

Risk Analysis of Infrastructure Security and Resilience Practicum

I.S. Skarga-Bandurova, A.Y. Velykzhanin, Y.P. Kovalenko
Edited by V.S. Kharchenko

Analysis of Security Auditing Tools

Network Security Risk Analysis using Attack
Graphs Approach

Cyber Threats Risks Modeling with Bayesian
Networks

Assessing Risks and Opportunities in
Enterprise Architecture



Risk Analysis of Infrastructure Security and Resilience. Practicum



PRACTICUM

RISK ANALYSIS OF INFRASTRUCTURE SECURITY AND RESILIENCE

2017



Co-funded by the
Tempus Programme
of the European Union

**Ministry of Education and Science of Ukraine
Volodymyr Dahl East Ukrainian National University
National Aerospace University “KhAI”**

I.S. Skarga-Bandurova, A.Y. Velykzhanin, Y.P. Kovalenko

Risk Analysis of Infrastructure Security and Resilience

Practicum

Edited by V.S. Kharchenko

Project
TEMPUS SEREIN “543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR
Modernization of Postgraduate Studies on Security and Resilience
for Human and Industry Related Domains

2017

UDK 004.7.056.5:005.334](076)=111
C42

Reviewers:

Dr Peter Popov, City University London, United Kingdom

Dr Olexandr Siora, Research and Production Corporation Radiy, Ukraine

C42 Skarga-Bandurova I.S., Velykzhanin A.Y., Kovalenko Y.P.

Risk Analysis of Infrastructure Security and Resilience: Practicum / Kharchenko V.S. (ed.) – Ministry of Education and Science of Ukraine, Volodymyr Dahl East Ukrainian National University, National Aerospace University “KhAI”, 2017. – 112 p.

ISBN 978-617-7361-23-6

Training support package for MSc course “Risk Analysis of Infrastructure (System of Systems, SoS) Security and Resilience”, was designed for master students within the framework TEMPUS project “Modernization of Postgraduate Studies on Security and Resilience for Human and Industry Related Domains” co-founded by the Tempus Programme of the Europe Union. Project Number: 543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR.

The course is devoted to methodology and practice for cyber threats evaluation, security risk analysis and resilience of large-scale critical infrastructures. The program of the practical part of the course covers a wide range of issues from studying security auditing tools, network security analysis, risk modeling, cyber risk assessment in enterprise architectures, till testing critical functions for software management systems.

Training support package includes a course outline, ad hoc teaching materials, borrowed open-source software and native software.

The book is intended for MSc studying cyber security, computer and program engineering, and computer science working in the field of SoS security evaluation and protection. It could be useful for lecturers and professors who conduct classes on corresponding courses.

Fig.: 43. Ref.: 29. Tables: 12.

Approved by Academic Council of National Aerospace University “Kharkiv Aviation Institute” (record № 6, February 22, 2017).

УДК 004.7.056.5:005.334](076)=111

ISBN 978-617-7361-23-6

© Skarga-Bandurova I.S., Velykzhanin A.Y., Y.P. Kovalenko, Kharchenko V.S.

© Volodymyr Dahl East Ukrainian National University

© National Aerospace University “KhAI”

This work is subject to copyright. All rights are reserved by the authors, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms, or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

ACRONYMS AND ABBREVIATIONS

BN	–	Bayesian Network
CVE	–	Common Vulnerabilities and Exposures
CVSS	–	Common Vulnerability Scoring System
DNS	–	Domain Name System
DMZ	–	Demilitarized Zone
EAAT	–	Enterprise Architecture Analysis Tool
EDB	–	Extensional Database
fw	–	Firewall
ICT	–	Information and Communication Technology
IDB	–	Intensional Database
JDK	–	Java Development Kit
MulVAL	–	Multi-host, Multi-stage Vulnerability Analysis Language
NFS	–	Network File System
Nmap	–	Network Mapper
NVD	–	National Vulnerability Database
OS	–	Operating System
P2AMF	–	Probabilistic Architecture Modeling Framework
SoS	–	System of Systems
VM	–	Virtual Machine

INTRODUCTION

Training support package for MSc course “Risk Analysis of Infrastructure Security and Resilience” was designed for master students within the framework TEMPUS project “Modernization of Postgraduate Studies on Security and Resilience for Human and Industry Related Domains” co-founded by the Tempus Programme of the Europe Union. Project Number: 543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR.

The main aim of the course is to create a knowledge base for multidisciplinary research on critical infrastructure risk management and develop a security curriculum of suitable and recognized industry and academic experts as well. Training support package includes a course outline, ad hoc teaching materials, borrowed open-source software and native software. The labs we proposed for the course were on the following topics, with particular tools written in parentheses: (1) Analysis of security auditing tools (nmap, Nessus, Nexpose, etc.). (2) Network security analysis using attack graphs approach (MulVAL and as alternative freeware tool to study attack graph approach TVA, Attack Graph Toolkit, NetSPA, etc. can be used). (3) Cyber threats risks modeling with Bayesian networks (AgenaRisk and as alternative tool to study risks modeling techniques with Bayesian networks variety tools such GeNIe && SMILE, SamIam, etc. are also available). (4) Assessing risks and opportunities in enterprise architecture (EAAT Object Modeler and CySeMoL class model. Systems Aris, ETIS, QualiWare, System Architect, etc. can be used alternatively). (5) Risk-based and functional security testing the program source code (native software AutoTestDFB).

The structure of the program contains three complementary educational aspects. First, the core educational component combines an intensive training with the small groups on labs. Second, this course provides the students with the opportunity to enhance their skills by wide involvement industrial practice and case study. Case study enables students to develop realistic solutions to the industrial security problems and to understand crucial nature of complex analysis both specifically and generally. Finally, we try to encourage the students to use their knowledge and ambitions to draw information security concepts into their research activity.

By the end of semester, the successful student should be able to:

- employ the functional representation of industrial system interconnections for structural and resilience analysis in the framework of resilience and risk assessment;

- select appropriate risk analysis method for different tasks of cyber security;
- prepare test scenarios designed to validate the performance of security systems and protective force in detecting, interdicting and countering threats;
- utilize different software tools to compile and analyze data to identify trends regarding security assurance activities and create reports to support customer requirements and/or compliance related requirements;
- perform risk analyzes and process analyzes related to security data manipulation and management;
- analyze ICT risks to ensure their security and resilience.

As acquired professional competencies we expect a) effective analytical and problem-solving skills to contribute to creative solutions to complex cyber security problems, b) gaining experience in working under limited direction within scope of the assignment and using independent judgment in choosing methods, techniques, software, and evaluation criteria and c) ability to interact effectively with peers and customers.

Training support package prepared by the professor of Computer Engineering department of Volodymyr Dahl East Ukrainian National University, D.Sc. Skarga-Bandurova I.S., Ph.D. student Kovalenko Y.P. (laboratory work 5), and master student Velykzhanin A.Y. who fulfilled all practical tasks in good faith and adjusted laboratory works 1-4 (parts (1.3), (2.3), (3.3), (4.3)). General editing was performed by Professor, Head of Computer systems and networks department of Kharkiv National Aerospace University “KhAI”, D.Sc. Kharchenko V.S.

Much of the work was inspired and supported by our colleagues on TEMPUS project. We are very grateful for this and thank for their contribution to this book. The authors are indebted to Dr. P. Popov for his longstanding fruitful cooperation in the study of risk analysis, articles, presentations, and his contribution to understanding critical infrastructure (system of systems, SoS) interdependencies. We thanks Dr A. Siora for useful recommendations and discussion of results considering long-time experience of RPC Radiy.

This project has been funded with support from the European Commission. This publication (communication) reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Laboratory work 1

ANALYSIS OF SECURITY AUDITING TOOLS

Goal and objectives: This laboratory work is an introduction to one of the most well known port scanning tools, *nmap*. We'll discover and scan hosts on the virtual network. The results of the host discovery section of this lab will be used for vulnerability assessment.

Learning objectives:

- study security auditing technique;
- study network inventory, managing service upgrade schedules, and monitoring host methodology.

Practical tasks:

- acquire practical skills in working with security scanning tools;
- gather information on targets, including reverse DNS names, operating system guesses, device types, and MAC addresses.

Exploring tasks:

- discover hosts on the closed network;
- investigate how to restrict the application scans to specific sets of port numbers.

Setting up

In preparation for laboratory work it is necessary:

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1,2];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources: *nmap* (<https://nmap.org/>) security scanner (as alternative port scanning tools you can use OVAL, Nessus, or Nexpose).

1.1 Synopsis

In this laboratory work you will learn one of the most common utilities for exploring a network within port scans. Specifically you will use the network port scanning tool *nmap*. While you explored this tool using the Linux operating system, the same tool is available for Windows operating systems.

1.2 Brief theoretical information

Vulnerability scans are used to collect information about the vulnerabilities in the hosts. These scans perform repeatedly on the individual vulnerabilities on each of the deployed hosts in a controlled environment to ensure a reliable output.

Nmap (“Network Mapper”) is an open source tool for network exploration and security auditing. It was designed to rapidly scan large networks, although it works fine against single hosts. *Nmap* uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics. While *nmap* is commonly used for security audits, many systems and network administrators find it useful for routine tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime.

The output from *nmap* is a list of scanned targets, with supplemental information on each depending on the options used. Key among that information is the “interesting ports table”. That table lists the port number and protocol, service name, and state. The state is either `open`, `filtered`, `closed`, or `unfiltered`. `Open` means that an application on the target machine is listening for connections/packets on that port. `Filtered` means that a firewall, filter, or other network obstacle is blocking the port so that *nmap* cannot tell whether it is `open` or `closed`. `Closed` ports have no application listening on them, though they could open up at any time. Ports are classified as `unfiltered` when they are responsive to *nmap*'s probes, but *nmap* cannot determine whether they are `open` or `closed`. *Nmap* reports the state combinations `open | filtered` and `closed | filtered` when it cannot determine which of the two states describe a port. The port table may also include software version details when version detection has been requested. When an IP protocol scan is requested (`-sO`), *nmap* provides information on supported IP protocols rather than listening ports.

In addition to the interesting ports table, *nmap* can provide further information on targets, including reverse DNS names, operating system guesses, device types, and MAC addresses [1]. Note that *nmap* also has a GUI front end named *zenmap*.

1.3 Execution order and discovery questions

1. Set up your lab environment according to the specifications below; draw and annotate your testing environment.

- a. Open <https://www.virtualbox.org/wiki/Downloads>.
- b. Download VirtualBox 5.x.x for system that you are using, install VB.
- c. Download preconfigured images from <http://virtualboxes.org/images/> or you can manually install OS that you want to test.
- d. Open VirtualBox, and choose import Appliance

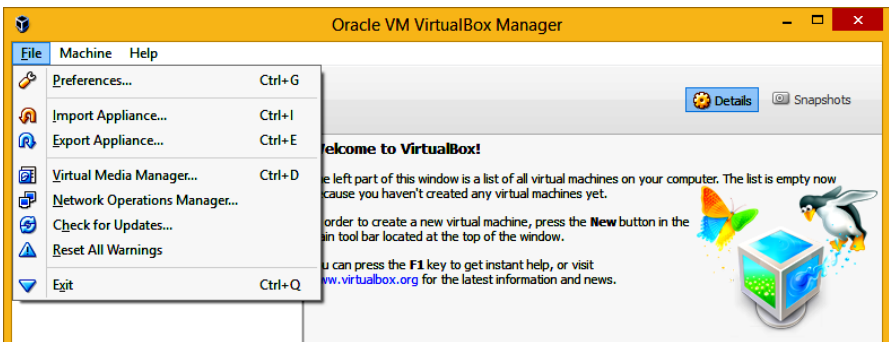


Figure 1.1 – Initial window of Oracle VM VirtualBox Manager

- e. Choose a virtual appliance image

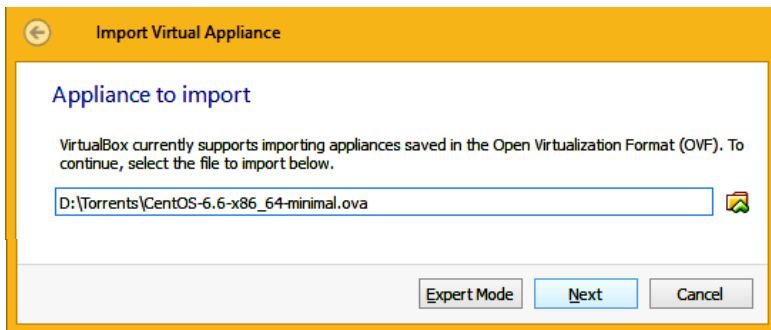


Figure 1.2 – Window for selecting file to import

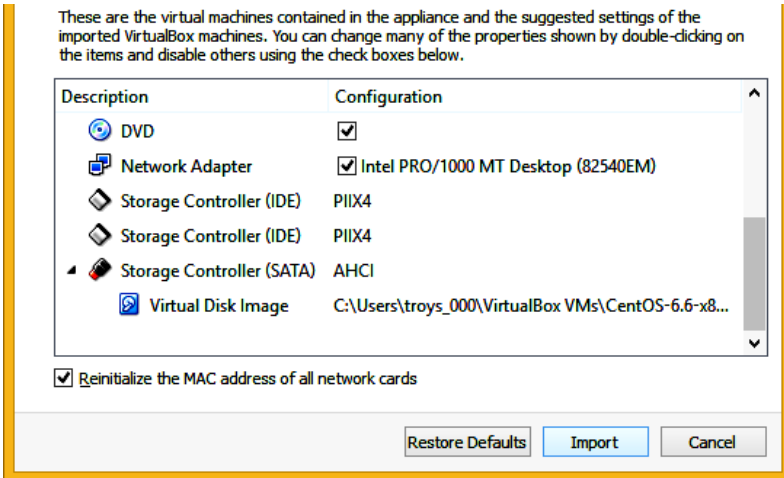


Figure 1.3 – Appliance settings menu

Then change network configuration.

