Management of Security Policy Configuration using a Semantic Threat Graph Approach

Simon N. Foley and William M. Fitzgerald,
Cork Constraint Computation Centre,
Computer Science Department,
University College Cork, Ireland.
s.foley@cs.ucc.ie, wfitzgerald@4c.ucc.ie

Abstract

Managing the configuration of heterogeneous enterprise security mechanisms is a complex task. The effectiveness of a configuration may be constrained by poor understanding and/or management of the overall security policy requirements, which may, in turn, unnecessarily expose the enterprise to known threats. This paper proposes a threat management based approach, whereby knowledge about the effectiveness of mitigating countermeasures is used to guide the autonomic configuration of security mechanisms. This knowledge is modeled in terms of *Semantic Threat Graphs*, a variation of the traditional Threat/Attack Tree, extended in order to relate semantic information about security configuration with threats, vulnerabilities and countermeasures. An ontology-based approach to representing and reasoning over this knowledge is taken. A case study based on Network Access Controls demonstrates how threats can be analysed and how automated configuration recommendations can be made based on catalogues of countermeasures. These countermeasures are drawn from best-practice standards, including NIST, IETF and PCI-DSS recommendations for firewall configuration.

1 Introduction

A significant challenge in the process of securing complex systems is attaining a degree of confidence that a security configuration adequately addresses the (security) threats. *Threat Trees* [39, 61], *Attack Trees* [56] and similar tree-based threat-modeling methodologies [6, 17] are used to help

identify, represent and analyse threats to an enterprise's assets. Their top-down approach provides a semi-formal but methodical way to determine viable threat vectors (who, why and how a system can be compromised). In practice these trees are used for threat elicitation and analysis: representing threats at a high-level of abstraction and they tend not to be used to capture low-level or concrete security configuration detail. For example, while a threat tree may identify a firewall countermeasure for a Denial of Service attack, a threat tree is not effective, or intended to model low-level configuration details. For example, distinctions between SYN-proxy versus SYN-threshold configurations [16] for a firewall in a subnet downstream from other similarly configured firewalls. In the latter case much of semantics of the threats and corresponding countermeasures are implicit and outside of the threat tree structure. Threat trees are useful for analysing threats in a local context that is decomposed from some root threat, however their advantages are diminished when one considers the threat in a global context (across multiple root threats).

In this paper, we consider how a threat tree style approach can provide a basis for automatically testing whether a security configuration adequately mitigates the identified threats. We extend the threat tree model to include semantic knowledge about the security configuration and how it relates to assets, threats, vulnerabilities and countermeasures. Knowledge, such as security and network configuration, and relationships with vulnerabilities and threats, is represented using ontologies [1], providing a framework in which to extend threat trees to *Semantic Threat Graphs* (*STG*s).

Semantic Threat Graphs are used to model knowledge about threat mitigation by security configuration. We take the Open World Assumption [3], whereby Semantic Threat Graphs are easily extended to incorporate knowledge about the configuration of new threats and/or additional security mechanisms. For example, building a Semantic Threat Graph using existing firewall ontologies [25, 24] in order to describe specific SYN-proxy and SYN-threshold countermeasure configurations.

We use Semantic Threat Graphs to build a knowledge-base of best-practice defenses and security configurations against known threats. For example, bogon firewall rules [35, 49, 67] are best-practice protection against spoofing-threats for internal servers and end-user workstations, while NIST recommend multiple countermeasures over an n-tier network hosting a Web-server [64]. This knowledge-base is searchable—a suitable countermeasure can be found for a given threat—and provides the basis for autonomic security configuration.

This paper is a revised and extended version of the paper in [26]. Sec-

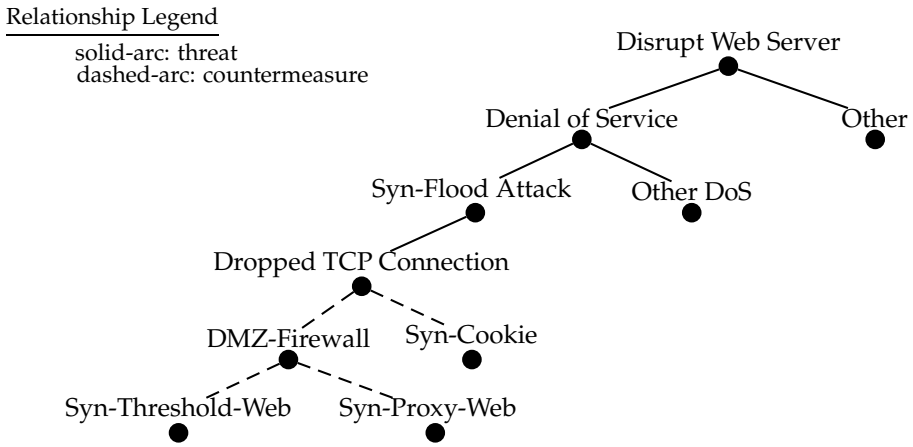
tion 2 provides an introduction to traditional threat trees and considers their limitations with respect to capturing low-level security configuration. Section 3 proposes Semantic Threat Graphs as a more natural approach for representing and reasoning about security configuration knowledge. Section 4 models security policy configurations as Semantic Threat Graphs. Semantic Threat Graphs are compared with traditional threat trees in Section 5. Section 6 considers how existing firewall configuration recommendations can be systematically encoded as Semantic Threat Graphs thereby providing catalogues of best-practice. The exercise of encoding these catalogues also provides a systematic evaluation of the effectiveness of Semantic Threat Graphs in practice. A case study involving a 3-tier firewall environment is used in Section 7 to illustrate the application of Semantic Threat Graphs to the automated synthesis and analysis of firewall configurations. Tool support and a preliminary prototype are outlined in Section 8. Section 9 discusses related research.

2 Threat Trees

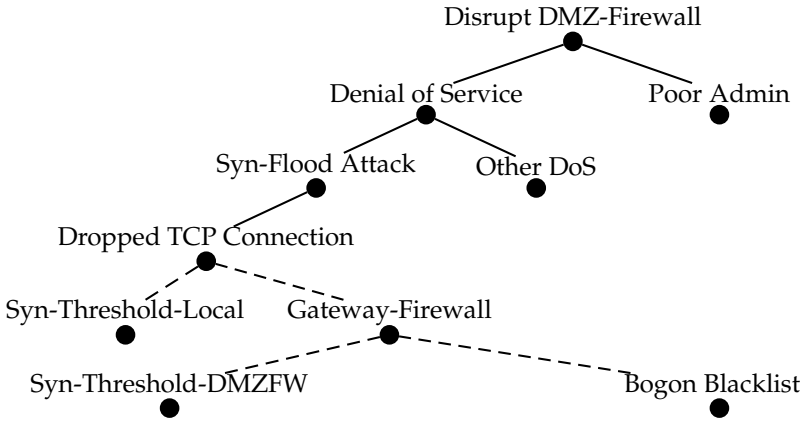
A *threat* can be defined as "*a potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm*" [59]. Threat trees, and similarly, *attack trees* [56], provide a semi-formal way of structuring the various threats that an asset may encounter. Various extensions of the threat tree paradigm have been developed. For example, the inclusion of countermeasures within the tree provides a new kind of tree called a *Defense Tree* [6] or *Protection Tree* [17]. For simplicity, and when no ambiguity arises, these approaches are collectively referred to as threat trees.

A threat tree is composed of a single root node that defines the primary threat to an asset (for example, 'Disrupt Web Server' in Figure 1a). A threat may be decomposed into additional fine-grained sub-threats (for example, 'Denial of Service'), thereby forming a tree hierarchy [39]. A *threat profile* can be described as the path from a leaf node to the root node which represents a specific set of states involved in either achieving the primary threat or countering it.

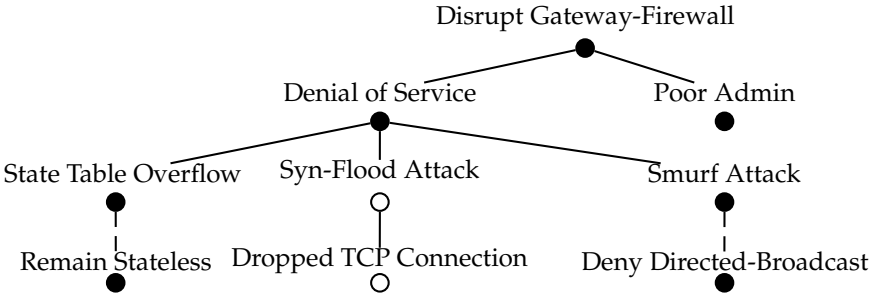
Threat trees provide a top-down approach to determining viable threat vectors (who, why and how a system can be compromised). In practice, these trees are used for threat elicitation and analysis—representing threats at a high-level of abstraction—and have tended not to be used to capture low-level or concrete security configuration detail. For example, while a



(a) Partial Threat Tree: Web Server Syn-Flood DoS Attack.



(b) Partial Threat Tree: DMZ Firewall Syn-Flood DoS Attack.



(c) Partial Threat Tree: Gateway Firewall Smurf DoS & State Table Attack.

Figure 1: Threat Tree Forest.

threat tree may identify a firewall countermeasure for a Denial of Service attack, a conventional threat tree is not effective, or intended to model, for example, distinctions between Syn-Proxy versus Syn-Threshold configurations [16] for a firewall in a subnet that is downstream from other similarly configured firewalls. In the latter case, much of the semantics of the threats and corresponding countermeasures are implicit and outside of the threat tree structure. Threat trees are useful for analysing a threat in a local context that is decomposed from some root threat. However, their advantages are diminished when one considers the threat in a global context (across multiple root threats), as described below.

Everything is a Threat. Each node in a threat tree is either a threat or a countermeasure. In practice, threats are not viewed in isolation and additional concepts must be implicitly encoded within the nodes of the tree. For example, in Figure 1a, various enterprise *Assets* are referenced: a Web server and firewall are implied by the 'Disrupt Web Server', 'DMZ-Firewall' nodes respectively. Similarly, the Web server's TCP/IP stack indicates an implicit *Vulnerability* in the 'Dropped TCP Connection' threat node (Figure 2). By viewing everything as a threat, implicit information may be overlooked.

Implicit Threat Relations. A threat tree decomposes threats and does not explicitly model other relationships between threats (or concepts related to threats). For example, in Figure 2, the 'Dropped TCP Connection' *exploits* (relationship) a TCP stack connection overflow vulnerability (implied concept) and *threatens* (relationship) the Web server (implied concept).

Cascading Threats. Countermeasures themselves may also have threats whereby the entity that protects another is itself vulnerable. Cyclic dependencies between disparate trees cannot be explicitly modelled using threat tree constraints. From Figure 1a, installing a firewall ('DMZ-Firewall') with configuration 'Syn-Threshold-Web' and/or 'Syn-Proxy-Web' will mitigate or reduce the threat of a 'Syn-Flood Attack' on the Web server. The 'Syn-Proxy-Web' countermeasure in effect shifts the threat posed to the Web server onto the firewall itself and thus the 'Syn-Flood Attack' has now indirectly migrated to 'DMZ-Firewall' giving rise to the threat tree outlined in Figure 1b. One of the ways that 'DMZ-Firewall' can be protected is for 'Gateway-Firewall' (an implicitly defined asset) to filter traffic via its 'Syn-Threshold-DMZFW' configuration, thereby giving rise to the implicit

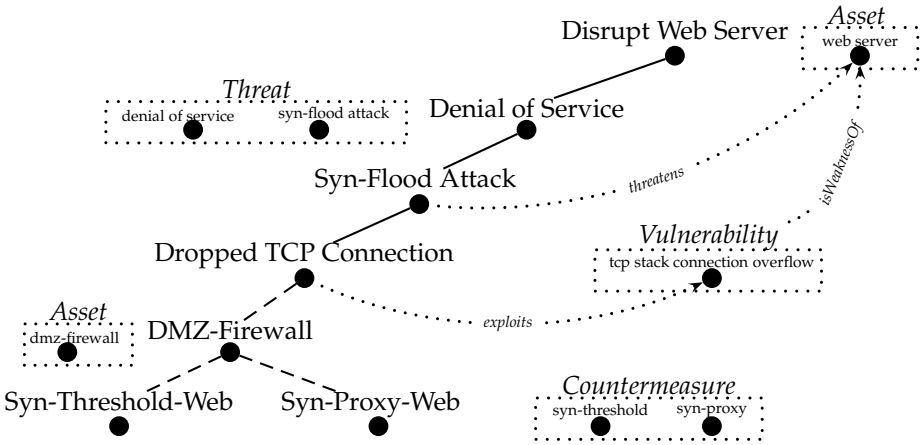


Figure 2: Threat Tree with Implicit Concepts, Individuals and Relationships.

dependency cycle. Should the 'Gateway-Firewall' implement the 'Syn-Threshold-DMZFW' countermeasure, then it subsequently controls the rate of TCP syn-based connection attempts towards the DMZ-Firewall. However, if that 'Syn-Threshold-DMZFW' countermeasure is a stateful firewall rule, then the 'Gateway-Firewall' runs the risk of an overflow of its stateful connection tracking table [30] ('State Table Overflow') while managing the Syn-based connection attempts. To avoid this scenario the 'Gateway-Firewall' should 'Remain Stateless'.

Unclear Threat Hierarchy. Threat trees are typically developed in isolation, that is, they focus on a single threat target (for example, decompose 'Disrupt Web Server' threat), and as a consequence it becomes difficult to inter-relate implicit information across multiple threat trees. Both 'Disrupt Web Server' and 'Disrupt DMZ-Firewall' (Figure 1) form part of a forest of (implicitly related) threat trees. By the definition of a tree, a node should have one parent in the tree and given that the threat tree structure allows for only one type of relationship (subsumption) a problem arises whereby the threat hierarchy becomes ambiguous when constructing the overall forest of trees. Either a new root node 'Disrupt Servers' is created where both 'Disrupt Web Server' and 'Disrupt DMZ-Firewall' are treated as disjoint siblings, or the 'Disrupt DMZ-Firewall' tree becomes a sub-node of the 'DMZ-Firewall' node within the 'Disrupt Web Server' threat tree. The language provided by conventional threat trees is not sufficient to state explicitly that the former disjoint sibling approach should be adopted with the

inclusion of a dependency relationship (for example, a *protects* or *threatens* relationship) that links the two trees together.

