NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

THESIS

EVALUATION OF A MULTI-AGENT SYSTEM FOR SIMULATION AND ANALYSIS OF DISTRIBUTED DENIAL-OF-SERVICE ATTACKS

by

Tee Huu, SAW

December 2003

Thesis Advisor:	James B. Michael
Thesis Co-Advisor:	Mikhail Auguston

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** December 2003	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE: **Evaluation of a Multi-Agent System for Simulation and Analysis of Distributed Denial-of-Service Attacks**	**5. FUNDING NUMBERS**
6. AUTHOR(S) Tee Huu, SAW	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for pubic release; distribution is unlimited	**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

DDoS attack is evolving at a rapid and alarming rate; an effective solution must be formulated using an adaptive approach. Most of the simulations are performed at the attack phase of the DDoS attack; thus the defense techniques developed focus mainly on filtering and isolating the attack. In order to develop and verify the effectiveness of a defense strategy, we needed a robust and flexible simulation tool. The Multi-Agent System Development Kit (MASDK) provided us a means to generate DDoS attack in a safe experimental environment for testing and validating security solutions, starting from the implantation phase: this allows researchers to develop new defense strategy even before the DDoS attack is launched. The paper begins with the study of the characteristics of DDoS attacks, the types of detection-and-response techniques, and the available DDoS attack simulation tools. The result generated by the MASDK simulation tool was used to evaluate the performance of the tool in simulating the DDoS attack over the networking environment.

14. SUBJECT TERMS DDoS, MASDK, Simulation Tool, Attack Tool, Computer Network.	**15. NUMBER OF PAGES** 72
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

THIS PAGE INTENTIONALLY LEFT BLANK

EVALUATION OF A MULTI-AGENT SYSTEM FOR SIMULATION AND ANALYSIS OF DISTRIBUTED DENIAL-OF-SERVICE ATTACKS

Tee Huu, SAW
Captain, Singapore Armed Forces (Army)
B.ENG (EE), Nanyang Technological University, 1999

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 2003

Author: Tee Huu, SAW

Approved by: James B. Michael
Thesis Advisor

Mikhail Auguston
Thesis Co-Advisor

Peter J. Denning
Chairman, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

DDoS attack is evolving at a rapid and alarming rate; an effective solution must be formulated using an adaptive approach. Most of the simulations are performed at the attack phase of the DDoS attack; thus the defense techniques developed focus mainly on filtering and isolating the attack. In order to develop and verify the effectiveness of a defense strategy, we needed a robust and flexible simulation tool. The Multi-Agent System Development Kit (MASDK) provided us a means to generate DDoS attack in a safe experimental environment for testing and validating security solutions, starting from the implantation phase: this allows researchers to develop new defense strategy even before the DDoS attack is launched. The paper begins with the study of the characteristics of DDoS attacks, the types of detection-and-response techniques, and the available DDoS attack simulation tools. The result generated by the MASDK simulation tool was used to evaluate the performance of the tool in simulating the DDoS attack over the networking environment.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

The author would like to thank Naval Postgraduate School for the support in purchasing the software for the research and the following people:

1. Professor Michael, for his patience, advice and guidance to ensure that the research was thorough, relevant and focused.

2. Professor Auguston, for providing his expertise advice to reinforce the research study.

3. Professor Kotenko and his research team, for their assistant in modifying the program to fulfill the research requirements.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Distributed denial-of-service (DDoS) attacks have become commonplace. A *DDoS attack* is a simultaneous assault on one or more nodes of a computer network, with the aim of reducing the availability of computing resources to legitimate users of those nodes. For instance, an attacker may use a DDoS attack to exhaust bandwidth, router processing capacity, or network stack resources in order to reduce or eliminate network connectivity on a targeted portion of the Internet.

DDoS attack programs vary in terms of technical sophistication, but their effects, irrespective of sophistication, can have an adverse effect on the security of a nation state or organization, such as first-order effect on electronic commerce and having higher-order effects on the economic stability of a nation or commercial enterprise. In other words, a simple DDoS attack program used by a script kiddy (i.e., someone with little technical expertise who primarily relies on downloading and running attack programs written by others) can have as great or greater an impact as a sophisticated attack created by information warriors (i.e., military and intelligence personnel conducting information operations).

In order to counter such attacks, one needs to be able to dissect existing attacks to understand how they work so that systems can be designed to be more robust to attack, one can predict how the vulnerabilities of new or composite systems might be exploited by DDoS attackers, and one can assess the potential effects of different types of DDoS attacks. It is not possible to make any system perfectly secure (except for trivial systems). Thus, one needs to be prepared to respond and mitigate likely DDOS-type attacks. One way to accomplish the aforementioned tasks is to model and simulate real-world, potential, and yet-to-be unleashed DDoS attacks, with the latter to be used by information warriors when assessing the effectiveness and efficiency of such attacks from an offensive perspective.

In this thesis we investigate the efficacy of using the Multi-Agent System Development Kit (MASDK) [Gorodetsky 2003] to model and simulate DDoS attacks on computer networks. Our objective is to assess the extent to which one can utilize the MASDK tool to access the effectiveness of protection mechanisms against DDoS attacks.

The scope of the thesis is as follows:

1. Identification of the weakness in a computer network's protection mechanisms, using the MASDK to simulated variants of DDoS attack scenarios
2. Sensitivity analysis on the protection software

The simulation-based exploration has the following purposes:

1. Checking a computer network security policy at stages of conceptual design and logic design of network security mechanisms. This type of checking is performed by simulation of attacks at a macro-level.
2. Checking security policy of a real-life computer network. This task is performed via simulation of attacks at a micro-level, which is by generating network traffic corresponding to the real activity of the attackers in the set up of the actual network architecture.

Our investigation begins with a study of the characteristics of DDoS attacks and the types of detection-and-response techniques. It continues with the development of attack scenarios and test cases for use in simulation-based experimentation. The computation models of the scenarios and test cases are submitted to the MASDK simulation engine. The simulation output from the macro-level simulation can provide an overview of the effectiveness of the protection mechanism. Following that, a micro-level simulation can be carried out to define the more subtle issues in the defense against the DDoS attacks.

Through our research, we observed that DDoS attack varies from the simple attack in the form of using e-mail messages to flood a mail server to a more sophisticated well-planned attack through infiltration of the network protection system and implant of zombies. The latter is a time bomb in the system control by the attacker. It is very difficult to defend against the DDoS attack in its first phase (implantation phase) because of the wide resources that it can use; the zombies are not necessary in the target network. As such, most of the defense concepts focus on the second phase (attack phase) of the DDoS attack. The simulation tools available to test such defense concept are limited; they mostly require expert labor-intensive manual procedures even in the conceptual design phase. The details of the findings will be further elaborated in the report.

II. DISTRIBUTED DENIAL-OF-SERVICE ATTACKS

From 1997 to 2000, DDoS attacks have evolved from a simple one-tier attacks (e.g., Ping flood, SYN flood, UDP flood) to the two-tiers attacks (e.g., Smurf attack) [Navratilova 2000]. After 2000, attacker found even more ways to accomplish such attack by communicating through networks to slave more attack computers. In this chapter, we will describe the phases and structure of the DDoS attack as well as highlight some of the recent attacks and their impact to the network users.

A. WHAT IS DDOS?

A DDoS attack is a distributed form of denial-of-service attacks. DoS attacks consume the resources of a remote host or network by sending large numbers of IP packets over a short time period. While a single host can cause significant damage by sending packets at its maximum rate, attackers can mount more powerful attacks by leveraging the resources of multiple hosts. In a typical DDoS attack, an attacker first intrudes into as many hosts as possible and installs two kinds of 'zombie' program: control program (master zombie) and flooding program (slave zombie). When the attacker triggers the master zombies, they order the slave zombies to launch malicious traffic towards a target server, thus depriving it of its resources so that it becomes unavailable to legitimate users. There are two principal types of DDoS attack: logic attacks and flooding attacks [Kashiwa 2002].

1. Logic Attacks

Logic attacks aim to crash the server or starve the server's system resources, such as its CPU utilization, file storage or memory, by exploiting flaws in the server software. Common examples of this type of attack include SYN flood, IP Fragmentation Overlap and Buffer Overflow [Scambray 2001].

2. Flooding Attacks

In flooding attacks, the attackers do not care what software the victim is using. Instead, they simply try to consume all available network bandwidth of the target network by bombarding it with massive amounts of traffic. Some of the well-known examples include TCP flood, UDP flood, Smurf, Fraggle and ICMP flood [Scambray 2001]. Most

DDoS attack tools, such as TFN, TFN2K and Stacheldraht, intensify these flooding attacks by launching them from multiple sources [Dittrich 2003].

B. PHASES IN DDOS ATTACK

According to [Cabrera 2001], the DDoS attacks have two phases, and involve three classes of systems. A simplified topology is given in Figure 1. The master system coordinates the whole effort.

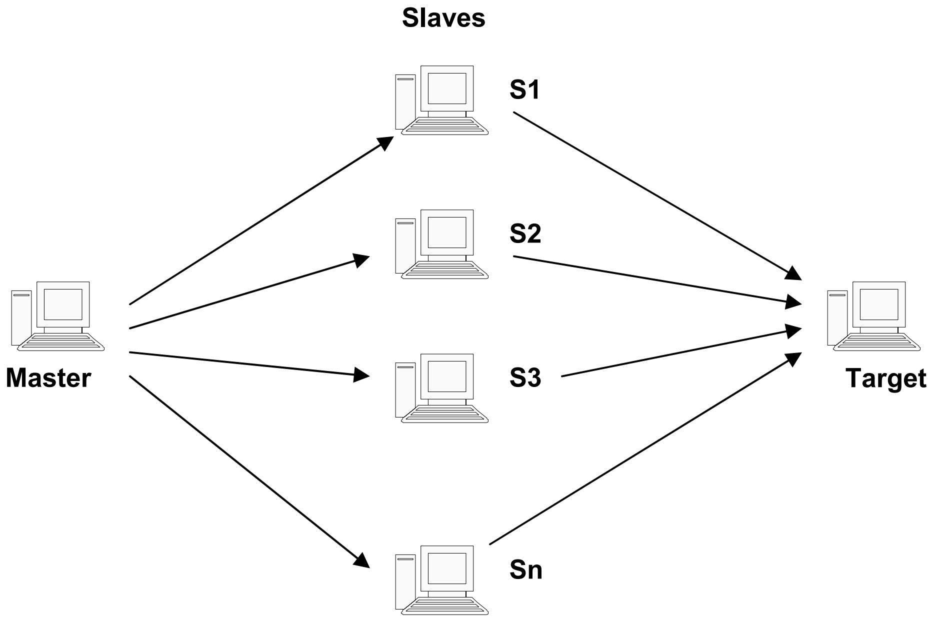


Figure 1. Distributed Denial-of-Service Attacks - A Simplified Topology.

In the first phase of attack, the *master* infiltrates multiple computer systems, and installs the DDoS attack tools, which are scripts capable of generating large volumes of traffic under command from the *master*. The infiltrated systems are known as the *slaves*. The second phase of attack cannot take place until phase one is completed, that is, the local copy of the attack program has a pipeline-type pipe-and-filter architecture.

The second phase is the actual DDoS attack. Under command from the master, the slaves generate network traffic to bring down the target system. Any system connected to the network can be a target.

C. RECENT ATTACKS

In a recent DDoS attack, the Slammer Worm (sometimes called Sapphire) was said to be the fastest computer worm in history. Slammer began to infect hosts on 25

January 2003, by exploiting buffer-overflow vulnerability in computers on the Internet running Microsoft's SQL Server or Microsoft SQL Server Desktop Engine (MSDE) 2000. Exploiting this vulnerability, the worm infected at least 75000 hosts, perhaps considerably more, and caused network outages and unforeseen consequences such as canceled airline flights, interference with elections, and ATM failures [Paxson 2003]. Slammer's most novel feature is its propagation speed. In approximately three minutes, the worm achieved its full scanning rate (more than 55 million scans per second), after which the growth rate will slow down because significant portions of the network will suffer insufficient bandwidth to accommodate more growth.

In August 19, 2003 the U.S. Department of Homeland Security (DHS) released an advisory warning user that a variant of Blaster worm, dubbed "nachi," "welchia" or "msblast.D," could cause denial-of-service conditions within organizations. The variant takes advantage of the same security weakness exploited by the Blaster worm and infects only systems that have not been properly patched. After infecting vulnerable Windows 2000 or Windows XP machines, the new worm then searches for and removes the Blaster worm file and attempts to download and install a patch from the Windowsupdate.com web site to close the hole. If the patch installation is successful, the worm then automatically reboots the systems and promptly begins looking for other machines on the network on which to copy itself. The scanning process can flood networks with high volumes of Internet Control Message Protocol (ICMP) traffic, causing network congestion which can result in denial-of-service conditions [Vijayan 2003].

DDoS attacks can be launched in many forms. [Jakobsson 2003] shows a “poor-man” DDoS attack using an email-based attack on selected victims, using only standard scripts and agents. For example, in August 2003, the Sobig.F attack program crashed email servers around the globe, as well as a few complete enterprise networks. The worm infects the computer and scans the files on the hard disk for e-mail addresses. It then uses those names as the “From” in new e-mails sent with copies of itself as attachments [Cherry 2003].

In each case, those breakdowns affected emergency services, military readiness, retail establishment, and government services. The new generation of DDoS attacks can be sophisticated and stealthy, making it difficult to detect them; this new generation of DDoS attacks in some instances even has the capability of covering their tracks.

III. PREVENTION, DETECTION AND RESPONSE

There is no simple answer to the prevention and detection of DDoS attacks. We can either defend against the attack in the first phase before the attack deployment is completed or isolating the attacking during the second phase. This chapter will give an overview of some of the prevention, detection, and response techniques.

A. PREVENTION, DETECTION AND RESPONSE

[Cabrera 2001] proposes a methodology for utilizing a Network Management System (NMS) for detection of DDoS attacks. The NMS depends on the information from MIB (Management Information Base) traffic variables collected from the system participating in the attack. Using these datasets, the MIB-based technique renders it possible to detect the attack before the Target is shut down. [Mirković 2002] presents a similar defense concept known as D-WARD, a DDoS defense system deployed at source-end networks that autonomously detects and stops attacks originating from these networks. The detection algorithm relies on constant monitoring and periodic comparison of current system workload with predicted workloads obtained from models of normal flow. Any mismatching flows will be rate-limited. These two methodologies may not be able to detect any new forms of attacks that are not captured in the datasets or models.

Research conducted by some organizations suggests that statistical measurements and statistical processing are effective mechanisms for addressing some of the challenges of defending against DDoS attacks. [Feinstein 2003] presented a DDoS defense strategy by analyzing the live traffic traces from a variety of network environments. The detection algorithms (Entropy and Chi-Square Statistic) deduce an attack by measuring the statistical properties of anomalies behavior in the packets at various points in the internet. The analysis result will be used to target packet-filtering or limiting responses to mitigate the effects of DDoS attacks. [Sung 2002] also proposes a technique to filter out DDoS traffic to improve the overall throughput of the legitimate traffic. It has the Enhanced Probabilistic Marking (EPM) module and Attack Mitigation Decision-making (AMD) module scheme to improve the throughput of legitimate traffic during attack, by preferentially filtering out traffic that is more likely to come from an attacker than a legitimate host. The main weakness in these strategies is that the responses to the attack take place in the second phase of the attack. That means part of the system operation will be degraded and the cleaning up of infected hosts in the network will be tedious and time consuming. On top of that the determination of DDoS traffic and legitimate traffic is based on probabilities.

[Kashiwa 2002] proposes a countermeasure against DDoS attacks using a traffic control method known as Active Shaping. This approach uses the Active Networking Technology incorporates programmability into intermediate network routers that can deploy application-level functions to detect, traceback and defend at suitable network nodes. Once again the detection of DDoS attack is during the second phase of the attack.

Most of the detection techniques focus on the event of the second phase of the DDoS attack and each has their unique defense features against different form of DDoS attacks. To effectively defense against DDoS, we need to detect DDoS attack in its deployment phase, which is in the first phase. If the attacks are carried out by highly trained attackers, who unlike script-kiddies, continuously customize their existing arsenal of attack programs and create new ones in order to both avoid detection and achieve the maximum desired effect, we will need detection technique that can detect and construct new attack scenarios. [Ning 2002] and [Cuppens 2002] suggested technique in correlating intrusion alerts to construct a possible attack scenarios of a bigger attack.

Most of the responses of the detection techniques can be grouped under three main types stated in [Zhao 2001]; (1) Validation of routing devices (*filtering*), (2) Dropping packet with specific characters (*mitigating*), (3) shutting down the host that undergoing attack (*isolation*). Although D-WARD and Active shaping approaches are able to traceback and investigate the source of attack, but the response are meekly defending the attack at the nodes. This leaves the attacker to look for vulnerability at other area and launch another attack. There is a need for responses that will deter or stop the attackers from launching further attack. [Michael 2003] presented a software decoy concept that integrates intrusion detection and response to protect critical information systems.

The study of detection and response techniques is not the purpose of the thesis, rather we are looking at a suitable tool to demonstrate and evaluate the strengths and weaknesses of such existing techniques.

IV. MASDK SIMULATION TOOL

Three techniques for performance evaluation stated in [Jain 1991] are analytical modeling, simulation and measurement. If it is a new concept, analytical modeling and simulation are the only techniques from which to choose. Simulations can incorporate more details and require less assumption than analytical modeling and, thus, more often the result is closer to reality. This chapter provides an overview of existing DDoS simulation tools, the architecture of the MASDK simulation tool and its simulation technique.

A. RELATED WORK

The most common simulation technique for DDoS attack is to use a DDoS attacks tool to carry out a DDoS attack in the simulation network. Some of the tools we were able to locate include *Stacheldraht* [Dittrich 1999a]*, Tribe Flood Network (TFN or TFN2K),* and *trinoo* [Dittrich 1999b].

[Sterne 2002] used the Stacheldraht v4 attack tool to generate UDP and ICMP flood, TCP SYN floods, and Smurf attacks as part of simulation. The technique in evaluating the defense is still based on expert labor-intensive manual procedures by networks administrators. A network needs to be setup to collect data while the attack is carried out.

The simulation methodology in [Sung 2002] uses CAIDA (Cooperative Association for Internet Data Analysis) and Cheswick's Topology dataset. CAIDA and Cheswick's Topology dataset are network probing tools for measuring forward IP paths, measuring round-trip time, tracking persistent routing changes, and visualizing network connectivity. Similar to the methodology used by [Sterne 2002], it requires the user to set up a target network in order to perform the simulation.

