WHITE PAPER: IRONSHIELD BEST PRACTICES ENHANCING INTERNAL NETWORK SECURITY

Written By: Philip Kwan

March 2003

March 2003
Version 1.0.0

Summary

The IronShield Best Practices: Enhancing Internal LAN Security document is designed to help network and security administrators understand how to implement Foundry security features. The document gives the reasons why the security features are necessary and how to best implement them to compliment existing security devices, such as firewalls and IDS systems, to create a robust secure network infrastructure. The key is "Defense in Depth" and to apply security at all layers of the enterprise.

IronShield Security is not meant to replace Data Security infrastructures. With careful planning and implementation, IronShield Security features can help improve Network Security and enhance Data Security where it's needed.

Contents

Introduction

Foundry's IronShield Security "Best Practices" papers serves as a guide to assist network and security designers in architecting and applying Foundry security features in their networks. Modern enterprise security is applied in layers – Defense in Depth. Consideration must be given to all the various layers with a full understanding of what threats are possible at each layer before implementing a defense strategy. By applying security in multiple layers, the defense is strengthened and vulnerabilities in one layer will not likely lead to successful attacks of corporate resources. The goal is to make it harder to attack your network by layering security chokepoints.

These practices should accompany your Corporate Security and Computer Usage Policies and should not replace them. Applying the steps outlined in this "Best Practices" guide does not guarantee that attacks will not be successful against your security defenses and network resources. The best security design is dynamic. It must be coupled with a strong security policy, proactive network monitoring, diligent network and security staff that is working to stay on top of security alerts and system software upgrades and patches. Enterprise data security is always changing and growing with the advent of new security threats. Thorough, rigorous, and continuous inspection of all security components and processes will help you keep on top of the enterprise network.

This white paper will address the Internal Network Security concerns and demonstrate how Foundry's IronShield Security features can help enhance current security components on the Internal Network. This document is specifically written to work with Foundry products and work in conjunction with related Foundry documentation. Reference to other Foundry documentation is made with regards to command syntax and general feature information.

Audience

IronShield Best Practices White Papers are designed to help the personnel responsible for designing and configuring the network and security components of an enterprise network. The topics discussed are at an intermediate to advanced level and assumes that there is already a good understanding of TCP/IP and related technologies.

Nomenclature

This guide uses the following typographical conventions to show information:

Italic highlights the title of another publication and occasionally emphasizes a word or phrase.

Bold highlights a CLI command.

Bold Italic highlights a term that is being defined.

Underline highlights a link on the Web management interface.

Capitals highlights field names and buttons that appear in the Web management interface.

NOTE: A note emphasizes an important fact or calls your attention to a dependency.

Related Publications

The following Foundry Networks documents supplement the information in this guide.

Foundry Security Guide - provides procedures for securing management access to Foundry devices and for protecting against Denial of Service (DoS) attacks.

Foundry Enterprise Configuration and Management Guide - provides configuration information for enterprise routing protocols including IP, RIP, IP multicast, OSPF, BGP4, VRRP and VRRPE.

Foundry NetIron Service Provider Configuration and Management Guide - provides configuration information for IS-IS and MPLS.

Foundry Switch and Router Command Line Interface Reference - provides a list and syntax information for all the Layer 2 Switch and Layer 3 Switch CLI commands.

Foundry Diagnostic Guide - provides descriptions of diagnostic commands that can help you diagnose and solve issues on Layer 2 Switches and Layer 3 Switches.

The Local Area Network

The term Local Area Network (LAN) was invented in the early 1980's with the popularity of the Personal Computer. PC's gave new processing freedom to users and allowed them to perform their work independent of traditional mainframes or mini-computers. As the numbers of PC's grew, the demand to connect them together soon gave birth to new companies such as Novell, Banyan Vines, and Microsoft - accelerating PC networks and making connectivity easier.

Before the birth of the Personal Computer, computers were terminal based with the central processing unit contained in the mainframe or mini. Guarding corporate data resources was much simpler - usernames and passwords helped authenticate the users, directory and file permissions with access control lists guarded access to information, and physical security was mainly guarding one set of mainframes in a centralized location.

PC's brought new challenges to data security when they became part of the infrastructure. With freedom of processing came heightened data security challenges:

- PC users were able to store files locally on their hard disks.
- Information could be copied to removable media and taken off premise without permission.
- Foreign applications could be loaded and run without permission.
- Backing up critical intellectual property became much more difficult to do.
- A generation of new malicious software known as viruses and trojans were born.
- Physical security of data became a distributed model verses a centralized model.

The popularity of the Internet with the World Wide Web gave us the freedom to search and access information all over the world and the Dot Com era in the 1990's provided the ability to easily conduct business over the Internet - changing the way we conduct business forever. Now more than ever, IT Managers and Security Managers are challenged with designing, maintaining, and securing data infrastructure and services that must be available 24 hours a day, 7 days a week, 365 days a year.

Along this evolution, LANs have become a necessity for every modern business. They are the "glue" that tie all of our computing resources together to allow us to share information in near real-time environments with speeds ranging from 10Mbs to 10Gbs. The amount of information available is becoming immeasurable with advent of the Internet and computer literacy. The applications available to us are growing each day giving us more and more choices on how to create, exchange, and use information.

As technology advances and becomes easier to implement and use, the need to control and secure our information becomes more challenging – requiring IT and security professionals to layer defenses in every facet of the business – Defense-in-Depth.

Defense-in-Depth

Many IT managers may have heard the term "Defense-in-Depth" before and most, if not all, Security managers have learned to live by the expression. In its simplest terms, Defense-in-Depth means deploying security in layers using different technologies and methods to guard the data infrastructure and the intellectual property that resides on or is transferred over it. By combining many different security components at various levels, you are making it much harder to defeat and bypass security chokepoints without being noticed.

Many security experts will tell you that no single security component will protect your data network and applying security at just one area of the network, such as the perimeter, will fail to fully secure the modern network. The challenge is to deploy the right mix of security components and features that your budget, IT resources, and corporate culture will permit.

Modern Network Layers

The modern network is made up of three layers – each with its own components and set of security requirements.

Perimeter Network

The Perimeter is the edge of your corporate network. It is where the corporate network links up with the public Internet and is defined by many security specialists as "the first and last lines of defense" for your network. It is typically made up of the following components:

- Demilitarized Zones (DMZs)
- Screened Subnets
- Firewalls
- Intrusion Detection Systems (IDSs)
- Virtual Private Network devices (VPNs)
- Border Router
- Layer 2 and Layer 3 switches
- Layer 4 – 7 load balancers
- Ecommerce hosts (WEB, database, Ecommerce, etc)
- Infrastructure hosts (Email, DNS, etc)

Internal Network

The Internal Network is beneath or behind the perimeter network and contains the corporate resources that are required to conduct business. Many IT professionals often refer to it as the "trusted network" as corporate culture often deems all employees as "trusted individuals". For this reason, security may be overlooked at this network level. The Internal Network is made up of the following components:

- Layer 2 and Layer 3 switches
- File servers, printers, and other peripherals
- Infrastructure servers
- Storage systems
- Backup systems
- Network infrastructure components such as cable plants, MAN, and WAN connections

Human Factor

The Human or Cultural Factor is the non-technical aspect of the network. This layer of the security design is equally important as the technical and is often overlooked by many IT professionals. The components which make up the Human/Cultural Factor includes the following:

- Corporate Security Policy
- Computer usage guidelines
- User education
- Structured operation practices
- Enforceable policies

Perimeter Security Considerations

Each layer of the modern network requires different security considerations, design, implementation, and operation schedules. The perimeter layer is often considered the most critical layer, as it is the connection point where the corporate network is attached to the public Internet. Most IT and Security managers concentrate their efforts on this layer and spend the majority of security budgets to design and maintain the security components used in the perimeter layer.

The most common security components used in the Perimeter are:

Border Router

Stateless packet filtering, Denial of Service (Dos) features, and device hardening are the most common security features used to protect the border router. Implementing security features at the border router strengthens the security design by:

- Helping off load the firewall by pre-filtering known bad packets.
- Stopping well-known DoS attacks at the edge.
- Checking for proper state of TCP sessions.
- Stopping spoofing of outbound IP Addresses.
- Defending against intrusion of network devices.
- Border router ASICs are generally faster than firewall.

Firewalls

Modern firewalls are stateful. Stateful firewalls can check each packet entering the perimeter network to ensure that each packet is part of an established and permitted connection. Firewalls are traditionally slower than packet filtering routers due to the extra processing and state maintenance required for each session. Two types of firewalls are Stateful Firewalls and Proxy Firewalls.

IDS Sensors

Intrusion Detection Systems (IDS) are devices that gather and monitor network information to identify malicious traffic, intrusion attempts, and break-ins. The placement of IDS systems are key to being able to monitor and spot malicious traffic. The most common IDS placement points are:

- In front of the border router to spot attacks aimed at the Perimeter Network. This generally yields a high volume of IDS alert information as there is no pre-filtering being performed by the border router; but it gives the most visibility for outside intrusion attempts.

- Behind the border router to spot malicious traffic in the DMZ. Less alerts are generated due to pre-filtering activities performed by the border router.
- The Internal Network segment directly behind the firewall. This allows all traffic coming into and leaving the corporate network to be monitored. For installations performing Network Address Translation (NAT) on the firewall, placement behind the firewall allows the internal addresses to be seen by the IDS sensor.
- On the ingress and egress subnet uplinks that house mission critical resources – such as server farms, R&D servers, financial servers, etc. This allows the IT staff to monitor specific resources for attacks targeting the mission critical corporate resources.

VPN Servers	VPN servers create secure tunnels over the public Internet and allow data to be securely transferred over an unprotected medium. Typical uses are to provide secure remote access for employees or business partners and for creating secure cost effective WAN links to connect remote offices or business partners to your network. Security can become an issue if the VPN endpoints are not secure. An example is allowing an intruder to gain access to a remote workstation through a trojan, or some other means, and then gaining access to the corporate network through the VPN tunnel.
Perimeter Routers & Switches	Routers and switches are mandatory at the Perimeter layer. They are the infrastructure components that form the DMZs and Screened Subnets to house the external hosts and applications. The security concerns that routers and switches generate are their accessibility from outside threats and their ability to withstand DoS attacks. The devices must be robust, fast, and secure.

Internal Network Security Considerations

The Internal Network is just as critical to secure as the Perimeter when it comes to security concerns. This network layer is protected by the Perimeter layer and contains the servers, workstations, storage systems, and information that your company requires to conduct business. Because the Internal Network is deemed "trusted", it is often overlooked during security design phases – leaving it in a non-secure state that is open to attacks. With modern networks having full access to the public Internet, the Internal Network's security requirements need to be reviewed more than ever.

Many studies now reveal that internal security threats are just as prevalent as external security threats. Viruses, worms, trojans, destructive applications, password crackers, malicious web applications are just a few tools hackers use to infiltrate the Internal Network. Applications that seem perfectly harmless are downloaded by employees and loaded onto their workstations. Some of these applications are trojan horses in disguise; opening the workstation up to malicious intents ranging from remote reconnaissance, destruction of information, to unauthorized access and possibly remote administrative control of the host. The threat to the Internal Network doesn't have to come from a disgruntle employee or an external intruder - it can come accidentally from an uneducated or careless employee.

The most common security components used for the Internal Network layer are:

Anti-Virus Software	Anti-virus software is the most common defense tool companies use to protect themselves from viruses and other related malicious software. Companies fully understand the consequences and damage that viruses can cause and implement the necessary anti-virus applications to help secure their information. Some of the drawbacks with anti-virus applications include: users not having the most up-to-date virus signatures, not all viruses are caught by the brand of anti-virus software, and anti-virus software can be deactivated by users.
Host Hardening	Many seasoned IT professionals harden hosts that contain mission critical information. Host hardening involves the removal of unnecessary services and applications, protecting or removal of root access and unnecessary accounts, applying the most recent OS or application patches, and configuring tough passwords and account policies. The security implications of this process include: host hardening is resource intensive, not all systems are hardened in a standard way, and documentation on how each system is configured may be lacking.
Standardization	Standardization of configurations ensure that devices and hosts are setup in a standard fashion which makes them more secure and easier to support. There is a set of approved applications and OS versions that must be tested before they are allowed into production. Standardization relies on a coordinated effort and the downfall of this security component is the resources and the human error that can occur between many system administrators.
Network & Security Audits	Audits are a way of checking and comparing what is implemented to what is perceived. It helps identify the areas that are lacking and provides feedback for improvement – it is a good practice to get into as it is preventative maintenance. It's always better when the team conducting the audit finds the weaknesses - before an intruder finds them for you.
Routers & Switches	Many modern switches provide security features such as packet filtering, rate limiting, logging, and alert notification to help add extra layers of security to the Internal Network. These features can help separate resources and control access based on Layer 2 and Layer 3 information and can help contain virus outbreaks and contain intruders that have successfully compromised other security layers.
Firewalls & IDS Sensors	Firewalls and IDS systems can also be applied at the Internal Network layer to provide separation of resources, stateful inspection of packets, control access, and security monitoring. The issues with using traditional firewalls and IDS sensors in the Internal Network include: • Performance considerations with Gigabit and multi-Gigabit uplinks. • Cost of implementing firewalls and IDS sensors can be expensive. • Limitations of mirrored ports confine visibility in terms of what can be monitored effectively. • The resources required to monitor and maintain numerous firewalls and IDS sensors can be overwhelming.

Personal Firewalls

With the popularity of high-speed Internet access came the Personal Firewall. Personal Firewalls are cost effective security applications that help secure individual hosts. They are finding a place in the Internal Network as users and IT professionals are searching for more ways to secure their internal resources. Personal Firewalls can protect a host from trojans, worms, malicious applications, DoS attacks, and external access. Many of these applications come with anti-virus capabilities and can be deployed and maintained from a central management console. The downside of Personal Firewalls include: additional cost and support, outdated firewall rules over time, and the user's ability to uninstall the firewall.

Human Factor Security Considerations

The Human Factor is the non-technical aspect of designing a sound security defense. It includes items such as:

Corporate Security Policy
Computer Usage Policy

A good Corporate Security Policy is a requirement in your security plan. The security policy dictates what security components and measures are implemented and how they are configured to parallel your policies. It contains key points such as:

- The scope of the policy and what it's designed to secure.
- It identifies who has authority to perform which functions.
- Sets an expiration date as to when the policy ends.
- It specifies the exact components and procedures required.
- It's written in a clear and easy to understand format.
- Its designed to be flexible - allowing it to adapt to business requirements.

A good Computer Usage Policy outlines the "do's and don'ts" of using corporate computer resources and points out the common dangers of careless computing practices. It includes topics such as strong password selection, screen savers, how to send corporate information over insecure mediums, the use of unauthorized applications, and so forth.

User Education

Good user education is key to keeping your Internal Network and hosts secure. Making users aware of the Computer Usage Policy and each of the individual "do's and don'ts" will go a long way in helping reduce security risks. Educate users on common social engineering hacks used by intruders. Forms of user education can include:

- Placing the Security Policies in HR handbooks that are given to every employee.
- Have employees sign acknowledgements that they have received, understood, and acknowledged the security policies.
- Place security policies and security FAQ's and tips on internal web sites.
- Send regular security and policy tips through email as reminders.
- Post access security tips close to entrances. Social engineering is an intruder's favorite tool.

Structured Operations	Having all of the security hardware components in place will have little value if the logs and alerts are not monitored. Regular, structured, and diligent review of security logs from all network and security devices is necessary to maintain a strong security posture.
Enforceable Policies	Security and computer usage policies must be approved at the executive level and must be endorsed by Human Resources. To have all the security components in place with an unenforceable security policy will not be effective when time comes to reprimand employees who are in violation of corporate security policies.

No matter how well you prepare for the Human Factor, there will always be cases of non-compliance. Human nature dictates that user's will periodically make mistakes, get careless, and possibly become disgruntle – exposing the Internal Network to security threats. Downloading and installing unauthorized software, using Internet Relay Chat programs to transfer sensitive corporate information, sharing files with peer-to-peer applications, using their laptops on the public Internet without proper hardening, and so forth are sure to pose challenges to your corporate security policy.

The Perfect Security Model

The best security model is layered. Security is applied at each of the various network points in layers, like an onion, making intrusion and hack attempts more difficult as each layer is encountered. Each security feature enhances the next – so if one is compromised, the others are used to slow or contain the attack. A good security implementation will consider all network layers, as well as the Human Factors, and be designed to parallel the Corporate Security Policy and the Computer Usage Policy. Diligent monitoring and alert mechanisms must be in place to help spot intrusions and hack attempts before they damage corporate resources.

No matter how much security defenses are implemented, there is no guarantee that a security threat will not be successful. A carefully thought out and implemented security design will make it much harder for a security breach to occur, but it will never stop all possible security threats. This white paper will help you understand Foundry's IronShield Security features and how to apply them to compliment and enhance your company's existing security defenses. It will concentrate on the Internal Network layer and demonstrate how careful design and application of IronShield Security features can help reduce the risk of exposure and help contain the threats that have made their way into the corporate network.

Foundry IronShield Security

Foundry's IronShield Security solution offers a full range of standard features that come on every Layer 2 – 7 device. These features can be used to enhance your existing security devices and add layers to reinforce your security stance. Refer to Appendix A for a complete list of standard IronShield Security features.

Enhancing Internal Network Security

Security for the Internal Network layer can be enhanced through careful network design and layering of IronShield Security features at critical network chokepoints. If implemented correctly, many of these standard features can help businesses reduce their exposure to internal and external threats and provide containment when attacks and intrusions are successful. For many companies, implementing firewalls and IDS sensors throughout the Internal Network can be a very big task – both in terms of capital expenditure and manpower.

Using IronShield Security's standard features can help lower the cost of purchasing new firewalls and IDS sensors. IronShield Security features do not replace these security components; they enhance existing defenses and often decreases the need to implement additional firewalls and IDS sensors.

IronShield Security features can be grouped into the following categories:

Hardening Network Devices	Stability of routers and switches are critical to providing a fast, robust, and secure networking infrastructure. Intruders and hackers often use network devices as part of their reconnaissance to learn more about the network topology. Being able to groom or manipulate router tables, ARP tables, IP cache tables, and change device configurations can allow intruders to fingerprint your entire corporate network. From this information, they can setup backdoors or shut down critical networking services. Securing all network devices against unauthorized access is the first step to obtaining a secure network infrastructure.
DoS Prevention	Configuring the network devices to limit or block certain protocols can help slow or block Denial of Service attacks that attempt use all of your networks bandwidth or services on a host. This will ensure that your network will continue to function when DoS attacks are launched on the Internal Network. Although not as critical as the Perimeter Network, DoS defenses can be implemented on Internal Network segments that may be exposed to such attacks.
Restricting Access And Containment	One of the golden rules of Defense-In-Depth is to limit access to authorized users. Many businesses' Internal Networks are completely open from end-to-End - allowing users access to every part of the corporate network. By carefully designing smaller subnets to contain critical resources and user activity, you can start compartmentalizing security. This has two distinct advantages – unauthorized users are blocked from critical resources and attacks are contained within subnets when they break out.
Authenticating Access And Host Restriction	In some high security installations or shared common areas where your employees may not be the only people using the network, authentication for using the network may be desired. 802.1x is an authentication technology based on EAP standards that require all users to authenticate to an authentication server before gaining access to the network. Locking down ports using MAC addresses can also limit the physical host that can use the network – providing additional layers of defense against unauthorized users and devices.

Case Study Company Network

To illustrate Foundry's IronShield Security solution for enhancing Internal Network defenses, a fictitious company called Widget-Works.COM will be used throughout this Best Practices guide. Widget-Works.COM employs approximately 600 employees. Most of the employees are located at the head quarter office in San Jose, California where the company occupies three separate buildings in a campus environment. It has one remote sales and training office on the east coast and a VPN server is used to support their sales staff and remote users.

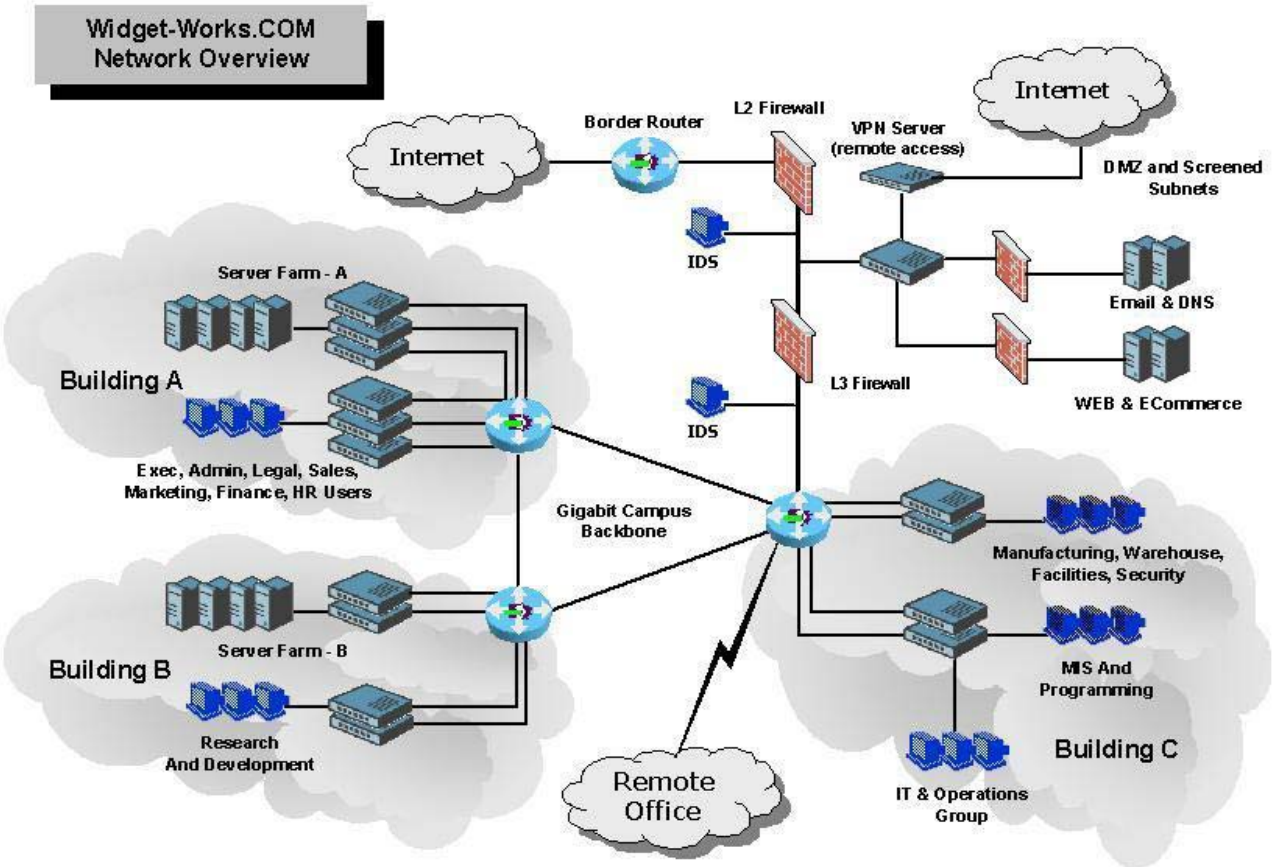


Figure 1. Fictitious Company Widget-Works.COM Network Diagram

The head quarter campus has the following network characteristics.

Building A
The head quarter main building that house the Executive, Finance, Sales and Marketing, HR, and Legal departments. All users must use picture ID badges to gain access and there are cameras at every entrance. Main reception is located in this building and all visitors must check in and register before entering the company. Company personnel must escort all guests.

Server Farm A

Houses all executive, finance, sales, marketing, legal, and general access file and print servers. All of these servers are Windows 2000 servers using Active Directory. Each of the departmental servers are contained in their respective subnets:

Departmental Servers	Subnet
Executive/Admin Servers	10.32.1.0/24
Finance & HR Servers	10.32.2.0/24
Legal Servers	10.32.3.0/24
MIS Development Servers	10.32.4.0/24
Corporate General Storage Servers	10.32.5.0/24
Infrastructure Servers (email, calendar, etc)	10.32.6.0/24

User Subnets

All departmental users for Building A are in their respective areas of the building. As the company grew, there are a few users from each department sitting in other areas outside of their departmental space. The IT department has used VLANs to map the network ports for these users back into their respective User subnets.

Departmental Users	Subnet
Executive/Admin Users	10.33.1.0/24
Finance Users	10.33.2.0/24
Sales & Marketing Users	10.33.3.0/24
Legal Users	10.33.4.0/24
HR Users	10.33.5.0/24

Building B
A research building that house all the inventors and scientists that works on future products. Security is taken very seriously for this building and it is controlled very tightly using cameras, biometric access readers, limited authorized access, and guard patrols. All employees have been educated on physical security, the dangers of social engineering, and the company's Computer Usage Policy and Security Policy. They have monthly reminders and the IT department performs spot checks on computer configurations and user practices to ensure that policy is maintained.

Server Farm B

Houses all research and development servers. The intellectual property on these servers are mission critical to the company's survival. Competition for Widget-Works.COM is very high and biometric access is used to gain access to the server farm. Critical servers are locked behind server racks with a limited distribution of keys to key personnel only.

Departmental Servers	Subnet
Research and Development Servers	10.48.1.0/24
Staging and Testing Network	10.48.2.0/24

User Subnets

All users in Building B report into one R&D manager and the Research & Development group is in only department in this building. There are two groups of users. The first group is researchers and scientific staff and the second group is development and quality assurance.

Departmental Users	**Subnet**
Research and Development Users	10.49.1.0/24
Developers and Quality Assurance	10.49.2.0/24

Building C

Building C contains the departments that support the corporate infrastructure. The MIS department develops and supports all internal applications. Manufacturing and warehousing departments produce and ship the products. IT and Operations administer and support all corporate computer and telecom functions. Facilities and Security departments support the physical building and provide the necessary guard patrols. All users must use picture ID badges to gain access and there are cameras at every entrance.

User Subnets

There are multiple departments in Building C, each with their own subnet. Building C also contains the router that supports the Remote Office on the east coast.

Departmental Users	**Subnet**
MIS Programming Users	10.64.1.0/24
Manufacturing And Warehousing	10.64.2.0/24
IT Users	10.64.3.0/24
NOC Operations	10.64.4.0/24
Facilities and Security Users	10.64.5.0/24

WAN Network Information	**Subnet**
WAN Subnet	10.200.100.0/24

DMZ Perimeter Network

The Perimeter Network containing the companies DMZ, screened subnets, external Web servers, external email server, and external DNS server are kept in a secure server room in Building C. A dual firewall design is used to create the DMZ and extra firewalls have been used to create the Screened Subnets within the DMZ to house the external systems and services. The first firewall (closest to the border router) operates in Layer 2 and acts as pre-filtering firewall to restrict unwanted traffic after the border router's packet filtering rules. The second firewall (closest to the corporate internal network) operates in Layer 3 and also performs Network Address Translation for the internal 10.0.0.0/8 address space.