Threat Tree Reuse. Threat trees are not sufficiently expressive to capture the right level of explicit information to help make informed decisions as to how best to reuse portions of a tree. Consider the example in Figure 1. While it is perfectly acceptable to reuse the 'Denial of Service', 'Syn-Flood Attack' and 'Dropped TCP Connection' portion of the 'Disrupt Web Server' tree as part of the 'Disrupt DMZ-Firewall' tree, it cannot be generalised across all firewall threat trees. The 'Disrupt Gateway-Firewall' threat tree (Figure 1c) is not susceptible to a TCP stack connection overflow exploit ('Dropped TCP Connection'), indicated by the hollow tree nodes ('○'), as it does not examine TCP flags ('Remain Stateless') for TCP connections (Figure 1c). However, it may be susceptible to other kinds of denial of service threats, for example, a Smurf attack [59]. As a consequence, threat tree reuse is not straightforward.

3 Semantic Threat Graphs

A Semantic Threat Graph can be defined as a graph that represents the meaning of a threat domain. Intuitively, a Semantic Threat Graph makes explicit the information that is typically implicit in a threat tree. An abstract model of a Semantic Threat Graph that illustrates the concepts involved and their relationships is provided in Figure 3.

Semantic Threat Graphs are constructed in terms of an ontology [1]. An ontology provides a conceptual model of a domain of interest [22]. *Description Logic* (*DL*) is a formalism for representing knowledge (ontology) and belongs to a family of logic that represents a decidable portion of first-order logic [3]. Concepts represent sets of individuals and properties represent binary relations applied to individuals.

Enterprise IT assets are represented as individuals of the *Asset* concept. An asset may have one or more *hasWeakness*'s (property relationship) that relate to individuals categorised in the *Vulnerability* concept. Individuals of the *Vulnerability* concept are exploitable (*exploitedBy*) by a threat or set of threats (*Threat* concept). As a consequence, an asset that has a vulnerability is, therefore, also *threatenedBy* a corresponding *Threat*. A countermeasure *mitigates* particular vulnerabilities. Countermeasures are deemed to be kinds-of assets and thus are defined as a *subConceptOf Asset*. Figure 4 illustrates an example instantiation of the Semantic Threat Graph that explicitly

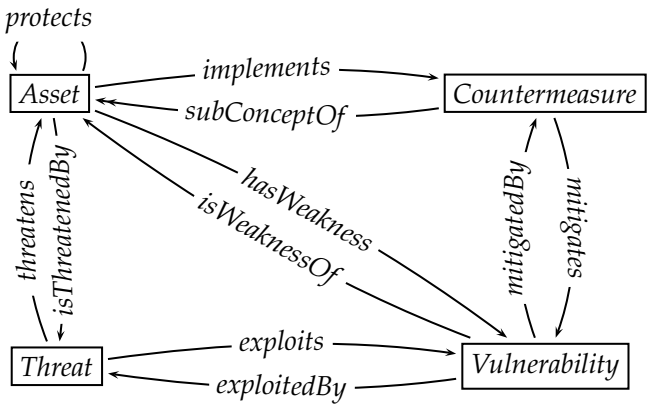


Figure 3: Abstract Semantic Threat Graph Model.

represents the threat tree example in the previous section. Regarding Figure 4, while countermeasures $\texttt{iptr}_{\texttt{SynThresholdWeb}}$ and $\texttt{iptr}_{\texttt{BogonBlacklist}}$ have the ability to mitigate vulnerabilities (dashed arc), they are not currently implemented (no *implements* property relationship) by the respective firewalls.

4 An Ontology for Semantic Threat Graphs

This section outlines a model for Semantic Threat Graphs encoded in terms of an ontology. Note that in presenting the model components, for reasons of space, we do not provide complete specifications in particular, definitions do not include disjoint axioms, sub-properties, data type properties or closure axioms.

Asset. Concept *Asset* represents any entity of interest within the enterprise that may be the subject of a threat. While assets can include people and physical infrastructure, this research only considers computer-system based entities such as Web servers, database servers, firewalls and so forth.

Individuals of concept *Asset* may have zero or more vulnerabilities ($\forall$ restriction) along property *hasWeakness*. As a result, those assets may be exposed to various individuals of the *Threat* concept. An asset may have the

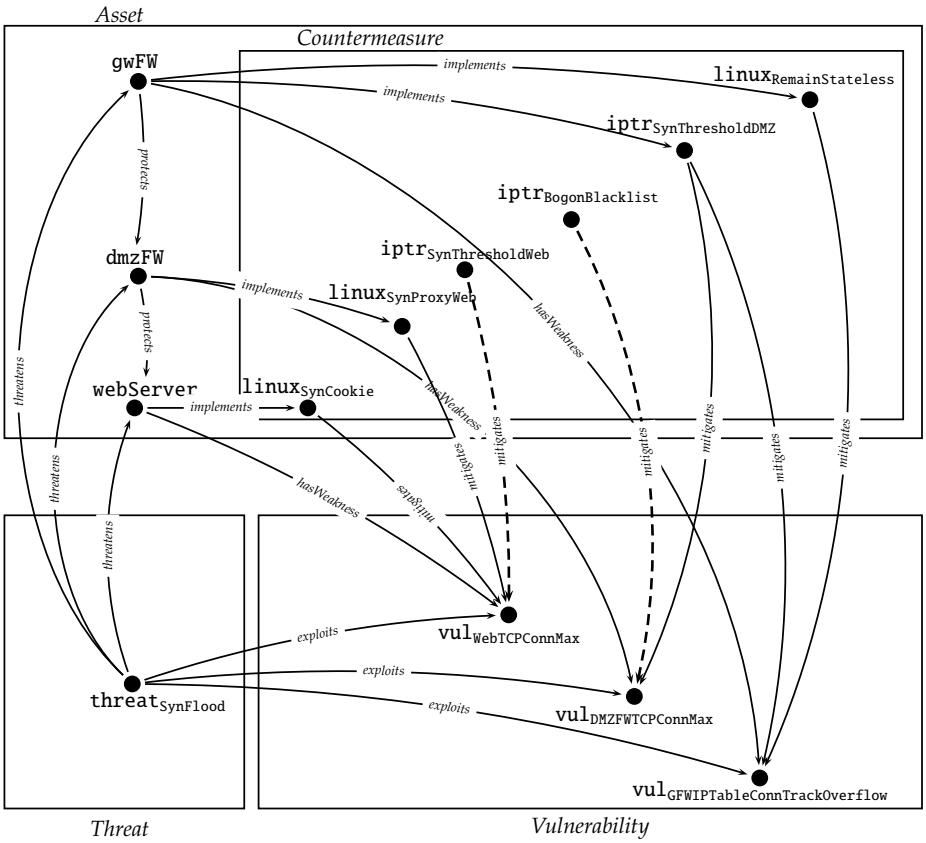


Figure 4: Fragment of Web Server and Firewall Semantic Threat Graphs.

capability to implement a countermeasure to protect itself or other assets.

$$\begin{aligned} \mathit{Asset} \quad \sqsubseteq \quad & \forall \mathit{hasWeakness}.\mathit{Vulnerability} \sqcap \\ & \forall \mathit{isThreatenedBy}.\mathit{Threat} \sqcap \\ & \forall \mathit{implements}.\mathit{Countermeasure} \end{aligned}$$

Concept *Asset* is further specialised to have more specific kinds of asset concepts. For example, $\mathit{BusinessServer}, \mathit{NetworkSecurityServer} \sqsubseteq \mathit{Server}$, where *Server* is a sub-concept of *Asset* and represents the set of servers (individuals) an enterprise may have. Figure 5 depicts a fragment of the *Asset* hierarchy.

An individual `webServer` of the *BusinessServer* concept (inferred as an individual of concept *Asset*) is vulnerable to a TCP stack connection overflow ($\mathtt{vul}_{\mathtt{WebTCPConnMax}}$) weakness. As a consequence, the `webServer`

isThreatenedBy a Syn-Flood attack represented as $\texttt{threat}_{\texttt{SynFlood}}$ individual. The following fragment of the ontology asserts these facts.

$$\begin{aligned} \mathit{Asset}(\texttt{webServer}) \quad \leftarrow \quad & \mathit{hasWeakeness}(\texttt{webServer}, \texttt{vul}_{\texttt{WebTCPConnMax}}) \sqcap \\ & \mathit{isThreatenedBy}(\texttt{webServer}, \texttt{threat}_{\texttt{SynFlood}}) \end{aligned}$$

The concept *NetworkSecurityServer*, represents the network access control systems within the network and have a role in protecting (*protects*) internal servers (including themselves). The *NetworkSecurityServer* concept definition further restricts the *implements* property by requiring a protection server to implement one or more ($\exists_{\geq 1}$) individuals of the *NACRule* concept (sub-concept of *Countermeasure*).

$$\begin{aligned} \mathit{NetworkSecurityServer} \quad \sqsubseteq \quad & \mathit{Server} \sqcap \\ & \exists_{\geq 1} \mathit{protects}.\mathit{Server} \sqcap \\ & \exists_{\geq 1} \mathit{implements}.\mathit{NACRule} \end{aligned}$$

The following ontology fragment, asserts that the gateway firewall (`gwFW`) protects the Web server from a Denial of Service by implementing a TCP syn-based rate-limit countermeasure (iptables rule).

$$\begin{aligned} \mathit{NetworkSecurityServer}(\texttt{gwFW}) \quad \leftarrow \quad & \mathit{protects}(\texttt{gwFW}, \texttt{webServer}) \sqcap \\ & \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{SynThresWeb}}) \end{aligned}$$

Note, in order to avoid explaining the low-level details of each ontology individual, human readable names are used to provide an intuition of its meaning. For example, individual $\texttt{iptr}_{\texttt{SynThresWeb}}$ can be interpreted as an iptables [28] rule (`iptr`) that limits the number of inbound TCP Syn packets destined for the Web server according to a specified threshold (`SynThresWeb`).

Threat. A threat is a potential for violation of security [59]. An individual of the *Threat* concept is considered to exploit one or more vulnerabilities ($\exists_{\geq 1}$ restriction).

$$\begin{aligned} \mathit{Threat} \quad \sqsubseteq \quad & \exists_{\geq 1} \mathit{exploits}.\mathit{Vulnerability} \sqcap \\ & \exists_{\geq 1} \mathit{threatens}.\mathit{Asset} \end{aligned}$$

For example, an individual of concept *SynFlood* (sub-concept of *Threat*), $\texttt{threat}_{\texttt{SynFlood}}$, threatens the `webServer`.

$$\begin{aligned} \mathit{Threat}(\texttt{threat}_{\texttt{SynFlood}}) \quad \leftarrow \quad & \mathit{exploits}(\texttt{threat}_{\texttt{SynFlood}}, \texttt{vul}_{\texttt{WebTCPConnMax}}) \sqcap \\ & \mathit{threatens}(\texttt{threat}_{\texttt{SynFlood}}, \texttt{webServer}) \end{aligned}$$

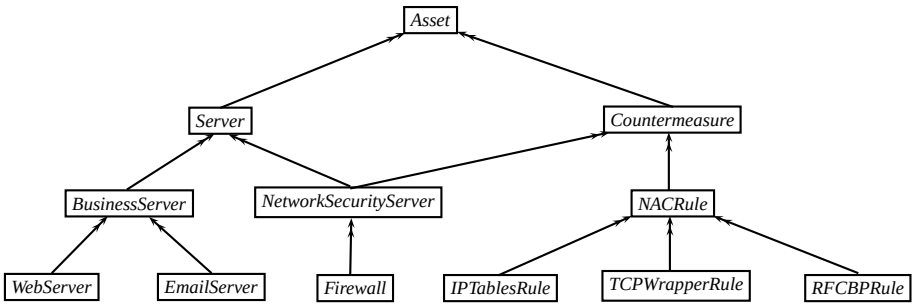


Figure 5: Fragment of Enterprise Asset Hierarchy.

Vulnerability. A vulnerability is a flaw or security weakness in an asset that has the potential to be exploited by a threat.

$$\begin{aligned} Vulnerability \quad \sqsubseteq \quad & \exists_{\geq 1} isExploitedBy.Threat \sqcap \\ & \exists_{\geq 1} isWeaknessOf.Asset \end{aligned}$$

For example, mis-configured firewall configurations have the potential to expose both internal servers and the firewalls themselves to threats. The following fragment in the ontology states that the `webServer` is susceptible to a $\texttt{threat}_{\texttt{SynFlood}}$ attack via the $\texttt{vul}_{\texttt{WebTCPConnMax}}$ weakness. Note, $\texttt{vul}_{\texttt{WebTCPConnMax}}$ is representative of a weakness in the TCP stack where it is possible surpass the maximum number of socket connections permitted by the TCP protocol due to a syn flood attack [30].

$$\begin{aligned} Vulnerability(\texttt{vul}_{\texttt{WebTCPConnMax}}) \leftarrow \; & isExploitedBy(\texttt{vul}_{\texttt{WebTCPConnMax}}, \texttt{threat}_{\texttt{SynFlood}}) \sqcap \\ & isWeaknessOf(\texttt{vul}_{\texttt{WebTCPConnMax}}, \texttt{webServer}) \end{aligned}$$

Countermeasure. A countermeasure is an action or process that mitigates vulnerabilities and prevents and/or reduces threats. A countermeasure is an asset.

$$Countermeasure \quad \sqsubseteq \quad Asset$$

Concept *NACRule* is representative of the network access-control rules that mitigate one or more vulnerabilities, provided they are implemented by *NetworkSecurityServer* individuals.

$$\begin{aligned} NACRule \quad \sqsubseteq \quad & Countermeasure \sqcap \\ & \exists_{\geq 1} mitigates.Vulnerability \sqcap \\ & \forall_{\geq 0} isImplementedBy.NetworkSecurityServer \end{aligned}$$

Ontologies have been developed for the Linux iptables firewall [28] and the TCP-Wrapper firewall [66], and are described in [22, 25].

