A Denial of Service Resistant Intrusion Detection Architecture

Peter Mell, Donald Marks, Mark McLarnon
National Institute of Standards and Technology
Computer Security Division
5-24-2000

Keywords: Intrusion detection, Denial of service, Security architecture, Mobile agents, Computer security

Abstract
As the capabilities of intrusion detection systems (IDSs) advance, attackers may disable organizations' IDSs before attempting to penetrate more valuable targets. To counter this threat, we present an IDS architecture that is resistant to denial-of-service attacks. The architecture frustrates attackers by making IDS components invisible to attackers' normal means of "seeing" in a network. Upon a successful attack, the architecture allows IDS components to relocate from attacked hosts to operational hosts thereby mitigating the attack. These capabilities are obtained by using mobile agent technology, utilizing network topology features, and by restricting the communication allowed between different types of IDS components.

1. Introduction
While not yet seen in the wild, intrusion detection systems (IDSs) may become a primary target for attackers. As IDSs become more widely deployed, and as their automated response capability increases, attackers may find it necessary to disable an organization's IDS before carrying out their nefarious activity. Furthermore, such attacks may be easy to launch against the new breed of distributed IDSs with hierarchical, interdependent components. In such systems, IDSs often have command and control or analysis components that, if disabled, render the IDS useless over a large portion of the protected network. We call these *critical* components and the host on which they reside *critical hosts*.

To counter the threat of attackers finding and disabling critical IDS components, IDS vendors devote many resources towards designing intrusion detection systems such that they cannot be penetrated. For the most part these efforts have been successful and stand-alone intrusion detection hosts have shown a strong resistance to penetration attacks. Less effort has been spent securing IDSs from flooding denial of service (DOS) attacks. In part, this is because it is widely thought that nothing can be done about this type of attack.

However, it is a common misconception that systems in general cannot avoid the consequences of a flooding DOS attack. To demonstrate this, our paper describes an IDS architecture that enables IDSs to become resistant to flooding DOS attacks. Furthermore, the model does not require IDSs to operate in an ineffective or obscure manner, but rather builds upon the traditional distributed hierarchical model used by the majority of IDSs today.

Our IDS architecture resists flooding DOS attacks using a combination of techniques. First, critical IDS components are made invisible to an attacker's normal means of "seeing" in a network: active network scanning and passive packet monitoring. Second, critical IDS

components are made adaptive to flooding DOS attacks in that they can automatically relocate to another host in the event of an attack. The relocation is invisible to the attacker who then cannot persist in the attack. Our model does not prevent an attacker from launching attacks but instead makes the important targets invisible which forces the attacker to fire blindly. Random attacks may actually hit a critical IDS host, but the ability of IDS components to move between hosts in the event of an attack mitigates the damage.

2. Vulnerable IDS Architectures

We envision distributed IDSs of the future consisting of thousands of small event gathering agents reporting to hundreds of event analyzers that are all part of a unified system with centralized reporting and control. Each of these agents may be configured to detect a single event, or they may detect several events. Each network component may host one or many agents. Since there will be a large number of event generators, for performance reasons the events must be abstracted, analyzed, and condensed by a hierarchy of agents before reaching the centralized reporting and control facility. Various types of hierarchies are shown in Figure 2.1. While not shown in the figure, event information flow in the hierarchy is usually up while command and control flow is usually down. Also, the number of IDS components higher up the hierarchy is usually, but not always, smaller than the number of components at the bottom of a hierarchy since event information is abstracted and condensed as it moves higher. The physical location of event generators will be fixed since they monitor stationary resources. However, the internal nodes of the hierarchy may exist at many locations in the network since they receive their input and give their output via network connections.

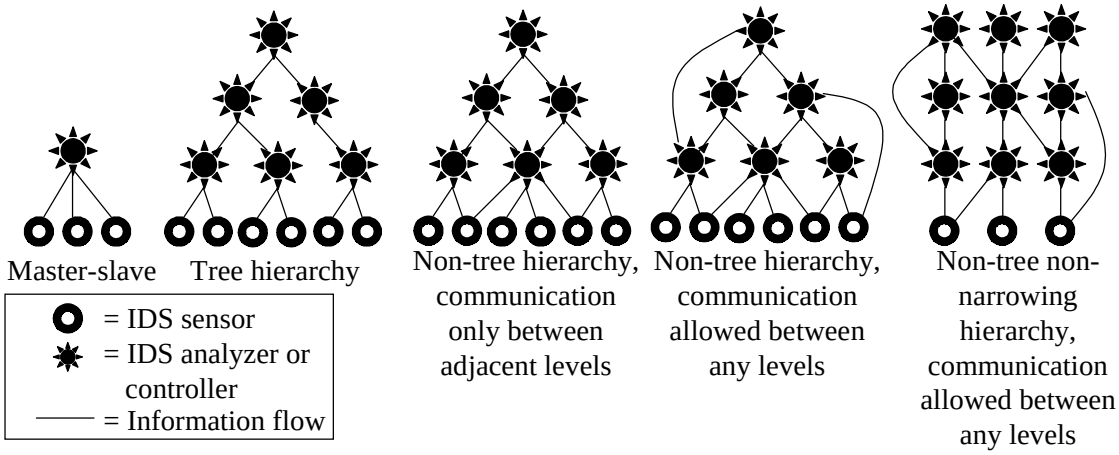

Figure 2.1: Various types of distributed hierarchical IDS architectures

A hierarchical architecture is already used today by many distributed IDSs that have reached the scalability limits of using a single analyzer and control component for all event generators in a network. Examples of IDSs using a hierarchical structure for their component communication model include: UC Davis's GrIDS [12], Lawrence Livermore's SPI-NET [8], Cisco's NetRanger [5], Axent's Intruder Alert [1], Internet Security System's RealSecure [6], Network Associates Incorporated's Active Security [10], and Purdue's AAFID [2].

While hierarchical IDS architectures provide many organizational and scalability benefits, they have a weakness that we must compensate for when designing them. The weakness is that

hierarchical IDSs are prone to have single points of failure that are easily discoverable by an attacker. If an attacker can disrupt such a failure point, a large portion of the networks IDS becomes inoperable and the attacker can then exploit weaknesses in an organization without fear of detection or retribution.

3. Methods to Target IDS Hosts

Before an attacker can disrupt an internal IDS component, he or she must find the host on which the component is residing. Unless the attacker is part of an organization's network management or security groups, the attacker has a limited ability to "see" hosts in a network. The two primary ways for a hacker to see are: sniffing and probing. Passive network sniffing is where an attacker listens to the network traffic passing by a host on which the attacker has control. Active probing is where a hacker maps out the hosts in a network by sending out packets to the IP addresses owned by an organization. Active probing can reveal hosts that are alive, the operating systems they are running, the server applications running on those operating systems, and even the version numbers of server software. The most popular software package for active probing is Nmap [11].