The Perimeter Network uses a class C subnet assigned by their ISP - 198.30.15.0/24. NAT is performed by the firewall joining the company's private 10.0.0.0/8 network to the 198.30.15.0/24 Perimeter Network.

DMZ Network Information	**Subnet**
ISP's IP Network	196.100.100.2/28
External Public IP Addresses	198.30.15.0/24
Internal IP Network to DMZ Connection	10.64.254.0/24
NAT'd External IP	198.30.15.253

The company supports remote users with a VPN server located in the DMZ. It has a dedicated Internet connection and a built-in firewall that only allow the VPN's IKE and IPSec protocol port numbers – all other traffic is automatically dropped by the external interface. A RADIUS server performs access authentication and every

remote user is required to use two factor security token cards that have randomly changing passkeys. Policies on the VPN server dictate where and what each remote user can access and it's primarily based on the user's departmental access policies. The pertinent VPN IP address information is:

VPN Server Information	**Subnet**
VPN Address Pool	198.30.15.10/24 thru 198.30.15.60/24
Management Address	198.30.15.1/24
Internal Ethernet Address	198.30.15.2/24

Widget-Works.COM has created a separate network to support visitors and guest access. Special visitor network ports are provided in each conference room, lobby, guest office, and other shared areas. These ports are aggregated together throughout the campus and placed into one guest network.

Guest/Visitor Network Information	**Subnet**
Guest Network	198.30.15.61 thru 198.31.15.80

Remote Office

Widget-Works.COM has one remote office on the east cost. This is primarily a sales and training office with about 25 employees. There is one Windows Domain Controller (running DNS), one Windows file and print server, and one remote email server that support the employees. There are several training servers that are on a separate isolated network in the training room. The remote office is connected to Head Quarters with a dedicated T1 link.

User Subnets

There is only one user subnet in the remote office. The training subnet is isolated on its own set of network switches and is not connected in any way to the corporate network.

Departmental Servers	**IP Address**
Remote Office Domain Controller	10.96.1.250/32
Remote Office File & Print Server	10.96.1.240/32
Remote Office Email Server	10.96.1.241/32

Departmental Users	**Subnet**
Sales and Training users	10.96.1.0/24
Training Room	192.168.1.0/24

WAN Network Information	**Subnet**
WAN Subnet	10.200.100.0/24

Other Information

The company uses a Windows Active Directory domain to manage all their servers. Nearly all of their file servers are Windows 2000 servers with a small population of UNIX servers in the Research and Development department. Most of their workstation PC's are using Windows XP and the IT group is very good about keeping on top of the latest OS patches and security fixes – Active Directory GPO's are used to roll out service packs and updates to critical applications.

There is a Windows Domain Controller in each of the buildings to provide authentication and directory services to the user community. Other than authentication, browsing, and global catalog services, there are no other applications running on the Domain Controllers – they are strictly dedicated to the Windows infrastructure. DNS

is configured on each of the Domain Controllers and Widget-Works.COM uses a split DNS architecture in the DMZ to safeguard the private name space.

Building A Domain Controller	10.33.4.250/32
Building B Domain Controller	10.49.1.250/32
Building C Domain Controller	10.64.1.250/32
Remote Office Domain Controller	10.96.1.250/32

There is also a Windows Print Server in each building to provide printing services to each of the departments.

Building A Print Server	10.33.1.240/32
Building B Print Server	10.49.1.240/32
Building C Print Server	10.64.1.240/32
Remote Office File & Print Server	10.96.1.240/32

The Gigabit Core runs OSPF and each building is configured with a redundant path to another neighboring building on the campus.

Backbone Network Information	**Subnet**
Building A – Building B	10.200.1.0/24
Building A – Building C	10.200.2.0/24
Building B – Building C	10.200.3.0/24

Hardening Foundry Routers & Switches

Hardening Foundry network devices involves several key areas:

- Guarding against unauthorized access
- Setting up Warning Banners to support enforceability
- Restriction of management stations to specific IP addresses, ports, or VLANs
- Implementing secure access methods such as SSH and SNMP v3
- Removing unnecessary services and functions
- Securing routing protocols
- Guarding against common DoS, ICMP, SYN, and Fragmentation attacks
- Creating a robust centralized logging strategy with synchronized logs
- Locking down unused ports
- Physically securing your network infrastructure devices and cable plants
- Staying current with the latest Foundry patches and OS releases

This white paper will concentrate on enhancing security for the Internal Network. For detailed information on Hardening Foundry Routers & Switches, refer to the following Foundry document:

White Paper: IronShield Best Practices
Hardening Foundry Routers & Switches

Denial of Service (DoS) Prevention

Preventing Denial of Service attacks on the Internal Network is becoming just as important as protecting the DMZ and Screened Subnets from DoS attacks. DoS and Worm attacks are becoming much more sophisticated. If crafted correctly, they can bypass traditional firewalls, IDS sensors, and packet filters to affect internal hosts. With the popularity of Internet Relay Chat programs peer-to-peer file sharing applications, vulnerabilities in Web browsers and email systems, employees can inadvertently infect internal hosts.

By implementing Foundry's IronShield Security features, you can decrease the chances of DoS and Worm attacks spreading throughout your business. There is no guarantee that implementation of these features will block all DoS and Worm attacks, as so many are crafted to use standard application protocol ports. For the attacks that you can't block, the next defense goal is to slow them down; to be alerted quickly of their presence and to contain them from spreading throughout the network.

The features that Foundry's IronShield Security solution can provide to enhance Internal Network defenses against DoS and Worm activity include:

- Anti-Spoofing protection
- Smurf attack protection
- TCP SYN Flood protection
- LAND Attack protection
- Proxy ARP protection
- ARP Attack prevention
- ICMP Attach prevention
- Broadcast Attack prevention
- UDP Broadcast prevention
- Fragmentation Attack prevention
- Containment through separation of resources

Stopping IP Spoofing

Denial of Service attacks and Worms often use spoofed addresses to spread and hide their tracks. By spoofing addresses, they hope to generate random infections and DoS conditions to many hosts throughout the Internet or your enterprise. Spoofing source addresses allows them to hide the original source of the packet and spoofing destination addresses allows them to attack many random hosts simultaneously.

Stopping spoofed IP packets from leaving your corporate network can greatly reduce this type of attack. If every ISP, business, educational institution, government organization, etc implemented anti-spoofing technology on their router and switches, the spread of attacks relying on spoofed source addresses can be greatly reduced. Stopping IP Spoofing is a simple task. It involves the use of anti-spoofing ACLs at each router gateway port within the Internal Network and one set of anti-spoofing ACLs at the border router.

Foundry Access Control Lists (ACLs)

Foundry ACLs are implemented differently depending on the version of hardware and OS your device is running. Older IronCore ASICs use Flow Based ACLs that are CPU driven. JetCore ASICs with OS version 07.6.01 and later can program ACLs into hardware (CAM) and either permit or deny traffic much faster and more efficiently than Flow Based ACLs – JetCore ACLs do not rely on the CPU to make decisions. For a complete reference between IronCore and JetCore ACL differences, refer to the following Foundry web link:

http://www.foundrynet.com/services/documentation/ecmg/ACL-rule-based.html#53096

Important points for Flow Based ACLs (IronCore) are:

- When a packet is received by the port, it checks the ACL CAM entries to see if the packet matches any of the same address information in the CAM. If there is a match in CAM, it uses the permit or deny rules to filter the packet.
- Deny ACLs normally send all packets to the CPU for processing. CAM entries for Deny ACLs can be created by programming the device with the **hw-drop-acl-denied-packet** command. This will use more CAM memory.
- Inbound ACLs are more efficient than outbound ACLs because the packet can be dropped by CAM on the inbound side without having to pass through the CPU first.
- Outbound Deny ACLs must be processed by the CPU before being passed or dropped.
- ACLs that use the log option to record matches to the system log are sent to the CPU for processing. To preserve performance while enhancing security, only turn on logging for critical Deny ACLs.
- ACLs that match on the ICMP protocol type are passed to the CPU for processing.
- If ACL Accounting is enabled, the packet is sent to the CPU for accounting purposes. To preserve performance, you can leave the accounting feature turned off unless it's absolutely required.

Important points for Hardware Based ACLs (JetCore) are:

- Inbound ACLs are programmed into Layer 4 CAM hardware for both Permit and Deny lists. One entry is used in the CAM for each ACL. Extended ACLs that match of multiple ports will require one CAM entry for each port.
- Hardware Based ACLs do not time out. They remain in CAM until they are removed or manually changed to Flow Based ACLs. Hardware Based ACLs are much faster than Flow Based ACLs and a maximum of 4096 ACLS can be programmed into the Layer 4 CAM.
- ACLs that use the log function must pass the packet to the CPU for the logging to take place.
- ACLs that match on ICMP type must pass the packet to the CPU for processing.
- If ACL Statistics is turned on, the packet must be passed to the CPU for processing.
- Outbound ACLs programmed on the interface are required to pass through the CPU. The exception is when they are used on NPA POS OC-48 ports. In this case, outbound ACLs are supported in hardware.

General Guidelines for Creating ACLs

When creating ACLs for layering security, remember the "important points" for Flow Based ACLs and Hardware Based ACLs and how the CPU is involved in each ACL that is implemented. By remembering these points and taking the following guidelines into your design consideration, you can maintain performance and enhance security defenses at the same time.

- Inbound ACLs are more efficient than outbound ACLs.
- Put the most specific rules at the top of the ACL list and the most general rules towards the bottom.
- Once a packet satisfies an ACL rule, it is executed by that ACL and stops flowing down the ACL list.
- Foundry uses an "implied deny" at the bottom of each ACL list - the last action is always to deny access.
- Turning on logging uses the CPU. To maintain peak performance, log traffic only when it's absolutely necessary.
- For security chokepoints that require high bandwidth, use JetCore products to implement Hardware Based ACLs.

- Load balance your ACL rules throughout the enterprise on both Layer 2 and Layer 3 devices to distribute security loads. By using Layer 2 ACLs, you can push security defenses to the edge where user and file server resources reside – stopping and containing malicious traffic sooner.
- For IronCore products, you can speed Deny ACLs by using the **hw-drop-acl-denied-packet** command.
- If changes are made to existing ACLs, they must be rebound to the interface using the **ip rebind-acl** command to take effect.

Stopping Inbound IP Spoofing From The Internet

Inbound IP Spoofing from the Public Internet occurs when an intruder attempts to spoof the source IP address on the packet with one of your DMZ or Internal Network IP addresses. By doing this, they hope to bypass packet filters, firewall rules, or IDS sensors that are programmed to permit traffic based on a valid source IP address from one of your own hosts. Since only devices behind your border router should have the internal or DMZ IP addresses, any packets coming in from the pubic Internet claiming to be from your corporate network has been crafted.

To stop this activity, you can setup inbound ACLs to block all packets coming from the Internet with source IP addresses used on your company's internal networks. As a precaution, you should also block any packets with a source IP address claiming to be from any one of the reserved or private networks, loopback address, broadcast address, multicast networks. The list includes the following:

- 127.0.0.0/8 (loopback address)
- 10.0.0.0/8 (private network)
- 172.16.0.0/12 (private network)
- 192.168.0.0/16 (private network)
- 224.0.0.0/4 (multicast network)
- 240.0.0.0/5 (multicast network)
- 255.255.255.255/32 (broadcast address)

Apply inbound anti-spoofing ACLs to all inbound router ports that are connected to the Public Internet or partner networks.

EXAMPLE:
Using Widget-Works.COM as an example, an ACL applied on the border router's inbound public interface, port Eth 16, would resemble the following ACL. For performance, only spoofed packets claiming to be from the company's DMZ Perimeter Network is logged. The company uses a Foundry JetCore Layer 3 switch for the border router to take advantage of the Hardware Based ACLs.

```
BRouter001(config)# access-list 10 deny 198.30.15.0/24 log
BRouter001(config)# access-list 10 deny 127.0.0.0/8
BRouter001(config)# access-list 10 deny 10.0.0.0/8
BRouter001(config)# access-list 10 deny 172.16.0.0/12
BRouter001(config)# access-list 10 deny 192.168.0.0/16
BRouter001(config)# access-list 10 deny 224.0.0.0/4
BRouter001(config)# access-list 10 deny 240.0.0.0/5
BRouter001(config)# access-list 10 deny 255.255.255.255/32
BRouter001(config)# access-list 10 permit any

BRouter001(config)# interface ethernet 16
BRouter001(config-if-16)# ip access-group 10 in
```

Stopping Outbound IP Spoofing From The Internal Network

Outbound anti-spoofing ACLs are used to prevent packets with spoofed source IP addresses from leaving your company. Outbound anti-spoofing ACLs should be applied to all inbound router ports and the border router to block spoofed packets from entering the corporate backbone and the public Internet. Spoofed source IP address packets originating from your Internal Network can come from several sources:

- Hosts infected with a worm, trojan, or other form of malicious code that is using it as a zombie host to perform distributed attacks against others.
- Unauthorized users on your Internal Network performing distributed attacks using crafted packets.
- Employees attacking other victim hosts using tools requiring spoofing of source IP addresses; such as a Smurf attack.
- Mis-configured host with the wrong IP address or mask information.

When designing your defense strategy using anti-spoofing ACLs, remember the ACL guidelines and take the performance implications of each security feature into consideration. The best location to place anti-spoofing ACLs which guard against invalid packets leaving your corporate network is on each of the Internal Network router interfaces that are used to create your internal subnets. Using inbound anti-spoofing ACLs on each Internal Network router gateway has the same effect as an outbound anti-spoofing ACL implemented at the border router.

By implementing anti-spoofing ACLs at every Internal Network router port, the following performance and security benefits are achieved:

- Inbound ACLs can be used to obtain better ACL performance than an Outbound ACL at the border router.
- Every subnet is protected by its own set of anti-spoofing ACLs to load balance the security load across multiple router ports.
- Spoofed packets are dropped at the first Layer3 ingress point and don't affect the performance of the corporate backbone. DoS and Worm attacks using this technology is thwarted earlier and contained within one subnet.
- By blocking spoofed packets at the server and user subnet router interfaces, anti-spoofing ACLs are not needed for the WAN and Backbone router ports.
- The border router only needs to apply one set of outbound anti-spoofing rules for each interface.

EXAMPLE – Building Routers:
Widget-Works.COM has the following number of subnets in each building:

Building A	6 Server Subnets and 5 User Subnets
Building B	2 Server Subnets and 2 User Subnets
Building C	5 User Subnets, 1 Publicly Routable Subnet, 1 Connection Subnet, 1 WAN Subnet
Remote Office	1 User Subnet, 1 WAN Subnet

Using Building B as an example to illustrate the implementation of anti-spoofing ACLs to block spoofed source IP address packets from leaving Widget-Works.COM, the security design takes the following subnets and backbone connections from Building B into consideration.

- Research and Development Servers 10.48.1.0/24
- Staging and Testing Network 10.48.2.0/24
- Research and Development Users 10.49.1.0/24

- Developers and Quality Assurance 10.49.2.0/24
- Building A – Building B 10.200.1.0/24
- Building B – Building C 10.200.3.0/24

Building B will require the following anti-spoofing ACLs to block outbound spoofed IP packets. Because all spoofed packets are dropped at the server and user subnet ingress router ports, anti-spoofing rules for the Backbone interfaces will be redundant and inefficient. All ACLs are applied in the inbound direction to boost performance and all spoofed packets are logged.

```
BigIron-BuildingB(config)# access-list 21 permit 10.48.1.0/24
BigIron-BuildingB(config)# access-list 21 deny any log

BigIron-BuildingB(config)# access-list 22 permit 10.48.2.0/24
BigIron-BuildingB(config)# access-list 22 deny any log

BigIron-BuildingB(config)# access-list 23 permit 10.49.1.0/24
BigIron-BuildingB(config)# access-list 23 deny any log

BigIron-BuildingB(config)# access-list 24 permit 10.49.2.0/24
BigIron-BuildingB(config)# access-list 24 deny any log

BigIron-BuildingB(config)# interface ethernet 1/1
BigIron-BuildingB(config-if-1/1)# ip address 10.48.1.254/24
BigIron-BuildingB(config-if-1/1)# ip ospf area 0.0.0.2
BigIron-BuildingB(config-if-1/1)# ip ospf hello 4
BigIron-BuildingB(config-if-1/1)# ip ospf dead 16
BigIron-BuildingB(config-if-1/1)# ip access-group 21 in

BigIron-BuildingB(config)# interface ethernet 1/2
BigIron-BuildingB(config-if-1/2)# ip address 10.48.2.254/24
BigIron-BuildingB(config-if-1/1)# ip ospf area 0.0.0.2
BigIron-BuildingB(config-if-1/1)# ip ospf hello 4
BigIron-BuildingB(config-if-1/1)# ip ospf dead 16
BigIron-BuildingB(config-if-1/2)# ip access-group 22 in

BigIron-BuildingB(config)# interface ethernet 1/3
BigIron-BuildingB(config-if-1/3)# ip address 10.49.1.254/24
BigIron-BuildingB(config-if-1/1)# ip ospf area 0.0.0.2
BigIron-BuildingB(config-if-1/1)# ip ospf hello 4
BigIron-BuildingB(config-if-1/1)# ip ospf dead 16
BigIron-BuildingB(config-if-1/3)# ip access-group 23 in

BigIron-BuildingB(config)# interface ethernet 1/4
BigIron-BuildingB(config-if-1/4)# ip address 10.49.2.254/24
BigIron-BuildingB(config-if-1/1)# ip ospf area 0.0.0.2
BigIron-BuildingB(config-if-1/1)# ip ospf hello 4
BigIron-BuildingB(config-if-1/1)# ip ospf dead 16
BigIron-BuildingB(config-if-1/4)# ip access-group 24 in
```

All the other Building Routers should be configured in the same way for all of the server and user subnet router ports. The 10.64.254.0/24 subnet that connects the internal 10.0.0.0/8 network to DMZ firewall will not have an inbound anti-spoofing ACL configured. This subnet will have source IP addresses from many sources to reflect the public Internet traffic entering the corporate network.

EXAMPLE – Border Router:
The border router will require anti-spoofing ACLs to be set on the router interface that defines the DMZ Perimeter Network – 198.30.15.0/24. Remember that all internal 10.0.0.0/8 IP addresses are translated using NAT at the firewall joining the 10.0.0.0/8 network to the 198.30.15.0/24 Perimeter Network so there should never be any 10.0.0.0/8 addresses traversing the Perimeter DMZ network. The only source IP addresses leaving the DMZ for the Public Internet should have a proper 198.31.50.0/24 address.

The following anti-spoofing ACL is required on the border router's Ethernet interface supporting the 198.30.15.0/24 network. It is applied as an Inbound ACL to boost performance and all spoofed source IP address packets are logged.

```
BRouter001(config)# access-list 20 permit 198.30.15.0/24
BRouter001(config)# access-list 20 deny any log

BRouter001(config)# interface ethernet 1
BRouter001(config-if-1)# ip access-group 20 in
```

Stopping IP Address Spoofing – Host Protection

Infected hosts are often used to spread DoS and worm attacks. To fully protect the network, consideration should be given to hosts that may be vulnerable to attack. Broad IP Anti-Spoofing ACLs placed at router ingress points can help block packets with spoofed source IP addresses from leaving the subnet, but it doesn't stop the spread of spoofed source IP address packets within the subnet. This section will discuss how to block and prevent spoofed source IP addresses at the host level.

During DoS and worm attacks, the goal is to block the malicious activity at the source. By blocking the source, the chance of the worm or DoS attack spreading is decreased significantly and the amount of malicious traffic entering the subnet and the corporate backbone is decreased. During the attack, the Foundry device performing the Anti-Spoofing ACL protection may undergo tremendous stress depending on how the malicious traffic is crafted and formatted. Under heavy attacks, this security design will sacrificing the performance of the Layer 2 device performing Anti-Spoofing protection and save the upstream devices, switches, and routers – limiting the damage to the least number of hosts and devices.

To protect hosts that may be used as sources for worm and DoS attacks, IP Anti-Spoofing ACLs can be applied on the Layer 2 switch ports used to connect the hosts. Consider the following ACL guidelines when designing Layer 2 ACL security defenses for individual hosts:

- Protect only the hosts configured with static IP addresses. DHCP clients cannot use IP Anti-Spoofing ACLs as their source IP addresses will change.
- Not all hosts will need protection. Concentrate on the hosts that are most vulnerable to such attacks. The CERT advisory board can help in making these decisions: http://www.cert.org/advisories/
- On older Foundry IronCore devices, Flow Based ACLs are implemented in software. Turning on many Layer 2 ACLs may impact the performance of the device.
- Foundry JetCore devices implement hardware Rule-Based ACLs and are much faster than Flow-Based ACLs.
- Inbound ACLs are more efficient than Outbound ACLs.
- There is a limit to the maximum number of standard ACLs (1–99) and Extended ACLs (100 – 199) that can be defined on each device.
- The maximum number of ACLs that can be programmed in the Foundry device will depend on the hardware and software revision.

EXAMPLE:
Widget-Works.COM has seen the destructive forces of several DoS and Worm attacks that have crippled the Internet and other companies. To prevent their servers from becoming casualties and being used as sources for attacking other hosts and companies, their Corporate Security Policy has dictated the use of Layer 2 IP Anti-Spoofing ACLs on all critical servers exposed to the Internet; Web servers, database servers, DNS servers, and email servers.

The IT team has turned on Layer 2 IP Anti-Spoofing ACLs for every host in their DMZ and Screened Subnets. In order to protect the internal file servers, they have also implemented Layer 2 IP Anti-Spoofing rules on each of the file server switch ports that may be likely candidates for such attacks. To make sure these additional security defenses would not lessen server farm performance, Widget-Works.COM upgraded the necessary switches to Foundry's JetCore technology using Rule-Based ACLs.

To protect a file server with the IP address 10.32.6.11, an inbound standard ACL resembling the following is used on interface Ethernet 3/15; the port that the file server is connected to.

```
FI_001-BuildingB(config)# access-list 10 permit host 10.32.6.11
FI_001-BuildingB(config)# access-list 10 deny any
FI_001-BuildingB(config)# int eth 3/15
FI_001-BuildingB(config-if-3/15)# access-group 10 in
FI_001-BuildingB(config-if-3/15)# write memory
```

If the number of file servers to be protected is greater than the permissible number of standard ACLs that can be defined on the device, defend the servers that would most likely be a victim of such attacks and let the IP Anti-Spoofing ACL defined on the inbound subnet router port defend the rest of the subnet. The IP Anti-Spoofing ACL defined on the router port will block any spoofed source IP addresses that originated on the subnet.

Stopping Smurf Attacks

In a Smurf attack, the attacker crafts an ICMP ping request by spoofing the source IP address of the desired victim and sending it to the broadcast address of a specific subnet. By allowing the ICMP request to hit the broadcast address, every device that can respond to the ping request will respond simultaneously back to the spoofed source address – the victim host. The victim host receives hundreds or perhaps thousands of ICMP echo responses from the Smurf attack and gets overwhelmed by the surge of packets.

Foundry has several ways of protecting your network from ICMP attacks. The first method is to stop all corporate routers from forwarding directed-broadcasts. A Smurf attack must have an intermediary that propagates the ICMP request to the broadcast address and that intermediary is the router. To block ICMP ping requests to a subnet's broadcast address, use the "**no ip directed-broadcast**" command.

The second method is to prevent hosts and network devices from being a victim in a Smurf attack by controlling the number of ICMP packets that they can receive. Foundry has a unique command that allows you to configure what the normal-burst and maximum-burst levels of ICMP are on the device and the lockout duration to block excessive ICMP traffic once the maximum-burst level has been reached. This command can be used to protect the Foundry device as well as hosts connected to the device.

Syntax: [no] ip directed-broadcast

Syntax: ip icmp burst-normal <value> burst-max <value> lockup <seconds>

The <value> is the number of ICMP packets/second
The lockup <seconds> specifies the duration to drop excessive ICMP packets

EXAMPLE:
For each Widget-Works.COM router, implement the command to block ICMP ping request to the each of the subnet broadcast addresses. Widget-Works.COM's corporate security policy also states that all routers and switches must be protected from Smurf and ICMP attacks by limiting the number of ICMP packets a network device and host can receive.

These commands are implemented at the global level for each building router. All building routers, the border router, and Layer 2 switches must be configured.

On the border router:
```
BRouter001(config)# no ip directed-broadcast
BRouter001(config)# ip icmp burst-normal 5000 burst-max 10000 lockup 60
BRouter001(config)# write memory
```

On each of the building routers:
```
BigIron-BuildingB(config)# no ip directed-broadcast
BigIron-BuildingB(config)# ip icmp burst-normal 1000 burst-max 2000 lockup 60
BigIron-BuildingB(config)# write memory
```

On each of the building and DMZ switches:
```
FI_001-BuildingB(config)# ip icmp burst-normal 1000 burst-max 2000 lockup 60
FI_001-BuildingB(config)# write memory
```

Stopping TCP SYN Flood Attacks

SYN floods are another common attack that can be used as a DoS or DDoS attack. With this attack, many SYN packets using a wide range of false source IP address are sent to a victim host. As the victim host receives hundreds to thousands of these connection requests, it will build many "connection attempt" entries in its connection queue for completing the 3-way TCP handshake. The victim host will issue a SYN-ACK packet to each spoofed host and wait for the completing ACK handshake packet to return.

If the spoofed host is a legitimate device, it should realize that it never originated the SYN connection request and quickly issue a RST command to tell the victim host to drop the connection. If the spoofed IP address is not a legitimate host, the victim host will hold the uncompleted connection entry in its connection queue until the timeout value is reached and then flush the entry from its queue. As you can see, the victim host's resources can be depleted rather quickly if it is waiting for hundreds or thousands of hosts to respond – hence, the DoS attack is successful at degrading or stopping service on the victim host.

With all Foundry devices, you can prevent excessive SYN connection requests from bombarding a host by configuring the device to drop TCP SYN packets after a maximum-burst level. This command can be used to protect the router itself or it can be applied to a specific interface to protect individual hosts.

For networks that have many SYN requests, you should perform an analysis to get an idea of what your network's peak SYN load is before setting the "burst-normal" and "burst-max" values. Using a network analyzer such as a sniffer, you can measure the number of SYN requests during peak times to obtain this information. Plot your results over several days to make sure your estimates include peak and low usage times. Once you

have a good understanding of the SYN request load on your network, the command can be implemented on your Foundry devices.

Syntax: ip tcp burst-normal <value> burst-max <value> lockup <seconds>

The <value> is the number of SYN packets/second
The lockup <seconds> specifies the duration to drop excessive SYN packets

EXAMPLE:
Widget-Works.COM performed an analysis of their network over a two-week time span that included a month end financial cycle to make sure it captured peak and low usage statistics. From their findings, they have determined that their corporate switches and routers should have different normal and maximum values.