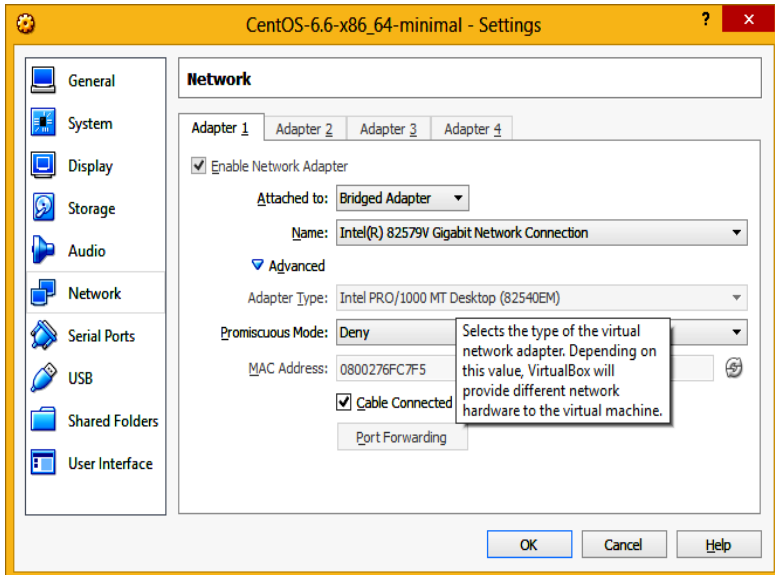


Figure 1.4 – Tab for changing network configuration

And click on the start button.

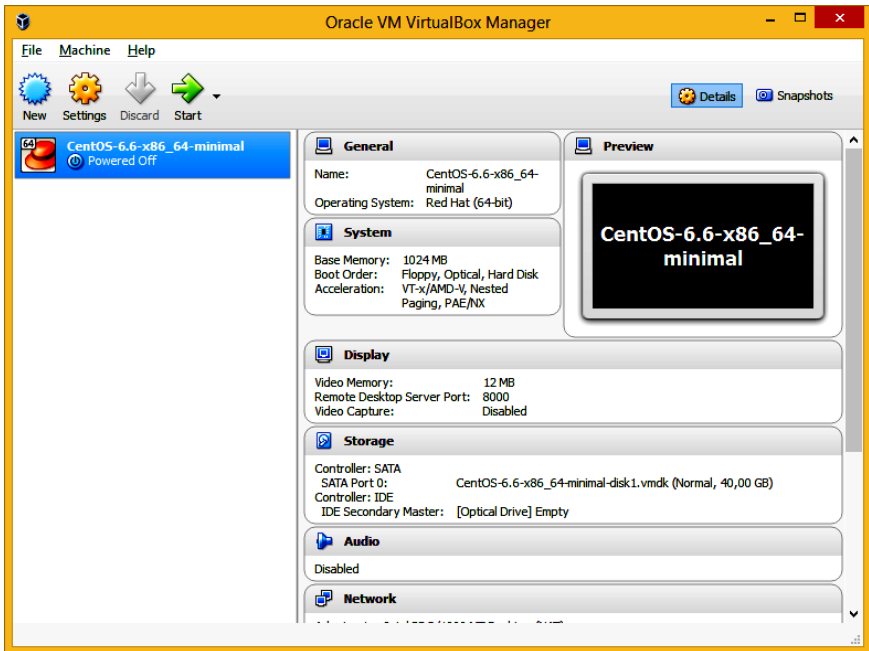


Figure 1.5 – VM characteristics

f. If `eth0` adapter does not working you need to fix it, before start login with defined login/password. You can find root password in description of virtual machine.

g. Type following command in command line: `#ifconfig` (see fig. 1.6).

If `eth0` does not exist you should conduct the following operations (vi commands see: <http://www.lagmonster.org/docs/vi.html>):

```
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
BOOTPROTO=dhcp
HWADDR=00:60:63:01:F4:84
ONBOOT=yes
TYPE=Ethernet
```



```
[root@localhost ~]#
Display all 942 possibilities? (y or n)
[root@localhost ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:6F:C7:F5
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe6f:c7f5/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:20 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:650 (650.0 b)  TX bytes:1732 (1.6 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128  Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
```

And change to

```
DEVICE=eth0
BOOTPROTO=dhcp
#HWADDR=00:60:63:01:F4:84
ONBOOT=yes
TYPE=Ethernet
```

then call

```
# vi /etc/udev/rules.d/70-persistent-net.rules
# PCI device 0x1022:0x2000 (pcnet32) (custom name provided by
external tool)
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
ATTR{address}=="", ATTR{type}=="1", KERNEL=="eth*", NAME="eth0"

PCI device 0x1022:0x2000 (pcnet32)
#SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
ATTR{address}=="", ATTR{type}=="1", KERNEL=="eth*", NAME="eth1"
```

and change it to


```
# PCI device 0x1022:0x2000 (pcnet32) (custom name provided by
external tool)
#SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
ATTR{address}=="", ATTR{type}=="1", KERNEL=="eth*", NAME="eth0"

#PCI device 0x1022:0x2000 (pcnet32)
#SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
ATTR{address}=="", ATTR{type}=="1", KERNEL=="eth*", NAME="eth1"
```

After that do `# reboot` and after restarting the system check wheter `eth0` exists.

h. For better *nmap* results you should create 2-5 virtual machines with various OS.

i. To know the IP of your VM use `ifconfig` command or similar commands to OS that you use in our case 192.168.1.1.

j. Test your network configuration by pinging the all VM.

2. Download *nmap* from <https://nmap.org/download.html> for you OS

3. Open *nmap* terminal window (or *Zenmap* GUI) and observe the command options available for *nmap*.

4. Leave the *nmap* main page open for reference.

a. Specifically enter the following scans:

```
nmap -n -sn 192.168.1.1/24
```

Record the results.

b. Answer the questions: Why is the `-n` option used? What happens if you rerun this command without the `-n` option? (Try it). What does the `/24` represent?

c. Use the same *nmap* command to start a ping scan on the entire /16 Common Network. Record at least one of your colleagues machines before terminating this scan, Ctrl-C. Define, how many hosts would this scan look for. Justify your answer.

5. Conduct an IP protocol ping (switch `-PO` / `-PS` / `-PU`) on the Common Network hosts. Note that for this scan "*nmap* needs to read raw responses off the wire"; you must use `sudo` to have sufficient privilege.

a. Detect how many TCP ports are open on each?

b. Define are there any UDP ports open on any machine?

6. Conduct an IP protocol ping on yourself and calculate how many ports are open. *Nmap* is often capable of determining the operating system of a scanned host. {Hint: read the OS Detection section of the man pages and again note that you will need to use `sudo` to have sufficient privilege}.

7. *Nmap* is also often able to determine the version number of various services running as software applications {Hint: read the Service/Version Detection section of the man pages}. Investigate how to restrict the application scans to specific sets of port numbers otherwise your scans may take a long time to complete.

1.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

1.5 Test questions

1. For what purpose *nmap* is used?
2. List main command options available for *nmap*.
3. What do the following switches *nmap* (sn, PO, PS, PU, sO, sV, O) do?
4. How many open TCP ports there were detected?
5. Are the results of IP protocol scan (-sO) different than that attained with the IP protocol ping? Explain.

1.6 Recommended literature

1. Nmap manual - <http://nmap.org/book/man.html>
2. Network Security 1 and 2 Companion Guide - Vulnerabilities, Threats and Attacks – Available from <http://www.lovemytool.com/files/vulnerabilities-threats-and-attacks-chapter-one-7.pdf> (accessed September 12, 2015).

Laboratory work 2

NETWORK SECURITY RISK ANALYSIS USING ATTACK GRAPHS

Goal and objectives: In this laboratory work we will focus on the analyzing the attack graph generation and visualization technology for network security analysis and risk assessment.

Learning objectives:

- study a methodology for security risk analysis that is based on the model of attack graphs and the Common Vulnerability Scoring System (CVSS).

Practical tasks:

- acquire practical skills in working with attack graph tools;
- to perform network security risk analysis.

Exploring tasks:

- discover attack graph generation and visualization techniques;
- investigate how individual steps can potentially enable an attacker to gain privileges deep within the network.
- analyze all attack paths through a network, providing a probabilistic metric of the overall system risk

Setting up

In preparation for laboratory work it is necessary:

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1-5];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

MulVAL (<http://www.arguslab.org/mulval.html>) (as alternative tool to study attack graph approach you can use open source tools like *TVA*, *Attack Graph Toolkit*, *NetSPA*, etc. or commercial tools, for example *Cauldron*, *FireMon*, *Skybox View*, etc.).

2.1 Synopsis

Network vulnerability can be analyzed automatically by attack graph. Attack graph tools can generate attack paths in network and show users the network vulnerabilities analyzing process for network security risk analysis. There are some problems such as state space explosion, the high complexity of algorithms, being difficult to demonstrate graphically, and so on, for attack graph generation and visualization techniques.

Attack graphs illustrate the cumulative effect of attack steps, showing how individual steps can potentially enable an attacker to gain privileges deep within the network. CVSS is a risk measurement system that gives the likelihood that a single attack step is successfully executed. In this laboratory work we will study a methodology to measure the overall system risk by combining the attack graph structure with CVSS.

Automated attack-graph tools show how individual vulnerabilities in a system can be composed to create meta-vulnerabilities or how individual sensor alerts can be joined to describe a multi-stage scenario attacks.

2.2 Brief theoretical information

A canonical representation of attack graphs was proposed in [6] is illustrated in fig. 2.1. In this representation, the nodes of the graph represent the state of the entire computer network at a given time. Here, a state characterizes both the state of the computer network itself (including hosts, software, configurations, authorized users and vulnerabilities) and the identities that the attacker owns (i.e., user accounts under the attacker's control).

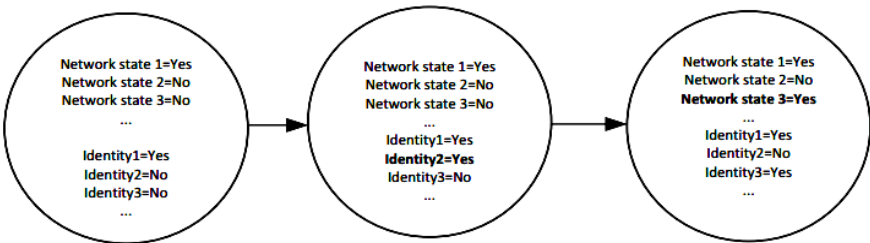


Figure 2.1 – A canonical representation of attack graphs, with transitions between states in the network and identities controlled by the attacker, adapted from Heberlein et al. [7].

With this state space as a starting point, an attack graph shows how the attacker can move from one state to another using the vulnerabilities and controlled identities present in the network.

An attacker can thus either change the state of the network itself (e.g., by reconfiguring a firewall) or obtain a new identity (e.g., by stealing a user's password or session). Heberlein et al. in [7] also describe some features of the analyses used in attack graph approaches. These features in [6] were summarized as follows:

Monotonic or non-monotonic exploits. To simplify the analysis and offer scalability, it is commonly assumed that exploits are monotonic, i.e., that once the attacker has obtained some identity, that identity is never lost. *MulVAL* assumes that exploits are monotonic.

Single path or all paths. For simplicity, the analysis is sometimes limited to generating a single attack path, i.e., the analysis stops as soon as one possible path to the specified state is found. *MulVAL* finds all paths.

Forward or backward chaining. In a forward chaining technique, the analysis starts with an assumption of the current state and assesses which states are reachable. In a backward chaining technique, an end-state of interest is defined, and the analysis assesses which states may lead to that end-state. *MulVAL* uses backward chaining.

2.2.1 General information about MulVAL

MulVAL stands for "Multi-host, Multi-stage Vulnerability Analysis Language". It is a research tool for security practitioners and system administrators to better manage the configuration of an enterprise network such that the security risks are appropriately controlled. It uses vulnerability scanners (OVAL, Nessus, etc.) and reasons about attack paths from the input, which includes both scanning results and network reachability information.

MulVAL requires information on vulnerabilities, subnets, users and host access control lists as input. For each vulnerability, the following fields are specified: the hostname, Common

Vulnerabilities Enumeration ID, program (target service or application), range list (if it is remotely or locally exploitable), type of loss (textual), severity (high/medium/low), access control requirement (high/medium/low), service and port. Subnets are described by the hosts they include, and the host access control lists describe by which protocol and on which port entities (i.e., hosts or subnets) they are allowed to communicate. *MulVAL* also makes it possible to include users and their accounts in the analysis. These users can be labeled as incompetent, meaning they are susceptible to social engineering attacks. In this test, machines with active user agents were labeled as incompetent to reflect that the users opened all email attachments and clicked links indiscriminately.

MulVAL uses Datalog rules to specify its input to Prolog. Descriptions of the systems used in this laboratory work will be provided to *MulVAL* by extracting data from:

- a) the system configuration files used to instantiate the computer networks in the cyber range and
- b) data from vulnerability scans (Laboratory work 1).

The current version of *MulVAL* will be tested on the Linux operating systems.

2.2.1.1 Formal model of reasoning

MulVAL adopts Datalog as the language to model network elements and their interactions. We first review some terminologies [2].

Datalog overview

Syntactically, Datalog is a subset of Prolog with limited forms of clauses. A *literal*, $p(t_1, \dots, t_k)$ is a predicate applied to its arguments, each of which is either a constant or a variable. In the formalism of Prolog, a variable is an identifier that starts with an upper-case letter. A constant is one that starts with a lower-case letter. Let L_0, L_1, \dots, L_n be literals, a Horn clause in Datalog has the form:

$$L_0 :- L_1, \dots, L_n$$

Semantically, it means if L_1, \dots, L_n are true then L_0 is also true. The left-hand side is called the *head* and the right-hand side is called the *body*. A clause with an empty body is called a *fact*. A clause with a nonempty body is called a *rule*. A significant difference between Datalog and Prolog is that Datalog has a pure declarative semantics. The order of clauses in a Datalog program is irrelevant to its logical meaning and evaluation result. Whereas in Prolog such order is important and affects the result of evaluation, due to the depth-first search strategy and side-effect operators like “cut”.

Datalog is often used in deductive databases. In such settings, data tuples in the database are represented as Datalog facts, and the deductive engine is implemented as a Datalog program that runs on the inputs from the database. The Datalog facts representing the original database are called the extensional database (EDB), and the Datalog facts computed by the deductive engine are called the intensional database (IDB). The complexity of computing whether a literal is implied by a Datalog program from EDB input (i.e. whether the literal is in IDB) is polynomial in the size of the EDB.

Datalog has also been used as a security language for expressing access control policies. The declarative semantics of Datalog makes specifying concepts such as delegation straightforward. The efficiency of Datalog and existing off-the-shelf Datalog evaluation engines make such languages readily usable in practice.

Analysis framework

The *MulVAL* core analysis framework is shown in figure 2.2. An analysis database is a collection of Datalog facts that represent the status of the network and the advisory information about software vulnerabilities. The interaction rules are Datalog clauses that specify how different pieces of a network can interact and affect security. These are reasoning rules that can simulate what an attacker can do in the network, given the configuration information in the analysis database. The security policy specifies the ultimate

property a system administrator wants to keep for the network. In *MulVAL*, the policy is simple Datalog tuples that list legal data accesses by principals.

