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

DIADEM Firewall

D7 – Initial Demonstrator Specification

Deliverable Type :	Document
Dissemination:	Public
Contractual date :	January 2005

Editor:	Piotr Piotrowski (TP)
File Name:	Diadem Firewall – D7 – Initial Demonstrator Specification.doc
Contributors :	See list of authors
Version :	Final
Version Date :	January 2005
Deliverable Status:	

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	Imperial	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 CORE/M2I
2 ave. Pierre Marzin,
22307 Lannion, France

List of authors:
Piotr Piotrowski, TP
Yannick Carlinet, FT
Olivier Paul, GET
Paweł Tobiś, TP

Executive summary
This document gives a description of the two selected use-cases – TCP SYN flood and Web Server Overloading. In particular, it describes how the different components of the architecture interact with each other during the execution of the use-cases. The document also contains a general introduction on the basics of firewall testing and traffic generation tools. It also gives a preliminary description of the testbed that will be setup and used in the project.

Acronyms

ACK	Acknowledge
ACL	Access Control List
API	Application Programming Interface
DDoS	Distributed Denial of Service
DMZ	Demilitarized Zone
DNS	Domain Name Service
DoS	Denial of Service
HTTP	Hyper Text Transfer Protocol
IPFIX	Internet Protocol Flow Information eXport
ISP	Internet Service Provider
OSI	Open Systems Interconnection
PMA	Policy Management Agent
Seq	Sequence
SMTP	Simple Mail Transport Protocol
SYN	Synchronize
TCP	Transmission Control Protocol
TFN	Tribe Flood Network
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VD	Violation Detection
VDF	Violation Detection Facility
WL	White List

Table of Contents

1 Introduction

The aim of this document is to present our thoughts on what the project demonstrator and the project testbed will be. The purpose of the testbed is threefold:

- validate the results of the project, in particular regarding the performance obtained with the architecture,
- show that it is possible to deploy the solution in an operational network,
- support the demonstrator.

The demonstrator is a key element in the dissemination of the project results. Indeed it will be used to show the main functionalities of the system and the value-added the architecture can bring. To this end, we have selected two use-cases that are described in details in part 2 of this document. Part 3 explains the basics of firewall testing and what the testbed will look like.

2 Detailed presentation of selected use-cases

2.1 TCP SYN flood use-case

2.1.1 TCP SYN flood attack description

The TCP SYN flood attack is one of the most common occurrences of flood-based attacks, as shown in [D3], section 2.5, and is based on the way a TCP session is established between two hosts. The session establishment phase is called the TCP 3-way handshake. The figure 1 shows the TCP 3-way handshake and how the TCP session is opened. The handshake ensures that both sides are ready to transmit data, and that both ends know that the other end is ready before transmission actually starts.

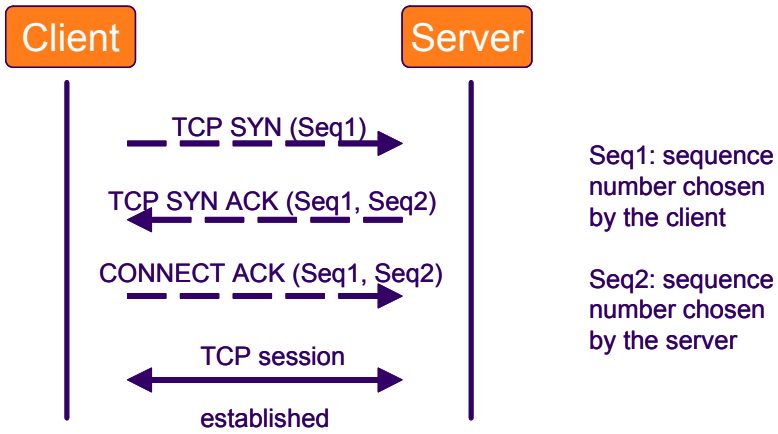


Figure 1: TCP 3-way handshake

This attack consists in overwhelming a host with TCP SYN messages without sending the CONNECT ACK messages. The receiving host queue of incoming TCP sessions (called "backlog") starts to fill up while the host waits for the CONNECT ACK messages. When the backlog is full, the host rejects any further connection attempts, thus denying service for legitimate users. As an example, the backlog size of Sun Solaris 8 is set to 1024 by default. System administrators of servers prone to TCP SYN flood attacks can make their server more resistant by increasing the backlog size or by decreasing the timeout waiting for a CONNECT ACK message. However the backlog size is limited by the memory available on the host, and if the timeout is too low, then the legitimate users with slower connections will be unable to connect.

