significantly taking into account the hands-in experiences from 2021. The need for aggregation and filtering of 'raw' events became evident, not only to stop flooding the Managing Components of FogProtect, but also to differentiate between temporal and perhaps accidental behaviour and deliberate misbehaviour. With the new Aggregator Component, we have showcased this functionality in Smart Media by being able to enforce service level agreements (SLAs) and by being able to differentiate between users accidentally querying protected data and users deliberately and repeatedly trying to do so.

In P2 we have also added a Vulnerability Detector Component to the system which enabled us to detect and report, in real-time, security-relevant events on hosts located in the cloud. The Vulnerability Detector is deployed in the Smart Cities and in the Smart Manufacturing use-cases.

The next new component in P2 is the Monitoring Component which is configured to feed the SIEA with tasks based on their service level. If more than one task must be reported, the highest priority task will be pushed to the SIEA and the lesser priority task will be cached and sent later.

Additionally, the SIEA has been simplified and made agnostic to the state of the Managing System.

The final major change in P2 was to introduce a database accessed by both the Monitoring Component and the SIEA which is being used for querying the state of one (and potentially more) SIEA instance. The Monitoring Component can therefore control better the flow of tasks towards the SIEA making the P1 component, the 'Kafka Latch', obsolete.

3. Risk Modelling for Populations

Within systems spanning the cloud and fog it is now very common that there are many fog nodes present compared to other elements within the system, For example, there may be many local wireless access points connected to a single data centre, or many smart lamp posts connected to the central management components. Both these examples involve risks to the data that the fog nodes generate, transmit, process and store and so are risks directly affected by the number of fog nodes present. This multiplicity of fog nodes needs to be modelled so that accurate predictions of the risks can be made. As the number of fog nodes can easily span from the tens to the hundreds and even thousands, it is important that computationally efficient modelling of all the fog nodes is achieved as well. Here we present limitations of the SSM regarding population modelling prior to this work, a theoretical derivation of efficient population modelling that overcomes these limitations, the application of this theory into the SSM core model, and a worked example showing the population modelling process within the FogProtect Use Case 1.

3.1. Background

3.1.1. SSM Core Model

The SSM has a core model that defines the fundamental building blocks of SSM that are the core concepts of the asset-based risk modelling approach that SSM uses. Some of these concepts have been previously defined in D7.1, including assets, consequences (alternatively referred to as "misbehaviours" or "adverse effects"[1]), threats, risk levels, trustworthiness, likelihood and controls. In addition to this, other core concepts include the system and risk. These are defined as follows.

- *System*: The set of applications, services, information, technology assets or other information handling components [ISO 27000].
- *Risk*: 'Effect of uncertainty on objective' [ISO 27000].

How each of these core model concepts relates to each other is shown in Figure 18.

[1] Different terms are used to refer to the same concept. This has resulted from different communities using different terms and is in the process of consolidation to determine a consistent single term that reflects the concept of the undesirable effects of a threat.

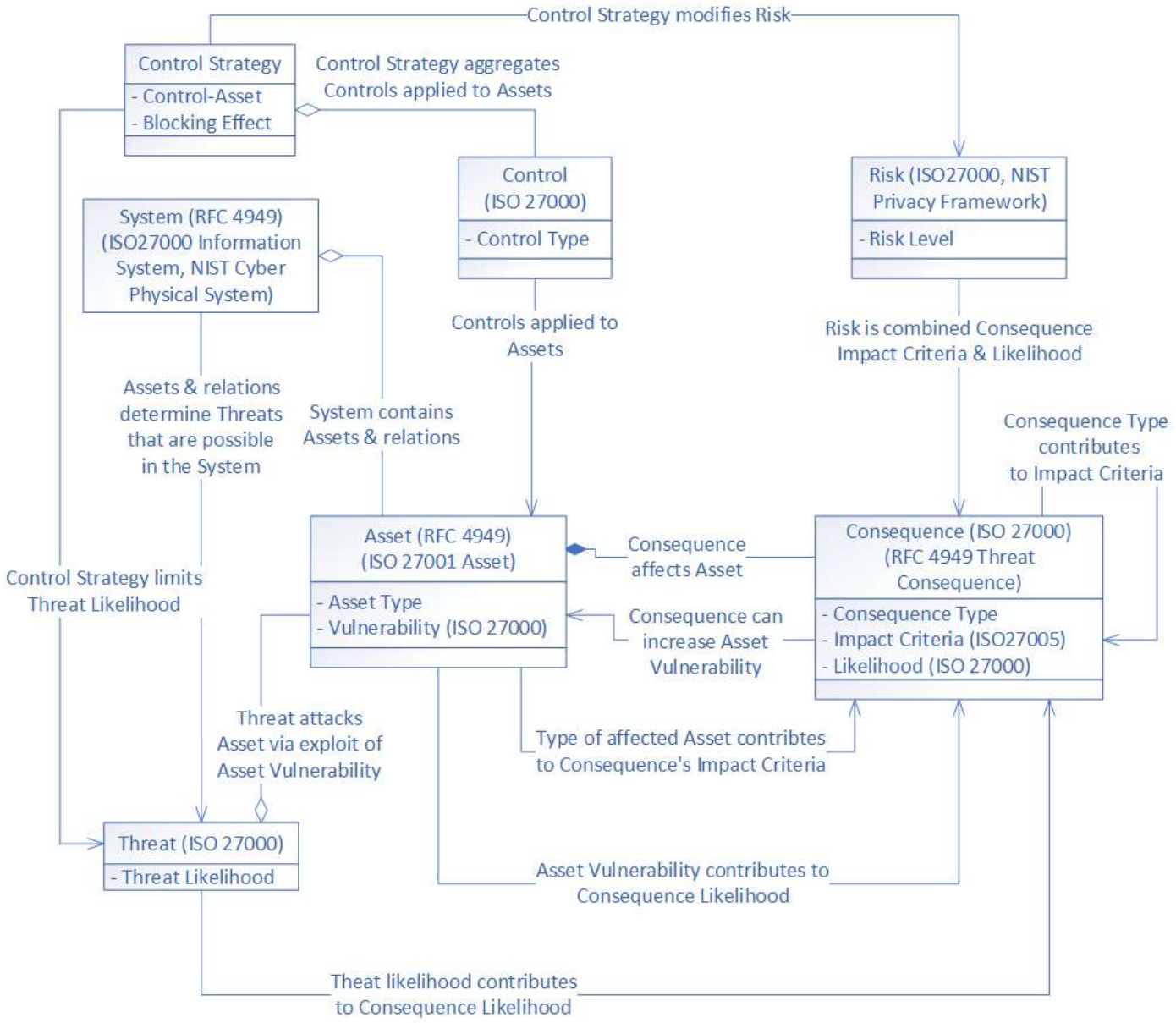


Figure 18: SSM core model, from Ref. [Boniface 2022].

In Figure 18, the System is a cyber-physical system consisting of different Asset types and relations. Assets can have Vulnerabilities, which describe weaknesses. Each threat requires specific assets and relations to be present for it to be triggered, described by the Threat's 'matching pattern'. Within a matching pattern there are nodes and links. A node in a matching pattern has a role in the Threat and restrictions on the type of Assets that can match it. A link in the matching pattern is a connection between two nodes, with a restriction on the type of Asset relationship that can match it. A pattern match is a collection of SSM system model assets with relationships that meet all the constraints specified in the pattern.

Threats attack Assets via asset Vulnerabilities. Each threat has a likelihood of it occurring that is based on factors such as how easy it is for that threat to be carried out and the motivations of actors attacking in this way. Assets also have Consequences (also known as "misbehaviours" or "effects"), which are an unwanted change in behaviour or state, and these have a likelihood to them occurring based on the likelihood of the Threats resulting in them. Each Consequence also

has an Impact Criteria that is a measure of the severity of the Consequence on the affected Asset. A specific Consequence on an Asset has a Risk Level that is a function of the Impact Criteria and the Likelihood. Controls are defensive measures applied to assets. Control Strategies are groups of controls that limit Threat Likelihoods and so block Threats.

3.1.2. Scale for Risk, Impact, Likelihood and Trustworthiness

The SSM implements ISO 27005, a well-adopted risk assessment methodology for cybersecurity, and uses algorithms to calculate the likelihood of security threats and their effects on assets. Risk levels are determined by combining the likelihood of threats and impact of their effects on assets. The risk level from a threat depends on the likelihood and impact of the threat effects (including where appropriate, secondary effects and other consequences such as increased opportunity for further attacks). Evidently, if risk levels are to be meaningful, the definitions for the risk, likelihood and impact scales must be consistent with each other.

Likelihood is a measure of the chance of something happening. Within SSM different likelihood values (or levels) form the likelihood scale, which is also related to the trustworthiness scale. A natural way to express likelihood, impact and risk in an ISO 27005 risk assessment is by using fuzzy scales composed of linguistic terms as commonly information on likelihoods, impacts and risks are given in linguistic terms and so inputs to risk calculations (e.g., the initial assumed trustworthiness of system assets) can in principle be extracted from documents describing a system and its context, or other sources, and used in these terms. The fuzzy logic approach of the SSM finds threat likelihood (and thence risk) using an automated rules-based procedure, given a suitable encoding of threats including their causes and effects.

The likelihood and trustworthiness scales and levels are defined linguistically in Table 1. These can be thought of as two opposite views of the same scale. Prior to the work presented here the likelihood and trustworthiness scales and levels were defined solely in the linguistic definitions given, though within the work presented here they are extended to include mathematical definitions.

Likelihood	Means	Trustworthiness
Very high	Something will definitely go wrong if the possibility exists even only for a short time.	Very low
High	Something is likely to go wrong if the possibility exists even only for a short time. Something will definitely go wrong if the possibility persists.	Low
Medium	It is unlikely that anything will go wrong if the possibility exists only for a short time. Something is likely to go wrong if the possibility persists.	Medium
Low	It is unlikely that anything will go wrong if the possibility exists only for a short time. Something is likely to go wrong if the possibility persists for a long time.	High
Very low	It is unlikely that anything will go wrong even if the possibility persists for a long time.	Very high
Negligible	Nothing will go wrong.	Safe

Table 1: Likelihood and trustworthiness linguistic definitions.

3.1.3. Threat Likelihood Calculation

To find the likelihood of a threat, one must determine the likelihood of each cause, which will be associated with an involved asset. Threats are classed as ‘primary’ or ‘secondary’, as discussed in D7.1. For primary threats, each cause is an asset Trustworthiness Attribute (TWA) modelling the propensity of the asset to act in such a way as to cause the threat The likelihood of this is determined from the trustworthiness level of that attribute. For secondary threats, which model the propagation of effects due to system dependencies, each cause to them at an involved asset is an effect caused by

other threats and so has a likelihood dependent on those other threats. This is demonstrated in Figure 19 and Figure 20, where the schematics of an example primary threat and secondary threat are given, showing how asset TWAs give the primary threat likelihood and how upstream threat effects give the secondary threat likelihood.

Figure 19 shows entry point threat causes, MS 1 and MS 2, setting asset trustworthiness attributes, TWAS 1 and TWAS 2, and then giving the threat likelihood, as well as the likelihood of the threat effects, MS X, MS Y and MS Z, as the minimum of those threat cause likelihoods. Here the minimum of the threat cause likelihoods gives the threat and threat effect likelihoods. This is because all the causes must be present for a threat to arise and so the overall threat causation likelihood is found using the fuzzy logic AND operator to be the minimum likelihood among its causes, and this then gives the threat effect likelihood too.

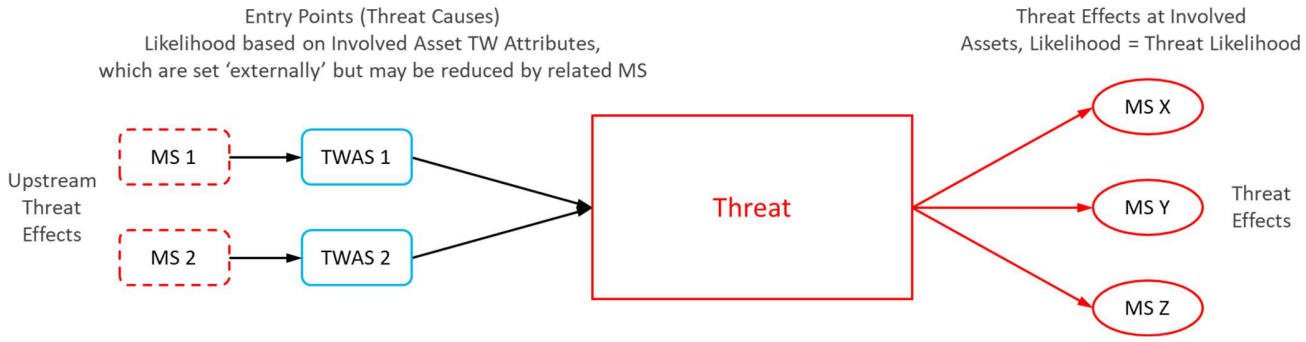


Figure 19: Schematic of an example primary threat with entry point threat causes and resulting threat effects.

Figure 20 shows the upstream threat effects, MS 1 and MS 2, giving the threat likelihood, as well as the likelihood of the threat effects, MS X, MS Y and MS Z, as the minimum of those upstream threat cause likelihoods. Here the minimum of the threat cause likelihoods again gives the threat and threat effect likelihoods as all causes need to occur for the threat to happen.

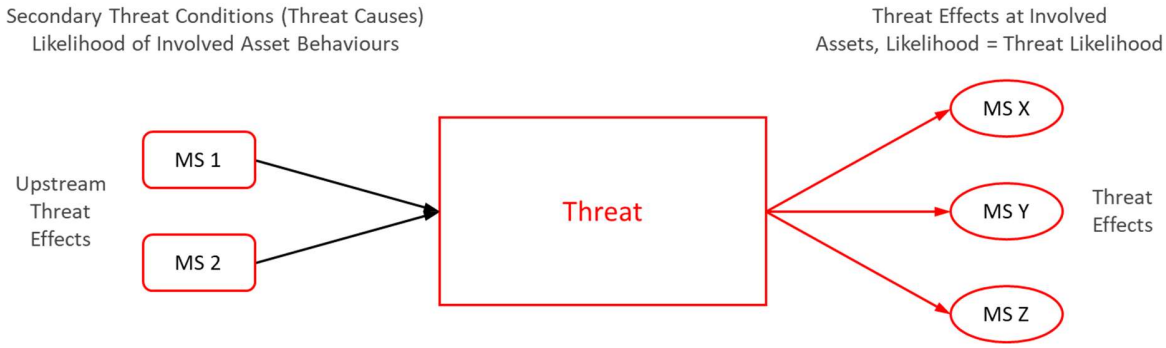


Figure 20: Schematic of an example secondary threat with upstream threat effect threat causes and resulting threat effects.

3.2. Limitations of Previous Approach

One limitation of the approach within SSM prior to this work is that each cause is associated with a single real-world asset instance which has a specific role or involvement in the threat. The contribution from that cause reflects how likely it is that the asset fulfilling that role exhibits an untrustworthy characteristic or threat induced behaviour. The likelihood of a

threat then depends on the least likely of its causes, plus further reductions achieved by any controls (security measures) that may be in place.

One can define threats in which multiple asset instances fulfil the same role in the threat at the same time. The previous algorithm of SSM then considered that each instance must represent a separate cause, where the least likely asset to have the untrustworthy characteristic or behaviour determined the contribution to threat likelihood arising from assets in that role. This works well for the explicit modelling of secondary threats involving a dependency on a set of redundant assets, as with redundant assets, the likelihood of a threat occurring in this circumstance depends on the least likely cause since all redundant assets need to be compromised for the downstream threat to occur. For example, following FogProtect Use Case 1, to model a collection of smart lamp post cameras all recording the same scene and transmitting the recordings to an aggregated data asset in the cloud, there would be a system model that includes all the lamp posts as well as the cloud-based aggregated data. If every lamp post was broken in to and the availability of each lamp post video data was lost then this would result in a secondary threat of the loss of availability in the aggregated data. However, with the lamp posts all recording the same scene, every lamp post needs to be experiencing that loss for it to occur in the aggregated data. In the system model each lamp post video data asset would match the threat cause role in the threat pattern and so the least likely lamp post video data to experience the loss of availability would dictate the loss of availability in the aggregated data. However, this approach has limitations, summarised as follows:

1. While it is easy to model secondary threats involving dependence on redundant assets, it is impossible to model primary threats involving exposure to multiple (untrustworthy) assets. This is since the multiple system assets will all match to the relevant node of a secondary threat pattern, but as primary threats are caused by single entry point TWAs each would match to a separate threat instead of all being included in the same matched threat pattern.

2. It is difficult to model likelihood and risk for threats involving large numbers of assets, since each asset would need to be represented separately on the system canvas, severely limiting efficiency, scalability and understandability.

Here we present extensions to the model and algorithms of SSM, whereby these limitations are overcome in an efficient manner. To do so, it is necessary to introduce some assumptions and definitions on how to interpret the results when applied in an ISO 27005 risk assessment.

3.3. Assumptions, Definitions and Interpretation

3.3.1. Calculations of Risks Involving Asset Populations

Our approach to address these limitations considers that each system model asset has the possibility to represent a **population** of real-world instances. For example, a single ‘lamp post’ asset on the system model canvas represents a collection of real-world smart lamp posts.

To efficiently model populations and their resulting risks, three scenarios (or cases) are given that each describe important behaviours of asset populations and threats. These are defined later mathematically, though are premised on sampling members of a population acting as a given distribution, and so allow for efficient modelling of the population without needing to explicitly model every individual member. These three scenarios of the population are the lowest, highest and average likelihood over an asset population. The choice of each case depends on the threat under consideration.

- The **lowest likelihood** case corresponds to cases such as redundancy, where every member of an asset population needs to be experiencing a threat for that threat to be considered present in the population as a whole and so the threat likelihood is determined by the lowest likelihood member of the population.

- The **highest likelihood** case corresponds to cases when just a single member of an asset population needs to be experiencing a threat for the threat to be considered present in the population as a whole and so the threat likelihood is determined by the highest likelihood member of the population. This is applicable in threats affecting properties such as confidentiality, where any leak of data will be sufficient to cause (adverse) consequences.

- The **average likelihood** case corresponds to the average (or randomly chosen) likelihood member of a population. This is the default likelihood, and is used when there are no reasons to use lowest or highest likelihood. Average is also used when a matched threat cause is not a population but a single asset.

To handle populations of assets we need the following extensions within SSM and the surrounding theory. These explanations are summarised here and explained in detail later in this section.

- Allow systems to include 'non-singleton' asset classes representing multiple assets (or in general, a population of assets) all fulfilling the same system role;

- Extend the formalism for modelling threats so it is possible to define threats with causes based on the lowest, average or highest likelihood over an asset population;

- Extend the formalism also for threat effects, so it is possible to specify if threat effects are distributed between an asset population, or applied equally to all its members;

- Define a fuzzy scale to capture the population size for such a non-singleton asset class.

For these extensions to occur, more precise definitions are required for the scales of likelihood, impact and risk than those given above are required, and this is discussed in the next section.

3.3.2. Refinement of Likelihood, Impact and Risk Level Scales

In this section the previous definitions for the likelihood, impact and risk scales are extended so that both precise and mathematical forms can be given to them.

In the approach of SSM, the likelihood scales are defined in a knowledge base that also contains models of system asset types and potential threats. These should be interpreted as a likelihood over the lifetime of the system being modelled and analysed. At present, the knowledge base used most often describes cyber-physical system assets and their threats and uses a likelihood scale ranging from **Negligible** (so unlikely that a threat or effect with this likelihood can be ignored), through **Very Low**, **Low**, **Medium**, **High** to **Very High** (practically inevitable). For a typical ICT system this lifetime might be 1000 days before a substantial redesign and/or overhaul of its components.

All risk assessment involves predicting the future. One challenge is that systems evolve during their lifetime even while the design and composition remains unchanged. For example, software-based assets will contain software vulnerabilities, whose discovery by or disclosure to attackers will alter the likelihood of some threats. Operational procedures like regular software patching may be used to reduce this effect, but nevertheless a system whose software seems secure will become less so over time. The SSM risk calculation is based on a 24-hour period during which the system does not evolve in

significant ways. The SSM software supports two options based on how far into the future the 24 hour risk calculation period occurs:

- Current risk calculations: **based on the next 24 hours**, given the current trustworthiness status of the system assets (e.g., known vulnerabilities in software assets)

- Future risk calculations: **based on an arbitrary 24-hour period in the future**, including the effect of system changes (e.g., software vulnerability disclosure and software patching).