4. The Susceptibility of Hosts to Denial of Service Attacks

While difficult, it is possible to secure a host against penetration attacks by carefully designing the software that runs on the host. Using the same technique, it is also possible to secure a host against DOS attacks that take advantage of a vulnerability in order to freeze, slow down, or shut down the host. However, it is impossible to secure a host (with no outside protections) against flooding DOS attacks. An attacker who can gain network connectivity to a target host can send it more information than it can process thereby overwhelming the host's ability to respond to legitimate requests. Countering this with a faster target system only requires that an attacker gather more network resources with which to flood. Given the number of poorly managed systems in the typical network, it is not difficult for an attacker to gain control of many "non-important" hosts and coordinate attacks with these hosts using tools like Tribe Flood Network and Stacheldraht [3]. Attacks like smurf make the task of gaining more flooding resources easier, and allow an attacker to amplify his flooding resources a hundred or even a thousand-fold [4]. It may not also be practical to defend all critical resources by installing extremely fast computers and even if an organization went to this extreme, a flooding DOS attack could fill the network pipe that is feeding traffic to the target. In this case, the target continues to function but cannot process any legitimate network requests because the attacker has consumed all of the network bandwidth.

5. Example Attack Scenario

We fear that hackers may discover the location of important IDS components and then launch flooding DOS attacks to freeze or shut them down. A hacker might distribute reconnaissance code throughout an organization using targeted viruses or worms as shown in Figure 5.1. This is simple to accomplish since virus detection systems only detect previously recorded viruses. Alternately, if the hacker is an insider he may already have access to a large number of systems and can manually place the reconnaissance code.

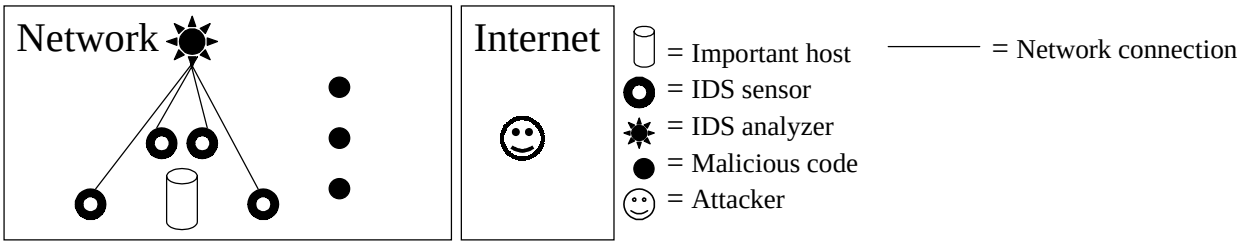

Figure 5.1: Attacker sneaks malicious code into the network

As shown in Figure 5.2, the malicious code can gather information about the network by actively scanning the internal hosts and by passively monitoring network traffic. Upon determining the location of the critical IDS components and suitable targets, the malicious code opens a covert, channel back to the attacker. Even if an organization became aware of the reconnaissance code, by the time a response was initiated the hacker would have gained a view of the organization's internal IDS topology. Someday there may exist a black market to sell such network topology information.

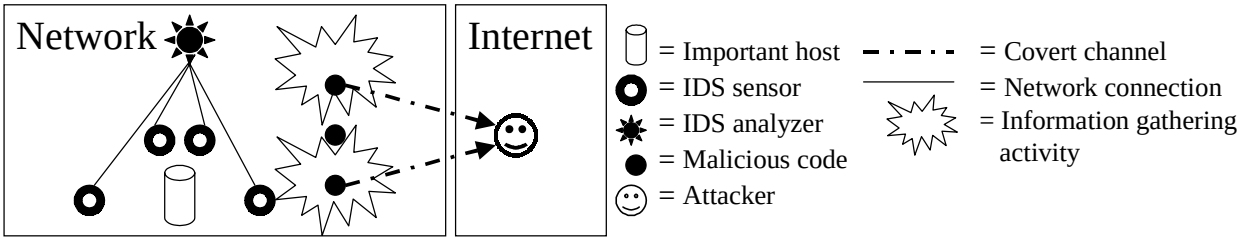

Figure 5.2: Malicious code gathers and reports information about the location of IDSs and important servers to the attacker

Upon discovery of this IDS topology, a hacker would like to penetrate and control the distributed IDS. However, key IDS components are likely to be well maintained and difficult to penetrate. Instead, as shown in Figure 5.3, each instance of the malicious program can launch a flooding DOS attack against critical IDS components. An organization's IDS, without the critical aggregation, analysis, reporting, and command elements can not effectively detect and respond to attacks. During this IDS downtime, the hacker, other instances of the malicious code, and possible others can launch attacks and probes that penetrate, steal information from, and install back doors in important systems.

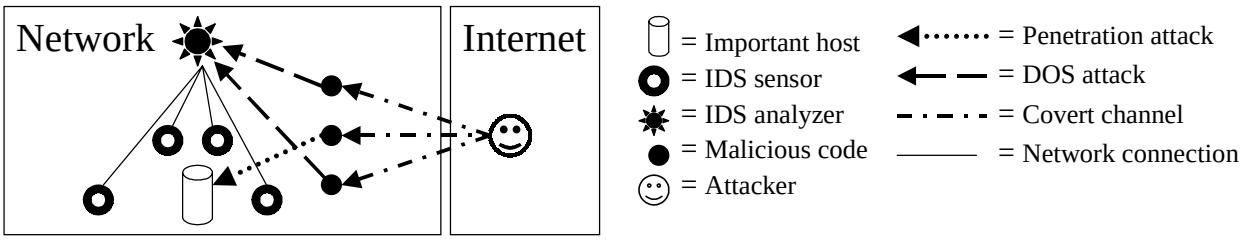

Figure 5.3: Attacker instructs the malicious code to disable the IDS and to penetrate an important host.

6. Existing Solutions

There are no known ways to prevent flooding DOS attacks against hosts that are visible on the Internet. One can purchase an ever faster set of servers and network connections, but an attacker with enough resources can flood those publicly accessible resources. Because of this, many assume that flooding DOS attacks in general can not be countered. As a result, little research is done to explore the different ways these attacks can be mitigated or prevented.

However, applications that do not have to be publicly visible on the Internet, like IDSs, can defend themselves against flooding DOS attacks. There exist several solutions for IDSs that offer different levels of protection and require the use of different levels of resources.

Active IDS Response

Some IDSs have the ability to add or change filtering rules in routers and firewalls [7]. With this capability an IDS sensor can detect a flooding DOS attack and quickly reconfigure the appropriate routers to stop the attack. Such an action can effectively stop a flooding DOS attack, however, the solution has some drawbacks:

1. The IDS must have control of most of the routers in an organization in order to initiate a response that can stop any possible attacking host from flooding any important IDS host.
2. Initiating filtering on routers decreases their performance and many organizations will desire solutions that do not reduce their network bandwidth.
3. Network operations personnel may be unwilling to relinquish control of all network routers to security operations personnel.
4. The fear that such automatically generated filtering responses could harm legitimate traffic may make this solution unappealing.
5. Unless filtering is enabled on most routers prior to any attack, an attacker may be able to launch a flooding DOS attack against the IDS response host, that contacts the routers to initiate filtering, thereby disabling the defensive system.
6. IDSs often take time to detect and respond to attacks (sometimes as much as several minutes). Attackers can use this window to temporarily disable the IDS with flooding DOS attacks while they penetrate their real targets.

Separate Communication Channels

Another solution is to have separate physical wires on which IDSs can communicate. This solution is effective (assuming that IDS nodes are not penetrated), but it is too costly for most organizations to install a separate network just for IDS systems.