In the distributed version of the attack, the hacker uses masters in order to coordinate all the compromised hosts (called zombies, or agents) that generate the attack traffic, as shown in Figure 2.

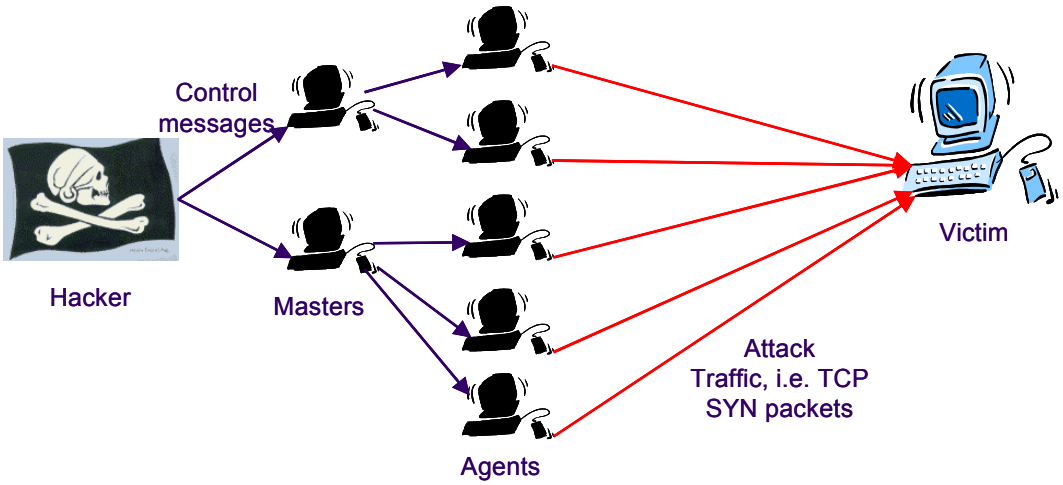


Figure 2: Distributed Denial of Service attack

2.1.2 Use-case setup

The victim machine is a server offering a TCP-based service (such as a web server, or a video game server for instance). Before the use-case starts, we suppose that the following conditions are verified:

- the administrator of the server has subscribed to a protection service, so that the operator knows that the traffic destined to this server must be monitored,
- a hacker has compromised a large set of hosts and has installed all the necessary software to throw an attack on the server.

2.1.3 Detection

When the use-case starts, a detection module (cf. [D5], part 6) is present in the Violation Detection and it collects the monitoring data related to the IP address of the server. The monitoring data are sent by the Monitoring Elements in the forms of IPFIX records. The detection module contains an algorithm that analyses the monitoring data in order to determine if an attack is underway. This algorithm is based on anomaly detection and is further described in [D5]. When the attack is started, it is detected in the detection module and the latter sends an event to the Policy Management Agent (PMA). The PMA then notifies the System Manager through the Notification Module. The event is received in the System Manager by another PMA that holds a policy which instruct the action to be taken. In this case the action is to load a reaction module specialised for TCP SYN flood attacks in all the Firewall Elements.

2.1.4 Reaction deployment

As a consequence of the action taken in the detection phase, the System Manager uses the Service API (cf. [D6]) offered by the Firewall Elements to instruct them to load and execute the TCP SYN flood reaction module. This reaction module will intercept all the traffic with a destination address that corresponds to the protected server. It is further described in the next sub-section.

2.1.5 Reaction function

The main problem that needs to be tackled is to tell the legitimate traffic from the malicious one. When this distinction is made, we enforce the following actions:

- block the traffic for which we are certain it is malicious
- let through the legitimate traffic
- rate-limit the traffic that puts a heavy load on the server, but that can be legitimate

To make things more complicated, the source IP addresses of the attack packets may be spoofed.

The basic idea of the reaction function is first to determine which are the non-spoofed connection attempts to the server. Afterwards we can gather statistics on the flows, based on the source addresses of the clients.