Concepts *IPTablesRule* and *TCPWrapperRule* are sub-concepts of concept *NACRule* and represent the set of iptables and TCP-Wrapper rules respectively. An iptables rule, represented as individual $\texttt{iptr}_{\texttt{SynThresWeb}}$, mitigates the vulnerability $\texttt{vul}_{\texttt{WebTCPConnMax}}$ on the Web server (`webServer`).

$$\begin{aligned} \mathit{IPTablesRule}(\texttt{iptr}_{\texttt{SynThresWeb}}) \leftarrow \mathit{mitigates}(\texttt{iptr}_{\texttt{SynThresWeb}}, \texttt{vul}_{\texttt{WebTCPConnMax}}) \sqcap \\ \mathit{isImplementedBy}(\texttt{iptr}_{\texttt{SynThresWeb}}, \texttt{gwFW}) \end{aligned}$$

Threshold. This value is used to define the minimum degree of effectiveness of a countermeasure in mitigating the impact of a threat on an asset.

$$\mathit{Countermeasure} \sqsubseteq \exists_{=1} \mathit{minEffect}.\mathit{Threshold}$$

Simlarly, each threat has a threat level that defines the maximum impact on the asset.

$$\mathit{Threat} \sqsubseteq \exists_{=1} \mathit{maxImpact}.\mathit{Threshold}$$

For example, *Threshold* can be defined as enumerated class:

$$\mathit{Threshold} \sqsubseteq \{\texttt{high}, \texttt{medium}, \texttt{low}, \texttt{nil}\}$$

and a fragment in the ontology is

$$\begin{aligned} \mathit{Threat}(\texttt{threat}_{\texttt{SynFlood}}) \leftarrow \; & \mathit{exploits}(\texttt{threat}_{\texttt{SynFlood}}, \texttt{vul}_{\texttt{WebTCPConnMax}}) \sqcap \\ & \mathit{threatens}(\texttt{threat}_{\texttt{SynFlood}}, \texttt{webServer}) \sqcap \\ & \mathit{maxImpact}(\texttt{threat}_{\texttt{SynFlood}}, \texttt{high}) \end{aligned}$$

The threshold level characterizes the degree to which a countermeasure mitigates a threat: an asset is considered secure if the effectiveness (threshold) of the countermeasures are greater than the impact (threshold) of the corresponding threats they mitigate. For example, if $\texttt{iptr}_{\texttt{SynThresWeb}}$ was considered to have `medium` effectiveness at mitigating $\texttt{threat}_{\texttt{SynFlood}}$ then the asset remains under threat, albeit less threat than having no countermeasure. While vulnerability databases such as CVE [13] may provide suitable threshold metrics, the elicitation of threshold weightings is not the focus of this paper.

For the sake of clear exposition, not all property relationships involving assets, threats, vulnerabilities and countermeasures are described in

this paper. For example, while we assume servers have one IP address, in practice they may have multiple IP addresses. In addition, when the meaning is clear we do not define property inverse relationships; for example, *isExploitedBy* is assumed to be the inverse of property *exploits*.

5 Semantic Threat Graphs Assessment

Explicit Concepts and Relationships. Semantic Threat Graphs provide a framework to explicitly define concepts (other than a threat concept).

Cascading Threats. Taking the graph-based approach, cascading threats are identified explicitly within the Semantic Threat Graph ontology. Figure 4 demonstrates a threat that cannot be easily represented within a threat tree structure. Asserted relationships (for example, *protects*) define that the `webServer` is protected by the `dmzFW` which, in turn, is protected by the gateway firewall, `gwFW`, thus identifying the cascade threat. For simplicity, the scenario shown in Figure 4 models the $\texttt{threat}_{\texttt{SynFlood}}$ as the same threat for firewalls and systems. As a result of DMZ firewall, `dmzFW`, mitigating the $\texttt{vul}_{\texttt{WebTCPConnMax}}$ vulnerability on the `webServer` by way of a $\texttt{linux}_{\texttt{SynProxyWeb}}$ (for example, SQUID [69]), it then adopts that threat while proxying the Web servers TCP stream vulnerability.

Graph Hierarchies. Although threat trees provide hierarchies of the *Threat* concept, they are unable to define a hierarchy for the implicit concepts within the tree. The Semantic Threat Graph model presented in Figure 3 can be further refined with sub-concepts that are more refined than their parent concepts. For example, the *Asset* concept can define a sub-concept *Server* to represent the set of servers (individuals) an enterprise might have (Figure 5). This concept can in turn be further categorised as *BusinessServer* (containing Web, Email, Application servers and so forth) and *NetworkSecurityServer* (for example, Firewalls, IDS's, VPN's, Anti-Spam). The *Threat* concept, as an additional example, can define a number of sub-concepts in accordance with best practice, such as the Microsoft STRIDE standard whereby threats are categorised as (Figure 6): ***S**poofing identity*, ***T**ampering with data*, ***R**epudiation*, ***I**nformation disclosure*, ***D**enial of service* and ***E**levation of privilege* [31].

Graph Reuse. Poly-hierarchies are permitted in a Semantic Threat Graph, thus allowing the reuse of concepts. This is illustrated in Figure 5, where

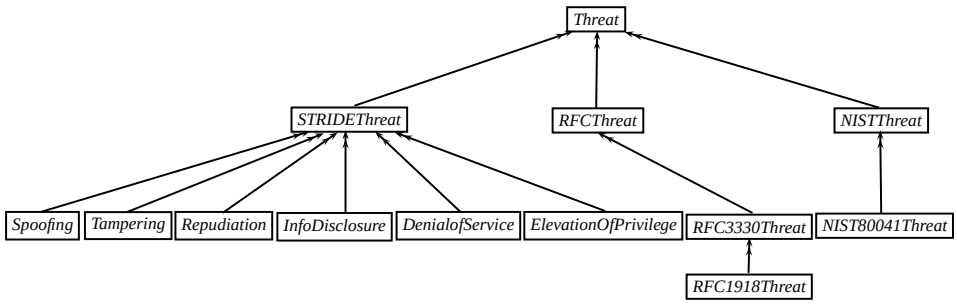


Figure 6: Fragment of Threat Hierarchy.

the *NerworkSecurityServer* concept has the *Server* concept and the *Countermeasure* concept as it's parent concepts. As a result, instances (for example, `dmzFW` and `gwFW`) of child concept *NetworkSecurityServer* are not only members of concept *Server* but also members of concept *Countermeasure*. As an other example, concept *RFC3330Threat* (Figure 6) represents known threats that should be prevented in accordance with best practice [35]. Individual $\texttt{threat}_{\texttt{Inbound127.0.0.0/8SrcIPPkt}}$, a member of concept *RFC3330Threat*, represents the following threat: "*127.0.0.0/8 - This block is assigned for use as the Internet host loopback address . . . no addresses within this block should ever appear on any network anywhere*" [35]. Individuals may be members of more than one concept that is not based on a subsumption relationship. For example, individual $\texttt{threat}_{\texttt{Inbound127.0.0.0/8SrcIPPkt}}$ is asserted or inferred (based on DL concept restrictions or SWRL [44] inference) to also be a member of concept *Spoofing*. Figure 4 illustrates explicitly the reuse of individuals, for example $\texttt{threat}_{\texttt{SynFlood}}$ and property relationships (*threatens* and *mitigates*).

6 Best Practice Catalogues

A best practice catalogue is a high-level document, written in natural language (typically English text), that defines a set of best practices (countermeasures) to protect sensitive and critical system resources. Conforming to best practice provides confidence that a security policy is upheld. For example, the National Institute of Standards and Technology (NIST) provide a set of recommended guidelines for securing public Web servers [64], recommending that "*all traffic between the Internet and Web server*" should be controlled and that "*all inbound traffic to the Web server except traffic which is required, such as TCP ports 80 (HTTP) and/or 443 (HTTPS)*" should be denied.

Best practice recommendations are typically countermeasure-centric where threats and vulnerabilities are implicit [22]. Thus, it is not always clear what the consequences are of not restricting access to the Web server over ports HTTP and HTTPS. Rather, the security administrator must draw upon his or her experience and/or additional information from other best practice recommendations such as [55] to conclude that the recommendation is intended to prevent the threat of unintended access to other services that the Web server may host. Furthermore, it is often a regulatory requirement for an enterprise to be compliant with best practice standards such as [12, 8, 37]. This further compounds the necessity to make explicit, the implicit knowledge about best practice recommendations in order to help generate effective network access control configurations. In this paper, a catalogue of low-level firewall configuration recommendations encoded as Semantic Threat Graphs is contructed from high-level best practice catalogues. An objective of this exercise was to evaluate the effectiveness of Semantic Threat Graphs in implementing a knowledge-base of real-world firewall configuration recommendations.

6.1 Semantic Threat Graphs for Best Practice Standards

Best practice standards for network access control, including PCI-DSS [12] for systems that process credit card information, NIST for secure Web-servers [64] and Internet RFCs for anti-bogon [35] have been encoded as Semantic Threat Graphs, providing a basis for firewall (iptables [28] and TCP-Wrapper [66]) configuration recommendations for known threats [22]. For the purposes of illustration in this paper the Semantic Threat Graph encoding of a fragment of NIST-800-41 [67], depicted in Table 1, is considered.

Firewall best practice, FBP-1 (Table 1), recommends that (spoofed) packets arriving on an external interface claiming to have originated from either of the three RFC1918 reserved internal IP address ranges should be dropped. Such traffic indicates a Denial of Service attack typically involving the TCP syn flag. Therefore, *Threat* individual $\texttt{threat}_{\texttt{Inbound192.168.0.0/16SrcIPPkt}}$ is asserted to be a member of sub-concepts *Spoofing*, *DenialOfService* and *RFC*1918*Threat*. Figure 6 illustrates a partial hierarchy of threats. A similar hierarchy is adopted for the corresponding vulnerability and countermeasure concepts.

For the sake of clarity, Table 1 excludes the relevant TCP-Wrapper countermeasures and instead illustrates the fourteen three-tuple relationships between threats, vulnerabilities and countermeasures with respect to the iptables firewall. Representing knowledge about the NIST-800-41 stan-

dard [67] in terms of TCP-Wrapper countermeasures requires a smaller number of three-tuple relationships between threats, vulnerabilities and countermeasures [22]. The reason for this, is that, TCP-Wrapper filters inbound traffic only (analogous to the iptables INPUT chain) and there are recommendations outlined by NIST-800-41 [67], for example FBP-2, for which TCP-Wrapper cannot provide that kind of network access control.

6.2 A Methodology for encoding STG Catalogues

This section describes the approach used to encode the best practice catalogues referenced in Table 2 in terms of Semantic Threat Graphs.

6.2.1 Effective Ontology Modelling

Our approach adheres to ontology engineering principles such as [1, 65, 3, 53], thus providing assurance that the level of granularity required by firewall configuration management is modelled within the Semantic Threat Graphs. That is, there are recommended guidelines for when to define a subconcept instead of an individual, a property instead of a concept and so forth. Similarly, these guidelines provide recommendations that help develop a model that avoids logical inconsistencies which is verified by Description Logic (DL) reasoning.

Ontology evaluation based on competency questions [29, 65] forms a core part of the ontology engineering best practice. Competency questions, that is, questions that are expected to be answered by the ontology, involve a top-down approach that define the scope of the domain of interest and a bottom-up approach that consider the level of granularity required. Both DL and SWRL reasoning is then performed on the ontology to verify that the Semantic Threat Graphs conform to the expected answers. The following are examples of such questions:

- What aspects of the firewall domain will the ontology model?
- What is the relationship between a countermeasure and a threat?
- Is best practice granularity sufficient to for low-level configuration?
- How effective is a countermeasure at mitigating a threat?

ID	Recommendation Description		
FBP-1	Deny *"Inbound or Outbound traffic from a system using a source address that falls within the address ranges set aside in RFC1918 as being reserved for private networks"* [67].		
	Threat	**Vulnerability**	**Countermeasure**
	$\texttt{threat}_{\texttt{Inbound192.168.0.0/16SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenInbound192.168.0.0/16PktToFW}}$	$\texttt{iptr}_{\texttt{DropIn192.168.0.0/16SrcIPPktInputChain}}$
	$\texttt{threat}_{\texttt{Outbound192.168.0.0/16SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenOutbound192.168.0.0/16PktFromFW}}$	$\texttt{iptr}_{\texttt{DropOut192.168.0.0/16SrcIPPktOutputChain}}$
	$\texttt{threat}_{\texttt{Inbound192.168.0.0/16SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenInbound192.168.0.0/16PktToHost}}$	$\texttt{iptr}_{\texttt{DropIn192.168.0.0/16SrcIPPktForwardChain}}$
	$\texttt{threat}_{\texttt{Outbound192.168.0.0/16SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenOutbound192.168.0.0/16PktFromHost}}$	$\texttt{iptr}_{\texttt{DropOut192.168.0.0/16SrcIPPktForwardChain}}$
	$\texttt{threat}_{\texttt{Inbound10.0.0.0/8SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenInbound10.0.0.0/8PktToFW}}$	$\texttt{iptr}_{\texttt{DropIn10.0.0.0/8SrcIPPktInputChain}}$
	$\texttt{threat}_{\texttt{Outbound10.0.0.0/8SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenOutbound10.0.0.0/8PktFromFW}}$	$\texttt{iptr}_{\texttt{DropOut10.0.0.0/8SrcIPPktOutputChain}}$
	$\texttt{threat}_{\texttt{Inbound10.0.0.0/8SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenInbound10.0.0.0/8PktToHost}}$	$\texttt{iptr}_{\texttt{DropIn10.0.0.0/8SrcIPPktForwardChain}}$
	$\texttt{threat}_{\texttt{Outbound10.0.0.0/8SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenOutbound10.0.0.0/8PktFromHost}}$	$\texttt{iptr}_{\texttt{DropOut10.0.0.0/8SrcIPPktForwardChain}}$
	$\texttt{threat}_{\texttt{Inbound172.16.0.0/12SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenInbound172.16.0.0/12PktToFW}}$	$\texttt{iptr}_{\texttt{DropIn172.16.0.0/12SrcIPPktInputChain}}$
	$\texttt{threat}_{\texttt{Outbound172.16.0.0/12SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenOutbound172.16.0.0/12PktFromFW}}$	$\texttt{iptr}_{\texttt{DropOut172.16.0.0/12SrcIPPktOutputChain}}$
	$\texttt{threat}_{\texttt{Inbound172.16.0.0/12SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenInbound172.16.0.0/12PktToHost}}$	$\texttt{iptr}_{\texttt{DropIn172.16.0.0/12SrcIPPktForwardChain}}$
	$\texttt{threat}_{\texttt{Outbound172.16.0.0/12SrcIPPkt}}$	$\texttt{Vul}_{\texttt{UnAuthenOutbound172.16.0.0/12PktFromHost}}$	$\texttt{iptr}_{\texttt{DropOut172.16.0.0/12SrcIPPktForwardChain}}$
ID	**Recommendation Description**		
FBP-2	Deny *"Inbound traffic containing ICMP (Internet Control Message Protocol) traffic"* [67].		
	Threat	**Vulnerability**	**Countermeasure**
	$\texttt{threat}_{\texttt{ICMPNetworkScan}}$	$\texttt{Vul}_{\texttt{InfoDisclosureICMPReplyPktFromFW}}$	$\texttt{iptr}_{\texttt{DropInICMPPktInputChain}}$
	$\texttt{threat}_{\texttt{ICMPNetworkScan}}$	$\texttt{Vul}_{\texttt{InfoDisclosureICMPReplyPktFromHost}}$	$\texttt{iptr}_{\texttt{DropInICMPPktForwardChain}}$

Table 1: Semantic Threat Graphs Extract for NIST-800-41: Guidelines on Firewalls and Firewall Policy.

