A Logic-Based Network Security Zone Modelling Methodology

Sravani Teja Bulusu, Romain Laborde, Ahmad Samer Wazan, Francois Barrère, Abdelmalek Benzekri Authors
IRIT / Université Paul Sabatier, Toulouse, France
sbulusu@irit.fr, laborde@irit.fr, ahmad-samer.wazan@irit.fr, francois.barrere@irit.fr, abdelmalek.benzekri@irit.fr

***Abstract*— Network segmentation and security zone modelling is a best practice approach, widely known for minimizing the risks pertaining to the compromise of enterprise networks. In this paper, we propose a security zone modelling methodology, which automates the process of security zone specification using a definite set of formalized rules. It mainly helps to derive network security requirements based on the Clark-Wilson lite formal model. We illustrate our methodology using an example case study of e-commerce enterprise network infrastructure.**

Keywords- Network Security requirements; Security zoning.

I. INTRODUCTION

Over the past years, the growing dependency of the business-critical applications and processes on network technologies and services, has expanded the threat landscape to a large extent. Today, networks constitute the main vector as well as the convenient platform, to launch attacks against organizations. An inadequate network security design can lead to data loss in spite of the monitored traffic, and security incidents handling. In addition adds overhead in terms of time, effort, and costs.

The current practice for eliciting and analyzing early network security requirements is driven by security zoning, a well-known defense in depth strategy for network security design [1]. Security zones constitute the logical grouping of security entities that are identified with similar protection requirements (e.g., data confidentiality and integrity, access control, audit, logging, etc.). Each security zone is identified with different trust levels, which exhibit the rigor of required protection. Determining security zones and respective trust levels is a preliminary step for security architects in capturing other network security requirements (e.g., related to data flows), and later in selecting the right network security controls/mechanisms (such as VPN, IP Firewall, etc.).

In this regard, several works, theories, and best practice approaches are available, explaining on various zone classification schemes and patterns [2]–[4]. Nevertheless, there there exists no standard methodology that can drive the specification of zones for a given infrastructure. In practice, the design of the security zone model is manual and depends on the expertise of the security architects who may forget some details while specifying the zone model. Given this situation, how to How to ensure that the proposed network segmentation is correct and cost-effective? How to ensure that no network security requirement is missing or irrelevant?

In this paper, we propose a security zone modelling methodology, which automates the process of security zones specification using a definite set of formalized rules, thereby leaving less space to any manual errors. It helps in deriving network security requirements based on the Clark-Wilson lite formal security model for integrity. We illustrate our methodology using an example case study of e-commerce enterprise network infrastructure.

The rest of this paper is organized as follows. Section II briefs the literature study on zone modelling. Section III describes the example case study. Section IV details the strategy our zone modelling methodology. Section V includes a discussion of proposed methodology. Finally, Section VI concludes this article.

II. RELATED WORKS

From our literature study, we noticed limited works concerning network security zones in academic sector [5], [6]. Majority of the existing works are found to be from industrial/government sectors [2]–[4], which mainly focus on providing foundational best practice guidelines, and reference modelling patterns, for building secured networks. In this section, we confine our discussion to these reference models. From a broad view, these reference models propose minimum set of zones as well as inter/intra zone interactions rules necessary to be implemented, for achieving basic logical network security design.

For instance, the British Columbia model [4] describes seven zones and allows communication inside the zones and only between adjacent zones. Secure Arc [3] defines eight zones. It also add a parallel cross-zones segmentation concept, called silos, see Figire1. Communications are allowed only between adjacent zones and within the same silo, or between adjacent silos within the same zone. The aim is to limit the interaction between the zones to only dedicated traffic even though they are adjacent to each other. Besides, there exists no restriction on either the number of zones or their category types, as they depend on the size and type of the business. Some of the commonly identified network zones include internet zone, demilitarized zones, etc. Internet zone, by default, is assumed as extremely hostile and least trusted, as it is publicly accessible to everyone including the anonymous threat actors. The Enterprise zone and restricted zone contain the set of security entities (e.g., users, desktops, servers, etc.) that are part of the enterprise. Sensitive assets are confined to highly restricted zones. The demilitarized zone (DMZ) is the intermediate zone that usually sits between the trusted and less trusted zone in order to reduce attacks surfaces. The extranet zone contains trust security entities that

 ISBN: 978-1-61208-661-3

belong to an external third-part domain (e.g., external internet service provider). Finally, the management zone constitutes of entities that are involved in security management activities such as monitoring, and administering the zones and their interactions. Likewise, different patterns propose different set of zones and interaction rules for zone interactions. The communication between zones are monitored and controlled by some security measures (e.g., a firewall, a gateway, etc.).

