Firewalls and Intrusion Detection Systems (IDS)

Firewalls

- Core concept: block network connections the network administrator (aka organization) already knows ahead of time that they DO NOT want established
- Could be seen in different forms:
  - Specific app that blocks or allows connections based on a pre-defined set of criteria
  - Network Address Translation (NAT) = hide the computer's true address from the outside world
  - Proxy = help prevent connections to un-authorized sites
    - Question: how does an organization ensure its members use the proxy?
    - Answer: have the firewall block connections that are not from the proxy or allowed by the network admin
- Internal DNS server can be paired with the proxy server
  - Block DNS requests to un-authorized sites

Firewalls - NAT

- Can help prevent an external malicious entity from scanning the internal network
- Allows the internal network to use private addresses like 192.168.0.0/16 etc
    - An organization does not need to buy a class A range of IP addresses in order for all its members to communicate with the Internet
    - Only specific services will need public address like the mail server
- The internal network can have a whole range of services without the external network from knowing they even exist

Firewalls - NAT

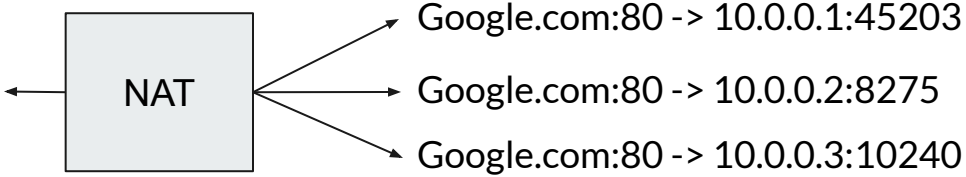
- This device maps a session based on a 4 tuple
  - Source IP address + Destination IP address + Source TCP/UCP port number + Destination TCP/UDP port number
- This translation is transparent to the user
  - User does not need to know this translation is occurring
  - This is why you cannot use “ip address” or “ifconfig” to find out your IP address while at home

Firewalls - NAT

- Pros
  - Can help prevent an external malicious entity from scanning the internal network
  - Allows the internal network to use private addresses like 192.168.0.0/16 etc
    - An organization does not need to buy a class A range of IP addresses in order for all its members to communicate with the Internet
    - Only specific services will need public address like the mail server
- Cons
  - 1 address is limited to 65535 connections
    - This is more of a problem for a large organization
    - Example on next slide shows this. Limitation = external's source port numbers
  - Valid connections cannot be initialized from the external network
    - A whole in the service is needed.
    - Really this can be seen as a problem for P2P connections

Firewalls - NAT Example

Google.com:80 <- corpwebsite.com:2973;
Google.com:80 <- corpwebsite.com:71039;
Google.com:80 <- corpwebsite.com:11123



External				Internal			
Src IP	Dst IP	Src Port	Dst Port	Src IP	Dst IP	Src Port	Dst Port
corpwebsite.com	google.com	2973	80	10.0.0.1	google.com	45203	80
corpwebsite.com	google.com	71039	80	10.0.0.2	google.com	8275	80
corpwebsite.com	google.com	11123	80	10.0.0.3	google.com	10240	80

Firewalls - Proxy

- Filters all of the connections leaving the internal network to block un-authorized connections
- These “un-authorized connections” are defined by the network administrator ahead of time
- Very helpful for preventing members from going to sites they should not be going to with an organization’s computer
    - Block connections to shopping sites (this can be a little extreme)
    - Block IP addresses that are known to be controlled by a malicious entity

Firewalls - Proxy

- Question:
  - How does an organization ensure its members use the proxy?
- Answer:
  - Have the firewall block connections that are not from the proxy or allowed by the network admin
- The user needs to configure their machine to use the proxy
  - Unlike with NAT, the user needs to actively know the proxy exists

Firewalls - Proxy

- Pros
  - Have a single point where all connections must ask for approval
  - Can log all external connections
  - Can block connections the network administrator knows are bad
- Cons
  - Can be a single point of failure (depending on implementation)
  - The hardware itself needs to be able to support the organization’s processing load

Firewalls - Specific App

- Can be as simple as an app that blocks or allows specific connections
- App can run on a specific server (like a Proxy server)
- App can run as a service on your personal computer
- This firewall will be a list of rules specified by the administrator
    - All connections must go through this list to see if it is an authorized or un-authorized connection
- List will compare against packet header information
    - Source/Destination IP address
    - Transport layer (layer 4) protocol
    - Source/Destination TCP/UDP port number
- Could also have more advance features
    - Ex: determine if it is an active connection or not

