

Application-Level & Agent based Simulation for Network Security using NeSSi²

Ms. Surbhi Khare¹ Ritesh Shrivastava²

¹Department of Computer Technology ²Department of Computer Science & Engineering

¹PIET Nagpur, India ²ACET, Nagpur, India

Abstract— NeSSi (Network Security Simulator) is a novel network simulation tool which incorporates a variety of features relevant to network security distinguishing it from general-purpose network simulators. We present NeSSi², the Network Security Simulator, a simulation environment that is based on the service-centric agent platform JIAC. It focuses on network security-related scenarios such as attack analysis and evaluation of counter-measures. We introduce the main NeSSi² concepts and discuss the motivation for realizing them with agent technology. Then, we present the individual components and examples where NeSSi² has been successfully applied. Network or security functionality, but although they provide detailed results, experiments are time consuming and remain complex to setup and maintain. Another approach is to represent the system with the aid of mathematical models and find analytical answers, i.e. logical and quantitative relationships between the entities. Typically, such models also become very complex, in particular for a concurrent system such as IDS. Therefore, simulations are useful for the evaluation of distributed systems and protocols. Depending on the evaluation metrics, the simulations allow the abstraction from irrelevant properties. In addition, hazard scenarios, called “what-if scenarios”, can be constructed which may not be possible in real-world test environments.

Key words: Network Simulation, Discrete-Event Simulation, Packet-Level Simulation, Application-Level Simulation AAMAS Proceedings, Simulation, Demo

I. INTRODUCTION

In contemporary communication infrastructures, IP-based computer networks play a prominent role. The deployment of these networks is progressing at an exponential rate as different kinds of participants such as corporations, public authorities and individuals rely on sophisticated and complex services and communication systems. With regard to information security, this leads to new challenges as large amounts of data, which may hold malicious content such as worms, viruses, or Trojans, are transferred over open networks. Network security measures dealing with these threats can be implemented in the network itself as well as at hosts connected to access routers of the network. The design and development of security solutions such as Intrusion Detection Systems (IDS) is a challenging and complex task. In this process, the evolving system needs to be evaluated continuously. There are several ways to study a system or technology. The most accurate is the analysis of the deployed production system. However, in the case of IDS evaluation, real experiments incorporating attack scenarios cannot be done in an operational environment because the induced risk of failures such as service loss is too high

A. NESSI Architecture

The design of a packet-level discrete-event simulator is a challenging task. In order to handle the inherent complexity,

NeSSi has been structured into three distinct components, the Graphical User Interface, the simulation backend and the result database. Each of these modules may be run on separate machines depending on the computational requirements; furthermore, this modular design facilitates network security researchers using NeSSi to easily ex-change network topologies, scenario definitions and simulation results via remote repositories. In the following sections, we will describe each of these modules in turn and delineate the workflow in NeSSi, beginning with the creation of a network from scratch up to the analysis of the simulation results.

B. Graphical User Interface

The graphical user interface is a Rich Client Platform (RCP) application based on the Eclipse framework [10]. It uses the Standard Widget Toolkit (SWT, [11]), a cross-platform open-source widget toolkit which allows NeSSi to be run on almost all operating systems. As an RCP application, NeSSi is structured into views, editors and perspectives.

A view is a composite widget for displaying data of a certain type to the user; the different views implemented in NeSSi will be described later in this section. Editor windows contain the main data of interest (in the case of NeSSi graphical representations of networks respectively subnets), while the surrounding views display context-related information based on the selection in the editor window. Perspectives on the other hand are used to bundle thematically related views and editors to present the user with an interface to execute the desired task while hiding functionality that is not essential to the current task.

The graphical frontend of NeSSi allows the user to create and edit network topologies, attach runtime information, and schedule them for execution at the simulation backend. On the other hand, finished (or even currently executing, long-running) simulations can be retrieved from the database server and the corresponding simulation results are visualized in the GUI. This constitutes two distinct use cases: pre- and post-simulation visualization. For this reason, the GUI has been divided in two perspectives, a Network Editor perspective and a Network Simulation perspective.

1) Network Editor Perspective

The Network Editor perspective comprises a main network editor window grouped with a variety of different views which provide additional information pertaining to the element currently selected in the network editor. Some of these views appear in both perspectives since they are valuable in the editing as well as the evaluation phase. These are:

Console Here, network status information and logged events are displayed. The desired level of verbosity can be configured.

