To appear in *Proc. Third Int. Conf. on Navy and Shipbuilding Nowadays (NSN)*, Krylov Shipbuilding Research Institute (St. Petersburg, Russia, June 2003).

MULTI-AGENT MODELING AND SIMULATION OF DISTRIBUTED DENIAL-OF-SERVICE ATTACKS ON COMPUTER NETWORKS

Vladimir I. Gorodetsky*, Igor V. Kotenko*, J. Bret Michael**

*** St. Petersburg Institute for Informatics and Automation, St. Petersburg, Russia**

**** Naval Postgraduate School, Monterey, USA**

АННОТАЦИЯ

Переход к реализации военно-морских операций, базирующихся на современной парадигме использования информационно-телекоммуникационного пространства, имеет несомненное преимущество, связанное с применением распределенной компьютерной обработки информации для получения превосходства над противником. Однако, противник будет пытаться атаковать информационные инфраструктуры, используемые силами флотов для выполнения военно-морских операций. Одно из действенных средств, приводящих к нарушению работы этих инфраструктур, — это распределенные компьютерные атаки типа “Отказ в обслуживании” (DDoS-атаки). Основная цель таких атак состоит в нарушении или снижении возможностей доступа авторизованных пользователей к распределенным вычислительным ресурсам, а также их компрометации. Повышение живучести информационных систем и инфраструктур в условиях реализации противником таких атак требует разработки адекватного теоретического и практического фундамента. Одной из важных составляющих такого фундамента является наличие средств моделирования DDoS-атак. Формальный подход к моделированию полного спектра атак данного класса, ключевыми элементами которого является онтология DDoS-атак, механизмы командной работы программных агентов, реализующих DDoS-атаки, а также программный инструментарий для разработки многоагентных систем MASDK, используемый для построения системы моделирования атак, составляют основное содержание данной работы.

ABSTRACT

The move toward practical use of modern naval network-centric warfare (NCW) brings with it the benefits caused by applying distributed computing to gain superiority over its adversary. However, the adversary will attempt to attack information infrastructures used in NCW. One effective means of destruction of such infrastructures is the use of distributed denial-of-service (DDoS) attacks. The primary goal of such attacks is to break or reduce the availability of distributed computing resources to legitimate users, with second-order effects such as causing nodes in the infrastructure to crash or even become compromised. Increase of survivability of information systems and structures requires the development of both strict theoretical and practical basis. The availability of DDoS attacks modeling and simulation means would be a significant component of such a basis. The paper introduces a framework for modeling and software tool for simulation of a broad spectrum of DDoS attacks, which key building blocks are ontology of DDoS-attacks, mechanisms for teamwork of software agents representing the hackers performing DDoS attacks and multi-agent platform called Multi-Agent System Development Kit supporting the agent-based DDoS attack modeling and simulation technology.

1. INTRODUCTION

DDoS attacks are commonplace in both the private and public sectors. Although the U.S. Navy has been a specific target of DDoS attacks, such as the well-publicized successful attack in May 1998, it is not necessary for an adversary to attack a navy directly. For example, the U.S. Navy relies heavily on the private sector to provide access to the information infrastructure over which NCW is conducted; it has been deemed to be too costly for the military services to build their own redundant infrastructures. In other words, the Navy shares what is known as the Critical Information Infrastructure (CII) with non-military entities. Hence, if the computing nodes of enough civilian users—who comprise the majority of the users on the CII—are affected by a DDoS attack, it could cause a major disruption of service for all of the users of the CII.

Numerous techniques and tools have been developed to detect and defeat DDoS attacks. However, the sophisticated attackers—who we refer to as information warriors—continuously modify their DDoS programs to exploit newly discovered Achilles heels in the protection mechanisms, user-applications, middleware, and operating systems. In addition, the sophistication of DDoS attacks continues to grow, as evidenced by the appearance of coordinated multi-agent DDoS attacks.

This paper argues that 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. We introduce a formal paradigm for modeling and simulation of a broad spectrum of classes of DDoS attacks, with the following key building blocks: an ontology of DDoS-attacks and mechanisms for cooperation between software agents representing teams of hackers implementing DDoS-attacks, and a software tool called Multi-Agent System Development Kit for modeling and simulation of agent-based DDoS attacks.

In addition, this paper describes the structure of teams of agents, interaction-and-coordination mechanisms used by agents, hierarchy of agent plans specifications, agent role-assignment mechanisms, and the allocation of plans to agents. We present our approach for conducting experiments to evaluate of computer network security and assess security policy.

The rest of the paper is structured as follows. *Section 2* outlines a general approach for modeling and simulation of DDoS attacks by imitating hackers-agents teamwork. *Section 3* describes an ontology of DDoS attacks. *Section 4* presents specifications of the structure of DDoS agents and outlines agent interaction-and-coordination mechanisms for maintenance of action coordination, monitoring and restoration of agent functionality, and also maintenance of communication selectivity. *Section 5* outlines specifications of agent plans, agents' role-assignment and allocation of plans. *Section 6* contains a state-machine based representation of DDoS agents' teamwork. *Section 7* describes the suggested formal model of the attacked computer network. *Section 8* covers the architecture and implementation of the Attack Simulator tool, following by concluding remarks.

2. TEAMWORK OF HACKERS-AGENTS

Teamwork among agents is said to occur when the agents fulfill joint operations to achieve one or more common long-term goals in a dynamic external environment and in the presence of noise and counteraction of opponents. The teamwork is something greater than simply a coordinated set of personal actions of individual agents: agents collaborate by means of a coordinated activity in which they jointly perform planning and perform actions to achieve a common goal. The main challenge for achieving teamwork is to reconcile inconsistent plans and actions amongst the agents, as discussed in the literature on multi-agent systems [2, 6, 7, 8, 9].

In this paper, models of hackers-agents performing DDoS attacks are based on joint intention theory [2], shared plans theory [6], and combined theories of agents' teamwork [7, 8, 9]; the agents' teamwork is modeled by usage of the group (i.e., team) plan of the agents' actions.

Agent plans have the following features:

(1) the group plan requires the group of agents to reach consensus on how to coordinate the execution of their individual plans;

(2) each agent should cooperate by honoring its commitments with other agents to coordinate its actions (approved intention); and

(3) the group plan can contain both the components of the plans of the individual agents for the assigned operations, and plans of subgroups.