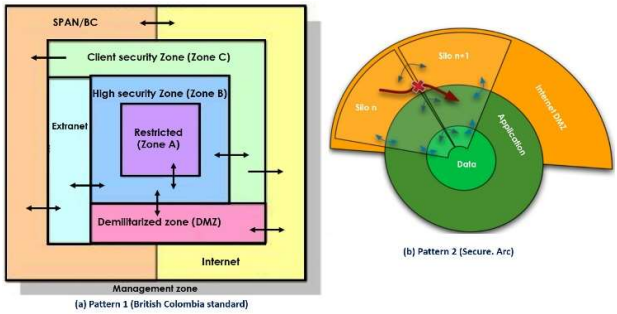


Figure 1. Example zone reference models [3], [4]

In academic sector, *Gontarczyk et al.* [6] proposed a standard blue-print that includes three classes of security zone (no physical measures, limited physical measures, and strong physical measures). It also provides a classifier to guide the deployment of systems/applications. *Ramasamy et al* [5] proposed a bottom-up approach for discovering the security zone classification of devices in an existing enterprise network. However, these documents are only guidelines and must be manually adapted. As a consequence, they exists no rigorous methodology to help security architects in validating their network security requirements.

III. EXAMPLE CASE STUDY

To illustrate our methodology implementation, we consider an e-commerce enterprise network case study [7]. The initial network architecture, as given in Figure 2 (a), consists of server components such as such as WEB server, DNS server, Application server, Database server, and the Accountability server.

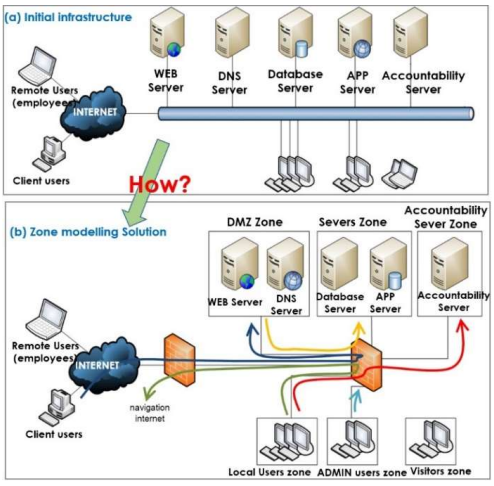


Figure 2. e-commerce example case study [7]

The employees are distinguished as administrators and standard users, who can connect to the network through LAN or WIFI. If the employees are outside the enterprise, they can remotely connect to the enterprise network. The Accountability server is said to be highly critical as it manages the financial information of the company (e.g., salaries of employers). Finally, when the clients visit the enterprise, they are allowed to connect to WEB through WIFI. Figure 2(b) depicts an example of zone modelling solution proposed by the network architects of the enterprise. It is evident that the solution reflects some best practice guidelines by defining some zones such as DMZ zone, user's zones, etc.

For instance, the Accountability server is isolated in a separate zone as it is critical. Comparatively, the application and data base servers are less critical, but cannot be exposed to Internet. Likewise, the arguments can be subjective, referring to the criticality of the assets and their risk impact, if compromised. However, how did the architects arrive to this solution (from the initial architecture in Figure 2(a) to Figure 2(b)? How can security architect demonstrate the correctness of the final security architecture? In this regard, a formal approach justifying the transition from the problem to the solution is required, for a traceable and verifiable security zone specification process.

IV. THE PROPOSED METHODLOGY - STRATEGY

The principle motivation of our work is to propose a generic methodology that can drive the specification of network security zones, with respect to the business interaction needs. The conception of our methodology commenced with an idea of merging the concepts of trust and criticality, using the integrity property. The reason behind choosing integrity is fundamental. According to the oxford dictionary, integrity from computer science perspective is defined as "*Internal consistency or lack of corruption in electronic data*", whereas integrity of humans is defined as the "*The quality of being honest*".