For their switches, a burst-normal value of 30 SYN packets/second is required and a maximum-burst value of 50 SYN packets/second is necessary to protect the hosts. A lockup period of 60 seconds is justified for the switches.

```
FI_001-BuildingB(config)# ip tcp burst-normal 30 burst-max 50 lockup 60
FI_001-BuildingB(config)# write memory
```

For their Building Routers, a burst-normal value of 60 SYN packets/second is required and a maximum-burst value of 100 SYN packets/second is necessary. A lockup period of 60 seconds is justified for the routers.

```
BigIron-BuildingB(config)# ip tcp burst-normal 60 burst-max 100 lockup 60
BigIron-BuildingB(config)# write memory
```

For their border router, a much lower value is required – they only have a 6 MB link to the public Internet and their WEB site and Ecommerce site generates little traffic. The analysis found that a burst-normal value of 15 SYN packets/second is sufficient and a maximum-burst value of 30 SYN packets/second should be adequate to trigger any lockup periods. A lockup period of 60 seconds is justified for the border router.

```
BRouter001(config)# ip tcp burst-normal 15 burst-max 30 lockup 60
BRouter001(config)# write memory
```

Stopping LAND Attacks

A Land Attack is a threat that uses the same source and destination IP Address in both the IP header and the same source and destination port in the protocol header. This attack is designed to cause a DoS condition on the router, or host, to either slow it down or to take it down. Using ACLs, you can prevent this attack from occurring on the router. The most obvious location to implement this defense strategy is at the border router's external interface – the external interface that connects to the ISP's network.

If the inbound Anti-Spoofing ACLs were applied on the border router's external port facing the ISP, the only network address that needs to be protected from LAND attacks is the router's gateway address that is connecting the border router to the ISP subnet. Otherwise, to protect the entire border router, all network addresses defined on the border router should be specified in the ACL - since all gateway addresses on the border router may be accessible from the public Internet.

EXAMPLE:
Using Widget-Works.COM's border router as an example, an inbound ACL would be applied on the external interface connected to the ISP network (IP Address 196.100.100.1/28). Widget-Works.COM has already applied the necessary inbound Anti-Spoofing ACLs to protect all Internal Networks from being spoofed by external hosts. Therefore, the internal interface that defines the DMZ's subnet (IP Address 198.30.15.254/24) doesn't need to be defined in the LAND Attack ACL.

Interface E1 is assigned to the 198.30.15.0/24 DMZ network and interface E16 is assigned to the 196.100.100.0/28) network.

```
BRouter001(config)# access-list 105 deny ip host 196.100.100.2 host 196.100.100.2
log
BRouter001(config)# access-list 105 permit ip any any
BRouter001(config)# interface E16
BRouter001(config-if-e100-16)# ip access-group 105 in
BRouter001(config-if-e100-16)# exit
BRouter001(config)# write memory
```

NOTE: If there are other ACLs on the same interfaces, remember to place the most specific ACLs at the top of the access list and the most general at the bottom. Otherwise, the general ACLs will be executed first and your specific rules will be skipped.

Since LAND attacks are very rare from within your Internal Network, you may decide to skip LAND attack ACLs for your internal router interfaces. If your installation is an educational institution where there are many untrusted users on your internal network, you should consider implementing this security defense on all router ports.

Disabling Proxy ARP

Proxy ARP can be enabled on each router to perform ARP requests on behalf of hosts that are not configured with a default route. Since most hosts on modern networks are configured with a proper default gateway address (usually by a DHCP server), proxy ARP is not really required. If proxy ARP is enabled, hackers can use the router's proxy ARP abilities to spoof packets and to gather information about the network and router.

By default, Proxy ARP is disabled on all Foundry routers. To make sure proxy ARP is disabled on all routers, you can manually configure the router to turn proxy ARP off.

Syntax: [no] ip proxy-arp

EXAMPLE:
Widget-Works.COM's corporate security policy requires that proxy ARP be disabled on all production routers. As a precaution, the IT team has configured every internal and external router with the **no ip proxy-arp** command. On Building B's router:

```
BigIron-BuildingB(config)# no ip proxy –arp
BigIron-BuildingB(config)# write memory
```

ARP Attack Prevention

ARP is a normal function of the TCP/IP protocol and is necessary to resolve MAC addresses from IP Addresses. ARP requests are broadcasts and if there is an abnormal amount of ARP traffic, all hosts on the broadcast domain will be affected. To protect against ARP storms or attacks that use ARP for DoS purposes, Foundry devices can limit the number of ARP requests that it can receive. For ARP Spoofing prevention, see the section titled, "Defending Against MAC Address & ARP Spoofing".

Because ARP packets are processed by the device's CPU, ARP can be used as a DoS mechanism. By configuring a maximum number of ARP requests that the Foundry device can receive per second, you can prevent or slow this type of attack from stopping services on your network devices.

NOTE: The CPU services this command and care should be taken when implementing this defense strategy. Implement this command on networks that you feel may be susceptible to ARP attacks.

Syntax: [no] rate-limit-arp <num>

The <num> parameter specifies the number of ARP packets that can be accepted per second. The range can be from 0 - 100. If you specify 0, the device will not accept any ARP packets. If you want to change a previously configured the ARP rate limiting policy, you must remove the previously configured policy using the **no rate-limit-arp** <num> command before entering the new policy.

EXAMPLE:
Widget-Works.COM has decided that the DMZ network may be susceptible to ARP attacks if an intruder can find a way into their external hosts. This example configures their border router to accept 50 ARP requests per second on the entire device. If more than 50 ARP packets are received during the one-second interval, they are dropped. The next one-second interval resets the counter.

```
BRouter001(config)# rate-limit-arp 50
BRouter001(config)# write memory
```

Stopping Hacks Using ICMP

ICMP is one of the most useful protocols for troubleshooting networks and it is used by many users and IT professionals to test connectivity between hosts. Unfortunately, hackers also use many features of ICMP to scan, fingerprint, and disrupt network services. The most important chokepoint to implement these features is on the border router. For a complete list of ICMP services that can be disabled on the border router, refer to the *Hardening Foundry Routers & Switches White Paper*.

For Internal Network security defenses, the following ICMP security features should be configured on your routers and switches to protect against possible ICMP attacks and misuse.

ICMP Redirects

ICMP redirects are used to instruct the router how to redirect traffic through the network. In essence, the packet bypasses the router's normal selected path and follows the instructions of the ICMP redirect. By allowing ICMP redirects on the Internal Network, hackers can steer traffic to a specific device for the purpose of hijacking

sessions or to monitor and record traffic flows. Since modern networks are configured to use routing protocols such as RIP or OSPF to determine the best path a packet should take, ICMP Redirects can be turned off without any impact to network functionality.

ICMP Redirects can be sent from your router's interfaces or can be received from other sources. To block outbound ICMP Redirects generated by your Foundry routers and the associated ICMP Redirect Message, use the following commands:

Syntax: [no] ip redirect (disables ICMP Redirect for a specific interface)
Syntax: [no] ip icmp redirects (disables ICMP Redirect Message)

EXAMPLE:
Widget-Works.COM's Corporate Security Policy has defined that all ICMP Redirect capabilities should be disabled on every routers' interfaces along with the associated ICMP Redirect Message. The following commands are used to configure each router's interface to stop the transmission of ICMP Redirects and its associated messages.

```
BigIron-BuildingB(config)# int e 3/11
BigIron-BuildingB(config-if-e100-3/11)# no ip redirect
BigIron-BuildingB(config-if-e100-3/11)# no ip icmp redirects
BigIron-BuildingB(config-if-e100-3/11)# exit
BigIron-BuildingB(config)# write memory
```

To block inbound ICMP Redirect packets (ICMP Type 5) from other sources, ACLs are used. The most important location to implement this security feature is on the border router. To prevent hackers from sending ICMP Redirect commands into your corporate network.

Widget-Works.COM has implemented this security feature on their border router's external interface facing the ISP's network. The ACL is programmed to block ICMP Redirects and allows all other IP traffic to pass and it is applied as an inbound ACL.

```
BRouter001(config)# access-list 101 deny icmp any any redirect
BRouter001(config)# access-list 101 permit ip any any

BRouter001(config)# interface ethernet 16
BRouter001(config-if-16)# ip access-group 101 in
```

ICMP Unreachable

ICMP Unreachable messages are used to inform the sender that the destination device is not available or a specific service is unavailable. Routers send this message to quickly inform the sender of a "no service" condition so they can move to other functions. Without this message, the sender must wait – timing out their sessions according to their local timeout settings. This can take several seconds to several minutes depending on how the local timeout settings are setup.

Unfortunately, many modern scanning tools used to probe networks also use ICMP Unreachable messages to get feedback from a probed device. When a probe is made to a device using a wide range of scanned ports, ICMP Unreachable messages are used to tell the scanning host that the port being scanned is unavailable, telling the scanner to move onto the next port to try. Turning off ICMP Unreachable messages that routers send can slow down these types of scans.

For legitimate users, turning this feature off may cause long pauses to occur while their requests are waiting to timeout. For thwarting hackers, this defense is a good approach as casual hackers often lack patience. ICMP Unreachables can be implemented on each router interface to allow you to control where to implement this feature. Foundry also allows you to block all ICMP Unreachable messages or to select specific messages to drop.

NOTE: You must evaluate the benefits of this security feature verses the downside of making legitimate users wait long periods for timeouts to occur. At a minimum, consider this feature for DMZs, screened subnets, and high security subnets.

Syntax: [no] ip icmp unreachable [network | host | protocol | administration | fragmentation-needed | port | source-route-fail]

If you enter the command without specifying a message type, every type of ICMP Unreachable message is disabled. If you want to disable only specific types of ICMP Unreachable messages, you can specify the message type. To disable more than one type of ICMP message, enter the **no ip icmp unreachable** command for each message type.

- The **network** parameter disables ICMP Network Unreachable messages.
- The **host** parameter disables ICMP Host Unreachable messages.
- The **protocol** parameter disables ICMP Protocol Unreachable messages.
- The **administration** parameter disables ICMP Unreachable (caused by Administration action) messages.
- The **fragmentation-needed** parameter disables ICMP Fragmentation-Needed But Don't-Fragment Bit Set messages.
- The **port** parameter disables ICMP Port Unreachable messages.
- The **source-route-fail** parameter disables ICMP Unreachable (caused by Source-Route-Failure) messages.

EXAMPLE:
Widget-Works.COM has determined that the border router should be protected against probing activity that use the ICMP Unreachable message. They have turned off all ICMP Unreachable messages on the external interface facing the ISP's network.

```
BRouter001(config)# interface e 16
BRouter001(config-if-16)# no ip icmp unreachable
BRouter001(config-if-16)# write memory
```

ICMP Timestamp and Information Requests

Two other ICMP types that may provide hackers with valuable information regarding your network devices are the ICMP Timestamp and Information Request packets. By implementing ACLs on your routers to filter out these ICMP request types, you can help prevent additional methods of fingerprinting your network devices. Timestamp can be used by hackers to find the time and date settings of a network device and allow them to craft packets to defeat time-dependent security defenses. Information Request can be used to find out the types of routers and hosts you have in your network.

The Timestamp Request operates on ICMP Type 13 and the Information Request operates on ICMP Type 15.

EXAMPLE:
Widget-Works.COM's Corporate Security Policy has determined that the only dangerous ICMP Timestamp and Information Requests are those from the Public Internet. The following ACL is used to block these two ICMP types at the border router's external interface facing the ISP's network.

```
BRouter001(config)# access-list 101 deny icmp any any 13
BRouter001(config)# access-list 101 deny icmp any any 15
BRouter001(config)# access-list 101 permit ip any any

BRouter001(config)# interface ethernet 16
BRouter001(config-if-16)# ip access-group 101 in
```

Stopping Foundry Devices From Responding to Broadcast ICMP Requests

By default, Foundry devices will respond to Broadcast ICMP requests. You can disable this ability on a global device level to stop Foundry devices from participating in ICMP broadcast requests.

Syntax: [no] ip icmp echo broadcast-request

EXAMPLE:
Widget-Works.COM's Corporate Security Policy has enforced this security feature on all Foundry routers and switches throughout the enterprise. On every router and switch, the ability to respond to ICMP Broadcast Requests has been disabled.

```
BRouter001(config)# no ip icmp echo broadcast-request
BRouter001(config)# write memory
```

Limiting Broadcasts

Broadcast packets are a necessity in network protocols such as TCP/IP, IPX, and Appletalk. Many popular operating systems such as Windows, as well as applications, use broadcast packets to perform specific functions to discover network resources. The downside with broadcast technology is the potential harm to network performance and connectivity if excessive packets are transmitted on a network. Since every device on the subnet must listen to each broadcast packet, it can be used as a form of DoS attack. Misconfigured devices or applications can inadvertently cause problems with excessive broadcasts; they don't have to be from an attacker.

Foundry devices include a unique feature to allow the limiting of broadcast packets that a port can forward onto the network. Since all broadcasts are examined by the CPU, this is a software feature and should be used with care to preserve network performance. For networks with very high levels of normal broadcast traffic, this feature may decrease performance - as normally required broadcasts are limited. This feature should be tested before it is implemented.

Syntax: broadcast limit <num>

Possible values: 0 - 4294967295; if you specify 0, limiting is disabled.

EXAMPLE:
Widget-Works.COM has analyzed their network over several days and calculated the peak broadcast usage. They have determined that this feature should be used in the Perimeter Network to protect the DMZ from broadcast storms and misconfigured or malicious applications.

On each of the interfaces where there is a potential for the initiation of broadcast storms the following broadcast limit command was applied.

```
BRouter001(config-if-1)# broadcast limit 100
BRouter001(config-if-1)# write memory
```

Preventing UPD Broadcasts or All Broadcasts

Broadcast traffic, whether it's TCP or UDP broadcasts, can be used as a mechanism for Denial of Service attacks. Many times, broadcast traffic is accidentally injected onto the network by a misconfigured host or a malfunctioning device. By flooding the network with TCP or UDP broadcast storms, valuable bandwidth is taken away from legitimate services. Foundry's IronShield Security provides the **broadcast filter** command to allow network administrators to control which ports can or cannot send broadcast traffic.

This command is very helpful in stopping broadcast packets from being sent (without shutting down the entire port) during a broadcast storm. It can also be enabled for ports that are not supposed to send any broadcast traffic or UDP broadcast traffic. Depending on the operating system and the applications being used on the device, enabling this feature may break applications that require broadcast traffic. Care must be taken before enabling this feature in non-emergency situations - compatibility testing should be performed beforehand.

Syntax: [no] broadcast filter <filter-id> any | ip udp [vlan <vlan-id>]

The <filter-id> specifies the filter number and can a number from 1 - 8. The software applies the filters in ascending numerical order. As soon as a match is found, the software takes the action specified by the filter (block the broadcast) and does not compare the packet against additional broadcast filters.

You can specify **any** or **ip udp** as the type of broadcast traffic to filter. The **any** parameter prevents all broadcast traffic from being sent on the specified ports. The **ip udp** parameter prevents all IP UDP broadcasts from being sent on the specified ports but allows other types of broadcast traffic.

If you specify a port-based VLAN ID, the filter applies only to the broadcast domain of the specified VLAN, not to all broadcast domains (VLANs) on the device.

As soon as you press Enter after entering the command, the CLI changes to the configuration level for the filter you are configuring. You specify the ports to which the filter applies at the filter's configuration level.

Syntax: [no] exclude-ports ethernet <portnum> to <portnum>

Or

Syntax: [no] exclude-ports ethernet <portnum> ethernet <portnum> ....

EXAMPLE:
Widget-Works.COM has a misconfigured server that is causing a very high volume of broadcast traffic to flood the development network. Using sFlow's network-wide monitoring capabilities, the NOC has quickly determined that the broadcast storm is coming from a large UNIX server in Building A's development server farm. Many programmers are currently using this server and shutting the port down to investigate the problem will cause major outages for the company.

The offending host is connected to Building A's first FastIron switch port 3/16. The NOC will use the Broadcast Filter command to turn off all broadcasts and allow all other traffic to pass through the port while they investigate the problem. Only port 3/16 will be excluded from broadcast traffic.

```
FI_001-BuildingA(config)# broadcast filter 1 any
FI_001-BuildingA(config-bcast-filter-id-1)# exclude-ports ethernet 3/16
FI_001-BuildingA(config-bcast-filter-id-1)# write memory
```

Fragmentation Attack Prevention

Fragmentation occurs naturally in TCP/IP communications when the MTU sizes between the sending and receiving hosts do not match. But hackers have used fragmentation as a means to bypass firewalls and IDS systems that do not support packet reassembly or perform it improperly.

How Fragmentation Works

Fragmentation occurs when the end-to-end MTU size is smaller than the packet being transmitted. ICMP is used to negotiate the MTU size between the hosts before data transfer begins and the sender typically uses the smallest MTU size for the path to avoid fragmentation – but this is not always possible and thus fragmentation occurs.

This example will illustrate the need for fragmentation. Sender A has a file it needs to send to host Receiver B and its normal datagram size is 4028 bytes. The negotiated maximum MTU size of all the routers in the path between host A and B is only 1500 bytes. Sender A will have to fragment the original packet into three separate packets to successfully transmit the information. Receiver B will be responsible for packet reassembly using the Fragment ID Number and Fragment Sequence Number. All fragments of the same packet must contain the following in the IP header:

- All fragments of the same original datagram must have the same Fragment ID Number.
- Each fragment must carry a Sequence Number that identifies its order or offset in the original packet.
- Each fragment must identify the length of the data carried in the fragment.
- Each fragment must inform the device if more fragments are coming after it using the More Fragment flag.
- The IP header is cloned from the first packet to every fragmented packet.
- The ***protocol header is not cloned*** to the fragmented packets and is only carried on the first packet.

How Hackers Use Fragmentation

By understanding fragmentation, you will be in a position to spot fragmentation attacks. Routers often do not look at every fragment in a fragmentation chain. Usually, it's only the first packet that is analyzed and depending on the rules of how to pass the packet, it is either passed or denied. Hackers rely on the fact that only the first

packet in the fragmentation chain contains any protocol headers: TCP, UDP, ICMP and all subsequent fragment packets only contain the IP header with the source and destination IP addresses.

By carefully crafting the first packet of the fragmentation chain to pass through the device's policy, all subsequent fragmented packets will also pass because the device has already made its decision based on the information provided in the first fragment (assuming the device isn't performing packet reassembly). By placing malicious code in the subsequent fragments, they can now bypass the router, firewall, or IDS system. Some of the most famous attacks that used fragmentation were the "Ping of Death" and "Teardrop".

Why don't devices such as routers analyze every fragmented packet? It is extremely resource intensive to keep track of each fragment's state – requiring each packet to be examined and stored before the decision can be made to either allow or drop the packet. Most routers make the assumption that if one of the fragments is missing or corrupted in the fragmentation chain, the entire packet will be resent. The router forwards all subsequent packets based on the first fragment to speed up transmission and maintain network performance.

How Foundry Treats Fragmentation

By default, Foundry devices will analyze the first fragment of a fragmentation chain and apply any ACLs to either permit or deny the packet. All subsequent fragments will either be passed or dropped in hardware (JetCore products) based on the decision made from analyzing the first packet. Since most normal applications cannot use the packet if one of the pieces/fragments is missing or bad, this method works well for many applications.

On JetCore products, you can enable fragmentation features that may help secure your environment against fragmentation attacks.

NOTE: Care must be taken to ensure that fragmented packets are not a frequent and normal part of your traffic flows. Use a protocol analyzer to review your traffic for fragmentation over a period of several days before using this feature.

NOTE: The fragmentation support described in this section applies only to JetCore devices and only to hardware-based ACLs. These commands are CPU based so care must be taken when using these features. Applications that use fragmentation frequently, such as NFS and some Microsoft management protocols, may cause performance to degrade if the amount of fragmented traffic is excessive.

CPU Inspection of Fragmented Packets

This command allows you to forward all fragmented packets to the CPU for inspection against the ACLs or to drop all fragments.

Syntax: [no] ip access-group frag inspect | deny

The **inspect | deny** parameter specifies whether the fragments should be sent to the CPU for inspection or dropped:

inspect - This option sends all fragments to the CPU for inspection to see if the fragment should be passed or dropped according to any ACLs defined.

deny - This option begins dropping all fragments received by the port as soon as you enter the command. This option is especially useful if the port is receiving an unusually high rate of fragments that are not part of your normal traffic - which can indicate a fragmentation attack.

EXAMPLE:
Because fragmentation attacks are most likely initiated from the Public Internet, Widget-Works.COM has decided to implement this security feature on their border router's external port facing the Public Internet.

```
BRouter001(config)# interface ethernet 16
BRouter001(config-if-16)# ip access-group frag inspect
BRouter001(config-if-16)# write memory
```

Controlling the Fragment Rate

If the device is configured to forward all fragmented packets to the CPU for inspection, there will be some additional CPU processing load. Normal fragmentation on most networks will not cause a heavy load on the CPU, but misconfigured hosts or hackers using fragmentation may introduce large amounts of fragmented packets and add a significant load on the CPU. The following command allows you to control the maximum number of fragments the device or interface can send to the CPU in a one-second interval.

To Set Fragmentation Rate On Entire Chassis

This command allows you to specify the maximum number of fragments that can be sent to the CPU for the entire chassis. Once the maximum level of fragments/second has been reached, the exceed-action is taken and an interval timer is started. The system will take the action specified for the number of minutes specified in the reset-interval.

Syntax: [no] ip access-list frag-rate-on-system <num> exceed-action drop | forward reset-interval <mins>

The **<num>** parameter specifies the maximum number of fragments the device or an individual interface can receive and send to the CPU in a one-second interval. The valid range is 600 – 12800 and the default is 6400.

The **drop | forward** parameter specifies the action to take if the threshold (<num> parameter) is exceeded:

- **drop** - fragments are dropped without filtering by the ACLs
- **forward** - fragments are forwarded in hardware without filtering by the ACLs

The **<mins>** parameter specifies the number of minutes the device will enforce the drop or forward action after a threshold has been exceeded. You can specify from 1 - 30 minutes.

EXAMPLE:
To protect the border router's CPU from fragmentation attacks, Widget-Works.COM has decided to limit the number of fragmented packets sent to the CPU for inspection. This example configures the border router to accept a maximum 6000 fragments/second for the entire chassis. If the maximum is reached, the action is to drop all subsequent fragments for duration of 2 minutes.

```
BRouter001(config)# ip access-list frag-rate-on-system 6000 drop reset-int 2
BRouter001(config)# write memory
```

To Set Fragmentation Rate On Interfaces
This command allows you to specify the maximum number of fragments that can be forwarded to the CPU from an individual interface. Once the maximum fragments/second has been reached, the exceed-action is taken and an interval timer is started. The system will take the action specified for the number of minutes specified in the reset-interval.

Syntax: [no] ip access-list frag-rate-on-interface <num> exceed-action drop | forward reset-interval <mins>

The **<num>** parameter specifies the maximum number of fragments the device or an individual interface can receive and send to the CPU in a one-second interval. The valid range is 300 – 8000 and the default is 4000.

The **drop | forward** parameter specifies the action to take if the threshold (<num> parameter) is exceeded:

- **drop** - fragments are dropped without filtering by the ACLs
- **forward** - fragments are forwarded in hardware without filtering by the ACLs

The **<mins>** parameter specifies the number of minutes the device will enforce the drop or forward action after a threshold has been exceeded. You can specify from 1 - 30 minutes.

EXAMPLE:
This example configures the JetCore device for a maximum of 1000 fragments/second for all interfaces on the chassis. If the maximum is reached, the action is to drop all subsequent fragments for duration of 2 minutes.

```
BigIron(config)# ip access-list frag-rate-on-interface 1000 drop reset-int 2
BigIron(config)# write memory
```

Dropping All Fragments For IronCore Products

IronCore products can be configured to drop all fragmented packets when the IP header information (source and destination address) matches the ACL - regardless if the accompanying ACL is designed to permit the packet. This command is used if you do NOT want to pass any fragments at all. Since all fragmented packets (besides the first one) have no Protocol Header information (source port, destination port, etc) to compare against the ACL, only the IP header information is used. This can pose a security risk if crafted packets are designed to bypass fragmentation algorithms.

Syntax: [no] ip access-group frag deny

NOTE: This command is applied on an interface level only and was added in release 07.5.04A and later revisions of IronCore software.

EXAMPLE:
This example configures the IronCore device to drop all fragmented packets that are forwarded to the CPU.

```
BigIron(config)# interface E 1/1
BigIron(config-if-1/1)# ip access-group frag deny
BigIron(config-if-1/1)# write memory
```

Containment Design

Trying to stop all possibilities of DoS attacks on the Internal Network is an impossible task. The number of attacks is growing and the sophistication is increasing with worm technology – eluding many traditional security devices. To combat DoS attacks, you must look at your defense strategy from two sides: prevention and containment. Implementing security devices and IronShield Security features like the ones discussed in this section can achieve stronger levels of prevention, as well as notification. Containment is accomplished through careful network designed and access restrictions.

In order to limit the damage caused by a DoS attack or the spreading of a malicious worm or virus, the Internal Network must be designed to limit the access of the infected hosts. With this defense strategy, you are protecting your most critical network resources by limiting access to only authorized personnel or hosts. By decreasing the ability of every host being able to navigate the entire corporate network, you are limiting your exposure and containing the amount of damage a malicious application or person can inflict.

What To Protect

In order to design a secure network, you must first understand what needs protecting and from whom it needs protection. A thorough security evaluation of your network, information resources, departmental usage patterns, physical security, customer access, etc, will need to be performed by your security and IT departments. The following will need to be identified:

- Determine the type of resources your company uses and categorize them into groups to determine the level of protection they will need.
- Determine the level of exposure for each group of resources and the risk of not protecting them.
- Determine the business requirements for each resource group.

With the information from the security evaluation, make sure your Corporate Security Policy is updated to reflect the findings. Your security design and defense strategies must parallel your Corporate Security Policy – a good security policy has the following components:

- It must be enforceable. Management and Human Resources have reviewed and signed off on the policy.
- It must match the corporate culture; otherwise, it will be ignored. Review the employee handbook and any corporate directives. Research the unwritten directives and policies.
- Compliance to the policy must be measurable.
- It must have authority, scope, and expiration defined.
- It must be well written – specific, concise, and clear in its message.
- It must be realistic in terms of implementation, compliance, and enforcement.
- It must address privacy and the rights to privacy.
- It must address the use of corporate resources for work and personal use.
- It must address how incidents are to be handled and enforced.

NOTE: Templates that can help you build your Corporate Security Policy can be found at the SANS web site: www.sans.org/newlook/resources/policies/policies.htm

---

As your security policies and security defenses are being designed, remember that networks are created to provide service. To service users and customers, the corporate network must be accessible, reliable, easy to use, and provide the performance needed to conduct business quickly and efficiently. Finding the right balance between usability, acceptability, and security is the goal. The hallmark of a well-designed network will have the right mix of all three.

