available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/ijcip

Application of trusted network technology to industrial control networks

*Hamed Okhravi**, *David M. Nicol*

University of Illinois at Urbana-Champaign, 1308 W Main Street Urbana, IL 61801, United States

ARTICLE INFO

Article history:
Received 13 January 2009
Received in revised form
24 June 2009
Accepted 11 July 2009

Keywords:
Industrial control systems
Trusted networks
Firewalls
Security architecture

ABSTRACT

Interconnections between industrial control networks and enterprise networks expose instrumentation and control systems and the critical infrastructure components they operate to a variety of cyber attacks. Several architectural standards and security best practices have been proposed for industrial control systems. However, they are based on older architectures and do not leverage the latest hardware and software technologies. This paper describes new technologies that can be applied to the design of next generation security architectures for industrial control systems. The technologies are discussed along with their security benefits and design trade-offs.

Published by Elsevier B.V.

1. Introduction

The increased interconnectivity of industrial control networks and enterprise networks has resulted in the proliferation of standard communication protocols in industrial control systems. Legacy SCADA protocols are often encapsulated in TCP/IP packets for reasons of efficiency and cost, which blur the network layer distinction between control traffic and enterprise traffic. The interconnection of industrial control networks and enterprise networks using commodity protocols exposes instrumentation and control systems and the critical infrastructure components they operate to a variety of cyber attacks.

Security surveys reveal significant increases in external attacks that target critical infrastructure assets [1]. The percentage of external attacks has increased from 26% (1982–2001) to 60% (2002–2006). The entry points in most of the incidents were corporate WAN, business network, modems, wireless access points, and the Internet.

Several government agencies and industry associations have proposed standards and security best practices for industrial control systems [2–5]. However, these efforts are based on older technologies and security architectures that rely on the differentiation and separation of enterprise and control traffic. While the efforts are, no doubt, important, the underlying security philosophy exposes industrial control systems to attacks that exploit misconfigurations, out-of-band connectivity and blind trust in the identities of traffic sources.

However, new technologies are emerging that provide more pervasive security within networks [6]. These technologies push security from perimeter devices such as firewalls to the networked devices themselves. This paper reviews technologies that can be applied to designing the next generation of secure industrial control systems. The technologies are discussed along with their security benefits and design trade-offs.

* *Corresponding author.* Tel.: +1 217 840 6837.
E-mail addresses: okhravi2@illinois.edu, hamed.okhravi@gmail.com (H. Okhravi), dmnicol@illinois.edu (D.M. Nicol).

1874-5482/$ - see front matter. Published by Elsevier B.V.
doi:10.1016/j.ijcip.2009.07.001

The rest of the paper is organized as follows. Traditional industrial control architectures are explained in Section 2. Security issues related to these architectures are studied in Section 3. Section 4 describes the concept of trusted industrial control network. Section 5 compares the number of rule conflicts in different architectures. The advantages of this architecture and its known issues and attacks are studied in Section 6. Finally, the paper is concluded in Section 7. Related works are presented throughout the paper and mostly in Section 2.

2. Control system security recommendations

Industrial control systems (ICSs) are highly distributed networks used for controlling operations in water distribution and treatment plants, electric power systems, oil and gas refineries, manufacturing facilities and chemical plants. Generally, an industrial complex comprises two distinct networks: a process control network (PCN) containing controllers, switches, actuators and low-level control devices, and an enterprise network (EN) incorporating high-level supervisory nodes and corporate computers [7]. PCN includes supervisory control and data acquisition (SCADA) systems and distributed control systems [2]. The main components of a PCN are the control server or master terminal unit (MTU), remote terminal units (RTUs), intelligent electronic devices (IEDs), programmable logic controllers (PLCs), operator consoles or human–machine interfaces (HMIs), and data historians.

The National Institute of Standards and Technology (NIST), Institute of Electrical and Electronics Engineers (IEEE), Instrumentation Systems and Automation (ISA) Society, International Electrotechnical Commission (IEC) and Industrial Automation Open Networking Association (IAONA) have specified guidelines for securing ICSs (see, e.g., [2–4]). In fact, most security best practices recommend the segregation of PCNs and ENs.

