Identifying the Missing Aspects of the ANSI/ISA Best Practices for Security Policy

Dinesha Ranathunga
University of Adelaide, Australia

Matthew Roughan
University of Adelaide, Australia

Phil Kernick
CQR Consulting, Australia

Nick Falkner
University of Adelaide, Australia

Hung Nguyen
University of Adelaide, Australia

ABSTRACT

Firewall configuration is a critical activity for the Supervisory Control and Data Acquisition (SCADA) networks that control power stations, water distribution, factory automation, etc. The American National Standards Institute (ANSI) provides specifications for the best practices in developing high-level security policy [9]. However, firewalls continue to be configured manually, a common but error prone process. Automation can make designing firewall configurations more reliable and their deployment increasingly cost-effective. ANSI best practices lack specification in several key aspects needed to allow a firewall to be automatically configured. In this paper we discuss the missing aspects of the existing best practice specifications and propose solutions. We then apply our corrected best practice specifications to real SCADA firewall configurations and evaluate their usefulness for high-level automated specification of firewalls.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*security and protection*

Keywords

SCADA network security; Zone-Conduit model; firewall auto-configuration; security policy

1. INTRODUCTION

Network security involves the protection of data and resources in a communications network, while providing access to authorised users [5]. It is a crucial element of any modern day business in maintaining productivity, minimising disruptions and achieving regulatory compliance. Firewalls are a standard mechanism for enforcing network security. They protect the Confidentiality (C), Integrity (I) and Availability (A) of data and resources inside a network.

SCADA networks control the distributed assets of many industrial systems. Power generation, water distribution and factory automation are just a few examples that illustrate the critical nature of these networks.

SCADA networks are not like corporate IT networks, they have been designed primarily for reliability and SCADA devices often lack built-in security features for protection from cyber attacks. Consequently, these devices depend on firewalls to protect them [13]. In these types of networks, the traditional security priority; CIA, is normally reversed to AIC, as availability has the highest priority. Disruptions and downtimes are generally unacceptable in SCADA networks, and rebooting of devices is often impractical. Firewalls help deliver high-availability of systems and are an integral part of the safe and reliable operation of SCADA networks.

Firewall configuration, in practice, is a complicated and repetitive manual task. It involves training in proprietary and device specific configuration languages and long and complex device configurations. Lack of automation tools to assist with such complexity has resulted in unoptimised, error-prone configurations that often deviate from the industry recommended network security guidelines [16, 17].

Incorrect configuration of SCADA firewalls can lead to security breaches that could result in significant environmental damage, financial loss or in the worst case, loss of human lives. Examples of past incidents include the hacking of Maroochy Shire Council's sewage system in 2000, which saw tonnes of raw sewage released into public park-lands and river systems [10] and the sophisticated Stuxnet worm which attacked and damaged Iran's nuclear facilities in 2010 [13].

The automation of firewall configuration has been investigated by Bellovin and Bush [2]. They identified requirements such as security, robustness and a database-driven approach as key, but left out high-level policies.

The starting point for automation is a high-level security policy description. Such a description would allow high-level policies to be specified by management level policy makers, who are often unable to specify policies in sufficient technical detail. These policy makers often have a good understanding of the industrial engineering requirements of a SCADA plant, and can use it to instrument an organisation's network security policies using simple high-level goals.

CPSS'15, April 14, 2015, Singapore.

http://dx.doi.org/10.1145/2732198.2732201.

The ANSI/International Society for Automation (ISA) best practices introduce useful security concepts to mitigate threats in control systems [9]. These concepts enable high-level policy specification for SCADA networks. Using them, a high-level description suitable for firewall auto-configuration can be obtained. We analyse real SCADA firewall configurations using these concepts to identify missing aspects required for auto-configuration and propose solutions. The study allows us to evaluate how well suited these security concepts are, as they currently exist, to cater for policy specification requirements found in practice. Key gaps that prevent clear specification of high-level policy, for example, in specifying firewall management requirements, are identified from our evaluation. The solutions we propose for these gaps increase the precision and usefulness of the best practice for the automated specification of firewalls.

Our contribution is to fill the gaps in the ANSI/ISA standard. We show the missing pieces needed to change the best practice from being a collection of good ideas into a tight specification. Additionally, through real-world case studies we illustrate that SCADA firewalls, even in simple cases are not configured appropriately.

2. BACKGROUND

SCADA networks are vital to the operation of a nation's critical infrastructure plants. Recently, there has been a significant increase in the number of plant disruptions and shutdowns due to cyber-security issues in these networks [3].

Poor internal network segmentation in SCADA systems is a significant contributor to the quick spread of security threats and attacks between subnets [3,9,13]. The ANSI/ISA standards introduce the concepts of *zones* and *conduits* as a way of segmenting and isolating the various sub-systems in a control system [9]. The *Zone-Conduit model* is a very useful starting point for a high-level description of security policy, and so we shall describe it in detail here.

A *zone* is a logical or physical grouping of an organisation's systems with similar security requirements based on criticality and consequence [9]. By grouping systems in this manner, a single security policy can be defined for all members of a zone. For example, 3 security zones can be defined to accommodate low, medium and high-risk systems, with each device assigned to its respective zone based on their security level needed. A low-risk system can be accommodated within a medium or high security zone without compromising security, but not vice versa.

The uniform security policy of a zone guides the construction and maintenance of all systems within the zone [9]. Hence, selected systems within a zone (*e.g.,* a server) should not have their own separate policies. Allowing separate policies would impart an incorrect sense of security to those systems. These systems are only as secure as the zone itself, in the absence of any firewalls enforcing a real separation.

A *conduit* provides the communication path between two zones as well as the necessary security functions for them to communicate securely [9]. Since availability is paramount in a SCADA network, a conduit should resist Denial of Service (DoS) attacks and preserve integrity and confidentiality of network traffic. This is achieved using security mitigation mechanisms (*e.g.,* firewalls) implemented within the conduit. In Figure 1, two typical zones in SCADA environments; the *SCADA-Zone* and the *Corporate-Zone* are shown connected by a conduit.

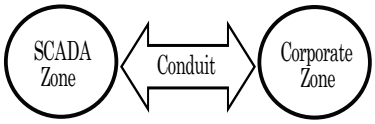


Figure 1: Example Zone-Conduit model, adapted from [3].

A conduit cannot be a communications link that simply inter-connects zones without restricting traffic-flow [9]. From a security perspective, this does not provide any mitigation capabilities to the connecting zones. Such a conduit fails to enforce a clear separation of zones. It is equivalent to the two zones being a single zone and would prove useless for security policy description purposes. A conduit, in this view, always offers some security mitigation capability, typically using single or multiple firewall(s).

A Zone-Conduit security model of a network allows accurate assessment of common threats, vulnerabilities, and required security mitigations to protect SCADA resources [9]. It provides a high-level view of an organisation's security and traffic control strategy. The model helps identify the disjoint security zones in the network, enabling the detection of serious design flaws such as the allocation of low and high security devices into a single zone. Such direct violations of the ANSI/ISA best practices would be a clear indication of exploitable vulnerabilities.

The Zone-Conduit model also reveals unwanted inter-zone communication paths. It enables us to understand whether the security mitigation devices installed on each path are capable of offering the level of mitigation required for secure inter-zone communications.

Zone and conduit concepts are intended as a platform for high-level security policy description. Before using these concepts in policy specification for firewalls, it is best to evaluate their usefulness. Particularly how well they cater for security architectures used in practice in real networks.