Restricting Access & Containment

Restricting access to authorized users is one of the most effective ways of enhancing your Internal Network security. Since many businesses treat the Internal Network as a fully trusted network, security defenses may not be implemented. All of their internal resources are accessible from all areas on the internal LAN. In these installations, directory and file rights using native OS or directory service authentication is protecting the corporate resources – using some form of username and password authentication with file and directory restrictions.

Regardless of the directory and file permissions, the ability to access the entire corporate network using protocols such as TCP/IP is still largely present in many installations. Sophisticated intruders and hackers only need connectivity to your network to begin their surveillance and attacks. By carefully designing your network scope for server resources and departmental users, you can begin to layer security defenses through network access restriction and containment. By limiting network access to critical resources, you can begin to insolate your mission critical resources and grant only authorized users network access using network Layer 2 and Layer 3 security features.

Coupled with OS or Directory Service authentication and existing directory and file access rights, network restriction and containment can add a tough layer to your Internal Network defenses. Access restriction and containment, performed by Layer 2 and Layer 3 security features, includes the use of:

- Standard Access Control Lists (ACLs)
- Extended Access Control Lists (ACLs)
- Policy Based Routing (ACLs)
- VLANs

With the sophistication of modern hacking and intrusion code, the intruder no longer has to be physically present in your buildings to breach your internal hosts. Social engineering and gaining physical access to networks is still a popular method for intruders, but backdoor trojans and other malicious code have made it easier to breach internal hosts from outside the network. Many security studies have shown that employees have been a major source of internal threats: accidental deletions, intentional sabotage, and disgruntled employees contribute a large percentage for the destruction and theft of information.

Containment is the key to limiting the damage caused by these activities. If the intruder, whether internal or external, is limited to where he can navigate on your network, the chances of spreading malware, accidental or intentional destruction of information, and data theft can be lowered.

NOTE: Protecting network devices, infrastructure servers, critical application servers, etc. with adequate physical security is absolutely necessary with any good security design. Your Corporate Security Policy must provide adequate guidance on how to setup and protect each critical network device and host. For more information on what to include in your Physical Security Design, refer to Appendix B.

If you were designing your network from the very beginning and a large IP address range was readily available, the job of creating a secure network would be much simpler. However, many installations do not have the luxury of designing an entire network from the beginning. The network may have grown in an unorganized fashion as new employees were hired or physical spaces was added or removed – creating situations such as:

- File servers are located in various subnets. They are not grouped together into server farms and are mixed in subnets with all other end user devices.
- Users from each department are not grouped together. They sit in various locations of the company in different subnets.
- Information is not well organized on the file servers. As the company grew, information from one department may be placed on another departments server, etc.
- User and group access rights do not fully reflect the information they protect. Users in groups cannot be confirmed. IT doesn't know if the users actually need the access rights to the directories and files.

Security Zones

Performing an in depth analysis of all data resources and how they are laid out will reveal the type of data resources available, their security requirements, and the people who need access to them. The findings will give you the information to start redesigning your network, file servers, and access rights to better build security defenses. The goal is to group resources based on security requirements and departmental similarities – to create definable ***security zones***.

With any redesign of the network for the purposes of layering additional security defenses, several guidelines should be followed – this will simplify the task and identify where to start the redesign process.

- Identify the groups of resources that will require the most security. Identify each group of users that will need network access to these high security resource groups. These will become your *security zones*.
- Sort the security zones, placing those with the highest security requirements at the top of the list.
- Within each security zone, identify the resources with the least number of required users. This will be your starting points.
- Critical high security resources should be placed on a single host. Do not mix information resources on the same host that do not have the same security requirements and user access requirements.
- Use multiple subnets to house each security zone to give your design more flexibility in implementing access control.
- Design your network addressing scheme to match your needs. Plan for enough growth, but do not use overly large address spaces. Attacks using broadcast packets will impact subnets with larger address ranges more than subnets with smaller address ranges. Larger subnets cannot be secured as granularly as smaller subnets.
- Layer your security starting with the most general access lists. Standard ACLs will protect the resources by limiting the subnets and hosts.
- Layer the most general access lists starting from the backbone or network core, most likely at the ingress points to each routed subnet.
- Layer more specific access lists closer to the hosts they protect. If the subnets are well organized, this may be at the switch uplink ports.
- Layer the most specific access lists at each interface connecting the host. Extended ACLs can be used to restrict access based on source/destination IP address (Layer 3) and source/destination port number (Layer 4).
- Separate guest or visitor access from your corporate data network. These include conference rooms, lobby access, cafeteria access, training rooms, and so forth.

- Isolate your wireless networks from the wired data network.
- Use network port authentication (802.1x) and port security features in mixed environments where legitimate user access is hard to enforce.
- Remember that not all resources need to be protected to the same degree. Resource isolation onto dedicated servers and simple access restriction ACLs can be a good effective start in many cases.

Once you have identified all of your resource groups and have sorted them from smallest to largest and from the most simple to the most difficult, you are ready to design and implement your security zones. Traditional firewalls and IDS sensors can be used at various ingress points to protect high security subnets. The difficulty with using firewalls and IDS sensors to protect all security zones in the Internal Network may include:

- Performance is not maintained. With single Gigabit, trunked-Gigabit, and 10 Gigabit uplinks connecting subnets to routers, stateful firewalls may not be able to keep up with performance demands.
- IDS sensors traditionally run off mirrored or monitored ports. With high-capacity switches and routers, the number of mirrored ports is limited to support the required IDS sensors. The performance level of the IDS may not support the high-performance Gigabit and multi-Gigabit uplinks.
- The cost of implementing firewalls and IDS sensors throughout the entire company can be expensive.
- The IT resources for managing and monitoring the additional firewalls and IDS sensors may not be available.

Foundry's IronShield Security solution offers several security zone defense features to enhance Internal Network security. These features are built into every Foundry Layer 2 – 7 device and include high-performance ACLs, Policy Based Routing, and VLANs. By layering these defenses, the Internal Network can be secured using wire-speed, low-cost features that can be used to enhance or be alternatives to traditional firewalls and IDS sensors.

Redesign Tips

There are many ways to redesign your network to begin the implementation of security zones. Some will be easier than others depending on how the existing network is laid out and how the existing IP addressing schemes are being used. The capabilities of existing routers and switches, router and switch capacity, existing cable plant, and IP address space will usually dictate the redesign approach.

The following methods can be used to help redesign the network to support security zones:

- If the existing router resources and address space permits, create new subnets for each security zone. Update DNS servers, DHCP servers, firewall rules, etc before transferring servers and workstations into each security zone. Perform a parallel conversion if possible.
- If address space is not limited, create subnets that parallel the security zone's size. Plan for adequate growth of each subnet, but do not use overly large subnets. For example, use several Class C subnets instead of a Class B subnet. Keep broadcast and ARP traffic in mind.
- If address space is limited, consider using a new private address space to perform the conversion into new security zones. Keep like functions in contiguous network address spaces and leave enough room to add new subnets for growth.
- If router ports are limited, consider multi-netting new subnets onto existing router ports to help organize address space and new security zones. After the transfers are complete, remove the old subnet from the router ports and keep the new addressing scheme.
- If router ports are limited and the necessary number of security zones to house individual server farms are not available, house servers with similar security requirements into the same security zone and use more specific ACLs at the host level to protect the most critical resources.

- If budgets permit, upgrade existing routers or switches to add extra capacity to create the necessary security zones.
- If switch ports are limited, use VLANs to group resources into their respective security zones. For high security installations, dedicated switches are always desired over the use of VLANs. If VLANs are used, port-based VLANs are the most secure.
- When migrating data on file servers to group like functions, you may need to purchase an additional server to start the migration process. Starting with the largest server, move the necessary information to the new server and other existing servers to free up the old server. Using the old server, migrate the next largest server and so forth. The oldest server with the least capacity will drop out of the production list when the migration is completed.
- Consolidate general access file servers and its information away from specific departmental servers and application servers. Place general access file servers into separate security zones away from high security resources.

Protecting Resources With ACLs

Once resources have been sorted into security zones, Access Control Lists can be used to provide a cost effective defense strategy to secure corporate resources. ACLs can be implemented in layers to govern access and block the spread of malicious traffic. There are two types of ACLs: standard and extended.

Standard ACLs

Standard ACLs only use IP address information to control access and does not use any protocol information such as IP, TCP, UDP, or ICMP and their respective port numbers. Standard ACLs can be created using a number identifier ranging from 1 – 99 or a name identifier using a text string. Standard ACLs are extremely valuable in security designs as they are used to govern general access based on subnet or host IP addresses.

Standard ACL Syntax

Syntax: [no] access-list <num> deny | permit <source-ip> | <hostname> <wildcard> [log]
or
Syntax: [no] access-list <num> deny | permit <source-ip>/<mask-bits> | <hostname> [log]
Syntax: [no] access-list <num> deny | permit host <source-ip> | <hostname> [log]
Syntax: [no] access-list <num> deny | permit any [log]

Syntax: [no] ip access-group <num> in | out

The <num> parameter is the access list number and can be from 1 - 99.

The **deny | permit** parameter indicates whether packets that match a policy in the access list are denied (dropped) or permitted (forwarded).

The <source-ip> parameter specifies the source IP address. Alternatively, you can specify the host name.

NOTE: To specify the host name instead of the IP address, the host name must be configured using the Foundry device's DNS resolver. To configure the DNS resolver name, use the **ip dns server-address...** command at the global CONFIG level of the CLI.

Extended ACLs

Extended ACLs govern access using IP Address information and IP Protocol information such as IP, TCP, UDP, and ICMP. Extended ACLs can be created with a number identifier ranging from 100 – 199 or a name identifier using a text string. With the additional protocol screening capabilities, extended ACLs can be used in security defenses that require more granular protection - guarding specific hosts and applications.

Extended ACL Syntax

Syntax: access-list <num> deny | permit <ip-protocol> <source-ip> | <hostname> <wildcard> [<operator> <source-tcp/udp-port>] <destination-ip> | <hostname> [<icmp-type>] <wildcard> [<operator> <destination-tcp/udp-port>] [precedence <name> | <num>] [tos <name> | <num>] [log]

Syntax: [no] access-list <num> deny | permit host <ip-protocol> any any [log]

Syntax: [no] ip access-group <num> in | out

The <num> parameter indicates the ACL number and be from 100 - 199 for an extended ACL.

The **deny | permit** parameter indicates whether packets that match the policy are dropped or forwarded.

The <ip-protocol> parameter indicates the type of IP packet you are filtering. In release 07.6.01 and later, you can specify a well-known name for any protocol whose number is less than 255. For other protocols, you must enter the number. Enter "?" instead of a protocol to list the well-known names recognized by the CLI.

The <source-ip> | <hostname> parameter specifies the source IP host for the policy. If you want the policy to match on all source addresses, enter **any**.

The <wildcard> parameter specifies the portion of the source IP host address to match against. The <wildcard> is a four-part value in dotted-decimal notation (IP address format) consisting of ones and zeros. Zeros in the mask mean the packet's source address must match the <source-ip>. Ones mean any value matches. For example, the <source-ip> and <wildcard> values 209.157.22.26 0.0.0.255 mean that all hosts in the Class C sub-net 209.157.22.x match the policy. The wildcard mask can also be specified in CIDR format with a forward slash and mask value.

The <destination-ip> | <hostname> parameter specifies the destination IP host for the policy. If you want the policy to match on all destination addresses, enter **any**.

The <icmp-type> parameter specifies the ICMP protocol type. If no <icmp-type> is supplied, all ICMP types are assumed. The following are valid <icmp-types>:

- echo
- echo-reply
- information-request
- mask-reply
- mask-request
- parameter-problem
- redirect
- source-quench
- time-exceeded

- timestamp-reply
- timestamp-request
- unreachable
- <num>

The <operator> parameter specifies a comparison operator for the TCP or UDP port number. This parameter applies only when you specify **tcp** or **udp** as the IP protocol. For example, if you are configuring an entry for HTTP, specify **tcp eq http**. You can enter one of the following operators:

- **eq** - The policy applies to the TCP or UDP port name or number you enter after **eq**.
- **gt** - The policy applies to TCP or UDP port numbers greater than the port number or the numeric equivalent of the port name you enter after **gt**.
- **lt** - The policy applies to TCP or UDP port numbers that are less than the port number or the numeric equivalent of the port name you enter after **lt**.
- **neq** - The policy applies to all TCP or UDP port numbers except the port number or port name you enter after **neq**.
- **range** - The policy applies to all TCP or UDP port numbers that are between the first TCP or UDP port name or number and the second one you enter following the **range** parameter. The range includes the port names or numbers you enter. For example, to apply the policy to all ports between and including 23 (Telnet) and 53 (DNS), enter the following: **range 23 53**. The first port number in the range must be lower than the last number in the range.
- **established** - This operator applies only to TCP packets. If you use this operator, the policy applies to TCP packets that have the ACK (Acknowledgment) or RST (Reset) bits set on (set to "1") in the Control Bits field of the TCP packet header. Thus, the policy applies only to established TCP sessions, not to new sessions. See Section 3.1, "Header Format", in RFC 793 for information about this field. This operator applies only to destination TCP ports, not source TCP ports.

The <tcp/udp-port> parameter specifies the TCP or UDP port number or well-known name. In release 07.6.01 and later, you can specify a well-known name for any application port whose number is less than 1024. For other application ports, you must enter the number. Enter "?" instead of a port to list the well-known names recognized by the CLI.

The **in | out** parameter specifies whether the ACL applies to incoming traffic or outgoing traffic on the interface to which you apply the ACL. You can apply the ACL to an Ethernet port, POS port, or virtual interface.

General ACL Principles

One of the major drawbacks in using ACLs to secure network access is the management aspect. In medium to large installations, the list of ACLs can be very long and difficult to manage. Foundry's Ironview Network Manager was designed to aid in this respect and the task of provisioning and managing ACLs can be greatly simplified.

When using ACLs to layer security, there are several guidelines that can help order the ACLs to limit the number of access lists used in the overall design and to keep ACL maintenance to a minimum. Keep the following design goals in mind when implementing ACLs for security defenses:

- Layer your security starting with the most general, least restricting ACLs at the backbone or core. Standard ACLs can be used at the router interface level to protect security zone subnets by limiting access – granting authorized users and subnets access, blocking non-authorized devices and associated traffic. Once these general ACLs are implemented and fine tuned, they should not require frequent modifications.
- Layer access lists that are more specific closer to the hosts they're designed to protect. With the ability to control access based on source/destination IP address (Layer 3) and source/destination port number (Layer 4), extended ACLs are more suited to protect individual hosts and applications.
- Don't over use ACLs. They are not meant to replace operating system file and directory rights - but to enhance them by limiting general access from all parts of your network. Secure your most critical information resources with ACLs and leave your general access and least critical resources to your existing security scheme. Don't try to force ACL security if it doesn't fit into the overall security solution as it may create more maintenance in the end.
- Make sure the security zones for each group of servers and users are properly defined and membership in each group is accurate before implementing ACLs to restrict access and contain resources. Otherwise, accessibility and usability will be lost.
- You must also consider any infrastructure devices that are required to support the resources in each security zone. Access to DNS, NIS, Active Directory, Domain Controllers, Print Servers, etc. must be maintained to ensure functionality.

As mentioned in the "Foundry Access Control Lists (ACLs)" section, remember the important guidelines for Flow Based ACLs verses Hardware Based ACLs and how the CPU is involved with each access list type. The most important points to keep in mind are repeated here. The goal is to maintain performance while adding security.

- Inbound ACLs are more efficient that outbound ACLs.
- Put the most specific rules at the top of the ACL list and the most general towards the bottom.
- Once a packet satisfies an ACL rule, it is executed by that ACL and stops flowing down the ACL list.
- Foundry uses an "implied deny" for their ACLs which means the last action is always to deny.
- Enabling the log feature uses the CPU. Only log traffic when it's absolutely necessary.
- For security chokepoints that require high bandwidth, use JetCore products to implement hardware Rule Based ACLs.
- Load balance your access list policies throughout the enterprise on both Layer 2 and Layer 3 devices to distribute security loads, push security to the edge layers where servers and workstations reside. This will block and contain malicious traffic before it enters the subnet or backbone.
- If changes are made to existing ACLs, they must be rebound to take effect.

NOTE: Refer to the "Foundry Access Control Lists (ACLs)" section in this White Paper for more information on Flow Based ACLs vs. Hardware Based ACLs and how CPU is used with the ACL features. For a complete explanation of how to implement Foundry ACLs, refer to the *"Foundry Enterprise Configuration and Management Guide"*

Inbound ACLs vs. Outbound ACLs

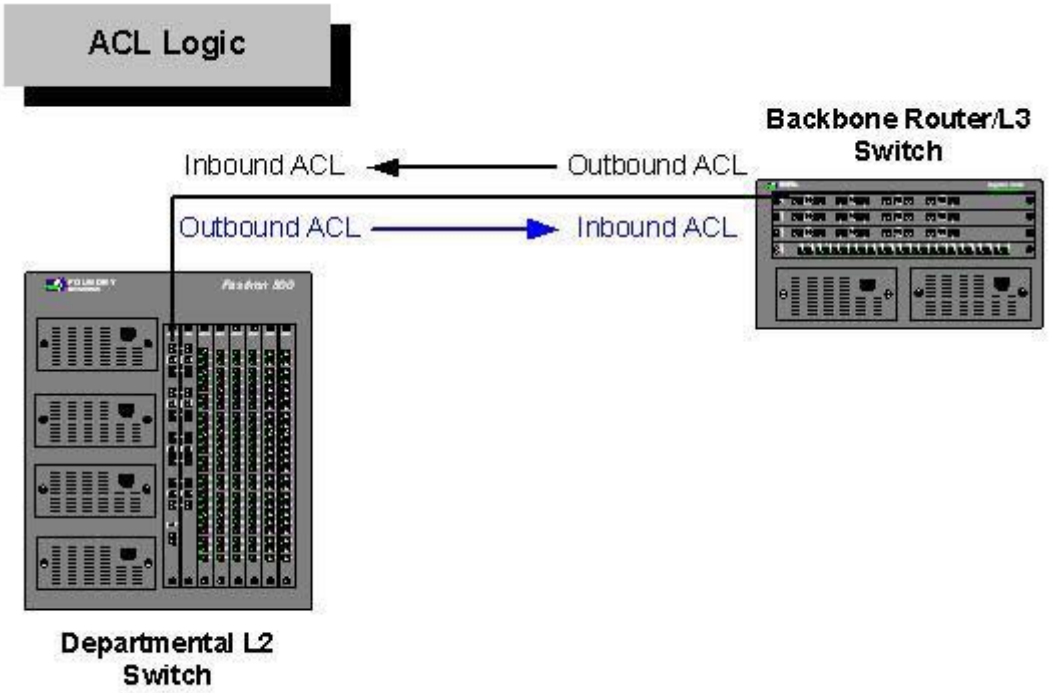


Figure 2. ACL Logic

When designing ACLs, always try to use inbound ACLs instead of outbound ACLs as they are more efficient in terms of CPU usage. Taking a look at the above diagram, an outbound ACL on one device can be reversed and implemented as an inbound ACL on the connecting device. Since Foundry devices support ACLs on their Layer 2 and Layer 3 switches, inbound ACLs can be implemented in all areas of the network – to achieve both security and performance.

Using only inbound ACLs, traffic **entering the Layer 2 switch** can be controlled with ACLs applied to the switch's uplink port (connecting to the backbone router). Traffic **leaving the Layer 2 switch** can be controlled with an inbound ACL on the router interface that connects to the switch.

Security Defense Example Using ACLs

Widget-Works.COM's network will be used throughout this section to illustrate the principles of using Standard and Extended ACLs to protect and contain security zones.

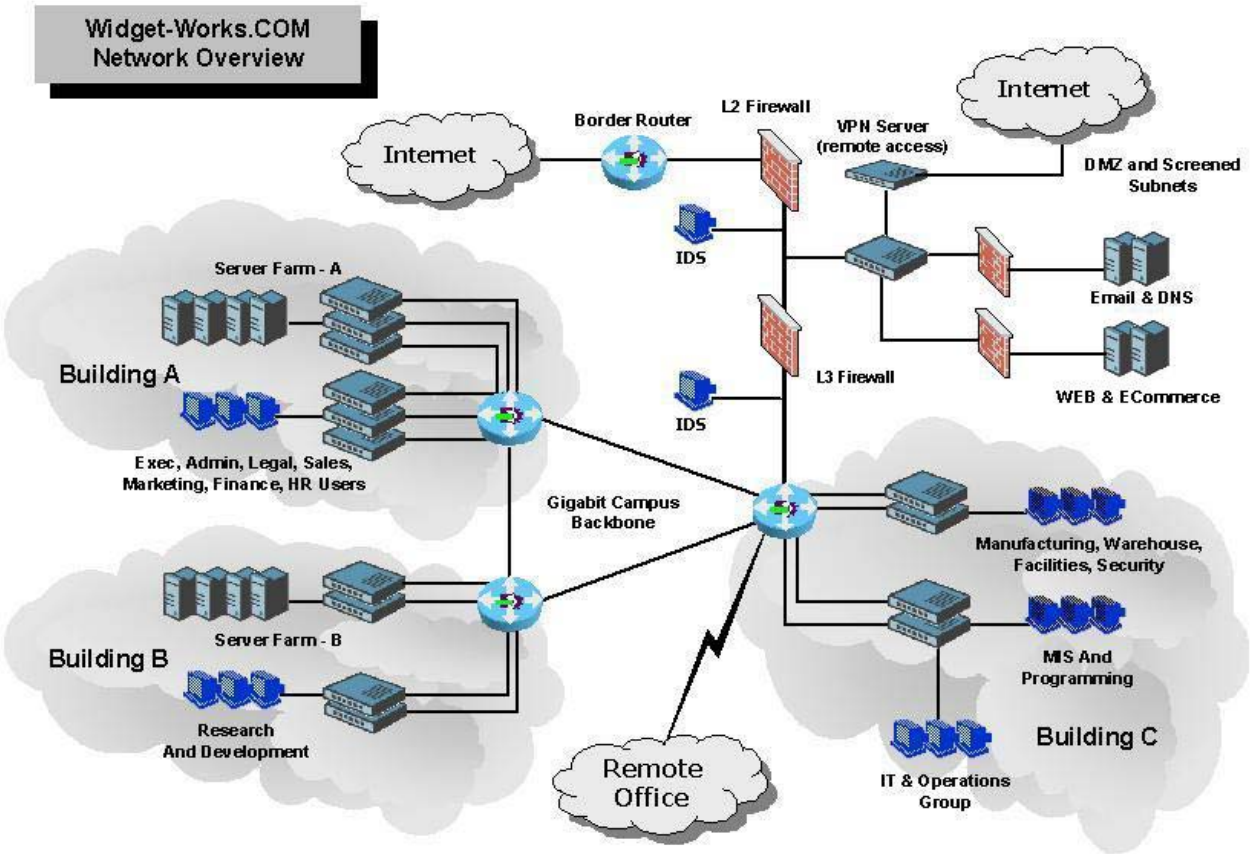


Figure 3. Widget-Works.COM Network Diagram

Widget-Works.COM's Security Zones

From the Widget-Works.COM's network diagram, we can see that the company has developed a structured approach for all of their sever and user subnets. They are using a large private address space to allow them the flexibility needed to create an individual security zone for each server and user subnet. The following subnets are being used:

Building A

Server Security Zones	**Subnet**
Executive/Admin Servers	10.32.1.0/24
Finance & HR Servers	10.32.2.0/24
Legal Servers	10.32.3.0/24
MIS Development Servers	10.32.4.0/24
Corporate General Storage Servers	10.32.5.0/24

Infrastructure Servers (email, calendar, etc)	10.32.6.0/24

Users Security Zones	**Subnet**
Executive/Admin Users	10.33.1.0/24
Finance Users	10.33.2.0/24
Sales & Marketing Users	10.33.3.0/24
Legal Users	10.33.4.0/24
HR Users	10.33.5.0/24

Building B

Server Security Zones	**Subnet**
Research and Development Servers	10.48.1.0/24
Staging and Testing Network	10.48.2.0/24

Users Security Zones	**Subnet**
Research and Development Users	10.49.1.0/24
Developers and Quality Assurance	10.49.2.0/24

Building C

Users Security Zones	**Subnet**
MIS Programming Users	10.64.1.0/24
Manufacturing And Warehousing	10.64.2.0/24
IT Users	10.64.3.0/24
NOC Operations	10.64.4.0/24
Facilities and Security Users	10.64.5.0/24

WAN Security Zone	**Subnet**
WAN Subnet	10.200.100.0/24

VPN & Guest Security Zones	**Subnet**
VPN Address Pool	198.30.15.10/24 thru 198.30.15.60/24
Guest Network	198.30.15.61 thru 198.31.15.80

Remote Office

Users Security Zones	**Subnet**
Sales and Training users	10.96.1.0/24
Training Room	192.168.1.0/24

Other Information

Infrastructure Hosts	**IP Address**
Building A Domain Controller	10.33.4.250/32
Building B Domain Controller	10.49.1.250/32
Building C Domain Controller	10.64.1.250/32
Remote Office Domain Controller	10.96.1.250/32

Building A Print Server	10.33.1.240/32
Building B Print Server	10.49.1.240/32
Building C Print Server	10.64.1.240/32
Remote Office File & Print Server	10.96.1.240/32

The Gigabit Core runs OSPF and each building is configured with a redundant path to another neighboring building on the campus.

Backbone Network Information	**Subnet**
Building A – Building B	10.200.1.0/24
Building A – Building C	10.200.2.0/24
Building B – Building C	10.200.3.0/24

According to their Corporate Security Policy, Widget-Works.COM will be implementing the following network access and containment policies:

- Only users requiring access to their respective departmental servers are to be given access to those servers.
- Server subnets should not have access to the Public Internet to protect them from trojans, DoS attacks, malicious software, and worms. All patches and necessary software is downloaded by the support team and transferred to the servers for installation.
- IT and Operations Groups must have access to these security zones to manage and monitor the resources.
- All Windows domain controllers and print servers are to be accessible from all user subnets. Domain Controllers must be available to each server subnet.
- Any guest or visitor subnet are to have access only to the necessary DNS servers, using UDP 53, and the Public Internet. A static IP address will be assigned to the guest computer in order to track usage from each common access area.
- The security policy's preference is to use inbound ACLs to protect each security zones. Their security design protects each security zones by limiting what can enter each zone. For the most part, each security zone is allowed to transmit all traffic into the backbone.

EXAMPLE - Building B's ACLs

This example will use Building B's subnets to create the necessary security defenses to protect each security zone. The following subnets are used in Building B. Each user security zone will have access to their respective server security zones. Two QA servers in the QA Server Security Zone are allowed into the R&D Server Security Zone to pull the latest versions for testing. All other access is forbidden.

