Project Number:	IST-2002-002154
Project Title:	Distributed Adaptive Security by Programmable Firewall

DIADEM Firewall

Response Requirements Specification

Deliverable Type:	Document
Dissemination:	Public
Contractual date:	June 2004

Editor:	Sherif Yusuf, Imperial College, (sy99@doc.ic.ac.uk).
File Name	WP4-D4-Response-Specification
Contributors:	See list of authors
Version:	Final
Version Date:	15 July 2004
Deliverable Status:	Deliverable D4

The DIADEM Firewall consists of:

	Partner	Short name	Country
1	France Telecom	FT	France
2	University of Tübingen	TU	Germany
3	IBM Research GmbH Zurich Research Laboratory	IBM ZRL	Switzerland
4	Imperial College London	ICL	United Kingdom
5	Jozef Stefan Institute	JSI	Slovenia
6	Groupe des Ecoles des Télécommunications	GET	France
7	Polish Telecom	TP	Poland

Project Management:

Yannick Carlinet (FT)
Phone +33 2.96.05.03.25
Fax: +33 2 96 05 37 84
E-mail yannick.carlinet@francetelecom.com
France Telecom DAC/R2I
2 ave. Pierre Marzin,
22307 Lannion, France

List of authors:

Yannick Carlinet
Dan Chalmers
Falko Dressler
Naranker Dulay
Wayne Luk
Emil Lupu
Olivier Paul
Morris Sloman
Sherif Yusuf

Executive summary

This document details the requirements specification for the proposed Diadem distributed firewall. The document discusses techniques and methods that can be employed in response to distributed denial of service attacks. We outline the required procedure of carrying out an automated response, then, discuss specific techniques that will be used in this project. The techniques will take advantage of our distributed firewall architecture to combat the deficiencies of current practices to responding to distributed denial of service attacks.

Acronyms

ACC	-	Aggregate-based Congestion Control
ACL	-	Access Control List
CACL	-	Classification Access Control List
CAR	-	Committed Access Rate
BOOTP	-	Bootstrap Protocol
DoS	-	Denial of Service
DDoS	-	Distributed Denial of Service
DHCP	-	Dynamic Host Configuration Protocol
DSCP	-	Differentiated Services Code Point
ECN	-	Explicit Congestion Notification
FIB	-	Forwarding Information Base
FPGA	-	Field Programmable Gate Array
FTP	-	File Transfer Protocol
IBGP	-	Interior Border Gateway Protocol
ICMP	-	Internet Control Message Protocol
IDS	-	Intrusion Detection System
IETF	-	Internet Engineering Task Force
INF	-	Ingress Network Filtering
IP	-	Internet Protocol
IPFIX	-	IP Flow Information Export
IPPT	-	IP Path Tracing
ISP	-	Internet Service Provider
LACC	-	Local Aggregate-based Congestion Control
NIDS	-	Network Intrusion Detection System
NOC	-	Network Operation Centre
PSAMP	-	IETF Packet Sampling
QoS	-	Quality of Service
RED	-	Random Early Detection
RFC	-	Request For Comment
TCP	-	Transmission Control Protocol
UDP	-	User Datagram Protocol
URPF	-	Unicast Reverse Path Forwarding

TABLE OF CONTENTS

References

[Adl02]	Tradeoffs in Probabilistic Packet Marking for IP Traceback, Micah Adler, 34th ACM Symposium on Theory of Computing (STOC), 2002.
[AT&T]	http://biz.yahoo.com/prnews/040601/nytu051a_1.html
[Bal03]	Using Specification-Based Intrusion Detection for Automated Response, Ivan Balpin et al. 2003
[Beh02]	Tracking Attacks, Michael Behringer, Presentation, Cisco Systems, 2002.
[Bel03]	ICMP Traceback Messages, version 4. Steve Bellovin, Marcus Leech and Tom Taylor. Internet Draft. January 2003.
[Bur00]	Tracing Anonymous Packets to their Approximate Source, Hal Burch and Bill Cheswick, LISA XIV, December 2000.
[Cla03]	IPFIX Protocol Specifications, B. Claise, M. Fullmer, P. Calato, R. Penno, IETF Internet draft, January 2003, http://www.ietf.org/html.charters/ipfix-charter.html
[Cla04]	Packet Sampling (PSAMP) Protocol Specifications, B. Claise, IETF Internet draft, February 2004, http://www.ietf.org/html.charters/psamp-charter.html
[Cis00]	Unicast Reverse Path Forwarding, Documentation, Cisco Systems, 2000.
[Cis02a]	Phase 5 – Reacting to the Attack, Documentation, Cisco Systems, 2002.
[Cis02b]	Ethernet Rate Limiting for the Cisco ONS 15327 Multi-Service Provisioning Platform using the Cisco Catalyst 3550 Series, Cisco Systems, 2002.
[Cis03]	Border Gateway Protocol, Cisco Systems, December 2003.
[COLT]	http://www.securite.org/presentations/ripe46/COLT-RIPE46-NF-MPLS-TrafficShunt-v1.ppt
[Dea01]	An Algebraic Approach for IP Traceback, Drew Dean, Matt Franklin and Adam Stubblefield, IEEE INFOCOM'01, March 2001
[Deb99]	Hervé Debar, Marc Dacier, and Andreas Wespi, "Towards a Taxonomy of Intrusion Detection Systems", Computer Networks, vol. 31, pp. 805-22, 1999.
[Dep02]	A Framework for Incident Response (Draft), Information Security Team, December 2002.
[Doe00]	Using Router Stamping to Identify the Source of IP Packets, Thomas W. Doeppner, Philip N. Klein and Andrew Koyfman, ACM Computer and Communications Security Conference, November 2000.
[Duf00]	Trajectory Sampling for Direct Traffic Observation, N. G. Duffield and M. Grossglauser, ACM SIGCOMM'00, October 2000.
[Du02a]	Trajectory Engine: A Backend for Trajectory Sampling, N. G. Duffield, A. Gerber and M. Grossglauser, IEEE NOMS'02, April 2002.
[Du02b]	A Framework for Passive Packet Measurement, draft-duffield-framework-papame-01, Nick Duffield, Albert Greenberg, Matthias Grossglauser and Jennifer Rexford, Feb. 2002.
[Fer98]	Defeating Denial of Service Attacks which employ IP Source Address Spoofing, Cisco Systems, January 1998.
[Fer00]	RFC 2827 – Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Address Spoofing, P. Ferguson, D. Senie, May 2000.
[Flo01]	Pushback Messages for Controlling Aggregates in the Network, Sally Floyd, Steve Bellovin, John Ioannidis, Kireeti Kompella, Ratul Mahajan, Vern Paxson, Internet Draft, July 2001.
[Gle03]	A Summary of DoS/DdoS Prevention, Monitoring and Mitigation Techniques in a Service Provider Environment, Michael Glenn, SANS Institute, August 2003.
[Goo02]	Efficient Packet Marking for Large-Scale IP Traceback, Michael Goodrich, Ninth ACM Conference on Computer and Communications Security, November 2002.
[Haw]	Intrusion Detection FAG: Network Intrusion and use of Automated Responses, Dan Hawrylkiw, SANS Institute.
[Lai00]	How To Guide – Implementing a Network Based Intrusion Detection System, Brain Laing, Internet Security Systems, 2000.