Firewalls - Specific App

- Overall: what type of firewall is it? In other words what is its default policy?
  - Block all connections?
  - Accept all connections?
- Default policy = DROP (aka block)
  - In general, the list of rules will state which connections are authorized.
  - Otherwise, if the network administrator DOES NOT list an IP address/protocol/port number as good
    - Drop the connection = prevent the connection from occurring
  - Pros:
    - Difficult for a malicious entity to infiltrate the network directly
      - They would need to pivot with one of the accepted IP addresses
  - Cons:
    - The network administrator has to audit all possible addresses.
      - This is really a problem when filtering the IP addresses.
      - Ex: Google's server farm and CDNs
    - Can make it difficult to manage if an accepted organization changes their IP address

Firewalls - Specific App

- Default policy = ACCEPT
  - In general, the list of rules will state which connections are un-authorized.
  - Otherwise, if the network administrator DOES NOT list an IP address/protocol/port number as bad
    - Allow the connection to occur
  - Pros:
    - Potentially less IP addresses to audit and keep track of
      - So, easier to manage
  - Cons:
    - Malicious entity can infiltrate the network directly
      - They just need to use an address that is not blocked by the firewall

Firewalls - Specific App

- In either case, each type requires the network administrator to know some info about the connections transmitted within the network
- Additionally, debugging firewall rules can be very difficult
  - Especially when there are alot of special cases
- Some firewall implementations use groups to help organize the list of rules
  - This helps with debugging but it is still difficult
- It is useful for a firewall to keep track of the client and server endpoint in a connection
  - Allows the firewall to block connections that are initialized by the server
  - In some cases, this can be seen as a bad thing and a potential attack
- Also useful for a firewall to keep track of the number of connections established by a single address
  - In this case, the firewall would be able to prevent the SYN flood attack