The primary weakness in these approaches is labor intensive manual procedures that involve. MASDK provides another type of simulation platform that allows the design to be tested on a single PC before an actual network is required. The following section will provide some details on the MASDK tool.

B. GENERALIZED ARCHITECTURE OF ATTACK SIMULATOR PROTOTYPE

The software prototype for computer network attack simulation is built as a multi-agent system that uses two classes of agents. The agent of the first class simulates defense system of the attacked computer network ("Network Agent") and the second one simulates a hacker performing attack against computer network ("Hacker Agent"). In the developed prototype each agent class has single instance although the developed technology makes it possible to simulate a team of hackers and a team of agents responsible for computer network security.

The aforementioned agents are implemented on the basis of the technology supported by Multi-Agent System Development Kit (MASDK) that is a multi-agent software tool aiming at support of the design and implementation of multi-agent systems of a broad range [Kotenko 2003a]. The developed and implemented simulator comprises the multitude of reusable components generated by use of the MASDK standard functionalities and application-oriented software components developed manually in terms of programming language MS Visual C++ 6.0 SP 5. Figure 2 illustrates the general prototype architecture.

Each agent operates using the respective fragment of the application ontology that is designed by use of an editor of MASDK facilities. The interaction between agents in the process of attack simulation is supported by the communication environment, which design and implementation is also supported by MASDK.

It is worthy to note that the first version of the prototype was implemented as a system that consisted of a single agent simulating a hacker's activity whereas computer network security system was simulated as a reactive system. In such architecture, it is not necessary to situate both attacking Hacker Agent and computer network security reactive system in a communication environment. In the first version of the prototype, the communication component plays a very important role. Indeed, the knowledge bases of Network Agent and Hacker Agent are implemented as two separate entities. An advantage of such a knowledge representation makes it possible to simulate adversary interactions. Such a model adequately implements interaction of both the above entities. In it, while simulating an attack in order to either obtain response providing it with the needed information (on the reconnaissance stage) or to perform an attack action (on the threat realization stage - Hacker Agent sends a certain message to the Network Agent). The Network Agent, as in real-life interactions, analyzes the received message and forms a responsive message. This message is formed based on the Network Agent's knowledge base that represents the network configuration and all its attributes needed to simulate real-life response. The Network Agent's knowledge base also uses information about possible existing attacks and provides reaction on them.

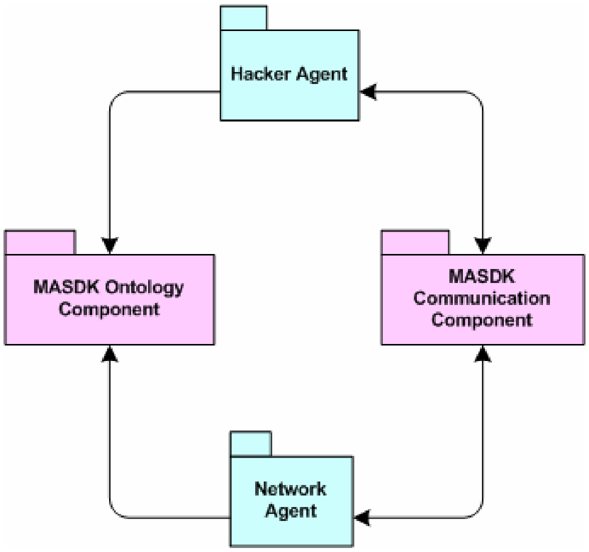


Figure 2. General Architecture of Attack Simulator.

The key components of both agents correspond to so-called kernels that are the modules written in C++ and compiled into a dll file. These components provide interface between the part of the software written in C++ and the components implemented through the use of MASDK. The kernels provide interfaces to the respective fragments of the application ontology, and initialize the state machines, which in turn execute their scripts.

Let us consider the main components of the Hacker Agent. The component model of the Hacker Agent is shown in Figure 3. It comprises the following main components:

1. The core (Agent Hacker Kernel)
2. Fragment of the application domain ontology
3. State machines model

4. Scripts
5. Attack task specification component
6. Probabilistic (stochastic) decision-making model with regard to the further actions
7. Network traffic generator
8. Visualization component of the attack scenario development

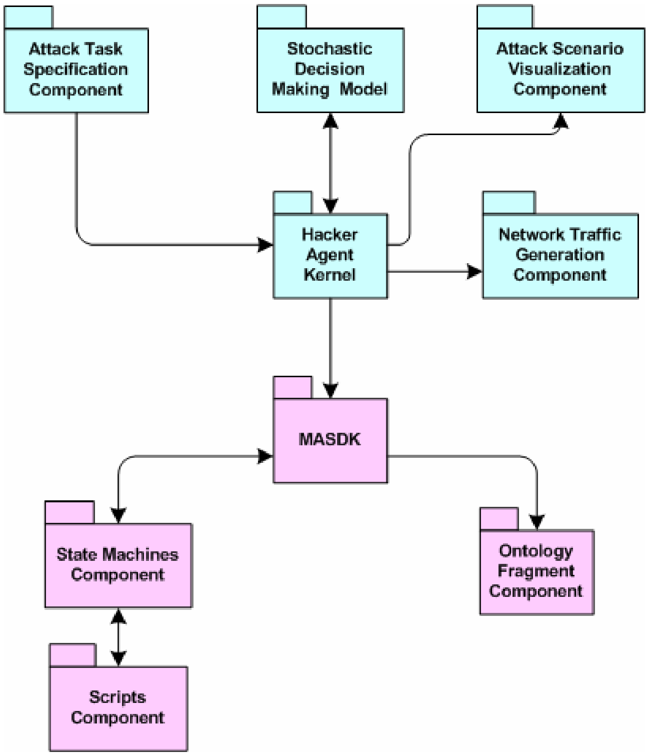

Attack Task Specification Component
Stochastic Decision Making Model
Attack Scenario Visualization Component
Hacker Agent Kernel
Network Traffic Generation Component
MASDK
State Machines Component
Ontology Fragment Component
Scripts Component

Figure 3. Component Model of the Hacker Agent.

The kernel of the Hacker Agent (Hacker.dll) contains a standard set of functions require for exploiting the ontology and running state machines. It is also provided with functions that call specification of attack task, compute next state-machine transition as well as initiate and perform visualization of the attack development.

Fragment of the application domain ontology specifies a set of notions and attributes used by the Hacker Agent.

The state machines model component is used for specification of the Hacker Agent behavior including decision-making mechanism used by the Hacker Agent for choosing the next action to perform. It is built on the basis of attribute stochastic grammars, and consists of over fifty nested state machines.

The script component specifies the set of scripts that can be performed by the Hacker Agent's state machines.

The attack task specification component provides user with interfaces require for specifying attack attributes.

A probabilistic decision-making model is used to determine the Hacker Agent's further actions in attack generation.

Network traffic generator is used to form the flow of network packets for several classes of attack directed to the hosts according to the attack specification. This component is initiated through calling the particular kernel function of the scripts of those states, for which the network traffic has to be generated.

Visualization component of the attack scenario development is used for visual representation of the attack progress, corresponding to each action of the attacker and respective responses from the Network Agent. The responses may be effective (i.e., the attack action was successful in part or in full), or ineffective (i.e., no response message, or the message saying that the attack was blocked by the firewall).

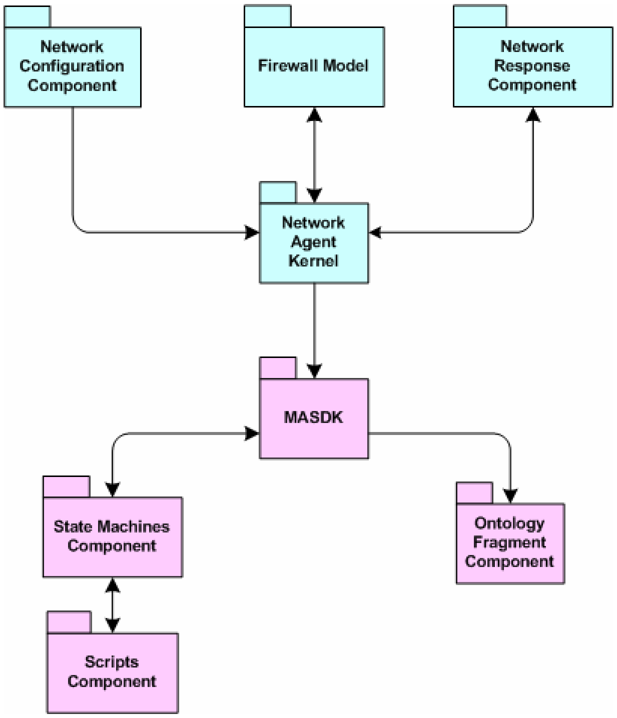

Network Configuration Component
Firewall Model
Network Response Component
Network Agent Kernel
MASDK
State Machines Component
Ontology Fragment Component
Scripts Component

Figure 4. Component Model of the Network Agent.

Let us consider the main components of the Network Agent. It is depicted in Figure 4. They are as follows:

1. The core (Network Agent Kernel)
2. Fragment of the application domain ontology
3. State machines model component
4. Scripts component
5. Network configuration specification component
6. Firewall model (implementation) component
7. Generator of the network's response to attack action

Network Agent Kernel (NetAgent.dll) contains the standard set of functions for processing the application domain ontology and the state machine model, as well as the functions used to specify the network configuration through the user interface, the firewall model initialization, and the computation of the network's response to an attacking action.

Fragment of the application domain ontology determines a set of notions and attributes used by the Network Agent.

State machines model of the Network Agent's specifies its behavior. This state machines model mostly performs communication functionality. It specifies the actions corresponding to the receiving of incoming message, classification, processing, and sending the response.

The scripts component specifies a set of scripts initialized from the state machines model of the Network Agent.

The network configuration specification component is used for the specification of a set of user interfaces for the description and configuration of the network to be attacked. All the notions and attributes that pertain to the networks and hosts, including the notions and attributes that describe firewalls, are described through these interfaces.

Firewall model (implementation) component is used to determine the firewall's response to the action generated by the Hacker Agent. Each incoming message from the Hacker Agent that constitutes an attack action is entered into the firewall model, which is assigned to the entire network (in imitation of a network firewall) or (and) the host (in imitation of a personal firewall). In the event of an attack is blocked by the firewall, the response message formed by the Network Agent contains the information about the attack that has been blocked by a specific firewall.

The generator of the network’s response to the attack actions is used for the generation of the responses (messages) according to the type attack action. It is initialized through the corresponding function exported by the agent’s kernel after the Hacker Agent has successfully overcome a firewall.

C. SIMULATION OF DDOS USING MASDK

In Chapter III, we have discussed various techniques and tools that developed to detect and defeat DDoS attacks. However, the DDoS attackers constantly modify their DDoS programs to exploit newly discovered vulnerabilities in the protection mechanisms, user applications, middleware, and operating systems. In order to combat DDoS, one needs to develop a strong theoretical basis upon which to harden information systems and infrastructures so they can survive such attacks.

In the beginning of this chapter, we mentioned two types of methods for simulation and testing the system that under DDoS attack. Most of these simulation methods required labor intensive manual setup even at conceptual design phase. To test a protection mechanism for various network infrastructure is too resource consuming.

MASDK supports two levels of simulation, namely the macro-level and micro-level.

1. Macro-level Simulation

In macro-level simulation, the simulation can be run in a standalone PC. The conceptual design of the protected network and the attacker capabilities can be input into the database of the "Network Agent" and "Hacker Agent" using the Microsoft Access. Figure 5 shows an example of the database file capturing the details of the protected network.

The user is required to enter the values of the tables in order for the simulation to produce a realistic result. For example, the user needs to specify whether the firewall is network-firewalls or host-firewalls and the probability of blocking undesired traffic and so on. The properties of the protective mechanism deployed in the network can be specified in this database. On the hand, the user can also specify the capabilities of the attacker, examples which are the networks that the attacker has gained trusted relationship.

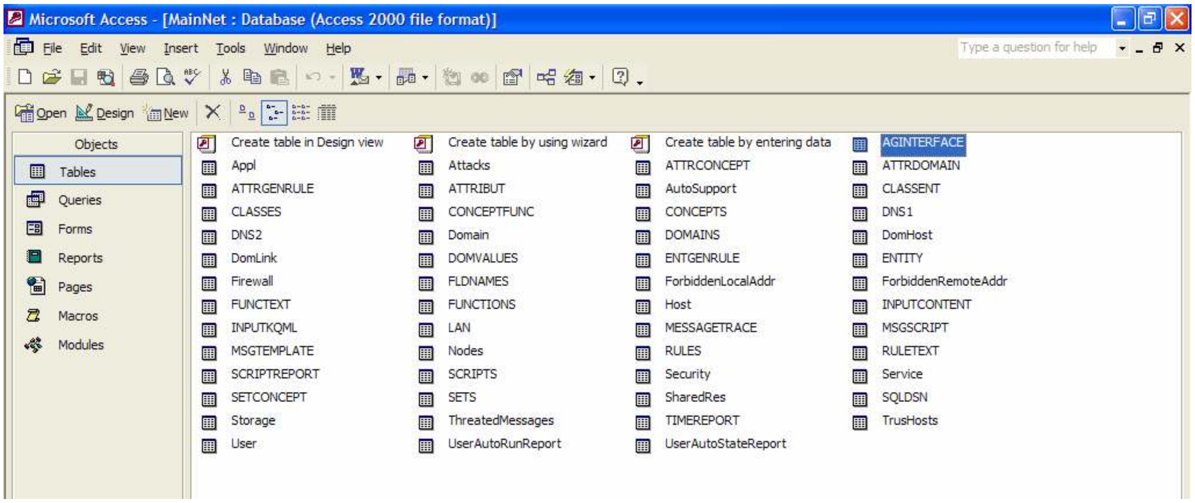


Figure 5. Database of network properties used in simulation

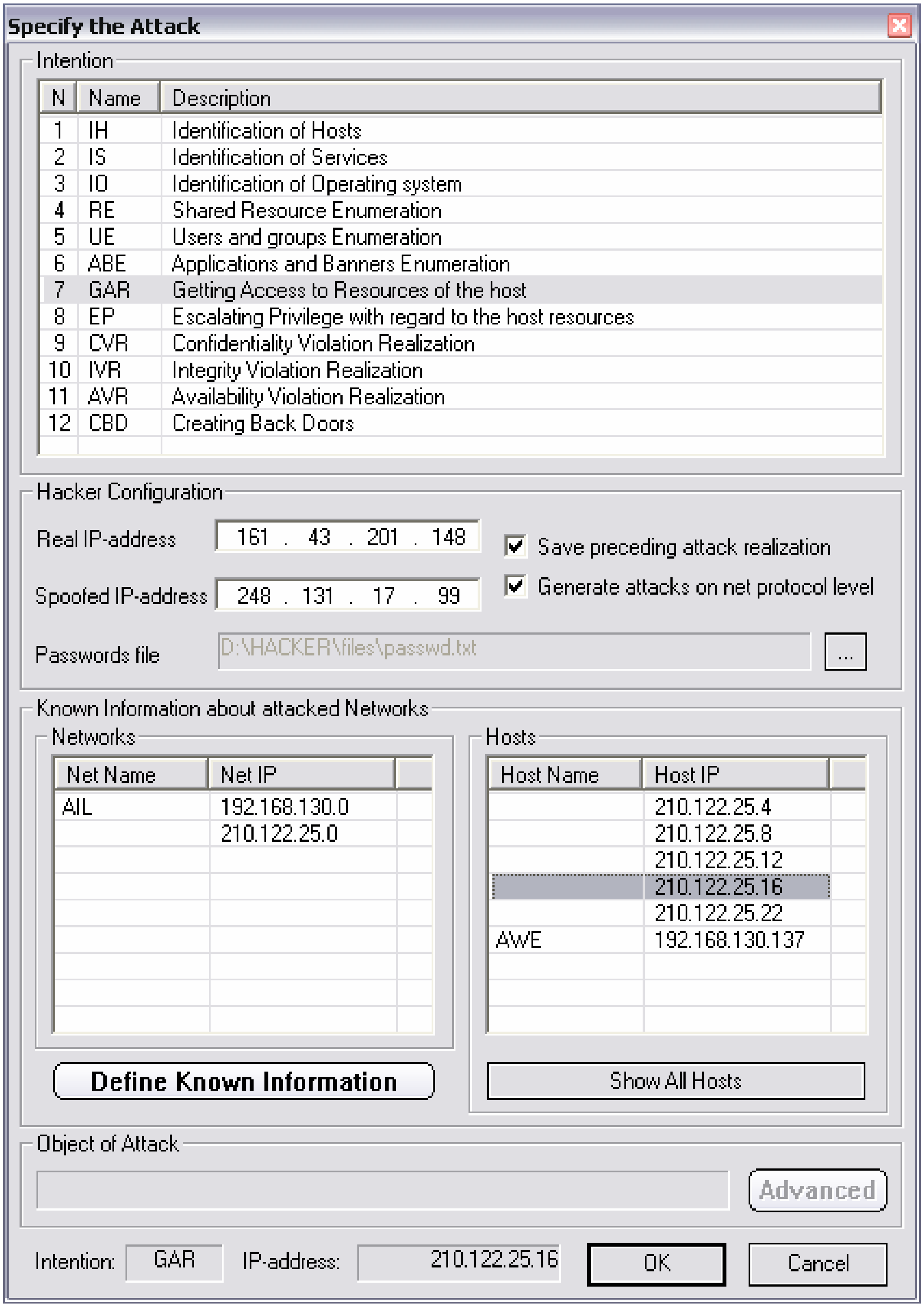
2. Micro-level Simulation

Micro-level simulation allows the user to setup part of the network for more details simulation and testing. The actual protective mechanisms (e.g. firewalls, tripwire, IDS, etc) can be deployed in the physical network and the MASDK simulation tool can be used to carried out the DDoS attack activities. Each of the activity can be carried out individually or combined as a sequence of events. In this mode, it allows more interactive simulation, the user is able to observer the behavior of the protective mechanism in the network.

3. Simulating the DDoS Attack

In Chapter II, we mentioned that there are two phases in a DDoS attacks. To prevent the launch of an attack, we must be able to stop it during the first phase. Otherwise we are only able to minimize the damage in phase two of the attack. The MASDK simulation tool breaks down the simulation of phase one of DDoS attack into twelve attack activities as listed in Table 1 [Kotenko 2003b].