Best Practice Catalogue	No. of Rec.	RFC2119-Style Classification			Completeness	
		Required	Optional	Additional	Concrete	Template
NIST SP800-41-rev1	16	14	1	1	7	8
NIST SP800-44v2	12	11	-	1	1	10
NIST SP800-45v2	10	10	-	-	-	9
NIST SP800-41	13	10	-	3	6	7
RFC 3330	18	6	-	12	18	-
RFC 1918	3	3	-	-	3	-
RFC 3920	2	-	2	-	-	2
PCI-DSS	11	8	-	3	1	9
XEP 0205	2	2	-	-	-	2
Total	**87**	**64**	**3**	**20**	**36**	**47**

Table 2: Summary of Best Practice Catalogues Evaluation.

6.2.2 Classification of Best Practice Recommendations

A RFC2119-style [7] approach was adopted to identify and categorise recommended firewall best practice. In [7], a set of key terms are defined to have an explicit meaning. In this research, a best practice recommendation is identified as either a *Required* recommendation, *Optional* recommendation or *Additional* recommendation.

Best practice recommendations that are expressed with terms, for example, *should* or *should not*, are categorised as a Required recommendation. The following RFC3330 best practice recommendation is identified as a Required recommendation: *"10.0.0.0/8 - This block is set aside for use in private networks. Its intended use is documented in RFC1918. Addresses within this block should not appear on the public Internet."* [35]. Similarly, recommendations identified with terms such as *may* or *might* are considered Optional recommendations. The following is an example of a NIST SP800-41-rev1 best practice recommendation, where connection-throttling is considered an Optional recommendation: *"A different type of firewall policy based on network activity is one that throttles or redirects traffic if the rate of traffic matching the policy rule is too high. For example, a firewall might redirect the connections made to a particular inside address to a slower route if the rate of connections is above a certain threshold."* [54]. Note, the set of identified NIST SP800-41-rev1 best practice RFC2119-style recommendations are outlined in [22].

An Additional recommendation, is defined to capture implicit best practice recommendations that are not explicitly identifiable in accordance with [7]. Implicit knowledge from the best practice document text and knowledge of other best practice standards are used to determine best practice recommendations of this category. For example, RFC3330 [35] explicitly defines six Required best practice recommendations regarding the filtering of IP address ranges (Table 2). However, it is also considered best practice to also restrict the use of the remaining (Additional) twelve network ranges when mitigating against the threat of denial of service or IP spoofing [67]. Table 4 presents the eighteen RFC3330 best practice recommendations according to the RFC2119-style classification.

Note, the best practice recommendations that do not explicitly use the terms synonymous with the RFC2119-style Required recommendations but do explicitly use 'action' terms such as *block* are also classified as Required recommendations. The second row of Table 3, provides other examples of 'action' terms (implicit should-do's). The following NIST SP800-44v2 best practice recommendation: "*Block all inbound traffic to the Web server except traffic which is required, such as TCP ports 80 (HTTP) and/or 443 (HTTPS)*" [64],

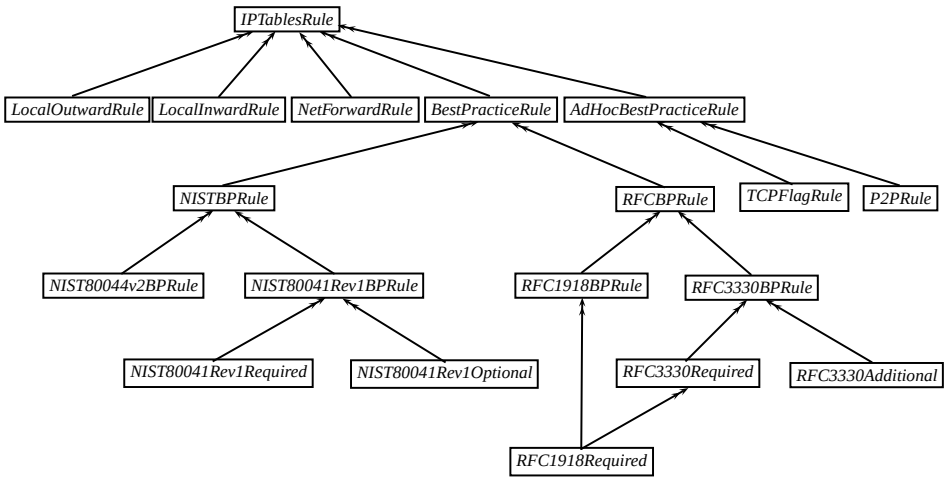


Figure 7: Sample Extract of the Best Practice Countermeasure Hierarchy.

is an example of a Required recommendation. This classification is also supported by the text that immediately precedes the list of specified recommendations, stating: "*To successfully protect a Web server using a firewall, . . . and is configured to perform the following*" [64].

The RFC2119-style classification provides a basis with which to construct a concept hierarchy of best practice recommendations (firewall rules). Figure 7 illustrates a fragment of the hierarchy. Countermeasures categorised as Required recommendations provide a basis to verify compliance with best practice. Confidence in a firewall configuration may be increased if Optional and/or Additional recommendations are also implemented.

6.2.3 Implicit Threats and Vulnerabilities in Best Practice Catalogues

While it maybe natural to consider vulnerability repositories such as [13] in terms of Semantic Threat Graphs, it is not always obvious how to interpret best practice catalogues in terms of assets, threats, vulnerabilities and countermeasures. For example in [67, 12, 54, 64, 35, 49], best practice recommendations tend to be countermeasure-centric and do not consider explicitly the threats and vulnerabilities within their text-based description. Fenz et. al. [19] encountered the same challenge when interpreting vulnerabilities and threats from the German IT Grundschutz Manual [8].

Ontology engineering best practice [1, 65, 3, 53] requires the informal enumeration of relevant document terms to provide a semantic context

Relevant Terms
filter, inspect, examine, matching, . . .
block, deny, drop, allow, accept, permit, . . .
source, destination, IP address, network mask, hostname, subnet, . . .
public, private, LAN, Internet, Intranet, inside, internal, external, . . .
direction, inbound, outbound, incoming, outgoing, ingress, egress . . .
service, application, ports, HTTP, SMTP, 443, 22. . .
malware, content, URL, keyword, worms . . .
spoofing, spoofed, Denial of Service, DoS . . .

Table 3: Excerpt of Best Practice Catalogue Key Terms.

in which concepts, individuals and their property relationships can be formally defined. Table 3 illustrates a sample set of relevant key terms extracted from the text describing best practice recommendations. Knowledge of the countermeasures being implemented and relevant terms were used to infer the threat and vulnerability concepts, individuals, and their associated relationships. Examples of inferred threat and vulnerability individuals that correspond to recommended low-level countermeasures are outlined in Table 1 and Table 5. An example STRIDE-based ontology concept hierarchy whereby threat individuals are classified is shown in Figure 6.

6.2.4 The Right Level of Abstraction

Modelling best practice recommendations that are at a sufficient level of granularity to represent low-level firewall configuration involves an ongoing Semantic Threat Graphs refinement process.

High-Level of Abstraction. Consider the RFC3330 [35] best practice catalogue that is described by a level of abstraction expressed in Table 4. Concepts *RFC*3330*Required* (Required recommendation) and *RFC*3330*Additional* (Additional recommendation) form part of the RFC3330 best practice Semantic Threat Graph hierarchy (Figure 7). Concept *RFC*3330*Required* is defined as an enumerated concept, containing six recommendations.

$$
\begin{aligned}
\mathit{RFC3330Required} \sqsubseteq\ & \mathit{RFC3330BPRule} \sqcap \\
& \{\mathtt{bprec}_{\mathtt{deny10.0.0.0/8}}, \mathtt{bprec}_{\mathtt{deny172.0.0.0/12}}, \\
& \mathtt{bprec}_{\mathtt{deny192.168.0.0/16}}, \mathtt{bprec}_{\mathtt{deny127.0.0.0/8}}, \\
& \mathtt{bprec}_{\mathtt{deny192.0.2.0/24}}, \mathtt{bprec}_{\mathtt{deny240.0.0.0/4}}\}
\end{aligned}
$$

The research carried out by Fenz et. al. [19, 21] in modelling the German IT Grundschutz Manual [8] and the ISO27001 Information Security Management System [37] catalogues operate at this level of abstraction. However, best practice recommendations at this level of abstraction, are not sufficient to represent threats, vulnerabilities and countermeasures required to synthesize and analyse low-level firewall configurations.

Low-Level of Abstraction. Semantic Threat Graphs that are capable of representing firewall configurations at the level of, for example IP addresses, ports and TCP flags, requires best practice recommendations to be further decomposed. Consider recommendation RFC3330-3 from Table 4. It explicitly specifies that the 192.168.0.0/16 IP address range should not appear on the Internet. However, there is no explicit indication as to what countermeasures are required (nor the threats or vulnerabilities giving rise to the countermeasure). For example, the text "*should not appear on the Public Internet*" may be interpreted to mean, from an offensive point of view, that a firewall must implement countermeasures to deny outbound packets within this range. Similarly, from a defensive point of view, it may be interpreted to mean that a firewall should implement countermeasures to deny the possibility of inbound packets within this IP address range. Thus, there is a certain degree of ambiguity when considering best practice countermeasures at low-levels of granularity and often requires knowledge from other sources. For example, recommendation FBP-1 outlined in Table 1 states that both inbound and outbound packets with a source address within the 192.168.0.0/16 IP address range should be denied. Therefore, to consider the 'direction' in which a packet is travelling in (inbound or outbound), requires concept *RFC3330Required* to be further specialised as concepts *RFC3330InboundRequired* and *RFC3330OutboundRequired*. The six abstract RFC3330 Required best practice individuals may be repesented as twelve Semantic Threat Graphs countermeasure individuals.

This level of abstraction in defining a granular concept hierarchy and its individuals is still incomplete. Other similar recommendations, for example, RFC3330-4 and RFC3330-18 are also best practice requirements of [67], where "*Inbound or Outbound network traffic containing a source or destination address of 0.0.0.0*" [67] and "*Inbound or Outbound network traffic containing a source or destination address of 127.0.0.1*" [67] state that the destination IP address within a packet header must also be considered. In order to represent a suitable best practice recommendation hierarchy that considers restricting combinations of 'source IP address' and 'destination

IP address', requires both concept *RFC3330InboundRequired* and concept *RFC3330OutboundRequired* to be further specialised.

$$
\begin{aligned}
\mathit{RFC3330InboundSrcIPRequired} &\sqsubseteq \mathit{RFC3330InboundRequired} \\
\mathit{RFC3330InboundDstIPRequired} &\sqsubseteq \mathit{RFC3330InboundRequired} \\
\mathit{RFC3330OutboundSrcIPRequired} &\sqsubseteq \mathit{RFC3330OutboundRequired} \\
\mathit{RFC3330OutboundDstIPRequired} &\sqsubseteq \mathit{RFC3330utboundRequired}
\end{aligned}
$$

Having defined a hierarchical structure that best reflects the RFC3330 Required recommendations when considering firewall configuration, individuals (iptables and TCP-Wrapper) of each sub-concept are then considered. For example, individual $\texttt{bprec}_{\texttt{deny192.168.0.0/16}}$, a high-level best practice recommendation, can now be refined as appropriate low-level firewall countermeasures. Thus, the six RFC3330 Required countermeasures are represented as forty-eight low-level iptables rules and six low-level TCP-Wrapper rules (countermeasures) [22]. A set of corresponding threats and vulnerabilities similar to those outlined in Table 1 are also defined.

6.2.5 Completeness of Semantic Threat Graphs

Best practice recommendations require for the most part customised 'tweaking' depending on the network in which they are applied. Therefore, modelling best practice recommendations means that not all assets, threats, vulnerabilities, countermeasures and their relationships are known in advance. Ontologies are based on Open World Assumption [3], thereby making it an ideal knowledge-based framework to model both known and unknown facts. For example, it may not be known in advance what the IP address range of a particular enterprise network is in which the catalogue is being applied to, or if a server is listening on a different port from the IANA [33] recommended default. Therefore, *template* Semantic Threat Graph individuals are defined as place holders for unknown knowledge. For example, NIST SP800-44v2 [64] best practice recommendation: "*Block all inbound traffic to the Web server except traffic which is required, such as TCP ports 80 (HTTP) and/or 443 (HTTPS)*", explicitly specifies the traffic direction (inbound) and service ports (HTTP and HTTPS) to be filtered, but the IP address of the Web server is currently unknown. A template Web server individual, `webServer`, and associated template iptables individuals ($\texttt{iptr}_{\texttt{AllowInHTTP}}$ and $\texttt{iptr}_{\texttt{AllowInHTTPS}}$) that are intended to permit HTTP(S) access, may have the following known facts:

ID	Recommendation Description	Type
RFC3330-1	*"10.0.0.0/8 - This block is set aside for use in private networks. . . . this block should not appear on the public Internet."* [35]	Required
RFC3330-2	*"172.16.0.0/12 - This block is set aside for use in private networks. . . . this block should not appear on the public Internet."* [35]	Required
RFC3330-3	*"192.168.0.0/16 - This block is set aside for use in private networks. . . . this block should not appear on the public Internet."* [35]	Required
RFC3330-4	*"127.0.0.0/8 - This block is assigned for use as the Internet host loopback address. . . . no addresses within this block should ever appear on any network anywhere."* [35]	Required
RFC3330-5	*"192.0.2.0/24 - This block is assigned as "TEST-NET" for use in documentation and example code. . . . this block should not appear on the public Internet."* [35]	Required
RFC3330-6	*"240.0.0.0/4 - This block, formerly known as the Class E address space, is reserved. . . . destination address 255.255.255.255 should never be forwarded . . . "* [35]	Required
RFC3330-7	*"14.0.0.0/8 - This block is set aside for assignments to the international system of Public Data Networks."* [35]	Additional
RFC3330-8	*"24.0.0.0/8 - This block was allocated in early 1996 for use in provisioning IP service over cable television systems."* [35]	Additional
RFC3330-9	*"39.0.0.0/8 - This block was used in the Class A Subnet Experiment."* [35]	Additional
RFC3330-10	*"128.0.0.0/16 - This block, corresponding to the numerically lowest of the former Class B addresses, was initially and is still reserved by the IANA."* [35]	Additional
RFC3330-11	*"169.254.0.0/16 - This is the "link local" block. It is allocated for communication between hosts on a single link."* [35]	Additional
RFC3330-12	*"191.255.0.0/16 - This block, corresponding to the numerically highest to the former Class B addresses, was initially and is still reserved by the IANA."* [35]	Additional
RFC3330-13	*"192.0.0.0/24 - This block, corresponding to the numerically lowest of the former Class C addresses, was initially and is still reserved by the IANA."* [35]	Additional
RFC3330-14	*"192.88.99.0/24 - This block is allocated for use as 6to4 relay anycast addresses"* [35]	Additional
RFC3330-15	*"198.18.0.0/15 - This block has been allocated for use in benchmark tests of network interconnect devices."* [35]	Additional
RFC3330-16	*"223.255.255.0/24 - This block, corresponding to the numerically highest of the former Class C addresses, was initially and is still reserved by the IANA."* [35]	Additional
RFC3330-17	*"224.0.0.0/4 - This block, . . . , is allocated for use in IPv4 multicast address assignments."* [35]	Additional
RFC3330-18	*"0.0.0.0/8 - Addresses in this block refer to source hosts on "this" network."* [35]	Additional