In this sense, the likelihood of an event (a threat, or a threat effect) is related to the probability and thus frequency with which the event occurs. If the frequency were as high as once per day, we might consider it an absolute certainty that the event will occur. Something that might happen once in the lifetime of a system is still quite likely – it will probability arise at some stage – but obviously it is far less likely to occur right away. If something is 10000 times less frequent than that, it may be something that could be ignored for all practical purposes as the chance it will not happen in 24 hours is 99.99999%. This leads to some assumptions, or rather axioms, concerning the likelihood scale:

- The likelihood scale is concerned with the magnitude of probabilities or frequencies of events occurring and thus each likelihood level represents a range between two different magnitudes of probability or frequency.

- The **Very High** likelihood level corresponds to events that are expected to occur often enough that they should be treated as practically certain. This does not mean they are certain to occur, just that the predicted occurrence is high enough to be indistinguishable from certainty for risk management purposes. For this, we say that a Very High likelihood event should be one that is expected to occur at least once every ten days over the 1000-day lifetime system.

- A **Medium** likelihood event should be one that is expected to occur no less than once over the lifetime of the system, e.g., at least once per 1000 days.

- A **Negligible** likelihood corresponds to events that for all practical purposes can be considered to not occur within the lifetime of the system. We say that a Negligible likelihood event is one that is 100 times less frequent than a Medium likelihood event, and so is one that is expected to occur in no more than once per 100 system lifetimes, e.g., no more than once per 100000 days. It is a special case in that it is essentially an absorbing boundary condition in which a Negligible likelihood cannot be increased to a higher level.

- A likelihood needs to exist that corresponds to events that are expected to occur many times over the system lifetime, but not so much that they are practically certain. For this we define the **High** likelihood event as one that should be expected to occur at least once every 100 days over the 1000-day lifetime of the system.

- Between the Negligible and Medium likelihood levels there should be two likelihoods, **Low** and **Very Low**. A Low likelihood level corresponds to events that are expected to occur at least once per ten system lifetimes, e.g., once per 1000 days, and a Very Low likelihood event is one that is expected to occur at least once per 100 system lifetimes, e.g., at least once per 10000 days.

Impact is usually defined in terms of the effect a threat would have on the ability of the affected system to achieve its (business) purpose. However, this leads naturally to a two-level ‘all or nothing’ classification: the threat either does or does not prevent the system achieving its purpose. This is of limited usefulness, so most practical impact scales take other factors into account, such as:

- How frequent would the threat need to be before the effects become intolerable?
- How widespread would the threat effects need to be before they become intolerable?

A threat that really does cause instant, total and permanent system failure will be intolerable even if it happened only once in the expected lifetime of the system. However, most threat effects prevent the system from functioning only while the effects are maintained. The effects may be temporary or reversible by means of appropriate recovery actions, so even if the system cannot function at all, the downtime may not be enough to prevent it achieving its purpose. Many threats also produce effects only for one or a few system users, so again, if the threat does not occur too often it may be tolerable.

We therefore conclude that the risk level should express whether the possibility of a future threat can be tolerated, given its likelihood and impact. The **likelihood** is related to **how often the threat is expected to occur** (how frequent, and/or to how many users), while the **impact** expresses how often it can be **allowed** to occur.

Given this, a medium risk level should represent a case where the two are roughly the same: the threat is just about tolerable given how often it is expected to occur. Such a risk is not one to be ignored, because the likelihood represents an expectation. If the threat occurs randomly, then at times it will occur more often than expected and exceed the tolerance threshold. Such a threat should be addressed, but it would be OK if that takes some time.

A very low risk level means the threat could be tolerated if it occurred far more often than expected, and hence the risk is acceptable, and no action is needed to address it. A low risk level would cover cases where the expected occurrence is significantly but not far lower than the tolerable level. Such a risk can also be accepted, but probably means the system will be degraded to some extent. Low level risks are therefore worth addressing, but only when there is nothing else that needs doing.

Conversely, where the threat is expected to occur more often than can be tolerated, the risk level would be considered high. Prompt action would certainly be needed to address the threat in such cases. If the expected occurrence is far above a tolerable level, the risk level is very high. Since the risk level represents a prediction on threats to come, for both current (immediate) and future risk (longer term) calculations, this does not mean the threat is already happening and the system is already unable to function. However, it does mean things are expected to get bad very fast, so if the risk level is predicted to be very high, then the system probably should not be operating in its current state.

With these definitions, likelihood can now be related to probability and so, building up on this, the theory describing how populations affect likelihood can be formed.

3.4. Theoretical Derivation of Population Behaviours

The objective is to model multiple instances of assets, threats and their effects in a computationally efficient manner. This can be done by using statistical modelling of asset populations, applied to and extending the previous approach of SSM. For this, it is first useful to consider two examples involving multiple copies of data each on a different host as they represent key behaviours regarding populations within risk modelling are determined.

These examples examine the availability and confidentiality of data respectively. Concerning data availability, if there are multiple identical copies of the data that are equally accessible, then if one copy becomes unavailable, overall the data is still available due to the redundancy of the copies. Therefore, for this example, as the number of instances of the data increases, it becomes less and less likely that there will be a loss of availability to the data. This is an illustration of the least likely of the population membership determining the overall likelihood of the population to be compromised.

Concerning data confidentiality, if there are multiple copies of the same data and the host of any of these instances becomes compromised resulting in an unauthorised user gaining access to it, then the data as a whole needs to be considered as having lost its confidentiality. Therefore, for confidentiality, as the number of instances increases, it becomes more likely that at least one of the data instances will have a loss of confidentiality and thus there will be a loss of confidentiality of the data as a whole too. Here, the likelihood of this occurring is given by the likelihood of the data instance that is most likely to be experiencing the loss.

Putting this in terms of members of a population, the number of instances is the size of the population, and each instance is a member of that population.

This leads to formal definitions of the three cases of threat type concerning populations.

- The **average likelihood** case corresponds to the average sampled member of a population, and is the same as the case where a population has just a single member, known as a singleton.

- The **lowest likelihood** case corresponds to such cases where all members of a population are needed to be experiencing an issue for it to be considered present in the population as a whole, such as in the case of redundancy where if there are multiple servers all providing the same service then all of those servers need to become unavailable for that service to become unavailable too.

- The **highest likelihood** case corresponds to when only one member of a population experiencing an issue is enough for the population as whole to be considered experiencing it, such as a user maliciously leaking sensitive data meaning that all the users of that userbase need to be considered untrustworthy until the issue is resolved.

For each consequence and cause, the members of the population form a distribution of likelihoods. For populations in which the consequence or cause at each member depends on those at the other members the lowest and highest likelihood cases are equal to, and given by, the average likelihood case. For populations in which the consequence or cause at each member is independent of those at the other members, the consequence and cause likelihoods fit the requirements to be described by a binomial distribution and so the lowest and highest likelihood cases are given from the average likelihood case in conjunction with a binomial calculation. Describing how populations behave within risk modelling therefore then becomes a case of using different statistical samplings of a population to determine the three different member likelihood types.

From these statistical samplings of populations, the lowest and highest likelihood cases can have rules determined that are given in a form usable by the SSM. For this, it is firstly mathematically considered how the probability of something occurring for the average case can be used to determine the probability of it occurring for the lowest and highest likelihood cases. Then the definitions for likelihood are used to deduce how different population sizes affect the likelihoods of the lowest and highest cases, and finally from this and the introduction of a fuzzy population scale the fuzzy SSM rules for populations are determined.

3.4.1. The Binomial Population Distribution

One of the most common distributions is the binomial distribution, in which the number of successes sampled from a population is modelled, with replacement in the sampling, and it is assumed that the draws are independent.

The probability mass function, which gives the probability of a specific number of successes in a given sample, is given by

$$f(k,n,p) = Pr(X = k,n,p) = \binom{n}{k} p^k (1-p)^{n-k}$$

(3.1)

for $k = 0, 1, 2, \ldots, n$, where

$$\binom{n}{k} = \frac{\mathrm{n!}}{\mathrm{k!\,(n-k)!}}$$

(3.2)

and k is the number of successes drawn from the sample of size n, p is the probability of a success, X is a binomially distributed random variable, $\binom{n}{k}$ is known as a binomial coefficient, and the operator denoted by '!' is the factorial operator.

Within the context of populations, the probability mass function would give the probability that a particular number of members of a population have a particular issue. This can be used to determine the probability for the lowest likelihood case though is not able to be used to determine the probability for the highest likelihood case and so the cumulative distribution function is needed instead. This is since the probability mass function gives the probability of a specific number of successes, which the lowest likelihood case can be described by, but the highest likelihood case is described by a collection of different numbers of successes, which the probability mass function cannot describe but the cumulative distribution can.

The cumulative distribution function gives the probability that X will take a value less than or equal to k, and for the binomial distribution is given by

$$F(k;n,p) = Pr(X \leq k;n,p) = \sum_{i=0}^{k} \binom{n}{i} p^i (1-p)^{n-i}$$

(3.3)

where Σ is the summation operator.

This is more useful than the probability mass function when determining the population behaviours for the highest likelihood case as it gives the probability that there are up to k members of a population with an issue, and this in turn can be reworked to give the probability of a population having at least one member that has an issue instead, matching the requirement for the highest likelihood case. Additionally, the cumulative distribution function can be easily used to determine the probability for the lowest likelihood case too.

3.4.2. Lowest and Highest Likelihood Cases

The probabilities of the lowest and highest likelihood cases of a population are determined from the average member likelihood case, given the assumptions for the binomial distribution, and where successes sampled from a population correspond to the sampled member exhibiting the behaviour under consideration.

For the highest likelihood case, any sampled member having that behaviour results in the behaviour being present in the population as a whole. In terms of successes being sampled from a population, this corresponds to the number of

successes drawn from the sample being greater than zero, that is $X > k$, where $k = 0$. The probability for $X > k$ is related to that for $X \leq k$ by the relation

$$Pr(X > k) = 1 - Pr(X =< k)$$

(3.4)

Using this with the cumulative distribution function of Eq. (3.3) and setting $k = 0$ gives

$$Pr(X > 0) = 1 - Pr(X \leq 0) = 1 - \binom{n}{0} p^0 (1-p)^n$$

(3.5)

which results in the probability of the highest likelihood case being

$$Pr(X > 0) = 1 - (1-p)^n,$$

(3.6)

or rather,

$$P_{Highest} = 1 - (1 - P_{Av})^n$$

(3.7)

where $P_{Highest}$ is the highest case probability, P_{Av} is the average member probability, and n being the size of the population.

For the lowest likelihood case, the whole population needs to be exhibiting the behaviour before the population as a whole is considered to have it. In terms of successes being sampled from a population, this corresponds to the number of successes drawn from the sample being equal to the size of the population itself. That is $X = k = n$, giving

$$Pr(X = n) = \binom{n}{n} p^n (1-p)^{n-n} = p^n,$$

(3.8)

or rather,

$$P_{Lowest} = (P_{Av})^n$$

(3.9)

where P_{Lowest} is the lowest case probability.

With equations (3.7) and (3.9) relating the highest and lowest cases to the average case and the population size, we now examine how they may be used to classify likelihood into levels via fuzzy logic.

3.4.3. Examining Population Size Effects on Probability and Likelihood

With the lowest case and highest case probabilities of Eqs. (3.7) and (3.9) now determined, they can be used to examine how these probabilities change from the average case when different population sizes are applied, and from this how

classification (or rather, fuzzification) of probabilities into the levels of the likelihood scale is also affected by different population sizes, for these highest and lowest likelihood cases. This in turn is used to determine how the probabilities for all three lowest, highest, and average cases fuzzify into SSM likelihoods, and how these three different cases behave within the fuzzy logic and set theory of the SSM risk calculation and propagation.

Here fuzzification is the process of going from a crisp precise number and transforming it into a level of a fuzzy scale (or more precisely, classifying that number in terms of how much it fits in each available fuzzy term or level). Defuzzification of the opposite process of this, in which the memberships of levels are evaluated to give a single crisp precise number as the output. To achieve this, fuzzy membership functions are used that say to what degree each input number belongs in each fuzzy term or level. Fuzzification is used to determine which likelihood levels different probabilities would fall into, or which population level a population size would fall into, and defuzzification is used to determine what each likelihood level, or also population level, would become when transformed back to a crisp number.

SSM is able to work purely in terms of fuzzy levels throughout as its calculations are able to take fuzzy inputs and give fuzzy outputs, without needing any inputs or outputs to be in crisp number terms, and so fuzzification and defuzzification are primarily used here for determining how the probabilities of the highest and lowest cases lead to corresponding likelihoods, and also how population sizes lead to population levels.

Before examining how population size affects the probability and likelihood, the fuzzy membership functions of SSM need to be defined so that probabilities can be transformed into likelihoods, and an appropriate defuzzification procedure also needs to be defined so that likelihoods can also be transformed back into probabilities.

Section 3.3.2 highlights that within risk modelling, the **magnitudes** of probabilities are typically considered rather than their precise values, due to this information typically being fuzzy and imprecise (i.e. typically actual numbers are not available, but humans are able to make judgements about the tolerability of an effect). This focus on magnitudes lends risk modelling to work within the logarithmic domain for probability, and therefore this is the domain that SSM works within. The decade-based likelihood levels defined in Section 3.3.2 form the basis of the membership functions, being logarithmic and disjoint (i.e. there is no overlap in set membership). Disjoint top hat membership functions are used so that each likelihood variable within SSM can take any value along the likelihood scale formed of the likelihood levels, though at any given time there is only one of these levels along the scale that has a non-zero degree of membership, and this value is unity, indicating total membership of one level and no membership of any other level.

Due to working within the logarithmic domain a logarithmic centre of maximum (*l-C.o.M.*) is used for the defuzzification procedure of a likelihood variable. The result of this defuzzification procedure is a probability that is logarithmically in the centre of the membership function corresponding to the one likelihood level with a non-zero degree of membership, out of the different levels the likelihood variable can take as a value.

Using this information and the decade scales defined in 3.3.2, Table 2 gives the probability range corresponding to each likelihood level, as well as the defuzzified logarithmic centre of maximum for each level. Here the Negligible likelihood level does not have a logarithmic centre of maximum value due to it not having a finite range within the logarithmic domain.

Likelihood Level	Probability Range	Logarithmic Centre of Maximum
Very High	$10^{-1} - 10^{0}$	3.16×10^{-1}

High	$10^{-2} - 10^{1}$	3.16×10^{-2}
Medium	$10^{-3} - 10^{2}$	3.16×10^{-3}
Low	$10^{-4} - 10^{3}$	3.16×10^{-4}
Very Low	$10^{-5} - 10^{4}$	3.16×10^{-5}
Negligible	$< 10^{-5}$	-

Table 2: Probability ranges and corresponding defuzzified logarithmic centre of maximums for each likelihood level.

As the logarithmic centre of maximum values give what each likelihood level defuzzifies to, these can be used as starting singleton probabilities in Eqs. (3.7) and (3.9) and then by changing the population sizes the probabilities and likelihood levels for the lowest and highest cases can be examined. This is shown in Figure 21. Here each starting probability is for a singleton (denoted by a coloured circle) and as the population size is increased the probability and likelihood level decreases for the lowest likelihood case and increases for the highest likelihood case.

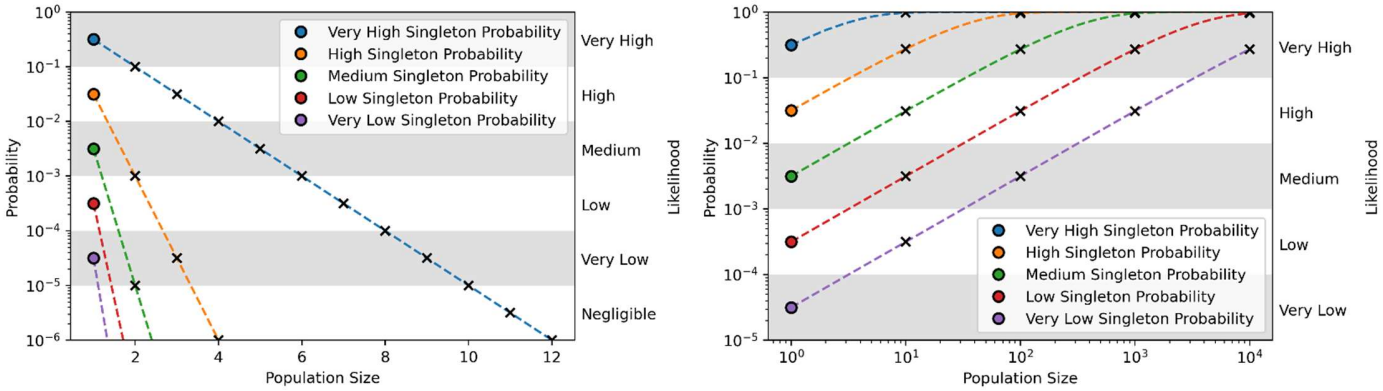


Figure 21: Probability changes for increasing populations sizes, starting from singleton, for the (left) lowest and (right) highest probability cases.

Figure 21 shows that the probabilities for the lowest case change at a much faster rate than those for the highest case, with all but the Very High likelihood starting probability being reduced to values corresponding to a Negligible likelihood after only an increase to a population of four, and that for the Very High likelihood taking an increase to a population of 11. On the other hand, for the highest case the probabilities increase as population increases and change at a much slower rate, with the population increasing by over three orders of magnitude for the Very Low likelihood starting probability to be increased to a value corresponding to the Very High likelihood. Due to the rates of change between these two cases being very different, and due to there being a need for a consistent set of population levels throughout all of the underlying theory and SSM, just one of the cases must be selected to determine the population levels from.

Using the highest case as the basis for determining the populations becomes the more appropriate choice. As large changes in probability occur for only small changes in population for the lowest case, for any SSM system asset population so small that their lowest case likelihoods have not been driven down to Negligible, these can be modelled explicitly in SSM as individual assets. Conversely, the population sizes required for the highest case are too large to model as individual

SSM system assets. Therefore, out of these two cases the most appropriate population scale to use is that for the highest case and so this should be used when determining the population levels.

With it decided to use the highest likelihood case, the population scale and levels can now in turn be formed using it.

3.4.4. Population Scale and Levels

An arbitrary number of population levels can be chosen for the population scale; however, an amount needs to be chosen that is both large enough to capture any required resolution and sufficiently small enough that the terms of the levels remain distinctively fuzzy rather than becoming, or being able to approximate to, crisp non-fuzzy levels. As will be shown below, a good choice for the number of population scale levels that meets these requirements is $N - 1$ non-singleton levels, and one singleton level, where N is the number of non-Negligible levels for likelihood. As SSM uses five non-Negligible likelihood levels (Very Low to Very High) this then results in four non-singleton population levels, which are named 'A Few', 'Several', 'Many' and 'Very Many', as well as the singleton population level that is named 'Singleton'.

As mentioned in Section 3.3.2, Negligible likelihood is a special case in which it is an absorbing boundary condition and as such, for any population sizes for both the lowest likelihood case and highest likelihood case, regardless of the population size and level a Negligible likelihood will remain Negligible despite any population level in either of the cases being applied to it.

With the population levels now determined, each level needs to be defined such that they each correspond to a range of population sizes. To do this, each likelihood level is first defuzzified to a probability value and then Eq. (3.7) is used to determine the minimum population size needed to shift the probability such that when re-fuzzified it now corresponds to a higher likelihood level. Then, with these minimum population sizes determined that give the population size required to shift a starting likelihood level to each subsequently higher likelihood level, a population range for each population level is formed.

Using the logarithmic centre of maximum probabilities as the starting probability, and using the probabilities corresponding to the start of each likelihood level as the target probabilities, both given in Table 2, the minimum population size required to shift a starting likelihood level to a given target likelihood level for the highest likelihood case is given in Table 3.

Highest Likelihood Case	Logarithmic Centre of Maximum	***Population Size Required to Shift to Target Likelihood Level***			
Starting Likelihood Level		*Low*	*Medium*	*High*	*Very High*
Very Low	3.1623×10^{-5}	3.2	31.6	317.8	3331.7
Low	3.1623×10^{-4}	-	3.2	31.8	333.1
Medium	3.1623×10^{-3}	-	-	3.2	33.3

High	3.1623×10^{-2}	-	-	-	3.3
Very High	3.1623×10^{-1}	-	-	-	-

Table 3: Minimum population sizes for shifting likelihood to higher levels, for the highest likelihood case.

In Table 3 the logarithmic centre of maximums have been given for all minimum population sizes at with decimal place for each. This is as population sizes are in reality integer numbers and so the minimum number required for a shift need to be rounded up to the next highest integer number, so calculating each to at least one decimal place, and also giving this value, shows clearly where the rounding up to the next highest integer number will occur. The blank entries correspond to shifts that cannot occur as the target likelihood level is either equal to the starting level or is lower level than the starting level. It is also seen that the minimum population to shift any likelihood level to the next highest level is four, after rounding up.