Server Security Zones	**Subnet**
Research and Development Servers	10.48.1.0/24
Staging and Testing Network	10.48.2.0/24

Users Security Zones	**Subnet**
Research and Development Users	10.49.1.0/24
Developers and Quality Assurance	10.49.2.0/24

NOTE: Remember that the most general ACLs will be used closest to the network backbone and the most specific and restrictive ACLs will be used closest to the hosts. Inbound ACLs will be used on the switch's uplink port to Building B's backbone router to control traffic entering the subnet. Inbound ACLs will be used on Building B's backbone router port to control traffic leaving the subnet. Place the most specific rules at the top of the ACL and the most general rules at the bottom of the ACL. Once a rule is matched, the ACL is executed and the remaining rules are ignored.

R&D Server Security Zone – 10.48.1.0/24

On Building B's R&D Server Security Zone's Layer 2 switch, the following inbound ACL is required on the uplink port(s) connecting to the Building B's backbone router. This ACL controls access into the subnet.

Access granted to the necessary Domain Controllers, DNS servers, etc.

```
B_R&D_Farm-FI-1(config)# access-list 1 permit host 10.33.4.250
B_R&D_Farm-FI-1(config)# access-list 1 permit host 10.49.1.250
B_R&D_Farm-FI-1(config)# access-list 1 permit host 10.64.1.250
B_R&D_Farm-FI-1(config)# access-list 1 permit host 10.96.1.250
```

Access granted to the necessary QA Servers

```
B_R&D_Farm-FI-1(config)# access-list 1 permit host 10.48.2.10
B_R&D_Farm-FI-1(config)# access-list 1 permit host 10.48.2.11
```

Access granted to all R&D users

```
B_R&D_Farm-FI-1(config)# access-list 1 permit 10.49.1.0/24
```

Access granted to the IT and NOC Operations teams

```
B_R&D_Farm-FI-1(config)# access-list 1 permit 10.64.3.0/24
B_R&D_Farm-FI-1(config)# access-list 1 permit 10.64.4.0/24
```

Deny all other users and servers access (this statement is optional as an implied DENY is used by default)

```
B_R&D_Farm-FI-1(config)# access-list 1 deny any
```

Apply the inbound ACL to the R&D Switch's uplink port

```
B_R&D_Farm-FI-1(config)# interface e 1/1
B_R&D_Farm-FI-1(config-if-1/1)# ip access-group 1 in
B_R&D_Farm-FI-1(config-if-1/1)# write memory
```

Staging & Testing Server Security Zone - 10.48.2.0/24

On Building B's Staging & Testing Server Security Zone's Layer 2 switch, the following inbound ACL is required on the uplink port(s) connecting to the Building B's backbone router:

Access granted to the necessary Domain Controllers, DNS servers, etc

```
B_QA_Farm-FI-1(config)# access-list 1 permit host 10.33.4.250
B_QA_Farm-FI-1(config)# access-list 1 permit host 10.49.1.250
B_QA_Farm-FI-1(config)# access-list 1 permit host 10.64.1.250
B_QA_Farm-FI-1(config)# access-list 1 permit host 10.96.1.250
```

Access granted to all Developers & QA users and R&D users

```
B_QA_Farm-FI-1(config)# access-list 1 permit 10.49.1.0/24
B_QA_Farm-FI-1(config)# access-list 1 permit 10.49.2.0/24
```

Access granted to the IT and NOC Operations teams
```
B_QA_Farm-FI-1(config)# access-list 1 permit 10.64.3.0/24
B_QA_Farm-FI-1(config)# access-list 1 permit 10.64.4.0/24
```

Deny all other users and servers access (this statement is optional as an implied DENY is used by default)
```
B_QA_Farm-FI-1(config)# access-list 1 deny any
```

Apply the inbound ACL to the Staging & Testing Switch's uplink port
```
B_QA_Farm-FI-1(config)# interface e 1/1
B_QA_Farm-FI-1(config-if-1/1)# ip access-group 1 in
B_QA_Farm-FI-1(config-if-1/1)# write memory
```

Controlling Undesired Outbound Traffic

This subnet is used to test new code from R&D. The company's product uses two customized UDP ports to communicate between the end-user client application and the server's database application: UDP 34800 and UDP 34801. In the past, this subnet has accidentally produced UDP storms that affected other subnets when the code was not properly written. In order to prevent this from happening again, the IT team has implemented ACLs to block these two UDP ports from leaving the Staging & Testing Server Security Zone.

Outbound traffic for this subnet is controlled on Building B's backbone router port E1/2. Notice the outbound IP Anti-Spoofing ACL that is incorporated into this ACL. The most specific rules are at the top of the ACL and the most general at the bottom.

```
BigIron-BuildingB(config)# access-list 115 deny udp 10.48.2.0/24 eq 34800 any
BigIron-BuildingB(config)# access-list 115 deny udp 10.48.2.0/24 eq 34801 any
BigIron-BuildingB(config)# access-list 115 permit 10.48.2.0/24 any
BigIron-BuildingB(config)# access-list 115 deny any any

BigIron-BuildingB(config)# interface E1/2
BigIron-BuildingB(config-if-1/2)# ip access-group 115 in
BigIron-BuildingB(config-if-1/2)# write memory
```

R&D User Security Zone – 10.49.1.0/24

On Building B's Research & Development User subnet's Layer 2 switch, the following inbound ACL is required on the uplink port(s) connecting to the Building B's backbone router. Since the users zones are allowed to access the public Internet, the source IP addresses entering the subnet will vary greatly. For the user subnets, the reverse logic is applied to deny unwanted internal access and to allow all other traffic. This will restrict internal users and allow Internet traffic through.

Access granted to the necessary Domain Controllers, DNS servers, Print Servers, etc.
```
B_R&D_Users-FI-1(config)# access-list 1 permit host 10.33.4.250
B_R&D_Users-FI-1(config)# access-list 1 permit host 10.64.1.250
B_R&D_Users-FI-1(config)# access-list 1 permit host 10.96.1.250
B_R&D_Users-FI-1(config)# access-list 1 permit host 10.33.1.240
B_R&D_Users-FI-1(config)# access-list 1 permit host 10.64.1.240
B_R&D_Users-FI-1(config)# access-list 1 permit host 10.96.1.240
```

Permit the infrastructure servers (email, calendar, etc) and deny all other servers from Building A's server farms.
```
B_R&D_Users-FI-1(config)# access-list 1 permit 10.32.6.0/24
B_R&D_Users-FI-1(config)# access-list 1 deny 10.32.0.0/16
```

Deny all remaining unwanted internal traffic - grant access to all R&D servers, Staging & Testing servers, IT Users, and NOC Operations by **leaving them off the DENY list**. The VPN subnet is allowed because access

restrictions are performed with the VPN Policies defined on the VPN server. The DMZ Perimeter subnet is allowed so users can access the corporate web servers and DMZ resources.

```
B_R&D_Users-FI-1(config)# access-list 1 deny 10.33.0.0/16
B_R&D_Users-FI-1(config)# access-list 1 deny 10.49.2.0/24
B_R&D_Users-FI-1(config)# access-list 1 deny 10.64.1.0/24
B_R&D_Users-FI-1(config)# access-list 1 deny 10.64.2.0/24
B_R&D_Users-FI-1(config)# access-list 1 deny 10.64.5.0/24
B_R&D_Users-FI-1(config)# access-list 1 deny 10.96.1.0/24
```

Access granted all other subnets and IP addresses to support Internet traffic

```
B_R&D_Users-FI-1(config)# access-list 1 permit any
```

Apply the inbound ACL to the R&D User Switch's uplink port

```
B_R&D_Users-FI-1(config)# interface e 1/1
B_R&D_Users-FI-1(config-if-1/1)# ip access-group 1 in
B_R&D_Users-FI-1(config-if-1/1)# write memory
```

Developers & QA User Security Zone - 10.49.2.0/24

On Building B's Developers & QA User subnet's Layer 2 switch, the following inbound ACL is required on the uplink port(s) connecting to the Building B's backbone router. Since the users zones are allowed to access the Public Internet, the source IP addresses entering the subnet will vary greatly. For the user subnets, the reverse logic is applied to deny unwanted internal access and to allow all other traffic. This will restrict internal users and allow Internet traffic through.

Access granted to the necessary Domain Controllers, DNS servers, Print Servers, etc

```
B_QA_Users-FI-1(config)# access-list 1 permit host 10.33.4.250
B_QA_Users-FI-1(config)# access-list 1 permit host 10.49.1.250
B_QA_Users-FI-1(config)# access-list 1 permit host 10.64.1.250
B_QA_Users-FI-1(config)# access-list 1 permit host 10.96.1.250
B_QA_Users-FI-1(config)# access-list 1 permit host 10.33.1.240
B_QA_Users-FI-1(config)# access-list 1 permit host 10.49.1.240
B_QA_Users-FI-1(config)# access-list 1 permit host 10.64.1.240
B_QA_Users-FI-1(config)# access-list 1 permit host 10.96.1.240
```

Permit the infrastructure servers (email, calendar, etc) and deny all other servers from Building A's server farms.

```
B_QA_Users-FI-1(config)# access-list 1 permit 10.32.6.0/24
B_QA_Users-FI-1(config)# access-list 1 deny 10.32.0.0/16
```

Deny all remaining unwanted internal traffic - grant access to all Staging & Testing servers, IT Users, and NOC Operations by **leaving them off the DENY list**. The VPN subnet is allowed because access restrictions are performed with the VPN Policies defined on the VPN server. The DMZ Perimeter subnet is allowed so users can access the corporate web servers and DMZ resources.

```
B_QA_Users-FI-1(config)# access-list 1 deny 10.33.0.0/16
B_QA_Users-FI-1(config)# access-list 1 deny 10.48.1.0/24
B_QA_Users-FI-1(config)# access-list 1 deny 10.64.1.0/24
B_QA_Users-FI-1(config)# access-list 1 deny 10.64.2.0/24
B_QA_Users-FI-1(config)# access-list 1 deny 10.64.5.0/24
B_QA_Users-FI-1(config)# access-list 1 deny 10.96.1.0/24
```

Access granted all other subnets and IP addresses to support Internet traffic

```
B_QA_Users-FI-1(config)# access-list 1 permit any
```

Apply the inbound ACL to the R&D User Switch's uplink port

```
B_QA_Users-FI-1(config)# interface e 1/1
B_QA_Users-FI-1(config-if-1/1)# ip access-group 1 in
B_QA_Users-FI-1(config-if-1/1)# write memory
```

ACLs For Building B's Router

In all of the examples listed in the previous sections, notice that control is based on inbound traffic into Building B's security zone subnets. In order to completely protect this building, remember to place the necessary IP Anti-Spoofing ACLs at the router interfaces connecting to these subnets. They should be applied in the inbound direction on the Building B Router's interfaces.

EXAMPLE

This example applies the remaining Anti-Spoofing ACLs to the router ports on Building B's router. Subnet 10.48.2.0/24 was combined with the UDP protection earlier.

```
BigIron-BuildingB(config)# access-list 21 permit 10.48.1.0/24
BigIron-BuildingB(config)# access-list 21 deny any log

BigIron-BuildingB(config)# access-list 23 permit 10.49.1.0/24
BigIron-BuildingB(config)# access-list 23 deny any log

BigIron-BuildingB(config)# access-list 24 permit 10.49.2.0/24
BigIron-BuildingB(config)# access-list 24 deny any log

BigIron-BuildingB(config)# interface ethernet 1/1
BigIron-BuildingB(config-if-1/1)# ip access-group 21 in

BigIron-BuildingB(config)# interface ethernet 1/3
BigIron-BuildingB(config-if-1/3)# ip access-group 23 in

BigIron-BuildingB(config)# interface ethernet 1/4
BigIron-BuildingB(config-if-1/4)# ip access-group 24 in
```

EXAMPLE – Remote Office's ACLs

The remote office is primarily a sales office with a training facility for Widget-Works.COM's customers. The sales and training staff located in this office needs access to the following internal resources.

Server and User Security Zones	**Subnet**
Executive/Admin Servers	10.32.1.0/24
Corporate General Storage Servers	10.32.5.0/24
Infrastructure Servers (email, calendar, etc)	10.32.6.0/24
Sales & Marketing Users	10.33.3.0/24
IT Users	10.64.3.0/24
NOC Operations	10.64.4.0/24

Infrastructure Hosts	**IP Address**
Building A Domain Controller	10.33.4.250/32
Building B Domain Controller	10.49.1.250/32
Building C Domain Controller	10.64.1.250/32
Remote Office Domain Controller	10.96.1.250/32

Building A Print Server	10.33.1.240/32
Building B Print Server	10.49.1.240/32
Building C Print Server	10.64.1.240/32

Controlling Inbound Traffic – 10.96.1.0/24

Since this office contains only one subnet with minimal file servers, explicit PERMIT ACLs will be used to grant access to specific internal resources and DENY ACLs will be used to restrict the unauthorized internal resources - an implied PERMIT at the end to allow all remaining internal traffic and pubic Internet traffic.

The most specific rules are placed at the beginning of the ACL and the most general rules at the end. Once a rule is matched, the ACL is executed and the remaining rules are ignored. The following inbound ACL is required on the remote office's Layer 2 switch on the uplink port(s) to remote office's WAN router.

Access granted to the necessary Domain Controllers, DNS servers, Print Servers, etc

```
Roffice_FI-1(config)# access-list 1 permit host 10.33.4.250
Roffice_FI-1(config)# access-list 1 permit host 10.49.1.250
Roffice_FI-1(config)# access-list 1 permit host 10.64.1.250
Roffice_FI-1(config)# access-list 1 permit host 10.33.1.240
Roffice_FI-1(config)# access-list 1 permit host 10.49.1.240
Roffice_FI-1(config)# access-list 1 permit host 10.64.1.240
```

Permit the necessary internal network subnets

```
Roffice_FI-1(config)# access-list 1 permit 10.32.1.0/24
Roffice_FI-1(config)# access-list 1 permit 10.32.5.0/24
Roffice_FI-1(config)# access-list 1 permit 10.32.6.0/24
Roffice_FI-1(config)# access-list 1 permit 10.33.3.0/24
Roffice_FI-1(config)# access-list 1 permit 10.64.3.0/24
Roffice_FI-1(config)# access-list 1 permit 10.64.4.0/24
```

```
Deny all remaining internal network subnets
Roffice_FI-1(config)# access-list 1 deny 10.32.0.0/16
Roffice_FI-1(config)# access-list 1 deny 10.33.0.0/16
Roffice_FI-1(config)# access-list 1 deny 10.48.0.0/16
Roffice_FI-1(config)# access-list 1 deny 10.49.0.0/16
Roffice_FI-1(config)# access-list 1 deny 10.64.0.0/16
Roffice_FI-1(config)# access-list 1 deny 192.168.0.0/16
```

Access granted all other subnets and IP addresses to support Internet traffic

```
Roffice_FI-1(config)# access-list 1 permit any
```

Apply the inbound ACL to the Remote Office Layer 2 switch's uplink port

```
Roffice_FI-1(config)# interface e 1/1
Roffice_FI-1(config-if-1/1)# ip access-group 1 in
Roffice_FI-1(config-if-1/1)# write memory
```

Controlling Outbound Traffic – 10.96.1.0/24

Without the presence of dedicated security guards, access to the remote office is not as well secured as the HQ campus. Reception watches the main entrance during business hours, but physical access to the building cannot be guaranteed at all times. There are also customers in the same building as the sales and training staff when classes are held. Widget-Works.COM's Corporate Security Policy has dictated that traffic coming from the Remote Office be scrutinized more tightly and the IT team have implemented additional Outbound policies to control traffic leaving the remote office network.

The following inbound ACL is required on the remote office's WAN router port that is used to define the 10.96.0.1/24 network. This ACL is an extended ACL and not a standard ACL.

Access granted to the necessary Domain Controllers, DNS servers, Print Servers, etc

```
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.33.4.250/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.49.1.250/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.64.1.250/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.96.1.250/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.33.1.240/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.49.1.240/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.64.1.240/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.96.1.240/32
```

```
Access granted to the necessary IT and NOC Operation management stations
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.64.3.100/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.64.3.101/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.64.4.15/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.64.4.16/32
```

Access granted to the necessary Executive/Admin servers

```
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.32.1.10/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.32.1.11/32
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.32.1.12/32
```

Permit the necessary internal network subnets

```
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.32.5.0/24
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.32.6.0/24
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 10.33.3.0/24
```

Deny all remaining internal network subnets. The training network is included as a precaution.

```
Roffice_Router-1(config)# access-list 101 deny ip 10.96.1.0/24 10.32.0.0/16
Roffice_Router-1(config)# access-list 101 deny ip 10.96.1.0/24 10.33.0.0/16
Roffice_Router-1(config)# access-list 101 deny ip 10.96.1.0/24 10.48.0.0/16
Roffice_Router-1(config)# access-list 101 deny ip 10.96.1.0/24 10.49.0.0/16
Roffice_Router-1(config)# access-list 101 deny ip 10.96.1.0/24 10.64.0.0/16
Roffice_Router-1(config)# access-list 101 deny ip 192.168.0.0/16 any
```

Access granted all other subnets and IP addresses to support Internet traffic. The IP Anti-Spoofing rule is added at this level to deny all other traffic that is not originating from this subnet.

```
Roffice_Router-1(config)# access-list 101 permit ip 10.96.1.0/24 any
Roffice_Router-1(config)# access-list 101 deny ip any any
```

Apply the inbound ACL to the Remote Office router's port that supports the 10.96.1.0/24 subnet

```
Roffice_Router-1(config)# interface e 1
Roffice_Router-1(config-if-1)# ip access-group 101 in
Roffice_Router-1(config-if-1)# write memory
```

EXAMPLE – Building C's ACLs

Access list security for the remaining buildings and security zones will be implemented in the same fashion as the examples listed in Building B and the Remote Office. However, Building C has some differences that are worth mentioning. With all of the IT, MIS, and NOC Operations groups being in Building C, extra security needs to be applied to their respective subnets. These department's workstations and monitoring stations have more access rights than the typical end user, and would be very valuable to a hacker if they could gain access to them.

The following example will demonstrate the use of Extended ACLs and how they can be used to provide additional layers of tighter security at the host or individual switch levels.

Protecting The NOC Operations Security Zone – 10.64.4.0/24

In order to protect the NOC Operations security zone, the IT department implemented the following policies:

- Only the SNMP, RMON, and sFlow data should be allowed into the subnet from every device.
- The IT Team has full access to the NOC Operations subnet. The NOC Operations group is a subgroup of the IT Team and report into the same IT Director.
- The NOC Operations team must be able to print to every printer to distribute trouble tickets to each helpdesk support rep.
- The NOC Operations team also needs access to the Internet to perform their work.

From the policies listed, the following inbound ACL will be applied at the NOC Operations group's Layer 2 switch uplink port(s) that connects the subnet to Building C's backbone router.

Permit all internal networks access to send SNMP, RMON, and sFlow datagrams to the NOC Operations monitoring stations. SNMP operations on UDP 161 and 162, RMON is a subset of SNMP, and sFlow operates on UDP 6343. The border router and firewall blocks all of these management ports from the Public Internet - the **any** source ip address wildcard is used to specify all internal networks.

```
C_NOC-FI-1(config)# access-list 100 permit upd any 10.64.4.15/32 eq 161
C_NOC-FI-1(config)# access-list 100 permit upd any 10.64.4.15/32 eq 162
C_NOC-FI-1(config)# access-list 100 permit upd any 10.64.4.16/32 eq 161
C_NOC-FI-1(config)# access-list 100 permit upd any 10.64.4.16/32 eq 162
C_NOC-FI-1(config)# access-list 100 permit upd any 10.64.4.15/32 eq 6343
C_NOC-FI-1(config)# access-list 100 permit upd any 10.64.4.16/32 eq 6343
```

Allow the management ports from the firewall and IDS sensors back into the NOC Operations security zone. Widget-Works.COM configured their management ports as follows: firewall is set to 8010 and IDS sensors are set to 8020. There are 3 firewalls in the 198.30.15.0/24 DMZ Perimeter Network and one in the 10.64.254.0/24 Network. The 2 IDS sensors are in the 198.30.15.0/24 DMZ Perimeter Network.

```
C_NOC-FI-1(config)# access-list 100 permit ip 198.31.15.250/32 10.64.4.0/24 eq 8010
C_NOC-FI-1(config)# access-list 100 permit ip 198.31.15.240/32 10.64.4.0/24 eq 8010
C_NOC-FI-1(config)# access-list 100 permit ip 198.31.15.230/32 10.64.4.0/24 eq 8010
C_NOC-FI-1(config)# access-list 100 permit ip 10.64.254.250/32 10.64.4.0/24 eq 8010
C_NOC-FI-1(config)# access-list 100 permit ip 198.31.15.201/32 10.64.4.0/24 eq 8020
C_NOC-FI-1(config)# access-list 100 permit ip 198.31.15.202/32 10.64.4.0/24 eq 8020
```

Grant access to the necessary Domain Controllers, DNS servers, Print Servers, etc

```
C_NOC-FI-1(config)# access-list 100 permit ip 10.33.4.250/32 any
C_NOC-FI-1(config)# access-list 100 permit ip 10.49.1.250/32 any
C_NOC-FI-1(config)# access-list 100 permit ip 10.64.1.250/32 any
C_NOC-FI-1(config)# access-list 100 permit ip 10.96.1.250/32 any
C_NOC-FI-1(config)# access-list 100 permit ip 10.33.1.240/32 any
C_NOC-FI-1(config)# access-list 100 permit ip 10.49.1.240/32 any
C_NOC-FI-1(config)# access-list 100 permit ip 10.64.1.240/32 any
C_NOC-FI-1(config)# access-list 100 permit ip 10.96.1.240/32 any
```

```
Grant the IT Team full access and block all other internal subnets
C_NOC-FI-1(config)# access-list 100 permit ip 10.64.3.0/24 any
C_NOC-FI-1(config)# access-list 100 deny ip 10.32.0.0/16 any
C_NOC-FI-1(config)# access-list 100 deny ip 10.33.0.0/16 any
C_NOC-FI-1(config)# access-list 100 deny ip 10.48.0.0/16 any
C_NOC-FI-1(config)# access-list 100 deny ip 10.49.0.0/16 any
C_NOC-FI-1(config)# access-list 100 deny ip 10.64.0.0/16 any
```

```
C_NOC-FI-1(config)# access-list 100 deny ip 10.96.0.0/16 any
```

Grant access to all IP addresses to support Internet traffic
```
C_NOC-FI-1(config)# access-list 100 permit ip any 10.64.4.0/24
```

Apply the inbound ACL to the NOC Operations groups switch's uplink port
```
C_NOC-FI-1(config)# interface e 1/1
C_NOC-FI-1(config-if-1/1)# ip access-group 100 in
C_NOC-FI-1(config-if-1/1)# write memory
```

EXAMPLE – Common Shared Areas

Every building that Widget-Works.COM occupies has commonly shared areas. These include conference rooms, reception areas, and common office space that visitors can use. The IT department has created two separate networks using multiple network jacks in each shared area. The primary network jack is part of the default corporate network and is protected in each of the shared areas with 802.1x Port Authentication – it is reserved for employees only. The secondary network jack is setup for guest access with limited access to the Public Internet.

The secondary network jack is connected to a separate subnet that is joined by multiple switches and port-based VLANs to span the campus. Widget-Works.COM has enough fiber cables between the buildings to place a dedicated switch or a port-base VLAN in each building to form the guest network. The Guest Network's subnet is terminated in Building C's DMZ Perimeter Network using the 198.30.15.0/24 address space – it is between the 2 perimeter firewalls and outside of the screened subnets.

There are less than 20 publicly accessed network jacks for the entire campus and the IT Team has decided to post the static IP address information in each conference room and guest area to accommodate Public Internet access. Static addresses allow the NOC team to quickly track each guest to a specific conference room or shared location. The company's Corporate Security Policy forbids a DHCP server to be placed in the DMZ. The external DNS server supplies DNS services for guests and all non-authoritative searches are forwarded to the ISP's DNS servers. The firewalls in the DMZ are used to protect and govern the access of these visitors – ensuring that they are only accessing the public Internet.

Since there will only be a maximum of 20 guest users and their access to the Public Internet should not be bandwidth intensive, the decision was made to incorporate many IronShield Security features that prevented DoS and broadcast style attacks. The additional defenses will use more resources on the switch, but the risk exposure justifies the added protection.

The IT and Security teams have implemented the following IronShield Security features to strengthen existing firewall and IDS defenses:

- DoS and DDoS protection features
  - Smurf protection
  - TCP SYN protection
  - Proxy ARP
  - ARP Attack protection
  - ICMP protection
  - Broadcast protection
  - Fragmentation protection
- MAC Address and ARP Spoofing protection
- IP Anti-Spoofing ACLs for each Layer 2 port
- DMZ Limitation ACLs for each Layer 2 port

For more information on DoS and DDoS protection, please refer to the previous section titled, "*Denial of Service (DoS) Prevention*".

For each Layer 2 guest network port, both inbound and outbound ACLs were applied. Performance is not a major concern for the guest network and Widget-Works.COM's corporate security policy has dictated security over speed on the guest network to protect DMZ resources.

Inbound ACLs Applied

Inbound ACLs will control traffic coming from the guest workstation. The ACL will be applied to each of the 20 guest ports and will stop any outbound IP Address spoofing activity coming from the guest workstation.

Define the Anti-Spoofing ACL for the IP address that will be used in each conference room and public accessible location.

```
B_Guest-FES-1(config)# access-list 10 permit host 198.30.15.61
B_Guest-FES-1(config)# access-list 10 deny any
```

Apply the IP address Anti-Spoofing ACL to the correct switch port to lock the ip address down to the specific conference room it services.

```
B_Guest-FES-1(config)# interface e10
B_Guest-FES-1(config-if-10)# ip access-group 10 in
B_Guest-FES-1(config-if-10)# write memory
```

Outbound ACLs Applied

Outbound ACLs will control traffic going to the guest workstation. This ACL will allow all DMZ traffic from the external DNS server, the public web site, and Public Internet to the guest workstation. All traffic from Widget-Works.COM's internal 10.0.0.0/8 address space must be blocked - the internal 10.0.0.0/8 address space is NATed to the external 198.30.15.253 IP address. Traffic from each of the 20 guest ports should not be allowed to see each other – this will prevent guest workstations from transferring traffic to each other.

Restrict NATed internal network traffic from 198.30.15.253 and block all other guest interface IP addresses. The source IP address for this guest workstation is also specified as a deny ACL to provide anti-spoofing protection. No other inbound packet to the guest workstation should claim to have a source IP address of the guest workstation.

```
B_Guest-FES-1(config)# access-list 11 deny host 198.30.15.253
B_Guest-FES-1(config)# access-list 11 deny host 198.30.15.61
B_Guest-FES-1(config)# access-list 11 deny host 198.30.15.62
B_Guest-FES-1(config)# access-list 11 deny host 198.30.15.63
B_Guest-FES-1(config)# access-list 11 deny host 198.30.15.64
            :                 :                 :
            :                 :                 :
B_Guest-FES-1(config)# access-list 11 deny host 198.30.15.78
B_Guest-FES-1(config)# access-list 11 deny host 198.30.15.79
B_Guest-FES-1(config)# access-list 11 deny host 198.30.15.80
```

Allow all other DMZ and Public Internet traffic

```
B_Guest-FES-1(config)# access-list 11 permit any
```

Apply the outbound ACL to the correct switch port

```
B_Guest-FES-1(config)# interface e10
B_Guest-FES-1(config-if-10)# ip access-group 110 out
B_Guest-FES-1(config-if-10)# write memory
```

NOTE: Widget-Works.COM could have also configured the guest network using NAT to translate the 20 guest ip addresses to just one 198.30.15.0/24 address - to conserve IP addresses. The drawback is losing the tracking visibility for each individual guest as NATing will combine all guest traffic into one NATed 198.30.15.0/24 address.