Routing Table When a network node, i.e. a client, server or router, is selected, the respective routing table is

displayed. For each reachable target machine, it contains the IP addresses of the gateway, the subnet mask and the hop count.

Properties Displays and allows editing a number of properties for the selected element. Among other information, device name and device type are displayed for nodes; for links, bandwidth, latency and maximum transfer unit (MTU) is available.

Statistics Graphical representation of simulation results. The content of this view adapts to the element selected in the editor or the outline.

Apart from these, by reusing existing plugins available for Eclipse, we offer valuable standard functionality in NeSSi such as:

Team Support Users of NeSSi can share networks, scenarios and sessions as well as database resources via remote repositories based on CVS or SVN.

Update Functionality New versions can be downloaded via an update site. The updates are deployed as separate features, allowing the user to decide which components of his NeSSi.

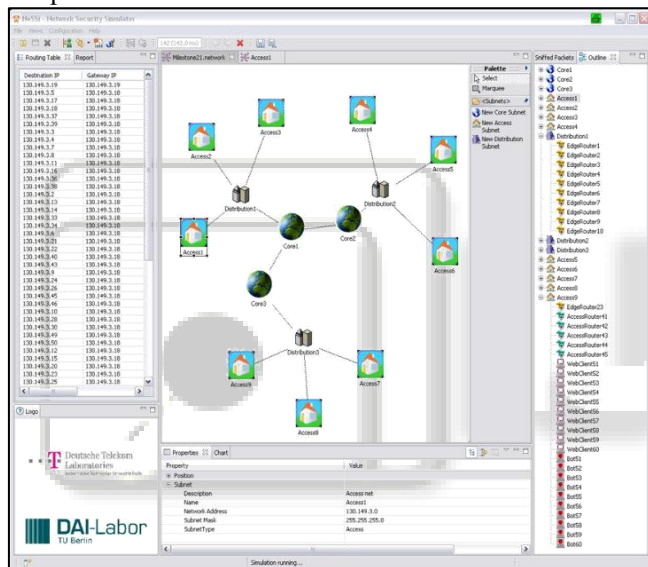


Fig. 1: Graphical user interface of NeSSi - Main Network

II. SOLUTION APPROACH

We introduce NeSSi², an agent-based simulation environment [3], providing telecommunication network simulation capabilities with an extensive support to evaluate security solutions such as IDS. In contrast to other network simulators, like e.g. NS-3 [2], NeSSi² also provides a comprehensive detection API for the integration and evaluation of IDS. In particular, special common attack scenarios can be simulated. Worm-spread scenarios and botnet-based DDoS attacks are only two of the supported example attacks. In addition, customized profiles defining the node behavior can be applied within the simulation.

NeSSi² is built upon the JIAC [1] framework, a service-centric agent-framework. The most recent version, JIAC V², is used in NeSSi². The network entities, i.e. routers, clients, servers, or IDS (nodes in the following) are simulated with the aid of JIAC agents. Dependent on configuration parameters and hardware characteristics, each agent simulates one or more nodes. NeSSi² is benefiting from agent technology in general and JIAC in special through the

service-centric, modular and flexible approach to realizing distributed execution environments. In addition, a common semantic data model enables interoperability of agents executing even different simulation models at the same time.

This semantic model also incorporates the main modeling concepts for the creation and administration of simulations. The first concept and step to setup a simulation is the creation of the network topology. This topology can then be re-used for different scenarios. The scenario is comprised of elementary building blocks for each device in the network, the node profiles. They allow the customization of node behavior to automatically generate traffic, simulate failures or apply network-based defense measures. Every profile consists of applications, representing mechanisms to be executed on an individual node, e.g. an attack, a detection mechanism or an application protocol such as HTTP. The sum of all profiles for a given network is called the scenario. In order to execute it, the length of simulation execution, the number of simulation runs and a recording configuration are configured within a session. As simulations often contain stochastically components such as distribution functions, e.g. the number/timing of HTTP-requests, multiple runs allow for the statistical analysis of mean values and standard deviations.

A. Architecture

NeSSi² has been structured into three distinct components, the graphical frontend, the agent-based simulation back-end and the result database. Each of these modules may be run on separate machines. The modular design facilitates the exchange of network topologies, scenario definitions and simulation results.