Filtering out of spoofed connection attempts

A way to check whether a connection attempts (i.e. a TCP SYN packet) is spoofed or not is to answer the packet with an ACK, and if a valid CONNECT ACK is received, then we can be sure that the client has not a spoofed source address (provided the sequence numbers chosen by the server cannot be predicted, which is the case in modern Operating Systems). However, doing so would require storing the state of the connection in the reaction module, which would make it vulnerable itself to TCP SYN flood attacks. We therefore describe in the following a way to check the validity of the source addresses of the TCP SYN packets without keeping state information in memory. The method presented here is partially based on 'syncookies' techniques, the difference being that syncookies are implemented on the server side, whereas here it is done in the network, which allows a much greater efficiency as it is shown below.

The reaction function contains a list (called 'white list') of authorized source IP addresses. This white list is empty at the start-up of the reaction function and it will contain the source addresses of the TCP connection attempts with a non-spoofed address. The reaction function generates a private key to use in an encryption function (called θ in the following). The reaction function intercepts all the TCP SYN and CONNECT ACK packets destined to the server. When a TCP SYN packet is seen, the reaction function checks if the source address is in the white list, if yes then the message is let through to the victim, if no then the following operations are performed:

- the TCP SYN message is blocked (let S_{syn} be the source address of the message),
- a TCP SYN ACK is sent back to the originator of the connection attempt, with a sequence number N equals to $\theta(S_{syn})$ – i.e. the encrypted value of S_{syn} (note: nothing is stored about the TCP connection state).

When a CONNECT ACK message is seen (let S_{ack} be the source address):

- the sequence number N is decrypted and the result is compared with S_{ack},
- if they are equal, this address is added in the white list,
- otherwise, the message is discarded.

These operations are summarized in Figure 3 below:

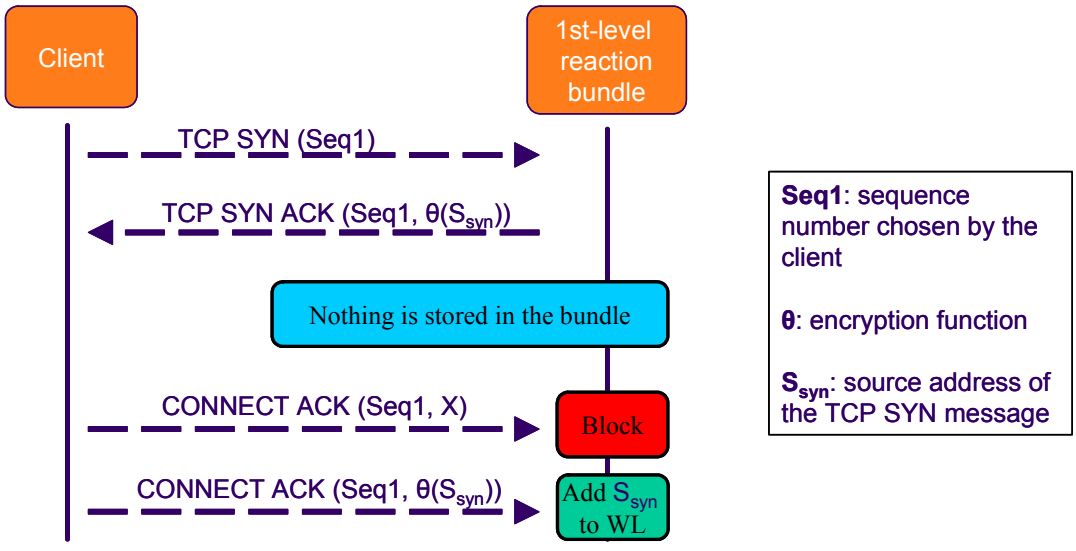


Figure 3: sequence diagram of the first-level reaction bundle

To put it simply, upon reception of a SYN packet, the reaction function challenges the originator of the packet to answer back to its ACK. If the client can answer it means it has received the ACK and therefore its source address was not spoofed. Consequently this source address is added in the white list. This processing ensures that only non-spoofed addresses are included in the white list. Nothing is assumed if no answer was received from the client, since the ACK packet could just have been lost.