2.1 Related Work

Real firewall configurations have been studied; a statistical analysis of corporate-firewall configurations [16] reported serious errors found in over 80% of the configurations analysed. Corporate-firewall configurations have also been analysed using a firewall modeling and analysis toolkit (FIREMAN) [21] and several serious errors were also unveiled.

The problem of firewall configuration is therefore well-known and research has started to address it. Lumeta [15] and Fang [11] are management and analysis tools that interact with users on queries about firewall rules. They take a minimum network topology description and firewall configurations as input. These inputs are used to build an internal representation of firewall rules which users can query to assist debugging and troubleshooting.

Margrave [12] is a similar firewall configuration analysis tool that allows users to check policies against security goals. MIRAGE [8] removes inconsistencies in security policies of distributed firewalls and Intrusion Detection Systems (IDS) systems. None of the above tools address the manual and device centric configuration approach that cause misconfigurations in the first place.

Studies have been carried out on auto-generation and relaying of organisational policies to distributed firewalls [2, 14]. But they lack support for high-level requirements.

The unified modeling of packet filters and routing protocols to characterise reachability of a network has been presented by Xie et al. [18]. However, it does not take into account implicit rules which enable services through a firewall over and above Access Control Lists (ACLs).

Tesseract [19] implements a network control plane enabling direct control of Ethernet and IP based services. Direct control promotes centralised policy implementation, but the approach lacks the ability to abstract minute policy configuration details and specify high-level requirements.

None of the related work refer to the ANSI/ISA Zone-Conduit model or attempt to identify a high-level policy description useful for auto-configuring firewalls. They also do not look at SCADA networks, with unique security requirements compared to Corporate networks. SCADA networks include highly constrained devices such as Programmable Logic Controllers (PLCs) that are incapable of supporting built-in network security features [13]. They have poorly designed TCP/IP stacks that easily fail. Installation of security patches and device upgrades are also infrequent as rebooting is often impractical in these networks.

A primary objective of our research is to identify missing aspects of the ANSI/ISA Zone-Conduit model as a high-level security policy description for SCADA firewalls. We pay close attention to the industry best practices surrounding SCADA network security to understand how well practical firewall deployments adhere to these guidelines.

3. METHODOLOGY

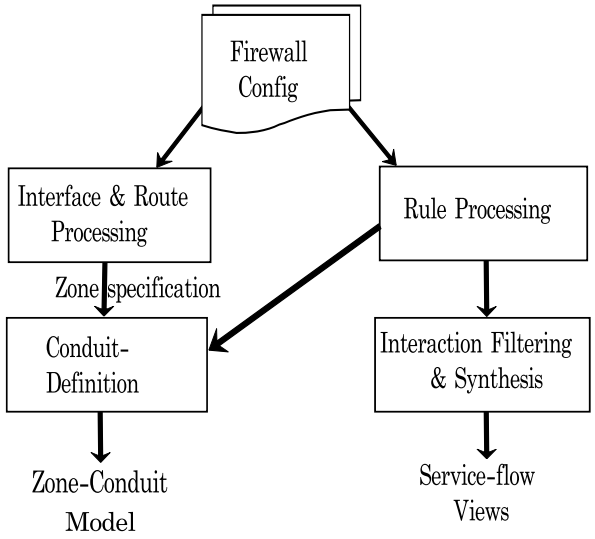


Figure 2: Firewall configuration parsing process.

Firewall configurations are long and complex. For example, the configurations we discuss have 1360 lines on average per firewall. We could not use existing tools such as Fang or Lumeta to analyse these as they do not support Zone-Conduit based high-level policies. So we built an automated parser to parse configurations using Zone-Conduit concepts.

We describe the Parser in detail here because it explains the use of Zone-Conduit concepts in the analysis of practical networks and allows to identify shortfalls in the model and help find solutions. The Parser is depicted in Figure 2. The details are described below:

Firewall Config: The input firewall configuration text-file containing interface configurations, static routes and ACLs. Multiple files can be input for simultaneous processing.
Interface and Route Processing: The processing of firewall interface configurations and static routes. This extracts interface names, subnet IP addresses, security levels, additional network and gateway IP addresses. Details of this processing are covered in Subsection 3.1.
Rule Processing: The processing of ACLs assigned to firewall interfaces and any implicit rules. Implicit rules enable services through the firewall over and above ACLs. More details are discussed in Subsection 3.2.
Conduit definition: The definition of conduits that interconnect the security zones in the SCADA network. Details are covered in Subsection 3.3.
Zone-Conduit Model: The zone and conduit topology output of the SCADA network.
Interaction Filtering & Synthesis: The filtering of ACL rule interactions and synthesis with implicit rules. Details of this stage are covered in Subsection 3.6.
Service-flow views: The output traffic-flow views for the firewall. A service-flow view describes an enabled protocol through the firewall by zone.

Our current Parser uses one or more Cisco Adaptive Security Appliance (ASA) or Private Internet eXchange (PIX) or Internetwork Operating System (IOS) firewall configurations as input. It begins by processing the individual firewall interface configurations. It also processes any static route configurations to identify the location of additional networks and gateways. Rule processing partly involves parsing the ACLs assigned to firewall interfaces. These indicate the traffic permitted to traverse each of the firewall interfaces. Additionally, rule processing also involves parsing implicitly enabled services. The Parser then performs *conduit definition*. This outputs the ANSI/ISA standard Zone-Conduit security model. ACLs and implicit rules are also analysed by the Parser to filter-out any interactions present and then synthesised to generate service-flow views as output.

3.1 Zone Construction

The Parser analyses the interfaces and subnets defined in the firewall configuration to construct zones. Initially it assumes each firewall interface connects to a disjoint zone, and looks for indications that these potential zones should merge. A zone merge can be identified via traffic leakages between the zones. These leakages occur outside of a firewall but can often be identified through the inspection of ACL contents.

Where such a leakage exists, ACLs should control traffic flow equally for those services, on both firewall interfaces connected to the respective zones. By inspecting the ACL contents of a firewall, and applying the policy that inter-zone traffic-flow is allowed via firewall-only paths when redundant paths are available (*i.e.,* not relayed through a 3rd zone), we can deduce that the assumed disjoint zones must form a single zone. The original disjoint *Zone-Firewall model* is then updated with identified merged zones.

Static routes contain IP address details of next-hop gateways and networks reachable via them. By identifying and including these additional networks and gateways, we can further extend our Zone-Firewall model.

3.2 Implicit Rules

In a Cisco firewall, traffic flows can be enabled explicitly through ACLs or implicitly via several alternate methods. One available method in ASA and PIX firewalls is to assign *security levels* to the firewall interfaces [6]. An interface security level is defined as a level of trust bestowed on the network connected to that firewall interface. In the absence of an ACL assigned to such an interface, certain traffic flows are permitted by default from an interface with a high security level to one with a lower security level [6].

Special configuration commands can also be used to enable services implicitly, for example to enable SSH or HTTP firewall management traffic into the firewall interfaces [6]. Accommodating such management traffic using zones is discussed later in Subsection 3.4.

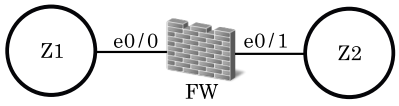
Implicit rules provide quick and easy alternatives to ACLs in enabling services through the firewall. They may not map to clear policies but are convenient. However, auto-configuration relies on clear security policies to permit traffic through a firewall. Implicit rules may aim to provide this, but we will see that they actually confuse the situation.

3.3 Zone-Conduit Model

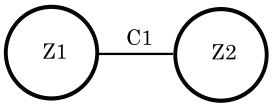
As Section 2 discussed, a Zone-Conduit model describes the logical grouping of systems in a network. It gives a high-level view of an organisation's network segregation strategy.