For the lowest likelihood case described above, this minimum population is large enough to shift all but the Very High likelihood level down to a Negligible level, showing further that for populations of up to 3 that they should be modelled explicitly as individual assets in SSM.

From Table 3 population ranges are deduced that correspond to a particular number of shifts from a starting likelihood to each higher likelihood. The minimum population sizes for a specific size shift are not all the same however, for instance, for the case of a shift of three likelihood levels moving from a Very Low likelihood to a High likelihood requires a population size of 318 whereas moving from a Low likelihood to a Very High likelihood requires a population size of 334. These are similar in size, but to determine generalised rules for shifts of specific likelihood levels these differences need resolving. Erring on the side of caution, it is better to overestimate the likelihood at these specific, few, and narrow edge cases. This leads to the conclusion that for determining a general rule for the minimum population required for a specific shift size in likelihood, if there a multiple different, but similar, population size changes that give rise to that shift then the lowest one should be used as the minimum population size required. This is graphically shown in Figure 22, where for the highest likelihood case, the minimum population sizes of 4, 32, 318 and 3332 have been chosen for likelihood shifts of 1, 2, 3 and 4 levels, respectively. Here it is shown that any overlap into a previous likelihood level is minimal.

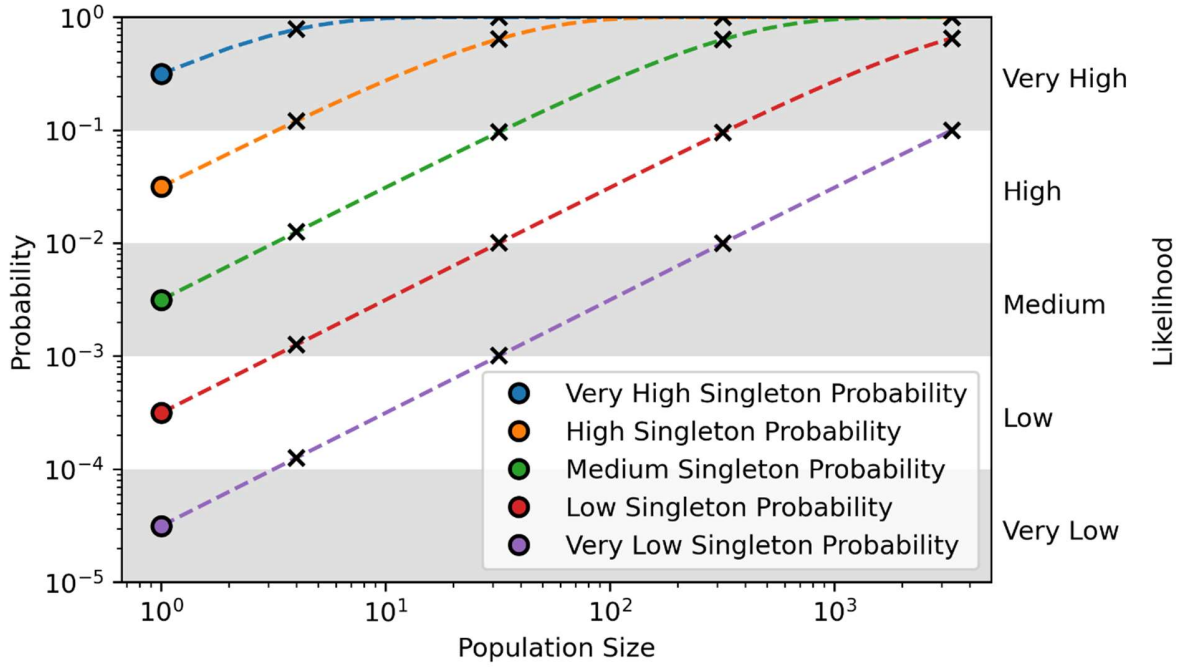


Figure 22: Probability changes for population sizes 4, 32, 318 and 3332, starting from a singleton, for the highest likelihood case.

To then determine the different population ranges from all this, a population range for each shift in likelihood level starts at the minimum population size required to cause that shift and ends at a population size of one less than the minimum size required for the shift to the next likelihood level above. The result is that four non-singleton population ranges are determined that correspond to, and define, the four non-singleton population levels and is given in Table 4.

Population Level	Population Range	Population Logarithmic Centre of Maximum	Number of Likelihood Levels Shifted
A Few	$4 - 31$	12	+1
Several	$32 - 317$	101	+2
Many	$318 - 3331$	1030	+3
Very Many	≥ 3332	-	+4

Table 4: Population levels and ranges, logarithmic centre of maximums and the number of likelihood levels shifted for the highest likelihood case.

In Table 4 the logarithmic centre of maximums are given for each of the population ranges as these correspond to what each population range would defuzzify to. Using these values and referring to Figure 21, a population level of A Few defuzzifies to a population size of 12, which is large enough to drive all of likelihood levels down to a Negligible level for the lowest likelihood case. From this it is concluded that for the lowest likelihood case, any population level greater than Singleton results in a likelihood of Negligible, regardless of the starting likelihood level. With the population scale and

levels now defined, fuzzy logic SSM rules can be made that give how the likelihood levels of the lowest and highest likelihood cases change from the average likelihood case for the different population levels.

3.4.5. Fuzzy Logic SSM Rules for Populations

Using the probabilities for the highest likelihood case and the definitions of the likelihood levels, population levels and their effects on shifts in likelihood levels for the highest likelihood case have been determined, as has a general rule for how the lowest likelihood case shifts likelihood levels for these given population levels. From this, fuzzy logic SSM rules are determined for how populations affect likelihoods in the lowest and highest likelihood cases.

For the **lowest likelihood** case, the resulting rules are that for populations up to three in size should be modelled explicitly as individual system assets in SSM, and for any non-Singleton population, the resulting lowest case likelihood is Negligible.

For the **highest likelihood** case, the resulting rules are given in the form of a lookup table (as lookup tables are what SSM uses internally). This lookup table is given in Table 5 and contains the full set of available fuzzy logic rules.

Highest Likelihood Case	***Average Member Likelihood***					
Population Level	*Negligible*	*Very Low*	*Low*	*Medium*	*High*	*Very High*
Singleton	Negligible	Very Low	Low	Medium	High	Very High
Few	Negligible	Low	Medium	High	Very High	Very High
Several	Negligible	Medium	High	Very High	Very High	Very High
Many	Negligible	High	Very High	Very High	Very High	Very High
Very Many	Negligible	Very High	Very High	Very High	Very High	Very High

Table 5: Fuzzy SSM rules giving highest likelihood case from average likelihood case, for different population levels.

With the fuzzy logic SSM rules for populations now determined, these need to be implemented into the SSM core model so that these can be implemented within SSM itself. How this is done is discussed in the next section.

3.5. Application of Theory into SSM Core Model

3.5.1. Introduction

For risk modelling in the SSM, there are several distinct stages: system definition, validation, risk calculation initialisation, and risk calculation.

Within the system definition stage, the user adds asserted assets and relationship. By default, the GUI inserts new assets as singletons, that is by default the population size of an asset takes the value ‘Singleton’. The user can then override this by changing the population size to one of the other values of ‘A Few’, ‘Several’, ‘Many’, or ‘Very Many’.

Within the validation, inferred assets are added into an SSM system model, created by construction patterns in the SSM knowledge base. For these inferred assets, their population sizes are given as the maximum population size from among the other nodes in the construction pattern. Additional properties are then added to all assets (including inferred assets):

- Misbehaviour Sets (MS): represent the likelihood and impact level for each type of (potentially threat induced) behaviour that can be displayed by the asset.
- Trustworthiness Attribute Sets (TWAS): represent the trustworthiness level (TWL) for each trustworthiness attribute (TWA) present at the asset.
- Control Sets (CS): represent the status of each type of control that may be present at the asset. Control sets have a coverage level that represent the proportion of an asset threat cause population that have the control applied to it, for assets matched to the threat cause within the matching pattern.

These are added based on the asset type: each type of asset will have certain types of controls, behaviours, and trustworthiness attributes. Their initial status is also defined based on default settings specified in the SSM's knowledge base (domain model):

- Misbehaviour Sets: are assigned an impact level.
- Trustworthiness Attribute Sets: are assigned an average trustworthiness level
- Control Sets: are assigned an initial average Coverage level.

For non-singleton assets which represent an asset population, the lowest and highest likelihood case TWLs and Coverage levels are then defined for each TWAS and CS respectively. This is done using one of two methods, as specified in the domain model by the domain modelling expert, for each combination of asset type and either TWA or Control type:

- using binomial calculations taking the average likelihood case level as input (equivalent to assuming that the members of the population are independent of each other), or
- making them equal to the average likelihood case (equivalent to assuming that the population is composed of identical, non-independent assets).

Before starting a risk calculation, the user can override the values of the impact level of any Misbehaviour Set, the average, lowest or highest likelihood case trustworthiness level of any TWAS, or the average, lowest or highest likelihood case Coverage level of any Control Set. For this, the user would need to know what the relevant likelihood or trustworthiness value represents in terms of population behaviour such that they can override it to an appropriate value. If they do override it, the risk calculator will use the level they specified as input for the subsequent calculation. If they do not provide an override, the risk calculator will re-initialise the corresponding level before proceeding with the rest of the calculation.

What this means is that a user can choose to specify their own level for the average likelihood case TWL or Coverage and allow the lowest and highest likelihood cases to be recalculated based on their input at the start of the risk calculation. Alternatively, they could specify their own level for a lowest likelihood and/or highest likelihood case TWL or Coverage, in which case the risk calculator will use that input instead of recalculating it. The risk calculator will then initialise the lowest, highest and average case likelihoods for each MS:

- if the MS represents a behaviour type that has an associated trustworthiness attribute, the MS likelihood will be set to the reflection of the corresponding TWAS trustworthiness level. A Reflection is an operation that relates trustworthiness to likelihood and is defined as the equivalent scales in Table 1, i.e. Very Low likelihood maps to Very High trustworthiness, etc. They are different representations of the same attribute, with each being a reflection of the other.

- if the MS has no associated TWAS, its likelihood will be initialised to the lowest possible likelihood level.

The final stage mentioned is the risk calculation stage, which involves an iterative process of calculations to find convergence to values of likelihoods and trustworthiness levels, which in turn determine risk levels of threats within a modelled system. The lowest, highest, and average case threat likelihoods are calculated, based on the likelihoods of causes (taking into account the effects of controls), including the effects of 'local' controls that may reduce the 'effective' likelihoods of specific causes. As discussed in more detail later, 'local' controls act within a threat on the threat cause assets and threat cause likelihoods whilst 'global' controls act at the level of the threat and threat likelihood itself.

Each cause of a threat is associated with properties of the asset or assets fulfilling a specific role in the threat (i.e. matching a specific node in the threat matching pattern):

- for a primary threat, each cause is a trustworthiness attribute of the relevant asset(s), where the effective trustworthiness level determines the likelihood that the cause is 'active'; and

- for a secondary threat, each cause is an asset (mis)behaviour, where the effective likelihood of that behaviour determines the likelihood that the cause is 'active'.

Each asset having the relevant role in the threat makes its own contribution to that threat cause. The effective trustworthiness or likelihood level is found from:

- the corresponding TW or likelihood level for the TWAS or MS representing the relevant trustworthiness attribute or misbehaviour at that asset; and

- the status of any CS representing a relevant 'local' control at that asset, where the CS forms part of an enabled control strategy.

Once the contribution to the threat cause has been found for each asset with the relevant role, the threat cause likelihood is found by combining them. How this is done depends on what type of node has that role in the threat matching pattern since the node type encodes the type of population behaviour for any assets matching to that node in the threat pattern, as discussed later.

Once the likelihood of each threat cause has been found by combining contributions from the relevant assets, the likelihood of the threat can be found. In the current version of SSM, this is that of the least likely asset contribution. The likelihood of the threat equals the overall threat causation likelihood, after reductions representing the effect of global controls in any enabled control strategy that addresses the threat.

Unless the threat is a singleton, it represents a population of threats (linked to populations of involved assets). In that case, these calculations are used to find the lowest, highest and average case likelihoods among members of the threat population, using the lowest, highest and average case likelihood for each asset cause contribution as appropriate.

The effects of a threat are specific misbehaviours at assets with specific roles in the threat (i.e. one or more MS related to the threat). The likelihood of an MS representing a threat effect must be at least equal to the likelihood of the threat. It may be higher because each MS may be caused by several threats. Since any of those threats would cause the relevant misbehaviour, the likelihood of each MS is found using the fuzzy logic OR operator to be the likelihood of the most likely threat causing the MS. If the associated asset is not a singleton, this calculation is done separately to find the lowest, highest and average case likelihood of the associated misbehaviour at that asset, with each o9f these being kept separate for future calculations.

The mechanism for threat propagation is that most misbehaviours are linked to a corresponding trustworthiness attribute which is then undermined by the misbehaviour, thus weakening the asset and making it more vulnerable to other threats. The last step is therefore to feed back for each asset by reducing the TW level of each TWAS to reflect the calculated likelihoods of the corresponding MS. If the asset is not a singleton, this is also done for the lowest, highest and average asset.

The whole process to calculate threat likelihood, MS likelihood and TWAS trustworthiness is then repeated iteratively until convergence, meaning that all the propagated threat and MS likelihoods are driven to equilibrium levels. Note that the binomial assumption (that assets in a population are independent) and calculations are not used in this iterative risk calculation procedure stage as this stage is a cause-and-effect calculation that takes into consideration dependencies between system assets based on the structure of the system. The binomial assumptions and calculations are only used in the risk calculation initialisation stage, as described above.

3.5.2. Node Types

The different node types that can be included in threat patterns, as well as what they mean, are given below. These previously included mandatory unique, mandatory non-unique, and non-unique optional nodes and have now been updated to instead include unique, non-unique necessary, non-unique sufficient, and non-unique optional nodes.

3.5.2.1 Unique Node (Root)

A unique node is matched by one asset at a time and is denoted schematically in Figure 23. If more than one system asset matches this node in a pattern, it means there is more than one match to the pattern, and a threat associated with the pattern will therefore have multiple instances, each involving a specific matching asset. Note that this means there will be a population of threats if the associated matching pattern contains a root node that is matched by a non-singleton asset, and it makes sense to find the lowest, highest and average case likelihood of threats in this population. If there is a threat cause at such a node, the lowest, highest or average case matching asset is used to determine the lowest, highest or average case threat likelihood.

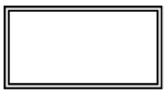


Figure 23: Schematic representation of a unique node.

3.5.2.2 Non-Unique Necessary Node

A non-unique necessary node can be matched by multiple system assets at a time, including populations and singletons, and is denoted in Figure 24. If there is more than one matching asset, there is one threat related to them all. All assets are *necessary* (need to be involved in the threat) for the threat effect to be present, so if there is a threat cause at such a node the lowest likelihood matching asset (least likely to cause the threat) is used to determine threat likelihood.

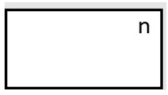


Figure 24: Schematic representation of a non-unique necessary node.

3.5.2.3 Non-Unique Sufficient Node

A non-unique sufficient node can be matched by multiple system assets at a time, including populations and singletons, and is denoted schematically in Figure 25. If there is more than one matching asset, there is one threat related to them all. Any asset is *sufficient* (just one asset needs to be involved) in the threat for the threat effect to be present, so if there is a threat cause at such a node the highest likelihood matching asset (most likely to cause the threat) is used to determine threat likelihood.

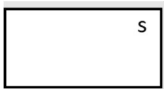


Figure 25: Schematic representation of a non-unique sufficient node.

3.5.2.4 Non-Unique Optional Node

A non-unique optional node can be matched by multiple system assets at a time, including populations and singletons, and is denoted schematically in Figure 26. This asset type cannot be a cause to a threat but may carry controls that affect the threat likelihood calculation.

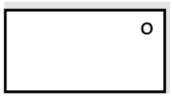


Figure 26: Schematic representation of a non-unique optional node.

3.5.3. Threat Cause Likelihoods

For a threat cause, the node type determines which of the lowest, highest, and average case likelihoods to use for the threat cause likelihood. A set of rules is given below for determining the likelihood associated with a threat cause coming from different node types. Rules for *unique*, *non-unique necessary*, and *non-unique sufficient* nodes are given, where the rules cover cases for singleton and non-singleton assets, as well as multiple matching system model assets that can be both singletons and non-singletons for the non-unique node cases.

The contribution to threat causation from each asset may be affected by some types of controls (see below). Where this is the case, the effect of the controls is applied to each asset before combining the contributions of different assets that match a specific node.

Once each threat cause likelihood has been determined, considering all assets that match the node based on the node type in the manner described below, the overall threat cause likelihood can be found. Since all causes of a threat must be present for the threat to arise, the overall threat cause likelihood is taken as the minimum of all threat cause likelihoods. For non-singleton threats (those with at least one root node matching a non-singleton asset), this applies separately to the lowest, highest and average case threat likelihood.

3.5.3.1 Threat Cause Likelihood for Unique Threat Cause Node

A unique node is the threat cause, which corresponds to a single asset. For this, the threat cause likelihood from this node is given by the average case member likelihood at that node.

Rule:

```
IF node type IS unique:

    FOR each population asset at node:

        Asset cause contribution to Average Case threat likelihood = Average
        Case likelihood of the population

        Asset cause contribution to Lowest Case threat likelihood = Lowest Case
        likelihood of the population

        Asset cause contribution to Highest Case threat likelihood = Highest
        Case likelihood of the population

    FOR each singleton asset at node:

        Asset cause contribution to Average Case threat likelihood = Average
        Case likelihood

        Asset cause contribution to Lowest Case threat likelihood = Average Case
        likelihood

        Asset cause contribution to Highest Case threat likelihood = Average
        Case likelihood
```

3.5.3.2 Threat Cause Likelihood for Non-Unique Necessary Threat Cause Node

A non-unique necessary node is the threat cause. This corresponds to potentially multiple system assets, of which some may be singletons and others may be populations.

A system model asset population in this scenario is acting as a *lowest likelihood* case scenario, where all the threat cause population need to be experiencing the threat cause for the threat to be present in the target asset. At the risk calculation initialisation stage the system model asset misbehaviour or trustworthiness attribute would have the relevant likelihood or trustworthiness level initialised using the lowest case binomial calculation and its average case and population size values (unless the user overrides the lowest or highest case trustworthiness level for a trustworthiness attribute). During the risk calculation phase itself, these initial values can be overwritten based on Misbehaviours resulting from upstream threats to the one being considered.

Once each population at a non-unique necessary node has been resolved, all the system assets matching that node then need to be resolved together as well. Each system asset can result in the threat cause and so the threat cause likelihood is the least likely from among the different population-resolved assets at the node.

Rule:

```
IF node type IS non-unique necessary:

    FOR each population asset at node:

        Asset cause contribution to Average Case threat likelihood = Lowest Case
        likelihood of the population

        Asset cause contribution to Lowest Case threat likelihood = Lowest Case
        likelihood of the population

        Asset cause contribution to Highest Case threat likelihood = Lowest Case
        likelihood of the population

    FOR each singleton asset at node:

        Asset cause contribution to Average Case threat likelihood = Average
        Case likelihood

        Asset cause contribution to Lowest Case threat likelihood = Average Case
        likelihood

        Asset cause contribution to Highest Case threat likelihood = Average
        Case likelihood
```

Therefore, threat cause likelihood = *min*(asset threat cause likelihoods) applied to lowest, highest, and average cases separately.

3.5.3.3 Threat Cause Likelihood for Non-Unique Sufficient Threat Cause Node

A non-unique sufficient node is the threat cause. This corresponds to potentially multiple system assets, of which some may be singletons and others may be populations.

A system asset population in this scenario is acting as a *highest likelihood* case, where only one member of the threat cause population needs to be experiencing the threat cause for the threat to be present in the target asset. At the risk calculation initialisation stage the system model asset misbehaviour or trustworthiness attribute would have the relevant likelihood or trustworthiness level initialised using the highest case binomial calculation and its average case and population size values (unless the user overrides the lowest or highest case trustworthiness level for a trustworthiness attribute). During the risk calculation phase itself, these initial values can be overwritten based on Misbehaviours resulting from upstream threats to the one being considered.

Once each population at a non-unique sufficient node has been resolved, all the system assets matching that node then need to be resolved together. Again, as each system asset can result in the threat cause, the overall threat cause likelihood is the most likely threat cause likelihood from among the population-resolved assets at that node.