[Ioa02]	Implementing Pushback: Router Defence against DDOS attacks, John Ioannidis, Steven M. Bellovin, NDSS, February 2002.
[Iss01]	RealSecure 6.5 FAQ, Internet Security Systems, 2001.
[Kos99]	Responding to Intrusions, Klaus-Peter Kossakowski et al. Carnegie Mellon Software Engineering Institute, February 1999.
[Lar02]	Understanding IDS Active Response Mechanism, Jason Larsen and Jed Haile, www.securityfocus.com/infocus/1540, January 2002.
[Lin03]	The World Wide Web Security FAQ, Lincoln Stein and John N. Stuart, http://www.w3.org/security/faq/, April 2003.
[Lo01a]	Appropriate Response: More Questions Than Answers, Christopher Loomis, www.securityfocus.com/infocus/1516, November 2001.
[Lo01b]	Defenders or Diligantes: An Overview of the Appropriate Response Debate, SecurityFocus, July 2001.
[Mah02]	Controlling High Bandwidth Aggregates in the Network, Ratul Mahajan, Steven M. Bellovin, Sally Floyd and John Ioannidis, Vern Paxson, and Scott Shenker, Draft, 2002.
[Ma01a]	Intention Driven ICMP Traceback, A. Mankin, D. Massey, C.L.Wu, S.F.Wu and L. Zhang, Internet Draft, November 2001.
[Ma01b]	On Design and Evaluation of Intention-Driven ICMP Traceback, A. Mankin, D. Massey, C.L.Wu, S.F.Wu and L. Zhang, IEEE International Conference on Computer Communication and Networks, October 2001.
[Mor]	BlackHole Route Server and Tracking Traffic on IP Network, Documentation, Chris Morrow and Brian Gemberling.
[Osb01]	Building an Incident Response Team, Tia R. Osborne, SANS Institute 2001.
[Par00]	On the effectiveness of probabilistic Packet Marking for IP Traceback under Denial of Service Attack, K. Park and H. Lee, CSD-TR 00-013, Research Report CERIAS, Purdue University, June 2000.
[Par01]	New Protocols to support Internet Traceback, C. Partridge, C. Jones, D. Waitzman and A. Snoeren, Internet Draft, November 2001.
[Pta98]	Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection, Thomas Ptacek, Secure Networks 1998.
[RSN00]	HRL: Hardware Rate Limiting with River Stone Networks, River Stone Networks, 2002.
[San01]	Hardware Support for a Hash-Based IP Traceback, Luis A. Sanchez et al. 2nd DARPA Information Survivability Conference and Exposition, June 2001.
[Sav00]	Practical Network Support for IP Traceback, Stefan Savage, David Wethrerall, Anna Karlin and Tom Anderson, ACM SIGCOMM'00, August 2000.
[Sch99]	Infrastructure for Intrusion Detection and Response, Dan Schnackenberg, 1999
[Sno01]	Hash Based IP Traceback, Alex C. Snoeren et al. ACM SIGCOMM'01, August 2001 – also IEEE/ACM Transactions on Networking (ToN), Volume 10, 2002.
[Son01]	Advanced and Authenticated Marking Scheme for IP Traceback, Dawn Xiaodong Song and Adrian Perrig, IEEE INFOCOM'01, March 2001.
[Spi03]	Honeypots: Definitions and Value of Honeypots, Lance Spitzner, www.tracking-hackers.com, May 2003.
[Sprint]	http://ipmon.sprint.com/pubs_trs/trs/RR04-ATL-013177.pdf
[Sto00]	CenterTrack, an IP overlay network for tracking DoS Floods, Robert Stone, 9th USENIX Security Symposium, August 2000.
[Ta02a]	Introduction to Autorooters: Crackers Working Smarter, not Harder, Matthew Tanase, SecurityFocus, August 2002.
[Ta02b]	Barbarians at the Gate: An Introduction to Distributed Denial of Service, Matthew Tanase, December 2002.
[Tan03]	Closing the Floodgates: DDoS Mitigation Techniques, Matthew Tanase, SecurityFocus, www.securityfocus.com/infocus/1655, January 2003.

[Wal02]	GOSSIB vs. IP Traceback Rumors, Marcel Waldvogel, 18th Annual Computer Security Application Conference, December 2002.
[Yam02]	Active Traceback Protocol, draft-yamada-active-trace-00.txt, T. Yamada, October 2002.

1. Introduction

This document presents the range of responses that could be used in the context of policy-based reaction mechanisms, in the Diadem Firewall project. The document starts by a description of current practices implemented in major ISPs and network operators, and highlights the limitations of such practices. Then an overview of possible procedures in response to attacks is given. Section 3 describes the state-of-the-art attack responses, while section 4 describes preventive counter-measures to some common attacks. Finally, section 5 summarises the limitations of automated response techniques and suggests the approach for the Diadem Firewall project.

1.1. Intrusion Detection

According to [Deb99] intrusion detection systems (IDS) can be classified according to several parameters:

- The detection method (Behaviour Based, Knowledge Based).
- The behaviour on detection (Passive, Active).
- The audit source location (Host Log Files, Application Log Files, Network Packets, IDS Alerts).
- The detection paradigm (State based, Transition based).
- The usage frequency (Continuous Monitoring, Periodic Analysis).

Behaviour based systems usually detect anomalies, by a periodic comparison of the current state of the system with normal system models. On the other hand knowledge based systems, also often called signature based systems look for attack signatures, which are specific patterns usually indicating that malicious or suspicious activities are occurring. As mentioned earlier IDS systems can also be classified according to the source of information they use. For example, when the IDS searches for these patterns in network traffic in promiscuous mode, the pattern is considered as a signature, and the IDS is termed a network based IDS (or NIDS). The information relevant to an IDS depends on precisely what the IDS is aiming to protect. For example, an IDS, which is attempting to detect attacks against FTP servers, will concentrate on the contents of all TCP connections to the FTP ports. Signature based detection simply refers to the fact that the NIDS is programmed to interpret a certain series of packets, or a certain piece of data contained in those packets, as an attack. For example, a signature based NIDS, which monitors web servers, might be programmed to look for the string “phf” as an indicator of a CGI program attack. However most existing IDS are not application specific but try to cover a large spectrum by either looking for signatures of attacks targeting popular services in the case of signature based systems or by using aggregated system description parameters in the case of behaviour based systems.

We now focus on NIDS as they are often involved in DoS scenario detection.

NIDS often use raw packets as the data source, and analyse all traffic as it travels across the network in real-time. Many of these commands are indicative of an attack, whether ultimately successful or not. Due to the real-time capture of network traffic, NIDS can also be used to gather evidence against an attacker.. The information gathered can include, but is not limited to, the signature of the attack, and perhaps sufficient details leading to the identification of the attacker.

A goal for IDS is to detect malicious and suspicious attacks as they are occurring in true real-time and provide faster response and notification to the attack at hand. This real-time notification allows for quick responses in a desired fashion based on a pre-defined policy. Some of the possible responses are discussed later in this document.

A NIDS (whether signature or behaviour based), however, presents two problems [Pta98]:

- Firstly, there is insufficient information in packets read off the wire to correctly reconstruct what is occurring inside complex protocol transactions. A packet in itself is not as significant to the system as the manner in which the machine receiving that packet behaves after processing it. NIDS work by predicting the behaviour of the networked machines based on the packets they exchange. The problem this presents is that passive network monitoring cannot accurately predict whether a given machine is going to see a packet, let alone process it in the expected manner. The main problem with a NIDS is that it is on a completely separate machine from the systems, which they monitor, and is often at a different point on the network. This results in inconsistencies between the NIDS and the machines they are trying to protect.
- Secondly, the NIDS is inherently vulnerable to denial of service (DoS) attacks, which jeopardises its availability. A DoS attack is one which is intended to compromise the availability of a computing resource. A NIDS is passive, and as such, is inherently "fail open" i.e. it ceases to provide protection when it is disabled, whereas, a "fail closed" system is one that leaves the network protected when it is forcibly disabled. If an attacker crashes the NIDS or starves it of its resources, then the rest of the network can be attacked whilst the NIDS is unworkable. For this reason, it is important to provide protection for the NIDS from DoS attacks, despite the fact that it is notoriously difficult to defend against such attacks.