Table 4: RFC3330 Special-Use IPv4 Addresses— RFC2119-Style Evaluation.

$$
\begin{aligned}
\mathit{Asset}(\texttt{webServer}) \leftarrow\ & \mathit{hasIPAddress}(\texttt{webServer}, \texttt{-}) \sqcap \\
& \mathit{hasPort}(\texttt{webServer}, \texttt{80}) \sqcap \\
& \mathit{hasPort}(\texttt{webServer}, \texttt{443}) \\
\mathit{IPTablesRule}(\texttt{iptr}_{\texttt{AllowInHTTP}}) \leftarrow\ & \mathit{hasChain}(\texttt{iptr}_{\texttt{AllowInHTTP}}, \texttt{forward}) \sqcap \\
& \mathit{hasDstIP}(\texttt{iptr}_{\texttt{AllowInHTTP}}, \texttt{-}) \sqcap \\
& \mathit{hasProtocol}(\texttt{iptr}_{\texttt{AllowInHTTP}}, \texttt{tcp}) \sqcap \\
& \mathit{hasDstPort}(\texttt{iptr}_{\texttt{AllowInHTTP}}, \texttt{80}) \sqcap \\
& \mathit{hasAction}(\texttt{iptr}_{\texttt{AllowInHTTP}}, \texttt{accept}) \\
\mathit{IPTablesRule}(\texttt{iptr}_{\texttt{AllowInHTTPS}}) \leftarrow\ & \mathit{hasChain}(\texttt{iptr}_{\texttt{AllowInHTTP}}, \texttt{forward}) \sqcap \\
& \mathit{hasDstIP}(\texttt{iptr}_{\texttt{AllowInHTTPS}}, \texttt{-}) \sqcap \\
& \mathit{hasProtocol}(\texttt{iptr}_{\texttt{AllowInHTTPS}}, \texttt{tcp}) \sqcap \\
& \mathit{hasDstPort}(\texttt{iptr}_{\texttt{AllowInHTTPS}}, \texttt{443}) \sqcap \\
& \mathit{hasAction}(\texttt{iptr}_{\texttt{AllowInHTTPS}}, \texttt{accept})
\end{aligned}
$$

where "−" signifies that the range of a given property is currently unknown. Deploying the Web server within the network, means that it will be assigned an IP address, with the result of the template Web server and iptables rule individuals being modified to reflect this new knowledge.

Automatic Synthesis of Firewall Rules. As knowledge about assets, vulnerabilities and threats become known, it becomes possible to not only modify existing template firewall rules, but also to consider automatic synthesis of firewall rules. The following SWRL rule dynamically creates a set of iptables rules, using the *swrlx:makeOWLIndividual* built-in, that will protect assets from spoofing threats. Knowledge about an asset's IP address (*hasIPAddress*(?*asset*, ?*aip*)), the IP address range in which the threat of spoofing (*Threat*(?*spoof*)) has been identified, is used to synthesise specific firewall rules (*?specific*).

$$
\begin{aligned}
&\mathit{Asset}(?\mathit{asset}) \wedge \mathit{Threat}(?\mathit{spoof}) \wedge \mathit{Vulnerability}(?\mathit{ipPktForge}) \wedge \\
&\mathit{TemplateIPTablesRule}(\texttt{iptr}_{\mathit{temp}}) \wedge \mathit{hasWeakness}(?\mathit{asset}, ?\mathit{ipPktForge}) \wedge \\
&\mathit{exploits}(?\mathit{spoof}, ?\mathit{ipPktForge}) \wedge \mathit{mitigates}(\texttt{iptr}_{\mathit{temp}}, ?\mathit{ipPktForge}) \wedge \\
&\mathit{hasIPAddress}(?\mathit{asset}, ?\mathit{aip}) \wedge \mathit{hasSrcIPAddressStart}(?\mathit{spoof}, ?\mathit{sip}) \wedge \\
&\mathit{hasSrcIPAddressEnd}(?\mathit{spoof}, ?\mathit{eip}) \wedge \\
&\mathit{swrlx} : \mathit{makeOWLIndividual}(?\mathit{specific}, \texttt{iptr}_{\mathit{temp}}, ?\mathit{spoof}, ?\mathit{asset}, ?\mathit{ipPktForge}) \\
&\rightarrow \mathit{IPTablesRule}(?\mathit{specific}) \wedge \\
&\quad \mathit{hasSrcIPAddressStart}(?\mathit{specific}, ?\mathit{sip}) \wedge \\
&\quad \mathit{hasSrcIPAddressEnd}(?\mathit{specific}, ?\mathit{eip}) \wedge \\
&\quad \mathit{hasDstIPAddress}(?\mathit{specific}, ?\mathit{aip}) \wedge \\
&\quad \mathit{hasAction}(?\mathit{specific}, \texttt{drop}) \wedge \\
&\quad \mathit{mitigates}(?\mathit{specific}, ?\mathit{ipPktForge})
\end{aligned}
$$

Concrete Best Practice Recommendations. Best practice recommendations that can be considered concrete are those that can be enumerated or counted (subject to the required level of abstraction) as threats, vulnerabilities and countermeasures. Concrete best practice recommendations are typically applicable to networks in general, independent of service ports, IP addresses in use within a network, network topology and so forth. The RFC3330 best practice catalogue, presented in Table 4, is an example where each best practice recommendation can be completely defined within the Semantic Threat Graphs, thereby providing a countable set of recommendations. For example, Linux iptables filters traffic to and from the firewall itself, and traffic being forwarded to and from systems behind the firewall [22]. This results in the eighteen high-level RFC3330 best practice recommendations outlined in Table 4 to be configured as one hundred and forty-four low-level iptables firewall countermeasures with corresponding threats and vulnerabilities. That is, there are thirty-six firewall rules that inspect source IP and destination IP addresses which are applied to both the INPUT and OUTPUT chains. The FORWARD chain has seventy-two firewall rules to inspect source IP and destination IP addresses, where thirty-six firewall rules are applied to inbound traffic and the remaining thirty-six firewall rules are applied to the outbound traffic.

6.3 Evaluation of Semantic Threat Graphs for Best Practice

The effectiveness of the Semantic Threat Graphs approach was explored by encoding the real-world best practice catalogues referenced in Table 2. A total of eighty-seven best practice recommendations were identified using the RFC2119-style approach. These recommendations were modelled within the Semantic Threat Graphs as approximately two-hundred firewall rules with a corresponding number of threats and vulnerabilities. Note, a total of four best practice recommendations are not modelled directly within the ontology, as these recommendations were found to be redundant with respect to other recommendations within their respective best practice catalogue [22]. A fragment of the NIST SP800-41 best practice catalogue is presented in Table 1. Section 7 provides a scenario driven approach [43] whereby the best practice catalogues encoded as Semantic Threat Graphs can be reasoned over.

Modeling each best practice recommendation as a Semantic Threat Graph, involved a degree of subjectivity. Section 6.2.3 discussed how threats and vulnerabilities had to be manually inferred from the best practice recommendations. Similarly, there was a degree of ambiguity within the recommendations themselves. It is important to point out that the level of subjectivity, is not a shortcoming of the Semantic Threat Graphs approach, rather it arises as a consequence of interpreting the best practice recommendations either formally or informally. However, as new knowledge became known about best practice recommendations through modelling of additional catalogues, the Semantic Threat Graphs where refined (for example, Section 6.2.4) such that the level of subjectivity was minimised.

Many best-practice recommendations are incomplete in the sense that certain facts are unknown. The corresponding firewall rules are modeled as 'template' rules containing unbound variables, for facts such as IP address, and port. Some 43% of best practice rules are complete, while the remaining 56% are modelled as template rules. This is not a shortcoming of the Semantic Threat Graphs approach. On the contrary, because ontologies conform to the principles of Open World Assumption, it is an ideal framework to model and reason about incomplete knowledge.

7 Case Study: Network Access Control Configuration

A simplified e-commerce 3-tier firewall environment, depicted in Figure 8, is used to illustrate the use of Semantic Threat Graphs. The network at Tier-

1, also known as a Demilitarised Zone (DMZ), hosts both a Web server and an Email server that are accessible from the Internet via the gateway firewall. The gateway firewall implements a firewall configuration that permits packets from the Internet to both the Web server on ports HTTP and HTTPS (for example, $\texttt{iptr}_{\texttt{AllowInHTTP}}$), and to the Email server on port 25 (for example,$\texttt{iptr}_{\texttt{AllowInSMTP}}$). All other irrelevant packets are denied ($\texttt{iptr}_{\texttt{DefaultDenyPkt}}$). It is considered best practice [35, 49, 67] for a gateway firewall to also implement anti-bogon controls, for example $\texttt{iptr}_{\texttt{DropIn192.168.0.0/16SrcIPPktInputChain}}$, to protect its internal servers and end-user workstations from packets claiming to originate from the internal network. The following is a fragment of the firewall configuration implemented by the gateway firewall (gwFW).

$$
\begin{aligned}
&\mathit{NetworkSecurityServer}(\texttt{gwFW}) \leftarrow \\
&\quad \mathit{protects}(\texttt{gwFW}, \texttt{webServer}) \sqcap \\
&\quad \mathit{protects}(\texttt{gwFW}, \texttt{emailServer}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{AllowInHTTP}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{AllowOutHTTP}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{AllowInHTTPS}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{AllowOutHTTPS}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{DropIn192.168.0.0/16SrcIPPktInputChain}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{DropOut192.168.0.0/16SrcIPPktOutputChain}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{DropIn192.168.0.0/16SrcIPPktForwardChain}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{DropOut192.168.0.0/16SrcIPPktForwardChain}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{DropIn10.0.0.0/8SrcIPPktForwardChain}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{DropIn172.16.0.0/12SrcIPPktForwardChain}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{DropInAllSrcIPPktInputChain}}) \sqcap \\
&\quad \mathit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{DefaultDenyPkt}})
\end{aligned}
$$

Each countermeasure has an associated threat and vulnerability defined within the model. For example, $\texttt{iptr}_{\texttt{DropIn192.168.0.0/16SrcIPPktInputChain}}$ (individual), represents an iptables firewall rule to prevent spoofing attempts toward the firewall, is described in Table 1.

The Email server is protected by a locally hosted TCP-Wrapper firewall (emailFW). Note, the gateway firewall and the Email firewall currently do not implement a firewall configuration for SMTP access. Synthesis of a

suitable firewall configuration is explored in Section 7.1.

$$\begin{aligned} \mathit{NetworkSecurityServer}(\texttt{emailFW}) \leftarrow\ & \mathit{protects}(\texttt{emailFW}, \texttt{emailServer}) \sqcap \\ & \mathit{implements}(\texttt{emailFW}, \texttt{twr}_{\texttt{DefaultDenyPkt}}) \end{aligned}$$

The application firewall (appFW) implements a countermeasure, individual $\texttt{iptr}_{\texttt{AllowInWebServerIPToAppServerIPPortSSL}}$, to permit the Web server in Tier-1 to communicate with the application server Tier-2 over an SSL tunnel.

$$\begin{aligned} \mathit{NetworkSecurityServer}(\texttt{appFW}) \leftarrow & \\ & \mathit{protects}(\texttt{appFW}, \texttt{appServ}) \sqcap \\ & \mathit{implements}(\texttt{appFW}, \texttt{iptr}_{\texttt{AllowInWebServerIPToAppServerIPPortSSL}}) \sqcap \\ & \mathit{implements}(\texttt{appFW}, \texttt{iptr}_{\texttt{DefaultDenyPkt}}) \end{aligned}$$

Hosted in Tier-3 is the database server (dbServer). The back-end data firewall (dataFW) permits an SSH tunnel between the application server and the database server ($\texttt{iptr}_{\texttt{AllowInAppServerIPToDBServerIPPortSSH}}$). Note, in the interest of providing security in-depth, both the application firewall and the data firewall also implement anti-bogon countermeasures in accordance with [67]. However, for the sake of clarity, these are excluded from the DL assertions.

$$\begin{aligned} \mathit{NetworkSecurityServer}(\texttt{dataFW}) \leftarrow & \\ & \mathit{protects}(\texttt{dataFW}, \texttt{dbServer}) \sqcap \\ & \mathit{implements}(\texttt{dataFW}, \texttt{iptr}_{\texttt{AllowInAppServerIPToDBServerIPPortSSH}}) \sqcap \\ & \mathit{implements}(\texttt{dataFW}, \texttt{iptr}_{\texttt{DefaultDenyPkt}}) \end{aligned}$$

7.1 Firewall Configuration Synthesis

Involving threshold metrics as part of the synthesis process, ensures suitable countermeasures (firewall rules) that reduce the level of threat to an acceptable level are recommended.

Consider the following scenario, where an Email server (emailServer) has been deployed within the network (Figure 8). The firewalls (gwFW and emailFW) protecting the newly deployed emailServer still have to provision network access control. In this example, recommended countermeasures are inferred from the ontology catalogue for NIST-800-45v2 [63] which provides guidelines on securing Email servers. Table 5 provides a fragment of

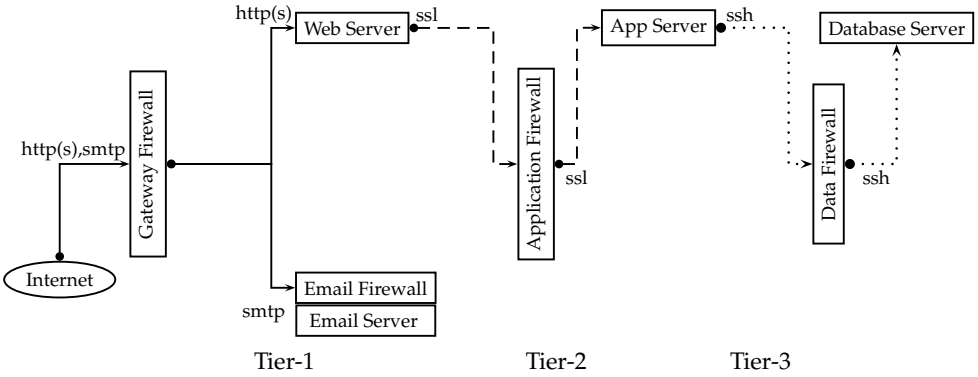


Figure 8: Example 3-Tier Enterprise E-Commerce Firewall Environment.

the NIST-800-45v2 recommendations encoded as Semantic Threat Graphs (ontology). In this paper, only the recommendations relevant to firewall access control are considered. How threats, vulnerabilities and countermeasures are interpreted from best practice guideline descriptions (English text), is not part of this discussion, and was explored in Section 6.2.