Every team has an agreement protocol for deciding who will execute the atomic actions within a plan. Two kinds of plans are distinguished: Full Shared Plan (FSP) and Partial Shared Plan (PSP) [6].

FSP describes in detail all aspects of joint operations of the team, which include joint beliefs of the agents and common agreement of the agents to execute joint operations according to some fully described order. This order contains the description of all particular actions and a set of conditions determining initialization of the actions' execution. However in practice the team does not have a FSP; instead it possesses only the *PSP*, representing some "section" of the mental state of the team in a specific context of teamwork activity. The purpose of the agents' communication with one another is to fill in the gaps of conditions of FSP. In some cases FSP cannot be constructed at all.

As in the joint intention theory, the basic elements allowing the team to perform a common task are common (group) intentions, but their structuring is carried out in the same way as the plans are structured in the shared plans theory [8, 9]. The common (group, individual) intention and commitment are associated with each node of the whole hierarchical plan; these intention and commitment are used to manage the execution of the whole plan. Each agent needs to possess the group beliefs about other teammates.

For achievement of the common beliefs at formation and abandonment of the common intentions, agents should communicate. All agents' communication is managed by means of common commitments built on the basis of common intentions. For this purpose it is supposed to use the special mechanism of agents reasoning about communications (e.g., when, between which agents, about what, what contents). It is assumed that agents communicate only when there can exist an inconsistency between their actions. This property is called “selectivity of communications.” It is important for reaction to unexpected changes in the environment, maintenance of the redistribution of roles of the agents failed or unable to execute some part of a general plan, and also at occurrence of previously unplanned actions [8, 9].

The developed *technology for creation of the hackers-agents’ team* (that is also applicable to other subject domains) consists in realization of the following chain of stages:

(1) formation of the subject domain ontology;

(2) design of the agents’ team structure;

(3) definition of agent interaction-and-coordination mechanisms (including roles and scenarios of an agents’ roles exchange);

(4) specification of the agents’ plans (specifying attack generation) as a hierarchy of attribute stochastic formal grammars;

(5) assignment of roles and allocation of plans to the agents;

(6) state-machine based representation of teamwork.

Formation of the subject domain ontology is an initial stage of the agents’ team creation. In any subject domain, design procedure supposes conceptual modeling, that is, design of the set of basic notions of the application domain, identification of the relations over them, and also description of the data and selection of the algorithms interpreting these notions and relations.

The agents’ team structure is described in terms of a hierarchy of group and individual roles in the common scenario. Leaves of the hierarchy correspond to individual roles of the particular agents, while intermediate nodes represent group roles.

The plan hierarchy specification is carried out for each role. For group plans it is necessary to explicitly express joint activity. Specification of each plan includes the following:

(a) entry conditions determining the plan execution start up;

(b) conditions determining when to halt execution of the plan (the plan is executed, impracticable or irrelevant on conditions);

(c) actions which are carried out at a team level as part of a common plan.

In this paper attribute stochastic formal grammars are interconnected through a substitution operation as a formal framework for specifying plan hierarchies.

The assignment of roles and allocation of plans to the agents is carried out in two stages: first, the plan is distributed in terms of roles, and second, each role is assigned to an agent. An agent can execute several roles. Agents can exchange roles in progress of the plan execution. Requirements for each role are formulated as the union of the requirements for those parts of the plan that are mapped to the role. The roles are divided into group and individual. Leaves of the plan hierarchy correspond to individual roles. Agents’ functionalities are derived automatically based on each agent’s roles.

For representation of the agents’ team operation in real-time, a *hierarchy of state machines* is used. These state machines are designed according to the hierarchy of attribute stochastic formal grammars specifying the plan hierarchy. The state machines choose the plan to execute and fulfill the chosen sub-plans in a cycle “agents’ actions - responses of environment.”

Coordination among agents is carried out by message exchange. As the team operates in a hostile environment, one or more of the agents can fail or crash. Restoration of lost functionality is carried out by means of redistribution of roles of the failed agent between other agents and cloning of new agents.

3. ONTOLOGY OF DDOS ATTACKS

The ontology comprises a hierarchy of notions specifying activities of the team of information warriors who aim to implement DDoS attacks at different levels of detail.

In this ontology, the hierarchy of nodes representing notions can be divided into two subsets according to the *macro-* and *micro-levels* of the domain specifications. All nodes of the ontology of DDoS attacks on the macro- and micro-levels of specification are divided into intermediate and terminal [5].