Decentralized Non-hierarchical IDSs

Some researchers have investigated creating IDSs that are not hierarchical and whose IDS components are not interdependent. While such an IDS could be very resistant to many DOS attacks, it has proven difficult to build such decentralized distributed IDSs. This area is actively researched, and viable prototypes may emerge.

Mobile Recoverable Components

Another solution is to make it difficult for an attacker to disable critical IDS components by wrapping the critical IDS components as mobile agents [9]. In this scheme, the mobile agent critical components randomly move around a network so that an attacker does not know where

they reside. If an attacker does manage to find and attack a critical component, other mobile agents immediately take over the functionality of the destroyed agent. Furthermore, the architecture is designed such that there exists no single points of failure that an attacker could use to take down the mobile agent system. While this scheme may be useful, it fails to take advantage of domain specific knowledge about IDSs and assumptions that can be made about network topologies. This paper builds upon the concepts presented in this work and attempts to build a stronger defensive mechanism by using IDS domain specific knowledge and assumptions about networking topologies.

7. Overview of Our Solution

To solve the problem of attackers finding and disabling critical IDS hosts, we propose an IDS architecture that will cloak the IDS targets from the hacker's sensors thereby forcing them to blindly launch easily evaded attacks. This architecture thwarts hacker attempts to use passive sniffing or active network probing to detect IDS topologies by making critical IDS hosts invisible to attackers. Attempts by hackers to shut down critical IDS hosts are thwarted by using mobile agent technologies to seamlessly relocate critical IDS processes on attacked hosts to hosts that are still operational. This combination of features provides our IDS architecture with strong mechanisms for resisting flooding DOS attacks.

Note that our formal model also addresses resistance to penetration attacks. However, this paper does not add substantive new ideas to this area, and so we do not emphasize resistance to penetration techniques but still include this capability in the model and in the formal proofs.

Thwarting Passive Sniffing

Critical IDS hosts evade detection from attacker's network sniffers by taking advantage of security assumptions that can be made about different parts of a network. Critical hosts are placed in backbone networks, which are assumed to only contain network elements that are difficult to penetrate (e.g. routers, switches, firewalls, network based IDSs). All communication from a critical host to a non-critical host must pass through an intermediary. This intermediary is called a proxy IDS host and is located in the network backbone. The proxy hosts can also be designed to be hard to penetrate by limiting its interface and functionality.

Since all network elements in the network backbone are hard to penetrate, an attacker cannot sniff on the backbone. As well, since all communication from a critical host to a non-critical host passes through a proxy host, an attacker sniffing near the non-critical host cannot discover the location of the critical host. The attacker can easily discover the location of a proxy host, but these hosts are expendable in our model. The model restricts how many proxy hosts an attacker can discover which prevents an attacker from disabling too many proxy hosts. This technique is shown in Figure 7.1.

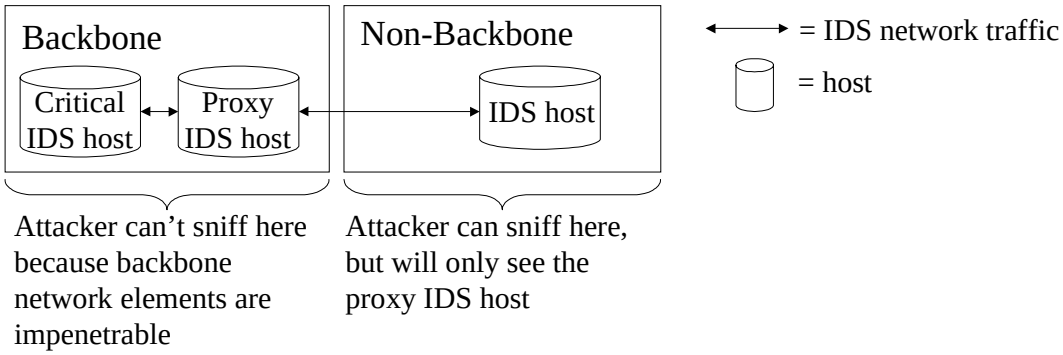

Figure 7.1: Critical host's protection against discovery through sniffing

Thwarting Active Network Probing

Critical IDS hosts evade discovery through active network probing by using what is becoming standard stealth technology [7]. Similarly to many existing IDSs, critical hosts do not react to incoming network packets unless the packets can be verified as originating from an authorized security host. If the packets are not from an authorized source, the critical host quietly drops the packets without sending any reply, not even a reset packet. These simple filters, which must be implemented at the TCP/IP layer, will stop the majority of active network probing attacks. Introducing elements of randomness into network traffic patterns can mitigate other types of probing attacks, which are based on covert channel analysis.

Thwarting Flooding DOS Attacks

By thwarting passive sniffing and active network probing, our IDS architecture hides critical IDS hosts from an attackers' view. Despite this, an attacker may choose a random address and launch a flooding DOS attack against the host at that address. If this random address belongs to a critical host, then the organization's IDS might be crippled. Our solution to these random bullet attacks is to have multiple backups for every critical IDS process. These backups exist on distinct, protected hosts on the network backbone and maintain full or partial state information of the process they are backing up. When a critical host is attacked, backups on other hosts assume the functionality of the halted critical IDS processes.

8. Network Topology Assumptions

We model an organization's network by dividing the organization's network elements (e.g. routers, switches, firewalls, or hosts) into sets. We then assume certain properties about the sets and about the relationship between sets. Most but not all networks can be mapped into our model.

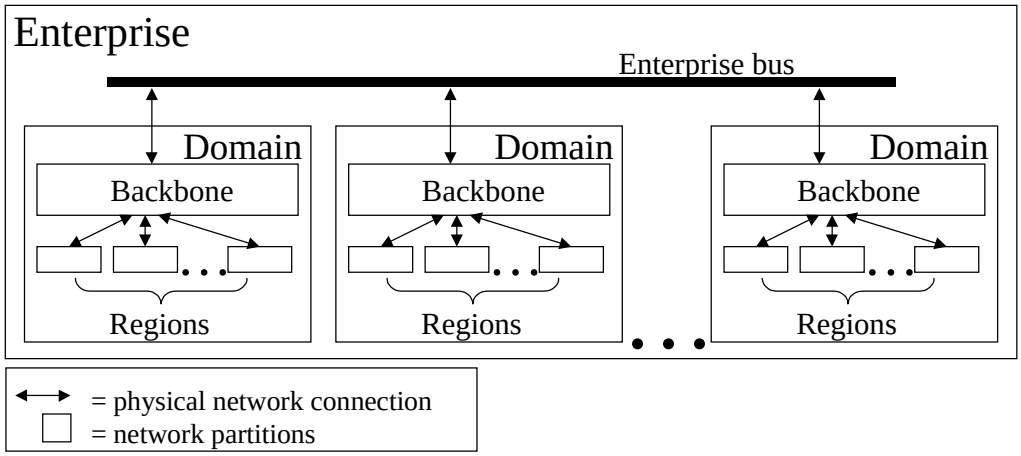

Figure 8.1: Schematic of Network Partitions

As shown in Figure 8.1, the superset of all sets of network elements is called the e*nterprise*. The enterprise is the union of a set of *domains*. Each domain is the largest possible partition of the network where every element in the backbone has a high bandwidth connection every other network element in the backbone. Furthermore, this high bandwidth connection does not traverse any public networks on which attackers may reside. Every domain has a physical network connection to the *enterprise bus*, which may include portions of the Internet or other public networks and which allows all domains to communicate. All domain traffic traversing the enterprise bus is assumed to be encrypted. The source and destination IP addresses are assumed to be hidden by an address translating firewall and link encryption, or by using a virtual private network. Since the enterprise bus is assumed to be a public network (probably the Internet), it is not required to have any degree of security and as such, attacks on domains may originate from it.