As a consequence of this processing, legitimate users will see their first connection attempt fail, but all the other attempts will pass with little overhead since the only processing performed is to check that the source address is in the white list. On the other hand, a standard TCP SYN flood attack (in the sense of an attack generated by a widespread tool such as TFN or Stacheldracht) will be blocked by this reaction function. To workaround this function, a hacker could have the agents send first a real connection attempt, and once their respective addresses are in the white list, they start the TCP SYN flood attack. This would work but in that case the source address of the agents cannot be randomly spoofed, that's why we can then base the attack mitigation on the source addresses, as describe below.

Attack mitigation based on the clients source address
In parallel to the filtering of spoofed IP addresses, the following processing is performed. The reaction function counts the number of TCP SYN messages sent by each host in the white list, over a certain time period. If this number reaches a threshold, the corresponding source address may be an agent performing a flood attack. The reaction function queues up further SYN packets from this source address in a rate-limiting queue. This case will usually happen when a worm is performing the attack (since the attack traffic is not spoofed), so the ISP can notify its customer that his/her computer has been infected.

2.1.6 Use-case termination

When the monitoring, as described in the detection sub-section above, reports that the traffic is back to normal, the policies in the Firewall Elements that were enforced can be disabled. Consequently the reaction functions are stopped. Whether they are only stopped or totally removed is up to each particular Firewall Elements.

2.2 Web Server Overloading use-case

2.2.1 Handling HTTP protocol by Web Server

One of the most popular protocols which are handled by Web Server is Hyper Text Transfer Protocol. HTTP is a request/response application protocol and operates over TCP. A client sends requests to the server, which transmits a reply message back. Request and response forms depend on various parameters like: request method, URI, protocol version, success/error code, client/server information, body/entity content, etc.
HTTP message example is “GET url”, to which the server replies by sending the named document.
Further information about functioning HTTP protocol can be found in [FGM99] and we won’t copy this document here.

2.2.2 Web Server Overloading attack

The attack is organized in several steps:
- One or several legitimate users perform legitimate user requests.
- After some time, several distributed attackers start to request a given object located on the server.
- Sub-Scenario 1. The number of requests performed by attackers is such that the web server is no longer able to serve legitimate users correctly.
- Sub-Scenario 2. The number of requests performed by attackers is such that the web server is able to serve legitimate users correctly however server resources are wasted.

2.2.3 Model generation and Violation Detection System Configuration

Prior to the attack, monitored information from the customer web site (log files, ...) is used to build a set of models for traffic inference and normal traffic characterization. These models can be updated in order to reflect recent changes in monitored web servers. Inference models are used to infer user actions from network level measures while normal traffic models are used to compare user actions to a model of legitimate actions. Models are built as soon as the web server owner provides access to the previously mentioned information and are then updated regularly in order to take new user behaviours into account.

Models are provided to a web server overload detection module by the system administrator. A detection module is thus specific to a web server. Associated with every overloading detection module is an overloading investigation module. This module constitutes the "memory" of what happened to a given server and coordinates the actions to be performed in the case of an attack. Each overloading detection module supports two modes:

- Detection mode. The module is able to identify that an attack is going on for a specific target.
- Identification mode. The module is able to identify legitimate users and report their identity to the detection system.

A module can switch between these two modes when instructed to do so. This is performed through the violation detection management module. Initially a detection module is configured to work in "detection mode". This configuration is triggered by the violation detection policy and performed by the overloading investigation module associated with the detection module. The coordination between these two modules is defined by the violation detection policy. This policy is defined by the system administrator and downloaded in the VD policy database through the service API (load policy) and activated (activate policy).

When launched, the violation investigation module will configure measurement elements/devices in addition to the detection module. This configuration provides the address and port of the web server to be monitored. This configuration is performed through the management module which checks configuration requests for possible interactions with existing policies in the system. Configuration between the management module and the Monitoring Elements is performed using the Netconf protocol. Monitoring Elements use received information to configure the metering process on the monitoring device as well as exporting process. The selection of adequate Monitoring Elements could be performed using a map of the network as well as a description of Monitoring Element capabilities. Monitoring Elements are also configured to report measurement information to the Violation Detection IPFIX collector.