Intrusion detection systems (IDS) look for attack signatures, which are specific patterns usually indicating that malicious or suspicious activities are occurring. When the IDS searches for these patterns in network traffic in promiscuous mode, it is considered a network based IDS (or NIDS). The information relevant to a NIDS depends on precisely what the NIDS is aiming to protect. For example, a NIDS, which is attempting to detect attacks against FTP servers, will concentrate on the contents of all TCP connections to the FTP ports. Signature based detection simply refers to the fact that the NIDS is programmed to interpret a certain series of packets, or a certain piece of data contained in those packets, as an attack. For example, a NIDS, which monitors web servers, are programmed to look for the string "phf" as an indicator of a CGI program attack.

NIDS use raw packets as the data source, and analyse all traffic as it travels across the network in real-time. NIDS examine all packet headers as well as investigate the content of the payload, in order to identify specific commands or syntax used with a variety of attacks. Many of these commands are indicative of an attack, whether ultimately successful or not. Due to the real-time capture of network traffic, NIDS are used to gather evidence against an attacker, and once captured the attacker is unable to remove this evidence. The information gathered can include, but is not limited to, the signature of the attack, and perhaps sufficient details leading to the identification of the attacker.

NIDS detect malicious and suspicious attacks as they are occurring in true real-time and provide faster response and notification to the attack at hand. This real-time notification allows for quick responses in a desired fashion based on a pre-defined policy. Some of the possible responses are discussed later in this document.

A NIDS, however, presents two problems [Pta98]:

- Firstly, there is insufficient information in packets read off the wire to correctly reconstruct what is occurring inside complex protocol transactions. A packet in itself is not as significant to the system as the manner in which the machine receiving that packet behaves after processing it. NIDS work by predicting the behaviour of the networked machines based on the packets they exchange. The problem this presents is that passive network monitoring cannot accurately predict whether a given machine is going to see a packet, let alone process it in the expected manner. The main problem with a NIDS is that it is on a completely separate machine from the systems, which they monitor, and is often at a different point on the network. This results in inconsistencies between the NIDS and the machines they are trying to protect.
- Secondly, the NIDS is inherently vulnerable to denial of service (DoS) attacks, which jeopardises its availability. A DoS attack is one which is intended to compromise the availability of a computing resource. A NIDS is passive, and as such, is inherently “fail open” i.e. it ceases to provide protection when it is disabled, whereas, a “fail closed” system is one that leaves the network protected when it is forcibly disabled. If an attacker crashes the NIDS or starves it of its resources, then the rest of the network can be attacked whilst the NIDS is unworkable. For this reason, it is important to provide protection for the NIDS from DoS attacks, despite the fact that it is notoriously difficult to defend against such attacks.

1.2. Distributed Denial of Service (DDoS)

Denial of Service (DoS) is an attack designed to render a computer or network incapable of providing normal services. The most common DoS attacks target the computer's network bandwidth or connectivity. Bandwidth attacks flood the network with such a high volume of traffic, that all available network resources, as well as operating system resources are consumed, and hence, legitimate user requests cannot be processed.

A DoS attack is characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service. Examples include:

- Attempts to "flood" a network, thereby preventing legitimate network traffic
- Attempts to disrupt connections between two machines, thereby preventing access to a service
- Attempts to disrupt service to a specific system or person

Not all service outages, even those that result from malicious activity, are necessarily denial-of-service attacks. Other types of attack may include a denial of service as a component, but the denial of service may be part of a larger attack. Illegitimate use of resources may also result in denial of service. For example, an intruder may use your anonymous ftp area as a place to store illegal copies of commercial software, consuming disk space and generating network traffic.

In an ordinary network-based denial of service attack, an attacker uses a tool to send packets to the target system. These packets are designed to disable or overwhelm the target system, often forcing a reboot. Often, the source address of these packets is spoofed, making it difficult to locate the real source of the attack. However, preventive measures can be taken against spoofed attacks, as described in Section 4.1.1, where we discuss the URPF technique.

A Distributed Denial of Service (DDoS) attack uses many computers to launch a coordinated DoS attack against one or more targets. Using client/server technology, the attacker is able to multiply the effectiveness of the DoS significantly by harnessing the resources of multiple unaware computers, which serve as attack platforms [Lin03]. In the DDoS attack, there might still be a single attacker, but the effect of the attack is greatly multiplied by the use of attack servers known as *agents* or *daemons*. Typically, a DDoS master program is installed on one computer using a stolen account. The master program, at a designated time, then communicates to any number of *agent* programs, installed on computers anywhere on the Internet. The agents, when they receive the command, initiate the attack. Using client/server technology, the master program can initiate hundreds or even thousands of agent programs within seconds.

Before an attacker can launch a DDoS attack, they might have to gain root or administrator access to as many systems as possible. To gain access, scanning tools like *sscan* are used to probe for systems with specific vulnerabilities. With a list of these systems ready, the attacker uses a script to break into each of them and install the server software.

DDoS attacks targeting major web sites or Internet services have been regularly reported in the press. In February 2000, Yahoo.com, E-Bay.com, Amazon.com were attacked and shut-down for four hours causing an estimated $1.2 billion in economic impact and millions of dollars in lost revenue, according to The Yankee Group. The person responsible for this attack was a 15-year old boy (nicknamed Mafiaboy) who was caught only because he could not restrain from publicly boasting about his "exploit". More recently, in June 2004, an attack targeting the Akamai DNS servers rendered Yahoo.com and Google.com (among others) unavailable during several hours.

1.3. Current Practices for DDoS Detection and Response

This section discusses some of the current techniques used by service providers, network operators, and Internet service providers (ISPs), to deal with distributed denial of service attacks.

DDoS has become a wide spread predicament for service providers and their clients. There are numerous commercial tools developed to detect, prevent, and contain (or eradicate) the effects of DDoS attacks.

In order to detect distributed DoS attacks, monitoring probes are usually placed in nodes around the network, and are used to detect surges or variations in network traffic. An alternative means of detecting an attack is with anomaly detection tools, where the typical usage of the network and its resources are known and any deviation from this triggers an alert. In addition, these tools can be used to identify known attacks.

In responding to DDoS attacks, commercial products, in general, opt for dropping packets, re-routing them or limiting packet rate. Both Cisco [Cis02a] and Juniper [Mor] routers allow users to set the ‘next hop’ address, which can be used for blackhole or sinkhole routing. In addition Cisco has products that can be used to provide an effective rate limiting solution that allows bandwidth to be guaranteed in increments as low as 8 Kbps by rate limiting based on source or destination IP address, MAC address or protocol [Cis02b]. River Stone Networks also provide products that allow service providers the ability to rate limit based on ports, aggregate or per-flow [RSN00].

1.3.1 DDoS Attacks Prevention

Currently in most ISP networks, no specific mechanisms exist to prevent DDoS attacks. Some DDoS-specific mechanisms can be found in new generation routers, but they are very limited and are usually not activated because of the negative impact on the performance of the router. The preventive measures commonly activated are:

- anti-spoofing functions (with uRPF), particularly on edge routers
- protection of the management network (called NOC or Network Operation Centre)
- access-list at the border of the network, in order to hide the IP addresses of the equipment in the network by preventing outside devices to test their existence.

The access-list functions very efficiently prevent attacks against network equipment, since the latter cannot be seen from outside the ISP management platform (the NOC). However additional measures like sink-holing may be necessary to protect against attacks using deflectors.

However, anti-spoofing functions prevent spoofing only to a certain extent (see Section 4.1.1), protection of management network does not prevent DDoS attacks on the ISP clients, nor does access-list functions.

1.3.2 DDoS Attacks Detection

Currently in most ISP networks, there are no detection mechanisms implemented. Detection is performed "manually" by the clients. When a client notices that one of his IP address is being attacked, he contacts his ISP and explains the problem. At this point, the ISP may then react to the attack. The limitation of this method is that someone must constantly monitor the availability of the services at the client side. There must also be an ISP employee available at the time the attack occurred, with the knowledge and ability to deal with the attack. Therefore, currently most of the attacks that start on a Saturday are only dealt with on the next Monday.