Firewalls are often used to segregate PCNs and ENs [2, 8,7]. A firewall can be configured to block unnecessary services, protocols and ports, thereby providing a higher degree of segregation between a PCN and EN. A router may be positioned in front of the firewall to perform simple packet filtering, leaving the firewall to perform more sophisticated tasks such as stateful filtering and acting as a proxy.

Using a single firewall between a PCN and EN has a serious drawback. This is because the firewall must allow the data historian to have a wide range of access to the PCN. Essentially, each service needs a "hole" in the firewall to operate correctly. Configuring too many holes in the firewall reduces PCN-EN segregation and opens the PCN to a slew of attacks. This problem is typically addressed by creating a "demilitarized zone" (DMZ) [2,8,7].

An architecture deploying a DMZ has three zones: an outside zone containing the EN, an inside zone containing the PCN, and a DMZ containing the data historian. Firewall rules are crafted to make the DMZ historian the sole point of contact between the EN and PCN. The historian can access PCN services that provide it data; in turn, the EN is allowed access to the historian. Firewall rules block access to the PCN by all devices.

Most attacks originating in (or passing through) the EN and targeting the historian will not affect the control systems; at worst, they would corrupt the historian's data (a redundant copy of this data is stored elsewhere).

A PCN architecture deploying paired firewalls separated by a DMZ [18,19] is shown in Fig. 1. It simplifies the firewall rules and achieves a clear separation of responsibility as the PCN-side firewall can be managed by the control group and the EN-side firewall by the IT group [2,8]. This architecture is highly recommended for ICSs, and best practices have been identified for configuring the firewalls (see e.g. [3,4,7]).

There are also mechanisms to enhance host security in a control system. An important example of such mechanisms is process-based security (PBS) which is implemented for some control devices [9]. It shifts the access control paradigm from a user-based model to a process-based one. In the user-based model, access control rules are tied to user identities. A rouge process, however, can escalate its privilege and damage the system. In the process-based security model, on the other hand, fixed access vectors are strictly tied to process profiles which cannot be modified in the field. PBS reduces the risk of privilege escalation as a result of an attack.

3. Security challenges

Firewall configuration errors can lead to security vulnerabilities. One problem is that firewalls often have large rule sets which are difficult to verify. According to a study by Wool [10], firewall rule sets may have as many as 2600 rules with 5800 objects, and a significant correlation exists between rule set complexity and the number of configuration errors. A second problem is that firewalls are usually the main line of defense. Configuration errors enable attackers to exploit holes in a firewall and target the otherwise defenseless devices inside the network.

Wool [10] notes that 80% of rule sets allow "any" service on inbound traffic and insecure access to firewalls. He emphasizes that "the analysis of real configuration data shows that corporate firewalls are often enforcing rule sets that violate well-established security guidelines". The Wool study and others demonstrate that firewall configuration errors pose a real threat to ICS security.

Even properly configured firewalls can be bypassed [11]. This occurs, for example, when a vendor creates a direct (e.g., dial-up) connection to a device for maintenance, or when unsecured wireless access points exist behind a firewall. Firewalls can also be thwarted by tunneling attack traffic using legitimate means (e.g., via a corporate VPN) or by using encryption (firewalls do not inspect encrypted packets). A widely-reported firewall breach occurred in January 2003, when the MS SQL Server 2000 worm infected systems at the Davis-Besse nuclear power plant in Oak Harbor, Ohio [12]. An investigation revealed that a contractor established an unprotected connection to the corporate network which bypassed the power plant firewall and provided a path for the worm.