Rule:

```
IF node type IS non-unique sufficient:

    FOR each population asset at node:
```

```
        Asset cause contribution to Average Case threat likelihood = Highest
        Case likelihood of the population

        Asset cause contribution to Lowest Case threat likelihood = Highest Case
        likelihood of the population

        Asset cause contribution to Highest Case threat likelihood = Highest
        Case likelihood of the population

    FOR each singleton asset at node:

        Asset cause contribution to Average Case threat likelihood = Average
        Case likelihood

        Asset cause contribution to Lowest Case threat likelihood = Average Case
        likelihood

        Asset cause contribution to Highest Case threat likelihood = Average
        Case likelihood
```

Therefore, threat cause likelihood = MAX(asset threat cause likelihoods) applied to lowest, highest, and average cases separately.

3.5.3.4 Summary

	Singleton	Population
Unique Node	Average = Average Case Likelihood Lowest = Average Case Likelihood Highest = Average Case Likelihood	Average = Average Case Likelihood Lowest = Lowest Case Likelihood Highest = Highest Case Likelihood
Sufficient Node	Average = Average Case Likelihood Lowest = Average Case Likelihood Highest = Average Case Likelihood	Average = Lowest Case Likelihood Lowest = Lowest Case Likelihood Highest = Lowest Case Likelihood
Necessary Node	Average = Average Case Likelihood Lowest = Average Case Likelihood Highest = Average Case Likelihood	Average = Highest Case Likelihood Lowest = Highest Case Likelihood Highest = Highest Case Likelihood

Table 6: Summarised threat cause likelihood rules.

3.5.4. Controls

The effect of controls on threat likelihoods depends on the node type where the control acts, and the types of controls. There are many different types of controls, but they fall into two classes that determine how they affect threat likelihood: local and global. The meaning of the two control types are firstly given below, then their effect on threat likelihood is

presented, the concept of control coverage is given, and finally the effect of controls on asset's threat cause contribution likelihood is discussed.

The likelihood of a threat is only affected by controls when there is a control strategy group that addresses the threat containing those controls at assets with specific roles in the threat, and when all global controls are enabled at the relevant assets or when there are no global controls in the control strategy group.

3.5.4.1 Global Control Type

Global controls act at the level of the threat as an essential part of the control strategy without which the other controls are useless. A global control strategy has a blocking effect (expressed as a TW level) on the likelihood of the threat itself that sets an upper bound to value that likelihood can take, thus also giving an upper bound to the given threat's likelihood contribution to any threat effects too. A global control applies to the whole population of any threat effect asset.

Global controls have a radio button for the user to select if the control is enabled or not. If any global control in a control strategy is not enabled, then the blocking effect is set to the lowest possible TW level, i.e. the control strategy (including local controls – see below) is ignored.

This is the only type of control previously supported by the SSM.

If a control strategy only contains global controls, its effect can be applied after the overall threat causation likelihood has been found, by reducing the threat likelihood before determining the likelihood of threat effects. As SSM only supported global controls up until now, the effect of controls is always applied at this point.

3.5.4.2 Local Control Type

Local controls apply to threat cause assets, for both singleton and population assets. Local controls act to reduce the likelihood of threat causes from those assets to which it applies, where it forms part of a control strategy to address the threat that is enabled. For example, a threat of a possible phishing attack involving users will have a node in the threat pattern that will be matched by users in the system and there will be a cause at this node linked to a user trustworthiness attribute, e.g. their astuteness. If a local control is applied and users are provided with training against phishing attacks, this will increase their effective astuteness against this specific threat.

The fact that a control is 'local' is specified in the domain model as a link from a control type to a misbehaviour type, indicating that it affects the effective likelihood for that type of misbehaviour.

- For secondary threats it decreases the 'effective' likelihood for a misbehaviour set of this type that causes that secondary threat.
- For primary threats, this increases the 'effective' trustworthiness level for the TWAS whose type is associated with that misbehaviour and causes that primary threat.

In each case, this is done IF the control is part of a control strategy addressing the threat, AND all global controls in that control strategy are enabled (see above), BEFORE calculating the contribution of the asset with the control to threat causation likelihood.

For example, in the case of user training against a phishing attack, the anti-phishing control would be linked to the misbehaviour deception, which is the misbehaviour that affects the astuteness TWA. Note that because this control only

affects phishing, it only increases 'effective' astuteness for this type of threat. The relationship between local controls and threats they affect is via a control strategy which specifies which threats are addressed, and which controls are needed at which nodes to achieve their effect.

A local control when enabled at an asset has a coverage level expressed as a trustworthiness level. Strictly speaking, this is the reflection of the likelihood that a member of the population selected at some arbitrary time does NOT have the control. Since the TWL will be higher if more members have the control, it may be easier to think of the TWL as a measure of likelihood that they DO have the control (a higher TWL means it is more likely that a selected member will have the control).

3.5.4.3 Control Type Effect on Threat Likelihood

Different combinations of control types can be present at the system assets of given matched threats. How those different combinations affect the threat likelihood is given below.

Control Type Present	Effect on Threat Likelihood
Only local controls	The blocking effect of the CSG comes from a reduction in threat cause contribution likelihood that depends on the blocking effect of the control strategy and the coverage of the local control. The reduction can be calculated as Reflection(MIN(Coverage, CSG Blocking Effect)). Note that the coverage level should be taken as the lowest, highest or average coverage, following the same rules as for the threat cause that is being reduced. See below.
Only global controls	The blocking effect of the CSG is applied at the end of the threat likelihood calculation, to cap the threat likelihood, if and only if all the controls have been enabled. This cap is applied to the lowest, highest, and average threat likelihoods.
Both local and global controls	The blocking effect of the CSG comes entirely from the reduction of cause contribution likelihoods achieved by the local controls but applies only if all the global controls are enabled.

Table 7: Threat likelihood effect for different combinations of controls present at system assets of matched threats.

A control being active means that they are selected by the SSM user. This is important as the SSM user needs to be aware of which controls are applied within a given system so that they can accurately predict risks and threats in real-life systems. While local controls have an effect based on their coverage, and that is specified in the domain model, such controls will be ignored unless they have been enabled by the user.

Note that there is a trade-off between SSM usability and user awareness creation. Forcing users to select controls before their effect can be considered means the SSM raises user awareness of the need for certain types of controls. However, it also means the SSM user must do more work to select controls when modelling typical (reasonably secure) system

configurations. We may at some stage need to add functionality allowing 'knowledgeable' users to automatically select local controls. This may be related to similar functions allowing users to automatically select certain types of controls, e.g. those that are mandated in a cyber security standard like Cyber Essentials.

3.5.4.4 Control Coverage

The coverage of a local control depends on the population behaviour type and so three different coverage levels exist for each local control, one for each of the lowest, highest, and average cases. The lowest and highest cases are determined during the risk calculation initialisation stage using lowest and highest case lookup tables respectively, as determined from the binomial assumptions and calculations, and also the population size and average case coverage level.

Population Behaviour Type	***Coverage Value for the Different Population Behaviour Types***
Average likelihood case	The default or user set value of the coverage level, representing the coverage of the average member of the population.
Lowest likelihood case	The coverage level of the lowest likelihood member of a population, as calculated from lowest likelihood binomial assumptions during the risk calculation initialisation phase. The value for this coverage will be equal to or greater than that for the average likelihood case and represents the member of the population that is most likely to have the control implemented.
Highest likelihood Case	The coverage level of the highest likelihood member of a population, as calculated from highest likelihood binomial assumptions during the risk calculation initialisation phase. The value for this coverage will be less than or equal to that for the average likelihood case and represents the member of the population that is least likely to have the control implemented.

Table 8: The coverage value for the different population behaviour types.

3.5.4.5 Control Effect on Threat Cause Contribution Likelihood

Rules for the effects of local controls on threat likelihood are given below. The effect of a control is limited by the overall effectiveness (blocking effect) of the control strategy addressing the threat of which the control is a part. This depends on whether the control strategy is enabled. If the control strategy has any 'global' controls that are not enabled, then the whole control strategy is not enabled. This may include global controls at non-unique as well as unique nodes, so there are four possibilities:

- if no assets match an optional node where a global control forms part of a control strategy, then the control strategy is incomplete and so is not enabled;
- if an asset matches a unique node, but it lacks the control or does not have it enabled, the control strategy is also not enabled;
- if one or more assets match a non-unique node, some of which lack the global control or do not have it enabled, the control strategy is also not enabled

- if one or more assets match a node, and they all have the global control enabled, the control strategy is complete and enabled.

It is possible to have a control strategy with no global controls. In that situation, the control strategy is assumed to be enabled. But implementers beware to check that an absence of global controls is not because they should be at optional nodes where there are no matching assets.

If a control strategy is not enabled, the blocking effect is set to the lowest possible TW level, so the control strategy (including its local controls) has no effect. If it is enabled, the blocking effect depends on the type of control strategy (and threat), as specified in the domain model. In either case, if the control strategy includes local controls, they can lower the effective likelihood of an asset cause contribution only to a level that reflects the blocking effect (TW level) for the control strategy.

It is possible for the same local control to be included in more than one control strategy that addresses the threat. Since any control strategy is enough to address a threat on its own, their blocking effects can be combined using the fuzzy logic OR operator, leading to the conclusion that the effect of the local control is limited by the highest blocking effect of any relevant control strategy of which it a part.

Having found the blocking effect of the (relevant) control strategy, the effect of local controls on asset cause contributions can be determined. Since threat causes cannot be associated with optional nodes, there is no need to consider the effect of local controls at optional nodes. The cases that do need to be considered are given below.

If a control strategy has any local controls, then its effect is entirely determined from the above rules. However, if a control strategy has no local controls, it limits threat likelihood to a level that reflects the blocking effect (TW level) for the control strategy. The above rules can be thought of as a reduction or cap to the local threat cause likelihood, and be applied after the cause contribution likelihoods (including the effect of local controls from other control strategies) have been determined and combined. The result is the overall threat likelihood. If the threat is a non-singleton (i.e. it involves non-singleton assets matching unique nodes in the threat), then the same reduction is applied to the lowest, highest and average case threat likelihoods.

3.5.4.5.1. Singleton Asset Matching any Node

Since the asset is a singleton, the control coverage represents only the temporal aspect (how much of the time is the control enabled). This is expressed as one average coverage, corresponding to the likelihood of the control being enabled when the threat arises. The threat may or may not be a singleton, as this depends on whether non-singleton assets match any of its unique nodes. Assuming the threat is not a singleton, the effect of the control will be:

```
FOR asset cause contribution to Average Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Average Case Coverage,
    CSG Blocking Effect)
FOR asset cause contribution to Lowest Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Average Case Coverage,
    CSG Blocking Effect)
FOR asset cause contribution to Highest Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Average Case Coverage,
    CSG Blocking Effect)
```

If the threat is a singleton (i.e. if its unique nodes are only matched by singleton assets), it will not have lowest or highest case likelihoods, and the effect of the control will be:

```
FOR asset cause contribution to Average Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Average Case Coverage,
    CSG Blocking Effect)
```

3.5.4.5.2. Population Asset at Unique Node

The asset is not a singleton and represents an asset population, so the control has three coverage levels, representing the temporal aspect (as above), and the prevalence (how many members of the population have it). The Average level corresponds to the likelihood of the control being enabled when the threat arises for a randomly chosen member of the population. The lowest and highest cases correspond to the same likelihood for the lowest and highest case members of the population. Because the asset is non-singleton, but the node is unique, there must be a population of threats corresponding to the population of assets. The three coverage levels affect the asset threat cause contribution for threats chosen on the same basis:

```
FOR asset cause contribution to Average Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Average Case Coverage,
    CSG Blocking Effect)
FOR asset cause contribution to Lowest Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Lowest Case Coverage,
    CSG Blocking Effect)
FOR asset cause contribution to Highest Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Highest Case Coverage,
    CSG Blocking Effect)
```

3.5.4.5.3. Population Asset at Non-Unique Necessary Node

Since the asset is not a singleton, the control has three coverage levels as described above. Because the asset matches a non-unique necessary node, all matching assets must contribute to cause the threat, so the asset cause contribution is based on the lowest case member selected from the asset population. Hence the effect of the control is based on the lowest case coverage level:

```
FOR asset cause contribution to Average Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Lowest Case Coverage,
    CSG Blocking Effect)
FOR asset cause contribution to Lowest Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Lowest Case Coverage,
    CSG Blocking Effect)
FOR asset cause contribution to Highest Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Lowest Case Coverage,
    CSG Blocking Effect)
```

3.5.4.5.4. Population Asset at Non-Unique Sufficient Node

Since the asset is not a singleton, the control has three coverage levels as described above. Because the asset matches a non-unique sufficient node, any one matching asset could cause the threat, so the asset cause contribution is based on

the highest case member selected from the asset population. Hence the effect of the control is based on the highest likelihood case coverage level:

```
FOR asset cause contribution to Average Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Highest Case Coverage,
    CSG Blocking Effect)
FOR asset cause contribution to Lowest Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Highest Case Coverage,
    CSG Blocking Effect)
FOR asset cause contribution to Highest Case threat likelihood:
    Local control set reduction/cap = Reflection(MIN(Highest Case Coverage,
    CSG Blocking Effect)
```

3.5.5. Threat Effect Likelihoods

The likelihood that a threat will cause an effect is simply the likelihood of the threat. However, as shown above, each effect may be associated with a population of assets, and have three likelihood levels (lowest, highest and average). Equally, a threat may represent a population of threats (where root nodes in its matching pattern are matched with non-singleton assets), which has lowest and highest case as well as average likelihood levels. Arguably even a singleton threat (one whose root nodes only match singleton assets) has three likelihood levels that just happen to be identical (since the lowest, highest and average cases all relate to the one threat).

A set of rules is therefore needed for determining the likelihood of threat effects arising at each type of node. These are given below, and specify how to find the lowest, highest, and average case likelihoods for affected nodes that match both singleton and non-singleton system model assets. Here the 'optional' nodes are ignored as optional nodes do not give rise to threat effects, although right now the SSM does allow effects at optional nodes.

The likelihood of an effect (Misbehaviour) at an asset is then found from the most likely threat that would cause that effect. Hence asset effect likelihood = MAX(threat contributions to that likelihood), applied separately to lowest, highest, and average cases.

3.5.5.1 Unique Node

The threat effect arises at a unique node. All matching assets are independent and can be affected differently because each one is affected by a different threat.

Rule:

```
IF node type IS unique:
    FOR each population asset at node:
        Threat contribution to Average Case asset effect likelihood = Average
        Case threat likelihood
        Threat contribution to Lowest Case asset effect likelihood = Lowest Case
        threat likelihood
```

```
        Threat contribution to Highest Case asset effect likelihood = Highest
        Case threat likelihood

    FOR each singleton asset at node:

        Threat contribution to Average Case asset effect likelihood = Average
        Case threat likelihood
```

3.5.5.2 Non-Unique Necessary Node

The threat effect arises at a non-unique necessary node. All matching assets are affected in the same way.

Rule:

```
IF node type IS non-unique necessary:

    FOR each population asset at node:

        Threat contribution to Average Case asset effect likelihood = Average
        Case threat likelihood

        Threat contribution to Lowest Case asset effect likelihood = Lowest Case
        threat likelihood

        Threat contribution to Highest Case asset effect likelihood = Highest
        Case threat likelihood

    FOR each singleton asset at node:

        Threat contribution to Average Case asset effect likelihood = Average
        Case threat likelihood
```

3.5.5.3 Non-Unique Sufficient Node

The threat effect arises at a non-unique sufficient node. There can be no threat effect here, though it is up to the domain modeler to enforce this rule. The way to avoid this (for the domain modeller) is to add a separate non-unique necessary node with the same relationships and that will match the same assets but the effect will be able to be put in.

Rule:

There can be no threat effect here (up to the domain modeller to enforce the rule). The effect should hit all assets in the match.

3.6. Worked Example with Lamp Posts from UC1

In the FogProtect Smart Cities Use Case 1, a population of smart lamp posts are all recording the same scene, each from a different view. The lamp posts all transmit their video data to the cloud where it is aggregated together and stored, ready to be retrieved if an event were to occur at the location of the scene that requires the security footage to be reviewed.

Each lamp post contains equipment to record the scene and transmit the data to the cloud, such as a router, server, LAN, and a camera that records the scene. If one of the lamp posts were to be broken into, there would be a loss to both the integrity and availability of the camera video data from that lamp post, with these losses potentially affecting the aggregated video data in the cloud.

To examine how the population of lamp posts affects the loss of integrity and availability of the aggregated video data, the threat path, arising from threat propagation and cascades, for one lamp post being broken into leading to these losses needs to be examined first.

For the loss of integrity of the aggregated data, the lamp post being broken into causes the LAN within the lamp post to become compromised, which in turn results in a loss of control of the lamp post hosting server. This then causes a loss of authenticity of the video data from that lamp post to become compromised as it is stored locally on that server, and then this compromises the video data sent from the lamp post to the cloud via the local authority proxy. This then finally results in a loss of integrity of the aggregated lamp post video data. This is shown in a general schematic representation in Figure 27, where each icon represents the resulting effects of a threat acting on assets either inside the lamp post, or on the lamp post itself in the case of the physical intrusion.

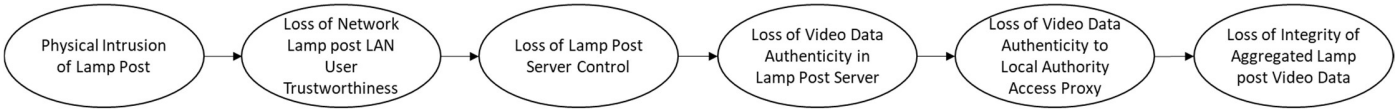


Figure 27: Threat effect path for the loss of integrity of the video data within a lamp post, starting from physical intrusion.

For the loss of availability of the aggregated video data, physical intrusion of the lamp posts leads to the video data within it being compromised due to destruction or theft of the lamp post server, this then causes a loss of availability of the video data within the cloud, accessed from the lamp post via the local authority proxy. This then finally results in a loss of availability of the aggregated lamp post video data. This is also shown in a general schematic representation, in Figure 28.

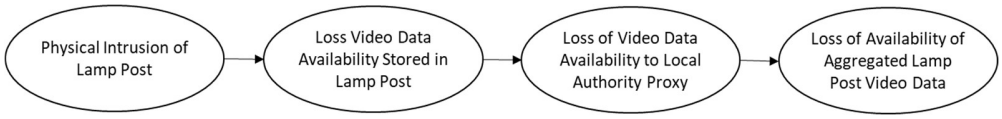


Figure 28: Threat effect path for the loss of availability of the video data within a lamp post, starting from physical intrusion.

For a population of lamp posts, the threat path that occurs for one lamp post also occurs for every other lamp post in the population too. The loss of integrity is a highest likelihood case as a compromise of any one lamp post results in the loss of integrity. This is shown in a general schematic representation in Figure 29, with there being N lamp posts in the population. Here any of the threat paths from each lamp post population member will cause the loss of integrity in the video data and so just any one of those paths needs to be completed for it to occur. This is shown by each threat path independently resulting in the aggregated video data loss of integrity.

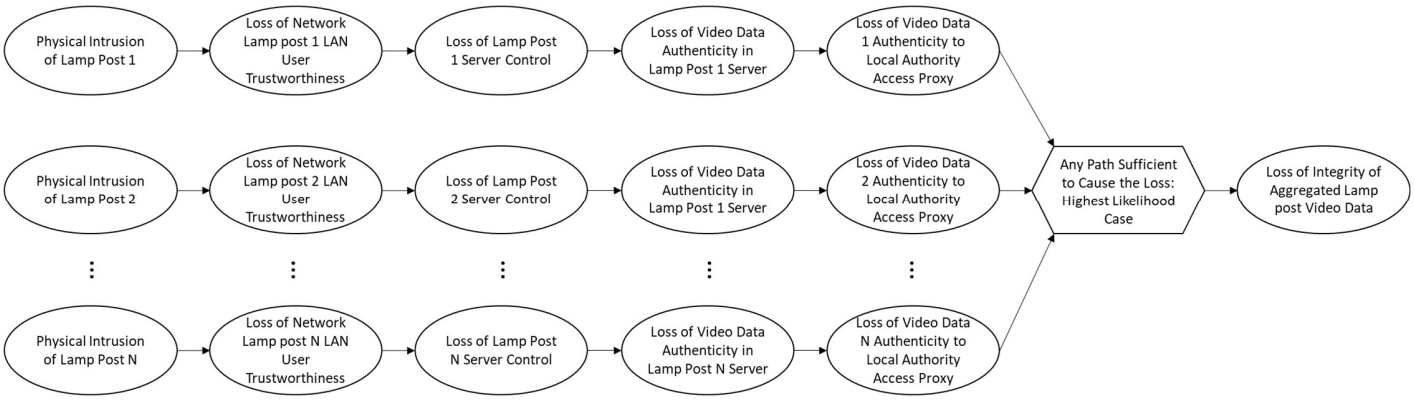


Figure 29: Threat effect paths for the loss of integrity of video data for a population of N lamp posts, starting from physical intrusion.