Every domain is the union of a backbone set with zero or more region sets. A backbone is a set of network elements that are typically secure against penetration from attackers on the network: firewalls, routers, and switches. Backbones are also allowed to contain security devices that are secured against penetration from network attacks. While backbone network elements are assumed secure against penetration attacks, they are not assumed secure against flooding DOS attacks. The backbone set of network elements have a direct physical connection to the enterprise bus.

Region sets, however, are assumed to be susceptible to both penetration and DOS attacks. Region sets may contain any type of network elements but usually contain hosts and servers used by an organization that are often penetrable. Region sets only have a physical network connection to their domain's backbone set. No region set has a direct physical network connection to another region, another domain, or the enterprise bus.

9. Three Types of Security Hosts and their Placement within a Domain

Every domain contains special security hosts that are mobile agent enabled. As shown in Figure 9.1, there are three types of security hosts: *critical*, *proxy*, and *child*. Analogous to the types of hosts, there exists three main types of mobile agents in our model: critical, proxy, and child. There also exist *backup* and *director* agents that are special types of critical agents. Generally,

critical agents must always reside on critical hosts, proxy agents on proxy hosts, and child agents on child hosts.

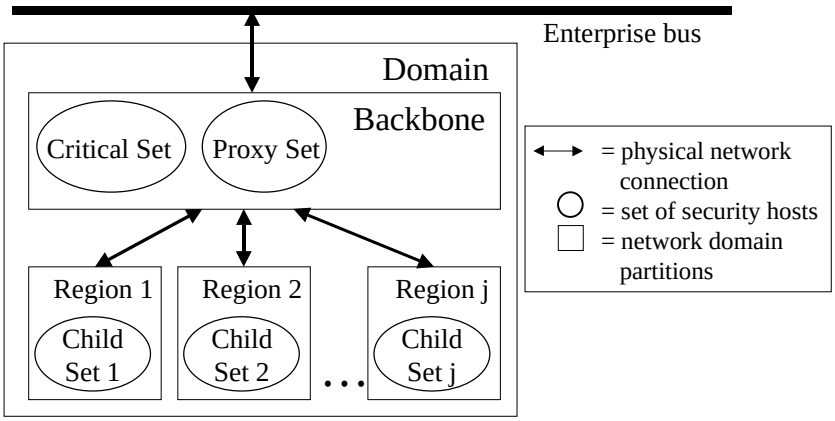

Figure 9.1: The three types of sets of hosts and their placement in a domain.

Critical hosts reside in a domain's backbone and house critical agents that perform intrusion detection aggregation, analysis, and control. The critical agents are the most important to protect against attacks. Child hosts reside in the regions of a domain and house child agents that gather and evaluate events. Proxy hosts reside in a domain's backbone and house proxy agents that provide communication services between child agents and critical agents. Critical agents are not allowed to directly talk to child agents as the communication could reveal the location of a critical hosts in the network. We explore this in detail later. Each host is assigned to a set: critical hosts make up the *critical set*, proxy hosts make up the *proxy set*, and the child hosts in some region, say j, make up *child set j*.

10. Allowed Intra-domain Interactions between Types of Security Hosts

The three types of security hosts (and thus their respective agents) within a domain are allowed to interact in only certain ways. The allowed interactions are shown in Figure 10.1.

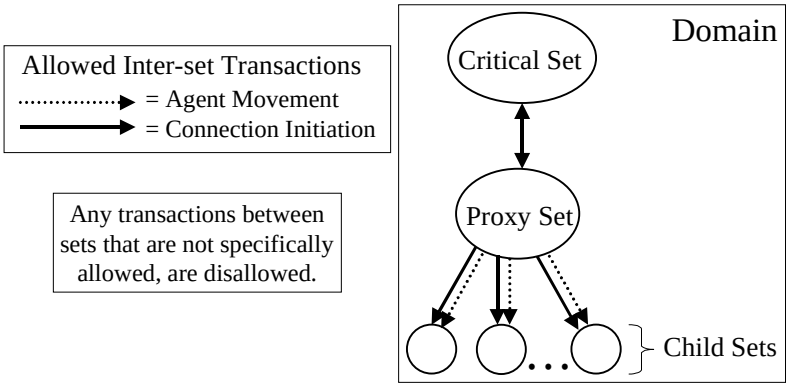

Figure 10.1: Allowed interactions between the three types of sets of security hosts in a domain

Agent Movement

Mobile agents may freely move between the hosts in their respective set. However, agents may not move between any sets except when moving from the proxy set to a region. Since child agents reside in regions that are by definition insecure, they are not allowed to move to other sets

for fear that malicious agents might be spread. Proxy agents are not allowed to move to the critical set in order to protect critical hosts from being visible to attackers in regions. If a proxy agent was on a critical host, the proxy agent would communicate with child IDS hosts thereby making the critical host visible to an attacker in the child set.

Connection Initiation

We restrict the ability of an agent in one set to initiate a conversation with an agent in another set in order to keep the location of certain hosts hidden. We say that a conversation is initiated when something analogous to a TCP connection is established. Once a conversation is initiated, either host may autonomously decide to send information. There are two restrictions in our model on conversation initiation. The first is that critical hosts may not initiate connections to child hosts in order to avoid allowing an attacker in a child set to see the location of a critical host. The second is that hosts in a child set may not initiate conversations with any other set. This restriction makes it unnecessary to give child hosts the location of any security host in another set. Since child hosts may be penetrable and reside in regions that may contain attackers, limiting the knowledge of a child host is important. Thus, a child host and its associated agents only know the location of the proxy agent that contacted it. This is an important property that we use in showing the security of the overall architecture.

11. IDS Communications Using the Proxy Set

Section 10 specifies that critical agents are not allowed to directly communicate with child agents. Instead, critical and child agents communicate through proxy agents. Proxy agents serve no other purpose except to route network communication.

There is a special critical agent, called the *director*, that controls which proxy IDS hosts communicate with which child IDS hosts. The director attempts to match each region of a network with a unique proxy host. By doing this, an attacker in a region can only see a single proxy host via sniffing. Ideally, there exists at least one proxy host per region (allowing for a one to one mapping) such that if an attacker wishes to eliminate all proxy hosts, the attacker would have to penetrate every region in a domain.

Child IDS hosts (and their respective agents) will only respond to IP addresses that are in the known backbone set of IP addresses for that domain. This set of allowable backbone addresses should be much larger than the actual set of backbone elements. Child hosts will therefore only know the addresses of those backbone IDS hosts that have communicated with it. By having the child host only accept IDS related packets from backbone hosts, an attacker in one region can not launch a port scan to detect IDS hosts in another region. If the attacker spoofed the probe packets as from a backbone host, the attacker could not sniff the targets reply (assuming the target is in another region) and thus spoofed probe packets would be useless to an attacker.