The attack activities can be carried out as independent events or combine as a sequence of events during the simulation. Other hacking software (e.g., password cracking programs) can be integrated with MASDK to enhance the realism of the attacker capability. The attacker real or spoofed IP address can be specified in the simulation to test the protection against such attack. Figure 6 shows one of the user interface menu for specifying the attack scenario.



Specify the Attack

Intention

N	Name	Description
1	IH	Identification of Hosts
2	IS	Identification of Services
3	IO	Identification of Operating system
4	RE	Shared Resource Enumeration
5	UE	Users and groups Enumeration
6	ABE	Applications and Banners Enumeration
7	GAR	Getting Access to Resources of the host
8	EP	Escalating Privilege with regard to the host resources
9	CVR	Confidentiality Violation Realization
10	IVR	Integrity Violation Realization
11	AVR	Availability Violation Realization
12	CBD	Creating Back Doors

Hacker Configuration

Real IP-address 161 . 43 . 201 . 148

Spoofed IP-address 248 . 131 . 17 . 99

☑ Save preceding attack realization

☑ Generate attacks on net protocol level

Passwords file D:\HACKER\files\passwd.txt

Known Information about attacked Networks

Networks

Net Name	Net IP
AIL	192.168.130.0
	210.122.25.0

Hosts

Host Name	Host IP
	210.122.25.4
	210.122.25.8
	210.122.25.12
	210.122.25.16
	210.122.25.22
AWE	192.168.130.137

Object of Attack

Intention: GAR IP-address: 210.122.25.16

Figure 6. User interface menu for specifying the attack scenario.

Class	Number	Designation	Interpretation
Reconnaissance	1	IH	Identification of Hosts
	2	IS	Identification of Services
	3	IO	Identification of OS
	4	RE	Shared resources enumeration
	5	UE	Users and groups enumeration
	6	ABE	Applications and banners enumeration
Implantation and Threat Realization	7	GAR	Get access to resources of the host
	8	EP	Escalating Privilege with regard to the host resources
	9	CVR	Confidentiality violation realization
	10	IVR	Integrity violation realization
	11	AVR	Availability violation realization
	12	CBD	Creating back doors

Table 1 List of attack activities (From: Kotenko 2003b)

During the simulation, the attack activities will be carried based on the attack scenario created. MASDK will feedback to the user whether the attack activities are successfully carried out or otherwise. Appendix A provides more details on the attack description.

V. TEST SUITE - TEST CASES AND SCENARIOS

The best way to learn the response and behavior is to apply the concepts to a real system. This is specially true of computer systems performance evaluation because even though the techniques appear simple on the surface, their application to the real systems offer a different experience since the real systems do not behave in a simple manner. The test scenarios in this chapter are developed to study and analysis the dynamic behavior of the network protection mechanism defense against the DDoS attack. The results obtained will be used in the analysis in the Chapter VI.

A. TEST SCENARIO 1

The server in the network is providing services to both its local hosts as well as the remote hosts (e.g. customer, employee) through the internet. The attacker prepared and launched a DDoS attack on the Server from providing the service.

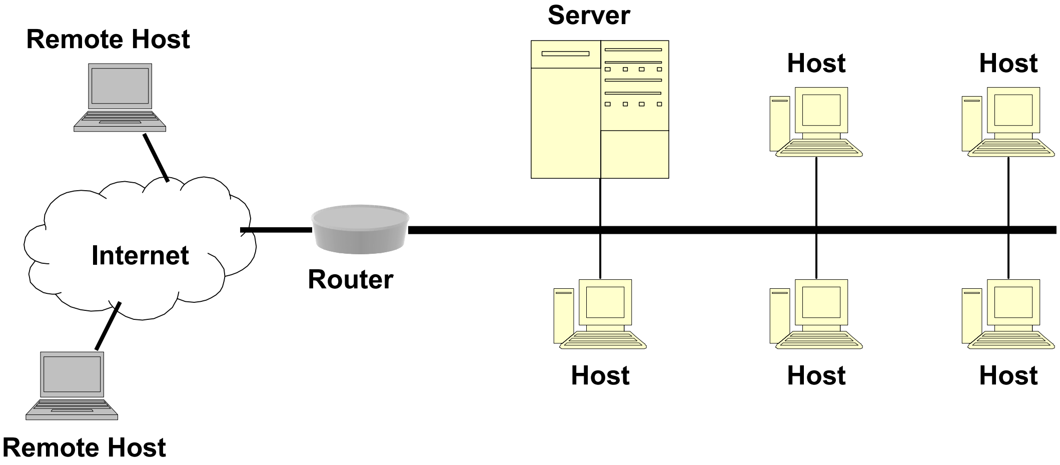


Figure 7. Network Architecture of Test Scenario 1.

1. Test Case 1: Direct Attack on Server

Attacker infiltrates the network, gain access to the local hosts (need not be all) and launched a direct attack on the server. The attacker will carry out the following five general attack activities.

Reconnaissance: Use open sources to gain available information about the network. Useful information includes phone numbers, postal addresses, internet addresses, names of system administrators, technologies in use, business partnerships, etc. R*econnaissance agent* gathers information from a variety of servers on the internet like,

1. Ping to check whether any of the host is alive (connected).
2. Do a DNS look up to map domain names to IP addresses
3. Do a DNS zone transfer and grab all information available about a particular domain.
4. Look up IP block registration to analyze the IP address range assigned to a given organization.
5. Trace-route to determine the list of routers between the source and destination.

Scanning: Using the information gather from the reconnaissance, the attacker will scan the network systems for vulnerabilities. Employ *scanning agent* to,

1. Search for modems that are connected using the range of phone numbers gathered.
2. Develop a network map of the target network to define the network topology.
3. Conduct port scan to determine open ports that indicated the types of service (e.g. type of OS) which are running.
4. Scan for vulnerabilities like misconfigurations, unpatched systems with known vulnerabilities, etc.

Gaining Access: There are many ways to gain access to the target network, the *gain_access agent* can,

1. Manipulate poorly written software.
2. Exploit weak password storage mechanisms.
3. Gather data that is not properly encrypted such as userIDs and passwords.

Maintaining access: After gaining access to the system, *gain_access agent* can maintain the access by utilize Trojan Horse and Backdoor techniques to hide its presence on the system and guarantee future access. These backdoors can be created in three levels,

1. Application-level Backdoors - install malicious application into the system
2. Traditional RootKits - modify existing programs on the system.
3. Kernel-level RootKits - modify the kernel of the system itself.

Covering the Tracks: The master and slave zombies must be installed without being detected by the protective mechanism in the host or the network. The *stealth agent* is employed to carry out:

1. File Hiding to hide the malicious program in the system (host or server).
2. Protocol Tunneling to hide the data moving across the network by carrying one protocol on top of another.

3. Covert Channels to hide data inside the openings of a protocol (e.g. the TCP header).

Once the deployment phase of DDoS attack is completed, attacker will launch an attack remotely through the internet.

Protective Mechanisms available include:

1. Intrusion detection system
2. Firewall
3. File integrity checking software (e.g. Tripwire)
4. Anti-virus software
5. Configuration management
    - Implement non-executable system stack to prevent buffer overflow attack.
    - Keep the system with up to date patches.
    - Identify users with weak password.
    - Shut off unwanted services (e.g. Telnet) that have inherent security weaknesses.
    - Close unused port.
    - Audit usage of system.

The simulation and analysis will include the following:

Run 1: Using the MASDK tool to attack the network without any protective mechanism. At micro-level simulation with the setup of actual network, we will examine how fast the MASDK tool can infiltrate the network and set the backdoor at the hosts.

Run 2: Using the MASDK tool to attack the network that has protection mechanism 1 to 4. At micro-level simulation with the setup of actual network, we will examine the following,

1. Is the IDS in the network able to recognize the attack signature?
2. Is the type of firewall (e.g. stateful or proxy firewall) and firewall configuration able to block the attack?
3. Is the file integrity checking software able to detect the insertion and modification of the files in the system?
4. Is the anti-virus software help in defending the system again such attack?

The protective mechanisms need to be set at different configurations to produce the best result.

Run 3: In addition to the protective mechanisms in Run 2, the network is further hardened with the configuration management. This will help to examine how effective by doing a proper network configuration defend against DDoS attack.

2. Test Case 2: Spoof as Remote Hosts to Attack the Server

In order to launch an effective DDoS, the attacker must control enough zombies to execute the attack. If the number of infected hosts is not sufficient, the attacker will spoof as numerous remote hosts to attack the Server.

The attack will still require to carry out the five general attack activities (reconnaissance, scanning, gaining access, maintaining access and covering the tracks) to gain enough information to spoof as legitimate hosts.

The available protective mechanism will be the same in Test Case 1.

The simulation and analysis will include the following:

Run 1: Using the MASDK tool to attack the network without any protective mechanism. We will examine how fast the MASDK tool can gain information of the system and spoof the network as a remote legitimate host.

Run 2: Using the MASDK tool to attack the network that has all available protection mechanisms. At micro-level simulation with the setup of actual network, we will examine the following,

1. Is the authentication system effective against the attack?
2. Is the IDS in the network able to recognize the attack signature?
3. Is the Firewall configuration able to block the attack?

The protective mechanisms need to be set at different configurations to produce the best result.

3. Test Case 3: Attack Other Vulnerabilities of the Network

Instead of attacking the server of the network that providing the services, we will attack the weakness in such network architecture.

The attack will still require to carry out the five general attack activities (reconnaissance, scanning, gaining access, maintaining access and covering the tracks) to gain enough information to spoof as legitimate hosts.

The available protective mechanism will be the same in Test Case 1.

The simulation and analysis will include the following:

Run 1: Using the MASDK tool to attack the network router. This will deny services provide by the server to the remote legitimate hosts. We shall examine how to modify the network architecture to overcome such attack and the resources that require.

B. TEST SCENARIO 2

In Scenario 1, the network does not have layers of protection mechanism. All systems within the network are protected by the same firewall or IDS. In most application, the network will like to separate untrusted users on the DMZ networks. DMZ (De-Militarized Zone) a "neutral zone" between a company's private network and the outside public network. For example, packet filtering might allow HTTP from the Internet to reach the DMZ but prohibit telnet, finger, and other protocols that might easily allow an attack on your trusted networks to be launched.

It is not necessary that this network architecture is more secure than the one in Scenario 1. However, this architecture is closer to the set up of an actual network.

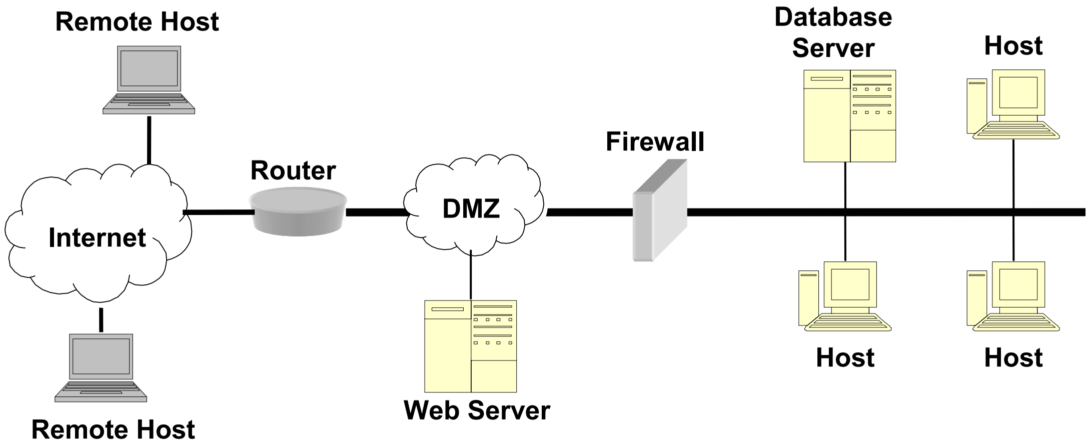


Figure 8. Network Architecture of Test Scenario 2.

1. Test Case 1: Direct Attack on Server

Attacker infiltrates the network, gain access to the local hosts and launched a direct attack on the server. The attacker will carry out the five general attack activities on the target network.

Protective Mechanism Available

1. Intrusion detection system

2. Firewall
3. File integrity checking software (e.g. Tripwire)
4. Anti-virus software
5. Configuration management

The simulation and analysis will include the following:

Run 1: Using the MASDK tool to attack the Database Server. At micro-level simulation with the setup of actual network, we will examine the following,

1. Is the IDS in the network able to recognize the attack signature?
2. Is the type of firewall (e.g. stateful or proxy firewall) and firewall configuration able to block the attack?
3. Is the file integrity checking software able to detect the insertion and modification of the files in the system?
4. Is the anti-virus software help in defending the system again such attack?

Run 2: With more security resources, we will examine the following,

1. Given more IDS sensors, how should the IDS sensors deploy to enhance the detection?
2. If another Firewall is employed between the router and DMZ, does it improve the defense against the attack? What should be the configuration of the two firewalls?

2. Test Case 2: Direct Attack on Server

This test case is similar to Test Case 2 in Scenario 1 with different network architecture.

The simulation and analysis will include the following:

Run 1: Using the MASDK tool to attack the network that has all available protection mechanisms. We will examine the following,

1. Is the authentication system effective against the attack?
2. Is the IDS in the network able to recognize the attack signature?
3. Is the Firewall configuration able to block the attack?

Run 2: With more security resources, we will examine the following,

1. Given more IDS sensors, how should the IDS sensors deploy to enhance the detection? The deployment of sensors may be different from Test Case 1 for best result.
2. If another Firewall is employed between the router and DMZ, does it improve the defense against the attack? What should be the configuration of the two firewalls?

3. Test Case 3: Attack Other Vulnerabilities of the Network

This test case is similar to Test Case 3 in Scenario 1 with different network architecture.

The simulation and analysis will include the following:

Run 1: Using the MASDK tool to attack the network router. This will deny services provide by the server to the remote legitimate hosts. We shall examine how to modify the network architecture to overcome such attack and the resources that require.

C. TEST SCENARIO 3

Network A and Network B is business partner, therefore they have a trusted relationship. The server in Network A is the target of the attack. Network A has a much stronger security setup than Network B. Attacker explore the weakness in Network B and use it as the platform to launch the DDoS attack.

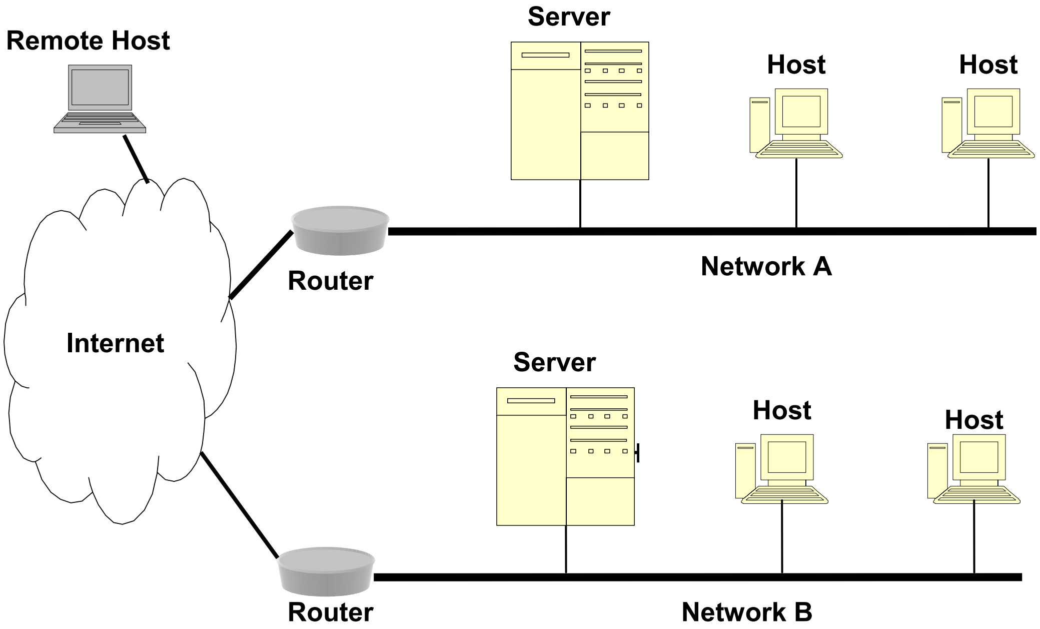


Figure 9. Network Architecture of Test Scenario 3.

1. Test Case 1: Attack on Server in Network A Using Network B Resources

Attacker infiltrates Network B, gain access to the local hosts (need not be all) and launched a DDoS attack on the server of Network A. The attacker will carry out the five general attack activities.

Protective Mechanism Available

1. Intrusion detection system
2. Firewall
3. File integrity checking software (e.g. Tripwire)
4. Anti-virus software
5. Configuration management

The simulation and analysis will include the following:

Run 1: Using the MASDK tool to take over the systems in Network B as zombies. Once the zombies are successfully install in Network B, the attack will make use of the trusted relationship to launch the DDoS attack on the server of Network A. We shall examine the whether the protection mechanism in Network A is able to detect the increase in network traffic between Network A and Network B.

Run 2: Employ a few more similar networks like Network B. Instead of using only the zombies in Network B, the attacker distributes the attack zombies over a few networks. We shall examine whether this form of attack are more stealthy as compare to Run 1.

2. Test Case 2: Attack Other Vulnerabilities of the Network

Instead of attacking the server of the network that providing the services, we will attack the weakness in such network architecture.

The simulation and analysis will include the following:

Run 1: Using the MASDK tool to attack the network router using Network B resources. This will deny services provide by the server to the remote legitimate hosts. We shall examine how to modify the network architecture to overcome such attack and the resources that require.

VI. RESULTS AND ANALYSIS

[Kotenko 2003a] states that the experiments conducted with the MASDK can be performed at the macro- or micro-level. Macro-level simulation for checking a computer network security policy at stages of conceptual and logic design of a network security system and micro-level for checking the security policy of the real-life computer network. However the present version of MASDK is only capable of simulating an attack at the macro-level with the limitation stated in Chapter VII. The results of the simulation are shown in Appendix B. In this chapter we present our analysis of the results obtained from the simulation.