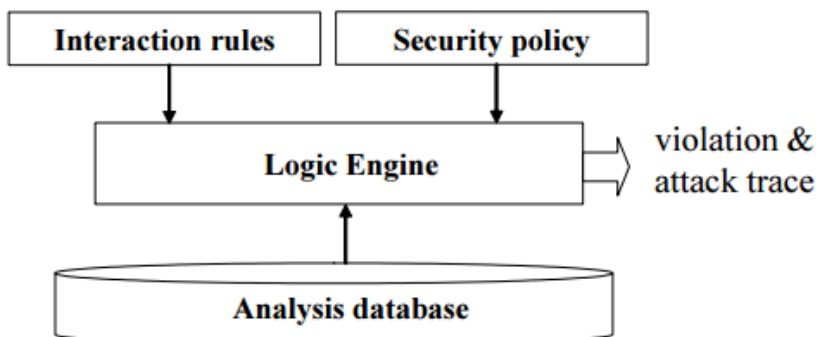


Figure 2.2 – Analysis framework

Interaction rules

MulVAL interaction rules specify the semantics of: different kinds of vulnerabilities and their exploits, normal software behaviors that affect security, and multihop network access. Many of those rules are operating-system specific. The rules discussed here apply to the Unix-family operating systems. Currently there are about 20 rules in *MulVAL*. The *MulVAL* rules are carefully designed so that information about specific vulnerabilities is factored out. The interaction rules characterize general attack methodologies (such as “remote exploit of a buffer overrun bug,” or “Trojan Horse client program”), not specific vulnerabilities. Thus the rules do not need to be changed frequently, even though new vulnerabilities are reported frequently. The rules are also independent of specific configurations of a particular network setting and thus can be applied across different sites.

Types of constants

In Datalog, a term is either a constant or a variable. Datalog is an untyped language, so a predicate can be applied to arbitrary

terms. However, to make a Datalog sentence meaningful, the arguments to a predicate should take value from certain domains. This section lists the types used in the Datalog interaction rules.

1. **Host.** A host is represented as a symbolic name, such as `webServer` and `fileServer`. In the real implementation, it is represented as an IP-address range.

2. **Protocol.** A transport or application layer protocol, such as `tcp`, `udp`, and `rpc`.

3. **Port.** A number differentiating different services within the same protocol.

4. **Principal.** A symbolic name representing a certain group of people, such as `employee` and `attacker`.

5. **Data.** A symbolic name representing an abstract notion of a data item, such as `webPages` and `projectPlan`.

6. **String.** Single-quoted strings are used to represent file-system paths, vulnerability identification numbers, etc.

7. **Exploit range.** Either `localExploit` or `remoteExploit`.

8. **Exploit consequence.** One of four possibilities: `confidentiality`, `integrity`, `privilegeEscalation`, and `dos`.

9. **Program.** The name of a program on a system, such as `httpd`.

10. **User/Group** The name of a user or group on a system.

11. **Access:** `read`, `write`, or `exec`.

Vulnerability rules

A vulnerability is an unintended behavior in a software system that can be utilized by an attacker to compromise the security of a host. Following are the predicates involved in rules about vulnerabilities. The arguments of the predicates are presented as variables, although in a specific rule they can either be a variable or a constant.

vuExists(Host, Program, ExploitRange, ExploitConsequence) is a derived predicate specifying that a vulnerability exists in the Program on a Host, and it has specific ExploitRange and ExploitConsequence. This is a derived predicate. Program is the

full path of the executable that contains the security bug. **ExploitRange** is either local or remote, indicating if the bug is locally exploitable or remotely exploitable. Two common values for **ExploitConsequence** are **privilegeEscalation**, meaning a successful exploit would enable an attacker to execute arbitrary code, and **dos**, meaning the attacker can crash the program (denial of service).

vulExists(Host, ID, Program) is a primitive predicate specifying that a vulnerability with identification ID exists in the Program on the Host.

vulProperty(ID, ExploitRange, ExploitConsequence) is a primitive predicate that specifies the exploitable range and consequence of the vulnerability with ID.

bugHyp(Host, Program, Range, Consequence) is a dynamic predicate that introduces a hypothetical bug in a Program on the Host which has ExploitRange and ExploitConsequence. More details about using dynamic predicates to conduct hypothetical analysis is discussed in [2].

dependsOn(Host, Program, Library) is a primitive predicate specifying that a Program on a Host depends on a Library, where the type of Library is also “Program”.

Exploit rules

We first introduce several predicates that are used in the exploit rules.

execCode(P,H,UserPriv) is a derived predicate specifying that principal P can execute arbitrary code with privilege UserPriv on machine H.

netAccess(P, Src, Dst, Protocol, Port) is a derived predicate specifying that principal P can send packets from machine **Src** to **Port** on machine **Dst** through **Protocol**.

networkService(H, Prog, Protocol, Port, User) is a primitive predicate specifying that a service program Prog is running on host **H** as user **User**. It is listening on port **Port** of protocol **Protocol**. For example, **networkService**(webServer, httpd, tcp, 80, apache)

means on machine webServer, a network service program httpd is running as user apache and listening on port 80 of the tcp protocol.

setuidProgram(H, Prog) is a primitive predicate specifying that Prog is a setuid program on host **H**. The executable file is owned by **User**.

clientProgram(H, Prog) is a primitive predicate specifying that Prog is a client program that when executed, may open a connection to a server over the network.

malicious(P) is a primitive predicate specifying that principal P would attack the network system to gain illegal privilege.

incompetent(P) is a primitive predicate specifying that principal P is not careful in using computers and its behavior may be utilized by a malicious attacker.

File access

The following predicates are used in computing the file access a principal can have on a Unix machine.

accessFile(P, H, Access, Path) is a derived predicate specifying that principal P can Access the files specified by Path on machine H. Access can be either read,write, or exec.

localFileProtection(H, User, Access, Path) is a derived predicate specifying that the User on machine H can have the specified Access to the file Path.

fileAttr(H, Path, R1,W1,X1,R2,W2,X2,R3,W3,X3) specifies the UNIX file attribute bits. For example, if on machine **workStation** the file /home/projectPlan.pdf's attribute is rw-r-----, the corresponding predicate would be **fileAttr** (workStation, '/home/projectPlan.pdf', r,w,0,r,0,0,0,0,0). Note that r and w are interpreted as 1 for the corresponding access bits.

2.2.2 Modeling for network topology analysis

The packet flow in the network affects an attacker's ability to launch attacks. Packet flow is controlled by firewalls, routers, switches, and other aspects of network topology. *MuVAL* uses an

abstract model, host access control list (HACL), to describe the topology of a network

2.2.2.1 Host Access Control List

A host access control list specifies all accesses between hosts that are allowed by the network. It consists of a collection of Datalog tuples of the following form:

```
hacl(Source, Destination, Protocol, DestPort).
```

It means machine Source can reach DestPort on machine Destination through Protocol. HACL is an abstraction of the ultimate effects of the physical topology, firewall rules, the configuration settings of routers and switches, and so on. It is compatible with the high-level specification used in many automatic network management tools. Those tools can be leveraged to provide this information.

2.2.2.2 Multihop host access

Predicate netAccess(P, Src, Dst, Protocol, Port) specifies that principal P can send network packets from machine Src to Port on host Dst through Protocol. Following are the rules deriving the predicate.

```
netAccess(P, H1, H2, Protocol, Port) :- execCode(P, H1, _User),  
hacl(H1, H2, Protocol, Port).
```

If a principal P has local access on machine H1 as some User and the network allows H1 to access H2 through Protocol and Port, then the principal can access host H2 through the protocol and port. This rule allows for reasoning about multihop attacks, where an attacker first gains access on one machine inside a network and launches further attacks from there.

2.2.2.3 Policy specification

The security policy is the only piece of information that a local administrator needs to provide. In *MulVAL*, a security policy

specifies which principal can access what data. Each principal and piece of data is given a symbolic name, which is mapped to a concrete entity by the binding information. Each policy statement is of the form *allow(Principal, Access, Data)*. The arguments can be either constants or variables.

Following is an example policy:

```
allow(_Everyone, read, webPages).
allow(employee, _Access, projectPlan).
allow(sysAdmin, _Access, Data).
```

`_Everyone` and `_Access` are anonymous variables. The policy says anybody can read `webPages`; `employee` can have arbitrary access to `projectPlan`; and `sysadmin` can have arbitrary access to arbitrary data. Anything not explicitly allowed is prohibited. The policy language presented in this section is simple and easy to be specified correctly. However, the *MulVAL* reasoning system can handle more complex policies as well. For example, in *MulVAL* one can use general Prolog as the policy language.

2.2.2.4 Binding information

The principal and data items mentioned in the *MulVAL* policy are just symbolic names. They are mapped to concrete entities by principal binding and data binding. Principal binding maps a principal symbol to its user accounts on network hosts, or a network zone from which the principal operates. The format of the binding information is

```
hasAccount(Principal, Host, Account), or
located(Principal, Zone),
```

where `Principal` is the symbolic name for a principal, `Account` is the name of a user account on `Host` that the principal can access, and `Zone` is the network zone a principal may operate from. Examples:

```
hasAccount(employee, workStation, ralph).
```



```
hasAccount(employee, workStation, james).  
hasAccount(sysAdmin, webServer, root).  
located(attacker, internet).
```

The account associated with a user does not necessarily correspond to a concrete account on the machine. It may stand for a group of accounts that have the same level of privilege. For example,

```
hasAccount(employee, workStation, employeeAccount).  
hasAccount(sysAdmin, webServer, root).  
located(attacker, internet).
```

Here `employeeAccount` represents any ordinary user accounts on the system. The principal binding may also contain information describing user behaviors. These are the malicious and incompetent predicates mentioned before. Example:

```
incompetent(employee).  
malicious(attacker).
```

Data binding maps a data symbol to its physical location in the network, typically a file path on a machine. The format of the binding is

```
dataBind(Data, Host, Path).
```

Path could be a directory, in which case the `dataBindDir` tuple will mean “all files under the directory are bound to data symbol Data”:

```
dataBindDir(Data, Host, DirPath).
```

Example of data bindings:

```
dataBind(projectPlan, workstation, '/home/projectPlan.txt').  
dataBindDir(webPages, webServer, '/www').
```

2.3 An example of *MulVAL* code for multi-host attack

In this section we set an example how to apply the *MulVAL* analysis engine to detect potential attack paths due to

vulnerabilities (both real and hypothetical) on the machines. Figure 2.3 shows a small network. We first, describe the various Datalog-tuple inputs to the analysis engine.

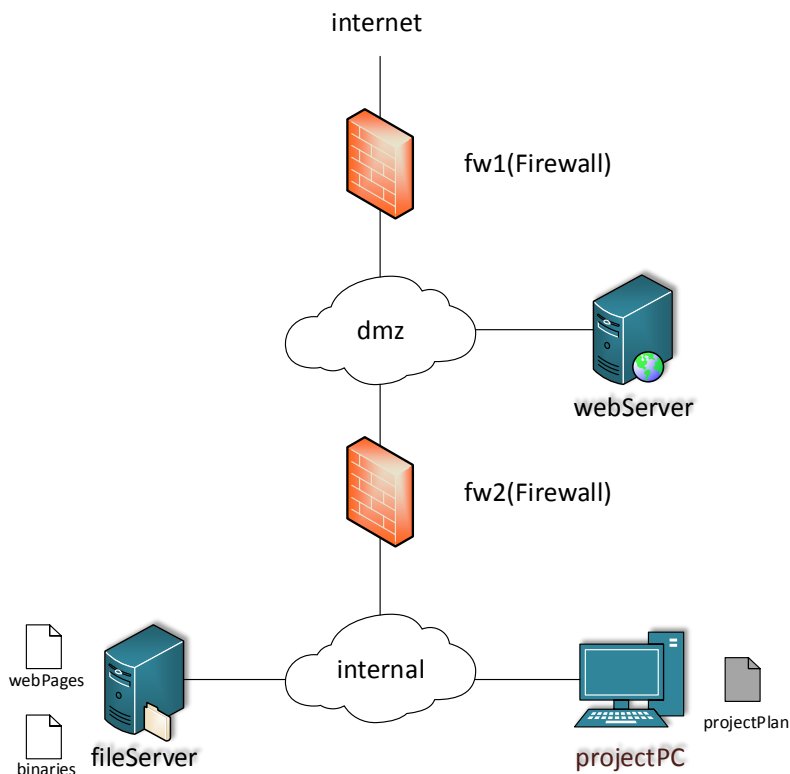


Figure 2.3 – Example of network

2.3.1 Network topology description

There are three zones (internet, dmz and internal) separated by two firewalls (fw1 and fw2). The administrators manage the webserver, the **projectPC** and the **fileServer**. The users have access to the public server workstation, which they use for their computing needs.

The host access control list for this network:


```
hacl(internet, webServer, tcp, 80).
hacl(webServer, fileServer, rpc, 100003).
hacl(webServer, fileServer, rpc, 100005).
hacl(fileServer, AnyHost, AnyProtocol, AnyPort).
hacl(projectPC, AnyHost, AnyProtocol, AnyPort).
hacl(H, H, AnyProtocol, AnyPort).
```

2.3.2 Vulnerability information definition

There are two vulnerabilities reported by the OVAL scanner, on the machine `fileServer` and `webServer` respectively. The corresponding NVD database entries describing the exploitable range and consequences of these two bugs are also shown above.

```
vulExists(fileServer, 'CVE-2003-0252', mountd).
vulExists(webServer, 'CVE-2002-0392', httpd).
vulProperty('CVE-2003-0252', remoteExploit, privEscalation).
vulProperty('CVE-2002-0392', remoteExploit, privEscalation).
```

2.3.3 Machine configuration

Following are relevant Datalog tuples describing machine configurations output by the MulVAL scanners.

```
/* configuration information of fileServer */
networkServiceInfo(fileServer, mountd, rpc, 100005, root).
nfsExportInfo(fileServer, '/export', read, workStation).
nfsExportInfo(fileServer, '/export', write, workStation).
nfsExportInfo(fileServer, '/export', read, webServer).
nfsExportInfo(fileServer, '/export', write, webServer).
/* configuration information of webServer */
networkServiceInfo(webServer, httpd, tcp, 80, apache).
nfsMounted(webServer, '/share', fileServer, '/export', read).
nfsMounted(webServer, '/share', fileServer, '/export', write).
/* configuration information of workStation */
nfsMounted(workStation, '/share', fileServer, '/export', read).
nfsMounted(workStation, '/share', fileServer, '/export',
write).
```

The `fileServer` serves files for the `webServer` and the `workStation` through the NFS protocol. There are actually many machines represented by `workStation`. They are managed by the administrators and run the same software configuration.