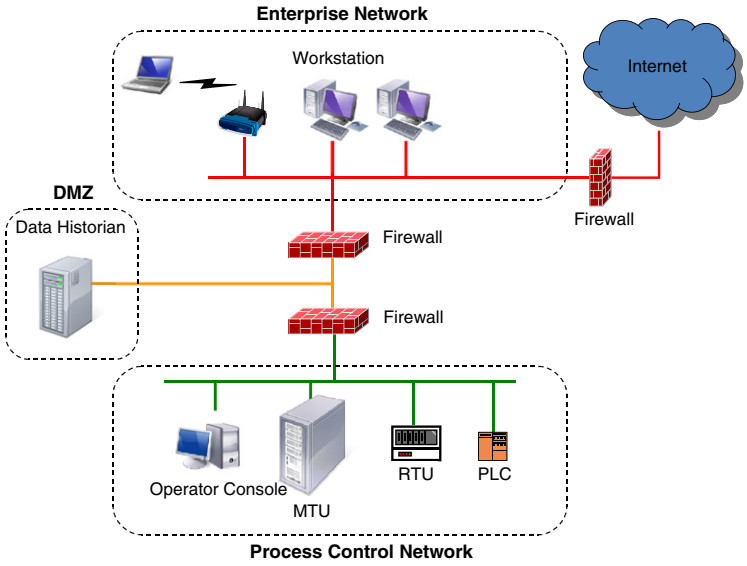

Fig. 1 – Paired Firewalls PCN architecture.

Vulnerable devices are typically secured by patching their services, updating software or installing the latest version of the devices. However, manual patch/update/version management are difficult and costly tasks, especially when careless users introduce vulnerable (wireless) devices into an industrial control network that establish new entry points for attackers. The mobility of wireless devices makes it difficult for the administrator to manually manage patches by visiting the devices frequently or dedicating a specific time slot for patching the systems. Hence, patch/update/version management should be done automatically and continuously.

Unsecured physical access also exposes ICSs to serious security threats. Open wireless access points and Ethernet ports on office walls enable attackers to enter ICS networks and target critical assets. Nothing in the traditional ICS architecture prevents suspect devices from connecting to the network; thus, serious threats are posed by devices whose hardware, operating systems, executables and/or configurations have been tampered with by attackers.

Many ICS vulnerabilities admit malware such as worms, viruses, Trojan horses and rootkits [12,13]. ICS security trends reveal that external malware attacks are becoming increasingly common [1]. Finally, rogue users (insiders) are an ever-present threat to ICSs.

4. Trusted process control networks

In a traditional network access control model, access is granted to a user without considering the security state of the user's machine. That machine may be running a secure operating system, or may be a machine that has not been patched for a decade and is riddled with vulnerabilities and malware. Likewise, firewall access control is agnostic about the security status of the device that sends traffic. A port on a machine is opened or not opened to traffic based entirely on the identity of the source.

A trusted network architecture uses information about the hardware and software states of devices in admission and access control decisions. When a device first "joins" the network, its hardware and software are checked; based on these checks, the appropriate access control rules are applied dynamically to the user, device and traffic. The same principle can be applied to process control architectures. This section discuss technologies that support this concept and their application to ICSs.

4.1. *Trusted networks*

A trusted network (TN) architecture uses the existing standards, protocols, and hardware devices to extend the concept of "trust" to the network architecture. TNs provide important security services such as user authentication, comprehensive network device admission control, end-device health check, policy-based access control and traffic filtering, automated remediation of non-compliant devices, and auditing.

The Trusted Computing Group (TCG) has promulgated industry standards for TNs [14]. Several commercial TN technologies have been developed, including Cisco TrustSec [15], Cisco CleanAccess [16] (formerly known as Cisco Network Admission Control (NAC) [17–19]), and Microsoft Network Access Protection (NAP) [20]. Cisco NAC is interoperable with Microsoft NAP; details about their interoperation can be found in [21].

4.1.1. *Trusted network components*

TN component vendors use a variety of names to describe their products. We use generic terms with a bias towards those adopted by Cisco CleanAccess.

A TN has the following components:

- Client device: Every client device must be evaluated prior to admission to a TN.