1.3.3 DDoS Attacks Response

The most common response mechanism among ISPs and network operators is blackholing (discussed in more detail in 3.3, although, some more sophisticated methods are now being deployed in commercial products. Three kinds of blackhole can be setup:

- *Standard:* a new route is announced and deployed in the network using the Internal Border Gateway Protocol (iBGP). All the traffic destined to the victim is then re-routed to a specific router that discards all the traffic it receives. It can also gather some statistics on the discarded traffic using mechanisms such as Netflow.
- *Remote:* the victim of the attack can announce the new route to the blackhole in the ISP or operator network. This can be done only if the operator trusts the client (and his ability to issue a correct iBGP announcement) and with a prior agreement signed by both parties. This method permits a better reaction time to the attack.
- *Edge:* the traffic is discarded by the edge routers. This prevents flooding the network if the attack traffic is very large. In some case, if the flooding traffic is entering the network through a few edge routers, only these edge routers will discard the suspected traffic. Therefore, in that case some of the traffic will reach its destination, as opposed to discarding everything.

The obvious limitation of blackhole routing is that it does not prevent the DDoS attack, but only discards some of the traffic destined to the victim; hence, it also discards the legitimate traffic thus actually denying the service to the legitimate users. In edge blackholing, it is not possible to assure that the traffic that is let through is legitimate.

A few network operators also provide advanced services to their customers. These advanced services include services like traffic shunting [Colt], [AT&T], [Sprint]. As with black-holing systems, the traffic is redirected to a predefined point in the network. This redirection can be performed through modifications to internal elements routing tables similarly to blackhole routing. However the traffic is then analysed through various means. Legitimate looking traffic is then re-injected in the network toward its original destination while suspicious traffic is dropped. Tunnels between the shunting location and the final destination may have to be used in order to prevent loops in the network.

2. Procedures Within an Organisations in Response to Attacks

2.1. Response Policies

Security policies determine the actions to be taken in response to a range of diverse attacks. These policies need to be written with a specification language which is capable of communicating with all the parties involved in responding to detected attack. The parties involved could be human network administrators, as well as physical network devices, which are required to perform certain actions in order to respond to the attack. The policies must be capable of specifying the actions needed to be undertaken by all parties responding to the attack, at the same we must also be able to distinguish between the attacks which require the attention of specific parties.

This approach will allow us to know what needs to be done in the event of an attack being detected, either whilst it is in progress, or after the attack has been carried out. This also allows us to identify the information that needs to be gathered during the response process.

Having a documented plan will allow the parties involved to conduct training or experiment with response procedures in order to efficiently co-ordinate their activities when responding to an attack.

Priorities need to be determined in terms of the actions to be taken. For example, it may be more important to protect classified or sensitive information than to have operational continuity, and thus such operation will be given a higher priority. The policies should indicate whether it is necessary to acquire approval from management before carrying out certain actions. For example, if one wished to carry on operation as usual in order to acquire information about the attacker, this may compromise the integrity of the system.

A most important phase of the response policies is to obtain legal approval of the response procedure in order to ensure that the policies are legally defensible and does not contradict the company policies [Kos99].

2.2. Characterisation of Attack

Characterisation of attacks is simply a process used to determine how the system has been compromised. It is possible to obtain this from the detection of the attack itself. It is necessary to identify exactly what has been compromised by the attack once access was gained, for example which files have been corrupted or has had data stolen from it. In addition, we need to determine how the intruder gained access to the network and its resources. Finally, we should have a mechanism to determine whether the intruder or their agents are still present in the network, and if not, if the attack was a successful one.

The characterisation of the attack is required in order to aid the decision of the necessary response to the attack, and in prioritising the actual response to the attack [Kos99]:

- If we identify the methods and/or tools used to gain access, so we can learn from these and hence identify the appropriate response to take which are known to be effective against such methods or attacks, or by having a pre-defined approach to containing or eliminating such attacks.
- If we identify the compromised system or service, then for an immediate response to the attack we can remove the system from the network, and restore its integrity before returning it to normal operation. Identifying the compromised system, services, or files are important, as this will indicate what may have been modified, and it would be futile to return the network to normal operation if intruders had modified files (e.g. passwords) that would allow them to gain access to the network later.
- If an intruder can be detected on the network, then there are various responses available. We may decide to allow him to remain on the network in order to gather evidence for legal reasons. If the intruder is attempting to access critical system resources, then it may instead be wise to kill his connection, either by disconnecting the target system from the network, or by making use of other techniques to kill the attackers source connection from network (see Section 3.2.2).

Characterisation of the attack can be obtained by analysing the logs from firewalls, routers or network monitors, even if the attacker manages to obtain administrator privileges and deletes the local system logs to hide information about the attack method or the access methods used. Therefore, if configured properly, the logs generated by these devices can be used to determine intruder activity. However, it should be noted that occasionally the logs from separate individual devices might not reveal any unauthorised activity. Thus, it is important to cross-correlate logs from the different devices (or systems). An intruder might have gained access in incremental stages, so it might be necessary to analyse logs over a period of time (perhaps over a week or a month), in which case the analysis may need to be focused on a particular protocol or denied access messages etc.

Characterisation is also needed to provide response functions like traceback, filtering or traffic redirection with sufficiently precise description of the attack traffic. Required information may include information such as the victim address or prefix, the port or service under attack, potentially true source address or prefix and the type of attack tool if known. It should also include the attack rate, which could be a severity indicator but could also be presented by the number of packets concerned.

2.3. Notification of Attack

There is an obvious need to inform those responsible for responding to an attack of the occurrence of such an event. This includes the network administrators, the firewall components, the routers and any other device or party required to deal with the attack.

The network security policies should include some form of notification mechanism, which specifies who or what needs to be notified in the event of an attack, as well as in what order the notification needs to take place. These policies could vary depending on the particular attack detected and what level of security risk the attack is deemed to be.

The response to the attack is not limited to network devices alone, i.e. routers, firewalls etc., but could require human intervention as well. For example, approval from senior management may be required for a particularly sensitive response to a particular attack.

In addition to the above-mentioned parties, it is also useful to inform organisations which may be under attack through your system by an intruder. In addition, we may wish to inform law enforcement agencies, users, and perhaps the site's ISP etc.

2.4. Logging of Evidence

The information about compromised system(s) and the cause(s) of an attack should be documented and safely stored. This includes the system and network log files, router logs, firewall logs, user files, analysis results etc. These must be carefully collected and catalogued, and securely stored at each stage of the attack analysis.

There are two main reasons for gathering evidence [Kos99]:

- Firstly, to use the information as a means to learn from the experience of both successful and unsuccessful attacks. This can then be used to improve the response techniques and/or approach. Such information can also be used to educate members of staff, who were performing an activity that was mistaken as an attack.
- Secondly, the information gathered can be used for legal reasons, such as for prosecution of the intruder. Hence the importance of having thorough, reliable and convincing evidence against the attacker.

2.5. Short-term and Long-term Solutions

Short-term and long-term responses indicate which actions need to be performed immediately after an alert in order to contain or limit the extent of damage by an intruder, and those which need to be performed in order to protect the network from the same or similar attacks in the future [Kos99].

Short-term solutions are required for containment of an intruder, so as not to further compromise the system. In taking such an approach, we must make decisions which relate to our policies. Such decisions include shutting down the system, but this may require management approval, as it could have economic or business reputation consequences, depending on the services being provided. In addition, we may wish to outline certain intruder actions that are deemed too risky to wait for management approval. In such cases, we will have to incorporate some immediate actions in our security policies.

Long-term solutions are just as important as short-term one, and both solutions must work together. Complete eradication of the root cause of an attack is a long-term goal, which can only be achieved by implementing continuous security improvement processes. After a specific attack, we must ensure that all loopholes utilised should be closed in order to ensure that the same or similar attacks are not possible in the future.