The graphical frontend of NeSSi² (c.f. Figure 2) allows to create and edit the necessary components of a network simulation as described in Section 2. On the other hand, finished (or even currently executing, long-running) simulations can be retrieved from the database server and the corresponding simulation results are visualized in the GUI. Accordingly, there exist two different perspectives in the GUI, the Network Editor perspective for the creation of simulations as well as the Network Simulation perspective to investigate simulation results.

In the backend, different agent roles carry out the task of the parallel simulation execution. On each backend, i.e. separate machine, there exists the Simulation Control Agent (SCA) administrating access to the resources of the system as well as the interaction with the GUI. In this way, the SCA interacts with the individual Network Simulation Coordination Agents (NCAs). For every executed simulation run, an NCA is invoked which starts a number of Device Management Agents (DMAs). The number of DMAs depends either on particular user configurations, e.g. “one agent for every node”, “x agents in total”, or follows the computational power of the backend system, i.e. “one agent per CPU core”.

Finally, the result database stores simulation results according to the configuration specified during the creation process of the simulation in the GUI. For every simulation run, the agents record selected events and traffic data to a specified log4j³ appender which handles the output according to the recorder configuration. By default, the results – such as attack-related events – as well as the model are

recorded to a database which allows for replaying the simulation. In addition, the recorded data can be used for evaluation purposes.

To generate attack data and evaluate detection algorithms implemented by students. In a recent industry research project, NeSSi² has been incorporated in an agent-based Decision Support System to forecast upcoming link congestions in the access network of a big German DSL-provider. NeSSi² is Open Source since January of 2009 and has been downloaded more than 6000 times.

III. SUCCESSFUL UTILIZATION

NeSSi² has demonstrated its value in recent research and was employed as a simulation environment for various security-related approaches. In this regard, NeSSi² was used to investigate optimal placement strategies for IDS, analyze worm propagation strategies and evaluate the benefit of collaborative IDS. NeSSi² has also been used in lectures.

<http://logging.apache.org/log4j/>

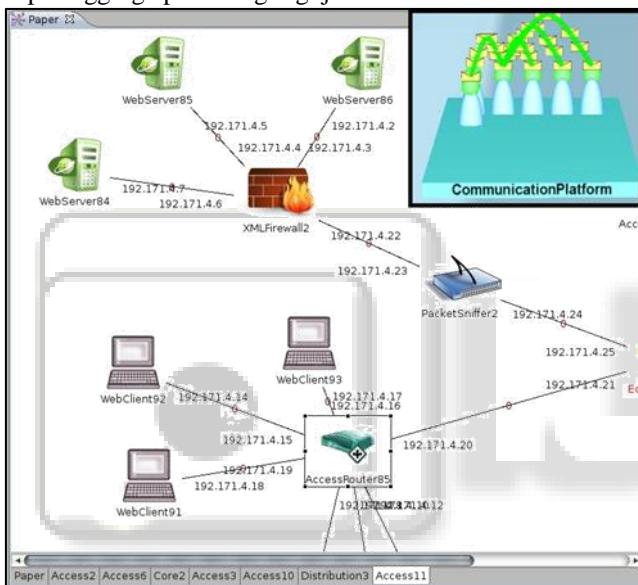


Fig. 2: GUI and Backend illustrated: The GUI enables the creation and administration of arbitrary networks and node configurations. After the setup process is finished, an agent-based simulation back-end (“Communication Platform”) executes the simulation and the results are stored in a database.

IV. CONCLUSION

We have presented NeSSi², a network simulation environment with a focus on security-related scenarios. The simulation backend is based on agent technology benefiting from the service-centric, modular and flexible design of the JIAC framework to load balance the complexity of the simulation runs. NeSSi² incorporates a semantic data model to reflect simulations of arbitrary networks and individual node configurations and has been used in various (industry) research projects as well as lectures.

REFERENCES

[1] Ulrik Brandes. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks*, 30(2):136–145, May 2008

[2] B. Hirsch, T. Konnerth, and A. Heßler. Merging agents and services — the JIAC agent platform. In *Multi-Agent Programming: Languages, Tools and Applications*, pages 159–185. Springer, 2009.

[3] ns3 project. NS-3 network simulator. <http://www.nsnam.org/docs/architecture.pdf>, last accessed on 02/24/2011.

[4] S. Schmidt, R. Bye, J. Chinnow, K. Bsufka, A. Camtepe, and S. Albayrak. Application-level simulation for network security. *Simulation*, 86(5-6):311–330, May/June 2010.