The Parser uses the Zone-Firewall model to build a corresponding Zone-Conduit model. To do so, we identify the security conduits based on ANSI/ISA guidelines. A conduit is defined between zones, based on the available *firewall-only paths* between them. An example conduit (C1) between 2 zones with a single-firewall path is shown in Figure 3.

This case is almost trivial, but the question of how to map a network to zones and conduits has more complex cases.



(a) Zone-Firewall model for 2 zones separated by a firewall.



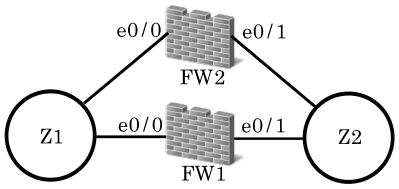
(b) Zone-Conduit model.

Figure 3: Single-firewall Conduit definition.

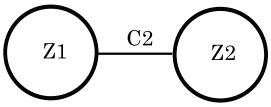
When two zones are connected by parallel links, the ANSI/ISA standard allows them to be modelled as multiple conduits. Doing so however, would imply that multiple security policies could exist between these zones when only one is possible from the strict interpretation of a zone. Hence, we define a single conduit to implement the single policy relationship. An example conduit (C2) is depicted in Figure 4b.

Firewall paths can include firewalls in series (Figure 5a). This back-to-back firewall architecture is one of the industry recommended security architectures [4] where *defence in depth* is achieved by using different vendors' devices.

This firewall setup can also provide DoS protection; one firewall performs simple processing of a large volume of packets and the other conducts complex processing (*e.g.,* Deep Packet Inspection (DPI)) on a smaller number of packets [4]. Network engineers may also setup logging and alerts to originate from the firewalls differently, in such a circumstance. ANSI/ISA guidelines lack clear specification on how to define zones and conduits to precisely capture the traffic-flow requirements in this context.

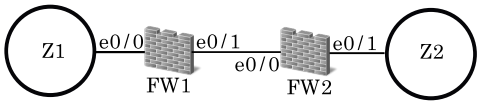


(a) Zone-Firewall model for 2 zones separated by parallel firewalls.



(b) Zone-Conduit model.

Figure 4: Parallel-firewall Conduit definition.



(a) Zone-Firewall model for 2 zones separated by serial firewalls.

(b) Zone-Conduit model.

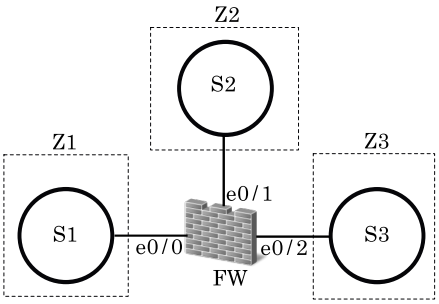
Figure 5: Serial-firewall Conduit definitions.

A single conduit containing both firewalls exists, if we dismiss the inter-firewall link. But, for automation, a single conduit hinders precise specification of the distinct firewalls.

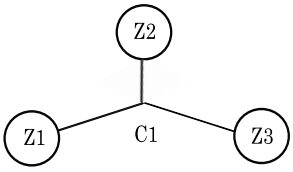
We propose to treat this connecting link as a separate zone, to overcome the specification shortfall. It is referred to as an *Abstract-Zone* in the absence of any real network devices within it (Figure 5b). The approach creates two separate conduits (C1 and C2), each containing one firewall. Auto-configuration can now leverage the distinct conduits to specify the individual policy requirements.

We can also model the security properties of the subnet between the serial firewalls as a Demilitarised Zone (DMZ), in case devices such as a logger are added. A DMZ is used to expose an organisation's external-facing services (*e.g.,* a mail server) to untrusted networks [4]. It adds a layer of security to the company's trusted internal networks by only providing direct external access to the hosts in the DMZ.

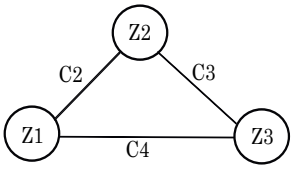
A conduit may also inter-connect more than two security zones [9]. ANSI/ISA guidelines lacks clear specification on appropriate conduit definitions in such circumstance. Consequently, the example Zone-Firewall model depicted in Figure 6a, could be modelled using a hyper-graph (Figure 6b). In this model, the firewall (FW) is located inside the hyper-edge conduit C1 which has one-to-many zone-communication paths. This complex conduit can implement



(a) Zone-Firewall model for 3 zones separated by a firewall.



(b) Hyper-graph Zone-Conduit model.



(c) Simple-graph Zone-Conduit model.

Figure 6: Conduit-definition alternatives.

multiple security policies; between Z1 and Z2, Z2 and Z3 and Z3 and Z1. Catering for this complexity requires the conduit to track the participating zones per policy. There is also no clear mapping of the ACL rules enforcing each policy to the firewall interfaces. Hence, the hyper-graph conduit model is difficult to use for firewall auto-configuration purposes.

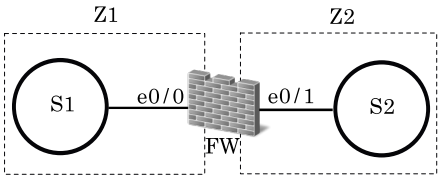
To simplify the complexities of hyper-edge conduits, we propose to generate a Zone-Conduit model that consists of only one-to-one zone-communication paths (Figure 6c). Each conduit now implements a single security policy between two zones. The simple design also requires each conduit to only contain the firewall interfaces attached to its connecting zones (*e.g.,* C2 contains e0/0 and e0/1). A conduit path now reveals the exact firewall interfaces and their layout with respective to the connecting zones, enabling easy placement of required ACL rules. Consequently, the choice of simple-edge conduits, allows us to enforce a strict 1:1 mapping between conduits and policies. This restriction yields a precise high-level specification, useful for firewall auto-configuration.

This logical method of conduit-definition can lead to multiple conduits sharing the same firewall in their mitigation offering (*e.g.,* C2, C3, C4 share FW in Figure 6c).

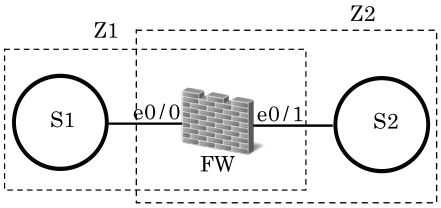
In summary, a single conduit need not always map to a single firewall. In-fact one-to-many and many-to-one mappings between conduits and firewalls are more useful for high-level security specification. However, our recommendation for firewall auto-configuration is that one conduit should always implement a single relationship between only 2 zones.

3.4 Firewall Management Access Control

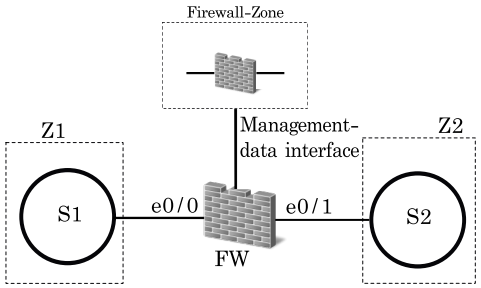
In addition to offering mitigation capabilities to zones, firewalls play a dual role by providing secure, authorised network management access to themselves. ANSI/ISA standard defines the use of a firewall within a conduit as a mitigation device, but does not clearly state how zone and conduit concepts should be used to capture firewall management traffic requirements. This is a critical shortfall, because if



(a) Firewall interfaces in zones.



(b) Firewall shared by zones.



(c) Dedicated Firewall-Zone.