The Email server, `emailServer`, is identified to have the following vulnerabilities regarding intended inbound ($\texttt{Vul}_{\texttt{NoInboundSMTPAllowRule}}$) and outbound ($\texttt{Vul}_{\texttt{NoOutboundSMTPAllowRule}}$) access (Table 5). Thus, the `emailServer` is *threatenedBy* the threat of no external access ($\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$). The catalogue for NIST-800-45v2 provides a number of candidate firewall countermeasures depending on the firewall technology (iptables or TCP-Wrapper) that mitigate the vulnerabilities, thereby resolving the threat. Consider the iptables-based countermeasures that mitigate vulnerability $\texttt{Vul}_{\texttt{NoInboundSMTPAllowRule}}$ (Table 5a). Countermeasure $\texttt{iptr}_{\texttt{AllowInStatefulSMTPPkt}}$ is a stateful firewall rule that permits access to the SMTP port on the Email server. A stateless firewall rule that also provisions SMTP access is represented by countermeasure $\texttt{iptr}_{\texttt{AllowInStatelessTCPSMTPPkt}}$. The Email server may belong to an Email server farm, in which case, a firewall rule ($\texttt{iptr}_{\texttt{AllowInStatelessGenericEmailDstIPSMTPPkt}}$) that permits inbound SMTP traffic in general, may need to be considered.

The effectiveness (*hasMinEffect* property) of each countermeasure is asserted within the ontology. Table 6 illustrates the qualitative threshold metrics used in this running example. For example, firewall rules $\texttt{iptr}_{\texttt{AllowInStatefulSMTPPkt}}$ and $\texttt{iptr}_{\texttt{AllowOutStatefulSMTPPkt}}$ are considered effective (`high`) countermeasures that filter packets to and from the Email server.

The *Semantic Web Rule Language* (*SWRL*), complements DL by providing

ID	Guideline Description		
EBP-1	*"Block all inbound traffic to the mail server except traffic which is required,such as TCP ports 25"* [63].		
	Threat	**Vulnerability**	**Countermeasure**
	$\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$	$\texttt{Vul}_{\texttt{NoInboundSMTPAllowRule}}$	$\texttt{iptr}_{\texttt{AllowInStatefulSMTPPkt}}$
	$\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$	$\texttt{Vul}_{\texttt{NoOutboundSMTPAllowRule}}$	$\texttt{iptr}_{\texttt{AllowOutStatefulSMTPPkt}}$
	$\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$	$\texttt{Vul}_{\texttt{NoInboundSMTPAllowRule}}$	$\texttt{iptr}_{\texttt{AllowInStatelessTCPSMTPPkt}}$
	$\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$	$\texttt{Vul}_{\texttt{NoOutboundSMTPAllowRule}}$	$\texttt{iptr}_{\texttt{AllowOutStatelessTCPSMTPPkt}}$
	$\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$	$\texttt{Vul}_{\texttt{NoInboundSMTPAllowRule}}$	$\texttt{iptr}_{\texttt{AllowInStatelessGenericEmailSMTPPkt}}$
	$\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$	$\texttt{Vul}_{\texttt{NoOutboundSMTPAllowRule}}$	$\texttt{iptr}_{\texttt{AllowOutStatelessGenericEmailSMTPPkt}}$
	$\texttt{threat}_{\texttt{InboundUnintendedSrcIPPkt}}$	$\texttt{Vul}_{\texttt{InboundPromisciousAccess}}$	$\texttt{iptr}_{\texttt{DenyOtherPktInbound}}$
	$\texttt{threat}_{\texttt{InboundUnintendedSrcIPPkt}}$	$\texttt{Vul}_{\texttt{InboundPromiscuousAccess}}$	$\texttt{iptr}_{\texttt{DefaultDenyPkt}}$

(a) Linux iptables-based NIST-800-45v2 Best Practice Semantic Threat Graphs.

ID	Guideline Description		
EBP-1	*"Block all inbound traffic to the mail server except traffic which is required,such as TCP ports 25"* [63].		
	Threat	**Vulnerability**	**Countermeasure**
	$\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$	$\texttt{Vul}_{\texttt{NoInboundSMTPAllowRule}}$	$\texttt{twr}_{\texttt{AllowSMTPPkt}}$
	$\texttt{threat}_{\texttt{InboundUnintendedSrcIPPkt}}$	$\texttt{Vul}_{\texttt{InboundPromiscuousAccess}}$	$\texttt{twr}_{\texttt{DefaultDenyPkt}}$

(b) TCP-Wrapper-based NIST-800-45v2 Best Practice Semantic Threat Graphs.

Table 5: Semantic Threat Graphs Extract for NIST-800-45v2: Guidelines on Electronic Mail Security.

Threat	Impact
$\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$	high
Countermeasure	**Effectiveness**
$\texttt{iptr}_{\texttt{AllowInStatefulSMTPPkt}}$	high
$\texttt{iptr}_{\texttt{AllowOutStatefulSMTPPkt}}$	high
$\texttt{iptr}_{\texttt{AllowInStatelessTCPSMTPPkt}}$	medium
$\texttt{iptr}_{\texttt{AllowOutStatelessTCPSMTPPkt}}$	medium
$\texttt{iptr}_{\texttt{AllowInStatelessGenericEmailSMTPPkt}}$	low
$\texttt{iptr}_{\texttt{AllowOutStatelessGenericEmailSMTPPkt}}$	low
$\texttt{twr}_{\texttt{AllowSMTPPkt}}$	high

Table 6: Example Threshold Metric Assignment.

the ability to infer additional knowledge, but at the expense of decidability. SWRL rules are Horn-clause like rules written in terms of DL concepts, properties and individuals [44]. The following SWRL rule, states that if Internet Email clients cannot access the internal Email server (threat of $\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$) as a result of the currently implemented configurations of the protecting firewalls ($?fw$), then search the catalogue of best practice for recommended countermeasures (*?rule*) that reduce the threat to an acceptable level. Once suitable firewall countermeasures have been inferred, automatically assert those inferences (*implements*($?fw$, *?rule*)).

$$
\begin{aligned}
&\mathit{BusinessServer}(\texttt{emailServer}) \wedge \mathit{NetworkSecurityServer}(?fw) \wedge \\
&\mathit{NACRule}(?\mathit{rule}) \wedge \mathit{Threat}(\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}) \wedge \\
&\mathit{Vulnerability}(?\mathit{vul}) \wedge \\
&\mathit{threatenedBy}(\texttt{emailServer}, \texttt{threat}_{\texttt{NoExternalSMTPCommunication}}) \wedge \\
&\mathit{exploits}(\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}, ?\mathit{vul}) \wedge \mathit{mitigates}(?\mathit{rule}, ?\mathit{vul}) \wedge \\
&\mathit{hasWeakness}(\texttt{emailServer}, ?\mathit{vul}) \wedge \mathit{isProtectedBy}(\texttt{emailServer}, ?fw) \wedge \\
&\mathit{hasMaxImpact}(\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}, ?\mathit{timp}) \wedge \\
&\mathit{hasMinEffect}(?\mathit{rule}, ?\mathit{ceff}) \wedge \\
&\mathit{hasThresValue}(?\mathit{timp}, ?\mathit{tv}) \wedge \mathit{hasThresValue}(?\mathit{ceff}, ?\mathit{cv}) \wedge \\
&\mathit{swrlb} : \mathit{greaterThanOrEqual}(?\mathit{cv}, ?\mathit{tv}) \wedge \mathit{isExecutableOn}(?\mathit{rule}, ?\mathit{ruleType}) \wedge \\
&\mathit{operatesAs}(?fw, ?\mathit{fwType}) \wedge \mathit{swrlb} : \mathit{equal}(?\mathit{ruleType}, ?\mathit{fwType}) \\
&\rightarrow \mathit{implements}(?fw, ?\mathit{rule})
\end{aligned}
$$

Note, further explanation is required for SWRL variables *?ruleType* and *?fwType*. In order to get the correct countermeasures to apply to their

respective firewalls, each countermeasure has an *isExecutableOn* property that states what firewall technology it is applicable to. Each firewall has an *operatesAs* property to specify the kind of technology that it is, for example, the `gwFW` *operatesAs* "`iptables`" and firewall rule $\texttt{iptr}_{\texttt{AllowInStatefulSMTPPkt}}$ *isExecutableOn* "`iptables`". The resulting countermeasures inferred from the previous SWRL rule are asserted back into the ontology as new facts.

$$\begin{aligned} \textit{NetworkSecurityServer}(\texttt{gwFW}) \leftarrow & \\ & \textit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{AllowInStatefulSMTPPkt}}) \sqcap \\ & \textit{implements}(\texttt{gwFW}, \texttt{iptr}_{\texttt{AllowOutStatefulSMTPPkt}}) \end{aligned}$$

$$\textit{NetworkSecurityServer}(\texttt{emailFW}) \leftarrow \textit{implements}(\texttt{emailFW}, \texttt{twr}_{\texttt{AllowSMTP}})$$

7.2 Query Analysis

A firewall can be analysed to check whether or not the answers to queries made of its configuration are consistent with the security policy and/or compliant with recommended best practice. For example, '*Are internal systems protected from bogon spoofing attempts?*'.

7.2.1 Compliance-Based Firewall Configuration Analysis

A firewall configuration, particularly that of a gateway firewall, which may not be fully compliant with, for example, best practice outbound anti-bogon recommendations may result in the enterprise unwittingly participating in a zombie-oriented Denial of Service attack against other external networks, should any of its internal systems become compromised. The following SQWRL query, analyses the gateway firewall (`gwFW`) for compliance with NIST-800-41 [67] regarding the denial of bogon spoofing attempts.

$$\begin{aligned} & \textit{NetworkSecurityServer}(\texttt{gwFW}) \wedge \textit{Server}(?srv) \wedge \textit{NIST80041SpoofBPRule}(?rule) \wedge \\ & \textit{NIST80041SpoofThreat}(?ipSpoof) \wedge \textit{NIST80041SpoofVulnerability}(?ipPktForge) \wedge \\ & \textit{threatens}(?ipSpoof, ?srv) \wedge \textit{exploits}(?ipSpoof, ?ipPktForge) \wedge \\ & \textit{isWeaknessOf}(?ipPktForge, ?srv) \wedge \textit{isMitigatedBy}(?ipPktForge, ?rule) \wedge \\ & \textit{isProtectedBy}(?srv, \texttt{gwFW}) \wedge \textit{implements}(\texttt{gwFW}, ?rule) \\ & \rightarrow \textit{sqwrl} : \textit{select}(\texttt{gwFW}, ?rule) \end{aligned}$$

The result of this query is shown in Table 7, and illustrates the set of bogon countermeasures currently implemented by the `gwFW`. NIST-800-41 [67] recommends four best practice recommendations, for example FBP-1 in Table 1, that are intended to prevent spoofing and Denial of Service attempts.

These recommendations are modelled as thirty-two iptables countermeasures [22]. However, only six of these have been implemented by the gateway firewall. Further inspection of Table 7, shows that the recommended 192.168.0.0/16 IP address range has been successfully implemented.

Firewall	NIST 800-41 Best Practice iptables Rules
gwFW	$\texttt{iptr}_{\texttt{DropIn192.168.0.0/16SrcIPPktInputChain}}$
gwFW	$\texttt{iptr}_{\texttt{DropOut192.168.0.0/16SrcIPPktOutputChain}}$
gwFW	$\texttt{iptr}_{\texttt{DropIn192.168.0.0/16SrcIPPktForwardChain}}$
gwFW	$\texttt{iptr}_{\texttt{DropOut192.168.0.0/16SrcIPPktForwardChain}}$
gwFW	$\texttt{iptr}_{\texttt{DropIn10.0.0.0/8SrcIPPktForwardChain}}$
gwFW	$\texttt{iptr}_{\texttt{DropIn172.16.0.0/12SrcIPPktForwardChain}}$

Table 7: NIST-800-41 Compliance Analysis Report.

7.2.2 Threshold Satisfiability Firewall Configuration Analysis

Firewall configurations are analysed to determine their effectiveness at mitigating the identified threats to an acceptable level. An example of threats to the servers (Figure 8), asserted within the ontology, are identified in Table 8. The focus here is on the threat of permitting a set of clients (IP Addresses) unintended access. For example, a Web server that is accessible by spoofed localhost IP packets is considered to have the following $\texttt{threat}_{\texttt{Inbound127.0.0.0/8SrcIPPkt}}$ which, if exploited (threat impact of `high`), will have a considerable impact on the server.

A sample collection of firewall countermeasures currently protecting the servers is provided in Table 9 and considers a number of inadequate countermeasures implemented by the respective firewalls that expose the servers to unintended access. Note, an assumption is made that these countermeasures work in conjunction with the default deny countermeasure employed by each firewall in the 3-tier network.

The application server, `appServer`, is identified as having a *threatenedBy* relationship with $\texttt{threat}_{\texttt{InboundTier1SrcIPToAppSeverIPDstPortAll}}$, where inbound packets from the Tier-1 subnet that attempt to access all service ports represents a significant threat. The `appFW` firewall implements a countermeasure, $\texttt{iptr}_{\texttt{AllowInWebServerIPToTier2DstPortAll}}$, that enables spurious SSL access from the Web server to the Tier-2 subnet which includes intended SSL access to the application server. In effect, $\texttt{iptr}_{\texttt{AllowInWebServerIPToTier2DstPortAll}}$ permits the Web server access to all systems and services in Tier-2. Given that

this countermeasure, in conjunction with the default deny countermeasure, only partially prohibits the Tier-1 subnet from accessing unintended services on the `appServer`, it is considered to have a `low` effectiveness. While the spurious access of the Web server may be considered less of a threat in comparison to the entire Tier-1 subnet, a compromised Web server can now be used as a launch pad for attacks on systems in Tier-2. A more restrictive $\texttt{iptr}_{\texttt{AllowInWebServerIPToTier2DstPortAll}}$ countermeasure, for example $\texttt{iptr}_{\texttt{AllowInWebServerIPToAppServerIPDstPort443}}$, that limits the destination ports accessible on the application server to the SSL port only is required.