To avoid the hassle of installing each application on each of the machines separately, the administrators maintain a collection of application binaries under `/export` on `fileServer` so that any change like recompilation of an application program needs to be done only once.

These binaries are exported through NFS to the `workStation`. The directory `/export` is also exported to `webServer` since the web pages are also stored on the file server.

2.3.4 Data binding

There are two kinds of data are mentioned by the security policy: `webPages`, which is stored on the file server, and `projectPlan`, which is stored on the individual workstations.

```
dataBind(webPages, fileServer, '/export').
dataBind(projectPlan, workStation, '/home').
```

2.3.5 Principals

The principal `sysAdmin` manages the machines with user name `root`. Since all other users in the corporation are treated equally for the purpose of this example, we model them as one principal `employee`. `employee` uses the `workStation` with user name `userAccount`. For this organization, the primary worry is a remote attacker launching an attack from outside the network. The attackers are modeled by a single principal `attacker` who is located in internet. The Datalog tuples for principal bindings are:

```
hasAccount(employee, workStation, employeeAccount).
hasAccount(sysAdmin, webServer, root).
hasAccount(sysAdmin, fileServer, root).
hasAccount(sysAdmin, workStation, root).
attackerLocated(internet).
attackGoal(execCode( _, _)).
malicious(attacker).
```

2.3.6 Set up of security policy

The administrators need to ensure that the confidentiality and integrity of users' files, specifically the data `projectPlan`, will not be compromised by an attacker. Thus the policy is

```
allow(Anyone, read, webPages).
allow(employee, Acc, projectPlan).
allow(sysAdmin, Acc, AnyData).
```

2.3.7 Graph generating

Using the code from the example, we can generate a graph. The input file has the form:


```
attackerLocated(internet).
attackGoal(execCode(webServer,_)).
hacl(internet, webServer, tcp, 80).
hacl(webServer, fileServer, rpc, 100003).
hacl(webServer, fileServer, rpc, 100005).
hacl(fileServer, AnyHost, AnyProtocol, AnyPort).
hacl(workStation, AnyHost, AnyProtocol, AnyPort).
hacl(H, H, AnyProtocol, AnyPort).
networkServiceInfo(fileServer, mountd, rpc, 100005, root).
nfsExportInfo(fileServer, '/export', read, workStation).
nfsExportInfo(fileServer, '/export', write, workStation).
nfsExportInfo(fileServer, '/export', read, webServer).
nfsExportInfo(fileServer, '/export', write, webServer).
dataBind(webPages, fileServer, '/export').
allow(Anyone, read, webPages).
hasAccount(sysAdmin, fileServer, root).
vulExists(fileServer, 'CAN-2003-0252', mountd).
vulProperty('CAN-2003-0252', remoteExploit, privEscalation).
networkServiceInfo(webServer, httpd, tcp, 80, apache).
nfsMounted(webServer, '/share', fileServer, '/export', read).
nfsMounted(webServer, '/share', fileServer, '/export', write).
accessFile(attacker, fileServer, write, '/export').
netAccess(attacker, webServer, tcp, 80).
hasAccount(sysAdmin, webServer, root).
allow(sysAdmin, Acc, AnyData).
vulExists(webServer, 'CAN-2002-0392', httpd).
vulProperty('CAN-2002-0392', remoteExploit, privEscalation).
execCode(webServer, apache).
malicious(attacker).
nfsMounted(workStation, '/share', fileServer, '/export', read).
nfsMounted(workStation, '/share', fileServer, '/export',
write).
accessFile(attacker, workStation, read, '/home').
accessFile(attacker, workStation, write, '/share').
accessFile(workStation, root, read, '/home').
dataBind(projectPlan, workStation, '/home').
hasAccount(employee, workStation).
hasAccount(sysAdmin, workStation, root).
allow(employee, Acc, projectPlan).
allow(sysAdmin, Acc, AnyData).
policyViolation(attacker, read, projectplan).
execCode(workStation, root).
```

Prescribes the environment variables:

```
export PATH=$PATH:/to/xsb/bin
export PATH=$PATH:/to/mulval
cd /to/mulval/utils/
```

Generate a graph with the command:

```
./graph_gen.sh /to/input.P -v -p
```

Example:

```
root@SkyNet:~/fiware-cybercaptor/mulval/utils# ./graph_gen.sh /root/in.P -v -p
Goal:
execCode( h331, h332)
Producing attack graph through GraphViz
If successfully produced, the attack graph should be in AttackGraph.pdf
```

After the computation of the attack paths a PDF file is generated. Using this file you can analyze the possible path of attackers.

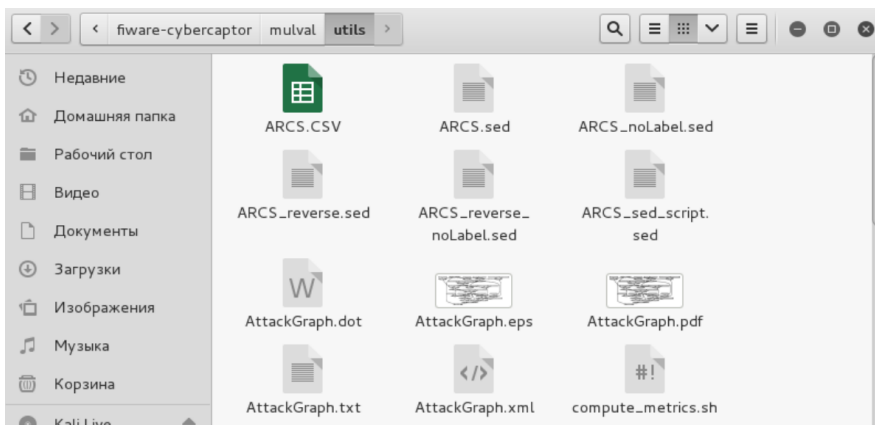


Figure 2.4 – Destination documents

2.4 Execution order and discovery questions

1. First, you need to install Linux to Oracle VM Virtual Box as you did it in Laboratory work 1.
2. Download *MulVAL* tool. For this purpose use the instructions from Appendix A.
3. Upload your data with description of targeted system in *MulVAL*. Descriptions of the systems used in this laboratory work will be provided to *MulVAL* by extracting data from a) the system configuration files used to instantiate the computer networks in the cyber range and b) data from vulnerability scans (Laboratory work 1).
4. Draw attack graph according to your variant (see exercises) and description given in the assignments below.

5. Analyse the possible path of attacker. If an attacker penetrates host X in your network, how deeply into your network can he penetrate? Can attack graph be of any use to system administrator?

2.5 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, procedures
- (R): narrate (like a story), code for your assignment, figure of attack graph with marked the attacker path;
- (D): answers on discovery questions, explanation of results, conclusion / summary

2.6 Test questions

1. What is attack graph?
2. What attack graph shows?
3. How to perform threat risk analysis using attack graphs?
4. What is the main difference between attack trees and attack graphs?
5. Why do we need specialized software to use attack graphs?

2.7 Recommended literature

1. Ou X. Quantitative Security Risk Assessment of Enterprise Networks / X. Ou, A. Singhal. – Springer Briefs in Computer Science, 2012. – 28 p.

2. Ou X. A logic-programming approach to network security analysis / X. Ou // A Dissertation presented to the faculty of Princeton University in candidacy for the degree of Doctor of Philosophy - <ftp://ftp.cs.princeton.edu/techreports/2005/735.pdf> (accessed September 12, 2015).

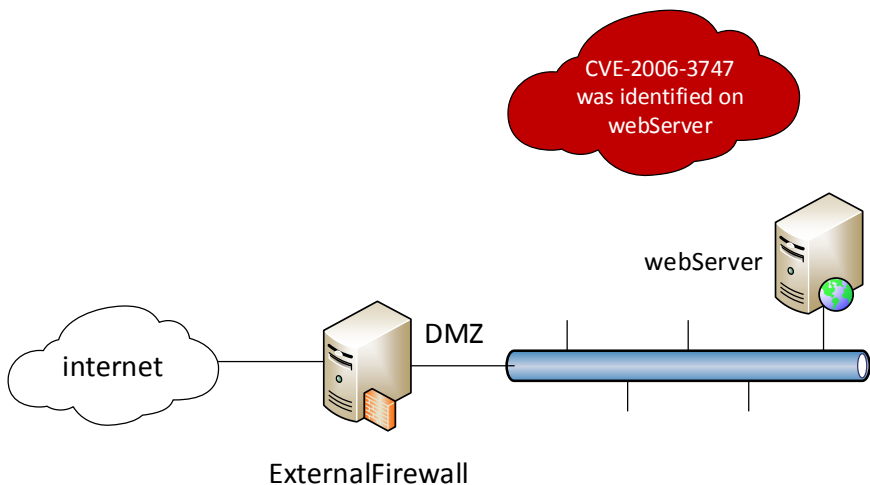
3. Singau A. Security Risk Analysis of Enterprise Networks Using Probabilistic Attack Graphs / A. Singau, X. Ou // National Institute of Standards and Technology Interagency Rep. 7788. – 23 p. (August 2011) <http://csrc.nist.gov/publications/nistir/ir7788/NISTIR-7788.pdf> (accessed September 12, 2015).

4. Liu C. Using Attack Graphs in Forensic Examinations / C. Liu, A. Singal, D. Wijesekera // Seventh Int. Conf. on Availability, Reliability and Security (ARES), 2012. – pp. 596-603. - <http://csrc.nist.gov/staff/Singhal/using-attack-graphs-forensic-examinations.pdf> (accessed September 12, 2015).
5. MulVAL: A logic-based, data-driven enterprise security analyzer - <http://people.cis.ksu.edu/~xou/argus/software/mulval/readme.html>
6. Sommestad T. An empirical test of the accuracy of an attack graph analysis tool / T. Sommestad, F. Sandström // Information & Computer Security, Vol. 23 Iss: 5
7. Heberlein T. A Taxonomy for Comparing Attack-Graph Approaches / T. Heberlein, M. Bishop, E. Ceesay, M. Danforth, C. Senthilkumar and T. Stallard // 2004 - <http://www.netsq.com/documents/attackgraphpaper.pdf> (accessed October 16, 2015).
8. Sandström F. A test of attack graph-based evaluation of IT-security / F. Sandström // Master's Thesis in Computing Science, 2014 – 32 p. - <http://www8.cs.umu.se/education/examina/Rapporter/FredrikSandstrom.pdf> (accessed September 12, 2015).

Assignments to laboratory work 2

Laboratory exercise 1

There is a firewall controlling network access from the Internet to the DMZ subnet of an enterprise network. The Demilitarized Zone (DMZ) is typically used to place publicly accessible servers, in this case the web server. The firewall protects the host in DMZ and only allows external access to ports necessary for the service. In this example, Internet is allowed to access the web server through TCP port 80, the standard HTTP protocol and port.



Suppose a vulnerability scan is performed on the web server, and a vulnerability is identified. The CVE ID of the discovered vulnerability is CVE-2006-3747. Using this ID as a key, one can query the National Vulnerability Database (NVD) and obtain a number of important properties of the vulnerability.

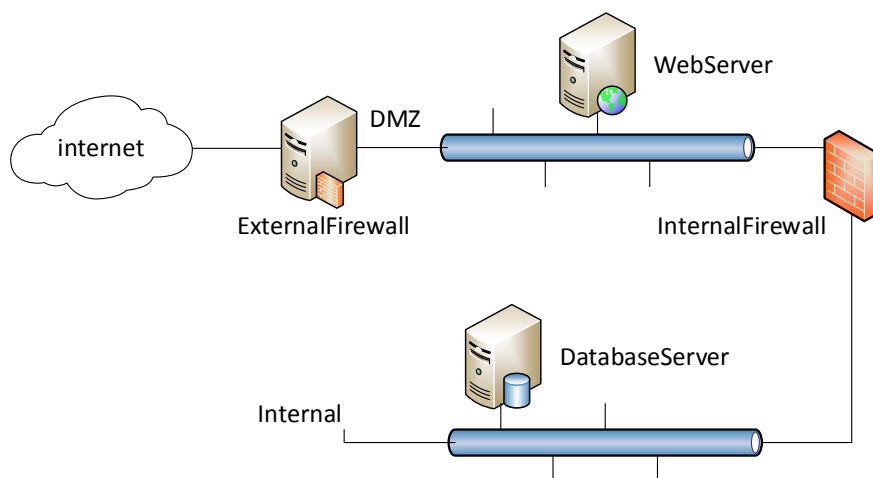
Laboratory exercise 2

The Demilitarized Zone (DMZ) is typically used to place publicly accessible servers, in this case the web server. The firewall protects the host in DMZ and only allows external access to ports necessary for the

service. In this example, Internet is allowed to access the web server through TCP port 80, the standard HTTP protocol and port.

Suppose a vulnerability scan is performed on the web server, and a vulnerability is identified. The CVE ID of the discovered vulnerability is CVE-2006-3747. Using this ID as a key, one can query the National Vulnerability Database (NVD) and obtain a number of important properties of the vulnerability.

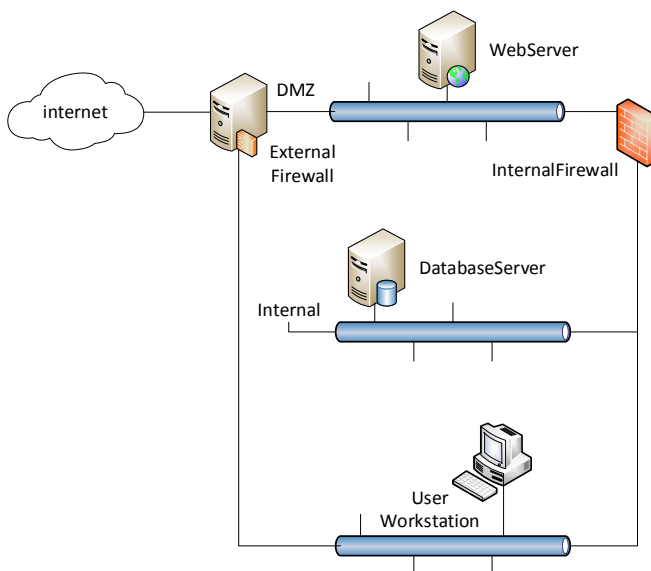
The access to the Internal subnet is mediated by an internal firewall. Only the web server can access the database server, which also has a remote vulnerability in the MySQL DB service (CVE-2009-2446).