For example, consider that we are reading a scientific article published as a security conference. In this case, we expect the information contained in this article to be scientifically true, because the content of each article is validated by reviewers, who are recognized in the domain of security. Contrarily, we can't have the same expectation for scientific articles published in teenager blogs since there is no content validation. The information available in the blog is not necessarily wrong. It just means that the readers do not have the same level of assurance. Scientific articles published in the security conferences are more trustful than those published in teenager blogs. Integrity is thus related to trust when considering the external or unmanaged systems. Likewise, integrity is also related to risk. A critical system must be consistent, « honest », which means it requires high level of integrity. In addition, systems take decisions based on information (e.g. a program executes an algorithm based on its inputs). If input information is wrong, then decisions can be wrong too. Therefore, we will only permit critical system to consider information with high level of integrity (i.e. high level of assurance). Hence, integrity is a pivot concept between trust and risk.

 ISBN: 978-1-61208-661-3

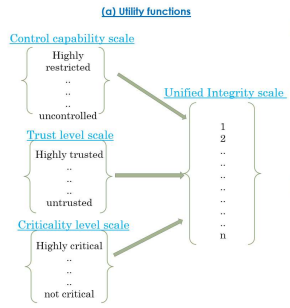
In practice, there exists several models for integrity such as Biba [8], clark-wilson [9], which propose abstract solutions to preserve the integrity of information flows. These models are widely used in current operating systems for improving the integrity protection of the information flows in inter-process communications (e.g., Microsoft Windows Integrity Mechanism [10]). In our methodology, we propose to integrate these formal integrity security concepts to security zone modelling design principles, for addressing the risks pertaining to traffic flows. We adapt the concepts of Clark-Wilson lite [11] model (lighter version of clark-wilson model), for verifying the integrity property of traffic flows traversing multiple zones. In below, we briefly discuss the underlying concepts of our methodology.

A. Security domains, security zones and agents

To facilitate the integration of security zoning concepts to our network requirement analysis context, we mainly consider three elements: domains, zones and agents. A security domain represents the organizational authority, which controls and manages the entities (i.e., servers, software, data, users, etc.) that belong to it. We call these entities as agents. Furthermore, a security domain can be refined into sub-domains highlighting different policies or procedures within the same organization. Agents are categorized into two groups. System agents refer to entities under direct control such as software/hardware systems that are developed and/or maintained by the enterprise. Environment agents are not under direct control and refer to humans, or to some purchased third party software/hardware. Finally, security zones constitute logical grouping of agents with common protection requirements.

B. Integrity levels

To facilitate the integration risk analysis concepts to our network requirement analysis context, we consider a unified scale of integrity levels for all the domains, zones and agents which is determined based on risk analysis. Figure 3(b) shows some hypothetical scales, assumed for the case study.



(b) Integrity levels of domains and agents for case study

Domain	Control capability	Integrity
Enterprise internal sub- domain	Highly restricted	5
Enterprise external sub-domain	Less controlled	2
internet	uncontrolled	1
System agents	**Criticality**	**Integrity**
WEB server	Partially critical	3
DNS server	Partially critical	3
APP sever	Partially critical	3
Database server	Critical	4
Accountability server	highly critical	5
Environment agents	**Trust**	**Integrity**
Admin user	highly trusted	4
Local user	trusted	3
visitor	Partially trusted	2
Remote user	Partially trusted	2
Client user	not trusted	1

Figure 3. Integrity values of domains and agents for the example case study

The integrity level of a domain is defined based on its *control capability* that describes the potential of a domain for controlling its agents. For instance, a well-controlled domain means that the security management activity within the domain is mature. In our scenario, the enterprise domain is divided into two sub-domains (see Figure 4). The *internal sub-domain* consists in the assets within enterprise premises and the *external sub-domain* is the remote users. Likewise, the integrity levels of *environment agents* are determined based on their *trust levels*. Trust level in general, specifies the degree of the trustworthiness over the expected behavior of environment agents in a given context. Since remote users are not in controlled domain, remote users are less trusted than local users.

Finally, the integrity level of *system agents* are determined based on their *criticality levels*. Criticality level determines the sensitivity to threats and their risk impact on the overall business. Here, the accountability server being highly critical requires a high level of integrity. On the other hand, the WEB server is considered less critical for business, which means it doesn't require as many as integrity requirements.