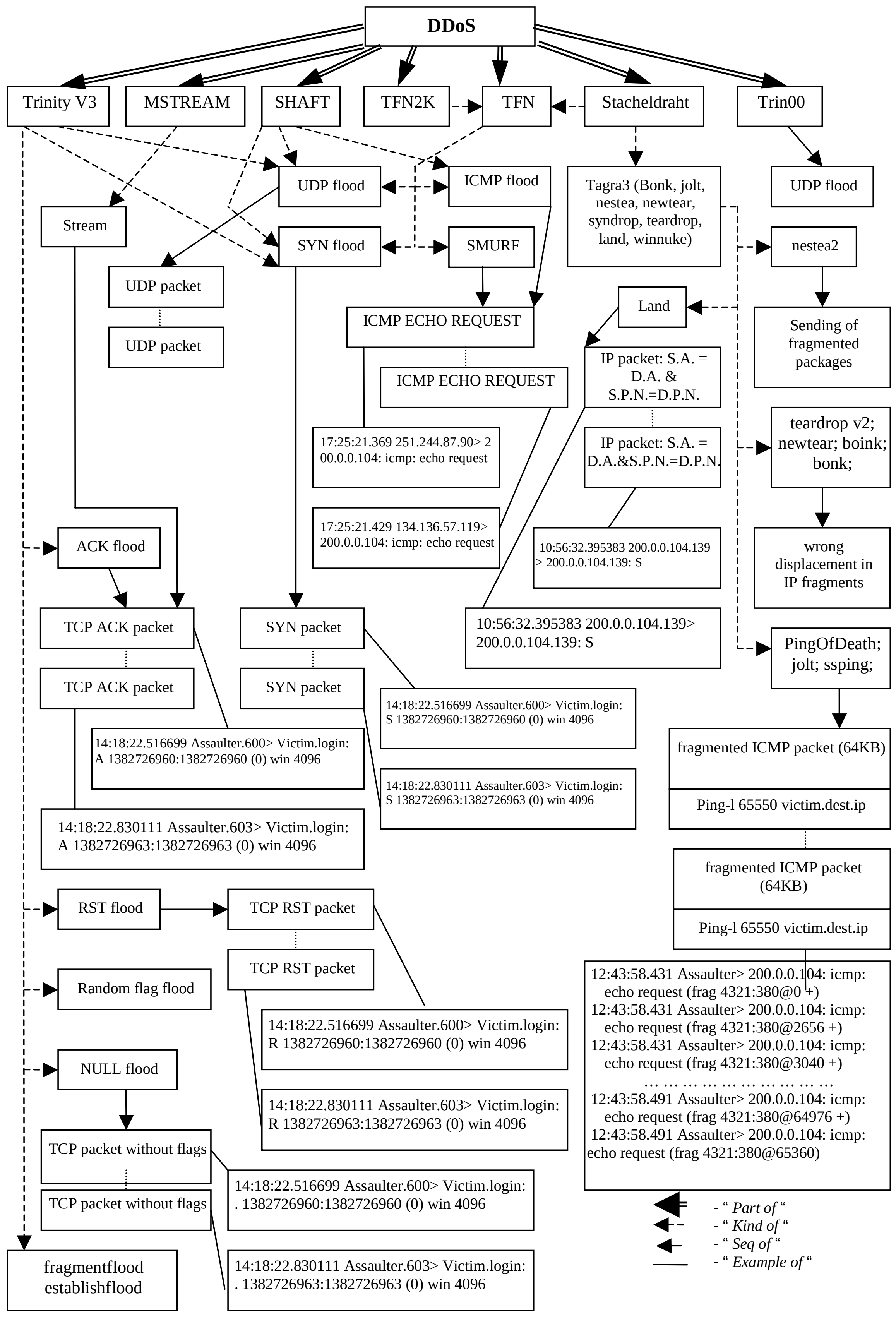
The notions of the ontology of an upper level can be interconnected with the corresponding notions of the lower level through one of the three kinds of *relationships*:

(1) “*Part of*” (decomposition);

(2) “*Kind of*” (specialization); and

(3) “*Seq of*” (sequence of operation).

High-level notions corresponding to the intentions form the upper levels of the ontology. These notions are interconnected by the “*Part of*” relationship. Attack actions realizing information warriors’ intentions (presented at lower levels with regard to intentions) are interconnected with the intentions by “*Kind of*” or “*Seq of*” relationship.


DDoS
Trinity V3
MSTREAM
SHAFT
TFN2K
TFN
Stacheldraht
Trin00
UDP flood
ICMP flood
Tagra3 (Bonk, jolt, nestea, newtear, syndrop, teardrop, land, winnuke)
UDP flood
Stream
SYN flood
SMURF
nestea2
UDP packet
UDP packet
ICMP ECHO REQUEST
ICMP ECHO REQUEST
Land
Sending of fragmented packages
IP packet: S.A. = D.A. & S.P.N.=D.P.N.
IP packet: S.A. = D.A.&S.P.N.=D.P.N.
teardrop v2; newtear; boink; bonk;
wrong displacement in IP fragments
ACK flood
TCP ACK packet
TCP ACK packet
SYN packet
SYN packet
PingOfDeath; jolt; ssping;
fragmented ICMP packet (64KB)
Ping-l 65550 victim.dest.ip
fragmented ICMP packet (64KB)
Ping-l 65550 victim.dest.ip
RST flood
TCP RST packet
TCP RST packet
Random flag flood
NULL flood
TCP packet without flags
TCP packet without flags
fragmentflood establishflood
- " *Part of* "
- " *Kind of* "
- " *Seq of* "
- " *Example of* "

```
17:25:21.369 251.244.87.90> 200.0.0.104: icmp: echo request
```

```
17:25:21.429 134.136.57.119> 200.0.0.104: icmp: echo request
```

```
10:56:32.395383 200.0.0.104.139 > 200.0.0.104.139: S
```

```
10:56:32.395383 200.0.0.104.139> 200.0.0.104.139: S
```

```
14:18:22.516699 Assaulter.600> Victim.login: S 1382726960:1382726960 (0) win 4096
```

```
14:18:22.830111 Assaulter.603> Victim.login: S 1382726963:1382726963 (0) win 4096
```

```
14:18:22.516699 Assaulter.600> Victim.login: A 1382726960:1382726960 (0) win 4096
```

```
14:18:22.830111 Assaulter.603> Victim.login: A 1382726963:1382726963 (0) win 4096
```

```
12:43:58.431 Assaulter> 200.0.0.104: icmp:
  echo request (frag 4321:380@0 +)
12:43:58.431 Assaulter> 200.0.0.104: icmp:
  echo request (frag 4321:380@2656 +)
12:43:58.431 Assaulter> 200.0.0.104: icmp:
  echo request (frag 4321:380@3040 +)
      … … … … … … … … … …
12:43:58.491 Assaulter> 200.0.0.104: icmp:
  echo request (frag 4321:380@64976 +)
12:43:58.491 Assaulter> 200.0.0.104: icmp:
echo request (frag 4321:380@65360)
```

```
14:18:22.516699 Assaulter.600> Victim.login: R 1382726960:1382726960 (0) win 4096
```

```
14:18:22.830111 Assaulter.603> Victim.login: R 1382726963:1382726963 (0) win 4096
```

```
14:18:22.516699 Assaulter.600> Victim.login: . 1382726960:1382726960 (0) win 4096
```

```
14:18:22.830111 Assaulter.603> Victim.login: . 1382726963:1382726963 (0) win 4096
```

Fig. 1. Fragment of the ontology of DDoS-attacks

The "terminal" notions of the macro-level are further elaborated on the *micro-level of attack specification***,** and on this level they belong to the set of top-level notions specified through the use of the relationships introduced above. In micro specifications of the computer network attacks ontology, besides the three relations described, the relationship "*Example of*" is also used. It serves to establish the "type of object – specific sample of object" relationship.

The ontology includes a detailed description of the DDoS domain in which the notions of the bottom level ("*terminals*") can be specified in terms of network packets, system calls, and audit data.