Laboratory exercise 3

The external firewall controls network access from the Internet to the enterprise network, and the internal firewall controls the access to the database server that can be accessed by the webservice and workstations. The webserver hosts a webpage at port 8080 using the Tomcat 7 server that provides access to Internet users. The eventual attack we wish to execute is to gain access to database tables as an Internet user. We do so by launching a SQL injection attack that exploits the following java servlet code that does not sanitize input values:

```
(theResult = theStatement.executeQuery( "select * from profiles  
where name='Alice' AND password='"+passWord+"'");).
```



Suppose an internal user uses a workstation that runs Windows XP SP3 operating system with IE6, which has vulnerability (CVE-2009-1918) that enables executing any code on this machine. As an external attacker we use social engineering to trick Alice to visit a malicious web page to control Alice's workstation.

The Demilitarized Zone (DMZ) is typically used to place publicly accessible servers, in this case the web server. The firewall protects the host in DMZ and only allows external access to ports necessary for the service. In this example, Internet is allowed to access the web server through TCP port 80, the standard HTTP protocol and port.

Suppose a vulnerability scan is performed on the web server, and a vulnerability is identified. The CVE ID of the discovered vulnerability is CVE-2006-3747. Using this ID as a key, one can query the National Vulnerability Database (NVD) and obtain a number of important properties of the vulnerability.

The access to the Internal subnet is mediated by an internal firewall. Only the web server can access the database server, which also has a remote vulnerability in the MySQL DB service (CVE-2009-2446).

Laboratory work 3

CYBER THREATS RISKS MODELING WITH BAYESIAN NETWORKS

Goal and objectives: In this laboratory work, we explore a Bayesian Network (BN) based model to incorporate relevant factors, such as the availability of exploit codes, into attack graph-based security metrics. More specifically, we first interpret an attack graph as a special BN; then we combine individual base scores of CVSS using their causal relationships; finally, we integrate the scores of CVSS to derive the final measurement of cyber threats risks.

Learning objectives:

- study a framework for using BN-based techniques to determine the combined effect of vulnerabilities in a networked information system.
- study theoretical foundation for cyber threats risk modeling and continuously measuring system security.

Practical tasks:

- acquire practical skills measuring network security by integrating attack graphs with CVSS scores provided by NVD in BN model;
- to assessment the security risks provided by different system configurations.

Exploring tasks:

- investigate how existing security metrics which generally focused on measuring individual vulnerabilities can be considered jointly and analyze their combined effects;
- explore the causal relationships between vulnerabilities encoded in an attack graph to model the overall security of a network using BN-based approach.

Setting up

In preparation for laboratory work it is necessary:

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1-3];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources: *AgenaRisk* (<http://www.agenarisk.com/>)

As alternative tool to study risks modeling techniques with Bayesian networks you can use open source tools like SamIam (<http://reasoning.cs.ucla.edu/samiam>), GeNIe && SMILE (<https://dslpitt.org/genie/>), etc. To selection appropriate tool refer to Appendix C.

3.1 Synopsis

Risk is the result of a Threat penetrating a Vulnerability impacting a particular asset. This puts “risk” and “consequences” on a common plane. A successful solution to system security should meet following criteria. First, it should model various aspects of the vulnerability. For this purpose CVSS metrics can be useful. Second, it is also desirable that all patterns can be discovered and used for reasoning about future security scores based on past incidents or observations.

In view of the initial assumption, there are two inputs to our model, namely, attack graph and CVSS scores [3]. First, we assume the attack graph of a given network can be obtained using existing tools. Second, we assume the CVSS scores of vulnerabilities in the given attack graph can be obtained from existing vulnerability databases, such as the National Vulnerability Database (NVD) [4].

3.2 Brief theoretical information

Bayesian networks (BNs) is one of the key computer technology for dealing with probabilities. BNs, also known as belief networks (or Bayes nets for short), belong to the family of probabilistic graphical models. These graphical structures are used to represent knowledge about an uncertain domain.

Modeling by BN is one the most usual methodology for solving the lack of information issue or cyber security issue. Another useful property BNs is that they offer a compact means to encode the entire range of conditional relationships, which is particularly suitable for representing security metrics based on attack graphs.

More information about BN in cyber security domains you can find in [1, 2, 5, 6, 8, 9].

3.2.1 How to use BN to representing cyber security probabilistic metrics

Let us consider an attack graph G as a directed graph $G(E \cup C, R_r \cup R_i)$ where E is a set of exploits, C a set of conditions, and $R_r \subseteq C \times E$ and $R_i \subseteq E \times C$ are two relations. We regard each exploit as a binary variable that can take discrete values of T (True), which signifies the exploit has been successfully performed by the attacker, or F (False) indicating the converse. Given any exploit $e \in E$, and its corresponding score BS , we assign conditional probabilities as follows [6]:

$$P(e = T | \forall c \in R_r(e) c = T) = BS/10 \quad 1)$$

BNs offer a compact means to encode the entire range of conditional relationships, which is particularly suitable for representing security metrics based on attack graphs.

The vertices of the BN represent exploits derived from the attack graph. Each vertex is annotated with a probability assigned according to Equation 1. The conditional probability tables can then be developed to encode the probability values for each vertex and its conditional dependencies. Such a BN-based model allows propagating probabilities of an attacker reaching each condition. In particular, we are interested in the goal state (the final conditions), which can be used as an indicator about the overall security of the network.

More formally, given an attack graph $G(E \cup C, R_r \cup R_i)$, we represent the attack graph using a Bayesian network which is a pair $B = (G, Q)$ where G is the directed graph corresponding to the attack graph but with a different semantics, that is, the vertices represent the binary variables of the system and the edges represent the conditional relationships among the variables. Q is the set of parameters that quantify the BN such as the conditional distribution values for each variable (vertex). The joint distribution for a Bayesian network is represented in the standard way as (the notations are self-explanatory):

$$P(X_1 \dots X_n) = \prod_{i=1}^n P(X_i | \text{parents}(X_i)) \quad 1)$$

The unique aspect of this BN representation is the following. In an attack graph, the causal relationships between exploits can be disjunctive or conjunctive based on how they are related through conditions [5]. Such relationships are represented in our BN representation using conditional probabilities of 0 or 1. More specifically, we say:

- a disjunctive relationship exists between any exploits e_1, e_2, \dots, e_n with respect to e_{n+1} when $e_j R_i c$ holds for all $j = 1, 2, \dots, n$ and some condition c , and $c R_i e_{n+1}$ is true. In such a case, the probability assignment based on Equation 1 will satisfy $P(e_{n+1} = T|X) = 1$ for all X that has $e_j = T$ hold for at least one $j \in [1, n]$.

- a conjunctive relationship exists between exploits e_1, e_2, \dots, e_n with respect to e_{n+1} when $e_j R_i c_j$ and $c_j R_i e_{n+1}$ both hold for all $j = 1, 2, \dots, n$ and some conditions c_j 's.

In such a case, we have $P(e_{n+1} = T|X) = 0$ whenever X has $e_j = F$ hold for at least one $j \in [1, n]$.

For example, Figure 3.1 shows a BN with three exploits A, B, and C in which an attacker can achieve the goal state by following one of either two paths (for simplicity, we shall omit conditions from now on). The probabilities are converted from BS scores (by dividing them by 10). Using Equation 1, we can construct the conditional probability tables for each vertex as shown on the right-hand side of Figure 3.1. From the conditional probability tables, we can observe that C is true as long as at least one of A and B is true. This indicates a disjunctive relationship between A and B with respect to C.

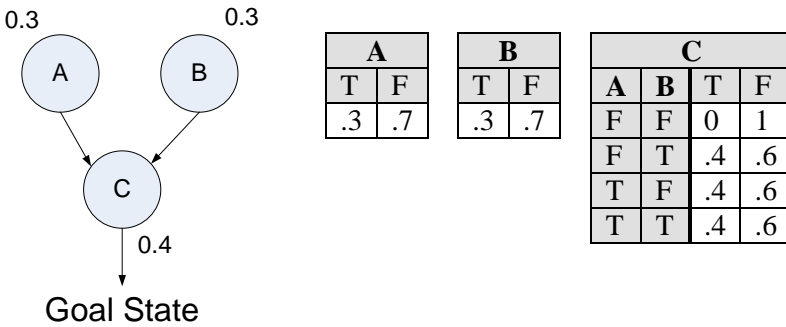


Figure 3.1 – Representing attack graph as BNs

The conditional probability tables allow us to calculate the joint probability function for any exploit or condition in the given network. In this case, we are interested in the probability that $C = T$ (that is, vulnerability C has been successfully exploited). This can be calculated as $P(C=T) = \sum_{A,B \in \{T,F\}} P(C=T, A, B) = 0.204$. As an example application, this calculation can be applied to different system configurations in order to compare their relative security.

3.2.2 Initial information about Agenda Risk tool

AgenaRisk is a powerful tool for modeling, analyzing and predicting risk. When you need make decisions involving uncertainty, *AgenaRisk* will be suitable [7]. A fragment of representation the *AgenaRisk* model is shown in fig.3.2.

AgenaRisk supports both diagnostic and predictive reasoning about uncertainty using risk maps, otherwise known as Bayesian networks. The state-of-the-art algorithms implemented in *AgenaRisk* allow you to model the following classes of problem:

- Simulation of statistical distributions for predictive inference.
 - Diagnostic inference for machine learning applications.
 - Hierarchical modelling as an alternative to Monte Carlo Markov Chains.
 - Construction of hybrid models containing discrete and continuous uncertain variables
 - Mixture modelling of discrete and continuous distributions.
 - Representation of expert judgement using subjective probability.
 - Dynamic modeling of time-based or evolving systems (e.g. Markov analysis).
 - Object-oriented modelling of complex systems involving multiple objects and interfaces.
- AgenaRisk* software has the following features:
- Risk Maps (Bayesian networks) for modeling causal relationships.
 - Probability Tables that allow quantifying these relationships using either expressions or explicit probability values.
 - Risk Graphs that display the results of running your models.
 - Risk Tables.

- A Scenario manager which allows creating multiple scenarios, clone, delete and show/hide visibility in Risk Graph/Table. Multiple scenarios give you the capability to compare data in both Risk Graph and Table views.
- A Sensitivity analyzer which allows analyzing sensitivity of nodes based upon a variation of sources.
- Import/Export functionality that allows to import / export data from models.
- A Risk Explorer that allows you to build and view complex, nested models.
- A large variety of Sample Models that cover different modeling problems.

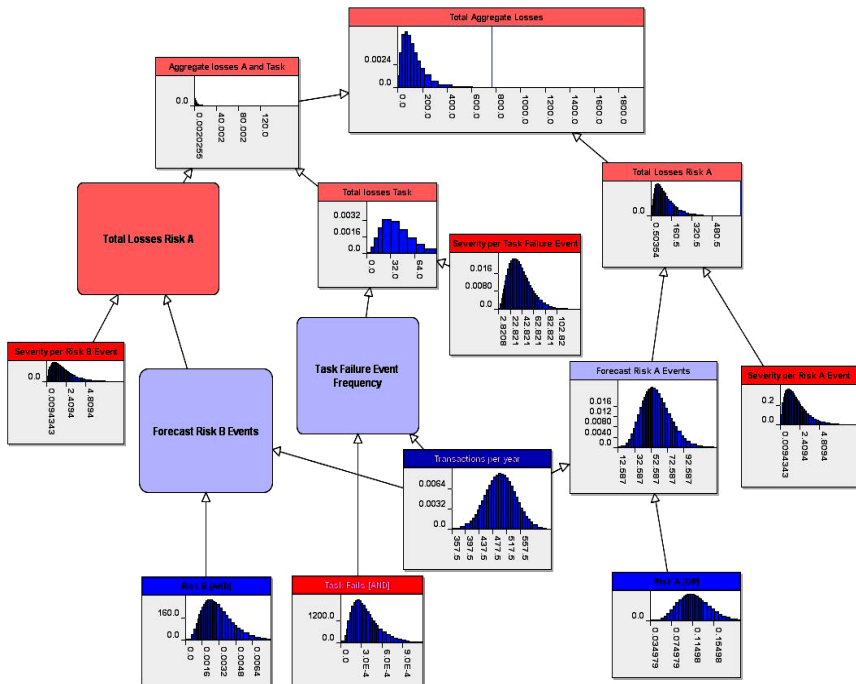


Figure 3.2 – Fragment of AgenaRisk model [7]

3.2.3 Creating new model in Agenda Risk environment

To create a new model you need to click File → Create New Model or Ctrl + N (see fig. 3.3).

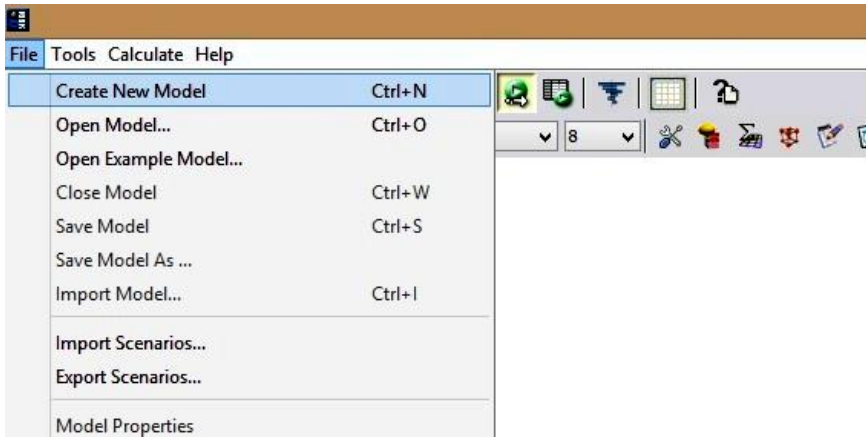


Figure 3.3 – Window for creating a new model

To create a new node, click on the icon "Create New Node" and drag on an empty space in the model (fig. 3.4).

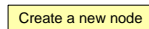


Figure 3.4 – Creating a new node

To set options select the node and click the right mouse button on the menu that appears, select "Properties" or select a node and then press Ctrl + Enter. Then specify node details typing them in appropriate cells "Node name" and "Unique Identifier" (the node ID) (see fig. 3.5).

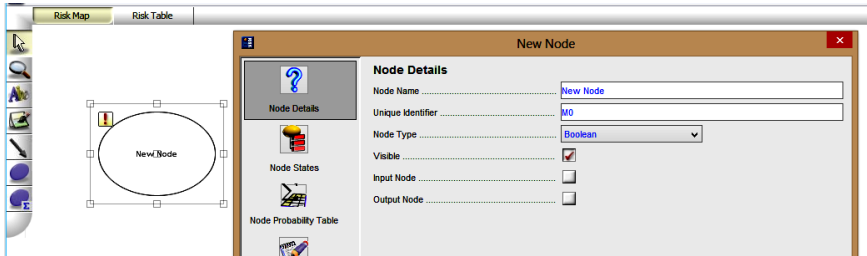


Figure 3.5 - Settings

To indicate the risk (1.0 - 100%) open paragraph "Node Probability Table" (see fig. 3.6).