A. INSTALLATION AND OPERATION OF THE SIMULATION TOOL

The instruction on installation and operation of the attack simulator is in Appendix A. There are four points that the user should be aware before using the software.

First, in addition to the installation and operation instruction that attaches with the software, the system requires Microsoft MS Studio Library to operate. The Microsoft MS Studio Library can be easily set up by installing any of the software in Microsoft MS Studio (e.g. Visual C++, Visual Basic). If the library is not installed, the user will experience error message on missing dll file during the execution of the program.

Second, in the present version of the simulation tool, the user may realize that the simulation 'hangs' after a few attack activities. The user should check the option 'save preceding attack realization' on the screen shown on Figure 6 for all simulations. This will allow the user to exit from the simulation if it hangs and continue the sequence of attack by re-entering the simulator with the same input parameters.

Third, the simulation will take as short as thirty-five minutes to achieve the attack objective for a network without any protection mechanism. A system with protection mechanisms can run as long as eight hours or more for multiple iterations before 'the end of attack' message is received when the attack activities in Table 1 are not successful. However the interval between each activity should not be more than three minutes, else the system is likely in the 'hang' state.

Lastly, the simulation result obtained in the 'Attack Scenario Realization' screen (see Figure 11) is not able to export to another file or save; therefore the user should copy the result screen to another file manually before exiting the program.

B. TEST SCENARIO 1

In this scenario, the network architecture is simplified to test and observe the behavior and characteristics of the MASDK simulation tool. The support for modeling the protection mechanisms is limited to a firewall at the network level for the present version of the MASDK simulation tool.

1. Test Case 1: Direct Attack on Server

Due to the limitations in the current version of the MASDK simulation tool, we were only able to perform *Run 1* and *Run 2*.

In *Run 1* with no protection mechanism, the Hacker Agent is able to carry out all the attack activities smoothly. The simulation took thirty-five minutes from reconnaissance to complete installation of backdoors and covering up its tracks. The result obtained is not surprising: in real-life, if the network has no protection at all, an attacker will require little or no effort to take over the control of the computer systems present in the network.

In *Run 2* with a network Firewall (set Probability at 0.9), there are total three runs with average of five hours per run. In the first two runs, the Hacker Agent is not successful in the attack: all attack activities are blocked by the firewall. In the third run the Hacker Agent has tried numerous attempts to gather information of the target network, and from screen 3 onwards the Hacker Agent is able to identify the services that are available on the network systems. The reconnaissance result gives the Hacker Agent the opportunity to install three backdoors on the network systems. This is true to real-life applications, in which the configuration of the firewall will change during different phases of an operation. For example, the firewall can be temporarily configured to allow Telnet for remote configuration or FTP for transferring of data. The attacker will periodically check the target network for such opportunities to deliver its attack payload.

VII. RECOMMENDATIONS

The present version of MASDK simulator places artificial limitations on simulating the real-world DDoS attacks. Here we will highlight our recommendations for improving the toolkit.

A. TYPES OF DDOS ATTACKS

There are two general types of DDoS attacks as (c.f., Chapter II): logic and flooding attacks. In the simulator, the DDoS attack is limited to phase one of the flooding attack. The user is not able to specify the type of attack. Furthermore, there are many forms of flooding attacks, each of which may require different defense strategies. In recent years, DDoS attacks have become even more sophisticated; for instance, the Sobig.F and MS Blast have developed unique defenses. For example, one of these defenses causes infected computers to reboot themselves every ten minutes which is faster than any software can be downloaded to install a fix [Cherry 2003].

Typically the DDoS attack depends on two properties: (i) the size of attack packets and (ii) the multiplying speed. The larger the attack packet, the higher the impact will be on the traffic congestion on the network. However, a small attack packet is more likely to get through the packet filtering but require a much higher multiplying speed to achieve the same impact.

The weaknesses and strengths of a protection mechanism can only be truly validated if various properties, types, and forms of attack can be built into the simulator.

B. SIMULATING THE ACTUAL NETWORK TRAFFIC

The element in the simulator that requires modification is the network traffic. The traffic flows in any network are analogous to human keystrokes: each has its unique pattern of behavior. The user must be able to program and control the network behavior and traffic density of the network for specific network architectures in order to observe the impact caused by the attack. Network behavior includes the type of messages (i.e., broadcast, unicast, or multi-cast) and varies with traffic density over time. *Traffic density* is the measure of the load versus the bandwidth. The simulator must be robust enough to simulate the network traffic density and network behavior in order to obtain an accurate simulation result.

C. VARIOUS NETWORK ARCHITECTURES

Although the present version of MASDK simulator can simulate the DDoS attack at macro-level in a standalone PC, it still requires the programmer assistant to change the network architecture. There should be a tool (e.g., GUI) for the user to build the network for simulation. This will give the user more flexibility during the design phase and a better view of both the strengths and weaknesses of different network architecture in the simulation-and-analysis phase.

D. SIMULATING THE PROTECTION MECHANISM

There are various commercial-available network protection elements such as firewalls, intrusion-detection systems, and tripwires. These elements can have various configurations or types for different protection requirements. A firewall can be stateful or proxy firewall, or it can be configured to block certain network protocols (e.g., ICMP or Telnet). An IDS can have host- or network-based sensors. The present version of the MASDK simulation tool is only capable of simulating a firewall. The library for protection mechanism needs to be expanded in order to permit macro-level simulation of real-world protection mechanisms and security policies.

E. SIMULATING THE TWO PHASES OF DDOS ATTACK

In Chapter II, we mentioned that DDoS attacks are comprised of two phases: (i) the implantation phase and (ii) launching of the attack phase. The version of MASDK used in the study is limited to simulation for the implantation phase. Although this gives the advantage in studying the behavior of the attacker and the attack strategy, it is not able to demonstrate the impact and degree of damage in such attack.

F. DOCUMENTATION OF MASDK SIMULATOR

The MASDK simulator still in the research phase, as such there is no user manual or guide in using the tool. A good documentation will help both the designer and user on the progress of the simulator development.

VIII. CONCLUSION

DDoS attack is evolving at a rapid and alarming rate; an effective solution must be formulated using an adaptive approach. Most of the simulations are performed at the attack phase of the DDoS attack; thus the defense techniques developed focus mainly on filtering and isolating the attack. We observed that the attackers, besides constantly advancing their attacking skill, also develop an array of defense strategies against the countermeasures. The anti-virus software, firewalls or IDS will need to be empowered with a similar capability in order to be effective against the DDoS attacks: the defense strategy must be able to adapt to changes. Defending against the attack in its implantation phase will likely prove to be more effective in the future.

In order to develop and verify the effectiveness of a defense strategy, we needed a robust and flexible simulation tool. MASDK simulation tool provided us a means to generate DDoS attack in a safe experimental environment for testing and validating security solutions, starting from the implantation phase: this allows researchers to develop new defense strategy even before the DDoS attack is launched. However, the study shows that MASDK development is still in its infancy. The result obtained from the simulation is static and does not show the robustness and complexity of the DDoS attack. To have a more realistic, interactive and dynamic simulation, modification is required on the hacker agent and network agent to model various types and forms of DDoS attack in addition to the network traffic and network architecture.

Nevertheless, the MASDK simulation tool has laid a foundation and gives a head start in modeling the DDoS attack right at the beginning of the attack. The two levels of simulation (macro-level and micro-level) also minimize the amount of resources required to verify the concept of any defense strategy or mechanism. The MASDK simulation tool can be further developed to include more agents as mentioned in [Gorodetsky 2003] to provide a more robust, interactive and realistic simulation environment.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

This appendix contains the instruction on installation and operation with Attack Simulator (in the environment MASDK 2.1).

System requirements:

1. OS Windows 2000 (with SP3) or XP (with SP1)
2. free disk space - 50 Mb
3. minimum of operation memory - 128 Mb
4. Presence of the established TCP/IP package (As a rule, it is automatically installed together with OS)
5. Presence of established driver Access ODBC (As a rule, it is automatically installed together with OS and MS Office)

Necessary additional software (on the disk):

1. Java Run Time Environment, version - 1.4.1_01 or higher (File j2re-1_4_2-win.exe of the distribution kit is on the disk).
2. MS XML, version - 4.0 or higher (File SP1.msi of the distribution kit is on the disk).

Sequence of actions on installation:

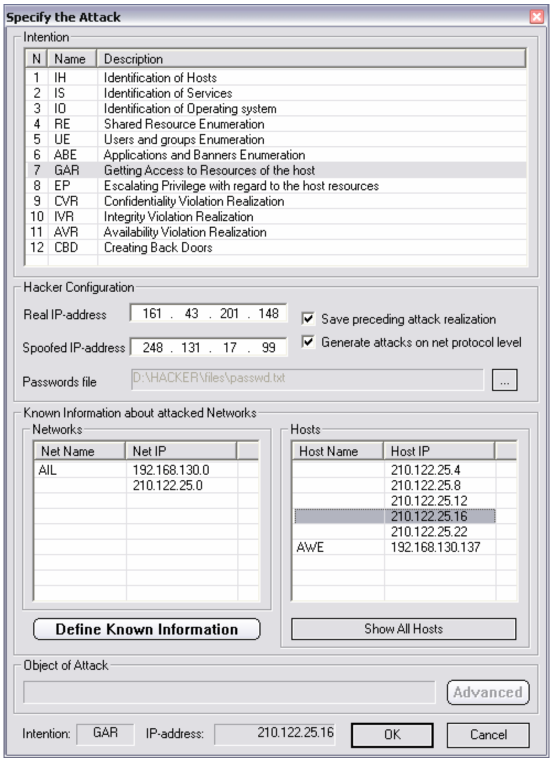
1. Copy catalogue MAS4 with all subdirectories to disk C.
2. Start the file C:\MAS4\Generic_agent\Server\server.bat
3. Start the file C:\MAS4\Generic_agent\Server\portal.bat [1]
4. Select the item of the menu "File → Start Agents" in window "AIL Registry" or press the button 🔌 . After this two agents "MainHack" and "MainNet" will be installed [2]
5. Press the button ✔ in the appeared affiliated window "Agent MainHack" of the window "AILab Agent Library :: Portal Component". As a result an user interface of hacker agent will be installed

[1] It is doesn't matter in which order to start the files "server.bat" and "portal.bat". As a result four windows will appear, including windows "AIL Registry" and "AILab Agent Library :: Portal Component"

[2] As a result under successful installation of agents the following information will appear in window "AILRegistry":

```
Added New Agent MainNet

Returning current agent information…

Returning current agent information to new agent…

Added New Agent MainHack

Returning current agent information…

Returning current agent information to new agent…
```

6. Select “Attack Description” in the appeared window “Predefined Actions” and press the button “Use”. As a result the window “Specify The Attack” will appear as in Figure 10.



N	Name	Description
1	IH	Identification of Hosts
2	IS	Identification of Services
3	IO	Identification of Operating system
4	RE	Shared Resource Enumeration
5	UE	Users and groups Enumeration
6	ABE	Applications and Banners Enumeration
7	GAR	Getting Access to Resources of the host
8	EP	Escalating Privilege with regard to the host resources
9	CVR	Confidentiality Violation Realization
10	IVR	Integrity Violation Realization
11	AVR	Availability Violation Realization
12	CBD	Creating Back Doors

Real IP-address 161 . 43 . 201 . 148

Spoofed IP-address 248 . 131 . 17 . 99

☑ Save preceding attack realization

☑ Generate attacks on net protocol level

Passwords file D:\HACKER\files\passwd.txt

Net Name	Net IP
AIL	192.168.130.0
	210.122.25.0

Host Name	Host IP
	210.122.25.4
	210.122.25.8
	210.122.25.12
	210.122.25.16
	210.122.25.22
AWE	192.168.130.137

Intention: GAR IP-address: 210.122.25.16

Figure 10. Window for Specifying the Attack.

The settings of the Hacker Agent include the following elements:

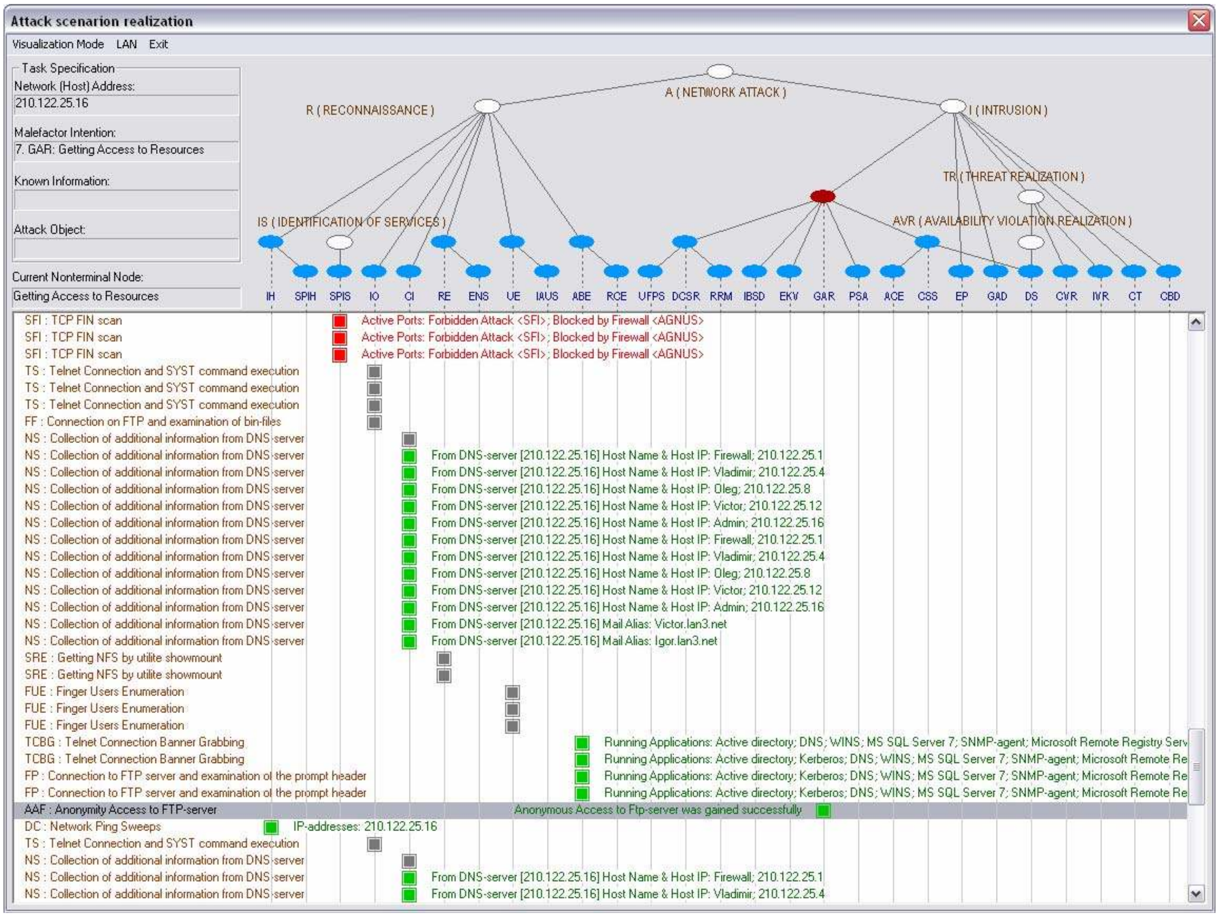
- *Real IP-address* – Hacker Agent’s real IP address. This field is mandatory. It is necessary for determining the attack scenario on both macro-level and micro-level (for forming network packets on the level of TCP/IP protocol stack);
- *Spoofed IP-address* – Hacker Agent’s spoofed IP address. This field is not mandatory, unless attacks have to be generated on micro-level;
- *Password file* – path to the file with a list of words for guessing the password. This file is used only in generating attacks “Password Guessing” (PG) and “Password Cracking” (PC) on micro-level;
- *Save preceding attack realization* – the tag that determines whether the results of the previous attacks will be saved. When this parameter is initialized, the attack specified by this component will be executed using the knowledge base formed in the previous realizations of attacks (not necessarily with the same intentions). All logs and traces are also saved in this case;
- *Generate attacks on net protocol level* – the tag that determines whether the attack will be generated on macro-level. When this parameter is initialized, besides the simulation of attack on macro-level, network packets are generated on the level of TCP/IP protocols stack.

In the given debugging version an updating of the information, known to the agent - hacker (button “Define Known Information”) is blocked.

The used initial data allow carrying out attacks for 12 classes of intentions at the macro-level.

For micro-level simulation of attacks it is necessary to set the real IP-address of an attacked host or a network. In the given version it is carried out directly through a hacker’s database (file “MainHack.mdb”).

7. Press the button “OK” in window “Specify The Attack”. As a result a process of attack simulation will be installed. It will be visualized as shown in Figure 11.



Attack scenarion realization

Task Specification
Network (Host) Address: 210.122.25.16
Malefactor Intention: 7. GAR: Getting Access to Resources
Known Information:
Attack Object:
Current Nonterminal Node: Getting Access to Resources

```
SFI : TCP FIN scan                                        Active Ports: Forbidden Attack <SFI> ; Blocked by Firewall <AGNUS>
SFI : TCP FIN scan                                        Active Ports: Forbidden Attack <SFI> ; Blocked by Firewall <AGNUS>
SFI : TCP FIN scan                                        Active Ports: Forbidden Attack <SFI> ; Blocked by Firewall <AGNUS>
TS : Telnet Connection and SYST command execution
TS : Telnet Connection and SYST command execution
TS : Telnet Connection and SYST command execution
FF : Connection on FTP and examination of bin-files
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Firewall; 210.122.25.1
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Vladimir; 210.122.25.4
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Oleg; 210.122.25.8
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Victor; 210.122.25.12
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Admin; 210.122.25.16
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Firewall; 210.122.25.1
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Vladimir; 210.122.25.4
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Oleg; 210.122.25.8
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Victor; 210.122.25.12
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Admin; 210.122.25.16
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Mail Alias: Victor.lan3.net
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Mail Alias: Igor.lan3.net
SRE : Getting NFS by utilite showmount
SRE : Getting NFS by utilite showmount
FUE : Finger Users Enumeration
FUE : Finger Users Enumeration
FUE : Finger Users Enumeration
TCBG : Telnet Connection Banner Grabbing                    Running Applications: Active directory; DNS ; WINS ; MS SQL Server 7; SNMP-agent; Microsoft Remote Registry Serv
TCBG : Telnet Connection Banner Grabbing                    Running Applications: Active directory; Kerberos; DNS ; WINS ; MS SQL Server 7; SNMP-agent; Microsoft Remote Re
FP : Connection to FTP server and examination of the prompt header   Running Applications: Active directory; Kerberos; DNS ; WINS ; MS SQL Server 7; SNMP-agent; Microsoft Remote Re
FP : Connection to FTP server and examination of the prompt header   Running Applications: Active directory; Kerberos; DNS ; WINS ; MS SQL Server 7; SNMP-agent; Microsoft Remote Re
AAF : Anonymity Access to FTP-server                        Anonymous Access to Ftp-server was gained successfully
DC : Network Ping Sweeps                                    IP-addresses: 210.122.25.16
TS : Telnet Connection and SYST command execution
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Firewall; 210.122.25.1
NS : Collection of additional information from DNS-server   From DNS-server [210.122.25.16] Host Name & Host IP: Vladimir; 210.122.25.4
```

Figure 11. Attack Scenario Realization.