- Network Access Device (NAD): All connectivity to a TN is implemented via a network access device (NAD), which enforces policy. NAD functionality may exist in devices such as switches, routers, VPN concentrators and wireless access points.
- Authentication, Authorization, and Access Control Server: The authentication, authorization and access control (AAA) server maintains the policy and provides rules to NADs based on the results of authentication and posture validation.
- Posture Validation Servers: Posture validation servers (PVSs) evaluate the compliance of a client before it can join a TN. A PVS is typically a specialization for one client attribute (e.g., operating system version and patch or virus signature release).
- Posture Remediation Servers: These servers provide remediation options to a client device in the case of non-compliance. For example, a server may maintain the latest virus signatures and require a non-compliant client device to load the signatures before joining a TN.
- Directory Server: This server authenticates client devices based on their identities or roles.
- Other Servers: These include trusted versions of Audit, DNS, DHCP and VPN servers [16,17,19].

4.1.2. *Trusted network protocols*

TNs leverage existing standards and protocols to implement the required security functionality; this reduces the cost of building TNs.

Protocols used in TNs include IPSec for hardening communications [16,18], EAP and 802.1x for authentication [15,18,19], RADIUS /LDAP /Kerberos for directory services and authentication [16,18,19], HCAP for compliance communication [18, 19], and GAME for communication between the AAA and audit servers [18,22].

4.2. *TPCN architecture*

A trusted process control network (TPCN) architecture is presented in Fig. 2. A client device intending to join the network communicates its request to the NAD. The NAD establishes the client device's identity using EAP over the 802.1x protocol and sends the results to the AAA server using the RADIUS protocol. The AAA server returns a list of posture validation requirements and the addresses of the appropriate PVSs.

The client then validates its posture with each of the PVSs. If the client is in compliance, the results are sent to the AAA server using the HCAP protocol. On the other hand, if the client lacks one or more requirements, the appropriate posture remediation servers suggest remediation actions to the client.

The directory server determines the client's group or role. Given all the results from the PVSs and the directory server, the AAA server determines the set of rules that apply to the client's access and traffic and sends them to the NAD for enforcement. From this point on, the client is permitted to communicate via the NAD and all its activities are monitored for policy compliance. Interested readers are referred to [16, 17,19] for additional details.

The policy held by the AAA server is in the form of an authentication requirement and a list of posture validation requirements. For example, token-based authentication may be required and postures must be validated with the anti-virus server, patch management server and driver validation server. When a client device joins the network, an NAD communicates with an AAA server on behalf of the device. The AAA server authenticates the device and provides rules based on the device's security postures to the NAD. From this point on, the NAD enforces the policy on all ingress and egress traffic to/from the device. For example, an RTU with valid firmware is allowed to communicate with the historian; all other traffic is blocked. The two examples below further clarify the workings of a TPCN.

Example 1. Consider a scenario where an analyst on a workstation intends to connect wirelessly to the PCN to access historical data about plant operations. The workstation connects to a wireless access point (AP) in the enterprise network with NAD functionality. The AP applies the default policy, which is to block all traffic except what is needed to establish trust. The workstation then authenticates with the AP using EAP over the 802.1x protocol to send a stored certificate. The AP uses RADIUS to send the workstation's identity to the AAA server. The AAA server then sends the user's identity to the directory server, which knows the user's role ("analyst"). The AAA server uses RADIUS to send the workstation a list of posture requirements (anti-virus version number and OS patch history). The workstation uses a trusted platform monitor (TPM) chip to sign in and send the posture values to the relevant PVSs, which proceed to validate these values. The patch management PVS discovers that the workstation OS has a missing patch and coordinates with the remediation server to have the appropriate patch sent to the workstation. The PVSs transmit the results back to the AAA server using the HCAP protocol. If the workstation is compliant, the AAA sends a rule set to the AP for enforcement. Since the user role is "analyst," the rule set allows TCP connections to the historian but blocks access to all other devices.