Figure 7: Firewall-Zone alternatives for 2 subnets S1 & S2 separated by a firewall.

management of the firewall is compromised, the entire system is compromised.

There are several possible ways to address the issue, as illustrated in Figure 7.

3.4.1 *Firewall Partially Included in Zones*

With this approach, each firewall *interface* belongs to the zone directly connected to that interface (Figure 7a). It implies that all IP traffic to the firewall from hosts and subnets of zones Z1 and Z2 is allowed.

While simple, this approach has obvious problems. The design prevents restriction of firewall access by its connected zones. For example, disallowing HTTP access to the firewall by zone Z1 would be impossible with this type of a model.

3.4.2 *Firewall Shared Between Zones*

This model assigns a firewall interface to all connected zones (Figure 7b), also implying removal of any traffic restriction between hosts and subnets within each zone and the firewall by default. The outcome is similar to that of 3.4.1, preventing placement of a required policy between a zone and the firewall.

3.4.3 *Firewall in its Own Zone*

Here, we exclude the firewall from belonging to any existing zone and place it separately in a new security zone on its own. This may seem more complex but actually represents the real situation well. This new Firewall-Zone (FWZ) is connected to the firewall (Figure 7c) via the Management-Data Interface (MDI). The MDI is a logical interface that

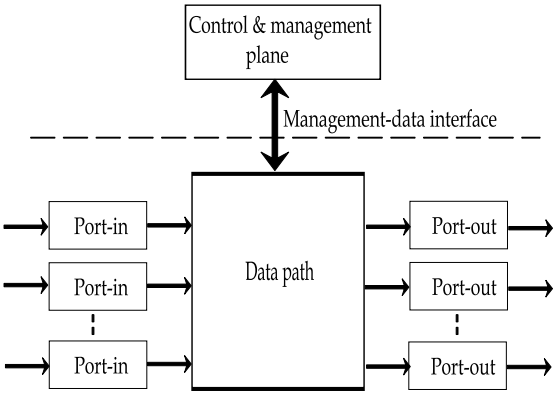


Figure 8: Logical firewall architecture adapted from [7], depicting the Firewall-Zone Management-data interface.

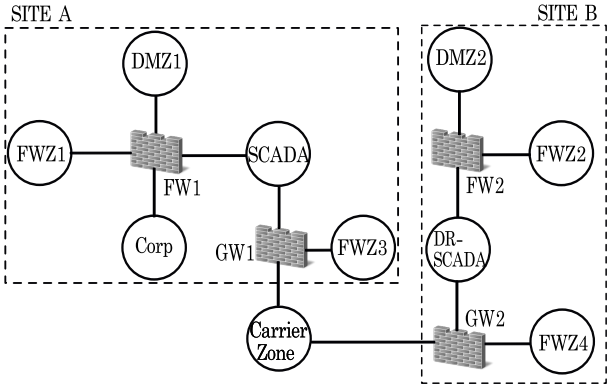


Figure 9: Carrier-Zone interconnecting geographically dispersed sites.

provides traffic packets to the firewall's control and management plane (Figure 8) from the data path [7]. The control and management plane is responsible for processing the firewall bound management traffic, while the data path handles the traffic forwarded through the firewall.

The Firewall-Zone allows restrictions to be placed on the firewall to regulate its management traffic. The model depicts the firewall's dual role precisely, and allows imposing restrictions such as disallowing HTTP access to the firewall by zone Z1 (*e.g.,* by placing ACL rules on interface e0/0).

Our introduction of a dedicated Firewall-Zone significantly simplifies management policy specification and hence, auto-configuration of a firewall. It allows firewall-management and non-management traffic to be considered equally, but to be specified separately in the auto-configuration process. This clean approach can further restrict the type of management traffic allowed (*e.g.,* block Telnet), to adhere to industry best practices. Of course additional security mechanisms (*e.g.,* password access) are required, but these are outside the current scope of this analysis.

The updated Zone-Firewall model, encompasses all disjoint zones and their interconnections to the firewall. It includes implicit zones such as the Firewall-Zone required to facilitate firewall management and explicit zones such as the Corporate-Zone. By compiling a model that embodies a rich collection of such zones, their contents (*i.e.,* network devices) and respective interconnections to the firewall(s), we obtain a high-level view of the security strategy employed in the SCADA network.

3.5 Carrier Network Abstraction

Real networks often utilise a Carrier Network (CN) provided by a telecommunication service provider to interconnect geographically dispersed sites. This is prevalent in SCADA networks which control distributed field-site equipment from a centralised control centre over, for example, a leased line Wide Area Network (WAN).

The traffic relayed via the CN is controlled by the security policies between the zones within the two interconnecting sites (Figure 9). Due to the unavailability of every gateway and network-device configuration for analysis, we model the interconnectivity provided by a CN by abstracting its underlying implementation details.

A simple yet effective strategy is to use a Carrier-Zone in the Zone-Firewall model as shown in Figure 9, that encompasses the CN. This zone provides connectivity, facilitates security policy specification between the sites and abstracts unwanted implementation details.

3.6 Service-flow Views

A service-flow view is a directed-graph of hosts/subnets (or their respective zones) that can initiate and/or accept that service protocol. The Parser generates these for the protocols; IP, TCP, UDP and ICMP, broken down by port, host/subnet and zone as applicable. The output views are graphical representations based on GraphML, readily viewable using tools that support the format such as yEd [20].

Processing the firewall ACLs helps us to understand the types of services enabled explicitly between the zones. A few obstacles need to be overcome to gain this understanding.

Primarily, an ACL rule-set can contain potentially interacting rules referred to as intra-ACL interactions. These interactions are caused by rule-overlaps, triggered by distinct rules having common packet matching criteria described in Subsection 3.1. An example of such a scenario is provided in Listing 1, where `rule1` and `rule2` both apply to HTTP packets originating at host 10.0.1.18 destined to host `web_svr`.

In Cisco firewalls, the outcome of such a pair of rules depends on several factors. These include the order in which they are listed, the level of overlap (*i.e.,* partial, full overlap or subset) and their rule actions. Traffic packets to which both rules equally apply, will be filtered via `rule1` in the list (*i.e.,* by line 2 in Listing 1). `rule2` is completely overshadowed. Traffic packets outside the rule-overlapping region (*e.g.,* host 10.0.1.20), that still apply to `rule1` or `rule2` will continue to be filtered by their intended rule.

Based on the extent of overlap, interacting rules can be classified as *generalisations*, *shadowed-rules*, *partial-overlaps* and *conflicts*. A *generalisation* occurs when a subset of the packets matched to a rule has been excluded by one or more preceding rules with an identical action. A *shadowed-rule* is the opposite, all packets applicable to such a rule have already been matched by a preceding rule with an identical action. A *partial-overlap* occurs when the set of packets matched to a rule partially-intersect with another preceding rule with a similar action. In a *conflict*, the current rule intersects with preceding rules but specifies a different action.

We derive the net-effect of the intra-ACL interactions and generate an interaction free equivalent version (ACL_V1) of the ACL. This allows to accurately view the services enabled

```
1  access-list acl_overlap_in remark rule1
2  access-list acl_overlap_in permit tcp 10.0.1.0 255.255.255.0  host web_svr eq 80
3  access-list acl_overlap_in remark rule2
4  access-list acl_overlap_in permit tcp host 10.0.1.18  host web_svr eq 80
```

Listing 1: Example intra-ACL rule interaction (comments are denoted by *remark*).

```
1  access-list acl-in remark rule1
2  access-list acl-in permit tcp host 10.0.1.25  host web_svr eq 80
3
4  access-list acl-out remark rule1
5  access-list acl-out deny tcp host 10.0.1.25  host web_svr eq 80
```

Listing 2: Example inter-ACL rule interaction (comments are denoted by *remark*).

by each ACL. As a by-product of this processing, the Parser creates a list of all intra-ACL interactions found. These inconsistencies can assist with security audits.