12. Allowed Interactions between Domains

We defined domains to be the largest network partition where any two network elements in the domain's backbone have a high bandwidth connection that does not traverse any public networks on which may exist attackers. Thus, an attacker on the enterprise bus can launch a flooding DOS attack against any domain and shut off its ability to communicate with other domains. Despite this, attackers on the enterprise bus may not affect the operation within a domain because we

assume that flooding DOS attack packets are stopped at the domain firewall. If the firewall doesn't prevent such packets, organizations should install IDSs next to their firewall that reactively block such attacks. Because attackers may be able to sever the communication between different domains, we require that the distributed IDSs that follow this model are designed such that each domain can function as stand alone IDSs.

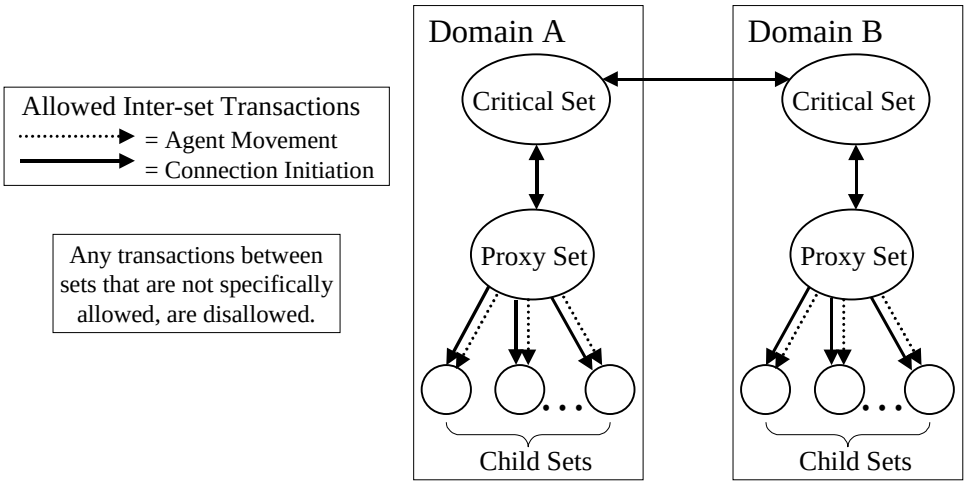

Figure 12.1: Allowed Inter-domain Communication

Since an attacker on the enterprise bus may sever communication between domains, it is important that all agents used to collect, analyze, and respond to attacks on a domain reside within that domain. Because of this, as shown in Figure 12.1, no agent movement is allowed between domains. We allow connection initiation between critical sets of different domains because that is the only way that a distributed IDS can analyze data from multiple domains. Inter-domain connection initiation involving child sets is not allowed since an attacker in a child set might discover the location of a security host in another domain.

13. Backing up Critical Agents

Every critical agent has one or more backup agents. An agent and its backup agents all reside on distinct critical hosts within the same domain. Backup agents maintain full or partial state information of the agent they are backing up. As shown in Figure 13.1, when an critical host is frozen or shut down by an attacker, an agent's backup agents contact each other to see which ones are operational. The operational backup agents decide on a successor for the frozen or shut down agent and this successor resumes the functions of the original agent. The unused backup agents terminate or become backups for the successor. The successor may have to create backups and send them to different critical hosts.

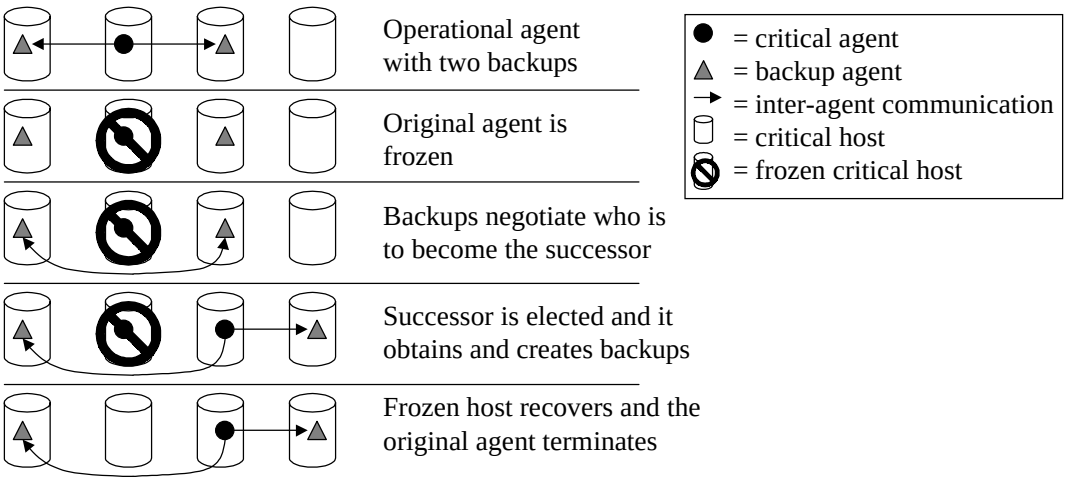

Figure 13.1: Example scenario of an agent becoming frozen and its backups resuming its functionality

If the original agent become unfrozen, it checks the system time to see how long it was non-operational. If more than a few seconds have passed then it terminates itself. If the interruption was short, the agent contacts its backups to see if they have already elected a successor. If they have, then the original agent terminates. If they have not, then the original agent resumes its former operation.

This scheme requires that each domain contain excess computing resources for the IDS agents since the IDS needs to operate even if several IDS hosts are disabled. Furthermore, IDS hosts in the backbone need to be able to automatically recover after a flooding DOS attack ceases so that resources that are not under current attack are available.

14. Stealthy Security Hosts

Critical and proxy hosts should be invisible to network mapping and port scanning software in order to keep their true location (IP address) hidden. To do this, backbone security hosts are assumed to have the ability to ignore incoming network traffic. Backbone IDS hosts should ignore:

- packets from non-IDS hosts
- packets from child IDS hosts where the child host does not already have a connection established with that critical or proxy host

Even with these stealthy features in place, there exists a probing technique based on covert channels whereby an attacker might be able to identify the IP addresses of critical and proxy hosts.

As shown in Figure 14.1, an attacker in a region of domain A sends a packet to a backbone host in domain B. The source IP address of this packet is a critical or proxy host located in domain A. The attacker simultaneously monitors the encrypted tunnel between domain A and B looking for sudden encrypted activity that could correspond to a reply from the host in domain B. If the attacker can match the activity in the encrypted tunnel to the probe packets returning from B then the attacker can identify that a specific host in the backbone of B exists. If this same IP address

does not respond to any other types of network mapping, then the attacker can conclude that it must be a critical or proxy IDS host.