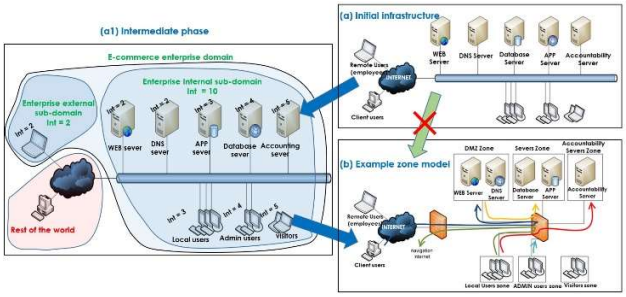


Figure 4. Our methodology conceptual initialization

Furthermore, in our methodology context, we assume the existence of some utility functions (Figure 3(a)) that map the control capability labels of domains, criticality and trust levels of agents into a unified scale of integrity levels. For instance, IEC 61508 [12] defines safety integrity levels (SIL) based on controllability of the system from the risk of failures. Similar, these utility functions must be determined based on business risk impact, which is a pre-requisite to define zone model [4].

C. The Clark-Wilson lite model

Finally, to introduce the security verification on data flow, we validate that the integrity of the information flow is respected. According to CW-lite integrity model [11], all information flowing from untrusted subjects to trusteed subjects must be filtered. Here the trust of the subjects are represented with integrity levels. The filter is placed at the receiving subject's side. Figure 5 shows the formal rule.

$$flow(s_i, s, I) \wedge \neg filter(s, I) \rightarrow (int(s_i) \geq int(s))$$

Figure 5. CW-lite security filtering rule [11]

This predicate should be read as follows: "if a subject s receives an information flow from a subject s_i at interface I, then either there is an integrity validation filter at interface I or the integrity level of s_i is greater or equal to the integrity of subject s". Here, the integrity validation filters correspond to security verification procedures (e.g., a WEB application firewall that checks SQL statements or URL formats).

V. PROPOSED METHODOLOGY

Our zone modelling methodology (see Figure 6) is divided into two main steps: (1) Determining the security zones and integrity validation filters and (2) Identifying data flows integrity requirements and flows access control filters.

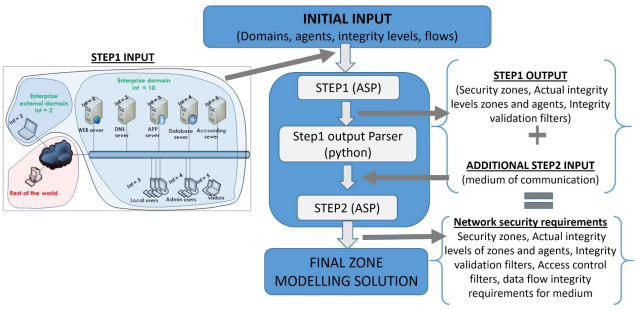


Figure 6. Our methodological approach overview

As shown in Figure 6, at step1, the initial input is the set of security domains, the set of agents, the integrity levels of domains and agents, and the data flows between agents. As a result of step1, our process computes the security zones and the integrity validation filters. In the second step, the designer needs to provide additional information about the media of communication (i.e., the networks). The final result is a set of network security requirements, which are a set security zones, integrity validation filters, agents integrity requirements, access control filters and integrity data flow protection requirements.

In the following, we discuss in detail the modelling rules at step1 and step2.

A. *Specifying zones and filtered flows*

The main goal of this step is to specify zones and identify integrity validation filters. At this step 1, we start with a system as a set of domains (DOMAIN), zones (ZONE) and agents (AGENT). We represent it as follows:

$$S = < \text{DOMAIN}, \text{ZONE}, \text{AGENT}, \text{FLOW}, \text{INSIDE}_Z^D, \text{INSIDE}_A^D, \text{INSIDE}_A^Z, \text{Int}, \text{Int}_{max}, \text{Int}_{min}, \text{Int}_{actual}, \text{Agent}_{server}, \text{Agent}_{client} >$$