The presented window depicts the fragment of attack development for the intention 7 ("Getting Access to Resources (GAR)"), where the hacker's IP-address is 161.43.201.148 and the host IP-address is 210.122.25.16.

In this window the information about attack is divided on the following four groups:

- the *attack task specification* units are mapped in the left top of the screen;
- to the right of them the *attack generation tree* is visualized;
- the strings of *generated malefactor's actions* are placed in the left part of the screen below the attack task specification;
- on the right of each malefactor's action a *tag of success (failure)* and *data obtained from an attacked host (a host response)* are depicted.

The *Attack task specification section* contains the information generated by the component of the attack task specification.

The graph showing the *Attack generation tree* represents a hierarchy of the malefactor's intentions and actions of different levels which correspond to non-terminal and terminal nodes. The non-terminal high level nodes are depicted by white ellipses. The terminal nodes of the attack model correspond to blue nodes. The brown node is the node of the current step of an attack scenario execution.

The transcriptions of the blue nodes can be seen in the section "Current non-terminal node".

All non-terminal nodes are realized as state machines.

When the attack scenario is been developed the strings with the following elements are appeared in the white window:

- Braun strings in left part of the diagram are descriptions of the *generated terminal malefactor's actions*.
- The result of each malefactor's action may be positive or negative. If the result is positive, the square block (designating the *tag of success*) is green, and green comments are printed from the right of the square block. The negative result means that the action was done unsuccessfully. The negative result is possible in two cases: if the attack is blocked by a firewall (in that case, the indicator and the comment are red); if the network response is negative (the indicator is grey, the comment is absent). When the string "END: Attack is over" is appeared, this means that a scenario realization is finished.

As shown in Figure 11, in the network attack implementation, each terminal action is performed on each host of the network, and in case of success or the attack being blocked by the firewall, right after the square block is the IP address of the host at which that terminal attack action was directed.

In case of success, the comment contains the decoding of the result obtained through that terminal action of the Hacker Agent, and the information obtained from the Network Agent as a result of the attacker’s action (that information may be absent).

In case of the hacker’s attack being blocked, the comment contains information on the reasons of the attack being blocked (either an illegal IP address of sender or receiver was detected, or the specified attack signature was detected), as well as the name of the firewall. If the attack was blocked on the level of the network firewall, then the IP address of the network is placed at the start of the comment.

After completion of the attack scenario the message “END: Attack is over” appears in the right part of the white window.

It is possible to look through the scenario tree by moving between the strings on the diagram.

The current scenario realization can be finished by closing the main dialog window. After that it is possible to begin another scenario.

Lastly, to stop an operation of agents it is necessary to press the button of an output 🚪 in windows “AIL Registry” and “AILab Agent Library :: Portal Component”.

THIS PAGE INTENTIONALLY LEFT BLANK

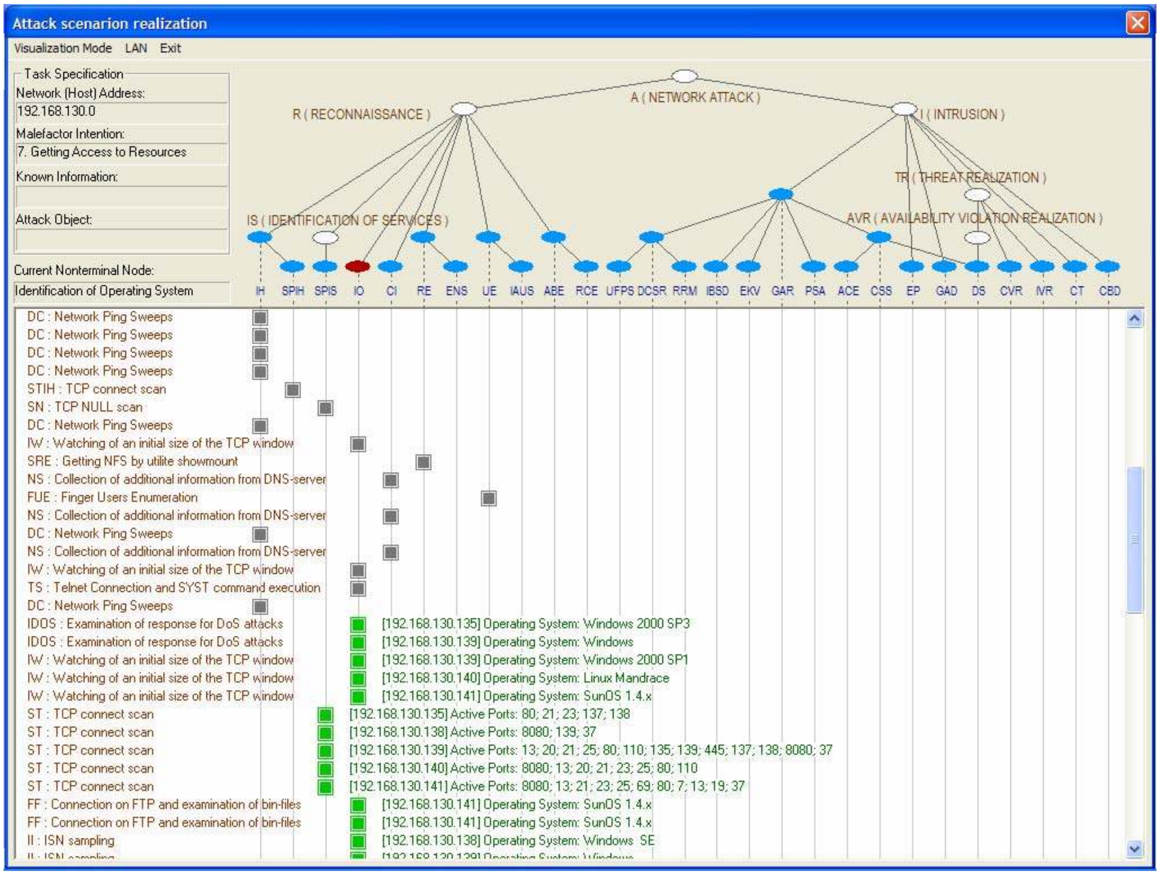
APPENDIX B

This appendix contains the results of Test Case 1 in Test Scenario 1.

Scenario 1

Test Case 1 - Run 1:

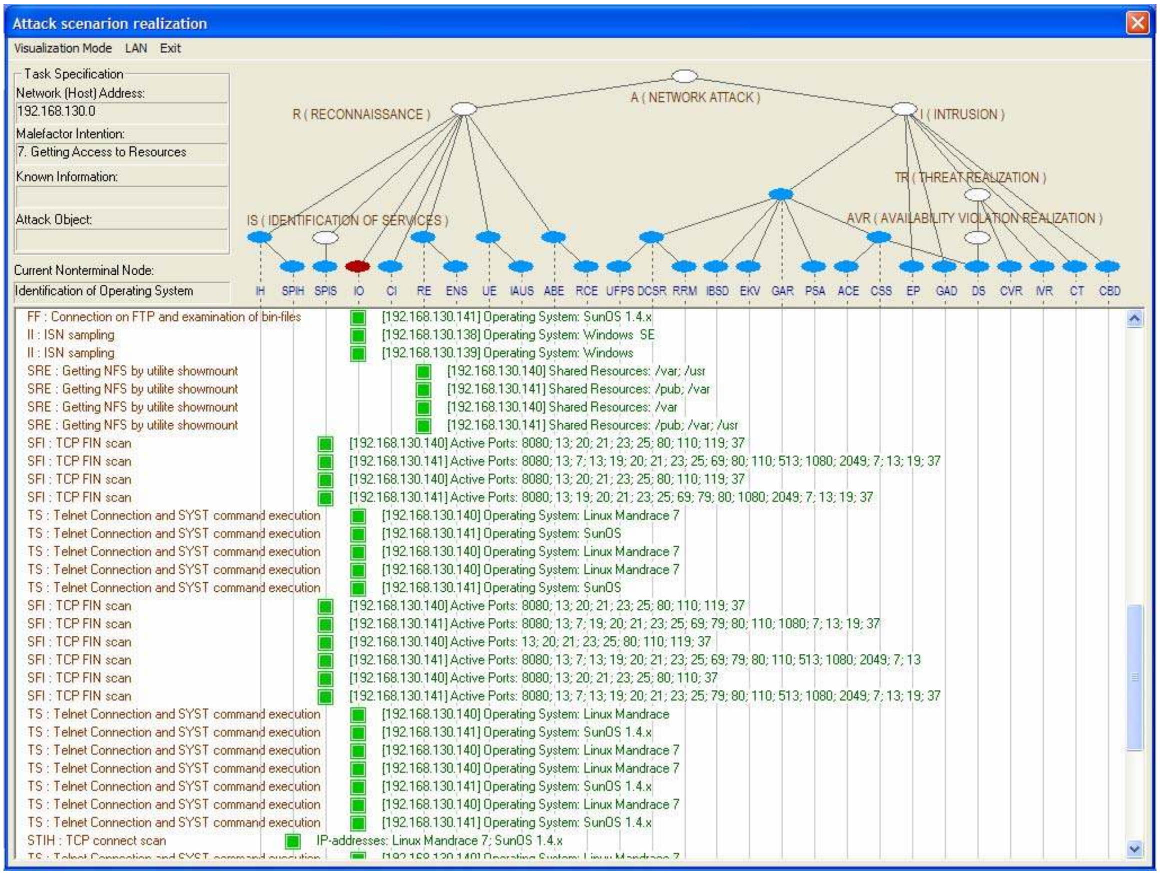
The MASDK tool is used to attack the network without any protective mechanism. At micro-level simulation with the setup of actual network, we will examine how fast the MASDK tool can infiltrate the network and set the backdoor at the hosts.



Task Specification
Network (Host) Address: 192.168.130.0
Malefactor Intention: 7. Getting Access to Resources
Known Information:
Attack Object:
Current Nonterminal Node: Identification of Operating System

```
DC : Network Ping Sweeps
DC : Network Ping Sweeps
DC : Network Ping Sweeps
DC : Network Ping Sweeps
STIH : TCP connect scan
SN : TCP NULL scan
DC : Network Ping Sweeps
IW : Watching of an initial size of the TCP window
SRE : Getting NFS by utilite showmount
NS : Collection of additional information from DNS-server
FUE : Finger Users Enumeration
NS : Collection of additional information from DNS-server
DC : Network Ping Sweeps
NS : Collection of additional information from DNS-server
IW : Watching of an initial size of the TCP window
TS : Telnet Connection and SYST command execution
DC : Network Ping Sweeps
IDOS : Examination of response for DoS attacks       [192.168.130.135] Operating System: Windows 2000 SP3
IDOS : Examination of response for DoS attacks       [192.168.130.139] Operating System: Windows
IW : Watching of an initial size of the TCP window   [192.168.130.139] Operating System: Windows 2000 SP1
IW : Watching of an initial size of the TCP window   [192.168.130.140] Operating System: Linux Mandrace
IW : Watching of an initial size of the TCP window   [192.168.130.141] Operating System: SunOS 1.4.x
ST : TCP connect scan                           [192.168.130.135] Active Ports: 80; 21; 23; 137; 138
ST : TCP connect scan                           [192.168.130.138] Active Ports: 8080; 139; 37
ST : TCP connect scan                           [192.168.130.139] Active Ports: 13; 20; 21; 25; 80; 110; 135; 139; 445; 137; 138; 8080; 37
ST : TCP connect scan                           [192.168.130.140] Active Ports: 8080; 13; 20; 21; 23; 25; 80; 110
ST : TCP connect scan                           [192.168.130.141] Active Ports: 8080; 13; 21; 23; 25; 69; 80; 7; 13; 19; 37
FF : Connection on FTP and examination of bin-files  [192.168.130.141] Operating System: SunOS 1.4.x
FF : Connection on FTP and examination of bin-files  [192.168.130.141] Operating System: SunOS 1.4.x
II : ISN sampling                                    [192.168.130.138] Operating System: Windows  SE
II : ISN sampling                                    [192.168.130.139] Operating System: Windows
```

Screen 1: We can observe that the attacker is doing a reconnaissance on the target network. The IP address and the service are identified by the attacker.



Task Specification

Network (Host) Address: 192.168.130.0

Malefactor Intention: 7. Getting Access to Resources

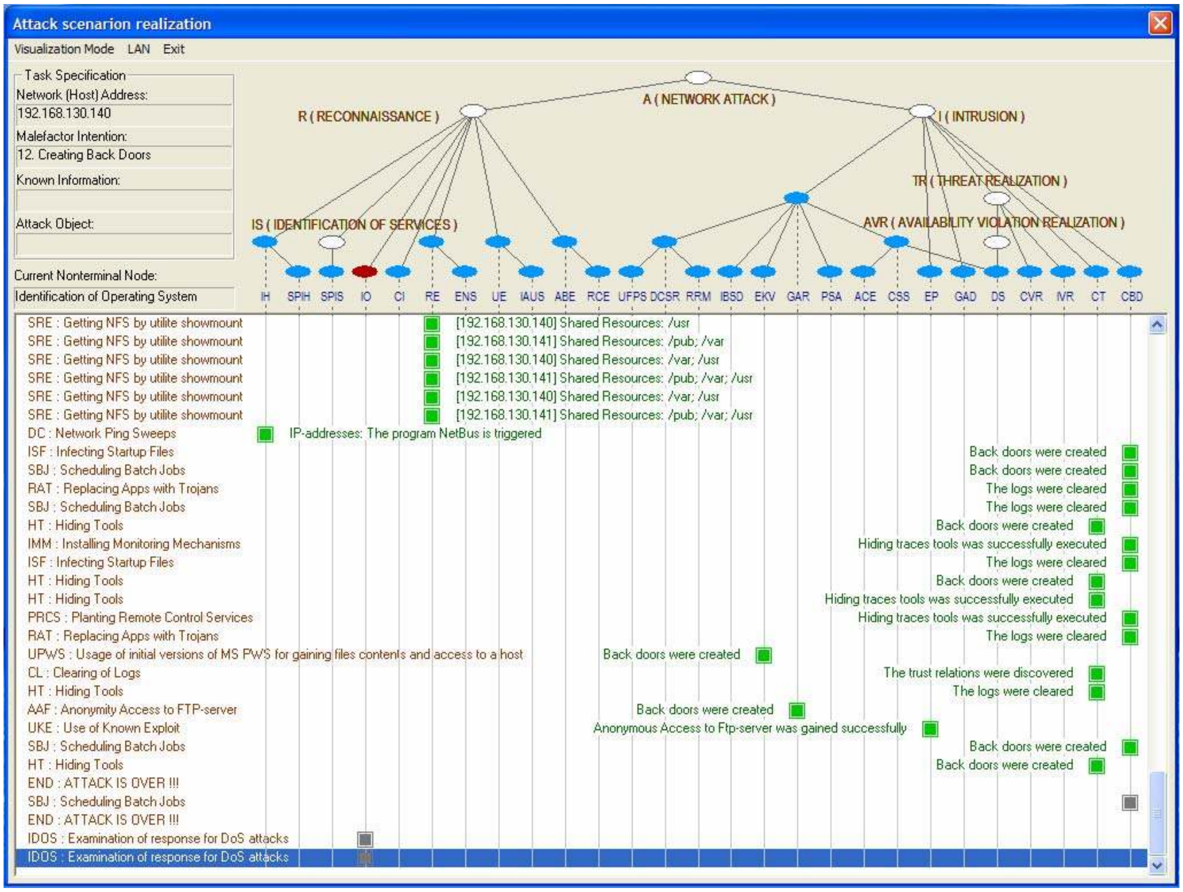
Known Information:

Attack Object:

Current Nonterminal Node: Identification of Operating System

```
FF : Connection on FTP and examination of bin-files   🟩 [192.168.130.141] Operating System: SunOS 1.4.x
II : ISN sampling                                     🟩 [192.168.130.138] Operating System: Windows  SE
II : ISN sampling                                     🟩 [192.168.130.139] Operating System: Windows
SRE : Getting NFS by utilite showmount                     🟩 [192.168.130.140] Shared Resources: /var; /usr
SRE : Getting NFS by utilite showmount                     🟩 [192.168.130.141] Shared Resources: /pub; /var
SRE : Getting NFS by utilite showmount                     🟩 [192.168.130.140] Shared Resources: /var
SRE : Getting NFS by utilite showmount                     🟩 [192.168.130.141] Shared Resources: /pub; /var; /usr
SFI : TCP FIN scan                                  🟩 [192.168.130.140] Active Ports: 8080; 13; 20; 21; 23; 25; 80; 110; 119; 37
SFI : TCP FIN scan                                  🟩 [192.168.130.141] Active Ports: 8080; 13; 7; 13; 19; 20; 21; 23; 25; 69; 80; 110; 513; 1080; 2049; 7; 13; 19; 37
SFI : TCP FIN scan                                  🟩 [192.168.130.140] Active Ports: 8080; 13; 20; 21; 23; 25; 80; 110; 119; 37
SFI : TCP FIN scan                                  🟩 [192.168.130.141] Active Ports: 8080; 13; 19; 20; 21; 23; 25; 69; 79; 80; 1080; 2049; 7; 13; 19; 37
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.140] Operating System: Linux Mandrace 7
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.141] Operating System: SunOS
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.140] Operating System: Linux Mandrace 7
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.140] Operating System: Linux Mandrace 7
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.141] Operating System: SunOS
SFI : TCP FIN scan                                  🟩 [192.168.130.140] Active Ports: 8080; 13; 20; 21; 23; 25; 80; 110; 119; 37
SFI : TCP FIN scan                                  🟩 [192.168.130.141] Active Ports: 8080; 13; 7; 19; 20; 21; 23; 25; 69; 79; 80; 110; 1080; 7; 13; 19; 37
SFI : TCP FIN scan                                  🟩 [192.168.130.140] Active Ports: 13; 20; 21; 23; 25; 80; 110; 119; 37
SFI : TCP FIN scan                                  🟩 [192.168.130.141] Active Ports: 8080; 13; 7; 13; 19; 20; 21; 23; 25; 69; 79; 80; 110; 513; 1080; 2049; 7; 13
SFI : TCP FIN scan                                  🟩 [192.168.130.140] Active Ports: 8080; 13; 20; 21; 23; 25; 80; 110; 37
SFI : TCP FIN scan                                  🟩 [192.168.130.141] Active Ports: 8080; 13; 7; 13; 19; 20; 21; 23; 25; 79; 80; 110; 513; 1080; 2049; 7; 13; 19; 37
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.140] Operating System: Linux Mandrace
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.141] Operating System: SunOS 1.4.x
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.140] Operating System: Linux Mandrace 7
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.140] Operating System: Linux Mandrace 7
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.141] Operating System: SunOS 1.4.x
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.140] Operating System: Linux Mandrace 7
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.141] Operating System: SunOS 1.4.x
STIH : TCP connect scan                             🟩 IP-addresses: Linux Mandrace 7; SunOS 1.4.x
TS : Telnet Connection and SYST command execution     🟩 [192.168.130.140] Operating System: Linux Mandrace 7
```

Screen 2: This screen shows that the attacker is doing a port scan on systems of the target network. The attacker has also succeeded in gaining access to the systems of the target network. The screen shows that the attack process is carried out without any obstruction.