Implementing countermeasure $\texttt{iptr}_{\texttt{AllowInAppServerToDBServerIPDstPort22}}$ as part of the back-end `dataFW` firewall configuration would help in mitigating the threats that are also mitigated by firewalls upstream. Thus defence-in-depth is borne out as a result. However, in practice, managing network accesss control configurations is complex and there may be a requirement for multiple application servers within Tier-2 to communicate with the database server. Rather than defining specific countermeasures for each Tier-2 system, a security administrator may, in the knowledge of being protected by a firewall upstream, implement a generic countermeasure such as $\texttt{iptr}_{\texttt{AllowInTier2SrcIPToTier3DstPort22}}$, that provides blanket SSH access to the database server from all systems in Tier-2. Other non-bastion hardened servers (such as an Intranet LDAP server) in Tier-2 could then be used as launch pad when attacking the database server (inclusive of other Tier-3 servers), for example, an SSH brute force attack. Countermeasure $\texttt{iptr}_{\texttt{AllowInTier2SrcIPToTier3DstPort22}}$, while providing promiscuous access from Tier-2 to the database server, it does so across port SSH only. Therefore, it provides some effectiveness (`low`) at mitigating the threat of unintended access to all ports of the database server.

Since countermeasure $\texttt{iptr}_{\texttt{AllowInTier1IPToDBServerIPDstPort22}}$ directly permits SSH access from all Tier-1 systems to the database server, the `dataFW` firewall inadequately mitigates $\texttt{threat}_{\texttt{InboundTier1SrcIPToDBServerIPDstPort22}}$. It may be that remote database administration is a requirement through a DMZ-based VPN server. In this case, unintended access from the Tier-1, rather than just from the VPN server, through the `dataFW` may be considered to have less of a threat impact if one can guarantee that firewalls upstream, apply more restrictive port-forwarding controls. However, in keeping with the defence-in-depth, countermeasure $\texttt{iptr}_{\texttt{AllowInTier1IPToDBServerIPDstPort22}}$ should be refined to have a more restrictive access control.

The Email server, `emailServer`, should not receive packets from the Web server. Therefore, the `emailServer` is considered to be *threatenedBy* the $\texttt{threat}_{\texttt{InboundWebServerIPPkt}}$ individual. The TCP-Wrapper firewall, `emailFW`,

Asset	Threat	Max. Impact
webServer	$\texttt{threat}_{\texttt{Inbound127.0.0.0/8SrcIPPkt}}$	high
emailServer	$\texttt{threat}_{\texttt{InboundWebServerIPPkt}}$	high
appServer	$\texttt{threat}_{\texttt{Inbound127.0.0.0/8SrcIPPkt}}$	high
appServer	$\texttt{threat}_{\texttt{InboundTier1SrcIPToAppSeverIPDstPortAll}}$	medium
dbServer	$\texttt{threat}_{\texttt{Inbound127.0.0.0/8SrcIPPkt}}$	high
dbServer	$\texttt{threat}_{\texttt{InboundTier1SrcIPToDBServerIPDstPort22}}$	low
dbServer	$\texttt{threat}_{\texttt{InboundTier2SrcIPToDBServerDstPortAll}}$	medium

Table 8: Example Threats of Unintended IP Address Access.

implements countermeasure $\texttt{twr}_{\texttt{AllowSMTP}}$. While this countermeasure is considered effective at mitigating $\texttt{threat}_{\texttt{NoExternalSMTPCommunication}}$ outlined in Section 7.1, it is considered ineffective at mitigating a Web server threat from within Tier-1, $\texttt{threat}_{\texttt{InboundWebServerIPPkt}}$. Therefore, an exclusion countermeasure that specifically denies access to the Web server is required.

The following SQWRL query analyses the overall firewall configuration in terms of countermeasure effectiveness outlined in Table 9. It reports if countermeasures (?*rule*) implemented by a firewall (?fw) to protect vulnerable assets (?*bs*) are not effective ($cv < tv$) at mitigating the identified threats (?*threat*) illustrated in Table 8.

$$
\begin{aligned}
&\mathit{BusinessServer}(?bs) \wedge \mathit{Firewall}(?fw) \wedge \mathit{NACRule}(?rule) \wedge \mathit{Threat}(?threat) \wedge \\
&\mathit{Vulnerability}(?vul) \wedge \mathit{threatenedBy}(?bs, ?threat) \wedge \mathit{exploits}(?threat, ?vul) \wedge \\
&\mathit{mitigates}(?rule, ?vul) \wedge \mathit{hasWeakness}(?bs, ?vul) \wedge \mathit{isProtectedBy}(?bs, ?fw) \wedge \\
&\mathit{hasMaxImpact}(?threat, ?timp) \wedge \mathit{hasMinEffect}(?rule, ?ceff) \wedge \\
&\mathit{hasThresValue}(?timp, ?tv) \wedge \mathit{hasThresValue}(?ceff, ?cv) \wedge \\
&\mathit{swrlb} : \mathit{lessThan}(?cv, ?tv) \wedge \mathit{isExecutableOn}(?rule, ?ruleType) \wedge \\
&\mathit{operatesAs}(?fw, ?fwType) \wedge \mathit{swrlb} : \mathit{equal}(?ruleType, ?fwType) \\
&\rightarrow \mathit{sqwrl} : \mathit{select}(?bs, ?threat, ?fw, ?rule, ?ceff, ?timp)
\end{aligned}
$$

8 Tool Support and Prototype

The ontologies for the management of network access control configuration discussed in this research where implemented in OWL-DL, a language subset of OWL which is a W3C standard that includes DL reasoning semantics [1]. *Protégé* is a plug-and-play knowledge acquisition framework

Firewall	Countermeasure	Min. Effectiveness
gwFW	$\texttt{iptr}_{\texttt{DropIn127.0.0.0/8SrcIPPktForwardChain}}$	high
emailFW	$\texttt{twr}_{\texttt{AllowSMTP}}$	low
appFW	$\texttt{iptr}_{\texttt{DropIn127.0.0.0/8SrcIPPktForwardChain}}$	high
appFW	$\texttt{iptr}_{\texttt{AllowInWebServerIPToTier2DstPortAll}}$	low
dataFW	$\texttt{iptr}_{\texttt{DropIn127.0.0.0/8SrcIPPktForwardChain}}$	high
dataFW	$\texttt{iptr}_{\texttt{AllowInTier1IPToDBServerIPDstPort22}}$	medium
dataFW	$\texttt{iptr}_{\texttt{AllowInTier2SrcIPToTier3DstPort22}}$	low

Table 9: Example Firewall Countermeasures & Mitigation Effectiveness

that provides a graphical ontology editor and an ontology API [27]. Protégé interfaces with a DL based reasoner called *Pellet* providing model classification and consistency [46]. In conjunction to DL reasoning support, the SWRL Protégé plug-in (SWRLTab), allows for the creation of horn-like logic rules that interfaces with an expert system called *Jess* [44]

The primary focus of this paper is to describe how knowledge about security configuration and best practice recommendations and their relationships can be modeled, queried and reasoned over within a Semantic Threat Graph ontology. Related research [23] considers how these results can be applied to the management of XMPP application-level and firewall-level access controls. A preliminary prototype provides a semi-automated configuration agent that represents the current configuration state (iptables and XMPP controls) as an ontology. High level administration requests, such as XMPP server federation [52] map to queries over the configuration ontology with best-practice (ontologies) used to guide the automated generation of new firewall and XMPP policy rules.

9 Related Research

Threat trees have been extended in a number of ways [6, 10, 17, 45]. Additional boolean node operators: *NAND, XOR* and *NOR*, with the incorporation of *defence nodes* as countermeasures is described by [45]. Edge et. al. [17] define a Protection Tree and Bistarelli et. al. [6] define a Defense Tree as countermeasure-centric extensions to the threat tree approach. The research carried out by [10] describes an *Enhanced Attack Tree (EAT)* that supports temporal dependencies and sequential threat events that must occur for an attack to be successful. Threat trees have also been extended to provide qualitative [56] and quantitative [6, 17, 15, 9] metrics for risk analysis.

For example, Bistarelli et. al. [6, 5] provide economic-based return on investment quantitative risk model that assigns risk attributes to threats and countermeasures within the Defense Tree. The return on investment metric is used as a measure of the effectiveness of a specific security investment in a countermeasure with respect to a corresponding threat [5].

While these extended threat trees are aimed at resolving particular inadequacies within the conventional threat tree model, they still operate at rather high-levels of abstraction and are limited with regard to viable threat-only-vectors that contain implicit information such as assets, vulnerabilities and so forth. The approach presented in this paper differs by extending the threat tree model to include semantic knowledge about fine-grained security configuration and, how it relates to assets, threats, vulnerabilities and countermeasures. Thus, the Semantic Threat Graph approach makes explicit the information that is typically implicit in a threat tree.

An *attack graph* models knowledge about the inter-dependency between system vulnerabilities and knowledge about the network topology in terms of a graph-based data structure [68, 48, 47]. Model checking analysis techniques are typically used to determine all possible sequences of an attack against a system [70, 40, 57]. While threat trees typically focus on the consequence of an attack, attack graphs typically focus on the attacker activity and his/her interaction with the system under threat [48].

Attack graphs are typically classified as one of the following [34]. A *connection-oriented attack graph* consists of nodes that represent system and network states; for example, host system, services, network connectivity and user privilege levels. Graph edges are used to represent exploits which denote state transitions. Both [47, 57] illustrate examples of condition-oriented attack graphs. An *exploit-oriented attack graph* (*exploit dependency graph* [2]), is the opposite of a condition-oriented attack graph with respect to nodes and edges. A *condition-exploit-oriented attack graph* [42] consists of nodes that represent both states and exploits where an edge represents the relationship between states and exploits [34].

Dacier et al. [14] present the concept of a privilege graph, where nodes represent a set of Unix-based privileges owned by a user and edges represent Unix system vulnerabilities. Dacier et al. also present a probabilistic approach thats assesses the likelihood of particular attack sequences. Philips et al. [47] similarly consider a probabilistic approach that determines the likelihood of an attack in terms of the 'effort' required by an attacker to exploit the sequence of identified vulnerabilities. A number of existing research approaches consider the development of automated tool support for the synthesis and analysis of attack graphs, for example [57, 42, 36]. Attack

graphs tend to be considerably more complex to manage than threat trees with regard to scalability [48, 36]. A number of existing research approaches consider the management of attack graphs. For example, Mehta et al. [40] investigate a number of ranking techniques, such as Google page ranking, as a means of identifying and presenting relevant portions of the attack graph to the security administrator. In [42], Noel et al. present a visual framework for managing attack graph complexity that uses 'hierarchical graph aggregation techniques' where non-overlapping sub-graphs are recursively collapsed to single abstract nodes. In [68], Wang et al. state that analysis of attack graphs are primarily based on proprietary algorithms. Therefore, interactive analysis, similar to decision support systems, is constrained by the current algorithm implementation. Wang et al. present a relational database model as a basis to perform interactive analysis of attack graphs. An in-depth discussion on attack graphs is beyond the scope of this paper and the reader is referred to [34] and [58] for further information.

Attack graphs are intended to operate at higher levels of abstraction than Semantic Threat Graphs. While attack graphs represent knowledge that is typically implicit within threat threes, the representation of such knowledge may overburden the system administrator with respect to what is intended to be conveyed. For example in [47] a single node may be used to model attacker threat capabilities and system attributes such as operating system, services, vulnerabilities, CVE information. However, using the Semantic Threat Graphs approach such knowledge is explicitly modelled in terms of individual threats, assets, vulnerabilities and their corresponding relationships.

Herzog et. al. [32] and Fenz et. al. [18, 20, 21] also consider the use of ontologies for threat trees. However, these works are intended to operate at higher levels of abstraction than what is intended to be represented by Semantic Threat Graphs [22]. How their research differs to that presented in this paper is discussed in [22].

10 Conclusion and Discussion

This paper outlined a threat management approach using an ontology to construct, reason about and manage security policy configurations in the context of Semantic Threat Graphs. A number of disadvantages are identified when using conventional threat trees to model threats to an enterprise at low-levels of granularity. In a threat tree, everything is considered a threat and as a consequence implicit concepts and implicit concept relations

maybe overlooked. Adopting the Semantic Threat Graph approach makes the implicit knowledge explicit. A threat tree only provides a hierarchy of the *Threat* concept. It lacks the capability to also include a hierarchy for the implicit concepts, for example vulnerabilities, within the tree. Semantic Threat Graphs enable explicit taxonomic hierarchies to be systematically developed. Cascading threats between disparate threat trees cannot be explicitly represented using threat tree constraints. However, modeling threats within the Semantic Threat Graph explicitly identifies any threat or vulnerability dependencies. In summary, threat trees are used to represent threats at a high-level of abstraction, their singular threat vector focus and their practical suitably in a localised context (individual trees) do not explicitly capture all the entities involved in the threat management process. The Semantic Threat Graph extends the threat tree model to include semantic knowledge about low-level security configurations.

A case study involving a distributed and heterogeneous firewall environment, demonstrated how security configurations can be analysed using knowledge about the countermeasures effectiveness in mitigating threats and how automated security mechanism configuration recommendations can be made based on catalogues of best-practice countermeasures. Emphasis on dynamic elicitation of threshold weightings needs to be considered as part of future work. Note, in [22] the ontology for Semantic Threat Graphs is extended to include knowledge about qualitative [62] and quantitative [60] risk-based metrics.

A number of best practice standards that make recommendations about how firewalls should be configured in practice were encoded as Semantic Threat Graphs. This served two purposes. The first, ensured that the knowledge-base of firewall rules developed were representative of firewall rules used in practice. The second was to evaluate the feasibility of the Semantic Threat Graphs approach to construct a knowledge-base of real-world detailed firewall configuration recommendations. Semantic Threat Graphs have successfully been used to model and reason about the following best practice standards: NIST-800-41 [67], NIST-800-41rev1 [54], NIST-800-45v2 [63], NIST-800-44v2 [64], RFC1918 [49], RFC3330 [35], RFC3920 [50], PCIDSS-v1.2.1 [12] and XEP-0205 [51]. The current ontology is populated with around two-hundred threat, vulnerability and countermeasure combinations providing a relatively small catalogue for evaluation purposes. A RFC2119-style approach was developed to identify and categorise recommended firewall best practice. This approach has been shown to be effective at identifying implicit recommendations that may not have been identified by a security administrator.

Future research will capture knowledge of additional best practice standards, for example [30, 16, 41]. A security administrator will typically draw upon ad-hoc recommendations that are based on general firewall literature described at a system-level [22]. Thus, integrating ad-hoc recommendations with the existing standards-based recommendations knowledge-base should also be considered. By developing additional ontologies, for example IDS [11], one can more effectively reason about firewall configurations.