Example 2. Consider a scenario where an RTU intends to join the PCN. The RTU connects to a switch on the factory floor via a network cable; the switch has NAD functionality. The protocols used are the same as in Example 1, so we avoid repetition. The switch authenticates the RTU using the RTU's stored token. The AAA server requires the RTU to validate its configuration with a configuration management server. The RTU sends its configuration to the configuration management server, which returns the successful result to the AAA server. The AAA server, in turn, sends the appropriate rule set for the compliant RTU to the switch for enforcement. The RTU may now communicate with other RTUs, the MTU and the historian; the switch blocks all other traffic. In the next section, we show how to perform authentication and posture validation without losing availability for important devices. Essentially, the AAA server holds two sets of roles and two sets of policies. The devices can still connect to the network through the backup policy before we ensure that configuration validation does not interrupt service.

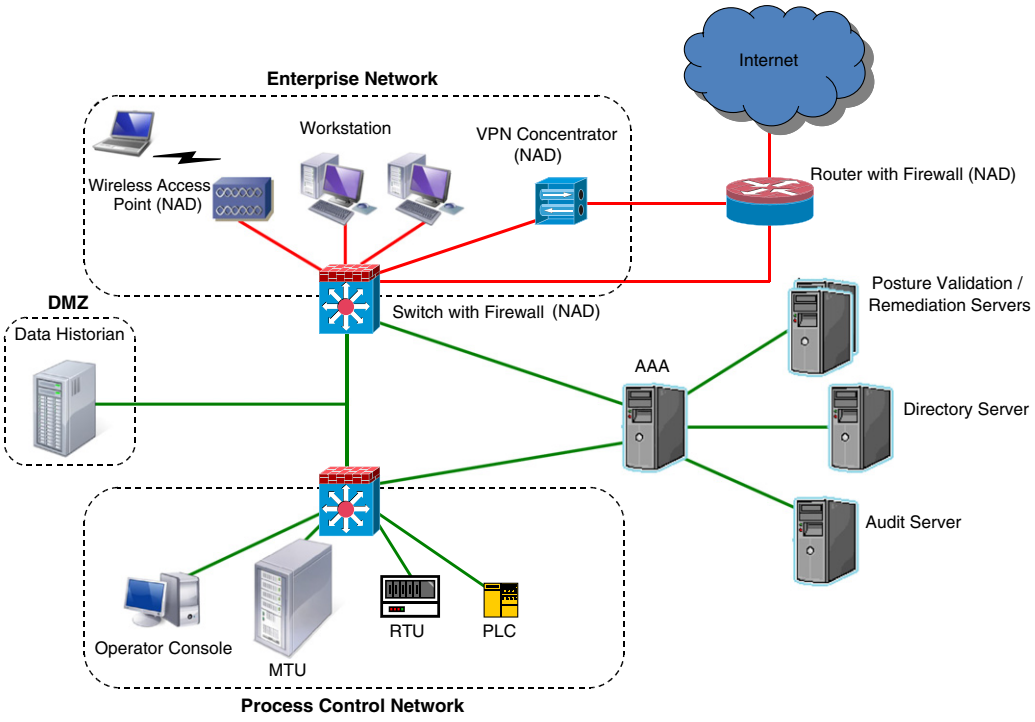

Fig. 2 – Trusted process control network (TPCN).

5. TPCN requirements and availability

In order to transform an existing PCN into a TPCN, there are specific changes necessary ranging from a simple patching of the existing software to new hardware and devices. This section discusses the requirements of a TPCN and the cost/security trade-off. More importantly, the issue of availability is the centerpiece in any process control system architecture. We discuss how to customize a TN to achieve high availability required for a TPCN.

5.1. *TPCN requirements*

For added security and separation of duty, a TPCN requires at least two NADs (switches with firewalls) and an AAA server (Fig. 2). An enterprise can add as many PVSs as required, e.g., an anti-virus validation server to ensure that devices have up-to-date virus protection, a patch management server to check that devices have the correct patches and a software validation server to verify the authenticity of embedded device firmware. Various PVSs are available off-the-shelf and often it is only necessary to configure them with the appropriate requirements for the control system. Incorporating multiple PVSs adds to the cost of a TPCN, but enhances security.