Policy-Based Routing (PBR)

Foundry's Policy-Based Routing (PBR) is used to allow the redirection of traffic based on Layer 3 IP address or Layer 4 port information to another subnet or host through one or multiple defined next-hop gateways. It is useful in performing the following tasks:

- Creating containment networks that can be used to trap and analyze network traffic – supporting "honeypots".
- Sending the traffic to a null interface (null0) based on Layer 3 and/or Layer 4 information. This feature is very valuable when your network is under attack and the attack packets are pushing CPU utilization to very high levels.
- Assisting border routers to differentiate and steer Layer 4 traffic to specific firewalls for processing.
- Directing traffic from specific subnets to dedicated subnets or devices for processing.

PBR can be used with both Standard and Extended ACLs and uses route-maps to redirect the traffic. PBR can be used on both IronCore and JetCore devices with the following differences.

IronCore Devices

- PBR is only supported on Layer 3 switches.
- On Chassis devices that use a Management I, II, III, or IV module, source routing occurs in the CPU, not in the ASICs.
- PBR is not supported for traffic coming from NPA or non-NPA OC-48 POS modules, from ATM modules, or on the FastIron 4802.
- PBR is supported on non-NPA OC-3 and OC-12 POS modules, on 10/100 modules, and on Gigabit Ethernet modules when these modules are in Chassis devices that are using the Velocity Management Module (VM1).

JetCore Devices

- The ACLs are programmed into Layer 4 CAM for faster performance.
- JetCore supports an unlimited number of PBR policies that contain a single route map instance and a single ACL.
- JetCore supports up to 64 PBR policies that have more than one route map instance or more than one ACL. In this case, a given policy can have up to six route map instances, with up to six ACLs in each instance, and up to six next hops in each ACL.
- The ACL **log** and <icmp-type> options cause PBR to be performed by the CPU instead of in hardware. No CAM entries are used.
- PBR ignores explicit or implicit **deny ip any any** ACL entries, to ensure that for route maps that use multiple ACLs, the traffic is compared to all the ACLs.
- PBR always selects the first next hop from the next hop list that is up, unless you use the **ip policy prefer-direct-route** option. If you use this option, PBR selects a direct route instead. If a PBR policy's next hop goes down, the policy uses another next hop if available. If no next hops are available, the device sends the traffic to the CPU for forwarding.

- By default PBR matches a fragment to an ACL if the source and destination addresses in the fragment exactly match an ACL. In this case, PBR uses the next hop that was used for the first fragment, which contains the Layer 4 UDP or TCP application port information. Alternatively, you can configure PBR to select the best next hop on an individual fragment basis.

Configuring PBR Policies

Enabling PBR is a four-step process:

1. Configure the ACLs with the Layer 3 and/or Layer 4 information to match the traffic on.
2. Configure a route map that matches on the ACLs and sets the route information.
3. Optionally, enable PBR to use the most direct route if available.
4. Apply the route map to an interface.

EXAMPLE – Null Route

Widget-Works.COM uses Null Routes (null0) and PBR to protect the network against high volumes of unwanted traffic. These rules are only implemented when they are needed - during DoS attack activity. The Null Route is created on the Layer 3 switch to protect the subnet or host IP address under attack.

The NOC is noticing a very high volume of UDP traffic on the MIS development network that is starting to spread to other areas of the internal network. sFlow traffic analysis is showing an increasing number of UDP 1434 packets originating from several Microsoft SQL servers in the MIS subnets 10.32.4.0/24 and 10.64.1.0/24. Not knowing what this is, the IT team has decided to create a Null Route on each of the corresponding routers and use PBR to redirect all UPD 1434 traffic to the Null Route to be dropped.

A DENY ACL can also be used, but NOC has recognized that the source IP addresses are being spoofed in the UDP packets – causing CPU utilization to run high. Adding another ACL to deny traffic with spoofed IP addresses may add to the already high CPU utilization. A decision is made to drop all UDP 1434 traffic with a Null Route in hardware.

Step 1: Create the Static Null Routes on both Building A and Building C routers where the SQL servers reside.

Syntax: ip route <ip-addr>/<mask-bits> null0

On Building A's router, configure the static null route

```
BigIron_BuildingA(config)# ip route 10.34.1.0/24 null0
BigIron_BuildingA(config)# write memory
```

On Building C's router, configure the static null route

```
BigIron_BuildingC(config)# ip route 10.65.1.0/24 null0
BigIron_BuildingC(config)# write memory
```

Step 2: Create the ACLs that identify the offending UDP 1434 packets

On Building A's router, create the ACL

```
BigIron_BuildingA(config)# access-list 150 permit udp any any eq 1434
BigIron_BuildingA(config)# write memory
```

On Building C's router, create the ACL

```
BigIron_BuildingC(config)# access-list 150 permit udp any any eq 1434
BigIron_BuildingC(config)# write memory
```

Step 3: Configure the Route Map that matches on the ACL for all UDP 1434 packets

Syntax: [no] route-map <map-name> permit | deny <num>

The <map-name> is a string of characters that names the map. Map names can be up to 32 characters in length. You can define up 50 route maps on the Layer 3 Switch.

The **permit | deny** parameter specifies the action the Layer 3 Switch will take if a route matches a match statement.

- If you specify **deny**, the Layer 3 Switch does not advertise or learn the route.
- If you specify **permit**, the Layer 3 Switch applies the match and set statements associated with this route map instance.

The <num> parameter specifies the instance of the route map you are defining. Each route map can have up to 50 instances. Routes are compared to the instances in ascending numerical order. For example, a route is compared to instance 1, then instance 2, and so on.

Syntax: [no] match ip address <ACL-num-or-name>

The <ACL-num> parameter specifies a standard or extended ACL number or name.

Syntax: [no] set ip next hop <ip-addr>

This command sets the next-hop IP address for traffic that matches a match statement in the route map.

EXAMPLE

On Building A's router, create the route map

```
BigIron_BuildingA(config)# route-map udp1434 permit 1
BigIron_BuildingA(config-routemap udp1434)# match ip address 150
BigIron_BuildingA(config routemap udp1434)# set ip next-hop 10.34.1.254
BigIron_BuildingA(config routemap udp1434)# exit
BigIron_BuildingA(config)# write memory
```

On Building C's router, create the route map

```
BigIron_BuildingC(config)# route-map udp1434 permit 1
BigIron_BuildingC(config-routemap udp1434)# match ip address 150
BigIron_BuildingC(config routemap udp1434)# set ip next-hop 10.65.1.254
BigIron_BuildingC(config routemap udp1434)# exit
BigIron_BuildingC(config)# write memory
```

Step 4: Apply the route map the proper router interface

Syntax: [no] ip policy route-map <map-name>

On Building A's router, apply the route map to the MIS Server Farm router port

```
BigIron_BuildingA(config)# interface E2/5
BigIron_BuildingA(config-if-2/5)# ip policy route-map udp1434
BigIron_BuildingA(config-if-2/5)# write memory
```

On Building C's router, apply the route map to the MIS subnet router port

```
BigIron_BuildingC(config)# interface E2/1
BigIron_BuildingC(config-if-2/1)# ip policy route-map udp1434
BigIron_BuildingC(config-if-2/1)# write memory
```

EXAMPLE – Honeypot

A "honeypot" is a term used in the security world to describe a fake host that is designed to look and perform like a legitimate host. The honeypot's role is to trick and divert intruders and hackers from the real host. Many IT and Security personnel use honeypots to slow hackers down and to learn how they are probing and infiltrating networks and hosts. Policy-Based Routing can be used to aid the creation of honeypots and subnets used to trap intruders. By setting up ACLs to match on suspected probing or intrusion traffic, PBR can redirect the undesired traffic to honeypot hosts for further examination.

Widget-Works.COM has decided to setup a honeypot in the DMZ to learn more about mischievous behavior entering the DMZ on the Telnet port. Telnet is a common port used by intruders to perform surveillance work. The honeypot host is armed with several traffic monitoring and analysis applications that will record the methods used by the attacker. Being very security conscious, Widget-Works.COM does not permit any Telnet access to their hosts. All remote access is performed by SSH or by dedicated terminal servers that have been hardened and require token-based authentication cards.

They are directing Telnet activity to a honeypot host to learn more about Telnet probing and intrusion attempts.

On the border router, create the ACL to look for inbound Telnet traffic

```
BRouter001(config)# access-list 130 permit tcp any any eq telnet
BRouter001(config)# write memory
```

On the border router, create the route map to specify the honeypot's address as the next hop

```
BRouter001(config)# route-map fake_telnet permit 1
BRouter001(config-routemap fake_telnet)# match ip address 130
BRouter001(config routemap fake_telnet)# set ip next-hop 198.30.15.23
BRouter001(config routemap fake_telnet)# exit
BRouter001(config)# write memory
```

On the border router, apply the route map to the external router port facing the ISP network

```
BRouter001(config)# interface E16
BRouter001(config-if-16)# ip policy route-map fake_telnet
BRouter001(config-if-16)# write memory
```

Virtual LANs (VLANs)

A network allows devices to talk to each other through a physical medium. In the simplest configuration, the physical medium is usually a series of Layer 2 switches and the cable plant that connects the hosts and switches together – forming a singe broadcast domain. As simple networks grow, routers are added to increase the number of broadcast domains and subnets - allowing more devices to be added to the network and to provide segmentation of traffic. Segmentation is possible because routers provide the physical barriers to separate broadcast domains – creating new physical LAN segments called subnets.

On modern networks, Virtual LANs (VLANs) can be used to group devices on different physical LAN segments to allow them to communicate and act as if the were on the same physical LAN segment. VLANs provide the following benefits:

Simplifies Manageability	VLANs allow devices from multiple physical segments to be grouped into the same broadcast domain. It offers greater flexibility and lowers costs for managing logical groups of devices in installations that are under constant change.
Flexible Network Configuration	VLANs allow network administrators to logically group hosts and devices together to form logical networks without rewiring the existing cable plant – allowing physical infrastructure independence. As networks grow and new devices are added, VLANs allow the administrators to quickly add the devices into the logical LAN segments and broadcast domains.
Group Like Devices	VLANs allow devices that are using the same protocols to be logically grouped into logical LAN segments. This can help keep specific traffic isolated to only the devices that require it. Examples include Appletalk, IPX, NetBIOS, and IP Protocol VLANs.
Increases Security Options	By grouping like devices together to form security zones, security defenses can be enhanced through the use of ACLs, Policy-Based Routing, and so forth. Sensitive information can be separated from regular data traffic regardless of physical location

Foundry VLANs

Foundry devices can support a wide range of VLAN types.

Layer 2 Port-Based VLAN	A Layer 2 port-based VLAN is a subset of ports on a Foundry device that has its own Layer 2 broadcast domain. By default, all ports belong to the default VLAN and share a single Layer 2 broadcast domain. As new port-based VLANs are created, the ports are removed from the default VLAN and moved into the new port-based VLAN. Port-based VLANs are logically separated from each other and is considered the most secure of all VLAN types. A port can belong to only one port-based VLAN, unless 802.1q tagging is applied to the port. 802.1q tagging uses a four-byte tag field on each packet transmitted from the port to identify the VLAN ID and allows the VLAN to span multiple devices.

Each Layer 2 VLAN runs its own separate instance of the Spanning Tree Protocol (STP) and all Layer 2 traffic is bridged within the port-based VLAN supporting a single broadcast domain.

Layer 3 Protocol VLANs

A Layer 3 Protocol VLAN is a subset of ports within a port-based VLAN that share a common, exclusive broadcast domain for a specified protocol type. It allows you to logically group devices using the same protocol type. The following Layer 3 Protocol VLANs are supported:

- Appletalk
- IP
- Ipv6
- IPX
- DECnet
- NetBIOS
- Other – all other protocols than the ones listed above

Foundry Layer 3 devices support Integrated Switch Routing (ISR) to allow routing between protocol VLANs without the need for an external router. The protocol VLANs must be using the same protocol.

IP Sub-Net VLANs

An IP sub-net VLAN is a subset of ports in a port-based VLAN that share a common, exclusive sub-net broadcast domain for a specified IP subnet. IP sub-net VLANs provide more granular broadcast control by limiting broadcasts to only the specified IP subnet (e.g. 10.32.1.0/24). IP Protocol VLANs provide the same broadcast domain for all IP networks regardless of the number of IP subnets. Can only be used on Layer 3 devices.

IPX Network VLANs

An IPX network VLAN is a subset of ports in a port-based VLAN that share a common, exclusive network broadcast domain for a specified IPX network. All devices using the IPX protocol can be logically grouped into one broadcast domain and only this VLAN will see the IPX traffic across the Layer 2 switch. IPX network VLANs can control IPX broadcasts more granularly by restricting only broadcasts specified for the IPX network VLAN. An IPX protocol VLAN will see all IPX broadcasts regardless of the IPX network. Can only be used on Layer 3 devices.

AppleTalk cable VLANs

An Appletalk cable VLAN is a subset of ports in a port-based VLAN that share a common, exclusive network broadcast domain for a specified AppleTalk cable range. All devices within the Appletalk cable range are treated as one logical network segment. Appletalk cable VLANs offer more granular broadcast control by allowing only the broadcasts directed to the Appletalk cable range. Unlike Appletalk protocol VLANs that broadcast all cable range broadcasts to every device in the VLAN. Can only be used on Layer 3 devices.

NOTE: IP Sub-Net VLANs, IPX Network VLANs, and Appletalk Cable VLANs can only route traffic between VLANs using the same protocols. For more information on how to setup and use VLANs on Foundry devices, refer to the *Foundry Enterprise Configuration and Management Guide* and the *Foundry Switch and Router Command Line Interface Reference.*

VLANs for Security Purposes

When considering VLANs for security purposes, port-based VLANs are more practical than protocol based VLANs. Port-based VLANs are very useful for grouping workstations and servers into security zones to implement access control and containment security strategies. As demonstrated in the previous sections, applying ACLs and Policy-Based Routing can greatly help enhance security defenses on the Internal Network. ACL security would be very difficult to implement if devices were not well-defined in proper security zones and port-based VLANs can help solve this problem.

Port-Based VLANs

If you will be using VLANs to create security zones, try to use only port-based VLANs. Port-based VLANs offer the simplest VLAN implementation and give full Layer 2 bridging characteristics to the VLAN ports. If your Layer 2 VLANs can be created without the use of 802.1q tagging, you can further simplify the VLAN implementation. For strong security, keeping the VLAN design simple has many advantages.

Foundry's implementation of Layer 2 port-based VLANs are very secure and offer the best protection against VLAN leakage. Port-based VLANs can span multiple boxes by using your existing cable plant to link the port-based VLANs together.

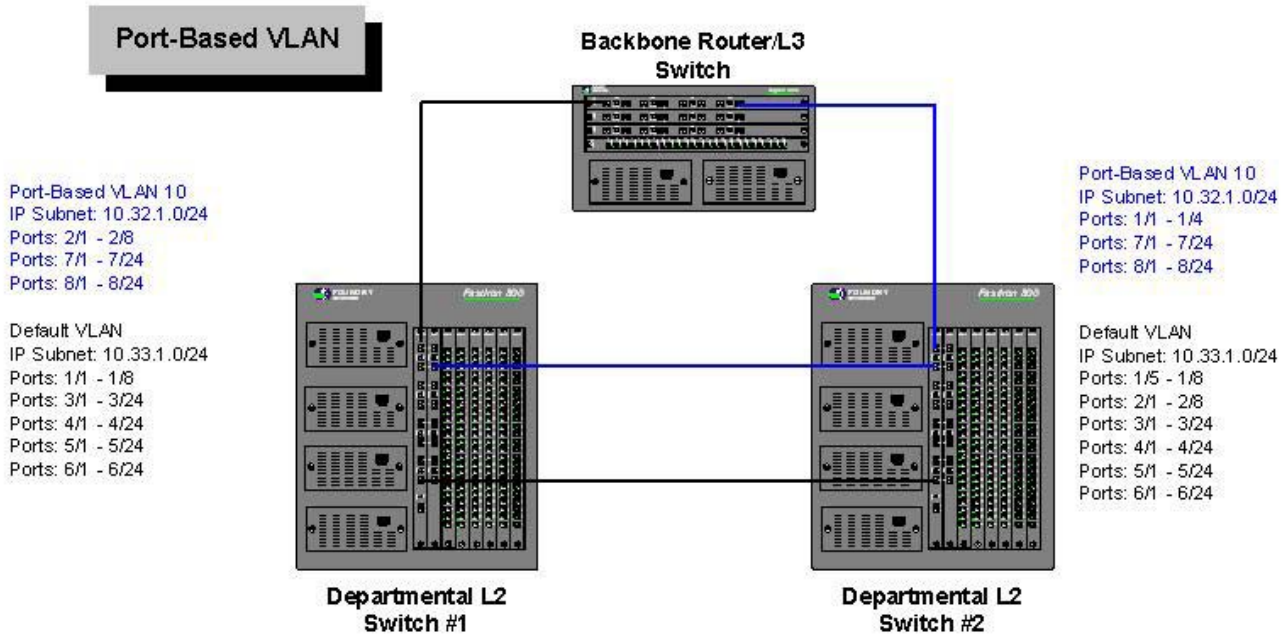


Figure 4. Port-Based VLAN

In this example, a port-based VLAN is defined on each departmental Layer 2 switch. The default VLAN is left intact on both switches to support the 10.33.1.0/24 subnet. A port-based VLAN numbered as VLAN 10 was created on each switch. Ports were removed from the default VLAN and added to VLAN 10 as outlined in the diagram. VLAN 10 is a completely separate broadcast domain and can be treated as a separate logical LAN segment.

A fiber cross-connect cable was used to tie both switches' default VLANs together to span the 10.33.1.0/24 subnet across both switches. Similarly, a fiber cross-connect cable was used to tie both switches port-based VLAN 10's together to span the 10.32.1.0/24 subnet across both switches.

To create the port-based VLAN: **Syntax:** vlan <vlan-id> by port

To add ports: **Syntax:** untagged ethernet | pos <portnum> [to <portnum> | ethernet <portnum>]

To turn on Spanning Tree Protocol: **Syntax:** [no] spanning-tree

EXAMPLE
To create the port-based VLAN used in Departmental Switch #1, perform the following:

```
Dept_Switch-1(config)# vlan 10 by port
Dept_Switch-1(config-vlan-10)# untagged eth 2/1 to 2/8 eth 7/1 to 7/24
eth 8/1 to 8/24
Dept_Switch-1(config-vlan-10)# spanning-tree
Dept_Switch-1(config-vlan-10)# exit
Dept_Switch-1(config)# write memory
```

The ports specified in the "untagged…" command will be removed from the default VLAN and placed in VLAN 10. All remaining ports are left in the default VLAN.

Port-Based VLANs With 802.1q

If you do not have the required cable plant to support the spanning of port-based VLANs across multiple switches, 802.1q tagging can be used to take advantage of the existing uplinks that are connecting the switches together.

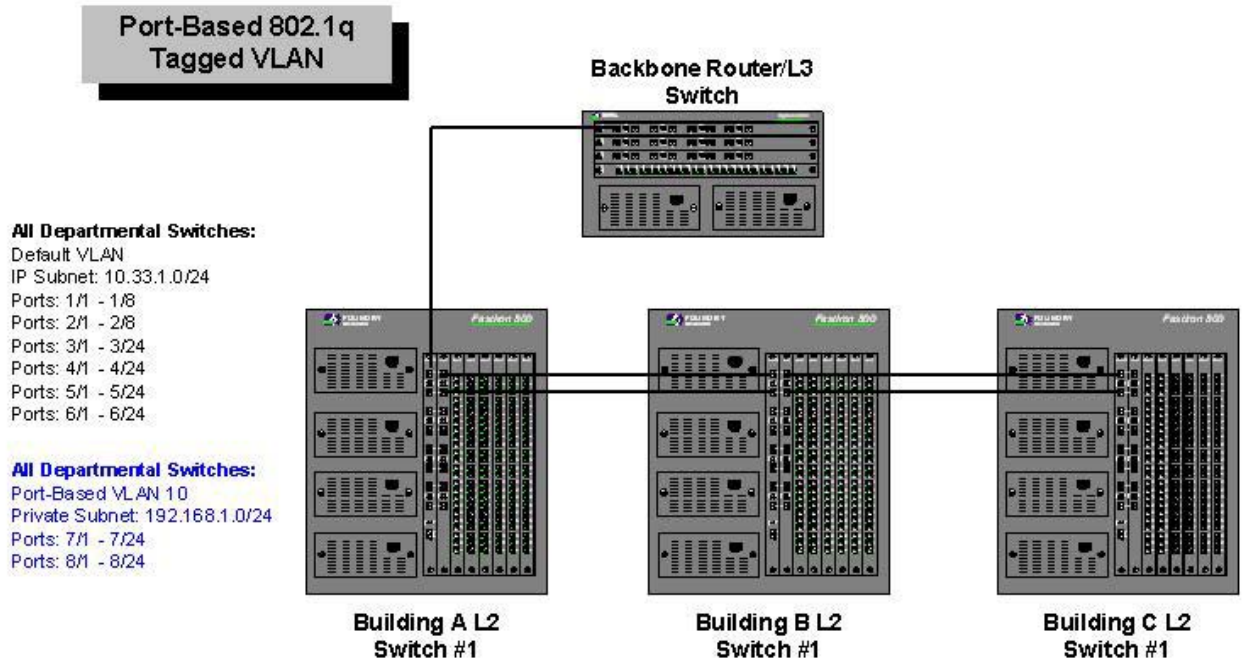


Figure 5. 802.1q Tagged VLAN

In the above example, there are three Layer 2 switches servicing the 10.33.1.0/24 subnet in three separate buildings: Building A, Building B, and Building C. Each switch is connected to the others with a 2 Gigabit trunked uplink. The company needs ports 7/1 – 7/24 and 8/1 – 8/24 on each switch dedicated to a test network (192.168.1.0/24) that will be completely isolated from the corporate data network. The test network will house its own test servers and QA workstations and will not need access to the Public Internet or any other corporate resource. A separate port-based VLAN will be created on each switch to separate the required ports from the default VLAN.

There are no fiber cables available to connect the test network's VLAN ports together across the three buildings. The company has decided to use 802.1q tagging to accomplish the task.

To create the port-based VLAN: **Syntax:** vlan <vlan-id> by port

To add untagged ports: **Syntax:** untagged ethernet | pos <portnum> [to <portnum> | ethernet <portnum>]

To add tagged ports: **Syntax:** tagged ethernet | pos <portnum> [to <portnum> | ethernet <portnum>]

To turn on Spanning Tree Protocol: **Syntax:** [no] spanning-tree

EXAMPLE

This example will create the necessary port-based VLANs on each of the three building switches and will tag the necessary uplink ports to allow the port-based VLAN to use the existing fiber uplinks. The port-based 802.1q VLAN will isolate the necessary ports on blades 7 and 8 by creating a new logical network segment and broadcast domain.

On Building A's Layer 2 Switch, perform the following:

```
Building_A-FI-1(config)# vlan 10 name testnet
Building_A-FI-1(config-vlan-10)# untagged eth 7/1 to 7/24 eth 8/1 to 8/24
Building_A-FI-1(config-vlan-10)# tagged eth 2/1 to 2/2
Building_A-FI-1(config-vlan-10)# spanning-tree
Building_A-FI-1(config-vlan-10)# exit
Building_A-FI-1(config)# write memory
```

On Building B's Layer 2 Switch, perform the following:

```
Building_B-FI-1(config)# vlan 10 name testnet
Building_B-FI-1(config-vlan-10)# untagged eth 7/1 to 7/24 eth 8/1 to 8/24
Building_B-FI-1(config-vlan-10)# tagged eth 1/1 to 1/2 eth 2/1 to 2/2
Building_B-FI-1(config-vlan-10)# spanning-tree
Building_B-FI-1(config-vlan-10)# exit
Building_B-FI-1(config)# write memory
```

On Building C's Layer 2 Switch, perform the following:

```
Building_C-FI-1(config)# vlan 10 name testnet
Building_C-FI-1(config-vlan-10)# untagged eth 7/1 to 7/24 eth 8/1 to 8/24
Building_C-FI-1(config-vlan-10)# tagged eth 1/1 to 1/2
Building_C-FI-1(config-vlan-10)# spanning-tree
Building_C-FI-1(config-vlan-10)# exit
Building_C-FI-1(config)# write memory
```


Network Address Translation (NAT)

As the popularity of the Internet grew in the early 1990's and more devices were added to support the expansion, the fear of depleting the IP V4 address space became very real. Network Address Translation (NAT) RFC 1631 was developed to address this problem - slowing down the depletion of IP V4 public addresses. NAT provides the following benefits:

- The ability to isolate a private network address space (RFC 1597) from a public Internet address space.
- Translation of private network addresses to one or more publicly addressable IP addresses.
- Providing flexible network administration and allow internal networks to grow through private network address spaces.
- Enhancing security by hiding internal IP addresses from the public Internet and creating a stateful translation between the private and public address spaces.

There are really two forms of Network Address Translation. By itself, NAT translates one private internal IP address to one external public IP address. The one-for-one translation allows the internal address to be hidden

from the public Internet. One-for-one NAT translation is particularly useful when external traffic needs to access an internal host using a private IP address.

The NAT device is configured with a pool of pubic addresses that it uses for performing the one-for-one translation. As long as there are available addresses in the public pool, the one-for-one translation is performed. For installations that have many internal users on the private network, the pool of public addresses may not be sufficient to provide for a one-for-one translation, causing an overloading condition to occur. In this case, some of the internal users would be denied access when all of the public addresses are used.

NAT provides an alternative to the one-for-one translation method with a technology called Port Address Translation (PAT) that protects against NAT overloading. With PAT, many internal addresses can be translated to one or a few external public addresses. When a packet arrives at the NAT device, PAT uses the packet's internal private IP address and source protocol port number to create a new unique external packet.