Future research should consider the automatic population of the Semantic Threat Graphs with threats and vulnerabilities from repositories such as CVE [13], and from vulnerability scanners, such as Nessus [4]. An ontology for network discovery, for example Nmap [38], that is automatically populated with knowledge about the network topology could be used to automate the assignment of IP addresses, ports and Operating System type to network resources (assets) encoded within the Semantic Threat Graphs. Our current evaluation suggests that representing knowledge about catalogues of best practice within Semantic Threat Graphs is reasonable.

Acknowledgments. This research has been supported by Science Foundation Ireland grant 08/SRC/11403.

References

[1] Dean Allemang and Jim Hendler. *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*. Morgan Kaufmann, May 2008.

[2] Paul Ammann, Duminda Wijesekera, and Saket Kaushik. Scalable, Graph-based Network Vulnerability Analysis. *9th ACM conference on Computer and Communications Security (CCS), Washington, DC, USA*, November 2002.

[3] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, March 2003.

[4] Jay Beale. *Nessus Network Auditing, 2nd Ed.* Syngress, November 2007.

[5] Stefano Bistarelli, Marco Dall'Aglio, and Pamela Peretti. Strategic Games on Defense Trees. *4th International Workshop on Formal Aspects in Security and Trust (FAST), Hamilton, Ontario, Canada*, August 2006.

[6] Stefano Bistarelli, Fabio Fioravanti, and Pamela Peretti. Defense trees for Economic Evaluation of Security Investments. *1st International Conference on Availability, Reliability and Security (ARES), Vienna, Austria*, April 2006.

[7] Scott Bradner. RFC2119: Key words for use in RFCs to Indicate Requirement Levels. *http://ietf.org*, March 1997.

[8] BSI. IT Grundschutz Manual. *https://www.bsi.bund.de*.

[9] Ahto Buldas, Peeter Laud, Jaan Priisalu, Mart Saarepera, and Jan Willemson. Rational Choice of Security Measures via Multi-Parameter Attack Trees. *1st International Workshop on Critical Information Infrastructures Security (CRITIS), Volume 4347 of LNCS, Greece*, 2006.

[10] Seyit Ahmet Camtepe and Bulent Yener. Modeling and Detection of Complex Attacks. *3rd International Conference on Security and Privacy in Communications Networks, SecureComm, Nice, France*, September 2007.

[11] Brian Caswell, Jay Beale, and Andrew Baker. *Snort IDS and IPS Toolkit*. Syngress, February 2007.

[12] PCI Security Standards Council. Payment Application Data Security Standard: Requirements and Security Assessment Procedures Version 1.2.1. *Payment Card Industry (PCI) Data Security Standard*, July 2009.

[13] CVE. Common Vulnerabilities and Exposures. *http://cve.mitre.org/*.

[14] Marc Dacier, Yves Deswarte, and M. Mohamed Kaâniche. Models and Tools for Quantitative Assessment of Operational Security. *12th International Information Security Conference (IFIP/Sec)*, May 1996.

[15] Rinku Dewri, Nayot Poolsappasit, Indrajit Ray, and Darrell Whitley. Optimal Security Hardening using Multi-Objective Optimization on Attack Tree Models of Networks. *14th ACM Conference on Computer and Communications Security (CCS), VA, USA*, pages 204–213, 2007.

[16] Wesley M. Eddy. RFC 4987: TCP SYN Flooding Attacks and Common Mitigations. *http://ietf.org*, August 2007.

[17] Kenneth Edge, Richard Raines, Michael Grimaila, Rusty Baldwin, Robert Bennington, and Christopher Reuter. The Use of Attack and Protection Trees to Analyze Security for an Online Banking System. *40th International Conference on System Sciences (HICSS), Waikoloa, Big Island, Hawaii, USA*, 2007.

[18] Andreas Ekelhart, Stefan Fenz, Markus Klemen, and Edgar R. Weippl. Security Ontology: Simulating Threats to Corporate Assets. *2nd International Conference Information Systems Security (ICISS), Kolkata, India*, pages 249–259, December 2006.

[19] Stefan Fenz and Andreas Ekelhart. Formalizing Information Security Knowledge. *ACM Symposium on Information, Computer and Communications Security (ASIACCS), Sydney, Australia*, March 2009.

[20] Stefan Fenz, Gernot Goluch, Andreas Ekelhart, Bernhard Riedl, and Edgar R. Weippl. Information Security Fortification by Ontological Mapping of the ISOIEC 27001 Standard. *13th Pacific Rim International Symposium on Dependable Computing (PRDC), Australia*, December 2007.

[21] Stefan Fenz, Thomas Pruckner, and Arman Manutscheri. Ontological Mapping of Information Security Best-Practice Guidelines. *12th International Conference on Business Information Systems (BIS), Poznan, Poland*, April 2009.

[22] William M. Fitzgerald. *An Ontology Engineering Approach to Network Access Control Configuration*. PhD thesis, University College Cork, Ireland, August 2010.

[23] William M. Fitzgerald and Simon N. Foley. Management of Heterogeneous Security Access Control Configuration using an Ontology Engineering Approach. *3rd ACM Workshop on Assurable & Usable Security Configuration, Chicago, USA*, October 2010.

[24] William M. Fitzgerald and Simon N. Foley. Aligning Semantic Web Applications with Network Access Controls. *Journal on Computer Standards & Interfaces, Volume 33, Issue 1, Special Issue: Secure Semantic Web, Pages 24-34, ISSN 0920-5489*, January 2011.

[25] William M. Fitzgerald, Simon N. Foley, and Mícheál Ó Foghlú. Network Access Control Configuration Management using Semantic Web Techniques. *Journal of Research and Practice in Information Technology, Volume 41 (2)*, May 2009.

[26] Simon N. Foley and William M. Fitzgerald. An Approach to Security Policy Configuration using Semantic Threat Graphs. *23rd Annual IFIP WG 11.3 Working Conference on Data and Applications Security (DBSec), Springer LNCS, Montreal, Canada*, July 2009.

[27] John H. Gennari, Mark A. Musen, Ray W. Fergerson, William E. Grosso, Monica Crubézy, Henrik Eriksson, Natalya F. Noy, and Samson W. Tu. The Evolution of Protégé: An Environment for Knowledge-Based Systems Development. In *Journal of Human-Computer Studies, Volume 58, Issue 1*, 2003.

[28] Lucian Gheorghe. *Designing and Implementing Linux Firewalls with QoS using netfilter, iproute2, NAT and l7-filter*. PACKT, October 2006.

[29] Michael Gruninger and Mark S. Fox. Methodology for the Design and Evaluation of Ontologies. *International Joint Conference on Artificial Inteligence (IJCAI), Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, Canada*, August 1995.

[30] Mark J. Handley and Eric Rescorla. RFC4732: Internet Denial-of-Service Considerations. *http://ietf.org*, November 2006.

[31] Shawn Hernan, Scott Lambert, Tomasz Ostwald, and Adam Shostack. Uncover Security Design Flaws Using The STRIDE Approach. *http://microsoft.com/*.

[32] Almut Herzog, Nahid Shahmehri, and Claudia Duma. An Ontology of Information Security. *International Journal of Information Security and Privacy (IJISP), Volume 1, Issue 4*, 2007.

[33] IANA. Port Numbers. *Internet Assigned Numbers Authority (IANA), http://www.iana.org/assignments/port-numbers*.

[34] Nwokedi C. Idika. *Characterizing and Aggregating Attack Graph-Based Security Metrics*. PhD thesis, Purdue University, West Lafayette, Indiana, USA, August 2010.

[35] IETF. RFC 3330: Special-Use IPv4 Addresses. *http://ietf.org*, September 2002.

[36] Kyle Ingols, Richard Lippmann, and Keith Piwowarski. Practical Attack Graph Generation for Network Defense. *22nd Annual Computer Security Applications Conference (ACSAC), FL, USA*, December 2006.

[37] ISO. ISO 27001 Standard. *http://www.27000.org*.

[38] Gordon Lyon. *NMAP Network Scanning: Official Nmap Project Guide to Network Discovery and Security Scanning*. Insecure LLC, CA, United States, December 2008.

[39] Sjouke Mauw and Martijn Oostdijk. Foundations of Attack Trees. *8th International Conference on Information Security and Cryptology (ICISC), LNCS 3935, pages 186-198, Seoul, Korea*, December 2005.

[40] Vaibhav Mehta, Constantinos Bartzis, Haifeng Zhu, Edmund Clarke, and Jeannette M. Wing. Ranking Attack Graphs. *9th International Symposium on Recent Advances in Intrusion Detection, Springer LNCS 4219, Hamburg, Germany*, September 2006.

[41] NISCC. Security Assessment of the Transmission Control Protocol (TCP). *CPNI TECHNICAL NOTE 3/2009, Centre for the Protection of National Infrastructure, (CPNI), http://www.cpni.gov.uk/*, February 2009.

[42] Steven Noel and Sushil Jajodia. Managing Attack Graph Complexity Through Visual Hierarchical Aggregation. *ACM workshop on Visualization and data mining for computer security, Washington DC, USA*, October 2004.

[43] Leo Obrst, Werner Ceusters, Inderjeet Mani, Steve Ray, and Barry Smith. *SEMANTIC WEB: Revolutionizing Knowledge Discovery in the Life Sciences*, chapter 7. Springer Science and Business Media, 2007.

[44] Martin O'Connor, Holger Knublauch, Samson Tu, Benjamin Grossof, Mike Dean, William Grosso, and Mark Musen. Supporting Rule System Interoperability on the Semantic Web with SWRL. *4th International Semantic Web Conference (ISWC2005), Galway, Ireland*, 2005.

[45] Alexander Opel. Design and Implementation of a Support Tool for Attack Trees. *Internship Thesis, Otto-von-Guericke University Magdeburg, Germany*, March 2005.

[46] Bijan Parsia and Evren Sirin. Pellet: An OWL DL Reasoner. *3rd International Semantic Web Conference (ISWC), Japan*, November 2004.

[47] Cynthia Phillips and Laura P. Swiler. A graph-based system for network-vulnerability analysis. *New Security Paradigms Workshop (NSPW), Virginia, USA*, September 1998.

[48] Indrajit Ray and Nayot Poolsappasit. Using Attack Trees to Identify Malicious Attacks from Authorized Insiders. *10th European Symposium on Research in Computer Security, Milan, Italy*, September 2005.

[49] Yakov Rekhter, Robert G Moskowitz, Daniel Karrenberg, Geert Jan de Groot, and Eliot Lear. RFC1918: Address Allocation for Private Internets. *http://ietf.org*, Febuary 1996.

[50] Peter Saint-Andre. RFC3920: Extensible Messaging and Presence Protocol (XMPP): Core. *http://ietf.org*, October 2004.

[51] Peter Saint-Andre. XEP-0205: Best Practice to Discourage Denial of Service Attacks. *http://xmpp.org*, January 2009.

[52] Peter Saint-Andre, Kevin Smith, and Remko Troncon. *XMPP: The Definitive Guide, Building Real-Time Applications with Jabber Technologies*. O'Reilly, 2009.

[53] Matthias Samwald. Classes versus Individuals: Fundamental Design Issues for Ontologies on the Biomedical Semantic Web. *Foundations of Clinical Terminologies and Classifications (FCTC), Romania*, April 2006.

[54] Karen Scarfone and Paul Hoffman. Guidelines on Firewalls and Firewall Policy: Recommendations of the National Institute of Standards and Technology. *NIST Special Publication 800-41, Revision 1*, September 2009.

[55] Karen Scarfone, Wayne Jansen, and Miles Tracy. Guide to General Server Security: Recommendations of the National Institute of Standards and Technology. *NIST Special Publication 800-123*, July 2008.

[56] Bruce Schneier. Secrets and Lies Digital Security in Networked World. *Wiley Publishing*, 2004.

[57] Oleg Sheyner, Joshua Haines, Somesh Jha, Richard Lippmann, and Jeannette M. Wing. Automated Generation and Analysis of Attack Graphs. *IEEE Symposium on Security and Privacy, Oakland, CA, USA*, May 2002.

[58] Oleg M. Sheyner. *Scenario Graphs and Attack Graphs*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2004.

[59] Robert Shirey. RFC 2828: Internet Security Glossary. *http://ietf.org*, May 2000.

[60] Wes Sonnenreich, Jason Albanese, and Bruce Stout. Return On Security Investment (ROSI): A Practical Quantitative Model. *Journal of Research and Practice in Information Technology*, 38(1):45–56, February 2006.

[61] Michael Stamatelatos, William Vesely, Joanne Dugan, Joseph Fragola, Joseph Minarick, and Jan Railsback. Fault Tree Handbook with Aerospace Applications. *NASA Office of Safety and Mission Assurance NASA Headquarters, Version 1.1, Washington, USA*, August 2002.

[62] Gary Stoneburner, Alice Goguen, and Alexis Feringa. Risk Management Guide for Information Technology Systems: Recommendations of the National Institute of Standards and Technology. *NIST-800-30*, July 2002.

[63] Miles Tracy, Wayne Jansen, Karen Scarfone, and Theodore Winograd. Guidelines on Electronic Mail Security: Recommendations of the National Institute of Standards and Technology. *NIST Special Publication 800-45, Version 2*, February 2007.

[64] Miles Tracy, Wayne Jansen, Karen Scarfone, and Theodore Winograd. Guidelines on Securing Public Web Servers: Recommendations of the National Institute of Standards and Technology. *NIST Special Publication 800-44, Version 2*, September 2009.

[65] Mike Uschold and Michael Gruninger. Ontologies: Principles, Methods, and Applications. *Knowledge Engineering Review, Vol. 11, No. 2., pp. 93-155*, 1996.

[66] Wietse Venema. TCP Wrapper: Network monitoring, access control, and booby traps. *3rd UNIX Security Symposium, Baltimore, USA*, September 1992.

[67] John Wack, Ken Cutler, and Jamie Pole. Guidelines on Firewalls and Firewall Policy: Recommendations of the National Institute of Standards and Technology. *NIST-800-41*, 2002.

[68] Lingyu Wang, Chao Yao, Anoop Singhal, and Sushil Jajodia. Implementing Interactive Analysis of Attack Graphs using Relational Databases. *Journal of Computer Security (JCS)*, December 2008.

[69] Duane Wessels. *Squid: The Definitive Guide*. O'Reilly Media, 2004.

[70] Jeannette M. Wing. Scenario Graphs Applied to Network Security. *Information Assurance: Survivability and Security in Networked Systems, Chapter 9, Morgan Kaufmann Publishers, Elsevier*, 2008.