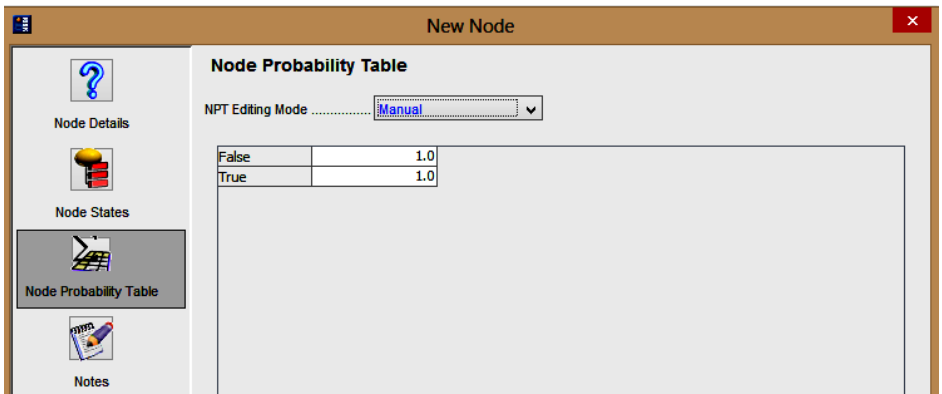


Figure 3.6 – Node probability table

To connect two nodes, click "Create a new link" (Fig. 3.7).

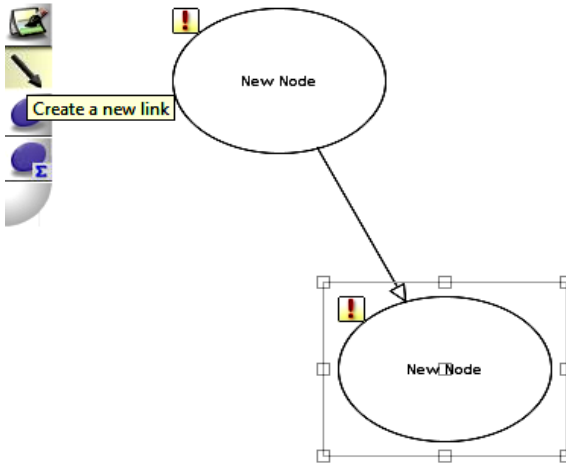


Figure 3.7 - Connect two nodes

To calculate risks you just need to specify the conditions in the nodes and to start up simulation mode (press Ctrl + R).

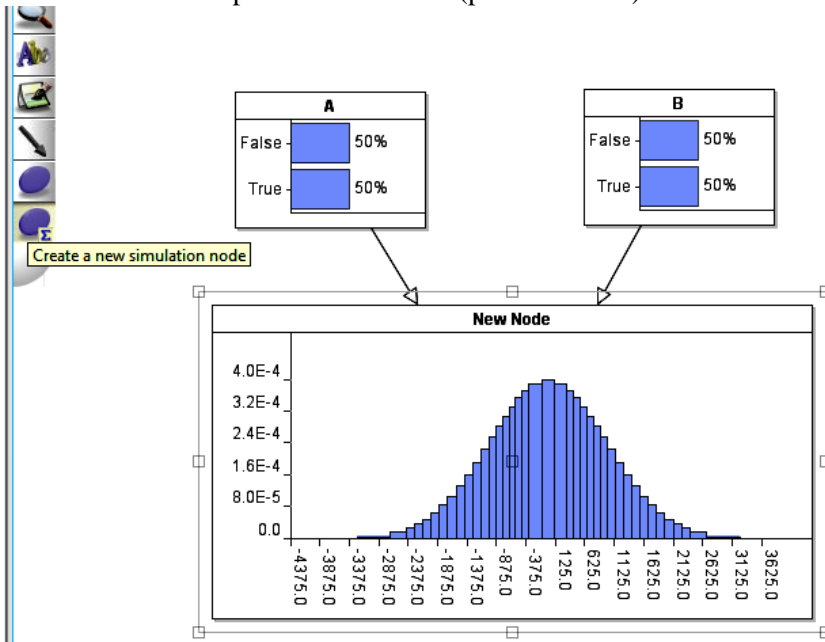


Figure 3.8 – Result of simulation.

3.3 Execution order and discovery questions

1. Obtain the initial data to perform individual tasks. An example of assignment is represented in the fig. 3.9 and in the table below.

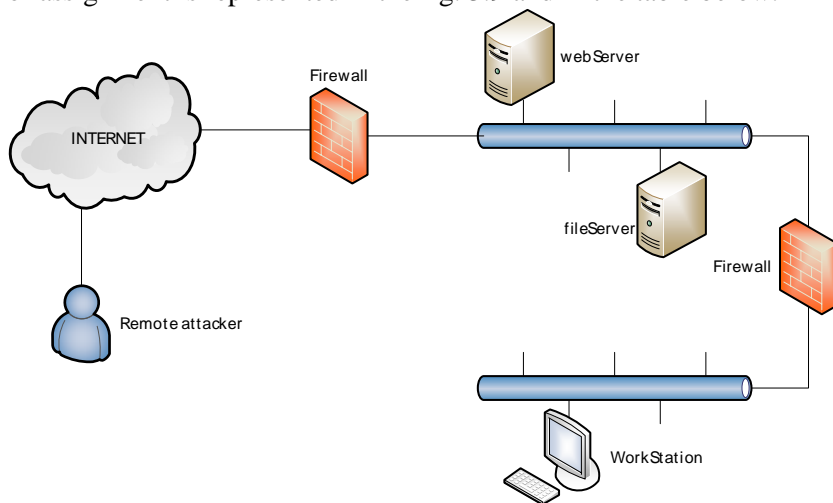


Figure 3.9 – An example of networked information system configuration

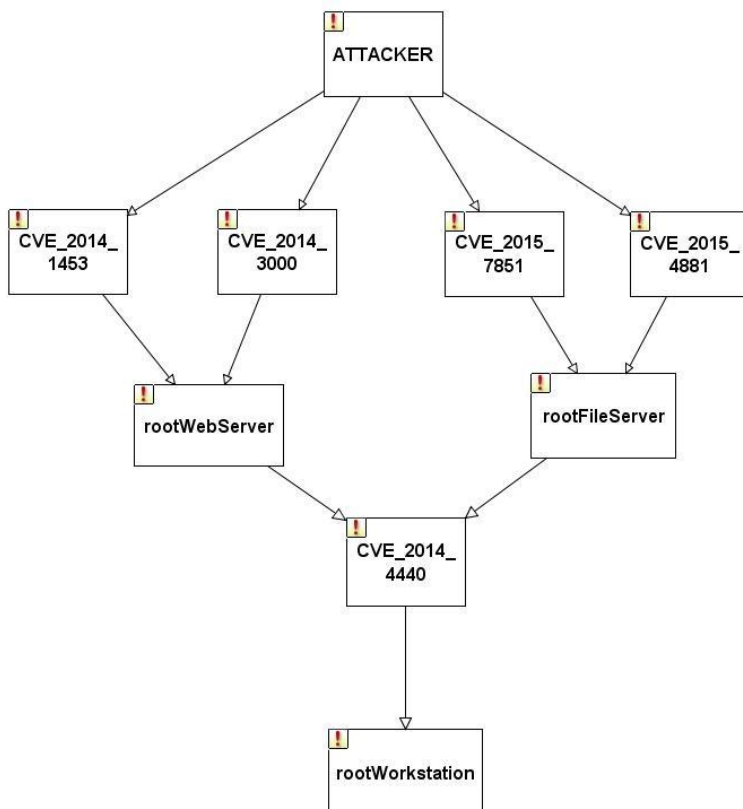
DEVICE	DESCRIPTION	CVE Number	CVSS	AttackPr.
Web Server	BSD (OS)	CVE-2014-1453	4.0	L(0.15) M(0.22) H(0.39)
		CVE-2014-3000	7.8	L(0.5) M(0.6) H(0.75)
File Server	Linux RedHat (OS)	CVE-2015-7851	4.3	L(0.18) M(0.25) H(0.41)
		CVE-2015-4881	6.8	L(0.31) M(0.44) H(0.66)
Work Station	Mac OS X	CVE-2014-4440	2.6	L(0.1) M(0.15) H(0.25)

Attacker: Low (0.2), Medium (0.6), High (0.8)

2. Draw an attack graph for your specified network configuration. As we studied in lab 2, attack graphs model describes the knowledge about how multiple vulnerabilities can be combined for an attack. The model represents system states using security-related conditions, such as the existence of vulnerabilities on a host or the connectivity between hosts, and state transitions using exploits of vulnerabilities. For our purposes, an attack graph is a directed graph with conditions and exploits as vertices, and their relationships as edges.

3. Download *AgenaRisk* or another tool to study risks modeling techniques with Bayesian networks (choose from the Appendix C).

4. Configure a Bayesian network by interpreting an attack graph and using information about attacker experience (level) and the vulnerabilities.



5. Define the settings for each item.

In case of AgenaRisk tool this procedure is carried out as follows:

a) Let us consider the high level (0.8) attacker. To do this we set the parameters of the attacker using the formula:

$$\begin{aligned} \text{False} &= 1 - \text{Attacker Skill (Low, Medium, High)} \\ \text{True} &= \text{Attacker Skill} \end{aligned}$$

False	0.2
True	0.8

b) For each CVE number we set the parameters of vulnerabilities and put the into the appropriate conditional probability table.

	False	True
False	1	0
True	1 - Probability	Probability

In our experiments we'll assume that "FF=1" and "FT = 0" are the constant (Bernoulli distribution). And as a result, we enter in appropriate AgenaRisk table the following values

ATTACKER	False	True
False	1	0.0
True	0.61	0.39

c) Next step devoted to the setting up the servers parameters in accordance with initial data (see the example below). Tips: It is recommended to consider the numbers in the highlighted orange cells of conditional probability table as constants, and to use them in further experiments.

CVE_name	False		True	
CVE_name	False	True	False	True
False	0	1 - CVSS_2/10	0	CVSS_2/10
True	1 - CVSS_1/10	0	CVSS_1/10	1

The tables for Root web server and file server is shown below:

Root Web Server

CVE_2014_...	False		True	
CVE_2014_...	False	True	False	True
False	0.0	0.22	0.0	0.78
True	0.6	0.0	0.4	1.0

Root File Server

CVE_2015_...	False		True	
CVE_2015_...	False	True	False	True
False	0.0	0.32	0.0	0.68
True	0.57	0.0	0.43	1.0

d) To CVE_2014_4440 we set the parameters of vulnerabilities using next conditional probability table:

Server_name	False		True	
Server_name	False	True	False	True
False	0	1 - Probability	0	Probability
True	1 - Probability	0	Probability	1

rootWebSe...	False		True	
rootFileServer	False	True	False	True
False	0.0	0.75	0.0	0.25
True	0.75	0.0	0.25	1.0

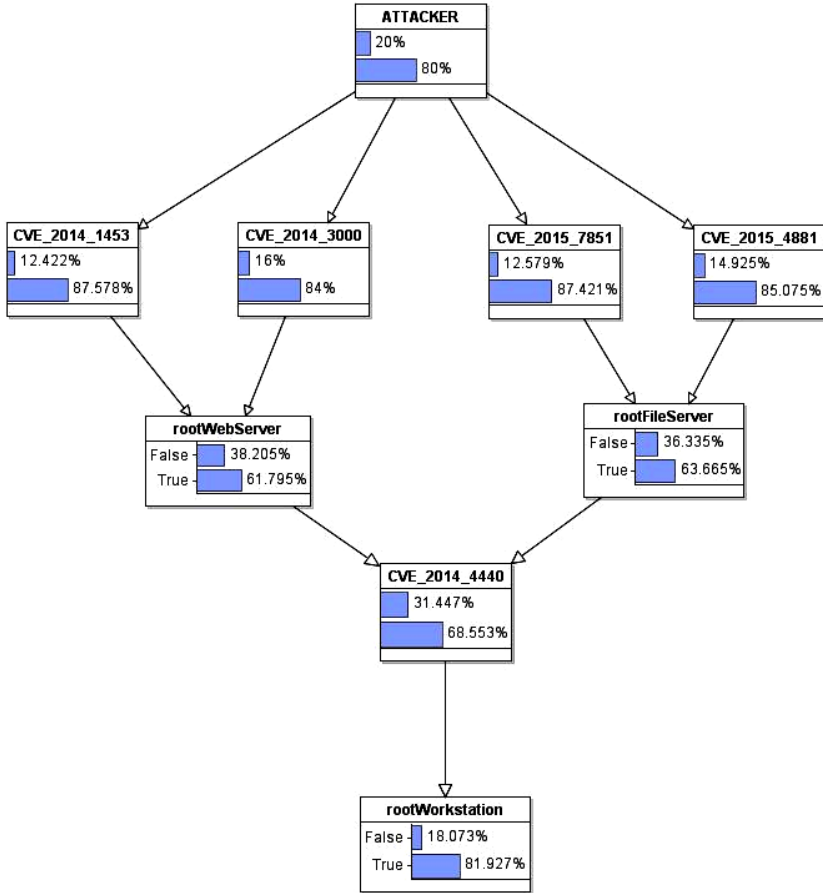
e) Then we set the parameters of the workstation using next conditional probability table. As previously, values marked in orange is recommended to uphold.

CVE	False	True
False	1	0
True	1 – CVSS/10	CVSS/10

Root Workstation

CVE_2014...	False	True
False	1.0	0
True	0.74	0.26

6. Press Ctrl + R (in case of *AgenaRisk*) and start simulating.



8. Make conclusions on the results of modeling. Describe possible scenarios for the onset of cyber-attacks and the weakest points in the system.

Example: For the present case, fully trained attacker is more likely will take advantage of the web server vulnerability (87.5%) CVE-2014-1453. Next he or she will exploit a vulnerability in Mac OS X, through which the attacker will receive administrative privileges on the workstation (82%).

9. How can these findings be considered by system architect and cyber security specialist? What practical value of BN model for analyzing cyber threats risks?

3.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions;
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures;
- (R): narrate (like a story), conditional probability tables, indicate final results;
- (D): answers on discovery questions, explanation of results, and conclusion.

3.5 Test questions

1. What is the main advantage of BN representation to for representing security metrics compared with the attack graph technique?
2. How to calculate the joint probability function for any exploit or condition in the given network?
3. Explain term “conditional probability”.
4. What metrics can be used for modeling various aspects of vulnerability?
5. What criteria should meet successful solution to system security?