Secondarily, there can also exist inter-ACL interactions that alter a rule's intended behaviour. Figure 10 and Listing 2 present an example, where `rule1` in `acl-in` permits HTTP traffic from host 10.0.1.25 to host `web_svr`. The same traffic is denied by `rule1` in `acl-out`. Since `acl-out` inevitably applies to any traffic packet traversing from `zone1` to `zone2`, the net-effect of `rule1` is the equivalent of a *null rule*. Hence, the Parser also needs to analyse potential inter-ACL interactions on ACL_V1, to derive a second version (ACL_V2) that is interaction free. ACL_V2 now reflects the net-effect of all rule interactions possible for a given network.

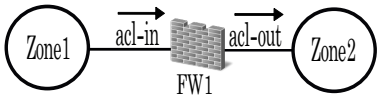


Figure 10: Zone-Firewall model for Listing 2.

The Parser additionally processes implicit rules based on interface security levels and builds an IP service-flow view depicting enabled generic traffic-flows. It also processes special Cisco configuration commands that permit firewall management traffic above ACLs. Corresponding implicit service-flow views are also created by the Parser.

Finally, the Parser synthesises the explicit and implicit service-flow views to derive a collection of *net service-flow views* per protocol. These views accurately describe the overall services enabled through the firewall(s).

4. CASE STUDIES

Obtaining real firewall configurations from working SCADA networks is very hard due to the sensitive nature of the data, and the naturally conservative nature of the owners. We were able to obtain such configurations from 4 SCADA networks and a high-level summary of the Systems Under Consideration (SUCs) is provided in Table 1. We will describe one of these case studies in detail here and refer to the other's to illustrate findings.

Due to security concerns and non-disclosure agreements, a modified version of each real SCADA network analysed is presented for discussion. Effort has been taken to ensure that the core security strategies and underlying issues uncovered remain intact. However, details such as IP addresses are anonymised.

4.1 Analysed Configuration Data

SUC 1 used two Cisco IOS routers. Their configurations were extracted in September 2011. Both routers had ACLs configured. One router (R1) consisted of 1149 Lines of Code (LoC) with 5 ACLs averaging 184 conventional rules each. The other router (R2) consisted of 1571 LOC with 5 ACLs averaging 290 rules each. Once ACLs are enabled, routers behave as firewalls.

The SUC is shown in Figure 11 which depicts the firewalls/routers employed (IOS ver 12.2 and 12.3) in a serial configuration. There were no network devices connected to the subnet between the firewalls. R1 connected to the Corporate network while R2 connected to the SCADA network.

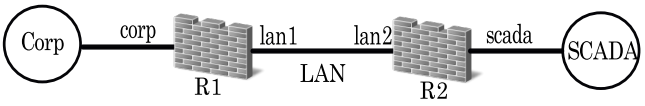


Figure 11: Dual firewall SCADA network studied.

R1 has 2 physical interfaces operational, pointing to `Corp` and `LAN` subnets. R2 also has 2 interfaces active, pointing to `SCADA` and `LAN`. The subnets are summarised below:

The Corporate network (Corp): Provides access to business applications and the Internet.
Local Area network (LAN): Responsible for enabling connectivity between R1 and R2.
The SCADA network (SCADA): Responsible for providing networked access to plant equipment.

`Corp` could accommodate up to 2046 hosts and `SCADA` accommodated up to 65534 hosts. There are 51 individually access controlled servers and hosts through the firewalls, across `Corp` and `SCADA`. `Corp` contains 12 management workstations, 2 monitoring stations and a DNS server. `SCADA` includes 8 `SCADA` servers, 2 DNS servers and a FTP server.

Listing 3 shows a snippet of the representative configuration data found on firewall R1. It reveals that both Cisco standard and extended ACLs were used with conventional rules. All 5 ACLs defined in R1 were in use while only 3 of the 5 ACLs defined in R2 were in use.

`Corp` has an extended ACL; `corp_access_in` assigned inbound on the `corp` firewall interface. Partially shown in Listing 3, the ACL aims to enable file transfer and sharing between corporate users and the servers located in `SCADA`

Table 1: High-level summary of analysed SUCs (* backup firewall, ** conventional format, LoC - Lines of Code).

SUC	Configuration date	Firewall type	Firewalls	Gateways	Zones	Average LoC	ACLs	Average rules per ACL**
1	Sep 2011	Cisco IOS	2	1	2	1360	8	237
2	Aug 2011	Cisco ASA	1(2)*	5	11	432	12	16
3	Oct 2011	Cisco PIX	2	2	5	125	8	6
4	Mar 2011	Cisco ASA	1	2	4	819	3	80

```
access-list acl_time_sync permit 172.27.0.1

access-list acl_snmp permit 172.27.1.3

access-list acl_vty1 permit 172.27.0.15 log
access-list acl_vty1 permit 172.27.0.7 log
access-list acl_vty1 deny   any log

access-list corp_access_in remark enable access to SCADA servers
access-list corp_access_in permit tcp host 172.27.1.6 host 172.19.0.1 eq 445
access-list corp_access_in permit tcp host 172.27.1.6 host 172.19.0.1 eq 3389
access-list corp_access_in permit tcp host 172.27.1.104 host 172.19.0.4 eq ftp-data
access-list corp_access_in permit tcp host 172.27.1.104 host 172.19.0.4 eq ftp
access-list corp_access_in permit tcp host 172.27.1.220 host 172.19.0.1 eq domain
access-list corp_access_in permit tcp host 172.27.1.98 eq smtp host 172.19.0.7
```

Listing 3: Configuration snippet of Firewall R1 (comments are denoted by *remark*).

using FTP and SMB. It also enables remote management of `SCADA` servers from the `Corp` management workstations using RDP. It allows DNS queries between the `Corp` hosted DNS server and the `SCADA` servers. The ACL also enables SMTP responses through to the Email clients in `SCADA`.

The LAN has two extended ACLs; `lan1_access_in` and `lan2_ access_in` assigned inbound on `lan1` and `lan2` interfaces respectively. `lan1_access_in` aims to enable all ICMP messaging between `SCADA` and `Corp`. It enables IP traffic between servers in `SCADA` and their clients in `Corp`. The ACL also allows IP traffic between selected `SCADA` hosts and the `Corp` hosted management and monitoring stations. It also enables `SCADA` mail clients to access Email servers located off the Corporate gateway and permits DNS requests from `SCADA` hosts to the `Corp` hosted DNS server.

`lan2_access_in` is similar to `corp_access_in` and aims to allow `Corp` user access to `SCADA` servers based on FTP and SMB. It too permits remote monitoring and management of `SCADA` servers from selected `Corp` workstations using RDP. The ACL also allows clients in `SCADA` access to the Email servers located off the Corporate gateway. It permits DNS requests from `SCADA` hosts to the `Corp` hosted DNS server.

R1 also has 3 standard ACLs defined; `acl_time_sync`, `acl_snmp`, `acl_vty1`. The first ACL aims to restrict NTP based time synchronisations to the known NTP servers located off the Corporate gateway. `acl_snmp` aims to restrict SNMP based alert exchanges with known SNMP servers located in the Corporate network. `acl_vty1` aims to restrict Telnet and SSH based remote access of R1 to selected hosts located off the Corporate gateway.