On the other hand, the loss of availability of the video data is a 'lowest likelihood case' effect since every copy of the data needs to be compromised before the scene is no longer being recorded and there is a loss of availability of the aggregated data (the data copies are effectively redundant). With each lamp post in the population recording the same scene, so long as there is at least one member of the lamp post population not compromised then the aggregated data is still being appended by the non-compromised lamp post. This is shown in a general schematic representation in Figure 30, with there being N lamp posts in the population. Here all the threat paths from every lamp post population member need to occur to cause the loss of availability in the aggregated video data. This is represented by the threat paths from all the lamp posts joining together before the final loss of aggregated data availability is reached in the figure.

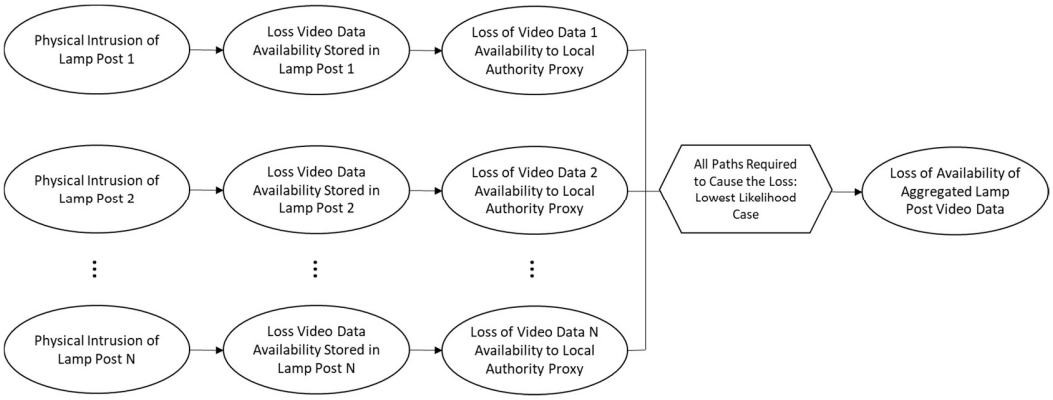


Figure 30: Threat effect paths for the loss of availability of video data for a population of N lamp posts, starting from physical intrusion.

Comparing the examples, as the number of lamp posts recording the scene increases, the number of integrity-based threat paths increases and, as each one can individually lead to the aggregated data loss of integrity, the likelihood of integrity violation happening increases with the population membership increasing in number. As the number of lamp posts recording the scene increases, the likelihood of the loss of availability of the aggregated data instead decreases as there are more lamp posts that all have to be experiencing a loss of availability in their video data.

If a local control is introduced that better secures the lamp posts, this helps prevent the lamp posts being broken into, though only for the proportion of the population with the control implemented. For the loss of integrity of the aggregated video data, if the control is applied on every member of the population except one then the threat path still exists and so the loss of integrity still occurs, as shown in Figure 31. Here the green ticks represent the lamp posts with the controls applied and the corresponding grey dashed threat paths show the threat paths that do not occur, and the red cross

represented the lamp post that does not have the control applied, with the red threat path showing the threat path that still occurs and leads to the loss of integrity.

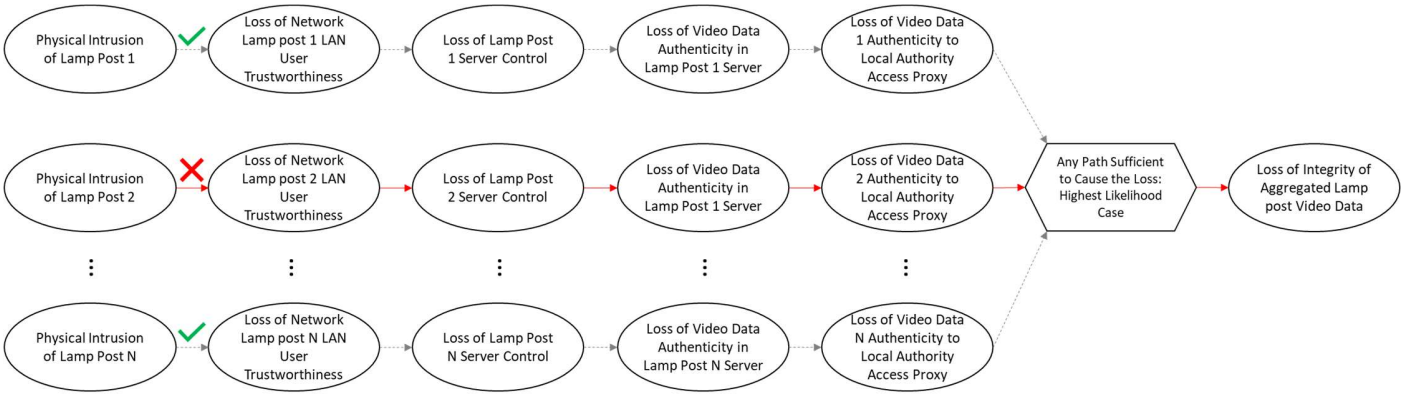


Figure 31: Threat paths for the loss of integrity of video data for a population of N lamp posts, with controls applied (green ticks) to all but one lamp post (red cross).

For the loss of availability of the aggregated video data, if the control is only applied on one member of the lamp post population, and every other member does not have it implemented, then the loss of availability does not occur, as shown in Figure 32. Here the red crosses and threat paths again represent the threat paths that do not have controls applied and so occur, and the green tick and grey dashed threat path represent the threat path with the control applied and so does not occur. The dashed red lines at the final stage of the threat path show that whilst those threat paths occur the loss of availability of the aggregated data does not occur as the greyed path at the stage is not present.

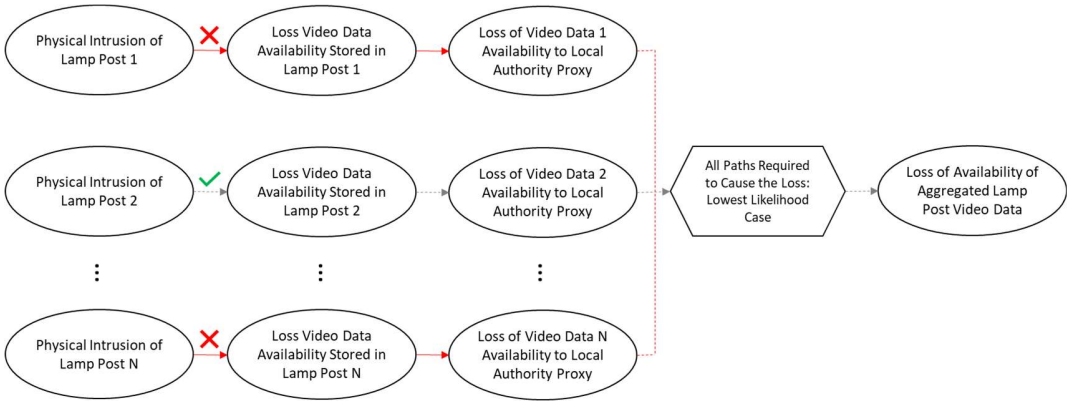


Figure 32: Threat effect paths for the loss of availability of video data for a population of N lamp posts, with a control applied to just one (green tick) of the lamp posts (red cross).

3.7. Summary

Limitations of the previous approach of SSM regarding risk modelling of populations have been discussed and different cases involving population sampling have been presented to overcome this. Two brief thought-experiment examples were used to show and highlight two different cases of population behaviours, one example corresponding to redundancy and the other to where just a single member of a population causing an effect results in that effect being present in the population as a whole. From these, three different cases of population behaviour were determined: the lowest, highest, and average likelihood cases. With these cases for populations determined, mathematics describing them was then derived to enable their application within the SSM core model.

For a population, probabilities and likelihoods of threat causes and threat effects form distributions. This allowed the lowest and highest probability cases to be determined from the average probability case. For situations where members of a population are independent of each other, this is done via the use of binomial distributions. For situations where the members of a population are dependent on each other, the lowest and highest cases are instead equal to the average case.

With equations derived relating those lowest and highest case probabilities to the average case probability derived, they were used to examine how the lowest and highest cases change as the population size is changed from a singleton to higher sizes. This examination was for both the probabilities and the likelihoods. It was found that the population sizes needed to cause likelihood level changes vary greatly between the lowest and highest likelihood cases and following this it was decided that it was most appropriate to use the population sizes for the highest likelihood case.

Population levels were then defined using the highest likelihood case, where each population level causes a specific shift in the likelihood level. It was also shown that any non-singleton population level results in a negligible likelihood for the lowest likelihood case.

Using all this, fuzzy SSM rules were formed. For the lowest likelihood case the rules formed were that population sizes less than four should be modelled as individual assets within SSM and that any non-singleton population level results in a negligible likelihood. For the highest likelihood case the rules formed were given in the form of a lookup table, Table 5.

These rules were then put in terms of the SSM core model. For this, new node types representing the different population behaviour types were introduced, as was a new control type. SSM core model rules for threat causes, control effect and threat effects were then given. Finally, a worked example was presented that demonstrated the differences within SSM when moving from a singleton asset to an asset population, how the threat effect paths differ for populations, and how the controls for populations affect the threat effect paths too.

4. Threat Diagnosis, Risk Analysis & Recommendations

This section describes updates to the threat diagnosis and risk analysis components. The general principles of Threat Diagnosis and Risk Analysis have not changed since D7.1. This section contains a reprise of the material from D7.1 with adjustments highlighted (in **bold**) that reflect recent developments. These developments are mainly to support recommendations (which are discussed in detail in Section 4.3). The recommendations illustrate controls for lowering the risk on key assets in the system under evaluation and are accompanied by the expected risk level, should those controls be applied in the real system.

4.1. Positioning within FogProtect Architecture

The Threat Diagnosis and Risk Analysis components form the basis of the runtime risk assessment work of WP7. The components described in this section are highlighted in Figure 33, which is a reprise of the FogProtect architecture picture from D3.2.

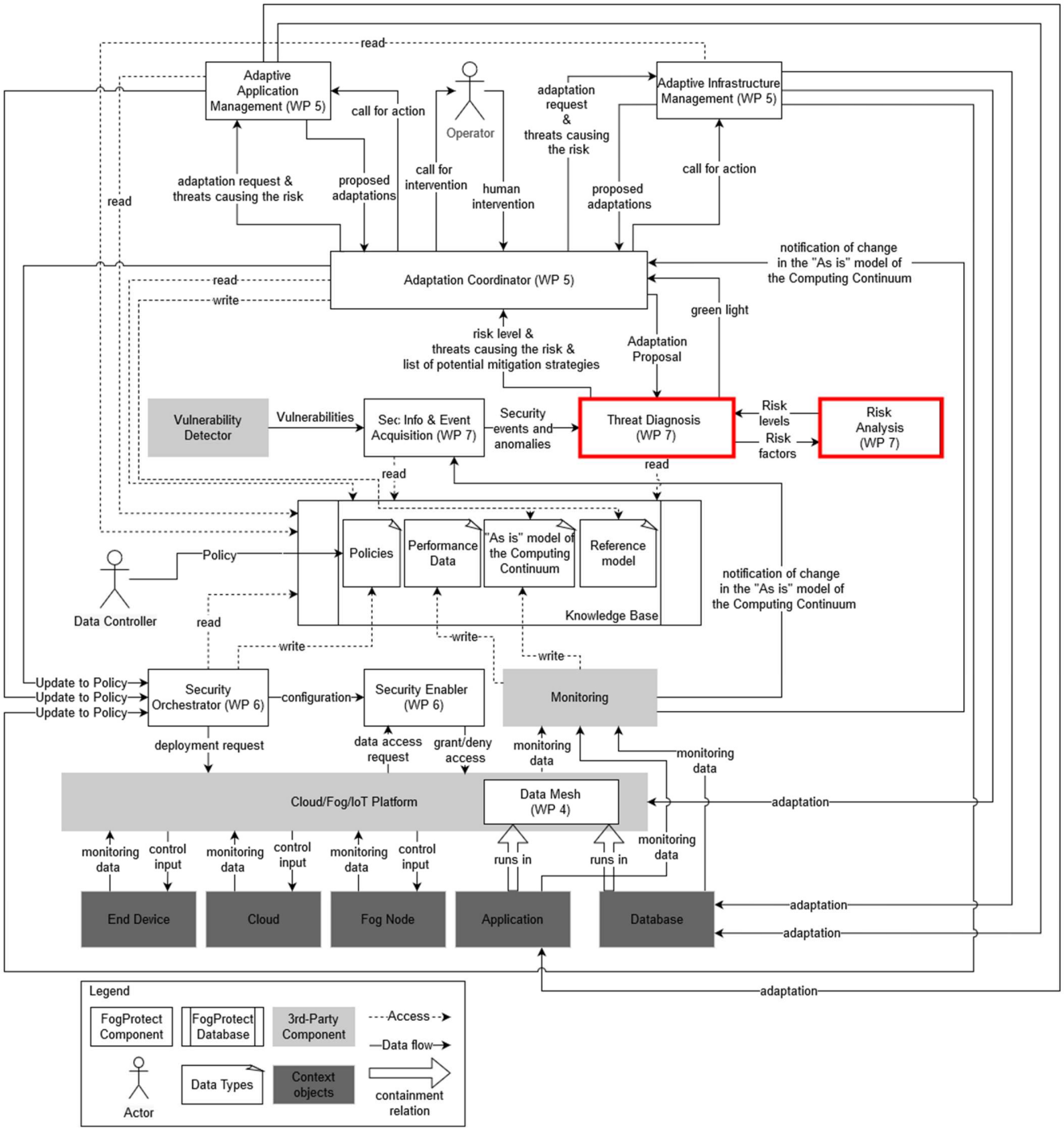


Figure 33: Focus of WP7 Threat Diagnosis and Risk Analysis

The Threat Diagnosis and Risk Analysis components together perform design-time and runtime risk assessment, and this concept is unchanged from D7.1.

4.2. Threat Diagnosis

The Threat Diagnosis component's purpose is to manage requests from other FogProtect components to evaluate the risks caused by different situations **and to propose recommendations to mitigate identified threats**. The Threat Diagnosis component exposes a REST API interface for the risk modelling component discussed above, that all other components can interact with. The main components that interact with the Threat Diagnosis are the SIEA and the Adaptation Coordinator as discussed later.

The architecture of the Threat Diagnosis component is shown in Figure 34. Development of this component is shared with H2020 ProTego, and the shared subcomponents within the Threat Diagnosis are indicated in orange. These are core subcomponents that are common to both projects. The blue subcomponents represent those that are dedicated to FogProtect.

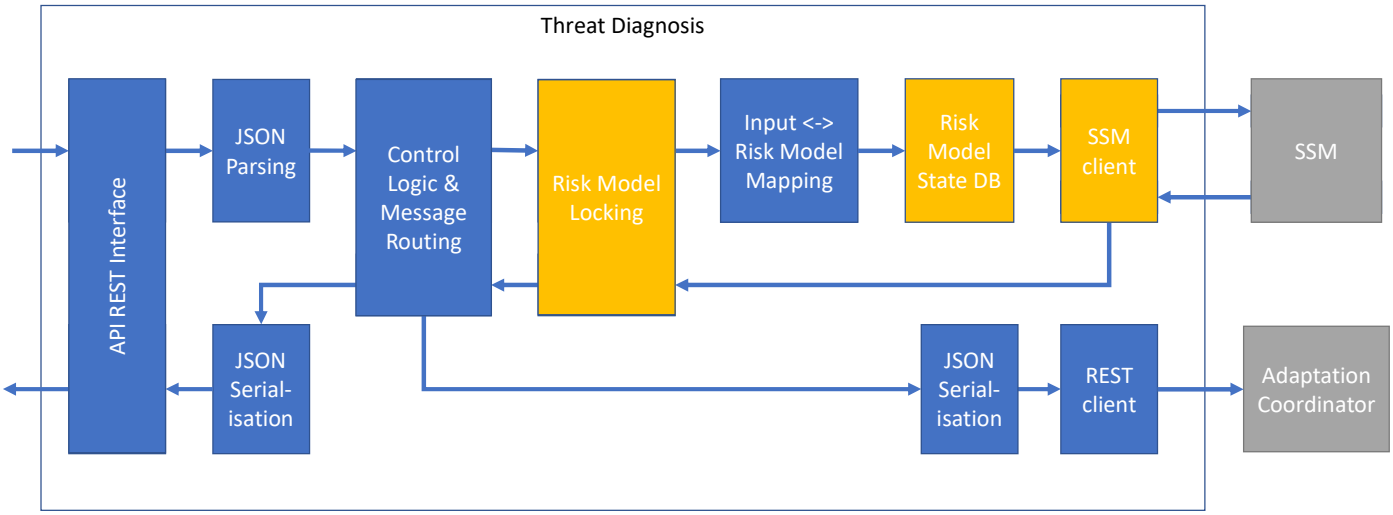


Figure 34: Threat Diagnosis

The Threat Diagnosis component exposes a REST / JSON API interface, which is its main integration point with other components. It exposes four main methods, which are discussed in detail later, but are introduced here.

The main FogProtect endpoints are the same as in period 1 (except with the addition of a "/v2"). The main differences in the API are in the JSON requests and responses, that have been updated in order to better support period 2 use cases.

- ***POST /api/v2/models/{modelId}/notify-immediate-action-event***
  - This is a notification of an event requiring immediate action. An example is in the UC2 smart manufacturing use case, where it is detected that the config file of the process controlling the safety of a manufacturing robot has been tampered with, it is necessary for an immediate action here to shut down the robot because the safety process cannot be relied upon and the robot needs to fail safe.
- ***POST /api/v2/models/{modelId}/evaluate-event-risks***
  - This is a request to evaluate risks due to an event **and propose recommendations for mitigating these risks**
- ***POST /api/v2/models/{modelId}/evaluate-adaptation-proposal-risks***
  - The Adaptation Coordinator computes proposals for adaptations to the system under test and this is a request for evaluation of the risks for each adaptation proposal.
- ***POST /api/v2/models/{modelId}/notify-adaptation-executed***

- This is a notification that an adaptation has been executed. Risk levels are recalculated after the adaptation has been applied.

Custom JSON parsers and serialisers have been created to translate JSON input request messages to objects, and response objects to JSON response messages. The JSON structures/formats have been agreed with WP5 for adaptation proposals and T7.3 for the events notified by the SIEA. These are illustrated by example later, in the section on the API and methods for the Threat Diagnosis component.

The Control Logic subcomponent determines how to process the input requests, and the action is determined by the REST method executed, as described below.

- ***POST /api/v2/models/{modelId}/notify-immediate-action-event***
  - Called by the SIEA. Because this is an 'immediate action' request, Threat Diagnosis initially forwards this message on to the Adaptation Coordinator for immediate action. After this, Threat Diagnosis simply updates the SSM model according to the incoming vulnerability, but otherwise doesn't perform any further processing (i.e. risk calculation).
- ***POST /api/v2/models/{modelId}/evaluate-event-risks***
  - Events are detected by the SIEA and sent to the Threat Diagnosis via a REST / JSON request. The risk level **and recommendations** are calculated and sent to the Adaptation Coordinator so that it can determine adaptations to address the risks.
- ***POST /api/v2/models/{modelId}/evaluate-adaptation-proposal-risks***
  - Called by the Adaptation Coordinator, Threat Diagnosis determines the risk level for each adaptation proposal and sends the results back to the Adaptation Coordinator as a separate request.
- ***POST /api/v2/models/{modelId}/notify-adaptation-executed***
  - Also called by the Adaptation Coordinator. The Threat Diagnosis component updates the risk model to reflect the current state of the system, runs the risk calculation to determine the current working risk level of the system under test and forwards the results to the Adaptation Coordinator.

The Threat Diagnosis component operates asynchronously, in that it returns a response message to the calling client immediately after the request then continues to execute the requested processing in a separate thread. This asynchronous behaviour is managed by the Control Logic subcomponent and is for two reasons: firstly, some of the operations are relatively long-lived (for example the risk calculation **(with recommendations),** which can take minutes depending on the size of the risk model), and the request should not be blocked for this amount of time; and secondly, in many cases, the operation requires invoking a component that is not the calling client (for example the SIEA sends an event to the Threat Diagnosis, the Threat Diagnosis returns an acknowledgement to the SIEA, calculates the risk level and forwards the results to the Adaptation Coordinator). Each request acknowledgement has a job ID returned that can be used optionally by the calling client to check the status of the request processing and to retrieve the results when the processing is complete. This auxiliary REST API call is as follows **(N.B. this endpoint has been updated since period 1 to also include the modelId)**:

- ***POST /api/v2/models/{modelId}/jobs/{jobid}***
  - Called by a client component (e.g., SIEA or Adaptation Coordinator) to get the status of a previously submitted request.