A fragment of the DDoS-attacks ontology is depicted in Fig.1. Nodes specifying a set of software exploits for generation of DDoS attacks (Trinity V3, MSTREAM, SHAFT, TFN2K, Stacheldraht, Trin00) make up a top level of the ontology fragment.

At lower levels shown in Fig.1, two classes of DoS-attacks are shown: "Land" attacks (sending an IP-packet with equal fields of port and address of the sender and the receiver, that is, Source Address = Destination Address, Source Port Number = Destination Port Number) and "Smurf" attacks (sending broadcasting ICMP ECHO inquiries on behalf of a victim host, with hosts that accept such packets replying to the victim host, thus resulting in an increase in the workload to be handled by the communication channel and, in some cases, full isolation of the attacked network).

4. AGENTS' INTERACTION-AND-COORDINATION

In the paper we model a team of agents as three-level structure (Fig.2). The team of agents consists of the "client" supervising a sub-team of "masters." Each master, in turn, manages a group of "daemons." Daemons execute immediate attack actions against victim hosts. Daemons consist of two subsets – scouts D_R and attackers D_A.

A set of DDoS agents can be described as a pair

$$A=\{M, D\},$$

where $M=\{m_1, m_2,\ldots, m_r\}$ – set of masters; $D=\{D_R, D_A\}=\{d_1, d_2,\ldots, d_n\}$ – set of demons.

Each agent can be represented as follows:

$$a_N = <K, B, De, I, P, G, C>,$$

where N – *identifier* of an agent; K – *knowledge* of the agent (a constant part of information of the agent about itself, environment and other agents, invariant during its use); B – *beliefs* of the agent (a variable part of information of the agent about itself, environment and other agents, which can be changed and become corrupted); De – *desires* of the agent (states that the agent wants to achieve, however these states can be inconsistent which is why the agent does not expect that all of the states can be reached); I – *intentions* of the agent (goals the agent has to reach or actions the agent has to fulfill because of commitments to other agents or the agent's desires, that is, a consistent subset of desires chosen for some reason to achieve that is compatible with the agent's commitments); P – a set of *parameters* determining a mode of operation of the agent, for example, minimal reaction time; G – a set of the *goals* and *actions* of the agent; C – *commitments* of the agent concerning other agents.

An operation of each agent can be represented as an alternation in a continuous cycle of phases (actions) on recognition of a current state, choice of action (in view of temporal constraints) and its performance.

For support of teamwork of DDoS agents, it is proposed to use three groups of mechanisms (procedures): (1) maintenance of action coordination, (2) monitoring and restoration of agent functionality, and (3) maintenance of communication selectivity (for choice of the most "useful" communication) [8].

The mechanisms of the first class are intended for realization of coordinated initialization and termination of actions. Coordinated initialization means that all members of a team (group) begin execution of the same plan at the defined time. It assumes an appointment of concrete agents to the fixed roles in a concrete scenario, their notification about the appointed scenario and role, and also

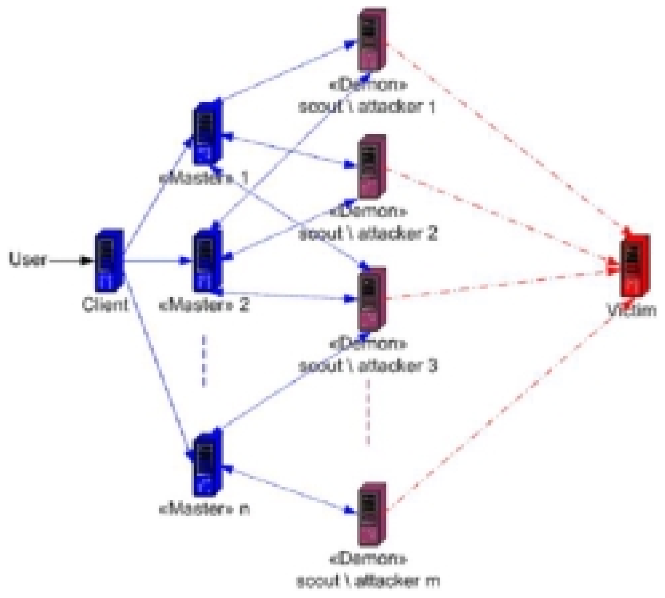

User
Client
«Master» 1
«Master» 2
«Master» n
«Demon» scout \ attacker 1
«Demon» scout \ attacker 2
«Demon» scout \ attacker 3
«Demon» scout \ attacker m
Victim

Fig. 2. Three-level structure of a team of agents

reception of confirmations on their readiness to execute the assigned role in the given scenario.

The coordinated termination of a common action (refusal of the common intention) also requires mutual notification of agents of the team about this action of the presence of specific conditions: such conditions can be determined by achieving the common goal, determining non-attainability of the goal, or recognizing the irrelevancy of the goal.

For example, the attack goal "increase of authority up to a level of superuser" is achieved if a certain hacker manages to penetrate a target host and raises its privilege level to that of superuser. The purpose is unattainable if one of obligatory actions to penetrate a target host is not executed. The purpose is irrelevant if the target host is switched off from the network.

Mechanisms of monitoring and restoration of agent functionality should provide supervision of some agents over others in order to identify the loss of capacity for work by the agent or a group of agents. The aim here is the rapid restoration of functionality of the team at the expense of reassignment of the "lost" roles to those teammates that can perform corresponding additional tasks.

For example, if one of the hackers-agents operating according to the intention "Identification of operating system of a host" is blocked by the firewall of a target network or other obstacle for achievement of this intention to take place, this agent (or other hacker-agent who discovered the ineffectiveness of a "colleague's" action) should send the respective information to a "leader" of the scenario. If there will be other agents capable of solving the task corresponding to this role, it should be assigned to this agent. Checking of rules and realization of reasoning entails respective communications of agents subject to communication protocols.

Mechanisms of maintenance of communication selectivity order the communication acts when the probability and cost of agents' coordination loss is great enough. These mechanisms are based on calculation of the message importance in view of the "costs" and benefits of this message. It is necessary to guarantee that the benefit of the message exchange for maintenance of agents' coordination surpasses the "cost" of the communication act (for example, a network security system, having intercepted agents' messages, can detect and "suppress" an attack). Therefore it is very important to choose those communication acts that will bring the greatest benefit to the team.