3.6 Recommended literature

1. Weber P. Overview on Bayesian Networks Applications for Dependability, Risk Analysis, and Maintenance Areas / P.Weber, G. Medina-Oliva, C. Simon and B. Iung // Engineering Application of Artificial Intelligence. 2012. – Vol. 25, issue 4. – P. 671-682.
2. Namwongse P. Application of Bayesian Network Model for Enterprise Risk Management of Expressway Management Corporation / P. Namwongse, Y. Limpiyakorn // International Conference on Innovation, Management and Service IPEDR. – 2011. – vol.14. – pp.

260-265. - <http://www.ipedr.com/vol14/46-ICIMS2011S10035.pdf> (accessed October 10, 2015).

3. Common Vulnerability Scoring System - Version 2 Calculator <https://nvd.nist.gov/cvss.cfm?version=2&calculator> (accessed October 10, 2015).

4. National vulnerability database. available at: <http://www.nvd.org>

5. Frigault M. Measuring network security using bayesian network-based attack graphs / M. Frigault and L. Wang // Proceedings of the 3rd IEEE International Workshop on Security, Trust, and Privacy for Software Applications (STPSA'08), 2008.

6. Frigault M. Measuring Network Security Using Dynamic Bayesian Network / M. Frigault, L. Wang, A. Singhal, S. Jajodia // Proceeding of the 4th ACM workshop on Quality of protection. – ACM NY, USA 2008. – pp. 23-30.

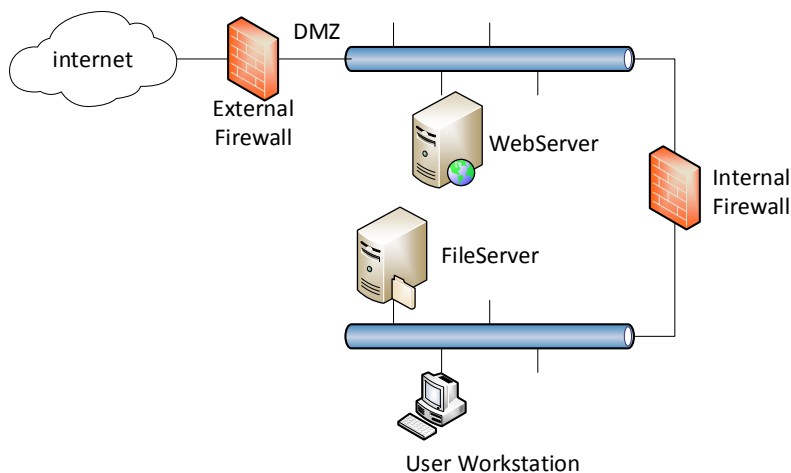
7. AgenaRisk models risk and uncertainty using Bayesian Networks <http://www.agenarisk.com/> (accessed October 10, 2015).

8. Korb K. B. Bayesian Artificial Intelligence / Kevin B. Korb and Ann E. Nicholson // Chapman & Hall/CRC Computer Science & Data Analysis, 2010. – 491 p. - <http://www.csse.monash.edu.au/bai/book/about.php> (accessed October 12, 2015).

9. Xie P. Using Bayesian Networks for Cyber Security Analysis / P. Xie, J. H. Li, X. Ou, P. Liu, R. Levy // http://people.cis.ksu.edu/~xou/publications/dsn10_preprint.pdf (accessed October 8, 2015).

3.7 Assignments to laboratory work 3

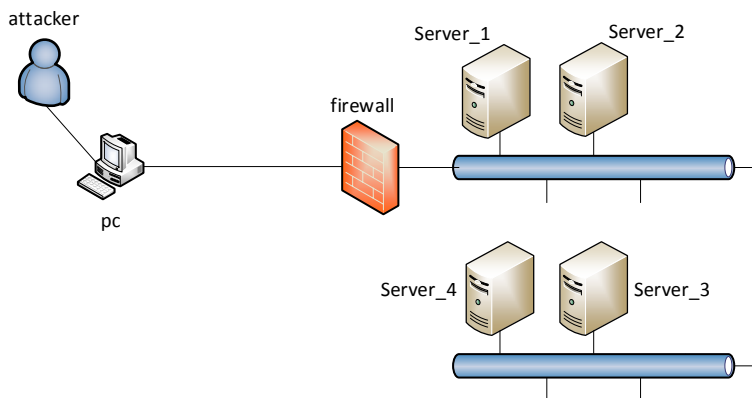
V1



Attacker: Low (0.2); Medium (0.6); High (0.8)

DEVICE	DESCRIPTION	CVE Number	CVSS	AttackPr.
Web Server	BSD (OS)	CVE-2015-3253	7.5	L(0.39) M(0.56) H(0.74)
File Server	Windows 2003	CVE-2013-1283	6.9	L(0.33) M(0.51) H(0.68)
		CVE-2010-0035	6.3	L(0.28) M(0.47) H(0.62)
User Workstation	Mac OS X	CVE-2015-5922	10.0	L(0.71) M(0.89) H(0.97)
		CVE-2015-5836	4.3	L(0.16) M(0.33) H(0.42)

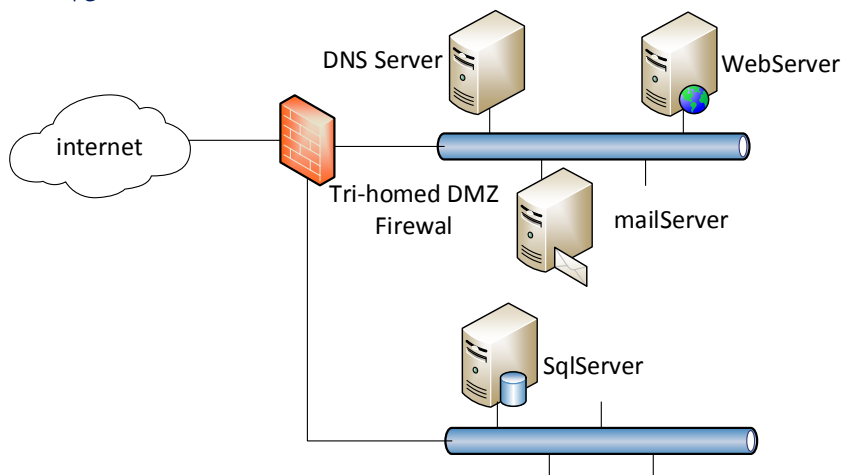
V2



Attacker: Low (0.2); Medium (0.6); High (0.8)

DEVICE	DESCRIPTION	CVE Number	CVSS	AttackPr.
Server1	Windows NT 4.0	CVE-2004-0330	10.0	L(0.68) M(0.79) H(0.85)
		CVE - 2004-1992	5.0	L(0.25) M(0.36) H(0.48)
Server2	Windows 2000	CVE - 2003-0533	7.5	L(0.48) M(0.57) H(0.69)
Server3	RedHat Linux 7.3	CVE - 2004-0417	5.0	L(0.25) M(0.36) H(0.48)
		CVE - 2004-0415	2.1	L(0.1) M(0.13) H(0.17)
Server4	RedHat Linux 7.3	CVE - 2002-0392	6.4	L(0.2) M(0.4) H(0.61)

V3



Attacker: Low (0.2); Medium (0.6); High (0.8)

DEVICE	DESCRIPTION	CVE Number	CVSS	AttackPr.
DNS Server	DNS Cache Poisoning	CVE 2008-1447	5.0	L(0.21) M(0.37) H(0.48)
Web Server	IIS vulnerability in WebDAV service	CVE 2009-1535	7.6	L(0.39) M(0.56) H(0.75)
Mail Server	Remote code execution in SMTP	CVE 2004-0840	10.0	L(0.51) M(0.69) H(0.85)
	Error message information leakage Squid port scan vulnerability	CVE 2008-3060	5.0	L(0.17) M(0.32)
		CVE 2001-1030	7.5	H(0.48) L(0.36) M(0.52) H(0.73)
SQL Server	SQL Injection	CVE 2008-5416	9.0	L(0.5) M(0.62) H(0.81)

Laboratory work 4

ASSESSING RISKS AND OPPORTUNITIES IN ENTERPRISE ARCHITECTURE

Goal and objectives: In this laboratory work we will focus on the analyzing security and resilience of enterprise architectures.

Learning objectives:

- study probabilistic framework for assessment and prediction of information security of enterprises architecture and their information systems;

Practical tasks:

- acquire practical skills in working with the tool for enterprise architecture analysis
- perform security risk analysis on enterprise architecture models;
- draw up the reports

Exploring tasks:

- discover predictive, probabilistic architecture modeling techniques to specify and apply assessment frameworks for performing property analysis on enterprise architecture models;
- investigate how to increase cyber security and resilience of enterprise architecture.

Setting up

In preparation for laboratory work it is necessary:

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1,3] (additional useful information you can find in [4-6]);
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

The Enterprise Architecture Analysis Tool (EAAT) *Object Modeler* –

<https://www.kth.se/ees/omskolan/organisation/avdelningar/ics/research/sa/p/eaat/downloads-1.387300>.

As alternative tool to analysis of enterprise architecture models you can use Aris, ETIS, QualiWare, System Architect, etc.

CySeMoL class model, used for assessment and prediction of information security –

<https://www.kth.se/en/ees/omskolan/organisation/avdelningar/ics/research/cc/cysemol/downloads-1.432383>

4.1 Synopsis

Cyber security risk assessment is often a manual process, requiring brainstorm sessions to identify possible attacks. Attack opportunities will be identified and prevented only if people can envisage them. In today's dynamic attack landscape, this process is too slow and exceeds the limits of human imaginative capability.

Emerging security risks and multi-step attacks demand tool support to predict, prioritize, and prevent complex attacks systematically. It seems natural to use existing models of organizations, such as enterprise architectures, as a basis for automating part of this process.

As it mentioned above in this laboratory work we will focus on the analyzing security and resilience of enterprise architectures by simulating in *EAAT Object modeler* and *CySeMoL*.

4.2 Brief theoretical information

Enterprise architecture models should be amenable to analyses of various properties, as e.g. the availability, performance, interoperability, modifiability, and information security of the modeled enterprise information systems.

As model developers we'll proceed from the fact that descriptions of the network topology and data flows can be key points to analyzing information security of enterprise architecture.

In this contest, it would be a good thing to use the best practices and software to modeling and performing probabilistic assessments of information security.

4.2.1 General information about EAAT Object modeler

EAAT stands for Enterprise Architecture Analysis Tool and is a software system for modeling and analysis of enterprises and their information systems. The *EAAT* tool is inspired by graphical decision-

theoretic methods, such as Bayesian networks and influence diagrams. *EAAT* is a versatile and user-friendly environment for modeling and analysis of enterprises and their information systems. The modules, developed at ICS at KTH in Sweden, are publicly available since September 2008 [1].

EAAT Object modeler is an enterprise architecture modeling tool that uses UML notation for visual representation and a probabilistic calculation engine Probabilistic Architecture Modeling Framework (P2AMF) for analysis.

P2AMF [2] is an extension of OCL (Object Management Group 2010) for probabilistic assessment and prediction of system properties. The main feature of P2AMF is its ability to express uncertainties of objects, relations and attributes in UML-models and perform probabilistic assessments incorporating these uncertainties.

To create a model in *EAAT Object Modeler* you should follow these steps [3]:

- I. Understand the real life situation in terms of the concepts defined in the (EAAT Class Modeler) class model.
- II. Create the structure of the model in the tool by adding objects and drawing relationships between them.
- III. Map the data needs for object attributes (evidence) and gather/ or estimate it, then input it into the model.
- IV. Choose proper calculation method and the amount of samples used. Depending on the class model, size of the object model, and the precision required, about 1000 to 10 000+ samples might be appropriate.
- V. Possibly modify the model and redo steps from I to IV.
- VI. Interpret or compare the calculation results.

4.2.2 General information about CySeMoL

The *CySeMoL class model* is used for assessment and prediction of information security. The system is represented as a graph with nodes representing parts of the system and edges how they are connected. The edges can be of different types, creating different conditional relations between nodes, even between the same pair of types depending on the kind of relation the nodes have.

CySeMoL was previously based on the Probabilistic Relational Model proposed by Sommestad but is now instead based on the P2AMF framework. The two frameworks are two different meta models from which different concrete models can be instantiated.

They describe what kind of objects should be contained in the model and how they can be related to each other. They are both based on Bayesian networks but differ on what classes of objects they contain and how different real world objects are mapped to objects in the model.

The *CySeMoL* graph is not a Bayesian network in itself but is instead used to generate one when the actual computation is performed. Every node in the *CySeMoL* graph has several attributes belonging to one of two categories: "attack steps" and "defences". Every such attribute is a node in the Bayesian network and their conditional dependencies is based on the *CySeMoL* model.

For example the *CySeMoL* subgraph shown in fig.4.1 has three nodes and two edges. When the calculations are to be performed, this is transformed into a Bayesian network with 27 nodes and even more edges. The relations in the network are based both on previous research and assessments from domain experts with Cooke's method.

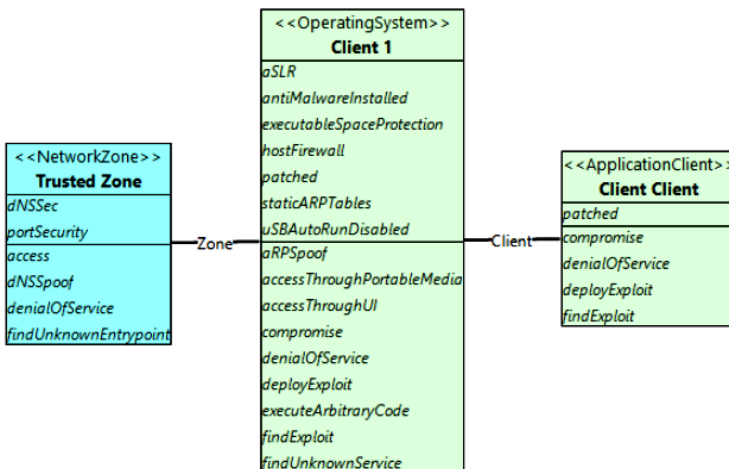


Figure 4.1 – A fragment of a *CySeMoL* model

By grouping attack steps together and focusing on more concrete parts of the system, *CySeMoL* helps abstract away a lot of the details of the attack graph by allowing the user to focus on the larger components of the system instead of details of the exact method of attack.

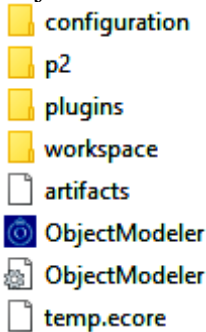
For example, there are a lot of different ways in which a server can be compromised which leads to the same outcome but the user only needs to define the server in terms of its operating system, other software and relations to the world around it.