The Risk Model Locking subcomponent implements transactional locking on the risk model. This is necessary to prevent updates to the risk model while other operations (e.g. risk calculation) are taking place. Should a request arrive to update the risk model while a risk calculation is taking place, the calling client will receive an HTTP status code of 423 ('resource locked') and must retry later.

The Input <-> Risk Model Mapping subcomponent handles the mapping between different input specifications and the risk model. For example, a mapping between the 'as-is' model of WP5 and the risk model has been agreed and this mapping is encoded in this subcomponent. Different mappings are possible depending on the type of input, and examples are illustrated later in the API section.

The Risk Model State Database subcomponent is responsible for keeping track of the changes made to the risk model and for rolling them back. Rolling back changes to the risk model is necessary when adaptation proposals are evaluated for risk. The Threat Diagnosis component applies changes to the risk model for each proposal, calculates the risk level, and then rolls back the changes so as to return the risk model to the current working state. This process occurs for as many adaptation proposals are in the request from the Adaptation Coordinator, but at the end of the evaluation, the risk model is always returned to its current working state.

Communication with the System Security Modeller (SSM) that holds the risk model is performed via a REST interface. The SSM exposes a REST API and the Threat Diagnosis has a built-in REST client to execute requests (e.g. to run a risk calculation) via this interface and to receive and parse the responses (e.g. to parse the risk vector response from a risk calculation).

The Threat Diagnosis has an additional REST client for executing requests onto the Adaptation Coordinator. This is used when the results of risk calculations need to be passed to the Adaptation Coordinator.

4.2.1. Threat Diagnosis Methods API and Functionality

The details of the REST API methods are discussed in this section. All requests to the Threat Diagnosis contain a modelID parameter. This identifies the risk model on which to apply the request. At the current time, there is one model per use case scenario, and the ID is passed out-of-band to the components that communicate with the Threat Diagnosis.

For the second phase of the project, the REST API has not changed drastically (although the internal functionality has been improved, including performance enhancements and adding recommendations support); the endpoints are the same, with the addition of a "v2" in each URL. The incoming JSON request format has changed with the addition of a "SieaTaskId" in each request. For event requests, the format has changed to include vulnerability information directly from the vulnerability detector. Risk results now include recommendations for how to block identified threats.

4.2.1.1 Notify Immediate Action

POST /api/v2/models/{modelId}/notify-immediate-action-event

This method is used by the SIEA to indicate that an event that requires an immediate action has occurred. The initial action of the Threat Diagnosis is to pass the event immediately onto the Adaptation Coordinator. After this, Threat Diagnosis simply updates the SSM model **according to the incoming vulnerability**, but otherwise doesn't perform any further processing (i.e. risk calculation).

An example request JSON is shown below **(N.B. format and example content has changed since period 1)**:

```
{
    "SieaTaskId": "1-33aa0b70-3dad-4c5c-969a-2907c27ce45e",
    "ChangesMadeToAsIsModel": [
            {
                "ObjectToIdentify": {
```

```
                    "name": "Attacker_WriteDataFlow",
                    "type": "fogprotect.adaptation.ComputingContinuumModel.WriteDataFlow",
                    "atid": "//@tosca_nodes_root.52"
                },
                "Changes": [
                    {
                        "ChangeType": "CHANGE",
                        "AttributeChanged": "disab",
                        "AttributeType": "EBoolean",
                        "AttributeOldValue": "true",
                        "AttributeNewValue": "false"
                    }
                ]
            }
        ],
    "EventName": "DataManipulationDetected",
    "Timestamp": 1658914511590,
    "Vulnerabilities":
        {
            "rule": "550",
            "agent": {
                "id": "001",
                "name": "green-ibis",
                "ip": "192.168.1.250"
            },
            "reason": "Sensitive data could have been read, modified and redirected by an unauthori
zed person",
            "filename": "app_config.py",
            "source": "wazuh",
            "serviceLevel": "high"
        }
}
```


'SieaTaskId' identifies a single session of the overall FogProtect workflow/sequence and is therefore used in every request related to that instance of the sequence. 'EventName' identifies the specific event in the Fogprotect use case. Here, it is '**DataManipulationDetected'**, which indicates that a **config file has been tampered with or corrupted**.

The 'Vulnerabilities' section describes the specific vulnerability in more detail, using data directly obtained from a specific vulnerability detector – in this case Wazuh (see the 'source' field). In this example, the tampered file ('filename' field) is named 'app_config.py'. The 'serviceLevel' indicates the severity of the vulnerability – in this case 'high'.

The Threat Diagnosis component extracts key fields from the event request (specifically the 'EventName' and 'filename' field values) and uses these to query the SSM risk model for the corresponding trustworthiness attributes (TWAs) to modify (N.B. there may be one or more TWAs on various assets in the risk model). These TWAs are identified in the SSM model configuration phase (prior to the system going "live"); this is achieved using manually added metadata (a.k.a. "Additional Properties") on the assets that exhibit these TWAs. This metadata is in the form of key/value pairs, where the key relates to an expected event and the value specifies a TWA to modify and the value to set. There may of course be several key/value pairs for different expected events.

An example of key/value pairs on SSM model assets is shown below:

"Config File" asset has:

- **Key = DataLeakDetected:app-config.py, Value = Confidentiality:Low**

"User" asset has:

- **Key = DataLeakDetected:app-config.py, Value = Benevolence:Medium**
- **Key = DataLeakDetected:app-config.py, Value = Reliability:Medium**

In the above example, this indicates that when the Threat Diagnosis component receives an 'EventName' and 'filename' of 'DataLeakDetected' and 'app-config.py' respectively, it queries the SSM model for the asset(s) containing the key 'DataLeakDetected:app-config.py'. Using the returned metadata values for these assets, the Threat Diagnosis component determines that it must set the "Confidentiality" TWA to "Low" on the "Config File" asset and the "Benevolence" and "Reliability" TWAs to "Medium" on the "User" asset. These specific TWA updates are then carried out via the related REST API call on the SSM.

The immediate action request also contains a 'ChangesMadeToAsIsModel' section, which lists the changes made to the WP5 'as-is' model, due to the vulnerability. Within this, the 'ObjectToIdentify' attribute specifies the affected object in the as-is model. Normally the 'name', 'type' and 'atid' of the object are agreed between WP5 and WP7 so the as-is model and the risk model share the same identifiers (metadata). However, in this particular example, there is no equivalent asset in the SSM model and the vulnerable asset TWAs have already been defined in the 'Vulnerabilities' section (see above).

The processing inside the Threat Diagnosis component is shown in Figure 35. There are three distinct elements of processing:

- The red line indicates the synchronous request and response to the calling client (the SIEA)
- The yellow line indicates asynchronous processing to immediately pass the event onto the Adaptation Coordinator.
- The green line indicates asynchronous update of the risk model to keep it up to date based on the event.

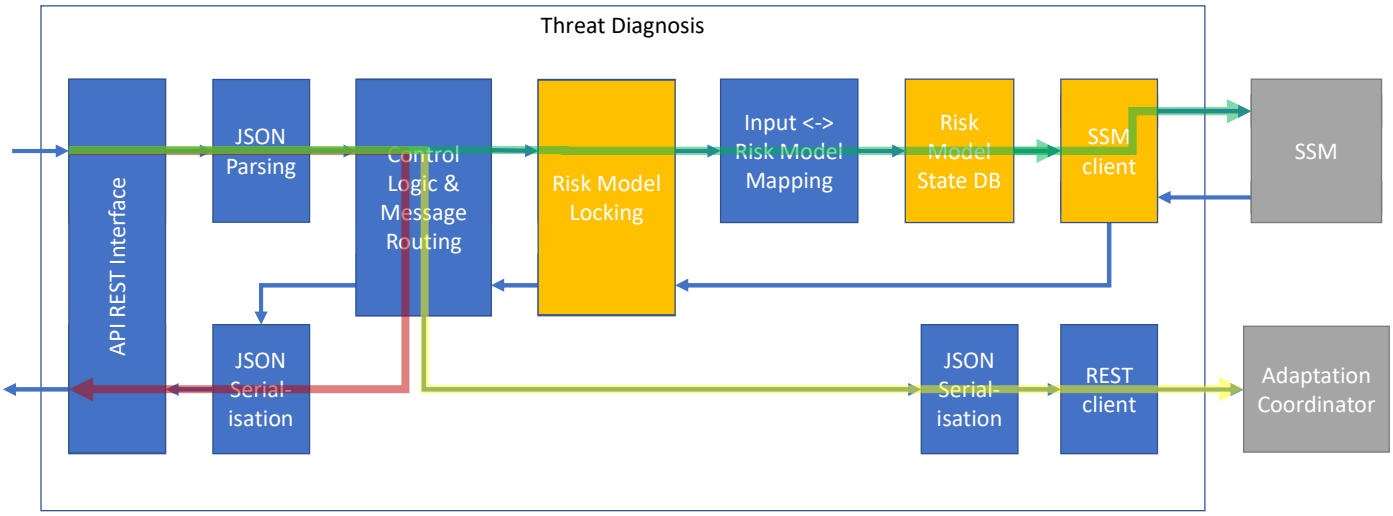


Figure 35: Immediate Action Event Processing

This event does not require immediate risk calculation because the Adaptation Coordinator has already been notified of the **'DataManipulationDetected'** event and has initiated the corresponding emergency adaptation, but the risk model must be updated with the event consequences, to keep it current; this is the TWAS level change, as described above.

The synchronous responses are:

- 200 Success – returned when the event is parsed and the pass-through to the Adaptation Coordinator is initiated
- 422 Validation Error – returned when the event is not parsed successfully.

4.2.1.2 Evaluate Event Risks

POST /api/v2/models/{modelId}/evaluate-event-risks

This method is called by the SIEA and is a request to evaluate the risks due to an event occurring. The risk level **and recommendations** are calculated and sent to the Adaptation Coordinator so that it can determine adaptations to address the risks.

An example JSON request body from UC2 smart manufacturing is as follows **(N.B. format and example content has changed since period 1)**:


```
{
    "SieaTaskId": "1-33aa0b70-3dad-4c5c-969a-2907c27ce45e",
    "ChangesMadeToAsIsModel": [
        {
            "ObjectToIdentify": {
                "name": "Attacker_ReadDataFlow",
                "type": "fogprotect.adaptation.ComputingContinuumModel.ReadDataFlow",
                "atid": "//@tosca_nodes_root.53"
            },
            "Changes": [
                {
                    "ChangeType": "CHANGE",
                    "AttributeChanged": "disab",
                    "AttributeType": "Eboolean",
                    "AttributeOldValue": "true",
                    "AttributeNewValue": "false"
                }
            ]
        }
    ],
    "EventName": "DataLeakDetected",
    "Timestamp": 1658914511590,
    "Vulnerabilities":
        {
            "rule": "550",
            "agent": {
                "id": "001",
                "name": "green-ibis",
                "ip": "192.168.1.250"
```

```
            },
            "reason": "Sensitive data could have been read by an unauthorized person",
            "filename": "app_config.py",
            "source": "wazuh",
            "serviceLevel": "medium"
        }
}
```

The request body is similar to the immediate action event described above but there are notable differences. The 'EventName' here is '**DataLeakDetected'**, which indicates that the **config file has been read by an unauthorised party**. **This is less severe than the 'DataManipulationDetected' event, hence the 'serviceLevel' defined in the vulnerability is only 'medium'. The Threat Diagnosis component maps this to a 'Low' trustworthiness level of the corresponding trustworthiness attribute set (TWAS) on the ConfigFile**.

This is an 'Evaluate Event Risks' request, which instructs the Threat Diagnosis to initiate a risk calculation based on the updated SSM model (i.e. adjusted TWAS level described above), and to send the results onward to the Adaptation Coordinator **along with recommendations for risk mitigations**.

This method's operational sequence shown in Figure 36 and is as follows:

1. Return a status code to the calling client (red line in Figure 36).
2. Lock the risk model identified in the request path. This means that any requests to update the model before the lock is released will result in a 'resource busy' response and the model will be unaffected by these requests.
3. Asynchronously evaluate risks **and recommendations**, then send the results to the Adaptation Coordinator (green line in Figure 36).
    a. Map the event type and affected objects to the risk model. This results in the list of updates to be made to the risk model.
    b. Record the risk model updates in the state DB.
    c. Update the risk model in the SSM with the updates.
    d. Run the risk calculation on the updated risk model, **then calculate recommendations**.
    e. When this is complete:
        i. Release the lock on the model.
        ii. Send the results to the Adaptation Coordinator using the REST client to execute the '/fogprotect/adaptationcoordinator/notify' endpoint with a JSON body containing the risk vector, **top risks and a list of recommendations**.

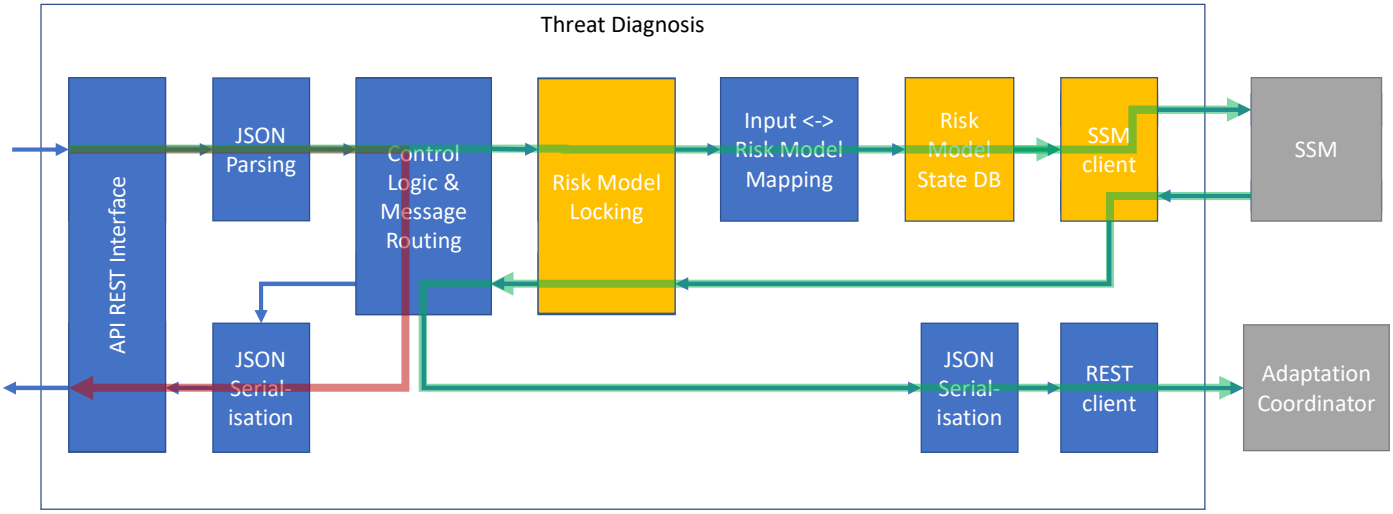


Figure 36: Evaluate Event Risks

The risk calculation results in a set of risks on specific assets in the risk model, along with a 'risk vector' and overall risk level. **We now also include a set of recommendations for addressing the identified risks.** These results are forwarded to the Adaptation Controller, and an example of the JSON message structure is given below, and recommendations are discussed in Section 4.3.


```
{
    "SieaTaskId": "1-33aa0b70-3dad-4c5c-969a-2907c27ce45e",
    "NotificationType": "ResultOfRiskCalculation",
    "Risks": [
        {
            "ObjectToIdentify": {
                "ObjectName": "FiaB Shipping Container",
                "ObjectType": "PrivateSpace",
                "ObjectID": "//@tosca_nodes_root.31"
            },
            "RiskName": "Loss of Control",
            "RiskDescription": "The asset is no longer controlled only by appropriate and authorize
d stakeholders.",
            "RiskImpact": "High",
            "RiskLikelihood": "Medium",
            "RiskLevel": "High"
        }
    ],
    "OverallRiskLevel": "Very High",
    "RiskVector": {
        "VeryHigh": 1,
        "High": 10,
        "Medium": 34,
        "Low": 40,
        "VeryLow": 14
    },
```

```
    "AcceptableRiskLevel": "Medium"
}
```

The structure contains an array of Risks, which are identified objects in the as-is model and the specific risk on each. The example shows a single risk, on the 'FiaB Shipping Container' asset, which is of the type 'PrivateSpace'. The risk is a 'Loss of Control', which occurs when 'The asset is no longer controlled only by appropriate and authorised stakeholders'. The risk impact is set to 'High' and the likelihood is calculated to be 'Medium', resulting in an overall risk level of 'High'.

The risk vector is determined from the risk levels of all identified risks on all assets. The risk vector here has the distribution shown above and represents counts of all risks having the specific risk level. The overall risk is the worst-case risk level – here it is 'Very High' because there is at least one risk at that level.

The synchronous HTTP response codes for this method are as follows.

- 202 – request is accepted, and processing has been initiated but it is not yet complete. This is the normal 'success' response.
- 404 – model ID or asset ID not found. If the model ID in the HTTP request is invalid, 404 is returned.
- 422 – validation error. If the JSON request body does not parse successfully, 422 is returned.
- 423 – resource locked. The model is being updated in answer to a previous request, so try again later.

4.2.1.3 Evaluate Adaptation Proposal Risks

POST/api/v2/models/{modelId}/evaluate-adaptation-proposal-risks

This method is invoked by the Adaptation Coordinator and its action is to evaluate the risk levels for a set of adaptation proposals specified in the JSON request body. There may be one or more adaptation proposals, and each is evaluated individually.

An example of the JSON request format is shown below.

```
{
    "SieaTaskId": "1-33aa0b70-3dad-4c5c-969a-2907c27ce45e",
    "AsIsModel": {
       …
    },
    "AdaptationProposals": [
        {
            "ID": 1,
            "AdaptationProposalModel": {
                "type": "CloudEnvironment",
                "atid": "//",
                "tosca_nodes_root": [...]
            },
            "ChangesMadeToAsIsModel": [
                {
                    "ObjectToIdentify": {
                        "name": "ReadDataFlow (User-NormalDB)",
                        "type": "fogprotect.adaptation.ComputingContinuumModel.ReadDataFlow",
                        "atid": "//@tosca_nodes_root.16"
```

```
                    },
                    "Changes": [
                        {
                            "ChangeType": "CHANGE",
                            "AttributeChanged": "disab",
                            "AttributeType": "Eboolean",
                            "AttributeOldValue": "false",
                            "AttributeNewValue": "true"
                        }
                    ]
                }
            ]
        }
    ]
}
```


The request body firstly contains the entire as-is model ('AsIsModel'), but this is not currently used for risk calculation and is therefore not shown here, for clarity. This is followed by an array of adaptation proposals; one of these is shown in the example. An adaptation proposal contains the updated as-is model (i.e. 'AdaptationProposalModel') and the changes to it for the adaptation, compared to the current as-is model (i.e. 'AsIsModel'); the changes are used to modify the SSM model, prior to the risk calculation.

One change is shown in the example. The change contains the standard identifying attributes for the as-is model entity (which are mapped to SSM risk model assets), as well as details of the changes to that entity. This example shows that data flow entity (between the User and the NormalDB) is disabled as a result of this adaptation; this is indicated by the elements "AttributeChanged": "disab", "AttributeOldValue": "false" and "AttributeNewValue": "true". These are reflected into the SSM risk model by identifying the appropriate data flow (using the shared identifiers "atid": "//@tosca_nodes_root.16", etc), and applying a control that disables the data link. This is defined as follows.

- ***Disabled:*** Means the asset is disabled by a suitable system reconfiguration so the asset cannot be used. This addresses threats involving the asset but may lead to different threats caused by a loss of availability in the disabled asset.

The processing for this method is shown in Figure 37.