Where,

- DOMAIN is the set of security domains.
- ZONE is the set of security zones.
- AGENT is the set of agents, named after entities. $\text{AGENT} = \text{ENV\_AGENT} \cup \text{SYST\_AGENT}$ with ENV_AGENT and SYST_AGENT being the set of environment and system agents such that $\text{ENV\_AGENT} \cup \text{SYST\_AGENT} = \emptyset$.
- Agent_{server} : AGENT → {TRUE, FALSE} states if agent is a server (e.g., WEB server).
- Agent_{client} : AGENT → {TRUE, FALSE} states if agent is a client (e.g., browser).
- $\text{FLOW} \subseteq \text{AGENT} \times \text{AGENT}$ is the set of allowed flow of information.
- $\text{INSIDE}_Z^D \subseteq \text{ZONE} \times \text{DOMAIN}$ is a relation that states a zone is in a domain.
- $\text{INSIDE}_A^D \subseteq \text{AGENT} \times \text{DOMAIN}$ is a relation that states an agent is in a domain.
- $\text{INSIDE}_Z^D \subseteq \text{AGENT} \times \text{ZONE}$ is a relation that states an agent is in a zone.
- $\text{Int}: \text{DOMAIN} \rightarrow \mathbb{N}$ returns the integrity level of a security domain which is fixed.
- $\text{Int}_{max}: \text{ZONE} \cup \text{AGENT} \rightarrow \mathbb{N}$ returns the maximum integrity of a zone or an agent. For environment agents, this value is directly derived from their trust label.
- $\text{Int}_{min}: \text{AGENT} \rightarrow \mathbb{N}$ returns the minimum integrity level of an agent. For system agents, this value is directly derived from the criticality label.
- $\text{Int}_{actual}: \text{ZONE} \cup \text{AGENT} \rightarrow \mathbb{N}$ returns the actual integrity of a zone or an agent, which are the final integrity values chosen at the end of the computation.
- integrity-validation-filter(a: AGENT, f: FLOW, val1: Int, val2: Int) states integrity validation requirements such that *integrity-validation-filter(a, f, val1, val2)* describes integrity protection mechanism at agent *a* must sanitize dataflow *f* with an integrity level of *val1* to achieve a data assurance level of *val2*.

In other words, Int , Int_{max} and Int_{min} represent the integrity utility functions in Figure 3. Accordingly, we define the rules of step1 as follows:

RULE1: Every agent is inside a domain.

$$\forall a \in \text{AGENT}, \exists\, d \in \text{DOMAIN} \mid (a, d) \in \text{INSIDE}_A^D$$

RULE2: Every security domain contains at least one security zone.

$$\forall d \in \text{DOMAIN}, \text{card}(\{z \mid z \in \text{ZONE}, (z, d) \in \text{INSIDE}_Z^D\}) \geq 1$$

RULE3: The maximum integrity level of a security zone is equal to the integrity level of the domain. This is because, a domain controls zone and therefore we cannot have more assurance on a zone than that of the domain.

$$\forall d \in \text{DOMAIN}, \forall z \in \text{ZONE}, (d, z) \in \text{INSIDE}_Z^D \quad \text{Int}_{max}(z) = \text{Int}(d)$$

RULE4: Similar to Rule 3, the maximum integrity level of an agent is equal to the integrity level of domain.

$$\forall d \in \text{DOMAIN}, \forall\, a \in \text{AGENT}, (a, d) \in \text{INSIDE}_A^D \quad \text{Int}_{actual}(a) \leq \text{Int}(d)$$

RULE5: The actual integrity of a zone cannot be greater than its maximum integrity.

$$\forall z \in \text{ZONE}, \text{Int}_{actual}(z) \leq \text{Int}_{max}(z)$$

RULE6: The actual integrity of agents must be between the maximum and the minimum integrity levels of the agents.

$$\forall a \in \text{AGENT}, \text{Int}_{min}(a) \leq \text{Int}_{actual}(a) \leq \text{Int}_{max}(a)$$

RULE7: The actual integrity levels of an agent is same as that of its residing zone.

$$\forall a \in \text{AGENT}, \forall z \in \text{ZONE}, (a, z) \in \text{INSIDE}_A^Z, \quad \text{Int}_{actual}(a) = \text{Int}_{actual}(z)$$

RULE8: The actual integrity levels of the interacting agents must adhere to the CW-lite integrity rule. In this way, an agent doesn't access a lower integrity information.

$$\forall a1, a2 \in \text{AGENT}, (a1, a2) \in \text{FLOW} \wedge \neg\, \text{integrity-validation-filter}(a2, \text{flow}(a1, a2), \text{Int}_{actual}(a1), \text{Int}_{actual}(a2)) \Rightarrow \text{Int}_{actual}(a1) \geq \text{Int}_{actual}(a2)$$

RULE9: Server agents and client agents cannot reside in same zone. Because, as per the zone modelling design principles, intra-zone interactions are usually not analysed.