Task Specification
Network (Host) Address:
192.168.130.140
Malefactor Intention:
12. Creating Back Doors
Known Information:

Attack Object:

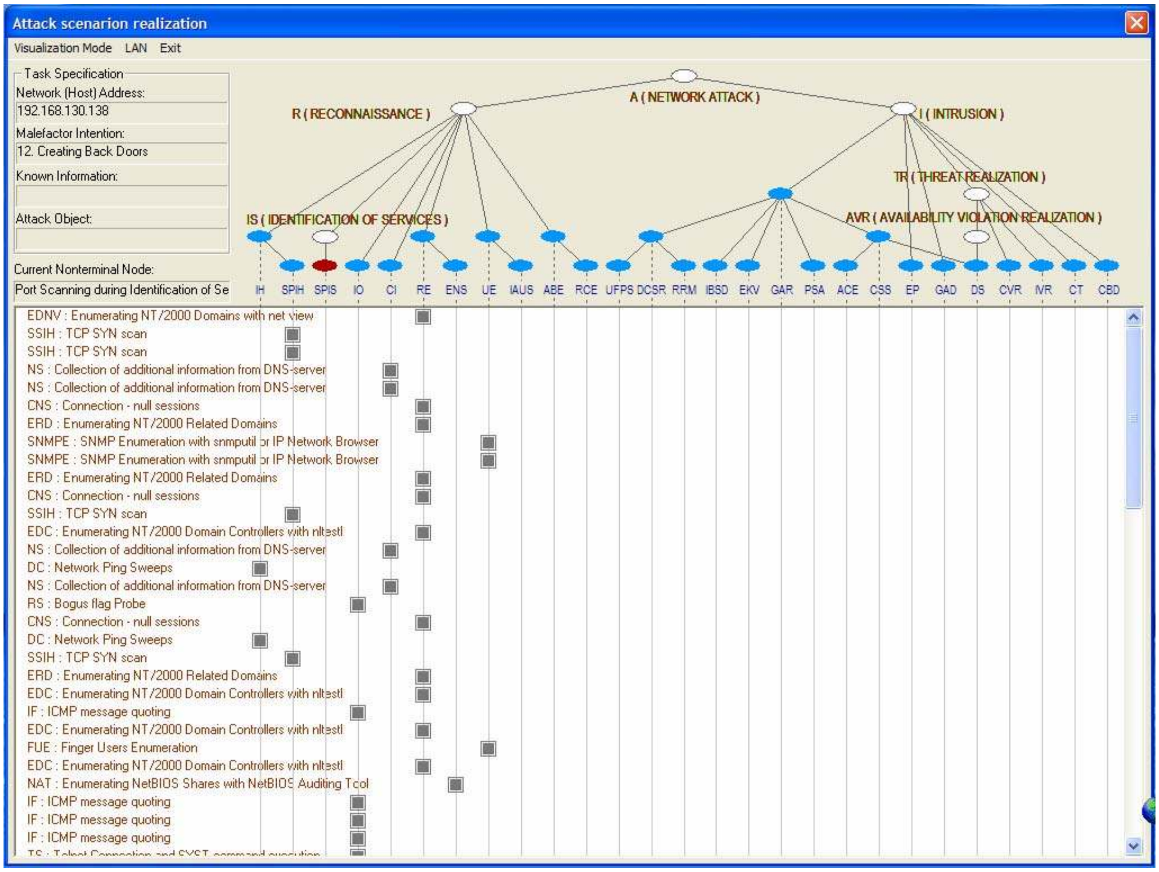
Current Nonterminal Node:
Identification of Operating System

```
SRE : Getting NFS by utilite showmount      [192.168.130.140] Shared Resources: /usr
SRE : Getting NFS by utilite showmount      [192.168.130.141] Shared Resources: /pub; /var
SRE : Getting NFS by utilite showmount      [192.168.130.140] Shared Resources: /var; /usr
SRE : Getting NFS by utilite showmount      [192.168.130.141] Shared Resources: /pub; /var; /usr
SRE : Getting NFS by utilite showmount      [192.168.130.140] Shared Resources: /var; /usr
SRE : Getting NFS by utilite showmount      [192.168.130.141] Shared Resources: /pub; /var; /usr
DC : Network Ping Sweeps      IP-addresses: The program NetBus is triggered
ISF : Infecting Startup Files      Back doors were created
SBJ : Scheduling Batch Jobs      Back doors were created
RAT : Replacing Apps with Trojans      The logs were cleared
SBJ : Scheduling Batch Jobs      The logs were cleared
HT : Hiding Tools      Back doors were created
IMM : Installing Monitoring Mechanisms      Hiding traces tools was successfully executed
ISF : Infecting Startup Files      The logs were cleared
HT : Hiding Tools      Back doors were created
HT : Hiding Tools      Hiding traces tools was successfully executed
PRCS : Planting Remote Control Services      Hiding traces tools was successfully executed
RAT : Replacing Apps with Trojans      The logs were cleared
UPWS : Usage of initial versions of MS PWS for gaining files contents and access to a host      Back doors were created
CL : Clearing of Logs      The trust relations were discovered
HT : Hiding Tools      The logs were cleared
AAF : Anonymity Access to FTP-server      Back doors were created
UKE : Use of Known Exploit      Anonymous Access to Ftp-server was gained successfully
SBJ : Scheduling Batch Jobs      Back doors were created
HT : Hiding Tools      Back doors were created
END : ATTACK IS OVER !!!
SBJ : Scheduling Batch Jobs
END : ATTACK IS OVER !!!
IDOS : Examination of response for DoS attacks
IDOS : Examination of response for DoS attacks
```

Screen 3: Shows that the attacker has succeeded in installing the backdoors on the target systems for future access and at the same time covering the track of the attack.

Test Case 1 - Run 2:

This test case is utilizing the MASDK tool to attack the network that has installed with protection mechanism 1 to 4. In the present version of MASDK simulation tool only the network Firewall is available.



Task Specification

Network (Host) Address: 192.168.130.138

Malefactor Intention: 12. Creating Back Doors

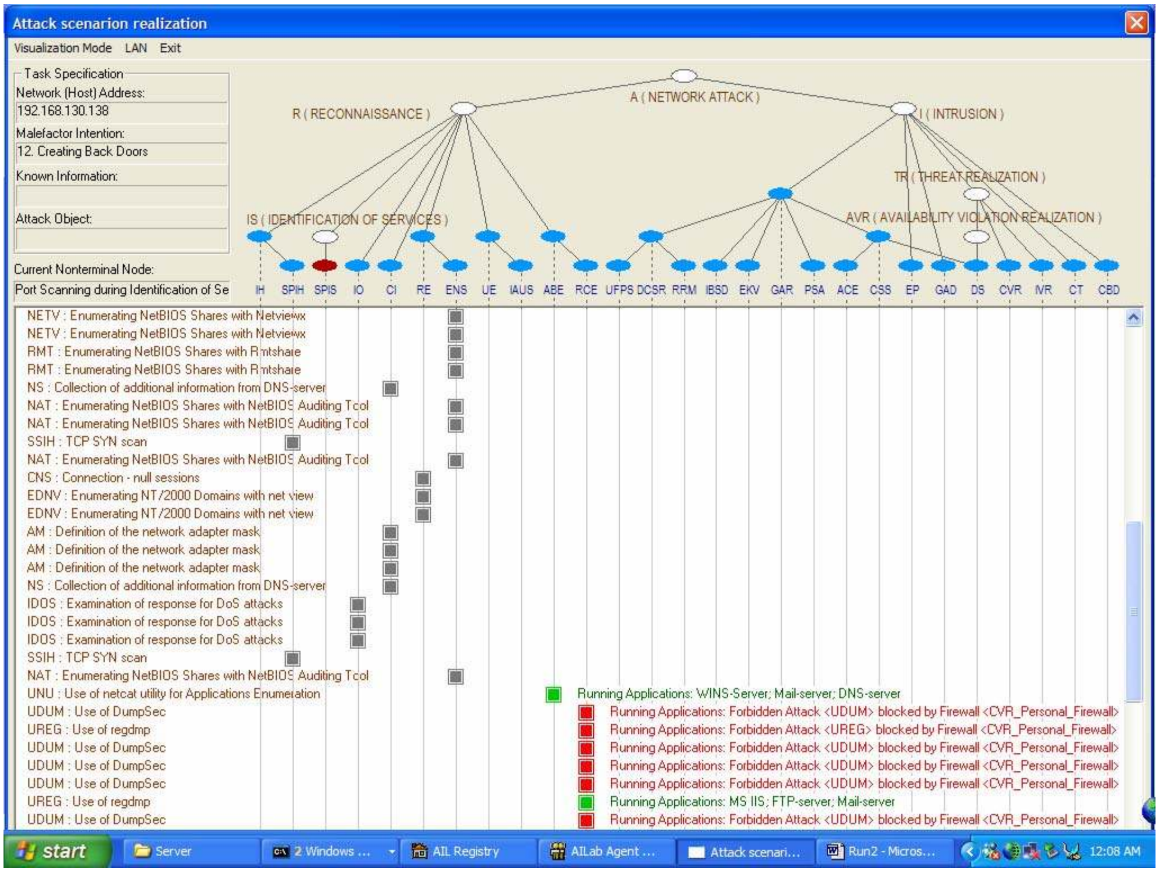
Known Information:

Attack Object:

Current Nonterminal Node: Port Scanning during Identification of Se

```
EDNV : Enumerating NT/2000 Domains with net view
SSIH : TCP SYN scan
SSIH : TCP SYN scan
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
CNS : Connection - null sessions
ERD : Enumerating NT/2000 Related Domains
SNMPE : SNMP Enumeration with snmputil or IP Network Browser
SNMPE : SNMP Enumeration with snmputil or IP Network Browser
ERD : Enumerating NT/2000 Related Domains
CNS : Connection - null sessions
SSIH : TCP SYN scan
EDC : Enumerating NT/2000 Domain Controllers with nltestl
NS : Collection of additional information from DNS-server
DC : Network Ping Sweeps
NS : Collection of additional information from DNS-server
RS : Bogus flag Probe
CNS : Connection - null sessions
DC : Network Ping Sweeps
SSIH : TCP SYN scan
ERD : Enumerating NT/2000 Related Domains
EDC : Enumerating NT/2000 Domain Controllers with nltestl
IF : ICMP message quoting
EDC : Enumerating NT/2000 Domain Controllers with nltestl
FUE : Finger Users Enumeration
EDC : Enumerating NT/2000 Domain Controllers with nltestl
NAT : Enumerating NetBIOS Shares with NetBIOS Auditing Tool
IF : ICMP message quoting
IF : ICMP message quoting
IF : ICMP message quoting
TS : Telnet Connection and SYST command execution
```

Screen 1: There is no response from the network to the attack activities that carried out by the Hacker Agent.



Task Specification

Network (Host) Address: 192.168.130.138

Malefactor Intention: 12. Creating Back Doors

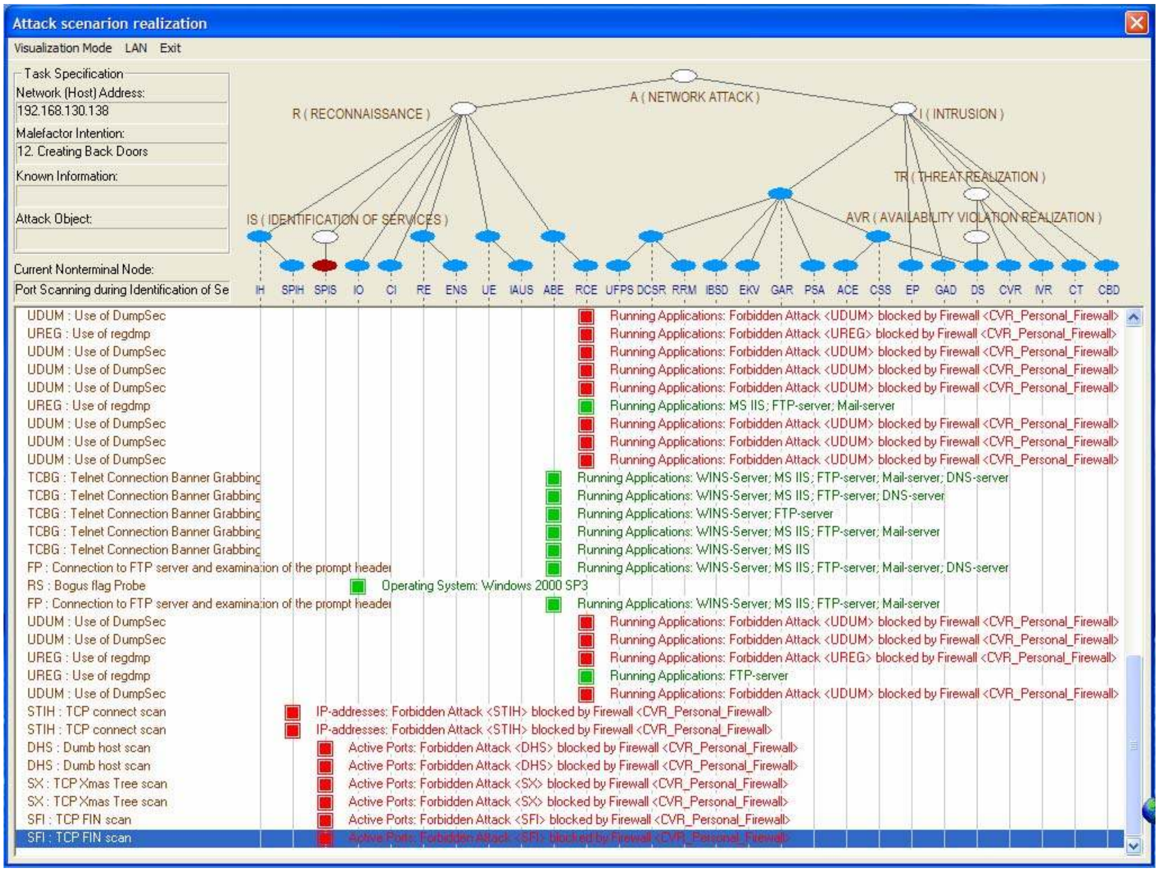
Known Information:

Attack Object:

Current Nonterminal Node: Port Scanning during Identification of Se

```
NETV : Enumerating NetBIOS Shares with Netviewx
NETV : Enumerating NetBIOS Shares with Netviewx
RMT : Enumerating NetBIOS Shares with Rmtshare
RMT : Enumerating NetBIOS Shares with Rmtshare
NS : Collection of additional information from DNS-server
NAT : Enumerating NetBIOS Shares with NetBIOS Auditing Tool
NAT : Enumerating NetBIOS Shares with NetBIOS Auditing Tool
SSIH : TCP SYN scan
NAT : Enumerating NetBIOS Shares with NetBIOS Auditing Tool
CNS : Connection - null sessions
EDNV : Enumerating NT/2000 Domains with net view
EDNV : Enumerating NT/2000 Domains with net view
AM : Definition of the network adapter mask
AM : Definition of the network adapter mask
AM : Definition of the network adapter mask
NS : Collection of additional information from DNS-server
IDOS : Examination of response for DoS attacks
IDOS : Examination of response for DoS attacks
IDOS : Examination of response for DoS attacks
SSIH : TCP SYN scan
NAT : Enumerating NetBIOS Shares with NetBIOS Auditing Tool
UNU : Use of netcat utility for Applications Enumeration   🟩 Running Applications: WINS-Server; Mail-server; DNS-server
UDUM : Use of DumpSec                                         🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UREG : Use of regdmp                                          🟥 Running Applications: Forbidden Attack <UREG> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec                                         🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec                                         🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec                                         🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UREG : Use of regdmp                                          🟩 Running Applications: MS IIS; FTP-server; Mail-server
UDUM : Use of DumpSec                                         🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
```

Screen 2: The Firewall is able to block most of the attack activities. The Hacker Agent has succeeded in identifying the services run in some of the network systems.