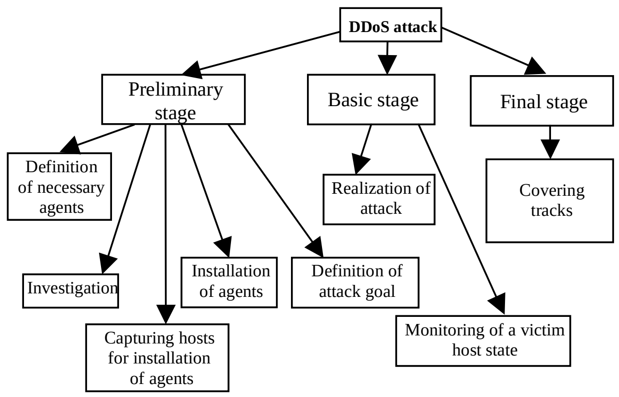

DDoS attack
Preliminary stage
Basic stage
Final stage
Definition of necessary agents
Investigation
Capturing hosts for installation of agents
Installation of agents
Definition of attack goal
Realization of attack
Monitoring of a victim host state
Covering tracks

Fig. 3. Upper level of hierarchy of agent plans

5. HACKERS-AGENTS' PLANS AND ROLE-ASSIGNMENT

Common formal plan of DDoS attacks implemented by team of hackers-agents is specified as three-level structure: (1) *Upper level* specifies intention-based scenarios of the information warrior team in terms of time-ordered sequences of intentions and negotiation acts; (2) *Middle level* specifies intention-based scenarios of each information warrior in terms of an ordered sequences of sub-goals; (3) *Lower level* specifies the information warriors' intentions in terms of sequences of low-level actions (commands).

The upper level part of the agent plans hierarchy is depicted in Fig.3. DDoS attack includes three stages: (1) preliminary, (2) basic and (3) final. The main operations of the preliminary stage are investigation (reconnaissance) and installation of agents. The content of the basic stage is the realization of a DDoS attack by joint actions of agents. Having received as a result of a chain of messages a "victim" address, agents-attackers begin to defeat a chosen host. At this time agents-scouts monitor a victim state. While achieving the success of the attack, agents-scouts inform other agents about the success. In case of irrelevancy of the attack against a host (for example, if the host has been switched off from a network) or impossibility of defeating it, the operation is terminated or a new victim for the DDoS attack is chosen. The portion that includes the upper and middle levels of the hierarchy of agent plans for preliminary and basic stages is presented in Fig.4. As an example, only two types of DDoS attacks are represented – SMURF and Land attacks.

Mathematical models of attacks are specified in terms of a set of interconnected *formal grammars* [1, 3, 5]:

$$M_A=<\{G_i\}, \{Su\}>,$$

where $\{G_i\}$ – the formal grammars, $\{Su\}$ – the "substitution" operations. The sequences of symbols ("strings", "words" – in formal grammar terminology) generated by *each* of such grammars correspond to the sequences of

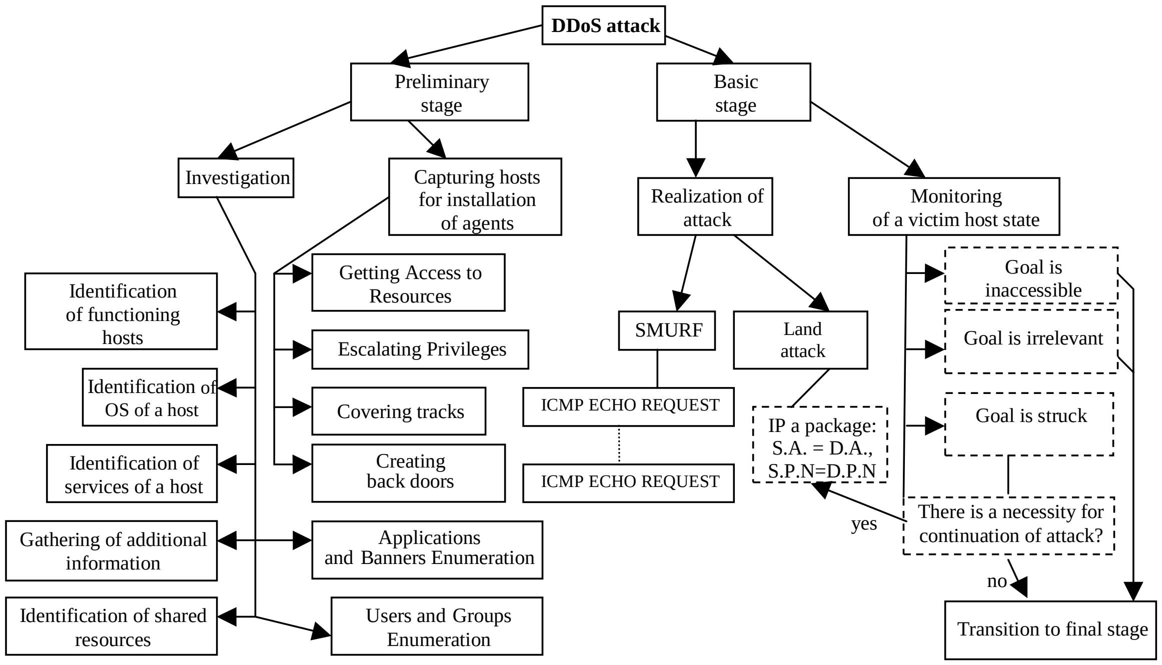

DDoS attack
Preliminary stage
Basic stage
Investigation
Capturing hosts for installation of agents
Realization of attack
Monitoring of a victim host state
Identification of functioning hosts
Getting Access to Resources
Escalating Privileges
Covering tracks
Creating back doors
Identification of OS of a host
Identification of services of a host
Gathering of additional information
Applications and Banners Enumeration
Identification of shared resources
Users and Groups Enumeration
SMURF
Land attack
ICMP ECHO REQUEST
ICMP ECHO REQUEST
IP a package: S.A. = D.A., S.P.N=D.P.N
Goal is inaccessible
Goal is irrelevant
Goal is struck
There is a necessity for continuation of attack?
yes
no
Transition to final stage

Fig. 4. Fragment of the upper and middle levels of hierarchy of agent plans

time ordered intentions and/or actions of the information warrior. At the scenario specification level (it was earlier called *macro-level*) such sequences correspond to the specification of scenarios in terms of the information warrior's intentions and actions.