2.2.4 Detection

Monitoring information originating from configured monitoring devices is received by the violation detection IPFIX collector and saved locally. Detection modules select the appropriate IPFIX records based on the server identity. IPFIX records are then used to infer user activities. These activities are compared to a normal behaviour. When suspicious activity is detected a notification is sent to the PMA. This notification contains an attack signature which includes the identification (i.e. address, port) of the server(s) under attack, the severity of the attack (this may be the variation from the attack threshold in the case of statistical models) as well as statistics on attack traffic destined to the victim The signature may also include the identity of the Monitoring Element and detection module involved in detecting the attack. Notifications are sent at regular intervals to the PMA as long as the traffic remains suspicious. The end web service may include their own monitoring and violation detection elements which can generate a notification to trigger a response without waiting for detection to occur in the network.

2.2.5 Response

When receiving a notification, the PMA stores the notification in the event database and forwards the notification to the appropriate investigation module depending on the victim identity. The investigation module sets up an attack context. This context will be used to match other incoming notifications targeting the same victim. These notifications may be generated through the following processes:

- The original Monitoring Element reporting monitoring information regarding the same traffic several times. This monitoring information when reported to a detection module will generate several notifications to the investigation module.
- Several Monitoring Elements reporting monitoring information regarding traffic associated with the same attack. This monitoring information when reported to several detection modules will generate several notifications from each detection module to the investigation module.

After setting up the context, the investigation module reconfigures the overloading detection module in order to switch it to identification mode.

In the cases where address spoofing is demonstrated, when the targeted server is vulnerable to sequence number prediction attacks or when the traffic cannot be analysed in depth on a single device, the detection module may choose to activate the traceback function. In order to do so, the notification sent to the PMA includes a traceback indication. When receiving a notification indicating the need for a traceback function, the PMA forwards the notification to the System Manager, which in turn activates the traceback function. The traceback function should identify the set of VDFs located the closest to the attackers, based on the attack signature. When provided with this set of VDFs, the System Manager configures VDFs so that:

- Remote detection modules work in "identification mode" and target traffic to the victim address.
- Notifications produced by remote detection modules are sent to the local VDF.
- The local VDF PMA forwards notifications produced by remote VDFs to the appropriate local investigation module.

When the investigation module uses the identity produced by the detection modules to identify the top contributors to the attack. For each contributor, it generates a notification to the PMA. This notification is forwarded to the system manager. When receiving such notifications the system manager reconfigures the Firewall Elements closest to the attackers to rate limit the amount of suspicious traffic by using the response interface. Note that several mechanisms may be used to provide the rate limiting service. The rate limiting factor is chosen depending on several parameters including:

- The victim capacity (i.e. the throughput the victim may be able to handle).
- Violation Detection and Monitoring Elements uncommitted resources. Violation Detection and Monitoring Elements must be protected from overloading. As a result the factor may depend on the amount of traffic associated with the attack that can be treated by the architecture.

The amount of traffic dropped by Firewall Elements is regularly reported to the System Manager. This amount is notified to the local investigation module by the System Manager in order to evaluate if the attack is still going on. When the amount of traffic becomes reasonable the rate limiting restrictions are gradually lifted to resume normal operations.

2.2.6 Use-case termination

When notifications related to a given attack context are no longer received over a given period of time, the investigation module generates a notification to the System Manager which restores the prior monitoring state of the system:

- The System Manager must deactivate and possibly remove the attack specific monitoring policies from remote VDFs and disable remote detection modules.
- The System Manager must reconfigure Firewall Elements in order to stop marking or rate limiting operations.
- The local investigation module must delete the attack context and all associated variables and set the local detection module in "detection mode".

Finally the System Manager must provide a report to the DIADEM architecture administrator as well as to the impacted customer regarding the treatment of the attack.

3 Firewall testing and testbed basics

This section gives an overview of firewall testing, please see [CERT01] for more details.

3.1 General tests scope

The following points should be tested:

- hardware layer (for example: processor, hard drive, memory, memory interfaces),
- equipment used for connecting the firewall to a network (wires, switches, routers, etc.)
- operating system (operating system booting, console access),
- software used for configuring the firewall (redirection and filtering rules with violations and garnering statistics options),
- other software run on firewall.

Additionally implementation of security policy should be examined. A proper test plan of a system implementation must contain:
a) a list of all exchangeable modules;
b) a list (for each of the modules mentioned above) of all operating modes, which could perturb the firewall activity;
c) a test scenario designed for each of the operating modes mentioned above.