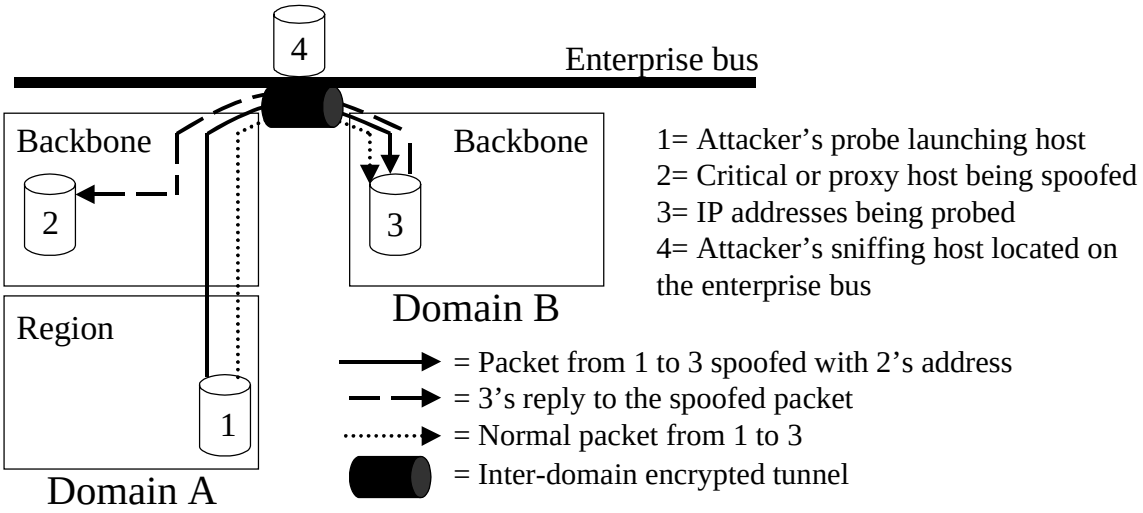

Figure 14.1: Network mapping covert channel attack

While this probing technique may be effective, it can be countered by sending constant random data through the virtual private networks connecting each domain. This activity makes it difficult for an attacker to match encrypted traffic to probe reply packets. Another technique is for domains to use a virtual private network that sends the same size packets at a set rate. Padding is used when not enough traffic needs to pass between two domains. Then, no matter what an attacker does, the same amount of encrypted traffic flows between any two domains.

15. Authentication Schemes Required by the Model

Every backbone IDS host has access to a centralized directory that contains the IP address of every other backbone IDS host in all domains of an organization. The centralized directory can confirm that a host is a backbone IDS host merely by looking at the source IP address of a request packet. If the source is a backbone IDS host then the directory responds to the query. If the request was actually spoofed by an attacker, the attacker will not be able to see the reply because the reply will stay within backbone networks and the enterprise bus. If the attacker is listening to the enterprise bus, his listening will be thwarted due to defenses introduced in section 14. Alternately, a centralized directory could use a public-private key solution to authenticate hosts, but it is interesting that such techniques are not necessary for the correct functioning of our model.

Child IDS hosts are provided with a public key that they use to authenticate any incoming network transmissions. This prevents an attacker from masquerading as a backbone IDS host and thereby commanding a child IDS host. The matching private key is cooperatively known and used by all proxy hosts in that domain.

16. Deploying the Attack Resistant Intrusion Detection Model

Our IDS model is very similar to the traditional hierarchical distributed IDS architecture and operates in the same fashion. Our model merely adds on or wraps the IDS components in extra

functionality to provide greater security. In fact, it should be possible to convert many existing IDSs to meet our model without altering their core functionality or architecture.

Any distributed hierarchical IDS that has any of the architectures shown in Figure 2.1 can be transformed to work using our model. To do this, label all of the IDS sensors as children and label all of the IDS analyzer components as critical. All the leaf nodes in an hierarchy will be children and all of the internal nodes will be critical.

All critical IDS components must be modified in several ways. First, they must be wrapped as mobile agents and given the process backup ability described in section 13. Second, critical IDS components must communicate with child components only through a proxy agent as described in section 11. The use of proxy agents is one of the major design enhancements of our model and there is no analogy to IDS components in traditional hierarchical IDS.

Child components will have to be modified such that they do not initiate communication. Also, they need to have the ability to recognize a valid proxy component by using digital signature technology. We must also design secure IDS hosts that can operate on a domain backbone. A critical assumption of the model is that the domain backbone hosts are attack-resistant components. We do not want to contaminate this portion of the network with vulnerable IDS components.

Relatively few modifications to the IDS components are therefore required. For the most part, they continue to detect attacks and communicate in their normal fashion. However, the model is more than a software product as it encompasses how one installs the components and on what hosts.

The IDS installation must follow the topology restrictions in sections 8 and 9. Critical and proxy IDS hosts must be given the stealth measures described in section 14. It should be noted that instead of installing different software packages on different IDS hosts, each host is installed to be a generic mobile agent platform that can house any functionality.

17. Provable Attack Resistant Properties of the Model

This model has been designed to protect the critical IDS components of a traditional IDS hierarchy. In this section we present formal proofs that describe the protections afforded critical IDS components. Section 18 describes what protections are lacking in the model and the possible impact these vulnerabilities might have on an actual implementation.

The main proof of this section is to claim that distributed IDSs that follow the model are “attack resistant”. Informally, a distributed IDS that is attack resistant is designed such that its critical components are invulnerable to any attacks but its event gathering or low level components (child agents) are vulnerable. The proofs we give are neither revolutionary nor complex. However, we present them in order to formally explore our assumptions and the logical steps that lead to the rather strong assertion of attack resistance.

Definition of Attack Resistant: A distributed IDS that follows the model is said to be “attack resistant” if it has the following two properties:

1. no critical host can be penetrated or have its location discovered by an attacker,
2. no critical agents can be disabled by an attacker unless the attacker can disable the entire backbone network in that host's domain.

We now present the definitions, axioms, and lemmas that are necessary to prove that a distributed IDS that follows the model is attack resistant.

Definition of a Penetration Attack: Any attack that violates the integrity and confidentiality of a host is called a penetration attack. Also called penetration attacks are those that violate the availability of a host by exploiting a vulnerability instead of simply overwhelming the host with "legitimate" requests.

Definition of a Flooding Attack: Any attack that violates the availability of a host by flooding the host with a large number of "legitimate" requests. In other words, the attack does not exploit any vulnerability in a host besides its inability to process events more quickly.

The properties below are formalized requirements of the model as described in the preceding sections. This list of properties is not complete but merely a selection of the important properties of which we make use in the following lemmas. Very simple properties of the model, like the existence of critical IDS hosts, are implicitly assumed and used by the lemmas.

Property 1: Backbone networks are physically protected such that attackers do not have physical access to any part of the network.
Property 2: All critical and proxy hosts are located in domain backbone networks.
Property 3: Critical and proxy hosts are not penetrable via network attacks that are not associated with the IDS application.
Property 4: The IDS application in critical and proxy hosts is built such that an attacker can exploit no flaws in order to gain unauthorized access.
Property 5: Child hosts may only send data to proxy hosts and nothing to critical hosts.
Property 6: Critical and proxy hosts are not allowed to receive any communication from non-IDS hosts.
Property 7: Critical and proxy hosts can and do mutually authenticate that each are backbone IDS hosts.
Property 8: Attackers can not penetrate non-IDS network elements on a domain's backbone.
Property 9: Communication between backbone networks of different domains can only occur using the enterprise bus.
Property 10: Critical hosts communicate only with critical and proxy hosts.
Property 11: Inter-domain communication traffic on the enterprise bus is encrypted.
Property 12: Any backbone IDS host not already communicating with a non-backbone host will not respond to any packets from that host.
Property 13: Random encrypted communication traffic or padded constant traffic is sent between domains over the enterprise bus to prevent covert channel analysis.
Property 14: All child hosts reside in regions.
Property 15: If an attacker launches a successful flooding DOS attack against a critical host, all critical agents will seamlessly move to another critical host.