`SCADA` has an extended ACL; `scada_access_in` assigned inbound on the `scada` interface. Similar to `lan1_ access_in`, this ACL aims to enable access between the `SCADA` hosted servers and their `Corp` hosted clients. The ACL also enables IP traffic between selected `SCADA` hosts and the `Corp` hosted monitoring and management stations. It also permits `SCADA` mail clients access to the Email servers located off the Corporate gateway. DNS traffic is also permitted by this ACL between `SCADA` hosts and `Corp` hosted DNS server.

R2 also has a standard ACL in use; `acl_vty2` aims to restrict Telnet and SSH based remote access of R2, to selected hosts in `SCADA` and `Corp.`

The dual firewall solution studied did not employ Network Address Translation (NAT) or a Virtual Private Network (VPN). NAT was not used because none of the networked machines in the SCADA subnet communicated through the firewalls directly to the Internet. The setup of a VPN tunnel between the firewalls and a remote host was also not required as the firewalls did not provide direct remote access over the Internet, and both were physically co-located.

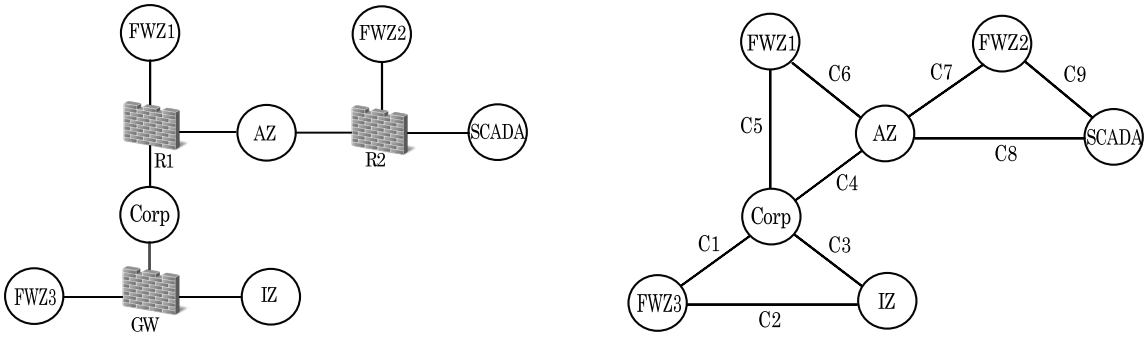
EIGRP routing was enabled on both firewalls using an identical Autonomous System Number (ASN). A default route was also present in each of the configurations, and revealed a new gateway located off `Corp`. It also informed us that R1 acted as the gateway for R2.

The configuration also revealed that a remote Syslog server located off the Corporate gateway was used to store local firewall log messages. Both firewalls used the *ip inspect name* command to automatically build state-tables and allow return traffic to bypass the ACLs as necessary.

SNMP messages were also enabled to be sent to a Network Monitoring Server (NMS) on authentication, link-up and link-down events.

Both firewalls were also configured as DHCP relay agents for local DHCP broadcasts.

The configurations also enabled R1 to be managed from R2, selected hosts in `Corp` as well as off Corporate gateway. R2 was allowed to be managed from R1, any `Corp` host and any `SCADA` host. Additionally R1 restricted remote logging via the vty lines to selected hosts in `Corp` using SSH only. Once remotely logged in, it also disallowed remoting out to other devices. R2 enabled remote logging via vty lines using Telnet or SSH.



(a) Zone-Firewall model including gateways.

(b) Zone-Conduit model of (a).

Figure 12: Security models of the network.

```
1  access-list corp_access_in permit tcp host 172.27.0.2 host 172.19.0.5 eq 135
2  access-list corp_access_in permit ip host 172.27.0.2 host 172.19.0.5
```

Listing 4: Example intra-ACL rule interaction identified by the Parser.

4.2 Results:

The Zone-Firewall model generated by parsing the configuration data for the System Under Consideration (SUC) is shown in Figure 12a. We can directly evaluate this model against the industry recommendations in [4, 11] on suitable secure segregation architectures for SCADA networks. The model falls into the category of paired-firewalls incorporating an empty demilitarised zone (*i.e.,* Abstract-Zone). It also complies with the requirement that the Internet-Zone should not be directly connected to the SCADA-Zone (it is reachable only via the Corporate-Zone's gateway). Figure 12b shows the corresponding Zone-Conduit model. It includes the Firewall-Zones (FWZ1, FWZ2) and an Abstract-Zone (AZ). The latter is used to correctly capture the individual policy requirements of the serial firewalls.

The Parser identified 167 intra-ACL interactions inside the 4 extended ACLs applied on the firewall interfaces, from ACL rule processing. These consisted of 7 generalisations, 146 shadowed-rules, 12 partial-overlaps and 2 conflicts. The generalisations and shadows were mostly caused by the use of 'any' as source and/or destination IP address and by the use of generic IP based rules. Listing 4 shows an example.

The Parser also identified 765 inter-ACL interactions involving the 4 ACLs, as summarised in Table 2. These overlaps contained 762 shadowed-rules and 3 generalisations. An example shadowed-rule is given in Listing 5 for the interaction between `corp_access_in` and `lan2_access_in`. The shadows consisted of many identical rules and a few different rules, indicative of the overlapping yet distinct nature of the policies of serial firewalls.

An example explicit service-flow view of SSH traffic is shown in Figure 13. We can check these service permissions against the industry recommendations in [4] for violations. In the case of SSH, which is usually considered a secure protocol, traffic can be allowed both inbound and/or outbound from the SCADA-Zone [4]. Figure 13 complies with this requirement. The service-flow view also shows the explicit intent to allow AZ to manage FWZ2 but not FWZ1. This reasserts the distinct policy requirements of the serial firewalls, showcasing the need for an Abstract-Zone to separate these firewalls to accommodate such policy specification.

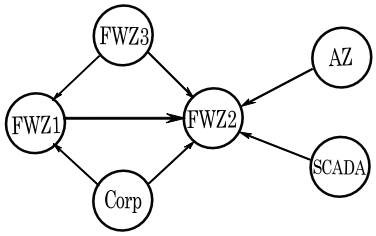


Figure 13: Explicit Service-flow View of SSH traffic.

The industry recommendations in [4, 13], call for traffic restrictions between the Corporate-Zone and the SCADA-Zone. This is readily met by the SUC through the presence of conduits C4 and C8 separating both zones in Figure 12b. It also unveils those zones that are not directly connected, as in this case, the Internet-Zone and the SCADA-Zone. This too is compliant with the industry recommendations in [4, 13].

An example implicit service-flow view generated in the case study is given in Figure 14a. It describes the generic IP traffic enabled between the Firewall-Zones and their adjacent zones. Since no outbound ACLs are assigned on the firewall interfaces, all IP based traffic flow is unrestricted between the Firewall-Zone and these adjacent zones.

Figure 14b describes the service-flow view for implicitly enabled Telnet firewall management traffic. It shows that Firewall R2 can be managed using Telnet from both the Corporate-Zone and SCADA-Zone.

4.2.1 *Security Violations*

In analysing this network we found several significant violations of the industry standards. These include direct transition of potentially unsafe traffic such as DNS, HTTP, SMTP and FTP between the SCADA-Zone and the Corporate and Internet-Zones (Figure 15). An unsafe protocol such as HTTP can be leveraged to transport a large number

Table 2: Inter-ACL interactions summary.