It is common for intruders who have achieved a successful attack to install back doors or other means for obtaining future access to the compromised system. Hence, it is important to take steps to identify and remove any modifications made or programs installed. Some examples of such steps include changing system passwords on all systems the attacker may have had access to, restore modified files, restore the original software configuration, or it may even be necessary to review our detection mechanism to better enable reporting and detection of attacks.

2.6. Learn from Experience

It is important to learn from the successful and unsuccessful actions taken in response to an attack. Determining what worked well and what did not, will help reduce the likelihood of similar attacks in the future, and will help to improve our security operations. Failure to learn from the experience will result in continued risky operations, and we will very likely have to contend with the same or similar type of attack again.

Every successful attack indicates a weakness in the system or network [Kos99], and thereby provides a chance for improved security, based on anything gained from such experiences. There are several additional tasks, which need to be performed in the aftermath of an attack.

We must consider that the learning experience should not be limited to human interaction only, and we should try to incorporate automated learning strategies as well. Human forms of analysing flow or response policies are time consuming and do not provide the quickest update to security policies. On the other hand, an automated learning process will enable us to learn from the experience of responding to an attack at a much faster rate and may even be more accurate, as the data that has to be analysed can be numerous and will require much human effort, resulting in a susceptibility to human error.

3. Attack Responses

3.1. Traceback

Traceback represents the basis or the starting point of any (or most) form of response to a DDoS attack.

Tracking techniques are used to find the source of packet or flow aggregates. From a functional point of view, tracking techniques will provide a description of an aggregate, the identity of the source or sources that generated this aggregate. The notion of identity used in tracking techniques changes depending on the environment to which it is applied:

- When applied to the whole Internet, the identity is understood as the true IP address and/or physical location of the network device producing an aggregate.
- When applied to a single administrative domain, the identity is understood as the peering or entry point through which an aggregate enters the administrative domain.

Tracking techniques are useful for two main reasons:

- Source address spoofing. As routing in the Internet is usually performed using only the destination address. Attackers can change the source address of IP packets without altering their ability to reach their destination. As a result using the source address is not always sufficient to determine the true identity of the packet's sender.
- Asymmetric routing. As routing tables can be configured to route packets asymmetrically, knowing the path from a node A to a node B does not necessarily provide any information regarding the path from B to A. As a result, further techniques have to be used to find entry points for aggregates.

During the last few years tracking has attracted a lot of attention from the research and standardisation communities. Existing work can be classified in two main classes:

- Flow extension techniques attempt to add additional information to aggregates when they travel across network elements in a network. This additional information can be recovered at extremities in order to discover which elements were crossed. The efficiency of flow extension approaches can be evaluated according to several parameters such as:
  - the number of packets required to build the path to the attacker,
  - the accuracy of that path (probability of the path being correct),
  - the number of paths that can be detected simultaneously,
  - the resources (spatial and temporal complexity) used to compute the path at the destination,
  - the resources used to compute the path code in each router, and
  - the need for additional information such as the network topology.

  The flow extension family can be further divided in two sub-classes:

  - Packet marking approaches consist in overwriting one or several fields in randomly captured IP packets in order to store information about the path taken by each packet. Approaches such as router stamping [Doe00], Fragment marking Scheme [Sav00], Extended Fragment Marking Scheme [Son01] and Efficient Packet Marking Scheme [Goo02] are based on hash function properties. Polynomial Packet Marking [Dea01] uses polynomial expressions as well as noisy polynomial equations resolution techniques. [Adl03] suggest using marking frequencies and investigates minimal code sizes for identity coding.
  - ICMP based approaches suggest transmitting information about DOS in additional ICMP packets. These packets should be sent at a much lower rate than packets handled in routers fast forwarding paths. The original ICMP traceback approach was suggested within the IETF itrace working group in 2000 [Bel03]. An extension called Intention Based Traceback and used to prevent the attenuation of path information was later proposed [Ma01b], [Ma01a]. Extensions to cope with traffic reflectors where also suggested [Bar01]. However increasing complexity in the traceback protocol as well as successful deployment of competing approaches led to the end of the working group in 2004.
- Backtracking techniques attempt to find aggregates' origins by tracking back their origin hop by hop. In order to do so neighbouring network elements are transmitted a description of the aggregate and are requested to identify if an aggregate is routed locally. If so, neighbours have to be identified in order to repeat the operation. This operation is performed recursively until identities are discovered. Note that being able to know if a given aggregate is routed locally supposes that a summary of routed flow is made available to the backtracking mechanism. As with flow extension techniques, backtracking can be further refined in two subclasses:

- Reactive techniques work by generating this “summary” on demand when an attack is detected. Reactive approaches have been popular for a long time among network operators. Most techniques use functions such as Access Control Lists, Flow Monitoring (i.e. Netflow) or routing protocol extensions that are available in routers for other purposes ([Cis99], [Jun00], [Beh02]). Tools and architectures such as DOSTrack [MCI97], CenterTrack [Sto00] and ICMP backscatter [UU02] were later developed by network operators in order to automate the configuration of routers. Some network devices manufacturers also developed specific functions (e.g. Cisco Address Source Tracker) in order to simplify tracking activities [Cis02]. Reactive techniques constitute by far the most popular techniques implemented in a large number of operational networks. Building over these techniques, extensions such as Pushback [Ioa02] and [Mah02] were developed to improve the definition of suspect flows as well as to automate the relation between tracking operations and other phases (detection, suppression). These approaches asume the ability for the tracking entity to control network internal devices. Some other approaches such as Backhacking [Bur00] provide similar functions without requiring internal device reconfiguration. Selectively perturbing internal routers and checking the influence of these perturbations on attack flows can do this.
- Preventive approaches work by generating this information independently from the presence of attacks. Existing approaches can be further refined depending whether the information is kept for all flows or for a sample of flows. An example of sampled information is represented by Hash Based Sampling [Duf00], [Du02a] and [Du02b]. Hash Based Sampling is a technique based on the use of hash functions and allowing the same packet to be sampled at several points in a network. This sampled information then allows the reconstruction of the path followed by the packet. Hash Based Tracking [Sno01] is an example of techniques where all packets are recorded by using bloom filters for packet header storage. [San01] presents an implementation of such a technique using FPGA in an OC12 network as well as a hierarchical architecture for tracking operations. Both approaches were considered at IETF respectively by PSAMP and IPPT working groups. Keeping the right "summary" information is today the main issue with preventive approaches as the amount of data to be kept on backbone routers is usually very large, even when specific data structures are used.

3.2. Intrusion Response

3.2.1 Kill Connection

An immediate response to a detected intrusion is to kill the connection of the attacker. This is the most popular and simplest action taken in response to an intrusion. This is achieved by injecting packets in the network, which disrupt the connection between an attacker and its victim. The most common approach to disconnecting a TCP connection is to forge *TCP RESET* packets, and then send them to both ends of the connection [Lar02]. On receipt of the reset packet from one side, the other side will treat the packet as a request from that side to end communication. The reset packet has to be sent with the correct sequence/acknowledgement numbers, otherwise it will be ignored.

The use of reset packets cannot guarantee that the session will be disconnected successfully. One reason for this is that the attacker may not be using an RFC standard TCP/IP stack, or he may have modified the stack to handle reset packets in a different way to that which would allow disconnection through injecting reset packets [Kos99]. In addition, the stack may be configured to accept only the first packet with the correct sequence number and to ignore all others. In such a situation our reset packet may be ignored, since our forged reset packet will be competing with other network traffic, thereby giving no guarantee that the reset packet will be processed at the other end.

3.2.2 ICMP Messaging

Disconnecting a session can only be used on a connection-based protocol such as TCP. Injecting a reset packet cannot disrupt protocols such as UDP, a connectionless protocol, as it does not support transport layer flags. To disrupt such sessions, we can use ICMP error messaging. This is achieved by sending an ICMP error message to the attacker's machine stating that the intended victim's machine is not available. The objective here is to make the attacker believe that the machine he is targeting does not exist, and hopefully the attacker will turn his attention elsewhere.