All NADs (switches, routers, wireless access points, etc.) must support trusted network functionality. Many vendors offer products with trusted network functionality. Therefore, if an enterprise is already using new equipment, implementing a TPCN may be very cost-effective. Older systems would likely involve significant upgrades, which can be costly. Note that in a TPCN architecture the firewall functionality is integrated in NADs.

Client devices may need software and firmware upgrades to support trusted network functionality. A trusted network client is required for authentication with the AAA server and for sending posture values. For secure applications, TPM chips can be used to verify configurations and obtain posture signatures. Devices such as RTUs and PLCs do not usually have TPMs; however, as some RTUs already come with built-in web servers, adding TPM to these devices is feasible, especially if government regulations mandate the implementation of trusted ICS architectures.

A client device must be intelligent and configurable to be evaluated by PVSs. At a minimum, the device must be able to run a small piece of software that sends its configuration to a PVS. If the end device is not intelligent (e.g., a simple mechanical relay), it is not necessary to check its configuration; rather, PVSs check the postures of the more intelligent device that controls it (e.g., the RTU that controls the relay.) The trusted network client software is available for different platforms off-the-shelf from various vendors, but for special or small devices it may be necessary to develop the piece of code that sends the configuration to PVSs.

In principle, TN can be implemented on top of any physical and link layer (e.g. wireless, LAN, or serial). However, to the best of our knowledge, the current TN technologies only support LAN and wireless media. There are two options to establish trust with serial control devices using the existing TN technology. The first option is to put the trusted network client software in the main device that is connected to the LAN and controls the serial device. In this case, the client software in the main device is responsible for acquiring the posture of the serial device and performing authentication with the AAA server. The client software on the controlling

device blocks any communication of the serially connected device before the authentication is successful and the posture is validated. This option has the benefit of using the existing hardware. The other option for connecting serial devices to a TPCN is to use serial-to-Ethernet converters [23,24] and directly connect the devices to the NAD. These converters translate serial (RS-232/422/485) to TCP or UDP packets and transmit them over the Ethernet. The converter can be configured to use a specific IP address. Serial-to-Ethernet converters can be used in the compact form [23] to connect one serial device or as a rack [24] to connect multiple serial devices to the Ethernet. Using serial-to-Ethernet converter has the benefit of making serial control devices stand-alone; on the other hand, it has the drawback of extra hardware cost and extra latency due to the network delays. The choice of which option to use when connecting serial devices to a TPCN depends on the acceptable cost and the latency requirements for the specific control system.

5.2. *TPCN availability*

To apply updates to the system, the administrator puts the new requirements in the AAA or posture validation servers. After that point, the AAA server informs end devices of the new policy. If they have the update, they establish its existence with a posture validation server and continue working in the network. However, if they do not have the new requirements, the server provides them with appropriate patches (or installs the patches automatically on them). Now we discuss the use of backup roles and policies to prevent the new requirements from interrupting the service.

TPCNs have the same availability issues as traditional PCNs — applying patches can cause components to crash. Therefore, every patch or update must be tested thoroughly before being placed on the AAA server. Exact replicas of TPCN components should be used for testing. If concerns exist after testing, a backup device may be placed in the TPCN. In such a situation, the AAA server holds two different policies for the device. One policy is associated with the actual role and the other policy with the backup role. The backup policy does not enforce the new requirement on the backup device until the actual device is verified to function correctly with the patch. It is only then that the administrator applies the requirement to the backup device as well. Note that if the actual device is affected by the patch, the backup device can function correctly since it is not required by its policy to have the patch in order to connect to the network. TPCNs do not positively or negatively affect system availability; they merely enforce the requirements. It is the testing phase, before the specification of a requirement, that determines whether or not system availability is affected.