The formal model of attack scenarios in terms of formal grammars are based on the attacks ontology described above. Note that each node of the ontology that is not "terminal" is mapped to particular grammar, which in turn can generate only admissible sequences realizing this intention in terms of symbols, corresponding to the ontology nodes of the immediately lower level. Depending on the required level of detail, these nodes may be represented by the terminal nodes of the macro- or micro-level. In the former case, the grammar may be used to visualize the information warrior's actions, and in the latter case – for attack simulation in the lowest level terms (if the "terminal" nodes of the micro-level are represented by network packets, system calls or remote invocations).

Every formal grammar is specified by a quintuple [1, 3, 5]:

$$G=<V_N,V_T,S,P,A>,$$

where G is the grammar identifier (name), V_N is the set of non-terminal symbols (that are associated with the upper and the intermediate levels of representation of the steps of an attack scenario), V_T is the set of its terminal symbols (designating the steps of a lower-level attack scenario), $S\in V_N$ is an initial symbol of an attack scenario, P is the set of productions that specify the refinement operations for the attack scenario through the substitution of the symbols of an upper-level node by the symbols of the lower-level nodes, and A is the set of attributes and algorithms of their computation.

The *attribute component* of each grammar serves multiple purposes. Firstly, it specifies *randomized choice of a production* at the current inference step if several productions have the equal left part non-terminals coinciding with the active non-terminal in the current sequence under inference. Secondly, the attribute component is used to check *conditions determining the admissibility of using a production* at the current step of inference. These conditions depend on task specification, configuration of the attacked computer network (host) and its resources and results of the information warrior's previous actions.

Assignment of roles and distribution of plans between agents are carried out as follows: roles of the agents necessary for the given goal are selected, the chosen roles are appointed to agents, and agents of corresponding types are replicated (i.e., cloned).

6. IMLEMENTATION OF HACKERS-AGENTS' TEAMWORK

Algorithmic interpretation of the attack generation specified by formal grammars is implemented by the family of state machines. The basic elements of each state machine are states and transition arcs. States of each state machine are divided into three types: first (initial), intermediate, and final (marker of this state is *End*). The initial and intermediate states are the

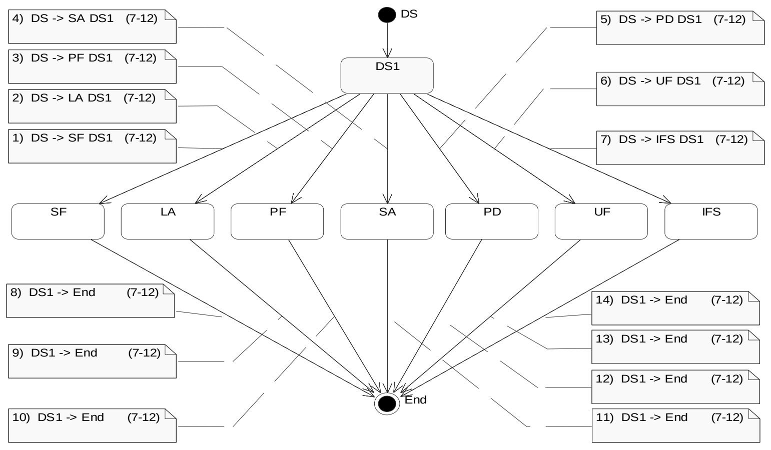


Fig.5. Diagram of the state machine DS (Denial of Service attack)

following: non-terminal, those that initiate the work of the corresponding nested state machines; terminal, those that interact with the host model; and abstract (auxiliary) states. Transition arcs are identified with the productions of grammars, and can be carried out only under certain conditions. Within the state, besides the transition choice depending on the intention and the current transition probability, the following types of actions can be performed:

Entry action (an action performed on entering the state);

Do action (a set of basic actions, including actions of transition to the nested state machine or realizing the host response model);

Exit action (an action performed on exiting).

Example of the state machine DS (Denial-of - Service (DoS) attack) is represented in Fig.5. The main parameters of such a specification of the state machine are given in Table 1.

Table 1. Main parameters of the state machine DS

States	DS1, SF (SYN flood - storm of inquiries on installation of TCP connections), LA (Land attack), PF (Ping flooding - storm of echoes-inquiries on the ICMP protocol), SA (Smurf attack), PD (Ping of Death), UF (UDP flooding), IFS (Storm of inquiries to FTP-server), End
First State	DS1
Terminal states	SF, LA, PF, SA, PD, UF, IFS
Auxiliary states	DS1

In the *state machine diagram*, the first and the final states are denoted as black circles, and the intermediate states – as rectangles with rounded corners. Arrows signify the transitions of the state machine. Dotted lines connect the transition arcs to explanatory texts with the grammar rules. The explanatory texts contain the number and contents of the *grammar production* that corresponds to the transition, numbers of *intentions* for which the rule should be implemented (numbers of intentions are in parentheses), and *conditions* of the transition (if they exist).

7. MODEL OF ATTACKED COMPUTER NETWORK

The attack development depends on the information warrior's "skill", information regarding network characteristics that he/she possesses, some other attributes of the information warrior [5], the security policy of the attacked network, etc.

The information warrior's action has to be generated at runtime in parallel with observation of the reaction of the attacked network. The proposed context-free grammar syntax provides the model with this capability. At each particular step of inference, the grammar generates no more than a single terminal symbol that is interpreted by the computer network model as an information warrior's action.

The network returns the value of the result (i.e., success or ineffectiveness of the action). The model of the attacker receives it and generates the next terminal symbol according to the attack model. This symbol depends on the returned result of the previous step of the attack.

The *model of the attacked computer network* is represented as a quadruple:

$$MA = \langle M_{CN}, \{M_{Hi}\}, M_P, M_{HR} \rangle,$$

where M_{CN} is the model of the computer network structure; $\{M_{Hi}\}$ are the models of the host resources; M_P is the model of computation of the attack success probabilities; M_{HR} is the model of the host reaction on attack. The network structure model is as follows:

$$M_{CN} = \langle A, P, N, C \rangle,$$

where A – the network address; P – a family of protocols used (e.g., TCP/IP, FDDI, ATM, IPX); N – a set $\{CN_i\}$ of sub-networks or a set $\{H_i\}$ of hosts of the network CN; C – a set of connections between the sub-networks (hosts) specified as a mapping matrix. If N establishes a set of sub-networks $\{CN_i\}$, then each sub-network CN_i can in turn be specified by the model M_{CNi}.