ACL1	ACL2	Interaction type	Interaction count
scada_access_in	lan1_access_in	shadow	229
corp_access_in	lan2_access_in	shadow	533
corp_access_in	lan2_access_in	generalisation	3

```
1  access-list corp_access_in permit ip host 172.27.0.96 host 172.19.0.100
2  access-list lan2_access_in permit tcp host 172.27.0.96 host 172.19.0.100 eq 445
```

Listing 5: Example inter-ACL rule interaction identified by the Parser.

of attacks and worms in to the SCADA-Zone, from these zones [4]. Moreover, all IP-level traffic was explicitly enabled between the SCADA-Zone and the Corporate-Zone. Such inherently broad rules admit far too many services than necessary, and elevate the risk of a cyber attack on the SCADA-Zone. TFTP was also enabled from the Firewall-Zones to hosts off the Corporate gateway. TFTP does not require user login prior to file transfer and the ISA recommends [4] avoiding its use altogether.

Additionally, EIGRP configuration was not carried out as per Cisco guidelines. Instead of enabling EIGRP *hello multicasts* between neighbours, broad IP-level multicasts were enabled between neighbours. To accommodate unicast EIGRP acknowledgments (responses), similar broad rules were used.

The configurations also enabled local broadcasts to be forwarded between the firewalls using the *ip-forward-protocol* configuration command. However, broadcasts on UDP port 80 were enabled which is not used by a common service.

4.2.2 Configuration Inefficiencies

Our Parser detected that each extended ACL in use contained entries with incorrect source and destination IP addresses. In some cases the order of the addresses were wrong while in others they were simply invalid. There were 70 distinct occurrences of this type. The Parser also found entries that attempted to explicitly block directed broadcasts from propagating between the firewalls, which were redundant as routers by default blocked these out.

We further uncovered that ACLs were copied between the firewalls but were left-in unused, wasting hundreds of lines in the configuration files. Additional copies were also made of these ACLs, renamed and briefly modified to cater for the distinct policy requirements of the serial firewalls. This configuration approach is itself inefficient, but accentuates the overlapping nature of the security policies enforced by a serial firewall configuration.

5. CASE STUDY VARIATIONS

SUC 2 and SUC 4 consisted of Cisco ASA firewalls. ASA allows, varying security levels to be assigned to each firewall interface (from 0 to 100), based on the security policies of the network connected to that interface. The firewall configurations in these case studies also used *object-groups* to classify devices, protocols and ports into groups. These groups were then applied to ACLs in a single rule.

Object-groups can be very useful in practice since the size of a conventional ACL can be large. For example, the ACLs studied in SUC 1 contained on average 237 conventional rules each, as opposed to 6 in SUC 3. With frequently changing rules, managing lengthy conventional ACLs is a difficult task. Using object-group based ACLs makes ACLs smaller, more readable and easier to configure and manage. This and *security-level* are nascent attempts to provide higher-level security policy description, but as we outline here, they don't map clearly to the Zone-Conduit model.

SUC 2 enforced NAT on all traffic traversing from high-security (*i.e.,* inside) interfaces to low-security (*i.e.,* outside) interfaces by default. This meant that unless the traffic matched a specified NAT rule, it was blocked by the firewall. To omit certain traffic having to undergo NAT translation, NAT-exemptions were used. These exemptions were provided via ACLs. Traffic that matched the ACL rules did not have their inside addresses translated when traversing outbound. A NAT-exemption allows both translated and remote hosts to initiate connections.

The firewall in SUC 2 also had an active/standby fail-over configuration enabled through a dedicated Ethernet link. This allows an identical standby firewall to take over the functionality of the primary unit on failure [6]. The primary firewall automatically replicates its configuration to the standby unit once special configuration commands are issued [6]. So the auxiliary unit always has an identical mirroring of the primary unit's configuration. The standby unit is also accessed only via the primary unit, so both are managed as one using a *single Firewall-Zone.*

Each SUC studied included static routes, additionally SUC 3 also utilised OSPF.

SUC 2 also had basic threat detection enabled on the firewall. This captured ACL statistics, recording packet denial rates and connection limit exceeds.

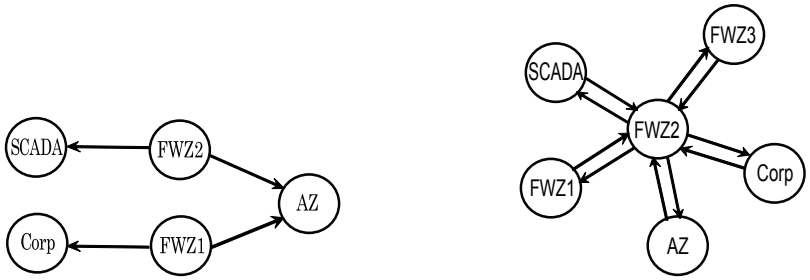
None of the other case studies (SUC 2, 3 or 4) included Abstract-Zones in their Zone-Conduit model. This is because their SUCs did not include any serial firewalls. Only SUC 3 included a Carrier-Zone in its Zone-Conduit model, interconnecting two geographically dispersed networks.

Intra-ACL interactions were present in all SUCs analysed except for SUC 3. These interactions were predominantly shadowed-rules and generalisations.

Inter-ACL interactions were also present in all SUCs. These were mostly shadowed-rules caused by identical rules in distinct ACLs, collectively enabling traffic flow between zones.

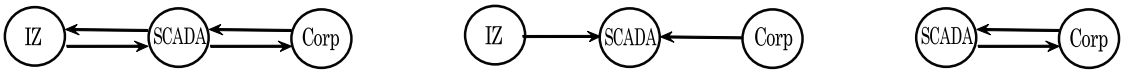
In all the SUCs studied, traffic restrictions were enforced between the Corporate-Zone and the SCADA-Zone through the presence of one or more conduits in the Zone-Conduit model. SUC 1 and 2 allowed direct communication between the SCADA-Zone and the Internet-Zone, violating industry best practices. SUC 3 and 4 met the industry recommendations disallowing such direct communication.

In all cases we found potentially unsafe traffic such as HTTP, Telnet and SMTP explicitly disallowed inbound to



(a) Generic IP traffic. (b) Telnet.

Figure 14: Implicit Service-flow Views (partially shown).



(a) DNS. (b) HTTP. (c) FTP.

Figure 15: Explicit Service-flow Views of DNS, HTTP and FTP traffic.

the SCADA-Zone. But in most cases, such traffic was then implicitly allowed to reach the same zone, breaching industry recommendations. In SUC 2 these implicit flows stemmed from incorrect security-levels assigned on the interfaces.

In most cases we also found explicitly allowed NTP, DNS and FTP traffic transiting directly between SCADA and Corporate Zones. This setup easily exposed the SCADA-Zone to threats or attacks present in the Corporate-Zone.

Implicit rules were used across all case studied to specify firewall management traffic. SUC 3 and 4 utilised HTTP while SUC 2 utilised Telnet.

6. DISCUSSION

Our case studies allowed us to identify the requirements for auto-configuration of firewalls. Most prominent is a good set of high-level abstractions.

Implicit rules are nascent attempts by firewall vendors to provide high-level abstractions, but are too restrictive in that you cannot write flexible rules. For example, Cisco security levels allow quick and easy access between internal and external firewall interfaces, but lack the flexibility to specify detailed traffic restrictions. Hence the large ACLs supplementing these levels.

Likewise, the ANSI/ISA Zone-Conduit security abstraction proved too flexible, allowing alternate ways of defining zones and conduits to cater for business models. The abstraction is good when used by humans, but for automation we need precision.

A good abstraction is therefore, a tussle between these two approaches. It should provide clear mapping between policies and networks, with some restrictions, but also the required amount of flexibility.