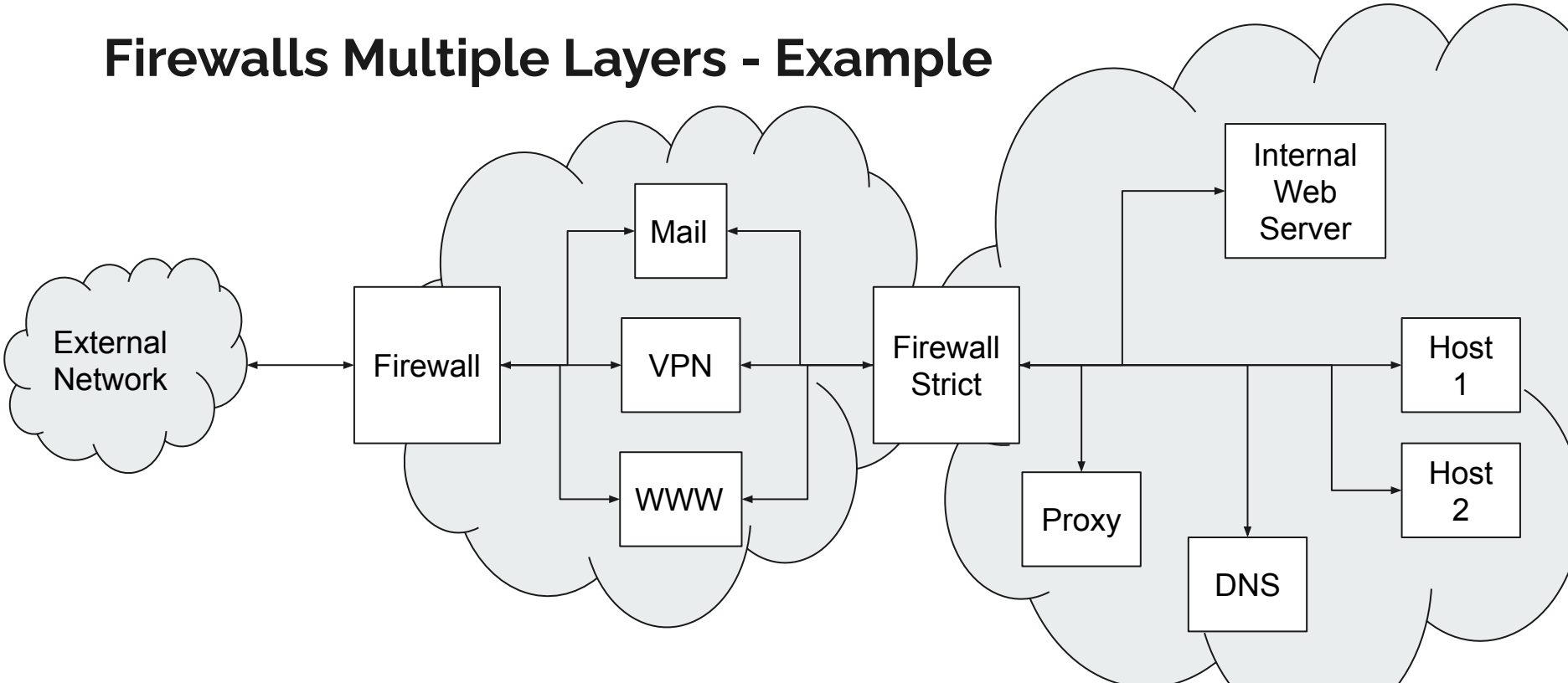
Firewalls - Specific App Example

- Let's use the "iptable" command as an example because it is used by the Ubuntu OS
- By default, the Ubuntu OS allows all connections to be established
- Let's change it so that the firewall only allows
  - Connections initialized by your local machine
  - Otherwise, drop the connections (default policy = DROP)
- We will be modifying the iptable's INPUT and OUTPUT chains for the FILTER table

Firewalls - Specific App Example

- Set the default policy to drop
  - sudo iptables -P INPUT DROP
  - sudo iptables -P OUTPUT DROP
- Rules for filtering all of the packets going to your local machine(INPUT):
  - sudo iptables -A INPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
  - sudo iptables -A INPUT -p udp -m state --state ESTABLISHED,RELATED -j ACCEPT
- Rules for filtering all of the packets going out to the server (OUTPUT):
  - sudo iptables -A OUTPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
  - sudo iptables -A OUTPUT -p udp -m state --state ESTABLISHED,RELATED -j ACCEPT
  - sudo iptables -A OUTPUT -p tcp -m state --state NEW -j ACCEPT
  - sudo iptables -A OUTPUT -p udp -m state --state NEW -j ACCEPT

Firewalls Multiple Layers - Example



Firewalls Leads to IDS

- Now that the connections are
  - Filtered
  - Logged (aka monitored)
- How can you effectively parse through the logs?
  - In a multi-user environment, that is a lot of data to audit
  - Think of the amount of connections established by the students on this campus's. That is a lot of connections.
- Instead of using the default policy of DROP and manually auditing all the addresses or
  - Ensure an authorized address is not attacking your network
- The network administrator can use the logs plus some IDS methodologies to actively filter connections based on specific metrics
  - Rather than just header information

IDS

- Core concept:
  - Review network connections for malicious behavior
    - If malicious, flag for auditor to review
  - Detect the malicious behavior in an “acceptable” amount of time
  - Gather all of the necessary information about the malicious activity so an auditor can make an informed decision
  - Be accurate (if possible)

IDS

- Three components of an IDS architecture
  - Agent
    - This generates the dataset (aka the log data)
  - Director
    - Obtains the dataset and analyzes it for vulnerabilities
  - Notifier
    - Is notified of the malicious activity and acts upon it
    - Since this is IDS and not IPS, it will tell an auditor

IDS

- Three different types:
  - Anomaly detection
    - Compare the current case against a base case.
    - If the current case does not match (or closely match) the base case, then it is an anomaly and it needs to be audited
  - Misuse detection
    - Does the current case match any known vulnerabilities
    - If so, then it is a malicious event and needs to be audited
  - Specification detection
    - Does the current case match the expected case
    - If NOT, then it is a malicious event and needs to be audited

IDS - Anomaly detection

- Perform a statistical comparison between the current case against a base case
- If the current case is not within range of the base case, then it is flagged as malicious behavior
- Methods for gathering the base case
    - Threshold
    - Statistical
    - Markov Models
    - Machine learning

IDS - Anomaly detection

- Easiest way to think of anomaly detection
  - In an isolated environment, where you know a malicious event cannot occur
  - Execute the service/process/etc and gather the necessary statistics
  - Since its an isolated env, an auditor does not need to review the dataset
    - Because of the assumption that it operated as intended

IDS - Anomaly detection: Threshold

- Most simplest case
- IDS is expecting an event to occur a specific amount of times
  - It can be a range
- When the event does NOT occur within the specified range, it is seen as a malicious event
- How is this threshold value specified? It varies.
  - Password attempts = arbitrarily low number
  - Amount of data sent by service = run in isolated env