 ISBN: 978-1-61208-661-3

With reference the security design principle known as complete mediation rule, every access to every object must be checked for authority[13]. By default, this complete mediation rule is checked for client-server models. Therefore, if server and client reside in same zone there will be a conflict.

$$\forall a1, a2 \in \text{AGENT}, \forall z1, z2 \in \text{ZONE}, (a1, z1) \in \text{INSIDE}_{\text{A}}^{\text{Z}}, (a2, z2) \in \text{INSIDE}_{\text{A}}^{\text{Z}}, \text{Agent}_{\text{server}}(a1), \text{Agent}_{\text{client}}(a2) \Rightarrow z1 \neq z2$$

RULE10: Server agents that are not equally accessible to the client agents cannot reside in same zone. This rule refers to least privilege principle [13] that permits only privileged flows. Since the intra-zone interactions are not controlled from network point of view (as mentioned earlier), once an agent connects to a server in zone, then the agent can potentially communicate with other servers within that zone. Therefore, our rule states that, if any two servers reside in a zone and a client is denied flow to one of them, then it will result in a conflict.

$$\forall a1, a2, a \in \text{AGENT}, \forall z1, z2 \in \text{ZONE}, (a1, z1), (a2, z2) \in \text{INSIDE}_{\text{A}}^{\text{Z}}, (a, z1), (a, z2) \notin \text{INSIDE}_{\text{A}}^{\text{Z}}, \text{Agent}_{\text{server}}(a1), \text{Agent}_{\text{server}}(a2), \text{Agent}_{\text{client}}(a), \text{flow}(a1, a), \neg \text{flow}(a2, a) \Rightarrow z1 \neq z2$$

B. *Step2: Specifying integrity requirements for the communication medium between zones*

At the end of step1, we have the set of zones along with the integrity validation filters. In step2, we address the security issues of inter-zone interactions, i.e., we consider the protection of the flow through the network communication medium (e.g., wired/wireless networks, etc.,) that connect zones. The main goal of this step is to protect the integrity of data flows when traversing untrusted media of communication. Suitably, we complete our system model as follows:

$$S = < \text{DOMAIN}, \text{ZONE}, \text{AGENT}, \text{FLOW}, \text{MEDIUM}, \text{INSIDE}_{\text{Z}}^{\text{D}}, \text{INSIDE}_{\text{A}}^{\text{D}}, \text{INSIDE}_{\text{A}}^{\text{Z}}, \text{INSIDE}_{\text{M}}^{\text{D}}, \text{CONNECT}, \text{PATH}, \text{Int}, \text{Int}_{\text{max}}, \text{Int}_{\text{min}}, \text{Int}_{\text{actual}} >$$

Where:

- MEDIUM is the set of media of communication.
- $\text{INSIDE}_{\text{M}}^{\text{D}} \subseteq \text{MEDIUM} \times \text{DOMAIN}$ is a relation, which states that a medium of communication is in a domain.
- $\text{CONNECT} \subseteq \text{MEDIUM} \times \text{ZONE}$ is a relation, which states that a zone is connected to a medium of communication.
- Int_{max}: $\text{ZONE} \cup \text{AGENT} \cup \text{MEDIUM} \rightarrow \mathbb{N}$ returns the maximum integrity level of a security zone, agent or medium of communication.
- $\text{Int}_{\text{actual}}$: $\text{ZONE} \cup \text{AGENT} \cup \text{MEDIUM} \rightarrow \mathbb{N}$ returns the actual integrity level of a security zone, agent or medium of communication.
- $\text{PATH} \subseteq \text{FLOW} \times (\text{ZONE} \cup \text{MEDIUM}) \times (\text{ZONE} \cup \text{MEDIUM})$, is a relation that stores where flows are transiting with the constraint that $\forall (f, e1, e2) \in \text{PATH} \Rightarrow (e1, e2) \in \text{CONNECT} \vee (e1, e2) \in \text{CONNECT}$. For instance, *(f,m,z) ∈ PATH* means that flow *f* transits between medium *m* to zone *z*.
- access-control-filter(c: CONNECT, f: FLOW states access control requirements such that *access-control-filter(c,f)* means flow *f* must be permitted at connection *c*.
- dataflow-integrity-protection(f: FLOW, e: ZONE∪ MEDIUM, value: INT) states dataflow protection requirements such that *dataflow-integrity-protection(f,e,val)* means some protection mechanism must be applied on dataflow *f* over zone or medium *e* to preserve an integrity level of *val*.