To enhance the availability of a TPCN, redundant servers must be used in the architecture. If important servers of a TPCN fail, devices cannot join the network which endangers the availability of the control system. Commercial TN servers support a mode called “high-availability” (HA-mode.) In this mode, important TN servers (such as AAA or PVSs) have a standby replica available. A “heartbeat” signal is exchanged frequently between the primary and the standby server over a dedicated serial connection or using UDP packets over the Ethernet. If a failover occurs and the heartbeat signal stops (e.g., as a result of a crash or a restart), the standby server immediately takes the role of the failed server preventing an interruption in the service. Since availability has a high priority in the context of process control, it is recommended that HA-mode is used for TPCNs.

It is important to differentiate between backup devices and standby servers. Backup devices and roles prevent failure of the control devices (e.g., MTUs or RTUs) as a result of applying new patches or updates. The use of backup policy ensures that the backup device can provide service if the main device fails after an update. Standby servers, on the other hand, increase the availability of the TPCN infrastructure ensuring that a control device can access the network at any time even when the primary server has crashed (using automatic failover mechanism). These two customizations address the high availability needs of a TPCN, minimizing the risk of service interruption.

6. NAD rule conflicts

Conflict in the firewall rules (more generally referred to as filter conflicts) always endanger the security or availability of instrumentation and control systems. Such conflicts may result in unwanted traffic being allowed to the process control network, exposing it to attacks or legitimate traffic being denied, endangering the availability of the system.

In this section, we study the nature of conflicts in firewall rules and show that the total number of effective conflicts in a distributed filtration environment is no greater than that of a traditional network.

6.1. *Filter conflicts*

Filter conflicts happen when there is an ambiguity in classifying packets using a set of filtration rules. Although we study filter conflicts in the context of firewall rules, they can exist in any network access device such as router, VPN, or QoS device that classifies packets based on a ruleset.

Each filter R is an n-tuple (R[1], R[2], ..., R[n]) where R[i] defines a subset of values acceptable for that field. Each field is usually defined as a prefix. For instance, the prefix 10.* for source IP refers to the subset of all IP quartets that have 10 in their first entry. A rule is a filter R followed by an action *Act(R)* (typically *allow* or *deny*). Although there can be many different fields in each filter depending on the device, we focus on the five most common fields in firewall rules: source IP, source port, destination IP, destination port, and protocol. Note, however, that the analysis presented here is general and can be applied to any n-tuple.

We use the definition by Hari, et al. [25] for rule conflicts.

Definition 1. Rule A is said to be the prefix of rule B if for every field i, A[i] is the prefix of B[i] and A[i] is a strict prefix of B[i] for at least one i.

Two rules conflict with each other if and only if their filters intersect (i.e. there exists traffic to which both rules apply), one is not the prefix of the other, and their actions are not equal. More formally, we have the following.

Definition 2. Rules A and B conflict with each other if and only if all of the following hold:

1. For all i, A[i] and B[i] are not disjoint
2. A is not a prefix of B
3. B is not a prefix of A
4. Act(A) is not equal to Act(B)

Definition 3. A ruleset is an *ordered* set of rules.

For example, consider the following rules in an internal firewall inside a PCN:

```
R1: <PCN.MTU.*    any   PCN.RTU_farm1.*   any   TCP> allow
R2: <PCN.MTU.*    any   PCN.RTU_farm2.*   any   TCP> allow
R3: <PCN.*    any   PCN.RTU_farm1.*   any   TCP> deny
R4: <PCN.*    any   PCN.RTU_farm2.*   any   TCP> deny
R5: <PCN.RTU_farm1.*   any   PCN.*   any   TCP> allow
R6: <PCN.RTU_farm2.*   any   PCN.*   any   TCP> allow
```

The first two rules ensure that any connection from MTU IP addresses to RTU farms is allowed. These two rules do not conflict with any other rule because either the fields are disjoint or they are prefixes of another rule. For instance, R1 is disjoint from R4, R5, and R6 while it is a prefix of R3.

However, there are conflicts between R3 and R6, and R4 and R5. R3 and R6 both apply to traffic from any address in RTU_farm2 to any address in RTU_farm1, yet R3 denies the traffic while R6 allows it. There is a similar conflict between R4 and R5.