We now introduce four axioms. Note that axioms 2 and 4 assume that the attacker does not have insider knowledge about the placement of critical hosts in a network. Thus, an IDS that follows the model is not guaranteed to be attack resistant in the face of an attacker that has extraordinary knowledge of the specific installation.

Axiom 1: There exist only four ways in which an attacker might penetrate a critical or proxy host:
1. through physical access,
2. through network attacks not associated with the IDS application,
3. through non-design vulnerabilities in the IDS application, and
4. through design vulnerabilities in the IDS application.

Axiom 2: An attacker is only able to search for a critical host via sniffing, active port scanning, and host penetration and analysis.

Axiom 3: There exist only four ways in which an attacker might disable a critical host:
1. through penetrating the host,
2. through flooding a host with information,
3. through flooding the network wire from which the host receives packets, and
4. through flooding the entire backbone network.

Axiom 4: It is impossible for an attacker to disable all critical hosts by blindly attacking IP addresses in a domain's backbone set.

Lemma 1: For an IDS that follows the model, attackers can not penetrate any critical or proxy host.
Proof:
By axiom 1, there are only four possible ways in which to penetrate a critical or proxy host:
1. through physical access,
2. through network attacks not associated with the IDS application,
3. through non-design vulnerabilities in the IDS application, and
4. through design vulnerabilities in the IDS application.

By model property 2, all critical and proxy hosts are located in domain backbones. By model property 1, no attackers are able to obtain physical access to the backbone hosts. Therefore, attackers can not penetrate a critical or proxy host that follows the model by using physical access.

By model property 3 critical and proxy hosts are not penetrable via network attacks that are not associated with the IDS application. By model property 4, the IDS application is built such that an attacker can exploit no flaws in order to gain unauthorized access. By model property 5, non-critical IDS may only send data to proxy IDSs and nothing to critical IDSs. Thus, by this design requirement it is impossible for a non-critical IDS to take advantage of a design flaw in order to penetrate a critical or proxy IDS. Furthermore, by model property 6, critical IDSs are not allowed to receive any communication from non-IDS hosts and thus could not be penetrated from these hosts.

However, one remaining vulnerability must be addressed. If a non-IDS host or a child host pretends to be a critical host then it could send instructions to a critical host and thereby penetrate it. However, by model property 7, critical and proxy hosts can and do mutually authenticate that each are backbone IDS hosts.

Lemma 2: For an IDS that follows the model, attackers can only control network elements or hosts on the enterprise bus and in regions.
Proof:
By model property 8, attackers can not penetrate non-IDS network elements on a domain's backbone. By lemma 1, attackers can not penetrate critical or proxy hosts. By model property 2 and 14, only critical and proxy hosts are installed in a domain's backbone. Therefore, an attacker can not penetrate any host or network element in any domain's backbone. By the definition of enterprise and domain, an organization's network consists of three parts: the enterprise bus, backbones, and regions. Since attackers can not penetrate any network elements or hosts in any backbone, all attackers must control hosts in the enterprise bus or regions.

Lemma 3: For an IDS that follows the model, attackers can not discover the location of any critical host.
Proof:
By axiom 2, an attacker is only able to determine the location of a critical host via sniffing, active port scanning, or host penetration and analysis.

First we show that attackers can not determine the location of a critical host by sniffing network traffic. By lemma 2, an attacker can only control hosts or network elements in the enterprise bus or in regions. By model property 10, critical hosts communicate only with critical and proxy hosts. By model property 2, all critical and proxy hosts reside in backbone networks. Furthermore, communication between backbone networks can only occur on the enterprise bus by model property 9 and by model property 11, that communication is encrypted. Therefore, it is not possible for an attacker to discover the location of a critical host by using sniffing.

Next, we show that attackers can not determine the location of a critical host by active network probing. As noted above, critical hosts are installed only in backbone networks and attackers can only control hosts in regions or the enterprise bus. If an attacker launches a port scan from a region or the enterprise bus, by model property 12 any backbone IDS host not already communicating with the scanning host will not respond to any packets from the scanning host. If the scanning host lies about its location and pretends to be a backbone host, the scanning host can not see the replies and thus can not determine whether or not a backbone IDS host exists at the scanned location. However, if the reply traverses the enterprise bus it will be encrypted but an attacker might use covert channel analysis to determine when a scan is receiving a response. Model property 13 counters this weakness and prevents attackers from gaining any information by monitoring the encrypted traffic on the enterprise bus. Thus, it is impossible for attackers to discover the location of critical host by active network scanning.

Lemma 4: For an IDS that follows the model, attackers can not disable a critical agent using a flooding attack unless the attacker can successfully flood an entire backbone network.

Proof:
By axiom 3, there exist only four ways in which an attacker might launch a successful flooding attack against a critical host:
1. through penetrating the host,
2. through flooding a host with information,
3. through flooding the network wire from which the host receives packets, and
4. through flooding the entire backbone network.

An attacker can not penetrate a critical host by lemma 1. By lemma 3, attackers can not discover the location of critical hosts. However, an attacker can blindly launch attacks. If an attacker does launch a successful flooding DOS attack against a critical host or the network segment housing that host, then by model property 15 all attacked critical agents will seamlessly move to another critical host. Again by lemma 3, since no attacker can discover the location of any critical host, the attacker will not know where the critical components have moved. Thus, for an attacker to disable a critical IDS component, the attacker would have to disable all critical hosts in a domain without knowing what IP addresses house those hosts. Axiom 4 claims that it is impossible for an attacker to disable all critical hosts by blindly attacking IP addresses. Therefore, an attacker can not launch a successful flooding attack against any critical agents unless the attacker can successfully flood an entire backbone network. Note, however, that an attacker can successfully flood a specific critical host but the critical agents on that host will survive.

Attack Resistance Theorem for IDSs: Distributed IDSs that follow the model are attack resistant.
Proof:
A distributed IDS that follows the model is said to be "attack resistant" if it has the following two properties:
1. No critical host can be penetrated or have its location discovered by an attacker
2. No critical agents can be disabled by an attacker unless the attacker can disable the entire backbone network in that host's domain.

Lemmas 1 and 3 state that if a distributed IDS satisfied the model, that the IDS satisfies the first property required for attack resistance. As far as property 2, lemma 4 ensures that no critical agent can be disabled by flooding attacks unless the attacker can disable the entire backbone network in that host's domain. However, critical agents could still be disabled by an attacker penetrating a critical host but by lemma 1, that is impossible. Therefore, any distributed IDS that follows the model is attack resistant.

18. Weaknesses in the Model

Despite the provable attack resistant properties of IDSs that follow the model, there still exist several ways that an attacker could disrupt the operation of such an IDS. These attacks can isolate domains, cut off communication between critical and child hosts, and penetrate and control child hosts.

<u>Isolating Domains</u>
If organizations use insecure communication channels for inter-domain communication, like the Internet, then attackers can disrupt the communication between IDS agents in different domains.