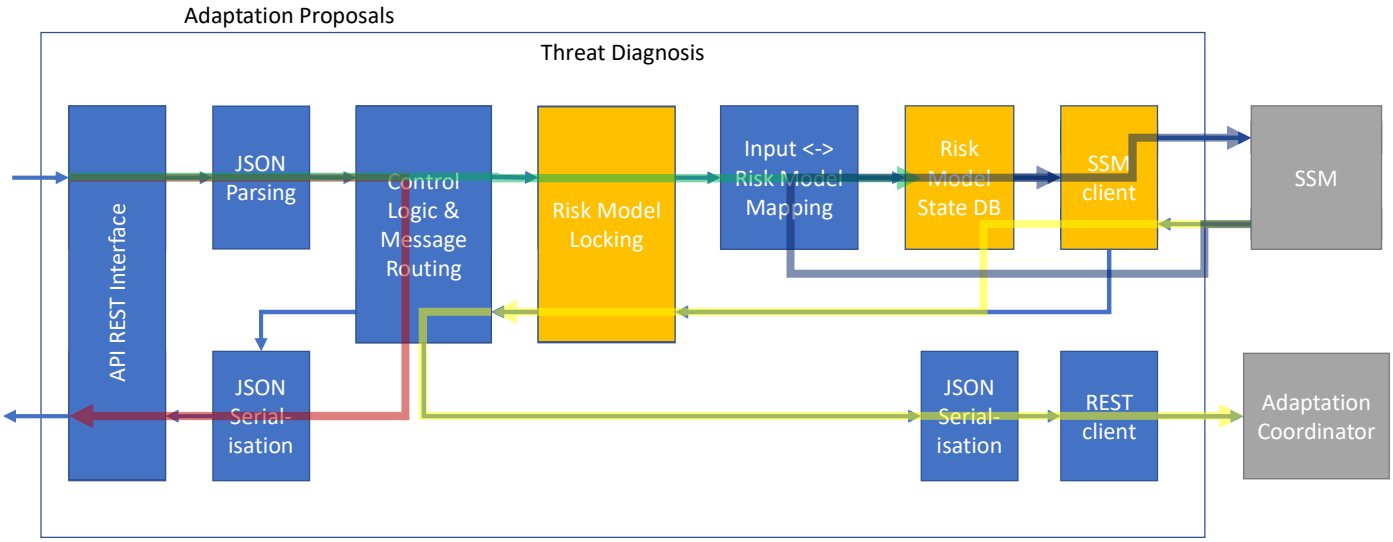


Figure 37: Evaluate Adaptation Proposal Risks

The operation is similar to the 'evaluate event risks' method in that a risk calculation is run, but this method runs the risk calculation for each adaptation proposal separately. **N.B. This method does not generate recommendations for each adaptation proposal, as this is unnecessary.** The overall processing sequence is as follows.

1. Return a status code to the calling client (red line in Figure 37). Here the calling client is the Adaptation Coordinator.
2. Lock the risk model identified in the request path.
3. Iterate over each adaptation proposal (purple loop in Figure 37). For each adaptation proposal:
    a. Map the changes and affected objects in the adaptation proposal to the risk model. This results in the list of updates to be made to the risk model.
    b. Record the risk model updates in the state DB.
    c. Update the risk model in the SSM with the updates.
    d. Run the risk calculation on the updated risk model.
    e. When the risk calculation is complete, add its results to the output format to be sent to the adaptation coordinator.
    f. Revert the changes to the risk model for this adaptation proposal.
4. Release the lock on the model (yellow line in Figure 37).
5. Send the risk calculation results for all adaptation proposals to the Adaptation Coordinator using the REST client by executing the '/fogprotect/adaptationcoordinator/notify' endpoint with a JSON body containing the risk results for each adaptation proposal (also yellow line in Figure 37).

After all the adaptation proposals have been evaluated, the state of the risk model is the same as it was before any of the adaptation proposals were evaluated. This is because adaptation proposals are as their name suggests – they are merely proposals, and they need to be evaluated for risk without changing the state of the real system. Therefore, the risk model updates for each proposal are temporary only.

The result format is as follows **(N.B. this is not the actual result, but an example that shows the general principles).**


```
{
  "SieaTaskId": "1-33aa0b70-3dad-4c5c-969a-2907c27ce45e",
```

```
  "NotificationType": "EvaluationOfAdaptationProposals",
  "AsIsRisk": {
     "@id": "/",
     "RiskLevel": {
      "OverallRiskLevel": "High",
      "RiskVector": {
       "Very High": 0,
       "High": 4,
       "Medium": 50,
       "Low": 16,
       "Very Low": 34
      }
    }
  },
  "AcceptableRiskLevel": "Medium",
  "AdaptationRisks": [
     {
      "AdaptationProposalId": "1",
      "RiskLevel": {
       "OverallRiskLevel": "Medium",
       "RiskVector": {
       "Very High": 0,
          "High": 0,
          "Medium": 32,
          "Low": 10,
          "Very Low": 17
       }
      }
     }
     {
      "AdaptationProposalId": "2",
      "RiskLevel": {
       "OverallRiskLevel": "Very High",
       "RiskVector": {
       "Very High": 2,
          "High": 16,
          "Medium": 12,
          "Low": 45,
          "Very Low": 8
       }
      }
     }
  ]
}
```


The current as-is overall risk level is given together with its risk vector. This gives a point of comparison with each of the adaptation proposals, so the Adaptation Coordinator can determine whether each proposal is lower risk than the current status quo. Next, the acceptable risk level is given. This is based on the user's level of risk acceptability. Following this, level vectors for each of the evaluated adaptation proposal are presented in an array. The example shows two adaptation proposals, of which the first is lower overall risk, and is acceptable.

The synchronous HTTP response codes for this method (red line in Figure 37) are as follows.

- 202 – request is accepted and processing has been initiated but it is not yet complete. This is the normal 'success' response.
- 404 – model ID or asset ID not found. If the model ID in the HTTP request is invalid, 404 is returned.
- 422 – validation error. If the JSON request body does not parse successfully, 422 is returned.
- 423 – resource locked. The model is being updated in answer to a previous request, so try again later.

4.2.1.4 Notify Adaption Executed

POST/api/v2/models/{modelId}/notify-adaptation-executed

This method notifies Threat Diagnosis that an adaptation proposal has been executed. The executed adaptation is the decision of the Adaptation Coordinator and is the selection of one of the adaptation proposals evaluated previously. The execution represents a change of state to the real system under test, so the risk model can be kept up to date with the actual system under test.

An example of the JSON request body is shown below.


```
{
    “SieaTaskId”: “1-33aa0b70-3dad-4c5c-969a-2907c27ce45e”,
    "NotificationType": "AdaptationExecuted",
    "AsIs": {
        "AsIsModel": {…},
        "OldAsIsModel": {…},
        "ChangesMadeToAsIsModel": [
            {
                "ObjectToIdentify": {
                    "name": "ReadDataFlow (User-NormalDB)",
                    "type": "fogprotect.adaptation.ComputingContinuumModel.ReadDataFlow",
                    "atid": "//@tosca_nodes_root.16"
                },
                "Changes": [
                    {
                        "ChangeType": "CHANGE",
                        "AttributeChanged": "disab",
                        "AttributeType": "EBoolean",
                        "AttributeOldValue": "false",
                        "AttributeNewValue": "true"
                    }
                ]
            }
        ],
        "Risks": []
    },
    "Adaptations": []
}
```


The format of the request body is:

- The current as-is model – this is the complete as-is model (not shown here for reasons of space).
- The previous as-is model (not shown for reasons of space).
- The changes made to transform the previous as-is model to the current. This is the key element relevant to this method. The changes are in exactly the same format as an adaptation proposal, which is logical as the changes represent the adaptation proposal selected by the Adaptation Coordinator.
- Optional risks and adaptations (not used).

The calling client for this method is the SIEA, which forwards the JSON request body on from an event detected on the Kafka queue that is generated when the Adaptation Coordinator has finished executing its chosen adaptation.

The operation of the method is to update the risk model with the changes in the JSON request body, evaluate the risks of the change and forward the risk level onto the Adaptation Coordinator. There is no reversion of changes in this operation, as it reflects a real change to the working system under test. **N.B. This method does not calculate recommendations, as this is simply updating the system after a chosen adaptation.**

The control flow for this method is as shown in Figure 38.

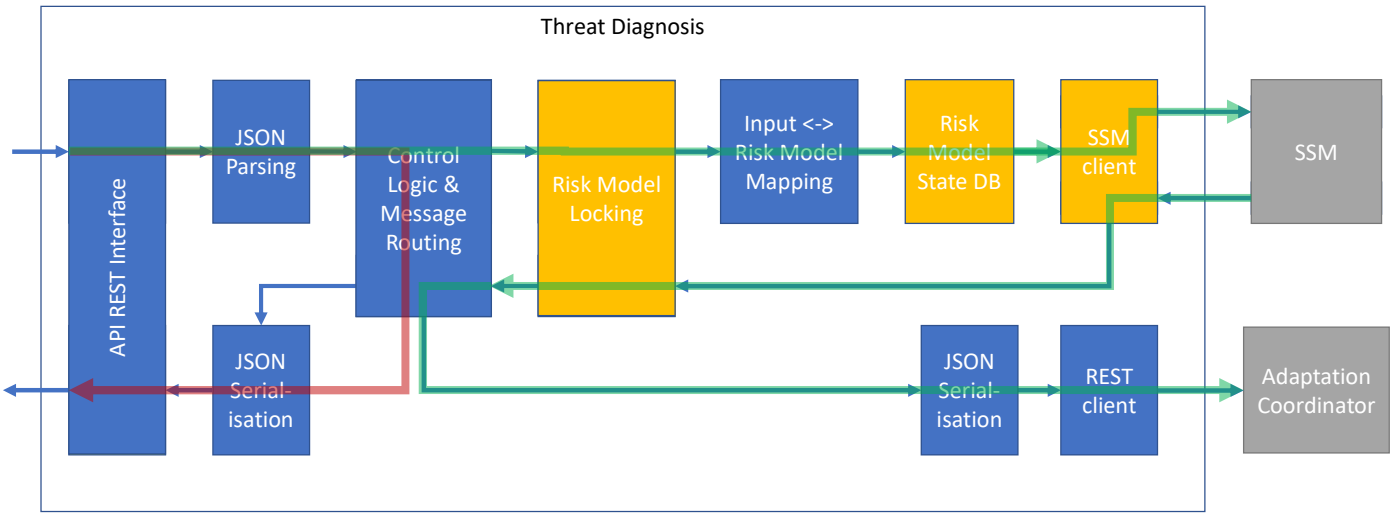


Figure 38: Notify Adaptation Executed

The operation of the method is as follows.

1. Return a status code to the calling client (red line in Figure 38). Here the calling client is the SIEA.
2. Lock the risk model identified in the request path.
3. Evaluate the risk level of the executed adaptation. (green line in Figure 38):
    a. Map the changes and affected objects in the adaptation description to the risk model. This results in the list of updates to be made to the risk model.
    b. Record the risk model updates in the state DB.
    c. Update the risk model in the SSM with the updates.
    d. Run the risk calculation on the updated risk model.
4. Release the lock on the model.
5. Send the risk calculation results for executed adaptation to the Adaptation Coordinator using the REST client by executing the '/fogprotect/adaptationcoordinator/notify ' endpoint with a JSON body containing the risks and risk vector.

The result format for the risk level results has the type 'ResultOfRiskCalculation' and is the same as for the 'evaluate event risks' method **(except that it does not include further recommendations)**. This is because the adaptation execution is a change of system state, as is the event, so both require the evaluation of the risks of the current system. The result format therefore has the form:

- Risks on assets within the as-is and risk model, each with an identified asset, risk name, description, impact level, likelihood and risk level.
- Overall risk level and risk vector.
- Acceptable risk level.

An example is given below, showing a loss of confidentiality risk on data from the UC2 smart manufacturing scenario. In an actual result message, there will be many risks, but one is shown here for brevity.


```
{
   "SieaTaskId": "1-33aa0b70-3dad-4c5c-969a-2907c27ce45e",
   "NotificationType": "ResultOfRiskCalculation"
   "Risks": [
      {
         "ObjectToIdentify":
         {
           "ObjectName": "Scene Data",
           "ObjectType": "Data",
           "ObjectID": "//@tosca_nodes_root.87"
         },
         "RiskName": "Loss of Confidentiality",
         "RiskDescription": "Data (including communications) are accessible and may have been accessed
         without authorization. Applies to data and also some communication channels, devices and
         processes that carry or manage data.",
         "RiskImpact": "High",
         "RiskLikelihood": "Very Low",
         "RiskLevel": "Low"
      }
   ],
   "OverallRiskLevel": "Medium",
   "RiskVector": {
      "VeryHigh": 0,
      "High": 0,
      "Medium": 17,
      "Low": 20,
      "VeryLow": 36
   },
   "AcceptableRiskLevel": "Medium"
}
```


The synchronous HTTP response codes for this method (red line in Figure 37) are as follows.

- 202 – request is accepted and processing has been initiated but it is not yet complete. This is the normal ‘success’ response.
- 404 – model ID or asset ID not found. If the model ID in the HTTP request is invalid, 404 is returned.
- 422 – validation error. If the JSON request body does not parse successfully, 422 is returned.
    - 423 – resource locked. The model is being updated in answer to a previous request, so try again later.

4.3. Recommendations

This section discusses recommendations, which are an output of the risk modelling that accompany a risk analysis. Recommendations provide suggested controls to address raised risk levels, and are determined by analysis of “attack paths”, which are propagations of threats and their effects (misbehaviours), which in turn can cause downstream threats. The recommendations algorithm is trying to identify such attack paths in the system model and suggest possible ways to mitigate adverse effects from those paths.

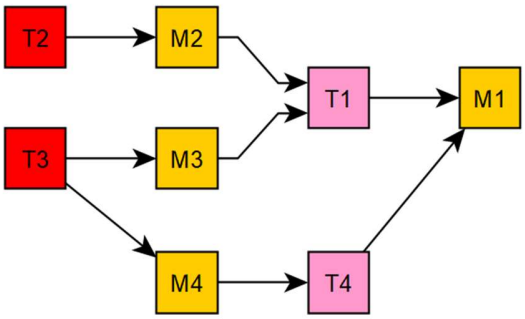


Figure 39. Attack Graph Example

Figure 39 shows how misbehaviours (in yellow) caused by threats (red and pink), which can themselves be caused by upstream misbehaviours. Since misbehaviours can be triggered by multiple threats and multiple misbehaviours can enable threats, attack paths can take a form of an attack graph: both branching and converging, as illustrated in Figure 39. In this Figure misbehaviour M1 can be caused by threats T1 *or* T4; in turn T1 and T4 are caused by other misbehaviours: T1 occurs only if M2 *and* M3 are present, T4 requires just M4; misbehaviours M2, M3 and M4 are caused by "root cause" threats T2 and T3. To reduce the likelihood of misbehaviour M1 occurring, an intervention in the attack graph must be made. In this case, the likelihoods of threats T1 and T4 must be reduced (using appropriate control strategies on those threats, should they exist), or threat T3 must be controlled.

The recommendations algorithm consists of two parts: one that finds the shortest paths from root causes to the adverse effects and the control strategies along those paths that can mitigate the risk of those threats (the "attack graph"), and one that uses the attack graphs to try various control strategy options until the residual risk is acceptable.

The output of the attack graph algorithm is a description of how misbehaviours and threats link together to cause the high-risk misbehaviours we are interested in. Alongside this is a logical expression in disjunctive normal form (DNF) describing the control strategy options to reduce risk, for example:

```
OR (
      Control Strategy 1 (at T3),
      AND(
           Control Strategy 2 (at T1),
           Control Strategy 3 (at T4)
      )
)
```

The above example will provide two recommendations i.e. a) control strategy 1, and b) control strategy 2 and 3 combined. Each of those recommendations will require different implementations and consequently produce different levels of risk and adverse effects.

The implementation of the algorithm has the following steps:

- Find system model misbehaviours[2] that exceed certain risk level of concern, e.g. Medium risk, sorted by their likelihood level.

[2] Recall that "misbehaviour" is synonymous with "adverse effect" and "consequence"

- Based on the identified misbehaviours, build the attack graph following the shortest paths from root causes to their adverse effects, along with a list of control strategy options that will reduce the risk.

- The algorithm then tries each control strategy option in turn, recalculating the risk level and generating a new attack graph for each one. If, as a result of applying the control strategy option, the risk levels of concern are now low enough, then that exploratory branch can be terminated and a potential recommendation made. Otherwise, the control strategy options described by the new attack graph must be explored in a recursive manner.

- The last phase of the algorithm explores and combines all control strategy options, as derived in previous step. These control strategy options also form a graph as Figure 40 shows. Each path in the control strategies graph can become a recommendation providing a model risk reduction. In Figure 40 the possible recommendations will be either [CSG2] alone, [CSG1 + CSG3], or [CSG1 + CSG4] control strategy combinations.

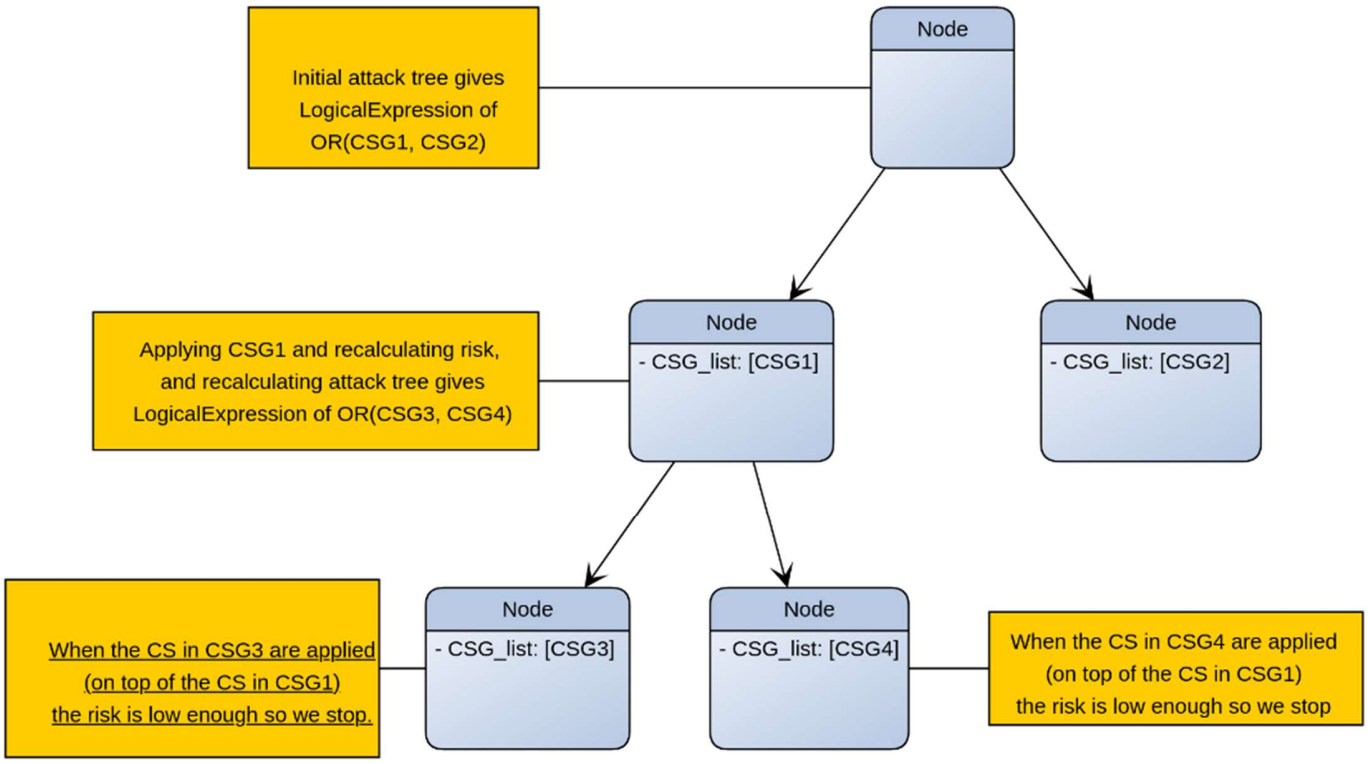


Figure 40 Control strategy options graph example

4.3.1. The Recommendations Output JSON Schema

The recommendations output is a JSON document with the following sections:

- Current state, that includes risk and consequences
- A list of recommendations, each of which includes:

- a list of control strategies with their associated controls,
- as well as the resulting risk level when such controls are applied.