The chances that the attacker's machine responds to an ICMP error message are quite low, for similar reasons to those for killing a connection [Kos99]. Many attack tools do not use the operating system's TCP/IP stack, and do not abide by the standards.

3.2.3 Firewall Rule Manipulation

Another method of immediately responding to a detected intrusion is to manipulate the firewall filtering specification. Once the intrusion is detected, the source address used to initiate the attack (even if it is forged) can then be included in the filtering rules as a packet that is to be dropped. This can be a short-term solution to the extent that the rule can be removed from the list after a specific time.

The drawback to this solution is that if the attacker recognises that we are manipulating the filter rules, he can spoof someone else's address. This may result in us blocking legitimate packets originating from that host. Even worse, if the attacker does this for multiple networks, it soon becomes apparent that we may disable connections with trusted hosts. In such a scenario, it is likely that we would have to disable this facility in the response system, and hence it will be possible for the attacker to then mount his attack after forcing us to change our response strategy [Lar02].

3.2.4 Block Access to Compromised Systems

If we can accurately determine which systems have been compromised by the intrusion, then one option is to simply isolate the compromised subnet from the rest of the network. This allows users, except those on the isolated subnet, to continue to access other networks. However, if we have incorrectly determined where the intrusion is taking place, then we run the risk of allowing the attacker to carry on with malicious activities on that network and all others connected to it.

Another (extreme) option is to completely isolate the attacked site from the Internet. In which case we run the risk of altering critical information on the compromised system as network use continues locally, as well as limiting access to other networks [Kos99].

3.2.5 Restore System Integrity and Update Security Measures

One way of ensuring that the same intrusion will not repeatedly be carried out in the same way is to make some modifications to the compromised system. One such modification is to reset all system passwords. This is especially useful when there are signs that an intruder might have accessed the password files. Obviously, passwords should be stored encrypted.

Based on the results of the analysis performed after an intrusion, we should be able to determine by what means the intruder gained access and these should be removed.
All files that may have been added, modified, deleted etc. by the intruder should be identified and restored or removed, for example, executable files or binary files.

Determine if there are any network or system vulnerabilities and correct them. This could be achieved by employing available patches from vendors for those vulnerabilities that exploit scripts and tools, which exist within the intruder community.

All security mechanisms, such as firewalls, routers etc, should be reviewed and their configurations should be updated based on what was learnt in response to successful intrusions.

3.2.6 Honeypots

A Honeypot is a resource, which pretends to be an attacked or compromised real target, but is a redundant or isolated resource where the attacker cannot do any real damage. The main goals are the distraction of an attacker and to monitor the attacker’s activity in order to obtain information about the techniques being used or the identity the attacker. Honeypots can be used for a wide range of attacks from encrypted attacks in IPv6 networks to online credit card fraud. Some honeypots are left open to be found by an attacker whereas in other systems, the attacker is reconnected to a honeypot once an attack is detected. It is this flexibility, which gives honeypots their true power, but at the same time makes them challenging to implement and understand.

There are two general categories [Spi03]: low-interaction and high-interaction. Low-interaction honeypots have limited interaction; they normally work by emulating services and operating systems. Attacker activity is limited to the level of emulation by the honeypot. For example, an emulated FTP service listening on port 21 may just emulate a FTP login, or it may support a variety of additional FTP commands. The advantage of a low-interaction honeypot is its simplicity. Such honeypots tend to be easier to deploy and maintain, with minimal pursuant risk. In addition, the emulated services mitigate risks by containing the attacker's activity - the attacker never has access to an operating system allowing him to attack or harm others. The main disadvantages in using low interaction honeypots is that they log only limited information and are designed to capture only known activity.

High-interaction honeypots are different; they are usually complex solutions as they involve real operating systems and applications. Nothing is emulated; attackers are given the 'real thing'. If a Linux honeypot running an FTP server is needed, a complete replica of the real server is built. The advantages of such a solution are two-fold. Firstly, it can capture extensive amounts of information - by giving attackers real systems to interact with, you can learn the full extent of their behaviour. The second advantage is that high-interaction honeypots make no assumptions on how an attacker will behave. Instead, they provide an open environment, which captures all activity. However, this increases the consequent risk of the honeypot as attackers can use this real operating system to attack non-honeypot systems. As a result, additional technologies have to be implemented that prevent the attacker from harming other non-honeypot systems.

Honeypots can help prevent attacks in several ways [Spi03]. The first is against automated attacks, such as worms or autorooters [Ta02a]. These attacks are based on tools that randomly scan entire networks looking for vulnerable systems. If vulnerable systems are found, these automated tools will then attack and take over the system (with worms self-replicating, copying themselves to the victim). One way in which honeypots can help defend against such attacks is by slowing their scanning down, potentially even stopping them. Known as 'sticky' honeypots, these solutions monitor unused IP space. When probed by such scanning activity, these honeypots interact with and slow the attacker down. They do this using a variety of TCP tricks, such as a Windows' size of zero, placing the attacker into a holding pattern. This is excellent for slowing down or preventing the spread of a worm that has penetrated your internal organisation.

The second way in which honeypots can help protect an organisation is through detection of an intrusion via a closely monitored system. Detection is critical; its purpose is to identify a failure or breakdown in prevention. Regardless of how secure an organisation is, there will always be failures, if for no other reasons than the involvement of human actors in the process. By detecting an attacker, you can quickly react to them, stopping or mitigating the damage caused. Honeypots excel at detection, addressing many of these problems related to traditional detection methods. Honeypots reduce false positives by capturing small data sets of high value, capture unknown attacks such as new exploits, and work in encrypted and IPv6 environments.

The third and final way in which a honeypot can help protect an organization is in responding. Once an organization has detected a failure, how can it respond? This can often be one of the greatest challenges facing an organization. There is often very little information on the identity of the attacker, how they were able to breach the system, and how much damage they have done, if any. In such situations, detailed information on the attacker's activity is critical. There are two problems compounding incident response. Firstly, often the very systems compromised cannot be taken offline for analysis, as they are often too critical. The other problem is that even if the system is pulled offline, there is so much data pollution involved that it can be very difficult to determine exactly what the attacker did. Honeypots can help to address both of these problems. Honeypots make an excellent incident response tool, as they can quickly and easily be taken offline for a full forensic analysis, without affecting the day-to-day business operations of the organisation. In addition, honeypots only capture unauthorized or malicious activity. This makes hacked honeypots much easier to analyse than hacked systems, as any data retrieved from a honeypot is most likely related to the attacker.

In general, high-interaction honeypots are the best solution for an effective response. To respond to an intruder, you need in-depth knowledge on exactly what they did, how they breached the system and the tools that were used. To acquire this type of data, the capabilities of a high-interaction honeypot are the ones most needed.

3.3. Distributed Denial of Service Response

3.3.1 Modification of detection strategy

In order to respond to the detection of anomalies or possible attacks, several actions have to be initiated. First, the monitoring environment must be reconfigured to examine the suspicious parts of the network traffic in more detail. Secondly, the intrusion detection strategy must be changed from statistical analysis to rule based analysis. Finally, countermeasures have to be applied if attackers were identified, e.g. the configuration of firewall systems.

This section focuses on the possible (re-)configuration of network monitoring probes following the IP Flow Information Export (IPFIX, cf. [Cla03]) and IETF Packet Sampling (PSAMP, cf. [Cla04]) standards. Because there is currently no complete reference implementation available, only limited information on exact parameter exchange is possible. We intend to specify and implement a protocol for efficiently controlling such network monitors in the Diadem-Project.

a) Configuration of Network Monitoring Probes

Filters and sampling algorithms are utilities which allow one to reduce the amount of network traffic to monitor at an observation point. Qualitative conclusions on suspicious traffic are possible using statistical methods for the analysis of the network traffic. In order to detect anomalies, only partial information on the network utilisation is required. Packet sampling is one of such methodologies to reduce the amount of data packets to analyse.