Even if the organization uses virtual private networking and firewalls, attackers on the enterprise bus can launch flooding DOS attacks against an organization's firewall. Such an attack, launched with sufficient resources, can cut off inter-domain communication. However, these attacks can be prevented by using dedicated lines for communication between domains. This prevents flooding DOS attacks from outside the organization but still allows attackers in the organization to attack inter-domain routers or firewalls unless the dedicated line bandwidth is greater than the sum of the attacker's region's bandwidths.

Cutting Communication Between Critical and Child Hosts

In the model, the hosts in a region should only have knowledge of the existence of one proxy host, the one serving that region. Attackers in a region can easily discover the location of that host and disable it with a flooding DOS attack. However, the model assumes that many proxy hosts exist (ideally one per region) such that an attacker can only attack the IDS services in the region in which the attacker already has influence. An attacker can not disable the entire organization's IDS unless the attacker controls hosts in the majority of the organization's region. If an attacker can penetrate that many regions undetected, then the IDS is somewhat inoperable anyway. In such a case, an attacker may not need to disable the organizations IDS and thus give away his presence with an obvious flooding DOS attack.

Penetration of Child Hosts

If IDS sensors are installed on insecure hosts, then our model does nothing to prevent their penetration. Instead, the model ensures that the integrity of the organization's distributed IDS as a whole is not compromised by the penetration of some child hosts. The model can not further protect child hosts since we assume that organizations will want to install IDS sensors on relatively insecure hosts. This is often done with host-based IDS sensors and while one should attempt to secure each IDS host, this may prove practically impossible for many installations.

19. The Benefits of Using Mobile Agents

This IDS model is not primarily a "mobile agent" model but it does make extensive use of mobile agent technology. This technology is crucial to the architecture because it provides the backup capability for critical agents. Furthermore, this technology can allow critical components to randomly move around hosts in a critical set so that even attackers with inside knowledge can not predict which components are running on which critical hosts.

One may argue that this capability could be achieved without using mobile agent technology, but such a person is assuming that an implementation of mobile agents has to use mobile code. Mobile code systems transfer the instructions and state of running programs from one machine to another. Mobile agent systems move running processes from one machine to another. It is not necessary to actually transmit instructions in a mobile agent implementation if the destination machine already has a copy of the program. In mobile agent systems, all that is necessary to transfer between hosts is the state of the moving process. Thus, the process backup solution is by definition a mobile agent solution.

20. Creating a Secure System Using Mobile Agents

Many people avoid mobile agent technologies because they believe them to be inherently insecure. This common perception has arisen because many people want to use mobile agents for

applications that involve multiple users, like in e-commerce, that do not necessarily trust each other. There are many problems with such applications: malicious agents can attack agent platforms, agent platforms can modify agents, and agents can attack other agents. In such applications mobile agent security is difficult to achieve.

However, one can implement a secure mobile agent application by creating a “closed” mobile agent system. If each mobile agent platform is owned by a single organization, well maintained to prevent penetration, and only runs mobile agents digitally signed by the organization’s security officer, security is much less of a problem. Nevertheless, there still exists a security threat with this type of architecture. If one mobile agent platform is penetrated, it could flood the other platforms with agents. A more subtle threat is that a penetrated platform could send validated agents to other platforms where the agent’s state has been modified such that they can perform malicious actions. It is because of these threats that in our model, we do not allow agents from untrusted regions to move to critical or proxy IDS hosts.

21. Conclusion

The area of creating IDSs that are resistant to flooding DOS attacks is neither well researched nor understood. Many believe that nothing can be done to prevent flooding DOS attacks and so they do not expend resources investigating this area. However, our model is a counter-example to this argument and proves that at least one effective method exists for designing distributed IDSs that are resistant to flooding DOS attacks.

Our model resists flooding DOS attacks using a passive response system. Instead of actively trying to stop an attackers actions, our IDS model attempts to hide IDS components and move them away from harm. Thus, our IDS components become invisible to an attacker’s normal means of seeing in a network: passive sniffing, active network monitoring, and host penetration and analysis. IDS components become invisible by using assumptions about the network topology and by restricting the communication allowed between certain types of components. In the event that a critical component is attacked, the component moves to an operational host. While it may appear impossible for an agent to move from an attacked host, we use mobile agent technology to enable a type of backup system for processes. Thus, the agents on attacked hosts can become disabled and mobile agents on other hosts will automatically pick up the disabled component’s duties.

The passive nature of this solution gives our model several advantages. Unlike other existing solutions, to implement this defensive technique, one does not have to control all the routers in an organization or to degrade the efficacy of those routers for the sake of security. There is no concept of responding to an attack other than to evade or hide components, which means that there is no concern that the IDS could launch responses that might harm legitimate network traffic. Also, since our model prevents attacks by making IDS components invisible, there is no issue of how quickly our IDS can respond to and mitigate an attack against itself.

Despite these advantages and other provable characteristics, our model is not a silver bullet solution as revealed by the weaknesses described in section 18. More research needs to be done to explore different passive resistance models and to compare them with the active resistance models presented in section 6.

In conclusion, we envision IDSs of the future playing an increasingly prominent role in securing organizations' networks, both from the detection and response arena. Because of this, attackers will attempt to disable IDSs before penetrating more valuable resources. We see flooding DOS attacks as the primary threat, since future IDS back end systems are likely to be highly resistant to penetration attacks. Therefore, researchers need to explore and compare system models for resisting these attacks and vendors need to implement them before flooding DOS attacks become a serious problem.

22. References

[1] Axent, Intruder Alert, http://www.axent.com/Axent/Products/IntruderAlert.

[2] Jai Balasubramaniyan, Jose Omar, Garcia-Fernandez, E.H. Spafford, and Diego Zamboni, Architecture for Intrusion Detection using Autonomous Agents, Department of Computer Sciences, Purdue University, Coast TR 98-05, 1998, http://www.cerias.purdue.edu/projects/aafid.html.

[3] CERT advisory CA-2000-01 on Tribal Flood Network and Stacheldraht denial of service attacks. http://www.cert.org/advisories/CA-2000-01.html

[4] CERT advisory CA-98.01 on the smurf denial of service attack. http://www.cert.org/advisories/CA-98.01.smurf.html

[5] Cisco, Net Ranger, http://www.cisco.com/univercd/cc/td/doc/product/iaabu/netrangr.

[6] Internet Security Systems, Real Secure Product, http://www.iss.net/prod/rs.php3.

[7] Kathleen A. Jackson, Intrusion Detection System (IDS) Product Survey, Los Alamos National Laboratory report, 6/25/99, http://lib-www.lanl.gov/la-pubs/00416750.pdf.

[8] Lawrence Livermore, Security Profile Inspector for Networks, http://ciac.llnl.gov/cstc/spi/spinet.html.

[9] Peter Mell and Mark McLarnon, Mobile Agent Attack Resistant Distributed Hierarchical Intrusion Detection Systems, Second International Workshop on Recent Advances in Intrusion Detection, September 7-9, 1999 at Purdue University.

[10] Network Associates Inc., Active Security Product, http://www.nai.com/asp_set/products/tns/activesecurity/acts_intro.asp.

[11] NMAP network mapping utility, http://www.insecure.org/nmap

[12] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, D. Zerkle, GrIDS -- A Graph-Based Intrusion Detection System for Large Networks. The 19th National Information Systems Security Conference, pp. 361-370, October 1998.