In many firewall implementations, the first encountered rule is given higher priority. However, in a situation like this, there can be no ordering which provides the desired behavior. This happens when rule ordering results in a cycle as shown by Hari et al. [25]. The solution in such a situation is to "add" rules that cover the intersection of the conflicting rules to the ruleset. Such rules are called "conflict resolution" rules. For instance, R7 is a conflict resolution rule for R4 and R5.

```
R7: <PCN.RTU_farm1.*   any   PCN.RTU_farm2.*   any   TCP>
deny
```

6.2. *Conflicts in distributed firewalls*

In this section, we show that the total number of conflicts in a distributed firewall environment (such as TPCNs) is no greater than the number of conflicts in traditional architecture. The total number of conflicts in a TPCN is the number of conflicts in all of the network access devices' (NAD) rulesets. Note that the rules are distributed among NADs and each NAD potentially has a much smaller ruleset which makes it less complex and more manageable. Now we show that the total number of conflicts is still bounded by the number of conflicts in the single ruleset model.

To prove this property, first consider the simple "ordered subset" operation on the ruleset. We say that ruleset R_A is an ordered subset of R_B if and only if:

1. Every rule in R_A is in R_B.
2. For each pair of rules R_i and R_j in R_A if R_i precedes R_j in R_B, then R_i precedes R_j in R_A too.

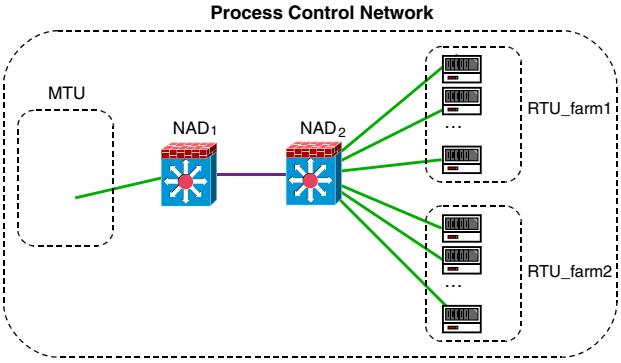

Fig. 3 – NADs and subnets inside a TPCN.

The ordered subset operation is simply taking a number of rules from a ruleset while preserving the order. We cannot state anything about the number of conflicts in R_A. It can even be more than the number of conflicts in R_B. To observe this, consider three rules: R1, R2, and R3 in which R1 and R2 have conflict and R3 is the intersection of the two rules that resolves the conflict. If R_B contains R1–R3 and R_A only contains R1 and R2, then R_B is conflict-free, but R_A has one conflict. The increase in the number of conflicts arises from the fact that the ordered subset operation may eliminate conflict-resolution rules.

Now consider the following method for distributing a single firewall ruleset among different NADs.

1. *Every* rule in the firewall ruleset for which the source or destination is in the subnet controlled by an NAD is added to that NAD's ruleset.
2. For each pair of rules R_i and R_j in the NAD ruleset if R_i precedes R_j in the firewall, it precedes in the NAD ruleset too.

This method puts every rule related to a subnet in the subnet's NAD ruleset while preserving the order. Note that there can be bad rules in the firewall (as a result of configuration mistakes) that do not apply to any subnet controlled by that firewall. These rules are not included in any NAD ruleset. Hence, here we consider "effective conflicts" which involve flows that actually reach the firewall.

Theorem 1. *In a distributed firewall system in which each NAD ruleset is constructed using the above method, the total number of effective conflicts is no greater than that of the single firewall.*

Proof. Suppose not. Let R1 and R2 be the rules that conflict under partition but not in the firewall. Note that R1 and R2 are in the same NAD without a conflict resolution rule, but there must be a rule R3 in the firewall which covers the conflict, yet is not present in the NAD ruleset. R3 is the intersection of R1 and R2, thus more restrictive than either of them. But the definition of the distribution method implies that R3 must be included in the NAD ruleset; hence, the presumed conflict between R1 and R2 does not exist. This is a contradiction.

As an example, consider the architecture in Fig. 3 and the following ruleset.