Testing should focus on:
a) boundary values for the parameters of a given filtering rule. The attributes of a filtering rule could be for example: kind of protocol, addresses, ports, connection state, application type, etc. Initially, all parameters could be independently analyzed
b) testing traffic specification, which allows to check the firewall behavior for both acceptable and ou-of-limit parameters.

3.2 Overview of tests techniques

The following steps are performed:
1. Turn all filtering options off – set „transparent firewall";
2. Generate test traffic for checking the general correctness of the system functioning: cooperation of supportive modules with firewall under test – physical connections, routing, some firewall hardware and software functions;
3. Turn specified filtering options on;
4. Generate test traffic in order to observe the firewall behavior with some parameter values for all the specified filtering rules.

The different test methods can be described with the following categories [CBR03]:

- manual – it relies among other things on looking up source code and firewall access control lists;
- automatic – it is computer aided and allows a fast analysis of a large number of access rules and dependencies between them;
- done in laboratory conditions (initial stage) and
- done in real conditions (final stage);
- functional – tests the enforcement of a given security policy, reaction to threats, way of handling exceptional situations;
- performance – tests the ability to handle large network traffic.

3.3 Tools for testing firewalls

The following tools were identified:
a) tools used for security audit of IT/telecom systems (such as [Pio03], [CERT01] for example):

- network monitors – SmartWhois, AdRem NetCrunch, Essential NetTools,...
- port scanners – Nmap, Netcat,...
- vulnerability assessment in systems and network – Nessus, SecurityCenter, Retina, Ping Scanner,...
- sniffers and network analyzers – EtherPeek, Ethereal, Sniff'em, tcpdump,...

- intrusion detection – Snort, Real Secure, BlackICE, SecurityCenter,…
- network traffic generators – SPAK, ipsend, Ballista,...

b) tools dedicated to checking the correctness of firewall operations:

- Spirent Avalanche & Reflector [SCSW] – Spirent solution can be applied to test the performance of network devices in the application layer (Layer 7), such as filtering routers, firewalls, or proxies. The Avalanche device can generate HTTP, SMTP, or DNS test traffic. The Reflector device simulates network servers for these protocols. Both devices support Gigabit Ethernet.
- Agilent NetworkTester [ANT04] – Agilent solution can be applied for testing the network performance of devices in layers 4-7 of the OSI model. This device can generate test traffic for a wide range of protocols, and simulate the presence of many servers and clients. The NetworkTester device is specialized in performance tests of firewalls and filtering routers.

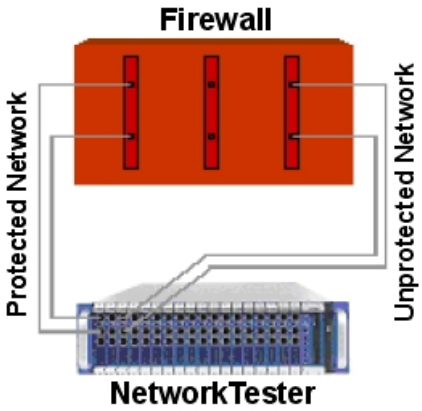


Figure 4: test of a firewall with Agilent NetworkTester [ANT03]

- Agilent RouterTester [ART02] – this tool is used to test the lower layers of the OSI model. It is mainly dedicated for routers. It has for example the ability to capture and decode layers 2-3 protocols.
- Blade Firewall Informer [BFI04] – this tool allows testing the settings of the access rules of a given protocol, IP address, or ports numbers for both traffic directions. This solution is used to assure corporate networks security, but tests can also be performed by end-users;