Each host H_i is determined as a pair $M_{H_i}= <A, T>$, where A – the host address, T – a host type (e.g., firewall, router).

Models $\{M_{H_i}\}$ of the network host resources serve for representing the host parameters that are important for attack simulation (IP-address, type and version of operating system, users' identifiers, domain names, host access passwords, running applications, etc.)

Success or ineffectiveness of any attack action (corresponding to the terminal level of the attack ontology) is determined by means of the model $M_P=\{R^{SPr}_j\}$ of computation of the attack success probabilities, where R^{SPr}_j is a special rule that determines the action success probability depending on the basic parameters of the host (attack target). The rule R^{SPr}_j includes *IF* and *THEN* parts. The *IF* part contains the action name and precondition (values of attributes constraining the attack applicability). The *THEN* part contains the value of the success probability (*SPr*).

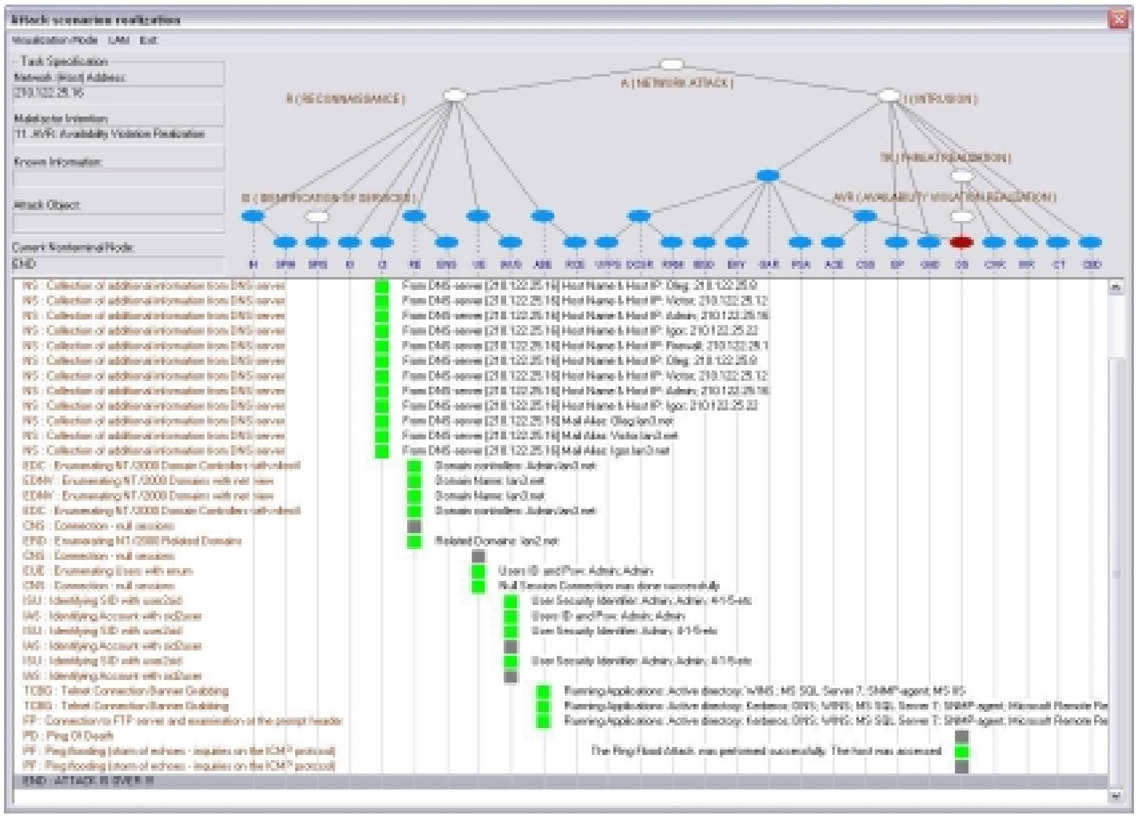
The result of each attack action is determined according to the model M_{HR} of the host reaction. This model is specified as a set of rules of the host reaction: $M_{HR} = \{R^{HR}_j\}$, R^{HR}_j: *Input* $\rightarrow$ *Output* [& *Post-Condition*]; where *Input* – the information warrior's activity, *Output* – the host reaction, *Post-Condition* – a change of the host state, & – logical operation "AND", [] – optional part of the rule.

The Attack Success Parameter is determined by the success probability of the attack that is associated with the host (attack target) depending on the implemented attack type. The values of attack success parameter are *Success* (*S*), and *Ineffectiveness* (*F*). The *Post-Condition* format is as follows: $\{p_1=P_1, p_2=P_2,\dots, p_n=P_n\}$, where p_i – ith parameter of the host which value has changed, P_i – the value of ith parameter.

8. MULTI-AGENT SYSTEM FOR SIMULATION OF DOS ATTACKS

The software prototype of the Attack Simulator tool only models DoS attacks. The tool is currently used for validation of the basic ideas, formal framework and exploration of implementation issues. The tool's architecture was designed as a single agent of multi-agent system (MAS). The engineering of the attack simulator was carried out on the basis of MASDK – Multi-Agent System Development Kit [4].

The MAS agents generated by MASDK have the identical architecture [4]. Differences are reflected in the content of the agents' data and knowledge bases. Each agent interacts with other agents, environment which is perceived, and, possibly, modified by agents, and user communicating with agents through the agent's interface.