In the presence of suspicious traffic, packets belonging to such stream have to be analysed in more detail. Therefore, filters must be applied together with the sampling algorithms. To allow the monitoring probes to work all the time at full capacity, the configuration of the filters and the sampling parameters must be adapted to the current network traffic. For example, if no suspicious traffic is present, all packets should go though the packet sampler and the subsequent analysis. The sampling rate should be based on the available resources at the monitor and the analyser. On the other hand, if there are suspect data streams, all packets belonging to these streams should pass through the sampler at the high sampling rates. The sampling parameters for all other packets must be reduced to allow the processing of as much suspicious data as possible.

b) IPFIX parameters (Netflow accounting)

Mandatory

- address of collector (where to export statistic data)
- source interface (where to look for network data)
- netflow template (which information should be exported)

Optional

- export rate (how often / how fast to export statistics)
- timer parameters

Additionally, aggregations might be specified, and, according to the IETF drafts, sampling algorithms. Aggregations of flows are specified to be statistics based on sub-networks. Nevertheless, implementations allow much more complex aggregates, such as all traffic with a particular destination TCP port.

A typical output can look like this:

<#packets> <#bytes> <Source IP> <Destination IP> <Transport Protocol> <Source Port> <Destination Port>

c) PSAMP parameters (packet sampling)

Mandatory

- address of collector (where to export statistic data)
- source interface (where to look for network data)
- psamp template (which information should be exported)

Optional

- filter (one or more filter to apply to received packets)
- sampling algorithm (one or more sampling algorithms)
- export rate (how often / how fast to export statistics)
- timer parameters

A typical output can look like this:

< Source IP> <Destination IP> <Transport Protocol> <Source Port> <Destination Port> <n octets of payload>

d) (Online) Configuration

The templates and configuration parameters can be changed at any time. This applies to both, the IPFIX and the PSAMP specification. A collector must be aware of changed templates to decode received statistic information. So far, there is no configuration protocol specified which can be employed to (re-)configure the measurement probes, e.g. from a management station.

e) Configuration of the data analysis

Together with the modifications at the monitoring probes, the analysis process of the intrusion detection must be reconfigured based on the current behaviour of the network traffic. In the absence of malicious traffic, samples of the packets should be verified using statistical analysis. The majority of the traffic should be analysed employing statistical methods in order to find anomalies pointing to possible attacks.

If attacks are identified or even if suspicious traffic exists, the intrusion detection strategy should be changed. All packets belonging to suspect data streams should be analysed using rule based algorithms. Any remaining resources of the analyser can be spent to the standard intrusion detection of looking for anomalies by employing the statistical methodologies.

3.3.2 Firewall Rule Manipulation

The manipulation of firewall rules as a response to a DDoS attack is similar to that for an intrusion. The only difference is that we might need to modify rules on multiple firewalls.

3.3.3 Black Hole Routing

Blackhole routing allows the administrator to take all malicious traffic and route it to a null IP address and drop it [Tan03]. Administrators at the enterprise level often use this technique, but it is far more effective when employed by the ISPs, as this prevents the malicious traffic from reaching the customer's network.

When an attack is detected, a static route is added to the router to route the packets addressed to an attack destination, to a 'black hole address', which effectively discards them. The problem is that all traffic, legitimate as well as attack traffic will be discarded. A Black hole can also be implemented within the final destination network [Gle03].

A new variation of black hole routing, developed by Cisco, is source address based [Gle03]. This technique works by modifying Unicast Reverse Path Forwarding (URPF). When URPF checks the source address to make sure a prefix exists in the router's tables, it checks that the route back is on a real interface; if the route is to the null interface, then the packet is dropped. The mechanism for the source address based black hole filtering is identical to the destination address based black hole filtering, with the addition that URPF must be configured on the ingress interface of the perimeter router.

3.3.4 Sink Hole Routing

Sinkhole routing is similar to blackhole routing, except that the traffic is sent to an IP address where it is logged for examination. Sinkholes are used to direct and trap traffic in a service provider's network [Gle03]. It can be used to redirect an attack against a customer to the sinkhole for traffic analysis. Sinkholes can be used in conjunction with Black Hole Routing to analyse spoofed DoS/DDoS attacks. The traffic for the sinkhole can be analysed by a traffic sniffer to determine the type of attack. As with URPF, a traceback technique is required to identify the ingress point to the ISP's network.

3.3.5 Shunting

As with sink-hole routing and black-hole routing the traffic is redirected to an analysis location within the operator network. Intermediate traffic analysis devices can then be used to distinguish legitimate from suspicious traffic. A few device manufacturers such as Riverhead networks or Arbor networks provide traffic analysis devices that perform such operations. The interested reader may refer to deliverable D3, which provides a description of such devices. Suspicious traffic is then dropped or rate-limited while legitimate looking traffic is forwarded to its original destination usually by using MPLS or GRE tunnels.

3.3.6 Rate Limiting

Solutions that prevent congestion in the network can also be useful for DDoS attacks, since a successful attack always creates congestion at some point in the network near the victim. Rate limiting is limiting the flow rate on specific flows either within the network or at the edge router. It might be used where DDoS flows cannot be completely isolated from legitimate traffic, so limiting the flow prevents overloading the attacked system and causing it to completely fail. It implies that a DDoS has been detected. It results in legitimate traffic being limited so users may experience degraded service.

Queue management mechanisms can be found in Cisco IOS (since version 12.2), in ALTQ (on FreeBSD, OpenBSD or NetBSD), or in Linux Iproute2. The Cisco IOS contains also a mechanism that performs rate limiting with no queue management algorithm: CAR (Committed Access Rate). It prevents bursts of packets on a specific traffic class on the router. The administrator of the router can set thresholds on the packet rate and several actions can be performed when a threshold is reached, such as dropping the exceeding packets, or setting the IP precedence, QoS or DSCP field in the header of the packet. CAR can also be combined with the ACLs (Access Control List) defined on the router to rate-limit a particular IP flow. Juniper routers offer similar mechanisms with CACL (Classification Access Control Lists). The advantages of such approaches are that they allow mitigating the impact of particular flows. They also are quite flexible since it is usually possible to drop, delay or mark the packets exceeding the rate limit. On the other hand, these mechanisms have a negative impact on the performance of the router on which they are activated.

ACC (Aggregate-based Congestion Control) is a complete method to deal with DDoS, from detection to trace-back and reaction. It is based on QoS mechanisms available on routers, such as RED (Random Early Detection), ECN (Explicit Congestion Notification) or CAR. ACC is composed of two sub-systems, LACC (Local Aggregate-based Congestion Control) and a protocol for routers to exchange information: pushback messages. LACC is used to detect congestion in the queue of a router, and it delays packets in order to rate-limit the flows that are responsible for the congestion. The pushback messages are used by a router to request that upstream routers rate-limit certain flows. This is an interesting approach but to date only a prototype implementation on FreeBSD exists. Router vendors did not show interest in implementing this scheme. A draft was proposed at IETF but it expired in 2002.

Experience has shown that rate-limiting solutions are a pragmatic and efficient way to deal with flooding-based attacks. As such, we may consider using them in the response mechanisms implemented in the project.

4. Preventive Mechanisms

As well as responding to attacks, there are techniques which can be used to help prevent attacks taking place in the first place, by removing packets with fake IP source addresses as these are likely to be used for some form of attack or spam email.