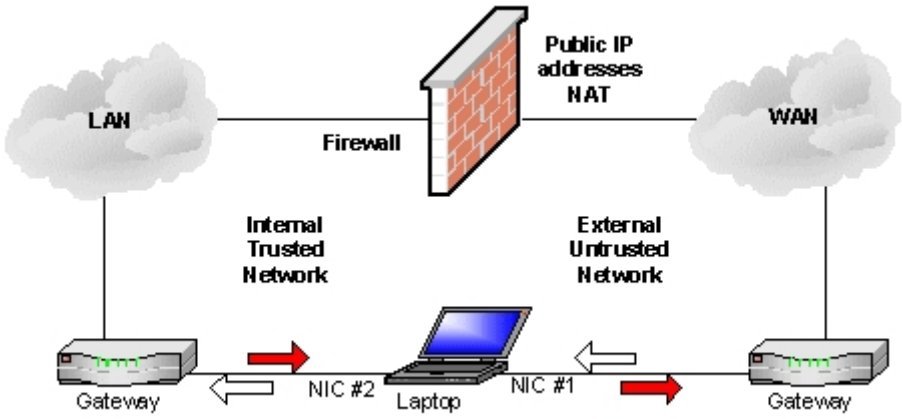


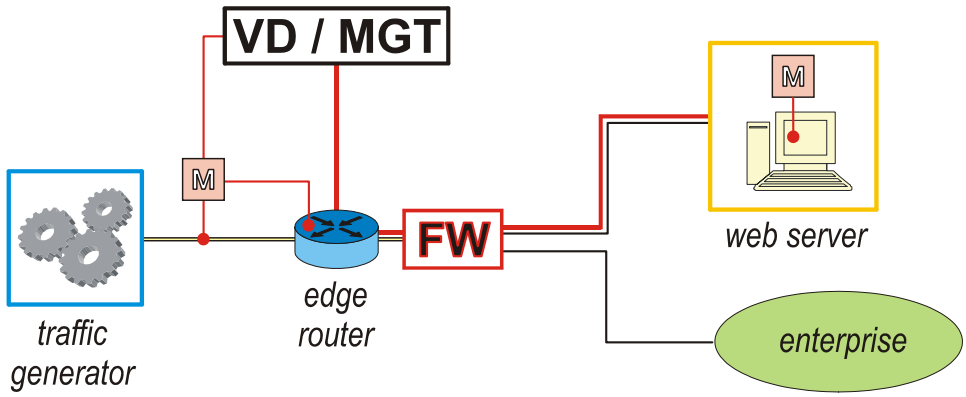
Figure 5: Test configuration with Blade Firewall Informer [BFI04]

Firewall tester [BarFT] – this tool has been written in Perl by Andre Barisani. It is used to test ACL in firewalls and IDS systems.

3.4 Testbed

3.4.1 Testbed for firewalls operating in ISP/Enterprise domain

We initially assume that firewalls have multi-homed architectures – many firewalls with several demilitarized zones (DMZs). Figure 6 shows an example of a firewall with a DMZ which contains a web server.

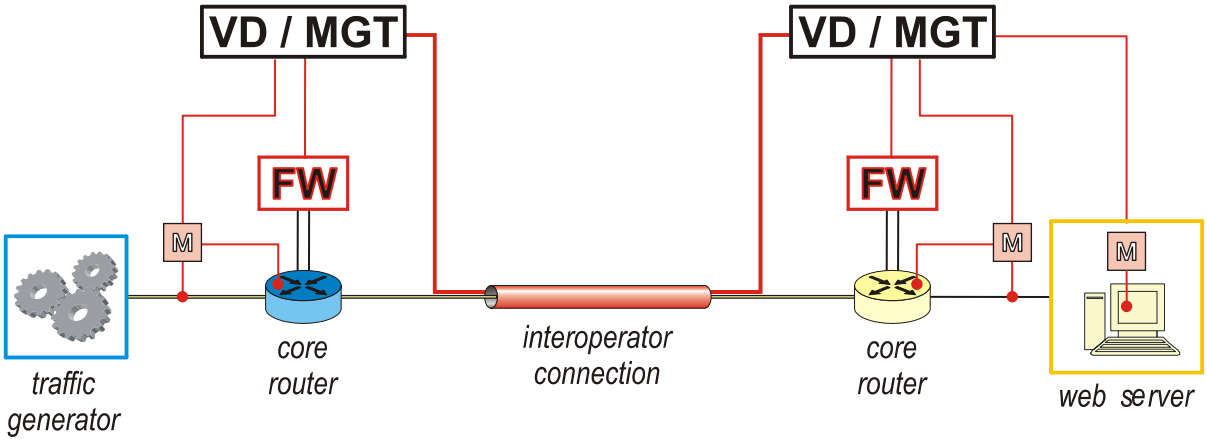


VD/MGT – Violation Detection & System Management **FW** – FireWall **M** – Monitoring Element
Red dots – Monitoring devices **Red lines** – Diadem traffic

Figure 6: testbed for firewalls in ISP/Enterprise domain.