Similar to domains, zones, and agent, the medium of communication m1 has two integrity levels: Int_{min} (m1), and $\text{Int}_{\text{actual}}$ (m1). Accordingly, we add new rules to include constraints on media of communication:

RULE11: Every zone must be connected to a medium of communication.

$$\forall z \in \text{ZONE}, \exists m \in \text{MEDIUM}, (m, z) \in \text{CONNECT}$$

RULE12: At each zone, there must be an access control filter that permits allowed flow of information. Not explicitly allowed flows are denied by default.

$$\forall (f, e1, e2) \in \text{PATH}, e1 \in \text{MEDIUM} \Rightarrow \text{access-control-filter}((e1, e2), f)$$

Respectively:

$$\forall (f, e1, e2) \in \text{PATH}, e1 \in \text{ZONE} \Rightarrow \text{access-control-filter}((e2, e1), f)$$

RULE13: The actual integrity level of a medium of communication is the minimum value of the integrity level of its domain, the trust on the medium (i.e., its maximum integrity), and the actual integrity levels of the connected zones.

$$\forall m \in \text{MEDIUM}, \text{Int}_{\text{actual}}(m) = \min(\{\text{Int}(d) | d \in \text{DOMAIN}, (m, d) \in \text{INSIDE}_{\text{M}}^{\text{D}}\} \cup \{\text{Int}_{\text{max}}(m)\} \cup \{\text{Int}_{\text{actual}}(z) \mid z \in \text{ZONE}, (m, z) \in \text{CONNECT}\})$$

RULE14: A flow that transits over a medium or a zone, requires an integrity protection, if the integrity level of the medium or the zone is lower than the level of integrity of the flow.

$$\forall (a1, a2) \in \text{FLOW}, \forall e1, e2 \in \text{ZONE} \cup \text{MEDIUM} \mid (\text{flow}(a1, a2), e1, e2) \in \text{PATH}, (\min(\text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)) > \text{Int}_{\text{actual}}(e1) \Rightarrow \text{data-flow-integrity-protection}(\text{flow}(a1, a2), e1, \min(\text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)))).$$

Respectively:

$$\forall (a1, a2) \in \text{FLOW}, \forall e1, e2 \in \text{ZONE} \cup \text{MEDIUM} \mid (\text{flow}(a1, a2), e1, e2) \in \text{PATH}, (\min(\text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)) > \text{Int}_{\text{actual}}(e2) \Rightarrow \text{data-flow-integrity-protection}(\text{flow}(a1, a2), e2, \min(\text{Int}_{\text{actual}}(a1), \text{Int}_{\text{actual}}(a2)))).$$

VI. DISCUSSION

Our zone modelling rules are abstract and design independent therefore does not restrict the design solutions. Therefore, we do yet classify the zone types like DMZ, restricted, etc. We implemented the whole process in ASP

using solver Clingo [14] and Python2.7 to automate the security zones computation. Due to space constraints, we do not detail on the tool implementation of the case study. Instead, we limit our discussion to the integrity levels and network security requirements.

The actual integrity levels of zones and agents correspond to the pre-requisite security requirements that ensure the expected behaviour of the agents as well as the expected security management capability of the zones. The future security design implementing these requirements must maintain these integrity levels at minimum. In practice, there already exist formally accepted approaches, which specify the profoundness of security verification required, for varying design assurance levels (known as DALs). DALs are determined from the safety assessment process and hazard analysis by examining the effects of a failure condition in aircraft systems [15]. The higher the DAL is, the higher the assurance activities or verification methods are demanded.