For instance, the standards allow 1:n or n:1 mapping between conduits, firewalls and policy. We argue that maintaining a 1:1 mapping between policies and conduits leads to a simple, understandable and useful abstraction for high-level policy specification. Otherwise the ambiguity might lead to specification of policies that breach the restrictions implied by a zone, *i.e.,* a single policy within a zone.

For another instance, when firewalls are placed in series, the best practice is vague about how zones and conduits should be defined. We argue that there needs to be an Abstract-Zone to capture the distinct policies that could be reasonably applied to the two firewalls.

ANSI/ISA best practices also lacked specification on how to precisely capture firewall management traffic. Adding a Firewall-Zone addressed the problem.

Service-flow views also play an important role in auto-configuration. They help verify that the net-traffic flows enabled through firewalls match those specified via high-level policy. Any discrepancy would indicate flaws in the auto-configuration process.

The average firewall configuration length in our case studies, was 684 lines. It is trivial to accidentally leave-in lapsed ACL rules inside a lengthy configuration, when the composition of network devices changes with time. These rules can lead to potentially dangerous latent errors and keep firewall configurations from being concise and up-to-date. An auto-configuration process should therefore, allow detection and removal of obsolete rules.

ACLs and implicit rules can have complex interactions. For example, a rule within an ACL can overlap and conflict with other preceding rules in the same ACL, potentially altering or even reversing its intended effect. With lengthy ACLs, managing interaction free rule-sets manually is a near impossible task but is addressable through automation.

Implicit rules can override ACLs, rendering the effort tendered to the careful design and deployment of ACLs obsolete. For example, consider configuring an ACL on a Cisco firewall interface to block a particular service and an implicit management policy on the same interface to allow that service. The management policy overrides the ACL [6].

We assert that the use of implicit rules should be avoided where possible, and replaced with explicit ACL based access control instead. This will be the difference in being able to automatically generate clear, simple and effective firewall configurations from confusing, complex and ineffective ones.

Our case studies did not comprise large, complex networks. This simplicity implies that the task of configuring

the network firewalls should be relatively easy. Additionally, due to the critical nature of the industrial control equipment protected by these firewalls, one expects them to be correctly configured. As we found, this is far from reality. Even in the simplest of cases, SCADA firewalls are still badly configured! Needless to say, what chances do we have of correctly configuring firewalls in a large, complex network?

We have taken a significant step towards making firewall auto-configuration a reality. By refining the ANSI/ISA Zone-Conduit abstraction we make it precise and complete. Firewall configuration is complex and difficult as re-asserted by our case studies. The refined Zone-Conduit model, provides a precise, simple yet rich high-level abstraction for firewall policy description, that is suitable for automation.

7. CONCLUSIONS AND FUTURE WORK

A SCADA firewall configuration consists of diverse configuration components that have complex interactions, making it intrinsically difficult to manage manually. ANSI/ISA best practices provide a Zone-Conduit model for firewall policy specification. However, it lacks key aspects for automation of firewall configuration.

To address the missing aspects, we propose several extensions. First is to use dedicated Firewall-Zones to precisely capture firewall management traffic requirements. Second, we propose to use Abstract-Zones to cater for distinct policy requirements of serial firewalls. Third, Carrier-Zones should be used to abstract any carrier based transit outside of our control. Finally, maintaining a 1:1 mapping between policies and conduits is also necessary. Through single and multi-firewall case studies, we verified the use of these extensions for high-level policy description.

Several additional requirements of auto-configuration were also identified through this research. Namely, eliminating ACL interactions, avoiding implicit rules, removal of lapsed rules and platform or device specific compilation of high-level policy. We hope to consolidate these requirements further to formulate a feasible firewall auto-configuration design. Our end goal is to realise the design into a software prototype and analyse and test to unveil findings.

8. ACKNOWLEDGEMENTS

This project was supported by the Australian Government through an Australian Postgraduate Award, Australian Research Council Linkage Grant LP100200493, and CQR Consulting.

9. REFERENCES

[1] Y. Bartal, A. Mayer, K. Nissim, and A. Wool. Firmato: A novel firewall management toolkit. *ACM Transactions on Computer Systems (TOCS)*, 22(4):381–420, 2004.

[2] S. Bellovin and R. Bush. Configuration management and security. *IEEE Journal on Selected Areas in Communications*, 27(3):268–274, 2009.

[3] E. Byres. Using ANSI/ISA-99 standards to improve control system security. White paper, Tofino Security, May 2012.

[4] E. Byres, J. Karsch, and J. Carter. NISCC good practice guide on firewall deployment for SCADA and process control networks. *National Infrastructure Security Co-Ordination Centre*, 2005.

[5] W. R. Cheswick, S. M. Bellovin, and A. D. Rubin. *Firewalls and Internet security: Repelling the wily hacker*. Addison-Wesley Longman Publishing Co., Inc., 2003.

[6] Cisco Systems. *Cisco ASA 5500 Series Configuration Guide using the CLI*. Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706, USA, 2010.

[7] Cisco Systems. Cisco ASA 5585-X adaptive security appliance architecture. White paper, Cisco Systems, May 2014.

[8] J. Garcia-Alfaro, F. Cuppens, N. Cuppens-Boulahia, and S. Preda. MIRAGE: A management tool for the analysis and deployment of network security policies. In *Data Privacy Management and Autonomous Spontaneous Security*, pages 203–215. Springer, 2011.

[9] ISA. ANSI/ISA-62443-1-1 security for industrial automation and control systems part 1-1: Terminology, concepts, and models, 2007.

[10] R. Jamieson, L. Land, S. Smith, G. Stephens, and D. Winchester. Critical infrastructure information security: Impacts of identity and related crimes. In *PACIS*, page 78, 2009.

[11] A. Mayer, A. Wool, and E. Ziskind. Fang: A firewall analysis engine. In *IEEE Symposium on Security and Privacy*, pages 177–187, 2000.

[12] T. Nelson, C. Barratt, D. J. Dougherty, K. Fisler, and S. Krishnamurthi. The Margrave tool for firewall analysis. In *LISA*, 2010.

[13] K. Stouffer, J. Falco, and K. Scarfone. Guide to Industrial Control Systems (ICS) security. *NIST Special Publication*, 800(82):16–16, 2008.

[14] T. Tuglular, F. Cetin, O. Yarimtepe, and G. Gercek. Firewall configuration management using XACML policies. *In 13th International Telecommunications Network Strategy and Planning Symposium, Sep*, 2008.

[15] A. Wool. Architecting the Lumeta firewall analyzer. In *USENIX Security Symposium*, pages 85–97, 2001.

[16] A. Wool. A quantitative study of firewall configuration errors. *Computer, IEEE*, 37(6):62–67, 2004.

[17] A. Wool. Trends in firewall configuration errors: Measuring the holes in Swiss cheese. *Internet Computing, IEEE*, 14(4):58–65, 2010.

[18] G. G. Xie, J. Zhan, D. A. Maltz, H. Zhang, A. Greenberg, G. Hjalmtysson, and J. Rexford. On static reachability analysis of IP networks. In *IEEE INFOCOM*, volume 3, pages 2170–2183, 2005.

[19] H. Yan, D. A. Maltz, T. E. Ng, H. Gogineni, H. Zhang, and Z. Cai. Tesseract: A 4D network control plane. *NSDI*, 7:27–27, 2007.

[20] yEd. yEd graph editor manual, http://yed.yworks.com/support/manual/index.html.

[21] L. Yuan, H. Chen, J. Mai, C.-N. Chuah, Z. Su, and P. Mohapatra. FIREMAN: A toolkit for firewall modeling and analysis. In *IEEE Symposium on Security and Privacy*, pages 15–213, 2006.