Attack scenarios realization
Visualization Mode
LAM
Exit
A (NETWORK ATTACK)
R (RECONNAISSANCE)
I (INTRUSION)
TR (THREAT REALIZATION)
IS (IDENTIFICATION OF SERVICES)
AVR (AVAILABILITY VIOLATION REALIZATION)

```
Task Specification
Network (Host) Address:
210.122.25.16
Malefactor Intention:
11. AVR: Availability Violation Realization
Known Information:

Attack Object:

Current Nonterminal Node:
END
```

```
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Oleg; 218.122.25.9
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Victor; 210.122.25.12
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Admin; 210.122.25.16
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Igor; 210.122.25.22
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Firewall; 210.122.25.1
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Oleg; 218.122.25.9
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Victor; 210.122.25.12
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Admin; 210.122.25.16
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Host Name & Host IP: Igor; 210.122.25.22
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Mail Alias: Oleg.lan3.net
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Mail Alias: Victor.lan3.net
INS : Collection of additional information from DNS-server   From DNS-server [218.122.25.16] Mail Alias: Igor.lan3.net
EDC : Enumerating NT/2000 Domain Controllers with nltest     Domain controller: Admin\lan3.net
EDNV : Enumerating NT/2000 Domains with net view             Domain Name: lan3.net
EDNV : Enumerating NT/2000 Domains with net view             Domain Name: lan3.net
EDC : Enumerating NT/2000 Domain Controllers with nltest     Domain controller: Admin\lan3.net
CNS : Connection - null sessions
ERD : Enumerating NT/2000 Related Domains                    Related Domains: lan2.net
CNS : Connection - null sessions
EUE : Enumerating Users with enum                            Users ID and Psw: Admin; Admin
CNS : Connection - null sessions                             Null Session Connection was done successfully
ISU : Identifying SID with user2sid                          User Security Identifier: Admin; Admin; 4-1-5-etc
IAS : Identifying Account with sid2user                      Users ID and Psw: Admin; Admin
ISU : Identifying SID with user2sid                          User Security Identifier: Admin; 4-1-5-etc
IAS : Identifying Account with sid2user
ISU : Identifying SID with user2sid                          User Security Identifier: Admin; Admin; 4-1-5-etc
IAS : Identifying Account with sid2user
TCBG : Telnet Connection/Banner Grabbing                     Running Applications: Active directory; WINS; MS SQL Server 7; SNMP-agent; MS IIS
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
PD : Ping Of Death
PF : Ping flooding (storm of echoes - inquiries on the ICMP protocol)   The Ping Flood Attack was performed successfully. The host was accessed
PF : Ping flooding (storm of echoes - inquiries on the ICMP protocol)
END : ATTACK IS OVER !!!
```

Fig. 6. Visualization of DoS attack development

As an example of a scenario, fulfilled by the Attack Simulator (as a single DDoS agent), let us consider a screen indicating the generation of DDoS attack closing by Ping flood (Fig.6). The information presented in the figure is divided in four groups: (1) the attack task specification placed in the left top-most part of the screenshot; (2) the attack generation tree visualized in the right hand part of it; (3) the strings of the information warrior's actions placed in the left hand part of the screenshot and below the attack task specification; (4) a tag of success (failure) as green (black) quadrate and data obtained from an attacked host (a host response) depicted on the right hand part of each information warrior's action.

The main objective of the *experiments* with the Attack Simulator is to evaluate the tool's efficiency for different variants of attacks and attacked network configurations. The simulation-based exploration of the approach, formal framework, and software have the following purposes:

(1) *Checking a computer network security policy at stages of conceptual and logical design of network security mechanisms*. This type of checking is performed by simulation of attacks at a macro-level;

(2) *Checking security policy (including vulnerabilities recognition) of a real-life computer network*. This task is performed via simulation of attacks at a micro-level, that is by generating network traffic corresponding to the real activity of information warriors.

These experiments were carried out for various parameters of the attack task specification and an attacked computer network configuration. In addition to information warrior's intention, the influence of the following *input parameters* on attacks efficacy was explored: degree of protection afforded by the network and personal firewall, and attacked host (e.g., how strong is the password, does the host use shared files, printers and other resources, does the host use trusted hosts), and the degree of information warrior's knowledge about a network. To investigate the Attack Simulator capabilities, the following *parameters of attack realization outcome* have been selected: number of terminal-level attack actions, percentage of the information warrior's intentions that are successful, percentage of "effective" network responses on attack actions, percentage of attack actions blockage by firewall, and percentage of "ineffective" results of attack actions (when attack is not successful).

9. CONCLUSION

In the paper a formal paradigm for modeling and simulation of a broad spectrum of DDoS attacks performed by a team of information warriors is proposed.

The paper presents the structure of a team of agents, agent interaction-and-coordination mechanisms, specifications of hierarchies of agent plans, agent role-assignment mechanisms, and the apportionment of plans among agents. The proposed technology for creation of the hackers-agents' team is also described.

The developed approach has been used for simulation-based evaluation of computer network security and analysis of both efficiency and effectiveness of security policy against DDoS attacks.

The Attack Simulator tool now supports only the simulation of a wide spectrum of real-life DoS attacks. It is implemented in Visual C++ 6.0, Java 2 version 1.3.1, KQML, and XML languages. Experiments with the Attack Simulator have been conducted, including the investigation of a wide spectrum of attack scenarios against networks with different structures and security policies.

The further development of the Attack Simulator tool will consist of enlargement of its capabilities in specification of the attack tasks, expansion of the attack classes, implementing more sophisticated attack scenarios, realizing the DDoS attacks simulation, etc.

REFERENCES

1. Aho A.V., Ullman J.D.: The Theory of Parsing, Translation, and Compiling, Vol. 1, 2, Prentice-Hall, Inc. (1972)
2. Cohen P.R., Levesque H.J.: Teamwork. Nous, 25(4) (1991)
3. Fu K.S.: Syntactic Methods in Pattern Recognition, Academic Press, New York (1974)
4. Gorodetski V., Karsayev O., Kotenko I., Khabalov A.: Software Development Kit for Multi-agent Systems Design and Implementation. Lecture Notes in Artificial Intelligence, Vol. 2296, Springer Verlag (2002)
5. Gorodetski V., Kotenko I.: Attacks against Computer Network: Formal Grammar-based Framework and Simulation Tool. Recent Advances in Intrusion Detection. RAID 2002. Zurich, Switzerland. Proceedings. Lecture Notes in Computer Science, Vol. 2516, Springer Verlag (2002)
6. Grosz B., Kraus S.: Collaborative plans for complex group actions. Artificial Intelligence, Vol.86 (1996)
7. Jennings N.: Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. Artificial Intelligence. No.75 (1995)
8. Tambe M.: Towards Flexible Teamwork. Journal of Artificial Intelligence Research, No.7 (1997)
9. Tambe M., Pynadath D.V.: Towards Heterogeneous Agent Teams. Lecture Notes in Artificial Intelligence, Vol.2086 (2001)