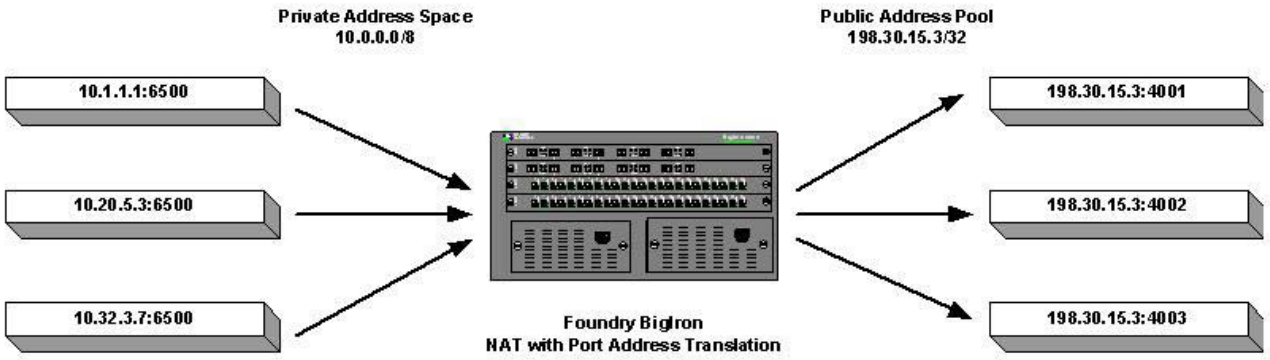


Figure 6. NAT With Port Address Translation

In the illustration above, three packets arrive at the Foundry NAT device with different source IP addresses but all are using the same TCP source port. With PAT turned on, the Foundry NAT device uses the packet's source IP address and its source TCP port to create a unique outbound external packet with the same pooled IP address but a unique source port for each packet. The NAT device creates an internal translation table to keep track of the translated internal packets to the unique external source ports. When the Foundry NAT device receives the return packet, the translation table will be used to match the returned packet back to the proper internal host and original port number.

NOTE: NAT can also be used to translate IP addresses between two private networks. One network must be designated as the inside private network and the other is designated the public network. For all NAT examples used in this document, the internal private network will be a private addresses as defined in RFC 1597 and the external public network will be valid Internet routable addresses.

Layering Security Using NAT

Although NAT is not considered as a firewall technology, some vendors offering NAT on their devices may refer to their implementations as a firewall feature. Many DSL and cable modem router vendors make this claim for securing the home environment due to the following NAT benefits:

- NAT will only allow connections that originated from the internal private network to come back through the NAT device. Connections that originate from the Public Internet are not allowed unless there is an explicit inbound mapping configured to allow the traffic in.
- Inbound Mapping allows the NAT device to redirect traffic originated from the Public Internet to computers on the internal private network. An example may be to allow inbound http port 80 traffic to a host with the internal IP address of 192.188.1.50.
- Packet filters can be used with NAT to provide access control - giving it firewall like rule capabilities.

Many installations use NAT to hide their internal IP addresses from the Public Internet and to redirect and map inbound traffic to specific hosts to prevent direct access from external hosts on the public Internet. NAT is especially useful for adding security to DMZ networks. Examples include implementing NAT on screened subnets that house specific extranet applications. An example is using NAT to separate front-end web servers from backend database servers. In addition to the security benefits, NAT is often used to expand the number of devices that can be supported on a network. By translating another network range into one or few external IP addresses, NAT expand networks that are running out of IP addresses.

Foundry's NAT Implementation

Foundry's NAT implementation supports two different types of NAT: Inside Source NAT and Inside Destination NAT. Inside Source NAT allows you to configure a Layer 3 device to translate IP addresses from the internal private network to one or more publicly routable external IP addresses. Inside Source NAT is used to allow traffic originating from the internal private network to flow out to the public Internet - providing your internal users with Internet access through a single or a few external IP addresses.

Inside Destination NAT is used to allow traffic originating from the public Internet to flow back into a particular host on the internal private network. The external application uses one of the global IP addresses (public IP address) defined on the NAT device as the destination IP address and NAT performs inbound mapping to translate and direct the traffic accordingly. An example is to allow POP3 traffic to come through the NAT device on the global IP address of 198.31.15.50 and have it translated and redirected to the mail server on the internal private network using the private IP address of 192.168.1.60.

NOTE: For more information on how to setup and use NAT on Foundry Layer 3 devices, refer to the *Foundry Enterprise Configuration and Management Guide* and the *Foundry Switch and Router Command Line Interface Reference.*

Configuring Inside Source NAT

Inside Source NAT is the most common NAT application used. Chances are, you are using this form of NAT to access the Internet from your corporate network or home ISP. Inside Source NAT requires two network interfaces: the inside private interface which controls the internal private network and the external public interface controlling access to the public network. There are two types of NAT translation supported on Inside Source NAT: Dynamic NAT and Static NAT.

Dynamic NAT uses a pool of public addresses called a global IP address pool to dynamically translate private internal IP addresses to. The NAT device randomly selects a free global IP address from the pool using a round robin approach to perform the translation. This is normally used when NAT is used to provide general Internet access for the corporate user population. The maximum number of global IP addresses that are allowed in the public address pool is 256 addresses.

Static NAT performs a one-for-one translation. A specific internal IP address is mapped to a specific external IP address. This feature is normally used to map a specific internal application or server to a specific external IP address to allow for proper return traffic.

NOTE: You can define both static and dynamic NAT on the same device. If they are both defined, static NAT will always take precedence over dynamic NAT.

Configuring Inside Source NAT consists of four steps:

- If Static NAT is used:
  - Configure the one-for-one static address mappings.
- If Dynamic NAT is used:
  - Configure the private internal network(s) that will use NAT to access the external public network. Standard or extended ACLs are used to define each range of private networks.
  - Configure the global IP address pool (public IP addresses) that will be used by NAT to translate the private network IP addresses. Each global IP address pool must be contiguous – there must be no breaks or gaps between the IP addresses in the pool. If needed, configure multiple consecutive pools to support your requirements.
  - Associate the private internal networks with the global IP address pools.
  - Enable the Port Address Translation feature if you will be overloading the global IP address pool. This will allow you to support many internal private IP addresses being translated to one or a few external global IP addresses.
- Enable Inside NAT on the interface connected to the internal private network.
- Enable Outside NAT on the interface connected to the external global IP address network.

Step 1: If Static NAT is used, define the one-for-one IP address translations. A single internal private IP address will be mapped to a single external IP address. Use the following command to define each one-for-one translation.

Syntax: [no] ip nat inside source static <private-ip> <global-ip>

The **inside source static** specifies a static NAT translation that will be performed from the private address to the public global address.

EXAMPLE:
This example will setup two servers on the internal private network to use two specific external global IP addresses. Server 10.64.254.10 will always be statically mapped to the public address 198.30.15.210 and server 10.64.254.11 will always be statically mapped to 198.31.15.211.

```
BigIron(config)# ip nat inside source static 10.64.254.10 198.30.15.210
BigIron(config)# ip nat inside source static 10.64.254.11 198.30.15.211
```

Step 2: If Dynamic NAT is used, the following configuration steps are required.

Step 2a: Define the private internal address range that will be translated by the NAT device. A standard or extended ACL is used to define the internal address range. The access list identifier must be a number. Text names are not supported by NAT.

Syntax: [no] access-list <num> permit <private-ip-range>

EXAMPLE:
This example defines the entire internal private network 10.0.0.0/8 as the range to be translated by the NAT device. This ensures that all internal devices can access the public Internet.

```
BigIron(config)# access-list 50 permit 10.0.0.0/8
```

Step 2b: Configure the global IP address pool to support the translation of internal private addresses. Remember that the global IP address pool must be consecutive with no gaps in the numbering scheme.

Syntax: [no] ip nat pool <pool-name> <start-ip> <end-ip> netmask <ip-mask> | prefix-length <length> [type match-host | rotary]

The <pool-name> parameter specifies the pool name. The name can be up to 255 characters long and can contain special characters and internal blanks. If you use internal blanks, you must use quotation marks around the entire name.

The <start-ip> parameter specifies the IP address at the beginning of the pool range. Specify the lowest-numbered IP address in the range.

The <end-ip> parameter specifies the IP address at the end of the pool range. Specify the highest-numbered IP address in the range.

The **netmask** <ip-mask> | **prefix-length** <length> parameter specifies a classical sub-net mask (example: **netmask** 255.255.255.0) or the length of a Classless Interdomain Routing prefix (example: **prefix-length 24**).

The **type match-host | rotary** parameter specifies the method the software uses to assign the host portion of the translated address.

- **match-host** - The software uses the same host address as the untranslated address. For example, if the untranslated address is 192.2.4.69 and the host portion of the address is 69, the translated address also uses the host address 69. This method results in the translated addresses always having the same host addresses as their untranslated counterparts.
- **rotary** - The software assigns a host address from 1 - 254, beginning with 1 for the first translated address. This is the default.

EXAMPLE:
This example configures a global IP address pool called “ExtIPs” and included 10 IP addresses that will be rotated to support the NAT translation.

```
BigIron(config)# ip nat pool ExtIPs 192.30.15.30 198.30.15.39 prefix-length 24
```

Step 2c: Associate the private addresses (ACLs) with the global IP address pool(s) and enable Port Address Translation if the global IP address pool will be overloaded.

Syntax: [no] ip nat inside source list <acl-id> pool <pool-name> [overload]

The **inside source** parameter specifies the direction of the translation – from the private addresses to the global addresses (Internet addresses).

The **list** <acl-id> parameter specifies a standard or extended ACL.

EXAMPLE:
This example associates the ACL that defined the internal private address range with the global IP address pool. Since there are more internal addresses requiring NAT translation than available global IP addresses in the pool, overloading will be performed and Port Address Translation will be enabled.

```
BigIron(config)# ip nat inside source list 50 pool ExtIPs overload
```

Step 3: Enable Inside NAT on the interface connected to the internal private network.

Syntax: [no] ip nat inside

EXAMPLE:
This example will enable Inside NAT on the router interface connecting to the internal private 10.64.254.0/24 network.

```
BigIron(config)# interface ethernet 5/1
BigIron(config-if-5/1)# ip nat inside
```

Step 4: Enable Outside NAT on the interface connected to the external global IP address network.

Syntax: [no] ip nat outside

EXAMPLE:
This example will enable Outside NAT on the router interface connecting to the external public 198.30.15.0/24 network.

```
BigIron(config)# interface ethernet 8/24
BigIron(config-if-8/24)# ip nat outside
BigIron(config-if-8/24)# write memory
```

Widget-Works.COM - Inside Source NAT Example
Widget-Works.COM has a high security network that is used for special R&D testing. This network is self-contained and occasionally needs access to the public Internet. No other traffic is allowed in or out of this high security subnet. The IT Team has decided to setup a private subnet using NAT to house this R&D network – all hosts will be hidden from the 10.0.0.0/8 corporate network.

The internal network will be given a private address space that is separated from the existing corporate network's 10.0.0.0/8 address space. The 10.0.0.0/8 network will be considered the public external network in this example and the private R&D subnet will be given the IP subnet range of 192.168.100.0/24. A new Foundry Layer 3

switch was purchased to route between these two subnets and interface Ethernet 1 is the R&D 192.168.100.0/24 subnet and Ethernet 24 is the connection to the 10.48.2.0/24 corporate subnet (the public network).

The IT department has decided to NAT all 192.168.100.0/24 addresses to one 10.48.2.0/24 external address. Port Address Translation will be turned on to support overloading.

The configuration on the Layer 3 switch would resemble:

```
R&D_RTR(config)# access-list 10 permit 192.168.100.0/24
R&D_RTR(config)# ip nat pool ExtIP 10.48.2.240 10.48.2.240 netmask 255.255.255.0
R&D_RTR(config)# ip nat inside source list 10 pool ExtIP overload
R&D_RTR(config)# int eth 1
R&D_RTR(config-if-1)# ip nat inside
R&D_RTR(config-if-1)# int eth 24
R&D_RTR(config-if-24)# ip nat outside
R&D_RTR(config-if-24)# write memory
```

Configuring Inside Destination NAT

Inside Destination NAT is used to allow traffic originating from the public Internet to be translated and redirected to a specific internal private IP address or a pool of internal private IP addresses. Inside Destination NAT is commonly used to allow a public Internet application to send information to a specific host placed behind the NAT device – http or email applications are good examples that make use of Inside Destination NAT. Inside Destination NAT supports both Dynamic NAT and Static NAT.

- Dynamic NAT maps global IP addresses to private addresses defined in a pool. Dynamic NAT translates the global IP addresses to the private addresses using a round robin technique.

- Static NAT performs a one-for-one translation for each global IP address to one specific private address. Traffic can be redirected to a specific internal IP address based on TCP or UDP port information to support specific application requirements.

NOTE: You can configure both dynamic and static Inside Destination NAT on the same Foundry device. When you configure both types of NAT, static NAT takes precedence over dynamic NAT. If you configure a static NAT translation for an address, the device always uses that translation instead of creating a dynamic one.

Configuring Inside Destination NAT consists of four steps:

- If Static NAT will be used:
  - Configure the one-for-one static address mappings.
- If Dynamic NAT will be used:
  - Configure the public IP address range that will use NAT to access the private internal network. Standard or extended ACLs are used to define each range of public IP addresses.
  - Configure the private IP address pool that will be used by NAT to translate the public network IP addresses. Each private IP address pool must be contiguous – there must be no breaks or gaps between the IP addresses in the pool. If needed, configure multiple consecutive pools to support your requirements.
  - Associate the range of public addresses with the private IP address pool.
- Enable Inside Destination NAT on the interface connected to the internal private network.
- Enable Outside NAT on the interface connected to the external global IP address network.

Step 1: If Static NAT is used, define the one-for-one address translation pairs between the external global IP address and the internal private IP address.

Syntax: [no] ip nat inside destination static <global-ip> [<global-port>] [tcp | upd] <private-ip> [<private-port>]

The **inside destination** parameter configures NAT to perform an inbound translation from the public network to the private network.

EXAMPLE:
This example configures a one-for-one NAT translation for global IP address 198.30.15.36 to a private internal IP address of 10.64.254.36. The global IP address will be the destination address for an email server that has been advertised on the External DNS server. When external email is received on this IP address, it will be redirected back to an email server (198.31.15.36) on the inside private network.

The command can be used with or without TCP or UDP port numbers. Using TCP or UPD port numbers will allow you to control the specific traffic to be redirected.

This first example doesn't use any port numbers – all traffic is translated and redirected back to the internal host 10.64.254.36 from 198.31.15.36.

```
BigIron(config)# ip nat inside destination static 198.30.15.36 10.64.254.36
```

This example uses port numbers to redirect only specific traffic back to the internal host 10.64.254.36 from the external IP 198.31.15.36. All POP3 TCP port 110 traffic is NAT translated from 198.31.15.36 and redirected to TCP port 110 on the internal mail server 19.64.254.36.

```
BigIron(config)# ip nat inside destination static 198.30.15.36 110 tcp 10.64.254.36
110
```

Step 2: If Inside Dynamic NAT is used, the following configuration steps are required.

Step 2a: Define the external global IP address range that will be translated by the NAT device. A standard or extended ACL is used to define the public address range. The access list identifier must be a number. Text names are not supported by NAT.

Syntax: [no] access-list <num> permit <private-ip-range>

EXAMPLE:
This example defines the external global IP address range to be translated by the NAT device. It will allow 6 external global IP addresses starting from 198.30.15.201/29 to 198.30.15.206/29 to be NAT translated and redirected to a pool of internal IP addresses.

```
BigIron(config)# access-list 5 permit 198.31.15.200/29
```

Step 2b: Configure the private internal IP address pool to support the translation of external global IP addresses. Remember that the private IP address pool must be consecutive IP addresses with no gaps on the numbering scheme.

Syntax: [no] ip nat pool <pool-name> <start-ip> <end-ip> netmask <ip-mask> | prefix-length <length> [type match-host | rotary]

The <pool-name> parameter specifies the pool name. The name can be up to 255 characters long and can contain special characters and internal blanks. If you use internal blanks, you must use quotation marks around the entire name.

The <start-ip> parameter specifies the IP address at the beginning of the pool range. Specify the lowest-numbered IP address in the range.

The <end-ip> parameter specifies the IP address at the end of the pool range. Specify the highest-numbered IP address in the range.

The **netmask** <ip-mask> | **prefix-length** <length> parameter specifies a classical sub-net mask (example: **netmask** 255.255.255.0) or the length of a Classless Interdomain Routing prefix (example: **prefix-length 24**).

The **type match-host | rotary** parameter specifies the method the software uses to assign the host portion of the translated address.

- **match-host** - The software uses the same host address as the untranslated address. For example, if the untranslated address is 192.2.4.69 and the host portion of the address is 69, the translated address also uses the host address 69. This method results in the translated addresses always having the same host addresses as their untranslated counterparts.
- **rotary** - The software assigns a host address from 1 - 254, beginning with 1 for the first translated address. This is the default.

EXAMPLE:
This example configures a private IP address pool called “PrivateIPs” and included 6 ip addresses that will be rotated to support the NAT translation. Notice that this example matched the number of internal IP addresses to the number of external IP addresses; the host octet was also kept in the same range (201 – 206). In this example, host matching will be performed to keep the host IP the same when the NAT translation is performed.

```
BigIron(config)# ip nat pool PrivateIPs 10.64.254.201 10.64.254.206 prefix-length 24
match-host
```

Step 2c: Associate the public global IP addresses (ACLs) with the private IP address pool(s) and turn on Port Address Translation if overloading is present.

Syntax: [no] ip nat inside destination list <acl-id> pool <pool-name> [overload]

The **inside destination** parameter specifies the direction of the NAT translation – from the public global IP addresses to the internal private addresses.

The **list** <acl-id> parameter specifies a standard or extended ACL.

EXAMPLE:
This example associates the ACL that defined the external global IP address range with the internal private IP address pool. Since there are the same number of external IP addresses and internal IP addresses in the translation, overloading will not be used.

```
BigIron(config)# ip nat inside destination list 5 pool PrivateIPs
```

Step 3: Enable Inside Destination NAT on the interface connected to the internal private network.

Syntax: [no] ip nat inside

EXAMPLE:
This example will enable Inside NAT on the router interface connecting to the internal private 10.64.254.0/24 network.

```
BigIron(config)# interface ethernet 5/1
BigIron(config-if-5/1)# ip nat inside
```

Step 4: Enable Inside NAT on the interface connected to the internal private network.

Syntax: [no] ip nat outside

EXAMPLE:
This example will enable Outside NAT on the router interface connecting to the external public 198.30.15.0/24 network.

```
BigIron(config)# interface ethernet 8/24
BigIron(config-if-8/24)# ip nat outside
BigIron(config-if-8/24)# write memory
```

Widget-Works.COM - Inside Destination NAT Example
To expand on Widget-Works.COM's high security R&D network that was setup in the previous Widget-Works.COM example, we will add Inside Destination NAT functionality. After a few weeks of operating in the private R&D subnet, some scientists realized that an external server located in Building B's R&D server farm needed access to one of the hidden R&D servers in the private 192.168.100.0/24 subnet.

The IT department has decided to allow access to this one hidden server from the corporate network. Inside Destination NAT will be setup to perform a one-for-one translation. To publish this hidden server to the corporate network (external public network), a new DNS entry was created:

R&D198.widgetworks.com 10.48.2.241

When external hosts want to communicate with the hidden R&D server on the private 192.168.100.0/24 subnet, they will reference it by the R&D198.widgetworks.com name. The hidden R&D server's IP address is 192.168.100.10. Static NAT will be used to define a one-for-one NAT translation and no port restrictions will be used to allow all traffic between the two servers.

The configuration on the Layer 3 switch would resemble:

```
R&D_RTR(config)# ip nat inside destination static 10.48.2.241 192.168.100.10
R&D_RTR(config)# int eth 1
R&D_RTR(config-if-1)# ip nat inside
R&D_RTR(config-if-1)# int eth 24
R&D_RTR(config-if-24)# ip nat outside
R&D_RTR(config-if-24)# write memory
```

Port Security And Port Authentication

Network security can be greatly enhanced by blocking unauthorized devices or users from using the internal LAN. As part of the "Defense in Depth" security concept, security is applied throughout the network at each of the various layers. Security should always be applied according to your Corporate Security Policy and each area of the network will require different security defenses to balance usability and security.

One of the strongest defenses that can be implemented on the Internal Network was shown through Access Restrictions and Containment. By using inbound access control lists at strategic ingress points into the corporate backbone and ingress points into various subnet switches, unauthorized access can be blocked. By implementing anti-spoofing ACLs for each subnet, we can ensure that DoS and Worm attacks that rely on spoofed source IP addresses are limited in what they can do and where they can spread. Containment is achieved through careful network design and layering the necessary Foundry IronShield Security features.

Port Security and 802.1x Port Authentication expands on "Defense in Depth" security design and further protects your internal network. With these features, enterprises can now control which devices and users are allowed onto the LAN, and defend against many types of attacks that rely on unprotected network access.

MAC Address & ARP Spoofing

MAC addresses and the ARP protocol are two required components of the TCP/IP protocol. MAC addresses are universally unique hardware addresses that are associated with each Ethernet device and are used to identify the device on the network. ARP is used to translate an IP Address to a MAC address and is used in setting up transmission sessions between hosts. Unfortunately, hackers have engineered many DoS, Worm, and malicious code to take advantage of these two functions. Through MAC Address Spoofing and ARP Spoofing, the following attack activities may occur on networks:

Man-in-the-Middle (MIM)	A MIM attack occurs when an intruder is able to slip their host in between two valid devices. The intruder redirects traffic from the victim host to their host so the traffic can be stolen, manipulated, or sniffed. In many cases, the intruder redirects traffic back to the original target host once they have successfully inserted their host in the middle of the traffic stream. This allows the hacker to intercept the traffic and keep the original transmission going – everything seems normal to the two original hosts. There are many variations to this attack as well as uses. See the MIM Example below for more information on how this works.
MAC Cloning	MAC Cloning is when the default hardware MAC address of the device is changed to allow the device to impersonate the MAC address of another device. 802.11x wireless solutions depending on MAC Address filtering are prone to this type of attack.

Session Hijacking	Session Hijacking is similar to the MIM attack where the intruder takes over an established connection between two hosts. In this case, the intruder may not reestablish communications to the original target host – simply hijacking the session. An example is taking over an established Telnet session by tricking the victim device into believing that the intruder's host is the legitimate target host.
Router Gateway Hijacking/ Switch Sniffing	Router Gateway Hijacking is a method used to sniff a switched network or to steal information from hosts on the switched network. Under this attack, the devices on a particular subnet are tricked into believing that the hacker's device is the default gateway. The hacker floods the Layer 2 network with spoofed ARP Replies claiming to be the default gateway's MAC address. When these devices hear the spoofed ARP Reply, they insert it into their local ARP caches and forward all external traffic to the hacker's device. The hacker in turn enables IP routing on their device to route the traffic back to the router gateway address once they are done sniffing the traffic. No one suspects what is happening and the switch is now "sniffable".
Broadcast ARP Attack	The Broadcast ARP attack is a variation of the Router Gateway Hijacking attack and is used to sniff switched traffic on a Layer 2 device. This attack floods the network with spoofed ARP Reply packets redirecting the router's default gateway MAC address to the Broadcast MAC Address (FF:FF:FF:FF:FF:FF). The devices on the network hear the spoofed ARP Reply and update their local ARP cache. All external traffic that would normally go to the default gateway is now sent to the broadcast address and sniffing is now possible on the entire switch.
ARP Flooding DoS Attack	An ARP Flood DoS Attack is used to flood the Layer 2 network with spoofed non-existing MAC addresses. The goal is to trick all the hosts into updating the local ARP caches with non-existing MAC addresses. This will cause the hosts to send information to non-existing devices and the packets will be dropped. The attacker can also fake their host's MAC address with MAC Cloning to hide their tracks after the attack has been launched.

Example - Man-in-the-Middle Attack

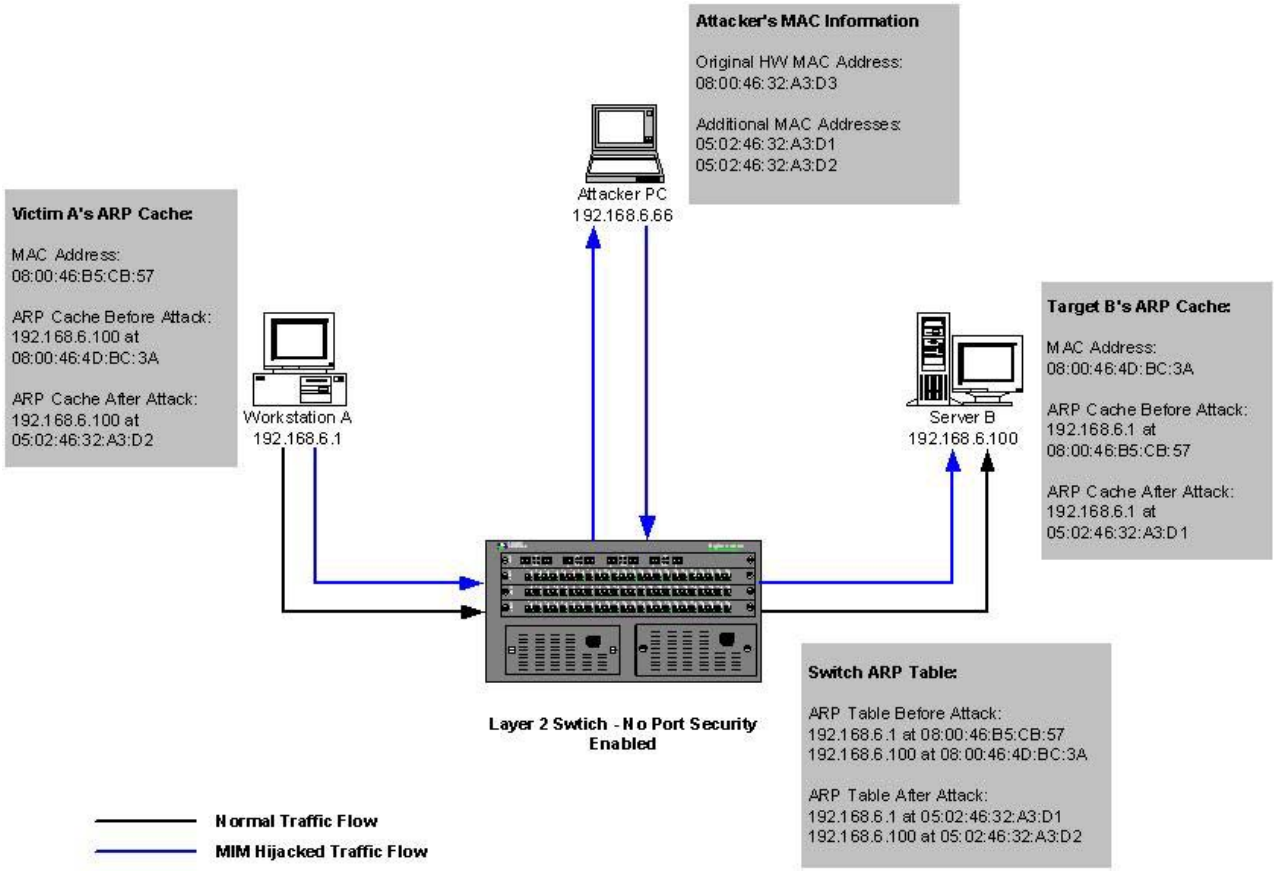


Figure 7. Man-in-the-Middle Attack

In this example Workstation A is using Telnet to Server B. The attacker will use his PC to perform a MIM Session Hijack attack to intercept and steal the session. Through the use of PING and trace route, the attacker learned the IP address and the MAC address of both the Victim (Workstation A) and the Target (Server B). Using multiple NIC cards, he prepares his Attack PC with additional MAC addresses for the redirection of traffic from the Victim and Target hosts.