IDS - Anomaly detection: Statistical

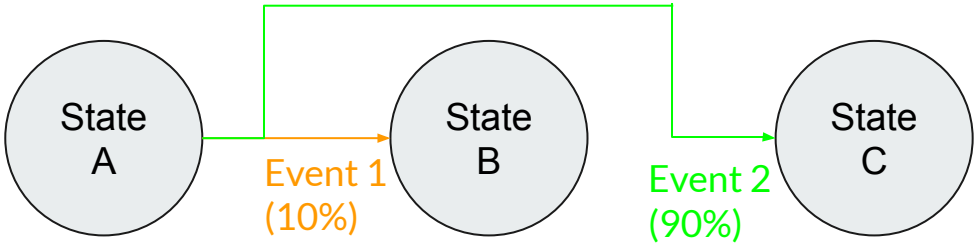
- IDS is provided the mean and standard deviation
  - In other words, the base case
- When an event is outside the standard deviation +/- the mean
  - The event is seen as malicious
- Example:
  - Employee SSH's into their system to work
  - Base case's statistics:
    - The number of times they SSH into the machine in a single day.
    - Amount of data that is passed within a session.
    - Weekday vs weekend.
    - The time of day. 8am-5pm (PDT)

IDS - Anomaly detection: Statistical

- Example (cont)
  - For months, they have been working from 8am-5pm
  - But now the project has changed, and they need to work with someone on the East coast
  - This means they will be working from 6am-3pm (PDT)
- Their AUTHORIZED morning (6-8am) SSH connections are flagged as MALICIOUS
  - This increases the amount of work for the auditors
- Meaning, the base case's statistical dataset needs to change over time
  - And take in consideration a real environment, not just an isolated one
- Pros:
  - Reduce the number of false positives
  - The statistical dataset stays current and keeps up with the needs of the employees
- Cons:
  - A serious malicious entity can use this to change the statistical dataset to meet their needs

IDS - Anomaly detection: Markov Model

- Core concept: use sequence of events and states to generate statistical dataset
  - Stats are based on NOT the occurrence of an event
  - Stats are based on the probability of a sequence of events
- When an event occurs, it will put the system/app/etc in a state
  - So… the state will change for the next event
- Throughout the lifetime of the system/app/etc, the dataset can produce a probability of state
  - Think of Computer Architecture and the jump prediction algorithms



IDS - Anomaly detection: Markov Model

- How is an event identified as malicious?
  - When an event causes the state to change
  - Where this state had a low probability of occurrence
- The dataset is also known as the “training data”
  - A good training dataset is crucial for this model to work adequately
  - Otherwise, the model will produce false positives and possibly false negatives (a real malicious event)
- The training data is obtained by executing the system/app/etc within an environment that contains no malicious activity

IDS - Anomaly detection: Machine Learning

- https://cs.ucdavis.edu/faculty-research/artificial-intelligence-and-machine-learning
- ECS 171 and 271 offered Spring 2022 (tentative)
- Supervised vs Unsupervised
- Supervised learning method
  - Similar to Markov model, requires a training dataset
  - The training data informs the algorithm of malicious and non-malicious activity
  - Then the algorithm is given a benchmark dataset to determine its efficiency
- Unsupervised learning method
  - No training data is provided the algorithm
  - The algorithm just runs against a dataset
  - With the assumption that this dataset does not contain malicious activity

IDS - Misuse detection

- Core concept: check for vulnerabilities attackers are going to attempt to exploit
- Compare the current case against a list of known vulnerabilities
- If the current case matches a known vulnerability, then flag it as a malicious event
- Pro:
  - Depending on the vulnerability's requirements, the flagged event will help in identifying a malicious entity
- Con:
  - Does NOT help against zero-day attacks or variants to known vulnerabilities

IDS - Misuse detection: Zeek

- Zeek
  - Previously called Bro
- Network based IDS
- Two parts
  - Event engine - backend, not really the interesting part
  - Policy script interpreter - frontend, configured by the network administrator
- Think of it like a framework

IDS - Misuse detection: Zeek

- Event engine
  - Just grabs any and all occurrences that meet a generic pre-defined rule
- Policy script interpreter
  - Like someone merged Python and C++ together
  - Network administrator specify the functions that meet a vulnerability's requirements
  - And within these functions, the network administrator can use variables (ex: arrays) and conditional statements (ex: if statements) to look for the exact requirements necessary for exploiting the vulnerability
  - The network administrator might need to use multiple functions, variables, and conditional statements to look for a single vulnerability

IDS - Specification detection

- Core concept: ensure the system/app/etc is operating as intended
- Compare the current case against the system/app/etc intended operating case
- If the current case does not match the intended operating case, then flag it as a malicious event
- Pro:
  - Will flag zero-day attacks, known vulnerabilities plus their variants
- Con:
  - The intended operating case needs to be accurate and needs to cover all possible intended cases
  - Can be difficult to maintain because any changes to the system/app/etc will probably require the intended operating case to be updated