Task Specification

Network (Host) Address:
192.168.130.138

Malefactor Intention:
12. Creating Back Doors

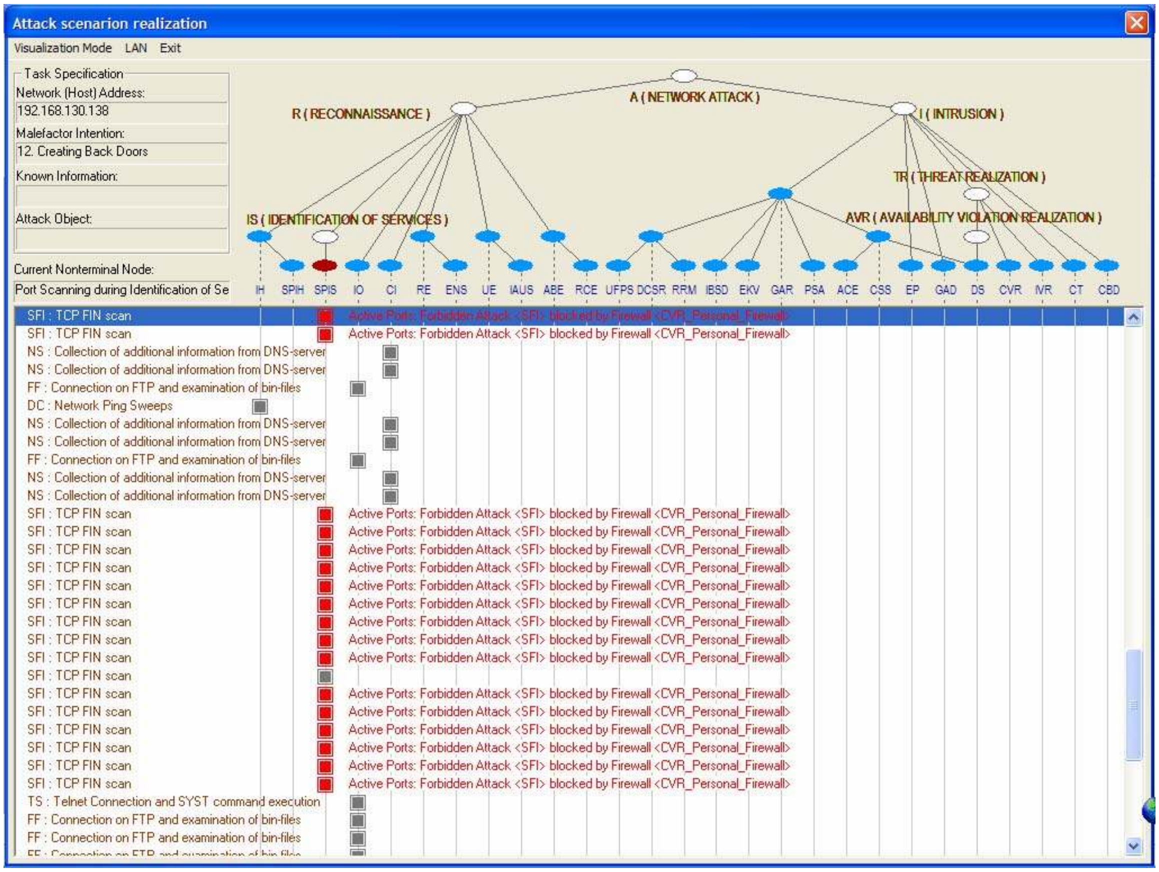
Known Information:

Attack Object:

Current Nonterminal Node:
Port Scanning during Identification of Se

```
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UREG : Use of regdmp       🟥 Running Applications: Forbidden Attack <UREG> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UREG : Use of regdmp       🟩 Running Applications: MS IIS; FTP-server; Mail-server
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
TCBG : Telnet Connection Banner Grabbinc   🟩 Running Applications: WINS-Server; MS IIS; FTP-server; Mail-server; DNS-server
TCBG : Telnet Connection Banner Grabbinc   🟩 Running Applications: WINS-Server; MS IIS; FTP-server; DNS-server
TCBG : Telnet Connection Banner Grabbinc   🟩 Running Applications: WINS-Server; FTP-server
TCBG : Telnet Connection Banner Grabbinc   🟩 Running Applications: WINS-Server; MS IIS; FTP-server; Mail-server
TCBG : Telnet Connection Banner Grabbinc   🟩 Running Applications: WINS-Server; MS IIS
FP : Connection to FTP server and examina:ion of the prompt header   🟩 Running Applications: WINS-Server; MS IIS; FTP-server; Mail-server; DNS-server
RS : Bogus flag Probe      🟩 Operating System: Windows 2000 SP3
FP : Connection to FTP server and examina:ion of the prompt header   🟩 Running Applications: WINS-Server; MS IIS; FTP-server; Mail-server
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
UREG : Use of regdmp       🟥 Running Applications: Forbidden Attack <UREG> blocked by Firewall <CVR_Personal_Firewall>
UREG : Use of regdmp       🟩 Running Applications: FTP-server
UDUM : Use of DumpSec      🟥 Running Applications: Forbidden Attack <UDUM> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan    🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan    🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
DHS : Dumb host scan       🟥 Active Ports: Forbidden Attack <DHS> blocked by Firewall <CVR_Personal_Firewall>
DHS : Dumb host scan       🟥 Active Ports: Forbidden Attack <DHS> blocked by Firewall <CVR_Personal_Firewall>
SX : TCP Xmas Tree scan    🟥 Active Ports: Forbidden Attack <SX> blocked by Firewall <CVR_Personal_Firewall>
SX : TCP Xmas Tree scan    🟥 Active Ports: Forbidden Attack <SX> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan         🟥 Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan         🟥 Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
```

Screen 3: The Hacker Agent went on to identify the OS of the systems on the network.



Task Specification

Network (Host) Address: 192.168.130.138

Malefactor Intention: 12. Creating Back Doors

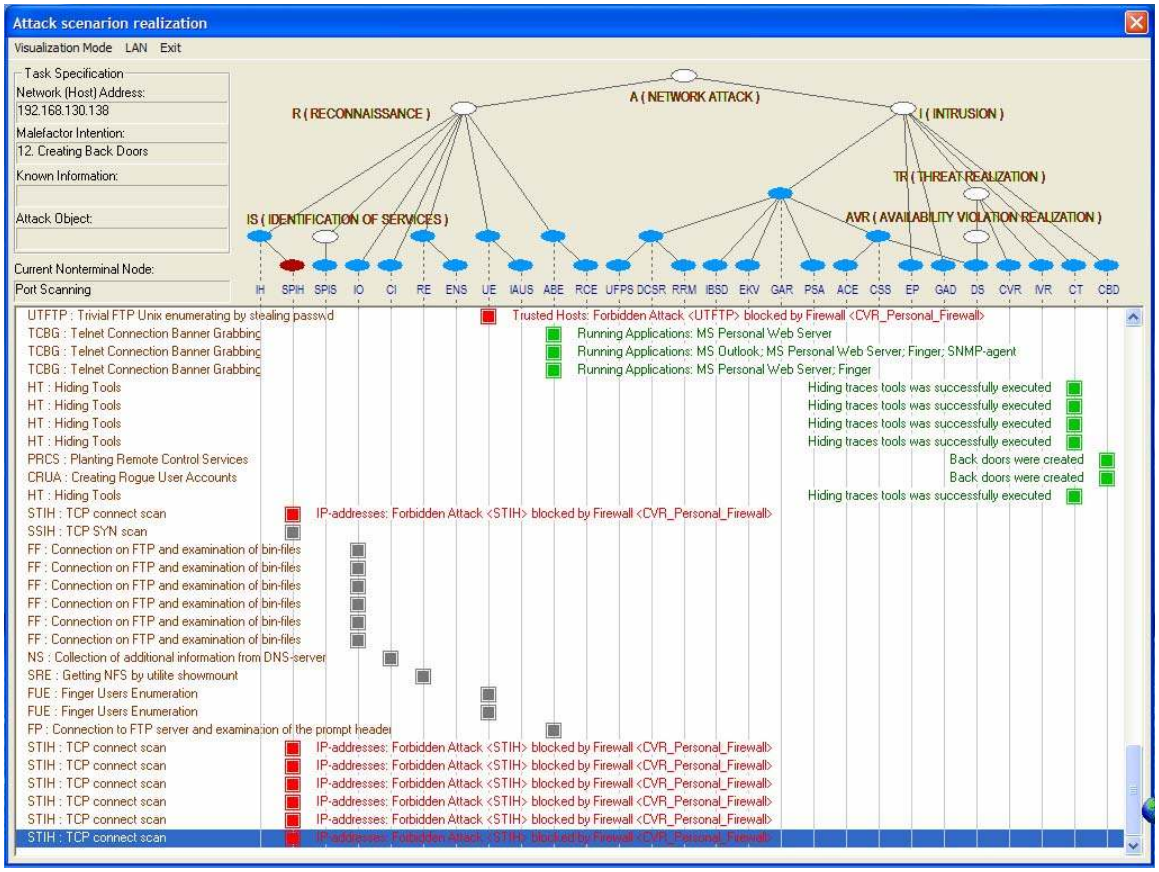
Known Information:

Attack Object:

Current Nonterminal Node: Port Scanning during Identification of Se

```
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
FF : Connection on FTP and examination of bin-files
DC : Network Ping Sweeps
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
FF : Connection on FTP and examination of bin-files
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan                                  Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
TS : Telnet Connection and SYST command execution
FF : Connection on FTP and examination of bin-files
FF : Connection on FTP and examination of bin-files
FF : Connection on FTP and examination of bin-files
```

Screen 4: The Hacker Agent tried other form of attack to gather information but was blocked by the Firewall.



Task Specification

Network (Host) Address: 192.168.130.138

Malefactor Intention: 12. Creating Back Doors

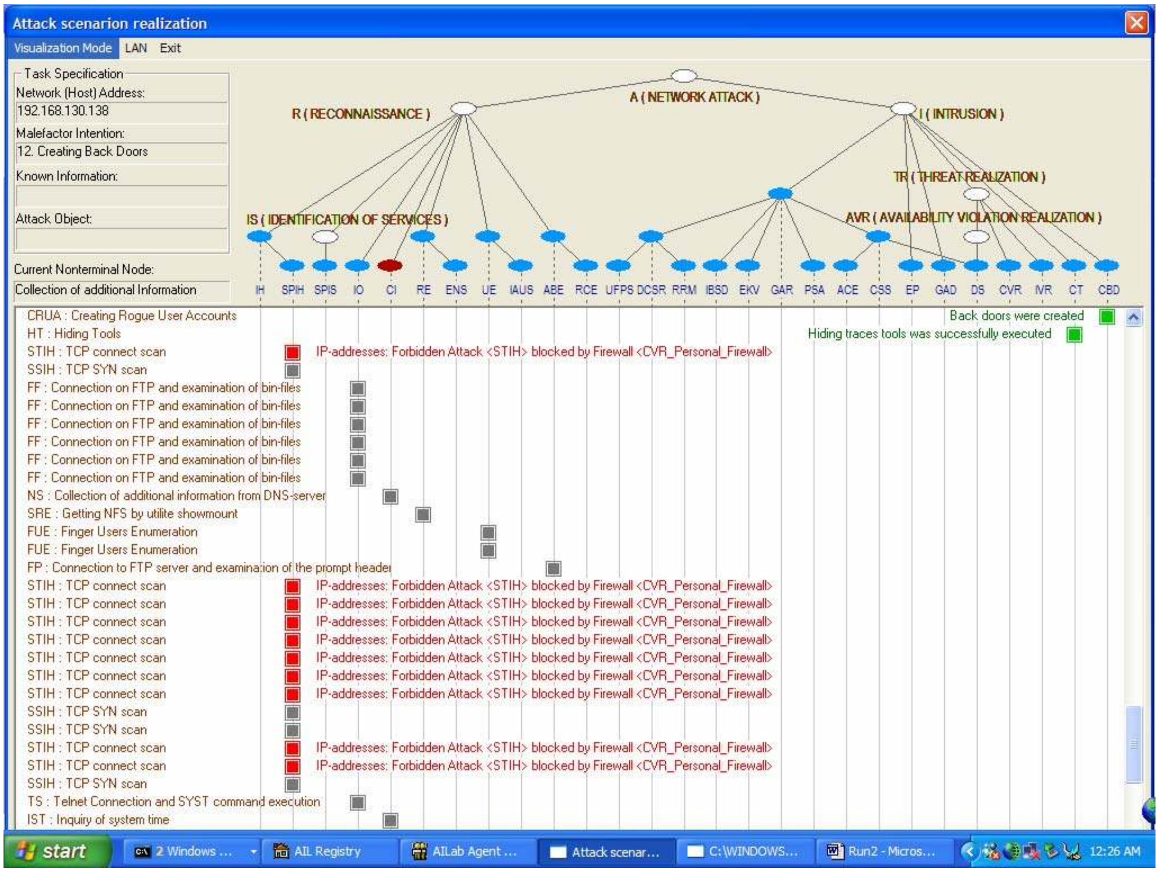
Known Information:

Attack Object:

Current Nonterminal Node: Port Scanning

```
UTFTP : Trivial FTP Unix enumerating by stealing passwd      Trusted Hosts: Forbidden Attack <UTFTP> blocked by Firewall <CVR_Personal_Firewall>
TCBG : Telnet Connection Banner Grabbing      Running Applications: MS Personal Web Server
TCBG : Telnet Connection Banner Grabbing      Running Applications: MS Outlook; MS Personal Web Server; Finger; SNMP-agent
TCBG : Telnet Connection Banner Grabbing      Running Applications: MS Personal Web Server; Finger
HT : Hiding Tools      Hiding traces tools was successfully executed
HT : Hiding Tools      Hiding traces tools was successfully executed
HT : Hiding Tools      Hiding traces tools was successfully executed
HT : Hiding Tools      Hiding traces tools was successfully executed
PRCS : Planting Remote Control Services      Back doors were created
CRUA : Creating Rogue User Accounts      Back doors were created
HT : Hiding Tools      Hiding traces tools was successfully executed
STIH : TCP connect scan      IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
SSIH : TCP SYN scan
FF : Connection on FTP and examination of bin-files
FF : Connection on FTP and examination of bin-files
FF : Connection on FTP and examination of bin-files
FF : Connection on FTP and examination of bin-files
FF : Connection on FTP and examination of bin-files
FF : Connection on FTP and examination of bin-files
NS : Collection of additional information from DNS-server
SRE : Getting NFS by utilite showmount
FUE : Finger Users Enumeration
FUE : Finger Users Enumeration
FP : Connection to FTP server and examination of the prompt header
STIH : TCP connect scan      IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan      IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan      IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan      IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan      IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan      IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
```

Screen 5: The Hacker Agent succeeded in installing two systems with backdoor in the network.



Task Specification

Network (Host) Address: 192.168.130.138

Malefactor Intention: 12. Creating Back Doors

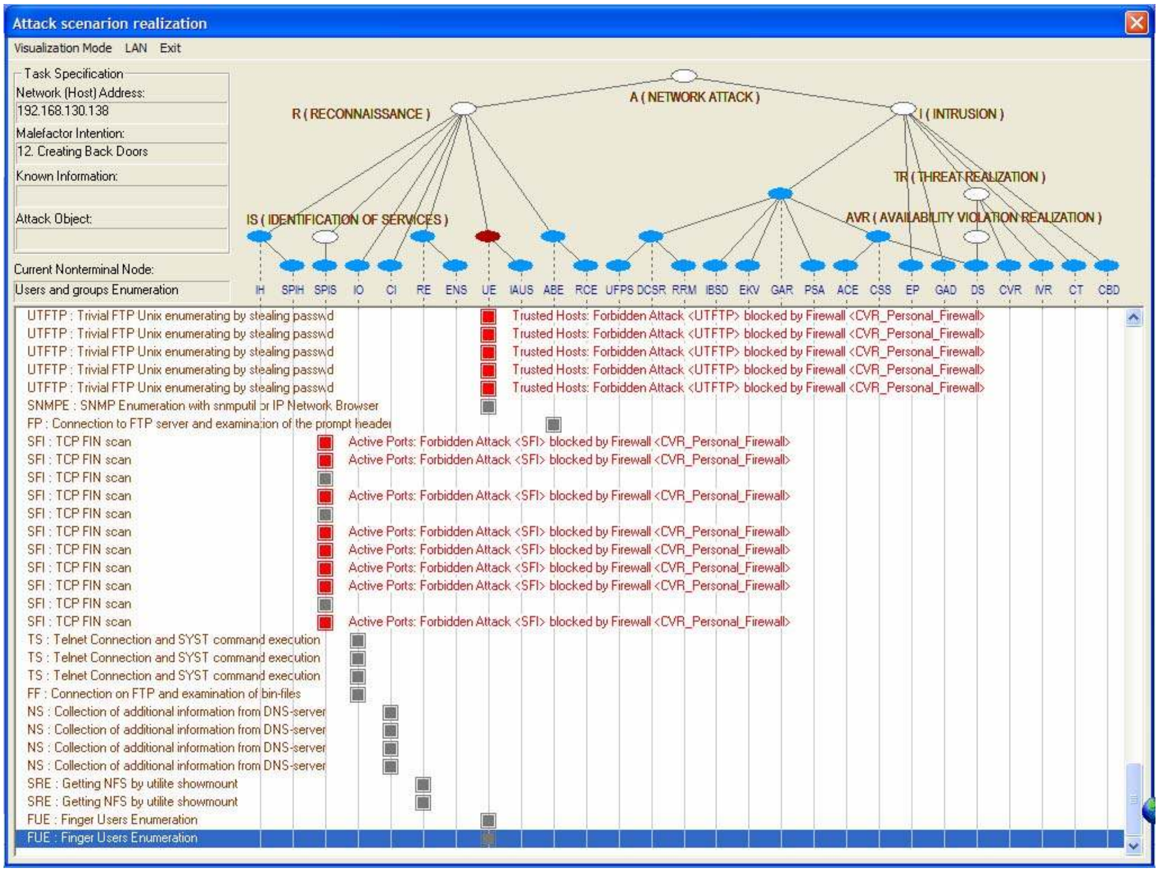
Known Information:

Attack Object:

Current Nonterminal Node: Collection of additional Information

```
CRUA : Creating Rogue User Accounts                                                   Back doors were created 🟩
HT : Hiding Tools                                                      Hiding traces tools was successfully executed 🟩
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
SSIH : TCP SYN scan              ■
FF : Connection on FTP and examination of bin-files  ■
FF : Connection on FTP and examination of bin-files  ■
FF : Connection on FTP and examination of bin-files  ■
FF : Connection on FTP and examination of bin-files  ■
FF : Connection on FTP and examination of bin-files  ■
FF : Connection on FTP and examination of bin-files  ■
NS : Collection of additional information from DNS-server  ■
SRE : Getting NFS by utilite showmount  ■
FUE : Finger Users Enumeration  ■
FUE : Finger Users Enumeration  ■
FP : Connection to FTP server and examination of the prompt header  ■
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
SSIH : TCP SYN scan              ■
SSIH : TCP SYN scan              ■
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
STIH : TCP connect scan          🟥 IP-addresses: Forbidden Attack <STIH> blocked by Firewall <CVR_Personal_Firewall>
SSIH : TCP SYN scan              ■
TS : Telnet Connection and SYST command execution  ■
IST : Inquiry of system time  ■
```

Screen 6: The Hacker Agent tried other form of attack but was blocked by the Firewall.