Once his Attack PC is prepared with the additional MAC addresses, the attacker sends spoofed ARP Replies to the Victim and the Target. The spoofed ARP Replies are accepted by both the Victim and Target - they overwrite the original ARP entries that they had for each other with the spoofed ARP entries supplied by the Attacker PC. With their ARP caches poisoned, they begin to transmit traffic to the Attack PC and the attacker has successfully hijacked the Telnet session between the Victim and the Target host.

In order to avoid suspicion, the attacker can also configure the Attack PC to relay the information back to the Victim once the hijack is successful. If this is done, the Victim host will see a temporary break in communication when the session is hijacked and then transmission will resume as normal. There are several versions of how this type of attack can be carried out, but most rely on ARP Reply spoofing to poison the ARP cache of the Victim host.

ARP Reply Spoofing

In order to perform Man-in-the-Middle type attacks, the intruder must poison the victim's local ARP cache to trick it into sending its traffic to the Attack PC. A spoofed ARP Reply will contain a crafted Source MAC address or IP address in the Source ARP address field. There are different ARP Reply spoofing techniques and tools that can be used and it depends on what the attacker is trying to achieve with the spoofed ARP Reply.

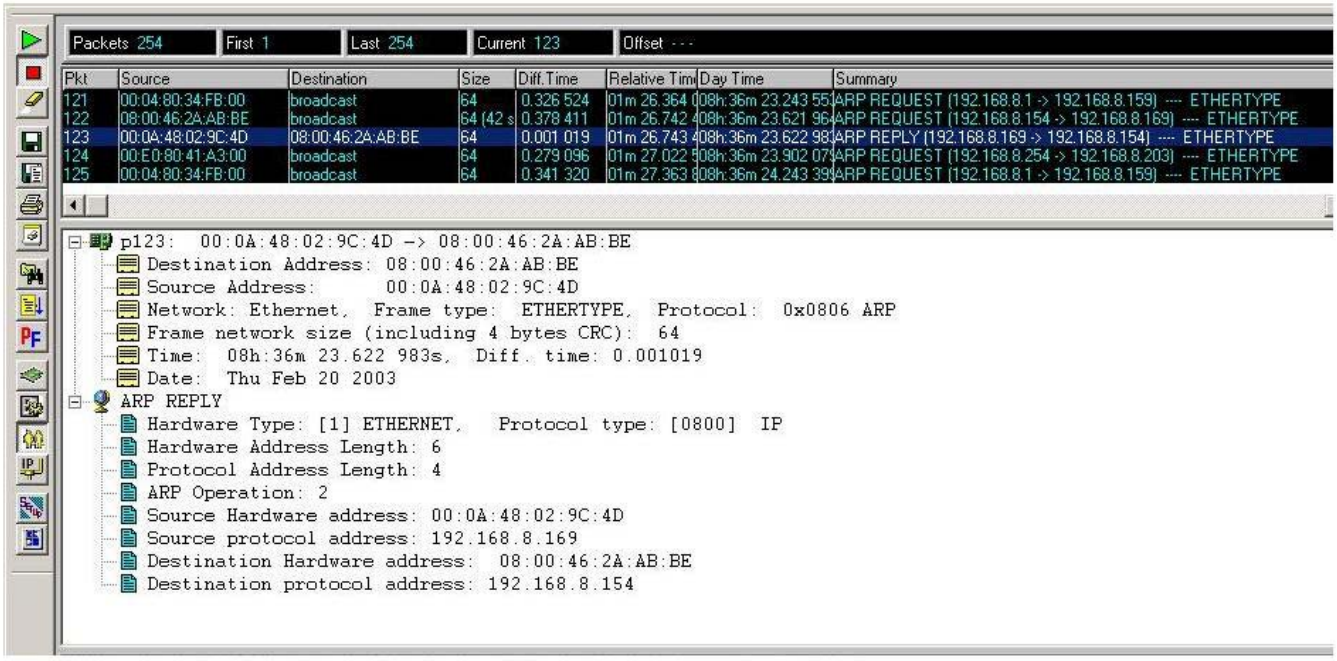


Figure 8. ARP Reply

Taking a look at a sample sniffer trace for an ARP Reply, the Source Hardware Address is normally set to the originating device's MAC address. Depending on the ARP Reply spoof being used, some hackers will use tools to craft the Source Hardware Address or the Source IP Address in ARP Replies to direct traffic to another device by poisoning the victim's ARP cache.

Defending Against MAC Address & ARP Spoofing

As the simplified MIM and Session Hijacking example shows, attacks that rely on MAC Address and ARP Spoofing are fairly simple to execute. There are many precompiled utilities that fully automate the attacks – making it very easy for someone to download the utilities and perform this type of attack on your internal networks. Although it is difficult to defend against all variations of these attacks, some of the more common defenses include:

- Using encrypted sessions such as SSH, SSL, and IPSec tunnels to replace plain text communications.
- Using detection systems such as Arpwatch – a UNIX application that resides on each subnet watching ARP responses and building a table to catch rapid MAC address changes.
- Enabling Foundry's Port Security to regulate MAC addresses and to prevent MAC Address Spoofing.
- Enabling Foundry's 802.1x Port Authentication to restrict non-authorized users.
- Applying IP Anti-Spoofing ACLs for each subnet. For key servers with static IP Addresses, inbound IP Anti-Spoofing ACLs can be applied at the Layer 2 port level to prevent an infected server from sending attack packets with spoofed source IP addresses.

- Creating Visitor and Guest VLANs to separate corporate data traffic from casual user traffic – conference rooms, guest offices, guest lobbies, etc are areas that can be used to attack the corporate network.
- Disabling all unused network ports on both Layer 2 and Layer 3 devices to prevent unauthorized access. Some attacks require the Attack PC to have multiple NIC cards.

Port Security - Restricting Source MAC Addresses

Foundry's Port Security feature is used to restrict inbound Source MAC addresses. The Source MAC address can be manually set or dynamically learned as each device is brought up on the network. One or more MAC addresses can be locked per physical port to support downstream hubs and switches. In ideal conditions, each Layer 2 network will have one device attached to each switch port. This gives the network administrator a simple way to enforce the first learned MAC address on each Layer 2 port and block all subsequent MAC addresses – potentially blocking attacks that rely on source MAC address spoofing.

Depending on the action defined, the offending packet with the invalid Source MAC address can either be dropped or the port can be disabled. Port Security is only available on Ethernet ports and permits a maximum of 64 Source MAC addresses to be configured per port. The total number of maximum Source MAC addresses permitted per device is 1024, 2048, or 4096 depending on the device's physical memory size.

By default, Port Security is disabled on Foundry devices. Port Security can be enabled for all interfaces by configuring it at the global device level or it can be enabled on specific ports at the interface level. Enabling Port Security at the interface level is preferred. Enabling it at the global level may accidentally lock the device down by limiting the number of valid MAC addresses on the device's uplink ports. Configuring Port Security requires the following steps.

Step 1: Enabling Port Security – Can be enabled at the global or interface level (Interface level preferred).

Syntax: port security
Syntax: [no] enable

```
BigIron(config)# int e10 to 20
BigIron(config-mif-10-15)# port security
BigIron(config-port-security-mif-10-15)# enable
```

Step 2: Setting the maximum number of secure Source MAC addresses for an interface. The default is 1 secure MAC address per port and the maximum of 64 MAC addresses per port can be set. The device maximum depends on available memory.

Syntax: maximum <number-of-addresses>

```
BigIron(config)# int e 10
BigIron(config-if-e100-10)# port security
BigIron(config-if-e100-10)# maximum 10
```

Step 3: Setting the port security age timer. By default, the learned Source MAC addresses stay secure indefinitely. The learned MAC addresses can be set to age out after a specified amount of time set in minutes. The default is 0 which means that the secure MAC addresses will never be aged out.

Syntax: [no] age <minutes>

```
BigIron(config)# port security
BigIron(config-port-security)# age 10
```

Step 4: Specifying the secure Source MAC addresses. By default, Port Security automatically learns the MAC Addresses on each port. If required, the MAC address can be manually specified.

Syntax: [no] secure <mac-address>

```
BigIron(config)# int e 7/11
BigIron(config-if-e100-7/11)# port security
BigIron(config-port-security-e100-7/11)# secure 0050.DA18.747C
```

Step 5: Configuring the device to automatically save the Secure MAC addresses to the startup-config file. By default, secure MAC addresses are not saved to the device's startup-config file. You can configure the device to save the learned MAC addresses to the startup-config file every 15 to 1440 minutes. Enabling this option will allow the device to protect the same MAC addresses after it is restarted. Otherwise, the device will learn the secure MAC addresses after each restart.

Syntax: [no] autosave <minutes>

```
BigIron(config)# port security
BigIron(config-port-security)# autosave 30
```

Step 6: Specifying the security violation action. A security violation occurs when a device is plugged into a port that already has the maximum number of secure MAC addresses learned or a static secure MAC address has been set. One of two actions can be taken when a violation occurs: the violating address is dropped or the port is disabled for a specified amount of time.

Syntax: violation restrict (drop violating packets)

```
BigIron(config)# int e 7/11
BigIron(config-if-e100-7/11)# port security
BigIron(config-port-security-e100-7/11)# violation restrict
```

Syntax: violation shutdown <minutes> (0 – 1440 minutes: 0 is permanent disabling of the port)

```
BigIron(config)# int e 7/11
BigIron(config-if-e100-7/11)# port security
BigIron(config-port-security-e100-7/11)# violation shutdown 60
```

NOTE: Depending on your version of hardware and software revision, some port security features may not be available. For complete information on MAC Address Locking, refer to the *Foundry Switch and Router Command Line Interface Reference* and *Foundry Security Guide*.

Other Port Security Commands

Other important Port Security commands that help in managing port security are:

Syntax: show port security autosave	Display Autosaved MAC Addresses
Syntax: show port security <module> | <portnum>	Display Port Security settings
Syntax: show port security mac	Display Secure MAC Addresses
Syntax: show port security statistics <portnum>	Display Port Security statistics by port
Syntax: show port security statistics <module>	Display Port Security statistics by module

EXAMPLE – Port Security MAC Lock

Widget-Works.COM's Corporate Security Policy has dictated that MAC Address protection must be applied on all Layer 2 switch ports to protect the network from spoofed source MAC addresses and possible ARP Reply attacks. By implementing port security to one valid MAC address, malicious code and attacks that rely on MAC Address spoofing (where the source MAC address is changed on the attack packets) can be prevented. The network has been designed to support one device per Layer 2 switch port so there are very little downstream hubs or switches that are used – these will be handled on an as needed basis.

Each 10/100 Ethernet port will be limited to one valid Source MAC address that will be learned automatically when the device is initially brought online. The learned addresses are not timed out while the device is connected to the port. To make sure there are no new devices brought online after a switch is restarted, Widget-Works.COM will configure the switch to automatically save the learned MAC addresses to the startup-conf file every 60 minutes. The security policy calls for permanently disabling the port when violations are encountered.

The following commands are used to configure each Layer 2 switch's 10/100 ports. File servers using Gigabit ports are also configured in the same way, but all switch Gigabit ***uplink ports are not configured*** with port security.

To enable Port Security for all the 10/100 ports on blades 2, 3 and 4, use the interface range command to select the range of ports. This will allow you to apply the Port Security commands to a group of ports:

```
B_R&D-FI1(config)# int eth 2/1 to 2/24
B_R&D-FI1(config-mif-2/1-2/24)# port security
B_R&D-FI1(config-mif-2/1-2/24)# enable
B_R&D-FI1(config-mif-2/1-2/24)# autosave 60
B_R&D-FI1(config-mif-2/1-2/24)# violation shutdown 0
B_R&D-FI1(config-mif-2/1-2/24)# int eth 3/1 to 3/24
B_R&D-FI1(config-mif-3/1-3/24)# port security
B_R&D-FI1(config-mif-3/1-3/24)# enable
B_R&D-FI1(config-mif-3/1-3/24)# autosave 60
B_R&D-FI1(config-mif-3/1-3/24)# violation shutdown 0
B_R&D-FI1(config-mif-4/1-4/24)# port security
B_R&D-FI1(config-mif-4/1-4/24)# enable
B_R&D-FI1(config-mif-4/1-4/24)# autosave 60
B_R&D-FI1(config-mif-4/1-4/24)# violation shutdown 0
```

Omitting the Maximum number of MAC addresses defaults the learned MAC address to 1.
Omitting the Port Security Age Timer defaults the learned MAC addresses to indefinite.

Defending Against Unauthorized Access

One of the most effective ways of preventing unauthorized users from using the corporate LAN to snoop, hack, or steal is to proactively stop them before they can gain access. Foundry's IronShield Security provides a standard way to authenticate users with 802.1x Port Authentication.

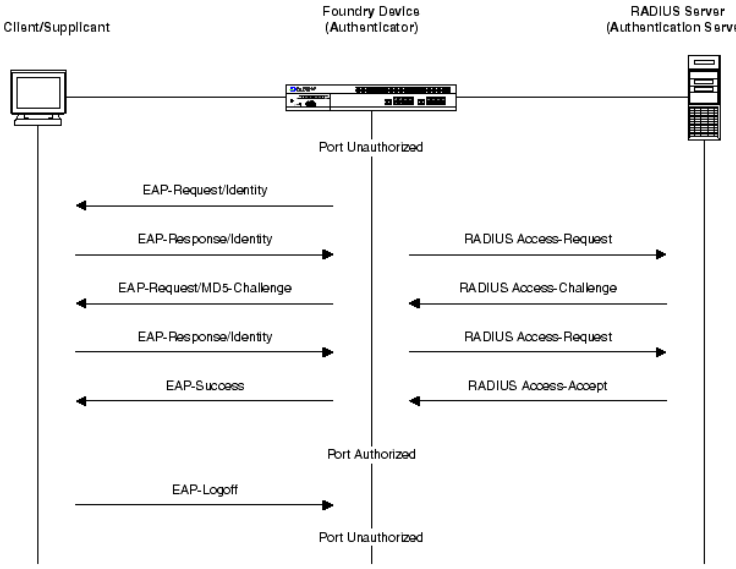
How 802.1x Works

Foundry's implementation of 802.1x is based on a series of standards:

- RFC 2284 PPP Extensible Authentication Protocol (EAP)
- RFC 2865 Remote Authentication Dial In User Service (RADIUS)
- RFC 2869 RADIUS Extensions

There are three 802.1x components that make up the security defense: Client/Supplicant, Authenticator, Authentication Server.

Client/Supplicant	The client, or supplicant, is the device that needs authenticating to the network. It supplies the username/password information to the Authenticator.
Authenticator	The Authenticator is the Foundry device performing the 802.1x port security – controlling access to the network. It receives the username/password information from the client and passes it onto the Authentication Server and performs the necessary block or permit action based on the results from the Authentication Server.
Authentication Server	The Authentication Server validates the username/password information from the Client and specifies whether or not access is granted. The Authentication Server may also specify optional parameters to control VLAN access. Foundry's 802.1x implementation currently supports RADIUS Authentication Servers.



802.1x uses the Extensible Authentication Protocol (EAP) and EAP Over LAN (EAPOL) to securely encapsulate the communications between the Client and Authenticator. The Authenticator uses RADIUS to communicate with the Authentication Server.

Before the Client is fully authenticated, the port is set to the ***uncontrolled*** (unauthorized) state and only allows EAPOL traffic between the client and the Authentication Server. All other normal data traffic is blocked. When the client authentication is complete and access is granted, the port is set in the ***controlled*** (authorized) state to grant full network access.

Figure 9. 802.1x Port Authentication Process

If a non-802.1x Client is connected to an 802.1x protected port, the Client will not recognize the EAPOL polling traffic from the Authenticator and authentication will fail. The client will not be granted any network access. If an 802.1x enabled Client is connected to a non-802.1x port, it will attempt to send an EAP start frame to the Foundry device. When the device doesn't respond to the EAP packet, the Client considers the port to be authorized and starts sending normal traffic.

By default, Foundry devices place all ports in the authorized state, allowing full network access. When 802.1 Port Authentication security is implemented, all ports are switched to the unauthorized state to prevent full network access.

Configuring 802.1x Port Authentication

To configure a Foundry device to support 802.1x Port Authentication, the following procedures are required:

- Configure the Foundry device (Authenticator) to interact with the Authentication Server
- Configure the Foundry device to act as the Authenticator
- Configure the Foundry device's interaction with the Client device (optional step)

Step 1: Configure the Foundry device to use RADIUS for authenticating 802.1x security.

Syntax: [no] aaa authentication dot1x default <radius | none>

```
BigIron(config)# aaa authentication dot1x default radius
```

Configure the device to use one or multiple RADIUS authentication servers. Set the authentication and accounting port numbers to match the RADIUS server's settings and specify the **secret key** to authenticate to the RADIUS server. The secret key string must be identical to the secret key string used on the authentication server.

Syntax: radius-server host <ip-addr> | <server-name> [auth-port <number> acct-port <number> default key <string> dot1x]

```
BigIron(config)# radius-server host 209.157.22.99 auth-port 1812 acct-port 1813
default key radpassword dot1x
```

Step 2: Enable 802.1x security for the Foundry device. This enables the device to act as an 802.1x Authenticator. When 802.1x is enabled, the default authorized mode is turned into unauthorized mode.

Syntax: [no] dot1x-enable

```
BigIron(config)# dot1x-enable
```

Example 1: To enable 802.1x port security for all interfaces on the device.

```
BigIron(config-dot1x)# enable all
```

Example 2: To configure 802.1x for individual ports, you can use the "enable" command with the port number. A range can also be specified.

```
BigIron(config-dot1x)# enable Ethernet 2/1 to 2/24
BigIron(config-dot1x)# enable Ethernet 3/1 to 3/12
BigIron(config-dot1x)# write memory
```

Step 3: For all interfaces using 802.1x authentication, enable the control mode to "force-authorized", "force-unauthorized", or "auto". Auto leaves the port in unauthorized mode until the RADIUS server validates the authentication.

```
BigIron(config)# interface e 3/1
BigIron(config-if-3/1)# dot1x port-control auto
```

The switch is now enabled for 802.1x Port Authentication. Make sure the RADIUS server is properly configured to authenticate each user.

NOTE: For more information on MAC Address Locking and 802.1x authentication, refer to the *Foundry Switch and Router Command Line Interface Reference* and *Foundry Security Guide*.

EXAMPLE – 802.1x Port Authentication

Every building that Widget-Works.COM occupies has commonly shared areas. These include conference rooms, reception areas, and common office space that visitors can use. The IT department created two separate networks using multiple network jacks in each shared area. The primary network jack is part of the default corporate network and is protected in each of the shared areas with 802.1x Port Authentication – it is reserved for employees only. The secondary network jack is setup for guest access with limited access to the Public Internet.

For the switches supporting the common shared areas, Widget-Works.COM's corporate security policy dictates 802.1x Port Security for all primary network ports. To configure 802.1x Port Authentication on these switches, the following commands are used:

Specify the RADIUS Server information and enable 802.1x Port Authentication for the necessary interfaces.

```
B_Guest-FES-1(config)# aaa authentication dot1x default radius
B_Guest-FES-1(config)# radius-server host 10.32.6.50 auth-port 1812 acct-port 1813
default key MySecretPWD dot1x
B_Guest-FES-1(config)# dot1x-enable
B_Guest-FES-1(config-dot1x)# enable Ethernet 1 to 5
B_Guest-FES-1(config-dot1x)# enable Ethernet 10 to 18
B_Guest-FES-1(config-dot1x)# write memory
```

Enable the control mode to "auto" for each interface enabled with 802.1x. Auto leaves the port in unauthorized mode until the RADIUS server validates the authentication. Repeat for all 802.1x ports.

```
B_Guest-FES-1(config)# interface e 1
B_Guest-FES-1(config-if-1)# dot1x port-control auto

B_Guest-FES-1(config)# interface e 2
B_Guest-FES-1(config-if-2)# dot1x port-control auto
```

NOTE: For a complete example of how to setup Foundry's 802.1x Port Authentication with Microsoft's IAS and Active Directory, refer to the *White Paper: 802.1x Port Authentication With Microsoft's Active Directory* document.

Appendix A - Foundry IronShield Security Enhancements

The following tables outline the standard IronShield Security features that can be used to enhance your security defenses at each layer of the corporate network. Please refer to the *Foundry Switch and Router Command Line Interface Reference and Foundry Enterprise Configuration and Management Guide* for more information on each command.

Device Protection

IronShield Standard Security Features For Hardening Network Devices		
Warning Banners	TACACS Authentication	Multi-Level AAA and Authentication Support
Support and Encryption of Strong Passwords	TACACS+ Authentication, Authorization, Accounting	Remote Management Restriction Support – ACLs, Built-in Commands, VLAN, Management VLANs
Automated Configuration Management With IronView Network Manager	RADIUS Authentication, Authorization, Accounting	High-performance Anti-Spoofing ACLs
Password Protection For: Privilege EXEC, Privilege Levels, Local Users, Remote Management	Secure Shell (SSH) and Secure Copy (SCP)	DoS Attack Prevention - Smurf, SYN Flood, Broadcast Limiting, ARP Request Limiting
Remote Management and Console Timeout	SNMP v1, v2c and v3 support, SNMP Views support	Fragmentation Attack Prevention
Secure Routing Protocols – RIP v2, OSPF, BGP4 with authentication and encryption, Route Filtering	ICMP Broadcast Rejection	SNTP Time Synchronization support
Enhanced Console and System Logging		

Denial of Service Protection

IronShield Standard Security Features For Denial of Service Protection		
High-performance ACLs	ICMP Burst protection	ARP Broadcast protection
Anti-Spoofing ACLs	ICMP Broadcast protection (Smurf)	Proxy ARP protection
QOS	ICMP Redirect protection	SYN Flood protection
Diff Serve	ICMP Unreachable Suppression	Fragmentation Attack protection, Fragmentation PBR
Broadcast Rate Limiting	UDP Broadcast Protection	All Broadcast Protection
L2 and L3 Rate Limiting	L3 Network Address Translation (NAT)	LAND Attack protection
SFlow Real-Time Monitoring	SNMP Traps & Alerts	Enhanced System Logging

Enhanced Perimeter Protection

IronShield Standard Security Features For Enhancing Perimeter Protection		
L2 & L3 High-Performance ACLs, Anti-Spoofing ACLs	Rich DoS Feature Set – see previous section	SFlow Real-Time Monitoring
Policy Based Routing	Broadcast Protection	SNMP Traps & Alerts
QOS, Diff Serve	L2 and L3 Rate Limiting	Enhanced System Logging
SYN-Guard (ServerIron)	TCP-SYN Attack (ServerIron)	SYN-Defense (ServerIron)
DDoS Attack Suppression (ServerIron)	Worm Filtering (ServerIron)	Transaction Rate Limiting (ServerIron)
Firewall Load Balancing/Active Square (ServerIron)	DNS Proxy (ServerIron)	SSL Acceleration (ServerIron)
Security Tracing With Debug Filter (ServerIron)	Connection Rate Limiting (ServerIron)	Enhanced Network Address Translation (NAT) (ServerIron)

Enhanced Internal Network Protection

IronShield Standard Security Features For Internal Network Protection		
High-performance L2 & L3 ACLs	ICMP Burst Protection	ARP Broadcast Protection
Anti-Spoofing ACLs	ICMP Broadcast Protection (Smurf)	Proxy ARP Protection
QOS, Diff Serve	ICMP Redirect Protection	SYN Flood Protection
L2 and L3 Rate Limiting	ICMP Unreachable Suppression	Fragmentation Attack Protection
L2 MAC Address Filtering	L3 Network Address Translation (NAT)	LAND Attack Protection
Embedded RMON and RMON II support	High-Performance L3 & L4 Access Policies	Port/Tagged/Dynamic/Multi-use/Protocol Based VLAN Support
SFlow Real-Time Monitoring	Secure Routing Protocols (RIP v2, OSPF, BGP4)	Enhanced System Logging, SNTP Time Synchronization
802.1x Port Authentication	SNMP Traps & Alerts	Port Security With MAC Address Locking
UDP Broadcast Protection	Broadcast Limiting	All Broadcast Protection

Enhanced Network Visibility

IronShield Standard Security Features For Enhanced Network Visibility		
JetScope Traffic Monitoring - sFlow	SNMP Traps & Alerts	Enhanced System Logging
RMON, RMON II Support	Multiple Port Mirroring	

Appendix B - Physical Security Design Considerations

Besides implementing all of the security features discussed in this document, it is very important that physical access to your network devices be secured. Good physical security is at the foundation of keeping your network secure. Nothing will prevent a knowledgeable intruder from gaining access to your devices if they can physically gain access to the device itself. Your Corporate Security Policy should contain the minimum levels of physical security required for your network devices and how they should be secured.

A good physical security policy will address and define the following levels of physical security:

- A secure location that places routers and switches in a locked room. The room should not have other access points other than the locked door(s) – such as raised floors or false ceilings that are not properly extended to prevent access from adjacent areas.
- All doors should have key locks or badge readers that limit the access to authorized personnel only. Solid doors without windows should be used for added physical security. Automatic door closers should be used to make sure the door isn't accidentally left open. They should never be propped open.
- Your locks should match your security requirements. For example, high security installations may require biometric readers over key code locks or keys that can be duplicated or lost.
- Limiting only authorized personnel into your network rooms is critical in keeping your network secure. Keep the number of authorized personnel down to a minimum and if possible, design the network rooms to be separate from other corporate functions such as telecom to keep the access lists separated.
- Secure backups should be kept for all device configurations and OS versions. You should encrypt the backup configurations and keep them on a secure server with limited access to only authorized personnel. Never leave your configurations on TFTP servers after they have been downloaded.
- Protection against fire, water, heat, humidity, dirt and dust is a requirement as these elements can cause severe damage to network devices.
- Cable splice points, repeater panels, or patch panels should be physically secured to prevent packet snooping.
- Protection against electrical surges or brownouts is required to ensure a reliable network. Uninterruptible Power Supplies (UPS's) and power filters should be installed for mission critical network components.
- Keep the rooms neat and tidy. Remember, it is a network room and not a storeroom.

Foundry Networks, Inc.
Headquarters
2100 Gold Street
P.O. Box 649100
San Jose, CA 95164-9100

U.S. and Canada Toll-free: (888) TURBOLAN
Direct telephone: +1 408.586.1700
Fax: 1-408-586-1900
Email: info@foundrynet.com
Web: http://www.foundrynet.com