The following section shows a pseudocode recommendation output document schema:

```
{
  "current": { // current risk level
    "state": {
      "risk": {
        "overall": "string",
        "components": { "veryHigh": 0, "high": 0, "medium": 0, "low": 0, "veryLow": 0 }
      },
      "consequences": [
        {
          "asset": {
            "label", "type", "uri", "identifier", "additionalProperties": [ { "key", "value" } ]
          },
          "label", "description", "impact", "likelihood", "risk", "uri"
        }
      ]
    }
  } , // end current risk level
  "recommendations": [ // list of recommendations
    { // recommendation
      "identifier": 0, // recommendation identifier
      "category",
      "controlStrategies": [ // list of control strategies in recommendation
        { "uri", "description" }
      ],
      "controls": [ // list of controls in recommendation
        {
          "label": "string", "uri": "string",
          "asset": {
            "label", "type", "uri", "identifier", "additionalProperties": [ { "key", "value" } ]
          },
          "action": "string"
        }
      ],
      "state": { // expected result of recommendation in terms of expected risk level and consequences
         "risk": {
           "overall": "string",
           "components": { "veryHigh": 0, "high": 0, "medium": 0, "low": 0, "veryLow": 0 }
        },
         "consequences": [
           {
             "asset": {
               "label", "type", "uri", "identifier", "additionalProperties": [ { "key", "value" } ]
             },
             "label", "description", "impact", "likelihood", "risk", "uri"
           }
         ]
      }
    } // end recommendation
  ] // end list of recommendations
}
```

4.3.2. FogProtect UseCase 2 Recommendation Example

This section presents an actual recommendation for the FogProtect Use Case 2. This example will be fully described in the upcoming D2.4 as it forms the basis of the enhanced smart manufacturing use case, but brief notes are presented here to

illustrate the effect of the recommendations. The actual recommendation JSON document is presented in the Appendix: UC2 Example Recommendations JSON, but is briefly described here.

The identified initial risk of the Use Case 2 system model is Very High:


```
"risk": {
     "overall": "Very High",
     "components": {
          "veryHigh": 2,
          "high": 1,
          "medium": 1,
          "low": 720,
          "veryLow": 1361
     }
 }
```


There are adverse effects such as loss of confidentiality, reliability, and integrity on assets User, ABB Robot, Robot QC Data, and Rood Cause Data. These are summarised in the following table and result in a worst case risk of Very High.

Table 9: Example Initial Risks

Asset	Consequence Type	Impact	Likelihood	Risk
Robot QC Data	Loss Of Confidentiality	High	High	Very High
ABB Robot	Loss Of Reliability	High	High	Very High
Root Cause Data	Loss Of Integrity	Medium	High	High
User	Loss Of Reliability	Medium	Medium	Medium

As an example risk, Robot QC Data potentially contains personal information so the impact of a confidentiality loss is high (i.e. a data breach), and the tool has computed its likelihood to be high, resulting in a very high risk, clearly highly undesirable. Further discussion of this, and other risks, will be presented in D2.4.

The recommendations algorithm has found a recommendation that reduces the overall system model risk to Medium, a considerable improvement over the untreated situation above.


```
"risk": {
     "overall": "Medium",
     "components": {
          "veryHigh": 0,
          "high": 0,
          "medium": 2,
          "low": 691,
          "veryLow": 1392
     }
 }
```


The recommendation output describes which control strategies (CSG) should be implemented as well as the required control sets (CS) that need to be applied. The recommended control strategies are:

- AutoSuspendUntrustworthyClientAccess
- SuspendUnreliableController

- AutoSuspendCorrptedDataFlow

The recommended CSGs can be implemented by applying the following control sets:

- DisableHost on ABB Robot
- DisableClientAccess on User Client
- DisableDataFlow on Kafka Server

The result of applying these controls is also given in the recommendation and is summarised in the following table.

Table 10: Example Risks After Recommendations

Asset	Consequence Type	Impact	Likelihood	Risk
User	Loss Of Timeliness	Low	Very High	Medium
Robot QC Data	Loss Of Confidentiality	High	Low	Medium

Other risks are generated by the algorithm but they are “low” or “very low” level, and the current configuration of the algorithm has instructed it to exclude these results, as there will be many of them and they are not significant.

5. Conclusions

This deliverable has described final enhancements to the dynamic risk management components from WP7 that detect events and threats and assess the risk levels due to events and adaptation proposals from the Adaptation Coordinator in WP5.

In Period 2, the upstream architecture to the Threat Diagnosis has changed significantly reflecting experiences from 2021. This has addressed the need for aggregation and filtering of ‘raw’ events became evident, not only to stop flooding the Managing Components of FogProtect, but also the need to differentiate between temporal (and perhaps accidental) behaviour and deliberate misbehaviour. To these ends, several new components have been added:

- A Vulnerability Detector Component, which enabled us to detect and report, in real-time, security-relevant events on hosts located in the cloud. The Vulnerability Detector is deployed in the Smart Cities and in the Smart Manufacturing use-cases.
- A Monitoring Component, which is configured to feed the SIEA with tasks based on their service level. If more than one task must be reported, the highest priority task will be pushed to the SIEA and the lesser priority task will be cached and sent later.
- A database, accessed by both the Monitoring Component and the SIEA which is being used for querying the state of one (and potentially more) SIEA instance. The Monitoring Component can therefore control better the flow of tasks towards the SIEA making the P1 component, the ‘Kafka Latch’, obsolete.

The section on populations modelling has presented the reasoning behind the need for populations in scenarios fog-cloud - i.e. to model the multiplicity of fog nodes efficiently. Two thought-experiment examples were used to show and highlight two different cases of population behaviours, one corresponding to redundancy (relating to availability of redundant resources) and the other to where just a single member of a population causing an effect results in that effect being present in the population as a whole (e.g. where a data leak of a single copy of a data set results in a loss of confidentiality). From these, three different cases of population behaviour were determined: the lowest, highest, and average likelihood cases. With these cases for populations determined, mathematics describing them was then derived to enable their application within the SSM core model. Population sizes were then defined to determine how changes in population

affect likelihood, where each population level causes a specific shift in the likelihood level. This resulted in a lookup table to determine how likelihood (e.g. of a threat is affected by the size of the population. These rules were then put in terms of the SSM core model. For this, new node types representing the different population behaviour types were introduced, as was a new control type to accommodate causes and controls on population assets. Finally, a worked example was presented that demonstrated the differences within SSM when moving from a singleton asset to an asset population, how the threat effect paths differ for populations, and how the controls for populations affect the threat effect paths too.

The Recommendations extension provides suggestions for runtime controls based on examination of the risk model's "attack graph", which is a representation of the threat and risk propagation throughout the modelled system. The SSM can automatically suggest controls to block threats at each point in the attack graph, and the recommendations work has determined an algorithm that analyses the attack graph to determine the optimum controls and control strategies to block the threats. In some cases, a single control strategy early in the attack graph (i.e. at or near its root) will be sufficient to block a whole chain of threats, and in other cases where this may not be possible, the attack graph is traversed to determine controls to block threats at intermediate threats in the graph.

Quoting the DOA, the objectives of WP7 are listed as follows.

1. *To develop a knowledge base describing Fog computing assets, cyber security threats, and control measures including but not limited to those in use case scenarios from WP2.*
2. *To provide tools supporting machine inference from this knowledge to model risks in specific applications and associated business goals running in the computing continuum.*
3. *To develop techniques for security intelligence acquisition and network monitoring in Fog based systems related to factors identified in the knowledge base as determinants of risk.*
4. *To devise new mathematical models for intelligent, dynamic threat diagnosis by combining predictive models with network analytics.*

Essentially the first two objectives relate to the development of knowledge and inference methods to construct predictive models of potential risks prior to deployment. The last two relate to the use of information acquired (and mostly only available) after deployment, to diagnose run-time threats and propose responses to manage the associated risks that can be implemented by tools from WP5.

Deliverable D7.1 described how WP7 has extended previous work to create a dynamic threat detection and risk modelling system. At the time of writing of D7.1, the contributions towards each objective were as follows

- Objective 1 was addressed via knowledge base extensions modelling assets, threats and controls arising from the the fog-cloud situations in the project's use cases, and also informed by the data lifecycle work in WP6, which was reported in D6.1.
- Objective 3 was addressed by the Security Information and Event Acquisition (SIEA), which detects runtime events and sends vulnerabilities to the Threat Diagnosis component.
- Objective 2 was addressed by Risk Analysis, which represents a design-time risk model of the current compute continuum situation. The tooling represented here is the background System Security Modeller toolkit, which was adapted to enable dynamic runtime updates to the risk models running within it (also contributing towards objective 4).
- Objective 4 was addressed by Threat Diagnosis receiving events form the SIEA and adapting static risk models accordingly, so that risk calculations can be rerun to reflect the current risk level

In P2, we have further extended this work via the extensions to the SEIA, the populations modelling work and the support for recommendations to address detected threats. The specific contributions towards the project's objectives are summarised below.

We have further addressed objectives 1 and 2 via:

- Modelling populations of assets (exemplified by but not limited to fog nodes). This is a major step change in functionality for the UoS System Security Modeller (SSM) toolkit, and as such has required knowledge base extensions to include these asset types and associated threats and controls, as well as more fundamental additions and adjustments in the core model and algorithms of the SSM toolkit to accommodate this enhancement.
- Automated recommendation of security controls to block threats.

Objectives 3 and 4 are addressed via:

- Extension of the SEIA to recognise, categorise and aggregate information from network monitoring tools into events that reflect fog-based intrusions or compromises.
- Mapping these events into the SSM system models and trustworthiness attributes.

This deliverable has described extensions to an integrated framework for dynamic (runtime) event detection, threat detection and risk analysis to specifically support situations commonly found in fog-cloud environments. This work is currently being evaluated in WP2 in the project's three use cases, and the results from this evaluation will be reported in the upcoming D2.4, due PM36, where updated risk models constructed to model the enhanced scenarios for each of the project's use cases will be demonstrated.

6. References

[Boniface 2022] Boniface, Michael, Carmichael, Laura, Hall, Wendy, McMahon, James, Pickering, J. Brian, Surridge, Mike, Taylor, Steve, Atmaca, Ugur Ilker, Epiphaniou, Gregory, Maple, Carsten, Murakonda, Sasi, & Weller, Suzanne. (2022). DARE UK PRiAM Project D3 Report: Privacy Risk Framework Application Guide (1.1). Zenodo. https://doi.org/10.5281/zenodo.7107466

[Chakravarthy 2015] Chakravarthy, Ajay, Chen, Xiaoyu, Nasser, Bassem and Surridge, Michael (2015) Trustworthy systems design using semantic risk modelling. 1st International Conference on Cyber Security for Sustainable Society, United Kingdom. 26 - 27 Feb 2015. https://eprints.soton.ac.uk/383465/

[Gol Mohammadi 2014] Gol Mohammadi, N.G., Bandyszak, T., Moffie, M., Chen, X., Weyer, T., Kalogiros, C., Nasser, B. and Surridge, M., 2014, September. Maintaining trustworthiness of socio-technical systems at run-time. In International Conference on Trust, Privacy and Security in Digital Business (pp. 1-12). Springer, Cham.

[Gol Mohammadi 2015] Gol Mohammadi, N., Bandyszak, T., Goldsteen, A., Kalogiros, C., Weyer, T., Moffie, M., Nasser, B. and Surridge, M. (2015) Combining risk-management and computational approaches for trustworthiness evaluation of sociotechnical systems. In, Grabis, J. and Sandkuhl, K. (eds.) Proceedings of the CAiSE 2015 Forum. CAiSE 2015, CEURWS.org, 237-244.

[ISO 27000] ISO/IEC 27000:2018. Information technology – Security techniques – Information security management systems – Overview and vocabulary, International Organization for Standardization, 2018.

[ISO 27005] ISO/IEC 27005:2011. Information technology -- Security techniques -- Information security risk management, International Organization for Standardization, 2011.

[Surridge 2018] Mike Surridge, Gianluca Correndo, Ken Meacham, Juri Papay, Stephen C. Phillips, Stefanie Wiegand, and Toby Wilkinson. 2018. Trust Modelling in 5G mobile networks. In Proceedings of the 2018 Workshop on Security in Softwarized Networks: Prospects and Challenges (SecSoN '18). ACM, New York, NY, USA, 14-19. DOI: https://doi.org/10.1145/3229616.3229621

[Surridge 2019] Surridge M. et al. (2019) Modelling Compliance Threats and Security Analysis of Cross Border Health Data Exchange. In: Attiogbé C., Ferrarotti F., Maabout S. (eds) New Trends in Model and Data Engineering. MEDI 2019. Communications in Computer and Information Science, vol 1085. Springer, Cham.

Appendix: UC2 Example Recommendations JSON

This appendix contains the full JSON structure for the recommendations described in Section 4.3.2.


```
{
  "current": {
    "state": {
      "risk": {
        "overall": "Very High",
        "components": {
          "veryHigh": 2,
          "high": 1,
          "medium": 1,
          "low": 720,
          "veryLow": 1361
        }
      },
      "consequences": [
        {
          "asset": {
            "label": "Robot QC Data",
            "type": "Data",
            "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness\
                   /system#b038b4bd",
            "identifier": "f22527cc",
            "additionalProperties": []
          },
          "label": "LossOfConfidentiality",
          "description": "Disclosure of data (which may be embedded in an IoT \
                          device) to unauthorised parties, or a state where \
                          prevention or detection of such a disclosure cannot \
                          be ensured.",
          "impact": "High",
          "likelihood": "High",
          "risk": "Very High",
          "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness/\
                 system#MS-LossOfConfidentiality-f22527cc"
        },
        {
          "asset": {
            "label": "ABB Robot",
            "type": "Controller",
            "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness\
                   /system#ec200add05477f9",
            "identifier": "a0806d46",
            "additionalProperties": []
```

```
        },
        "label": "LossOfReliability",
        "description": "The device, process or human is liable to make errors\
                        with an unacceptable frequency or extent. Caused by \
                        internal failings including lack of expertise, \
                        software bugs, etc., by using forged, corrupt or \
                        inaccurate information as input, or by a dependency \
                        on some other asset that is not reliable.",
        "impact": "High",
        "likelihood": "High",
        "risk": "Very High",
        "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness/\
               system#MS-LossOfReliability-a0806d46"
      },
      {
        "asset": {
          "label": "Root Cause Data",
          "type": "SensitiveData",
          "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness\
                 /system#309cd493",
          "identifier": "ec69d4d6",
          "additionalProperties": []
        },
        "label": "LossOfIntegrity",
        "description": "Alteration or corruption of data (which may be \
                       embedded in an IoT device) such that its use will \
                       produce incorrect outputs or outcomes, but which may \
                       or may not be malicious and designed to subvert \
                       assets consuming the data.",
        "impact": "Medium",
        "likelihood": "High",
        "risk": "High",
        "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness/\
               system#MS-LossOfIntegrity-ec69d4d6"
      },
      {
        "asset": {
          "label": "User",
          "type": "Human",
          "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness\
                 /system#9c47f662",
          "identifier": "da9b5a9e",
          "additionalProperties": []
        },
        "label": "LossOfReliability",
        "description": "The device, process or human is liable to make errors\
                        with an unacceptable frequency or extent. Caused by \
                        internal failings including lack of expertise, \
                        software bugs, etc., by using forged, corrupt or \
                        inaccurate information as input, or by a dependency \
                        on some other asset that is not reliable.",
        "impact": "Medium",
        "likelihood": "Medium",
        "risk": "Medium",
        "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness/\
               system#MS-LossOfReliability-da9b5a9e"
      }
    ]
  }
},
"recommendations": [
  {
    "identifier": 101,
    "category": "unknown",
```

```
"controlStrategies": [
  {
    "uri": "system#DF.C.CCDFSC.1.3-CCDFSC_1cc98b26_56fefc92_f22527cc_\
           d242983_95f4f858_1cc98b26_95f4f858-CSG-AutoSuspendUntrustworthy\
           ClientAccess-Implementation-Runtime",
    "description": "Access to service \"Safe VA DB\" by client \"User \
                    Client\" has been automatically disabled to prevent \
                    authenticated attacks by compromised clients. This \
                    strategy represents activation of a contingency plan \
                    at runtime, and can be selected to discover what \
                    effect this would have on risk levels, allowing this \
                    to be used for decision support calculations. \
                    Activation at runtime requires \"Safe VA DB\" to be \
                    managed by a suitable adaptation framework. The \
                    Disable Client Access control should be deselected \
                    if and when access by \"User Client\" to \"Safe VA \
                    DB\" has been enabled once again."
  },
  {
    "uri": "system#Co.U.CoPD.0.1-CoPD_a0806d46_46c276dd_d85b3350-CSG-\
           SuspendUnreliableController-Implementation-Runtime",
    "description": "Change from: The IoT controller device \"ABB Robot\" \
                    has been disabled when it becomes unreliable, to \
                    prevent it causing problems in the physical environment\
                    where it operates. This strategy represents activation\
                    of a contingency plan at runtime, and can be selected \
                    to discover what effect this would have on risk levels,\
                    allowing this to be used for decision support \
                    calculations. To activate it at runtime, signal user \
                    \"Admin\" who is responsible for managing the device. \
                    The Disabled Host control should be deselected only \
                    when the host has been restarted."
  },
  {
    "uri": "system#DS.I.HPDFDADS.0-HPDFDADS_a910570c_2d7c65e1_24248b09_\
           5b57f824_23604ba7_be3c9766_95f4f858-CSG-AutoSuspendCorrupt\
           DataFlow-Implementation-Runtime",
    "description": "The flow of data \"Scene Data\" from \"Kafka Server\"\
                    to \"Safe VA DB\" has been automatically disabled to \
                    prevent corrupt or malicious content (including \
                    malware) from disrupting the receipient \"Safe VA DB\"\
                    . This strategy represents activation of a \
                    contingency plan at runtime, and can be enabled to \
                    discover what effect this would have on risk levels, \
                    allowing this to be used for decision support \
                    calculations. Activation at runtime requires \"Safe \
                    VA DB\" to be managed by a suitable adaptation \
                    framework. The Disabled Data Flow control should be \
                    deselected if and when the flow of data is enabled \
                    once again."
  }
],
"controls": [
  {
    "label": "DisabledHost",
    "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness/\
           system#CS-DisabledHost-a0806d46",
    "asset": {
      "label": "ABB Robot",
      "type": "http://it-innovation.soton.ac.uk/ontologies/trustworthine\
              ss/domain#Controller",
      "uri": "system#ec200add05477f9",
      "identifier": "a0806d46",
      "additionalProperties": []
```

```
      },
      "action": "Enable control"
    },
    {
      "label": "DisableClientAccess",
      "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness/\
             system#CS-DisableClientAccess-56fefc92",
      "asset": {
        "label": "[ClientServiceRelationship:(User Client)-(Safe VA DB)]",
        "type": "http://it-innovation.soton.ac.uk/ontologies/trustworthine\
                ss/domain#ClientChannel",
        "uri": "system#ClientChannel_1cc98b26_95f4f858",
        "identifier": "56fefc92",
        "additionalProperties": []
      },
      "action": "Enable control"
    },
    {
      "label": "DisabledDataFlow",
      "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness/\
             system#CS-DisabledDataFlow-5b57f824",
      "asset": {
        "label": "[DataFlow:(Scene Data)-(Kafka Server)-(Safe VA DB)]",
        "type": "http://it-innovation.soton.ac.uk/ontologies/trustworthine\
                ss/domain#DataFlow",
        "uri": "system#DataFlow_a910570c_23604ba7_95f4f858",
        "identifier": "5b57f824",
        "additionalProperties": []
      },
      "action": "Enable control"
    }
  ],
  "state": {
    "risk": {
      "overall": "Medium",
      "components": {
        "veryHigh": 0,
        "high": 0,
        "medium": 2,
        "low": 691,
        "veryLow": 1392
      }
    },
    "consequences": [
      {
        "asset": {
          "label": "User",
          "type": "Human",
          "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthine\
                  ss/system#9c47f662",
          "identifier": "da9b5a9e",
          "additionalProperties": []
        },
        "label": "LossOfTimeliness",
        "description": "Represents a state in which a data asset is \
                        somewhat out of date, or a process or human has \
                        outdated or (temporarily) unavailable inputs.",
        "impact": "Low",
        "likelihood": "Very High",
        "risk": "Medium",
        "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthiness\
               /system#MS-LossOfTimeliness-da9b5a9e"
      },
      {
```

```
            "asset": {
              "label": "Robot QC Data",
              "type": "Data",
              "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthi\
                     ness/system#b038b4bd",
              "identifier": "f22527cc",
              "additionalProperties": []
            },
            "label": "LossOfConfidentiality",
            "description": "Disclosure of data (which may be embedded in an \
                            IoT device) to unauthorised parties, or a state \
                            where prevention or detection of such a disclosure \
                            cannot be ensured.",
            "impact": "High",
            "likelihood": "Low",
            "risk": "Medium",
            "uri": "http://it-innovation.soton.ac.uk/ontologies/trustworthi\
                   ness/system#MS-LossOfConfidentiality-f22527cc"
          }
        ]
      }
    }
  ]
}
```