3.4.2 Testbed for firewalls operating between operators networks

The firewalls are located between two operator networks.



VD/MGT – Violation Detection & System Management **FW** – FireWall **M** – Monitoring Element
Red dots – Monitoring devices **Red lines** – Diadem traffic

Figure 7: testbed for firewalls between operator networks.

The testbed for inter-operator domain use-case has to contain in its minimal configuration:

- two core routers,
- at least two firewalls dedicated to each of routers (for redundancy),
- workstations and servers acting as attack generators (DDoS) and targeted servers,

- additional equipment: traffic analyzers, switches and routers.

3.5 Tests scenarios for selected use-cases

Initially we will focus on functional tests.
For the TCP SYN flood use-case, we can check for example [BSI01]: number of allowed half-opened connections for given IP addresses, closing TCP connections after a given time, buffering all data that refers to TCP connection establishment process, forwarding (only) full-opened connections, handling IP spoofing, traffic throughput, number of concurrent TCP connections, TCP connections established and closed per second, etc.
Similarly, for Web Server Overloading use-case we will take into consideration [BSI01]: filtering some (e.g. HTML) characters, response time for HTTP request, traffic throughput, concurrent TCP connections, TCP connections established and closed per second, errors for HTTP protocol, etc.
Parameters that influence test values are [BSI01]: hardware configuration of firewall, throughput, number of clients, logging mechanism intensity, number and form of ACLs, localization of firewall, etc.

4 Conclusion

This document has given a description of the two selected use-cases – TCP SYN flood and Web Server Overloading. Each use-case shows how the Diadem Firewall system can detect and mitigate different types of attacks that exploit features of protocols located at different layers of the OSI model: TCP (layer 4) and HTTP (layer 7). The document has also described some basic firewall testing techniques that will be used to validate our approach. Finally, a brief description of what the project testbed will look like was given. This document will be used as a starting point to actually setup the project testbed and to write the architecture evaluation plan.

5 References

[ANT03] Agilent Network Tester. Firewall Performance Testing. Solution Note. http://www.agilent-csg-j.com/closeup/NetworkTester_FirewallSolutionNote.pdf 2003.
[ANT04] Agilent Network Tester. Layer 4-7 Test Solution. Technical Datasheet. http://advanced.comms.agilent.com/networktester/products/datasheets/5989-1483EN.pdf August 2004.
[ART02] Agilent Router Tester. http://advanced.comms.agilent.com/networktester/products/datasheets/5989-1483EN.pdf October 2002.
[BarFT] Barisani A.: Firewall Tester. http://ftester.sourceforge.net/
[BFI04] Blade Firewall Informer Overview. http://www.blade-software.com/FWInformer.htm 2004.
[BSI01] Bundesamt für Sicherheit in der Informationstechnik, BSI (Federal Office for Information Security). Firewall Studies of BSI. http://www.bsi.bund.de/literat/studien/firewall/fw01eng/fwstuden.htm May 2001.
[CBR03] Cheswick W. R., Bellovin S. M., Rubin A. D.: Firewalls and Internet Security. Second Edition. Addison Wesley Professional 2003.
[CERT01] CERT® Security Improvement Modules. Deploying Firewalls. Test the firewall system. http://www.cert.org/security-improvement/practices/p060.html May 2001.
[DI204] Initial Interfaces Specification, DIADEM Firewall deliverable D2, July 2004.
[DI304] Attack Requirements Specification, DIADEM Firewall deliverable D3, July 2004.
[DI505] Architecture Specifications, DIADEM Firewall deliverable D5, January 2005.
[DI605] Revised Interfaces Specification, DIADEM Firewall deliverable D6, January 2005.

[FGM99] Fielding R., Gettys J., Magul J.: Hyper Text Transfer Protocol – HTTP/1.1. RFC 2616. http://www.ietf.org/rfc/rfc2616.txt June 1999.

[Pio03] Piotrowski P., Szafrański K.: Teletransmission Networks and Systems. Laboratories. Testing Software Categories. January 2003. (polish)

[SCSW] Spirent Communications. Security & Web Infrastructure. Firewall. http://www.spirentcom.com/.