Attack scenarion realization

Task Specification

Network (Host) Address: 192.168.130.138

Malefactor Intention: 12. Creating Back Doors

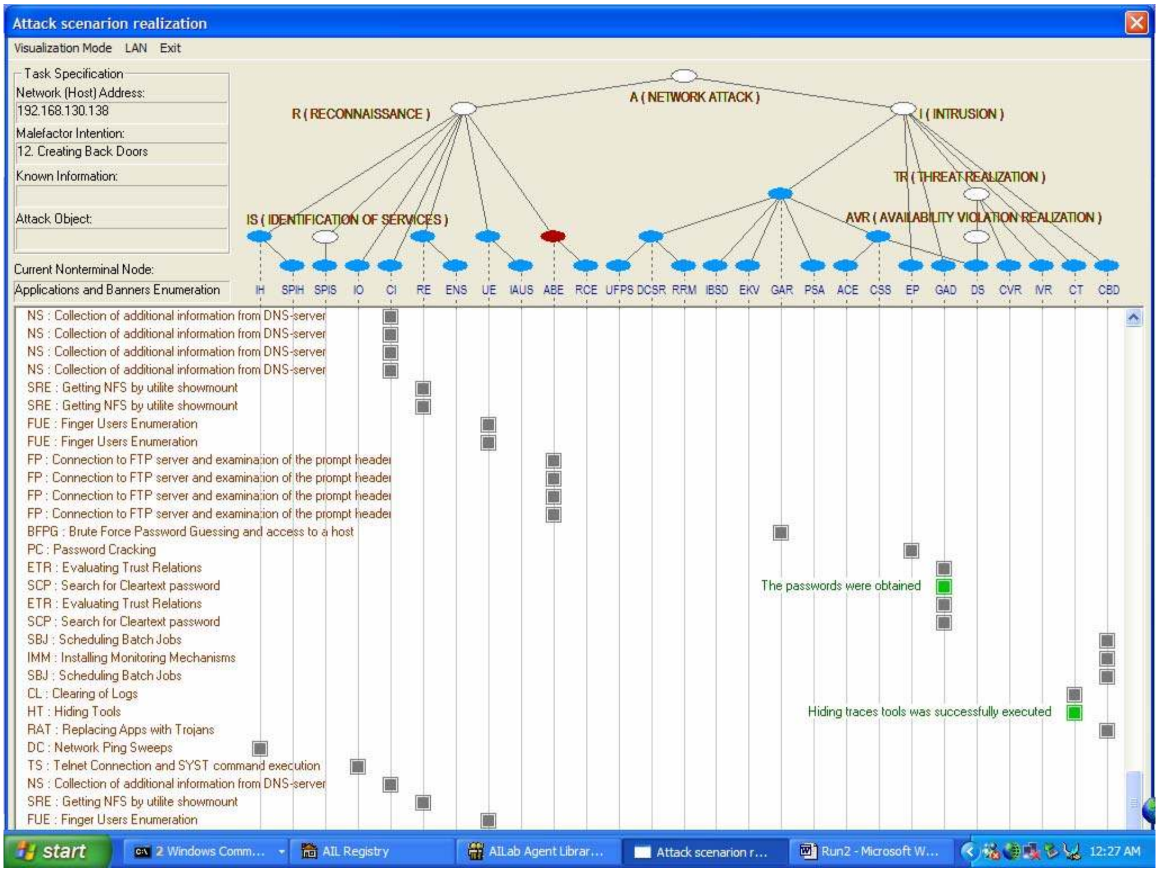
Known Information:

Attack Object:

Current Nonterminal Node: Users and groups Enumeration

```
UTFTP : Trivial FTP Unix enumerating by stealing passwd   Trusted Hosts: Forbidden Attack <UTFTP> blocked by Firewall <CVR_Personal_Firewall>
UTFTP : Trivial FTP Unix enumerating by stealing passwd   Trusted Hosts: Forbidden Attack <UTFTP> blocked by Firewall <CVR_Personal_Firewall>
UTFTP : Trivial FTP Unix enumerating by stealing passwd   Trusted Hosts: Forbidden Attack <UTFTP> blocked by Firewall <CVR_Personal_Firewall>
UTFTP : Trivial FTP Unix enumerating by stealing passwd   Trusted Hosts: Forbidden Attack <UTFTP> blocked by Firewall <CVR_Personal_Firewall>
UTFTP : Trivial FTP Unix enumerating by stealing passwd   Trusted Hosts: Forbidden Attack <UTFTP> blocked by Firewall <CVR_Personal_Firewall>
SNMPE : SNMP Enumeration with snmputil or IP Network Browser
FP : Connection to FTP server and examination of the prompt header
SFI : TCP FIN scan   Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan   Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan
SFI : TCP FIN scan   Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan
SFI : TCP FIN scan   Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan   Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan   Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan   Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
SFI : TCP FIN scan
SFI : TCP FIN scan   Active Ports: Forbidden Attack <SFI> blocked by Firewall <CVR_Personal_Firewall>
TS : Telnet Connection and SYST command execution
TS : Telnet Connection and SYST command execution
TS : Telnet Connection and SYST command execution
FF : Connection on FTP and examination of bin-files
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
SRE : Getting NFS by utilite showmount
SRE : Getting NFS by utilite showmount
FUE : Finger Users Enumeration
FUE : Finger Users Enumeration
```

Screen 7: The Hacker Agent repeat some of the attack cycle but was blocked by the Firewall.



Task Specification

Network (Host) Address: 192.168.130.138

Malefactor Intention: 12. Creating Back Doors

Known Information:

Attack Object:

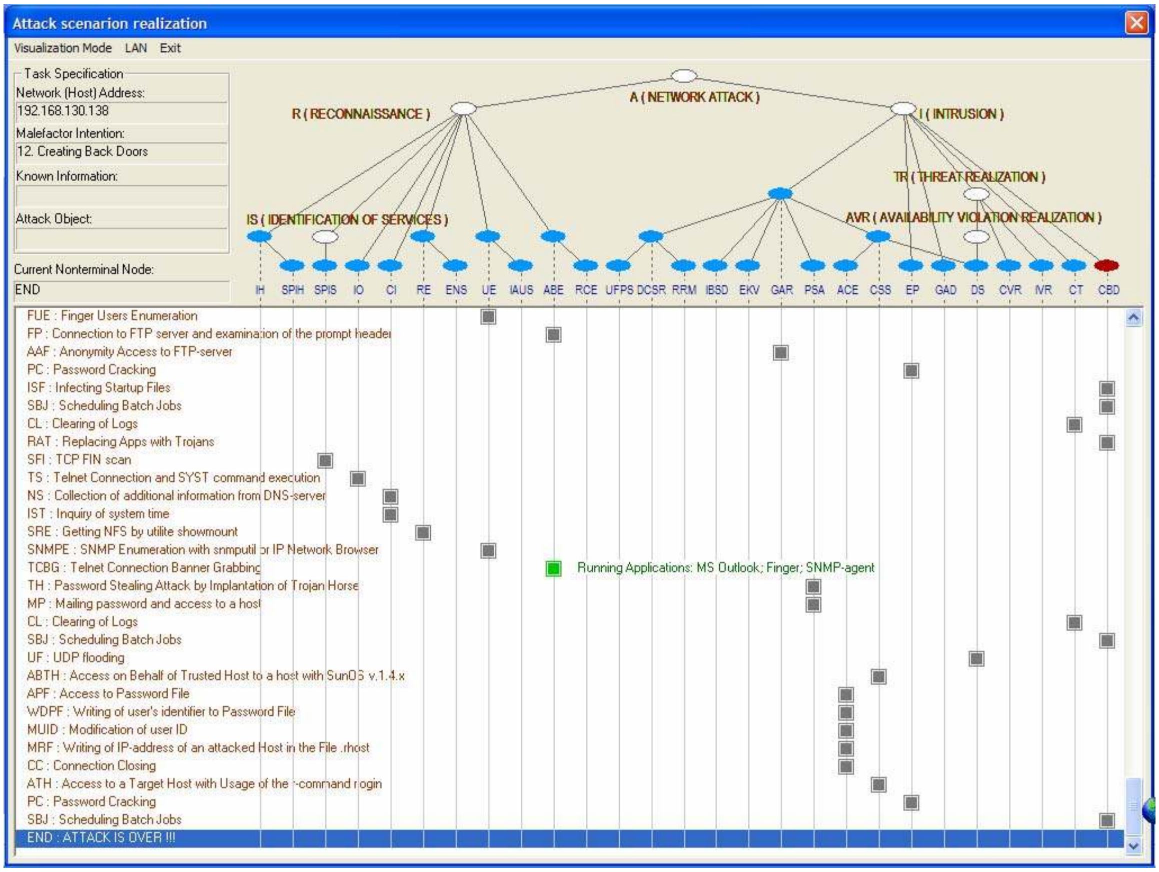
Current Nonterminal Node: Applications and Banners Enumeration

```
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
NS : Collection of additional information from DNS-server
SRE : Getting NFS by utilite showmount
SRE : Getting NFS by utilite showmount
FUE : Finger Users Enumeration
FUE : Finger Users Enumeration
FP : Connection to FTP server and examination of the prompt header
FP : Connection to FTP server and examination of the prompt header
FP : Connection to FTP server and examination of the prompt header
FP : Connection to FTP server and examination of the prompt header
BFPG : Brute Force Password Guessing and access to a host
PC : Password Cracking
ETR : Evaluating Trust Relations
SCP : Search for Cleartext password
ETR : Evaluating Trust Relations
SCP : Search for Cleartext password
SBJ : Scheduling Batch Jobs
IMM : Installing Monitoring Mechanisms
SBJ : Scheduling Batch Jobs
CL : Clearing of Logs
HT : Hiding Tools
RAT : Replacing Apps with Trojans
DC : Network Ping Sweeps
TS : Telnet Connection and SYST command execution
NS : Collection of additional information from DNS-server
SRE : Getting NFS by utilite showmount
FUE : Finger Users Enumeration
```

The passwords were obtained

Hiding traces tools was successfully executed

Screen 8: The Hacker Agent has obtained the password of the network systems.



Task Specification
Network (Host) Address:
192.168.130.138
Malefactor Intention:
12. Creating Back Doors
Known Information:

Attack Object:

Current Nonterminal Node:
END

```
FUE : Finger Users Enumeration
FP : Connection to FTP server and examination of the prompt header
AAF : Anonymity Access to FTP-server
PC : Password Cracking
ISF : Infecting Startup Files
SBJ : Scheduling Batch Jobs
CL : Clearing of Logs
RAT : Replacing Apps with Trojans
SFI : TCP FIN scan
TS : Telnet Connection and SYST command execution
NS : Collection of additional information from DNS-server
IST : Inquiry of system time
SRE : Getting NFS by utilite showmount
SNMPE : SNMP Enumeration with snmputil or IP Network Browser
TCBG : Telnet Connection Banner Grabbing        Running Applications: MS Outlook; Finger; SNMP-agent
TH : Password Stealing Attack by Implantation of Trojan Horse
MP : Mailing password and access to a host
CL : Clearing of Logs
SBJ : Scheduling Batch Jobs
UF : UDP flooding
ABTH : Access on Behalf of Trusted Host to a host with SunOS v.1.4.x
APF : Access to Password File
WDPF : Writing of user's identifier to Password File
MUID : Modification of user ID
MRF : Writing of IP-address of an attacked Host in the File .rhost
CC : Connection Closing
ATH : Access to a Target Host with Usage of the r-command rlogin
PC : Password Cracking
SBJ : Scheduling Batch Jobs
END : ATTACK IS OVER !!!
```

Screen 9: The Hacker Agent has successfully installed two backdoors in the network in the whole attack realization.

LIST OF REFERENCES

[Cabrera 2001] Cabrera, J. B. D., Lewis, L., Qin, X., Lee, W., Prasanth, R. K., Ravichandran, B., and Mehra, R. K.. "Proactive detection of distributed denial of service attacks using MIB traffic variables – A feasibility study." In *Proc. Int. Symposium on Integrated Network Management*, IEEE (Seattle, Wash., May 2001), pp. 609-622.

[Cherry 2003] Cherry, S. M. "Hell month," *IEEE Spectrum* (Oct. 2003), p. 48.

[Cuppens 2002] Cuppens, F. and Miege, A. "Alert Correlation in a Cooperative Intrusion Detection Framework." In *Proc. Symposium on Security and Privacy* (Berkeley, Calif., May 2002), pp. 187-200.

[Dittrich 1999a] Dittrich, D. "The Stacheldraht Distributed Denial of Service Attack Tool." http://staff.washington.edu/dittrich/misc/stacheldraht.analysis. 31 Dec 1999.

[Dittrich 1999b] Dittrich, D. "The Tribe Flood Network Denial of Service Attack Tool." http://staff.washington.edu/dittrich/misc/tfn.analysis. 21 Oct. 1999.

[Dittrich 2003] Dittrich, D. "Distributed Denial of Service (DDoS) Attack Tools." [http://staff.washington.edu/dittrich/misc/ddos/]. Sept. 2003.

[Feinstein 2003] Feinstein, L., Schnackenberg, D., Balupari, R., and Kindred, D.. "Statistical approaches to DDoS attack detection and response." In *Proc. DARPA Information Survivability Conf. and Expo.*, IEEE (Washington, D.C., Apr. 2003), vol. 1, pp. 303-314.

[Gorodetsky 2003] Gorodetsky, V. I., Kotenko, I. V., and Michael, J. B. "Multi-agent modeling and simulation of distributed denial-of-service attacks on computer networks." In *Proc. Third Int. Conf. on Navy and Shipbuilding Nowadays*, St. Petersburg: Krylov Shipbuilding Research Institute (St. Petersburg, Russia), June 2003, pp. 38-47.

[Jain 1991] Jain, R. The *Art of Computer Systems Performance Analysis*. First Edition. New York: John Wiley & Sons, 1991. p. 30.

[Jakobsson 2003] Jakobsson, M. and Menczer, F. "Untraceable Email Cluster Bombs: On Agent-Based Distributed Denial of Service." RSA Security and University of Iowa. Unpublished manuscript. http://arxiv.org/PS_cache/cs/pdf/0305/0305042.pdf. May 2003.

[Kashiwa 2002] Kashiwa, D., Chen, E. Y., and Fuji, H. "Active shaping: A countermeasure against DDoS attacks." In *Proc. Second European Conf. on Universal Multiservice Networks*, IEEE (Colmar, France, Apr. 2002), pp. 171-179.

[Kotenko 2003a] Kotenko, I., Gorodetski, V., Karsayev, O. and Khabalov, A. "Software Development Kit for Multi-agent Systems Design and Implementation." Russian

Foundation of Basic Research and European Office of Aerospace R&D (Project #1994P). St. Petersburg Institute for Informatics and Automation, 2003.

[Kotenko 2003b] Kotenko, I. and Man'kov, E. "Experiments with Simulation of Attacks against Computer Networks". Computer Network Security, Second International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security MMM-ACNS 2003. St. Petersburg Institute for Informatics and Automation, Sept. 2003, pp 183 - 194.

[Michael 2003] Michael, J. B., Fragkos, G., and Auguston, M. "An Experiment in Software Decoy Design." *Security and Privacy in the Age of Uncertainty*. Boston, Mass.: Kluwer Academic Publishers, 2003, pp. 253-264.

[Mirković 2002] Mirković, J., Prier, G. and Reiher, P. "Attacking DDoS at the source." In *Proc. Tenth Int. Conf. on Network Protocols*, IEEE (Paris, France, Nov. 2002), pp. 312-321.

[Navratilova 2000] Viki Navratilova. "A brief history of Distributed Denial of service Attacks". Uniforum Chicago. Security Architect, BlueMeteor, Inc. 22 Aug. 2000. http://www.uniforum.chi.il.us/slides/ddos/sld001.htm

[Ning 2002] Peng Ning, Yun Cui and Douglas S. Reeves. "Constructing attack scenarios through correlation of intrusion alerts." In *Proc. Conf. on Computer and Communications Security*, IEEE (Nov. 2002), pp. 245-254.

[Paxson 2003] Paxson, V., Moore, D., Savage, S., Shannon, C., Staniford, S., and Weaver, N. "Inside the Slammer Worm." *IEEE Security & Privacy* (July-Aug. 2003). pp. 33-39.

[Scambray 2001] Scambray, J., McClure, S., and Kurtz, G. *Hacking Exposed: Network Security Secrets & Solutions*. 2nd Edition. Berkeley, Calif.: McGraw Hill, 2001.

[Sterne 2002] Sterne, D., Djahandari, K., Balupari, R., La Cholter, W., Babson, B., Wilson, B., Narasimhan, P., Purtell, A., Schnackenberg, D., Linden, S.. "Active network based DDoS defense." In *Proc. DARPA Active Network Conf. and Expo.*, IEEE (San Francisco, Calif., May 2002), pp. 193-203.

[Sung 2002] Sung, M. and Xu, J. "IP traceback-based intelligent packet Filtering: A novel technique for defending against Internet DDoS attacks." *IEEE Trans. Parallel and Distributed Systems* 14, 9 (Sept. 2003): 861-872.

[Vijayan 2003] Vijayan, J. "Blaster Variant May Cause DOS Attacks". Department of Homeland Security issues advisory. http://www.computerworld.com/securitytopics/security/story. 2003.

[Zhao 2001] Zhao, W.-W. and Qin, S.-Y. “The diagnosis of DDoS attack and a novel approach to optimizing control”. In *Proc. Int. Conf. on Info-Tech and Info-Net*, IEEE (Beijing, China: Oct. 2001), pp. 278-283.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. James Bret Michael
   Naval Postgraduate School
   Monterey, Calif.

4. Mikhail Auguston
   Naval Postgraduate School
   Monterey, California

5. Igor Kotenko
   St. Petersburg Institute for Informatics and Automation
   St. Petersburg, RUSSIA

6. Vladimir Gorodetsky
   St. Petersburg Institute for Informatics and Automation
   St. Petersburg, RUSSIA

7. Yeo Tat Soon
   Temasek Defense System Institute
   Singapore