Furthermore, the network security requirements defined by our methodology. Firstly, the integrity validation filters (from RULE9) defined for the filtered flows represent validation processes to be implemented either by the target agent (e.g., by some specific validation code) or some external security mechanisms (e.g., deep inspection mechanisms). For instance, the data flow between the local users and the accountability server must be validated. Let's say their actual integrity values are 3 and 5 respectively. Then as per RULE9, an incoming data flow having an integrity level of 3 must be sanitized in order to conform integrity level 5. Interpretation of such integrity validation requirement, i.e. what means validation to conform integrity level of 5, which can be carried out on the basis of dedicated documents such as the specification for data assurance levels by EUROCONTROL [16]. Suitably, the filter validation can be implemented at the end of accountability server using a security mechanism such as a WEB application firewall that checks for SQL injection. As a result, the refinement of the filtering functionality may give rise to new security verification requirements. However, describing the refinement of the integrity verification filtering requirements is out of the scope of this article.

Secondly, access control filters (from RULE 12) defined at the entry/exit interfaces of each zone describe the need to control all the inter-zone communications. Depending on the security design specifications, these filters may correspond to firewalls, application gateways, etc., depending on the security design specifications. One access control filter may be implemented by one or more access control mechanisms (e.g., firewalls). This depends on the integrity level of the zones. Finally, the integrity flow requirements defined (from RULE14) for the data flow describe the need for security protection mechanism while transiting a medium or a zone.

VII. CONCLUSION AND PERSPECTIVES

Network security zone modelling is a well-known approach that contributes to the defense-in-depth strategy from the network security perspective. However, no rigorous approach formally defines this process. To address this issue, we proposed a zone modelling methodology based on Clark-Wilson lite formal model. We provide a set of formal rules as well as the list of initial integrity levels values computed based on risk impact, which makes our methodology approach traceable and verifiable.

As future works, we plan to integrate this work in the process of security requirements engineering. This allows refining business level security objectives into network security requirements. In parallel, we would like to extend our security zone modelling approach to consider the confidentiality and availability requirements as well.

ACKNOWLEDGMENT

This work is part of project IREHDO2 funded by DGA/DGAC. The authors thank all the security experts at Airbus who helped us with their useful comments.

REFERENCES

[1] SANS, 'Infrastructure Security Architecture for Effective Security Monitoring'. 2015.

[2] Government of Canada, Communications Security Establishment, 'Baseline Security Requirements for Network Security Zones in the Government of Canada'. 2007.

[3] Secure Arc, 'Logical Security Zone Pattern. http://www.securearc.com/wiki/index.php/Logical_Security_Zone_Pattern'.

[4] Province of British Columbia, 'Enterprise IT Security Architecture Security Zones: NETWORK SECURITY ZONE STANDARDS - Office of Chief Info Officer'. 2012.

[5] Ramasamy et al, 'Towards Automated Identification of Security Zone Classification in Enterprise Networks.', in *Hot-ICE*, 2011.

[6] A. Gontarczyk, P. McMillan, and C. Pavlovski, 'Blueprint for Cyber Security Zone Modeling', *Inf. Technol. Ind.*, 2015.

[7] Cybedu, 'Sensibilisation et initiation à la cybersécurité-consortium', 2017.

[8] Biba, 'Integrity considerations for secure computer systems', DTIC, 1977.

[9] D. D. Clark and D. R. Wilson, 'A comparison of commercial and military computer security policies', in *Security and Privacy, IEEE Symposium on*, 1987.

[10] Microsoft, 'Windows Vista integrity mechanism and earlier integrity models'. [Online]. Available: https://msdn.microsoft.com/fr-FR/library/bb625957.aspx.

[11] U. Shankar, T. Jaeger, and R. Sailer, 'Toward Automated Information-Flow Integrity Verification for Security-Critical Applications.', in *NDSS*, 2006.

[12] IEC 61508, 'Functional safety of electrical/electronic safety-related systems - Part 1: General requirements'. 2010.

[13] J. H. Saltzer and M. D. Schroeder, 'The protection of information in computer systems', *Proc. IEEE*, 1975.

[14] M. Gebser, B. Kaufmann, R. Kaminski, M. Ostrowski, T. Schaub, and M. Schneider, 'Potassco: The Potsdam answer set solving collection', vol. 24, pp. 107–124, 2011.

[15] Bieber et al, 'DALculus–theory and tool for development assurance level allocation', in *International Conference on Computer Safety, Reliability, and Security*, 2011.

[16] EUROCONTROL, 'Specification for Data Assurance Levels'. Mar-2012.

 ISBN: 978-1-61208-661-3