4.1. Ingress Network Filtering (INF)

Many denial of service generation tools generate packets with fake IP addresses in order to hide the origin of the attack. As a result checking the source address at the network ingress point can prevent attackers from taking addresses of hosts located outside of their own network (network A in our example, Figure 1) therefore reducing their ability to hide their identity. Note that, in our example, depending on the nature of network A, attackers may still spoof addresses as long as they belong to network A. As a result, the efficiency of Ingress Network Filtering increases when it is implemented closer to the source of packets to be checked. Note that some types of physical support like shared physical mediums (e.g. Local Area Networks, Cable networks) render this operation more difficult. Ingress Network Filtering [Fer00] is usually quite easy to implement on routers through access control lists. The management of these lists can however become a large task when large number of filters are used or when addresses change regularly (e.g. DHCP dynamic addresses).

An important point regarding Ingress Network Filtering is that it provides a protection to hosts that are located outside the network domain implementing it. For example if we look at our example network (Figure 1) it is easy to check that preventing the attacker from using host B address will protect B but will not provide any protection for any host connected to network A. As a result, Ingress Filtering can mainly be seen as an altruistic technique that provides a limited protection to networks implementing it.

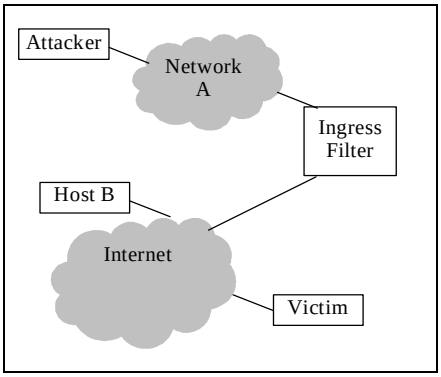

Figure 1. Ingress Network Filtering Location.

However, this measure has several limitations:

Some protocols like mobile IP require routers to route packets that do not originate from hosts with home addresses. A proposal has been made to avoid this problem by tunnelling packets from visiting hosts to their home network, however this proposal is in contradiction with other proposals aimed at optimizing packet paths in the network. Depending on the position of the filter, ingress filtering may cause problems with protocols like DHCP or BOOTP. Ingress Network Filtering becomes efficient only when everybody implements it on the Internet. Ingress Network Filtering is usually useless against encapsulated packets.

Whilst this filtering method does not protect against flooding attacks which originate from valid prefixes (IP addresses), it will prohibit an attacker within the originating network from launching an attack of this nature using forged source addresses that do not conform to ingress filtering rules [Fer00].

4.2. Unicast Reverse Path Forwarding (uRPF)

Unicast Reverse Path Forwarding [Cis00] is proposal to provide a service similar to Ingress Network Filtering but also aims at relieving network administrator from INF management. We explained earlier how current routing operations, routing packets using only the destination address, was a cause for IP address spoofing. The main idea behind uRPF is to overcome this problem by modifying routing operations in routers in order by taking source and destination addresses into account. When "uRPF enabled" a router becomes able to use its forwarding table to check if each incoming packet is coming from the "right" interface. Three notions of a "right" interface for a given packet coexist today:

- In the case of strict uRPF, this interface is defined by the interface to which the packet would have been sent if it had been sent from the destination to the source. When a packet is received by the inbound line-card forwarding engine, input ACLs are first checked, the engine then performs a reverse lookup in the FIB (Forwarding Information Base) in order to see if the packet has arrived on one of the best return path to the source. The engine then performs a regular address lookup for packet forwarding. The packet is then sent to the outbound interface.
- In the case of loose uRPF, this interface is defined by any interface that is included in the FIB for the source address carried by the packet.
- In the case of feasible uRPF, the right interface is an interface including one of the best routes to the source. The best route set includes k routes for which the packet may have come from the source using less hops. For example if we suppose that DoS packets are sent from A to B that k equals 3 and that fives routes R1, R2, R3, R4, R5 with route lengths of respectively 1,3,8,2,4 hops are available on router C for interface C1, the packet will only be forwarded if the inbound interface FIB includes R1, R4 or R2. Alternative routes are stored in the FIB similarly to the best route during the FIB configuration process.

The main advantage of uRPF over Ingress Network Filtering is to prevent lengthy configuration operations in edge routers by implementing an automated filtering process that derives configurations from routing information. Another advantage in the case of older devices is that uRPF is usually implemented in hardware thus preventing performance issues that can occur in the case of ACLs.

Similarly, to INF the efficiency of uRPF depends on the position of the uRPF filter in the network. Moreover, strict uRPF cannot be used with asymmetric routes or multi-homed networks since best paths may differ from one router to the other. On the other hand, loose uRPF does not bear this limitation but provides a limited protection. It is often recommended using strict uRPF in enterprise networks with a single connection to an ISP or in Network Access Servers and loose uRPF in other cases. Similarly to INF, uRPF is not able to handle encapsulated IP packets. Finally, feasible uRPF represents a trade-off between the two other techniques that may work better in some situations. Note that modes are usually vendor specific although most vendors will at least provide one mode.

To restrict forged packets, we must ensure the routers validate the source address, confirming that it has a prefix which exists in the forwarding tables for the interface on the router through which the packet came [Gle03]. If a route to the source does not exist, the packet should be dropped. We should log information on packets which are dropped; this then provides a basis for monitoring any suspicious activity.

5. Conclusion

5.1. Limits of Response Mechanisms

There are bound to be limitations on the effectiveness of any security system, as there are always innovative attack methods and tools available to the determined or expert attackers.

One of the main limitations of the suggested responses in Section 3 is the denial of service to legitimate users. Responses such as blackhole filtering, sinkhole routing, rate limiting etc are all effective means of mitigating the effects of a DDoS attack, but they do not really distinguish between legitimate users of services and attackers and so can affect traffic from the legitimate users.

Another limitation is that, it is probably not possible to develop a firewall that can work on all (or too many) layers of the network protocol. We must restrict our efforts on what we deem the most important layers of operation, for example, just a combination of the network layer, the transport layer, and the application layer.

In addition, we have to consider whether we can tackle some (or any) of the vulnerabilities in the TCP/IP protocol suite, such as the problems of attacks based on fragmentation.

5.2. Choice of Response Mechanisms

In the diadem project, we will focus on response to denial of service attacks rather than intrusions. For this reason we will not make use of honeypots as these are more relevant for dealing with intrusions to high-profile services and are expensive to set up and manage.

In order to make effective use of the Diadem's distributed architecture, we need to think of responses with a distributed approach. We suggest two practices to responding to DDoS attacks: location-based responses or device-based responses.

Location based response is to respond to an attack based on the location of the attacker's entry point, regardless of the location of the victim machine. This takes advantage of the distributed architecture, as the victim may be at a very distant location from where the response to its attacker is taking place, although the detection will still occur at the victim's site.

Device based response is to respond to an attack by selecting what we determine as the best device(s) to handle such an attack. This approach can work in conjunction with the location-based approach, as the selected device(s) can be at the victim's site or at a remote site. This method is based on deciding whether to make changes to a filter rule at some packet-filtering device or to instruct a router to route specified packets to a sinkhole. This decision is important; as there are subtle consequences depending on our decision, for example, dropping packets at the target's end can consume more bandwidth resources compared to routing to a blackhole/sinkhole from a point far from the victim.

Note that correlation of alerts is critical as attacks from multiple agents need to be identified before the system is overloaded. Since an attack is very likely to generate several alerts, it is also critical to correlate these alerts so that only one attack is reported to the System Manager (see deliverable D2). Otherwise, the latter could be too easily flooded with alerts, making it less efficient or even ineffective.

We intend to implement the following:

- Preventive mechanisms for discarding packets with fake IP addresses using INF and UPRF (see section 4).
- Adaptive intrusion detection techniques as outlined in section 3.2.1.

- A traceback technique (see Section 3.1) is required in order to attempt to filter packets as close to the sources as possible.
- Response mechanisms will include
  - Disconnecting a connection
  - Firewall Rule Manipulation
  - Black Hole Filtering
  - Sink Hole Routing
  - Rate Limiting

The assumption is that a distributed response which takes place as close to the sources of attack, within the network as well as at the target site can be more effective to deal with distributed attacks using multiple agents.