4.2.3 Setting up modeling environments

To perform simulation you should install software and prepare your modeling environment, please follow the next steps:

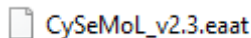
1. Install software package JDK (Java Development Kit) –
<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
2. Download archive *EAAT Object Modeler* –
<https://www.kth.se/ees/omskolan/organisation/avdelningar/ics/research/sa/p/eaat/downloads-1.387300>

Archive contains 8 objects:



3. Download archive *CySeMoL class model*.
<https://www.kth.se/en/ees/omskolan/organisation/avdelningar/ics/research/cc/cysemol/downloads-1.432383>

Archive contains 1 object:



Unzip the archive *Object modeler* and run ObjectModeler.exe.

Select CySeMoL's metamodel or an object model created using CySeModel.

4. If *Object Modeler* does not working you should change Environment Variables. To do this, refer to <https://www.java.com/ru/download/help/path.xml>

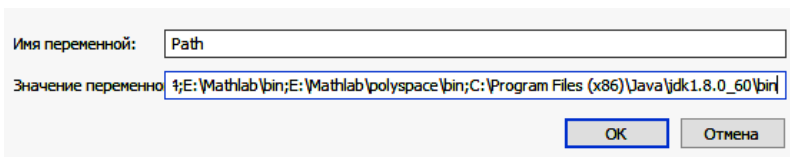


Figure 4.2 – Change the system variable

Instead of `\to\path\Java\jre1.x.x_xx\bin;`
write: `\to\path\Java\jdk1.x.x_xx\bin;`

As a result, you run *EAAT Object Modeler* (fig. 4.3).

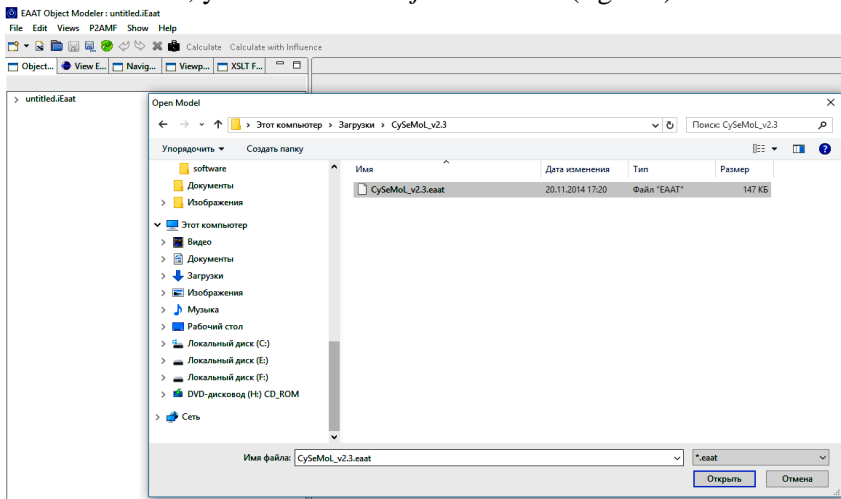


Figure 4.3 – EAAT active window

4.3 Modeling and simulating with *CySeMoL*

1. To develop the model of enterprise architecture you can use Templates (fig. 4.4).

4. Assessing Risks and Opportunities in Enterprise Architecture

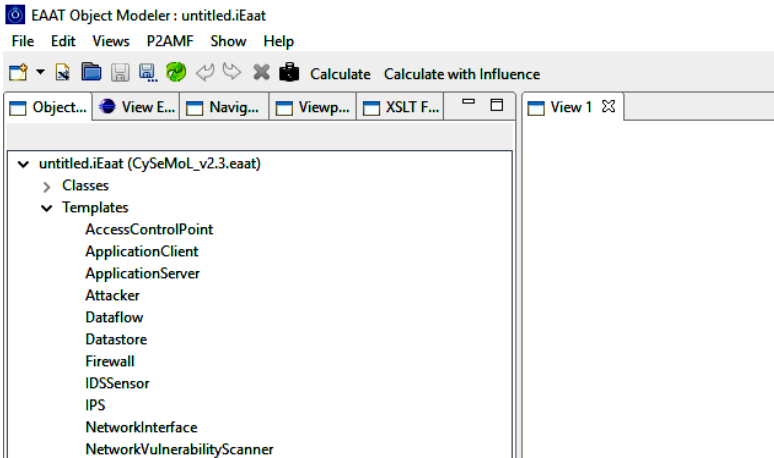


Figure 4.4 – Fragment of Templates menu

To depict a Template you must choose a necessary object and then simply drag-and-drop it to the canvas (fig. 4.5).

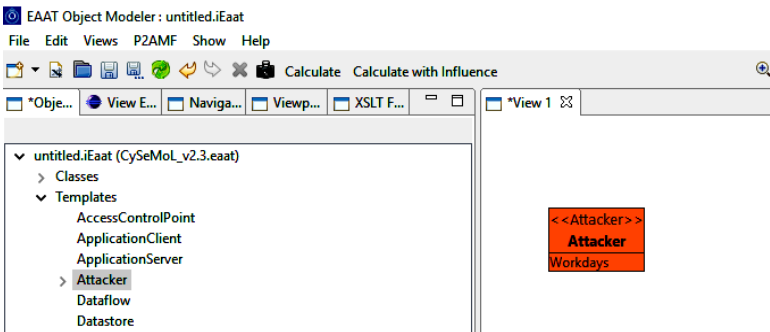


Figure 4.5 – Attacker representation

2. The next stage is setting attributes and characteristics of objects.

The state of an attribute can be altered by first selecting it and then changing its given value in the properties tab on the right (fig. 4.6).

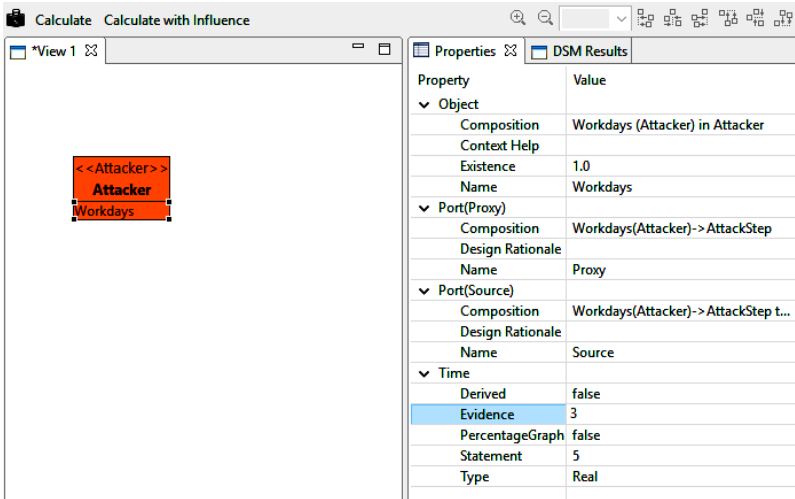


Figure 4.6 – The setting attributes

3. To connect two Templates hold ALT and left click the two Templates those are concerned. If there are more than one possible connection a dialogue will show up depicting the possible choices (fig. 4.7).

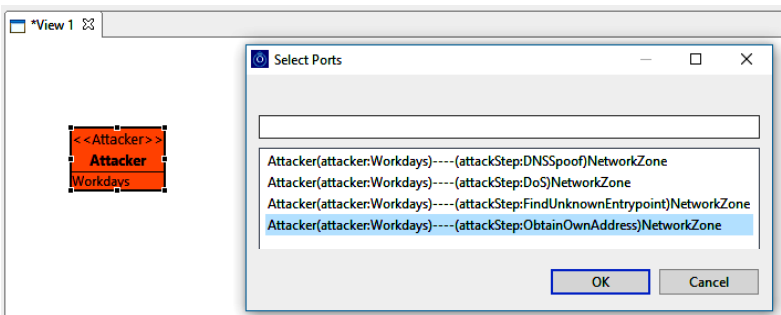


Figure 4.7 – Dialog window for choosing connection

After choosing required ports two Templates will be connected. By this route, step-by-step you can create your object model.

Remember that any object in your model can be renamed by double-clicking on its name (fig. 4.8).

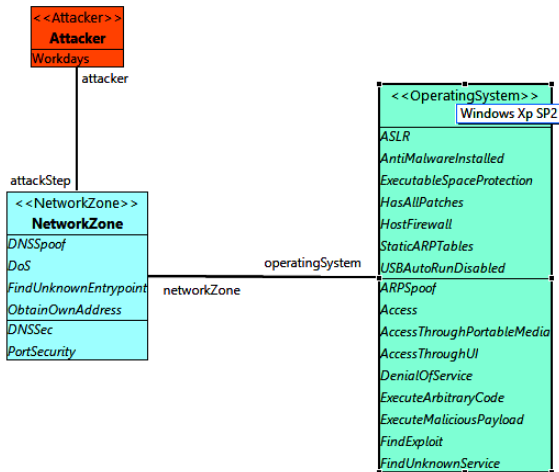


Figure 4.8 – Renaming objects

4. When the object model is ready you should check all states and if it is necessary change them manually (refer to fig.4.6).

If a state of any object is undefined its value is derived based on its default state given in *CySeMoL* metamodel. In the view ‘properties’ (given in the right column if you clicks an empty space in the canvas), it is possible to alter, how objects and connections are visually presented.

The connections and objects in an object model are not formally coupled to view; a view simply shows the parts of an object model that a user has specified. Existing objects can be depicted in multiple views. To add more views use the “Add View” button (fig. 4.9).

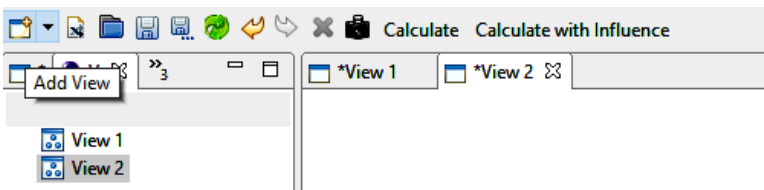
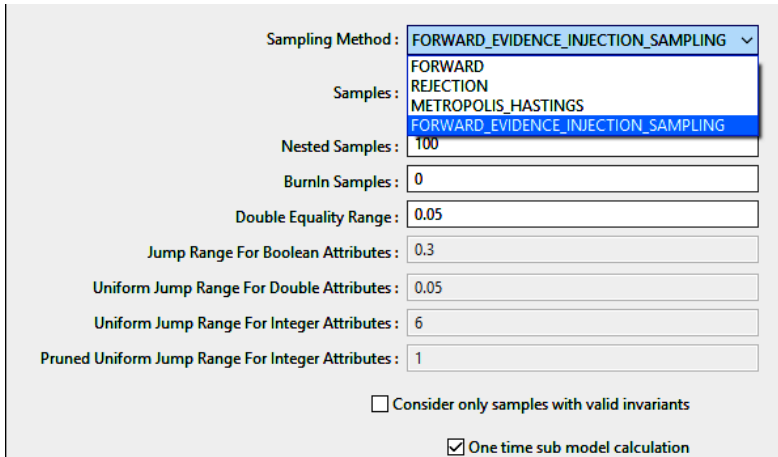


Figure 4.9 – ‘Add View’ button

5. Calculation. To enable calculations of vulnerability, click “Configurations” in the P2AMF tab to set up calculation properties.

Chose the `FORWARD_EVIDENCE_INJECTION_SAMPLING` method in the pop-up list (fig. 4.10).

Then choose a desired number of samples (more samples means more accurate results but also more time required for the calculation).



Sampling Method : `FORWARD_EVIDENCE_INJECTION_SAMPLING`

Samples : `FORWARD_EVIDENCE_INJECTION_SAMPLING`

Nested Samples : `100`

BurnIn Samples : `0`

Double Equality Range : `0.05`

Jump Range For Boolean Attributes : `0.3`

Uniform Jump Range For Double Attributes : `0.05`

Uniform Jump Range For Integer Attributes : `6`

Pruned Uniform Jump Range For Integer Attributes : `1`

☐ Consider only samples with valid invariants

☒ One time sub model calculation

Figure 4.10 – Configuration window

When the object model has been completed and calculation parameters have been defined press “Calculate” to start calculating vulnerability of the object model (fig. 4.11).

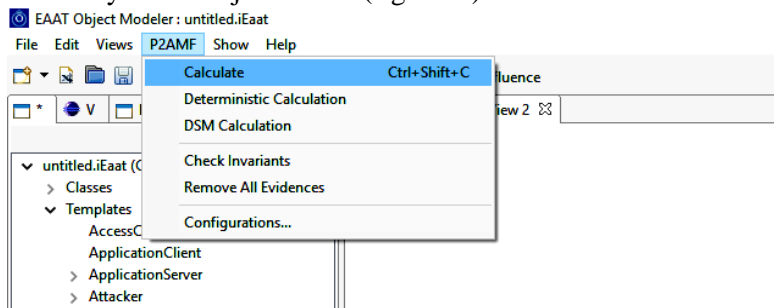


Figure 4.11 – Starter button of calculator

When a calculation has been completed, the probability of the attacker accomplishing the different attack steps in the object model are visualized using a color scale from 0% (green), -50% (yellow), -100% (red). Assets are color-based profile chosen by the user. Probabilities are also available at the bottom left on the screen.

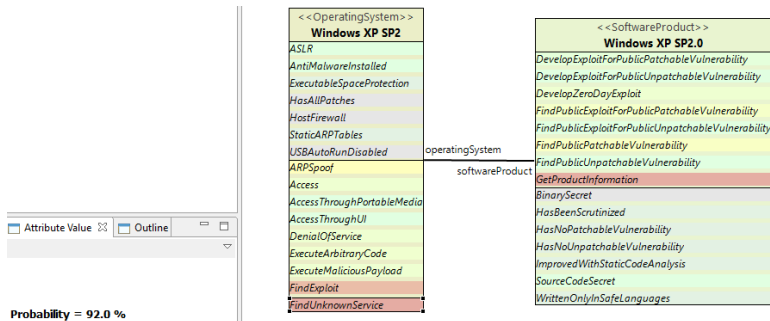


Figure 4.12 – Result visualization

4.4 Execution order and discovery questions

1. Download *EAAT Object Modeler & CySeMoL*.
2. Unzip and run *Object Modeler*.
3. Develop a scheme in the *Object Modeler* in accordance with your assignment.
4. Specify the settings (How to set the configuration see explanation and fig. 4.6).
5. Calculate the risks and define vulnerability of your model. Notice, what are the possible attack methods cyber criminals in your case? Assess the effectiveness of attacks for your architecture.
6. Record the results. If you obtained a high level of security risk you should make up the model reconfiguration to find answer the question how to increase security and resilience of your enterprise architecture. To do this modify the model and redo steps from 3 to 6.
7. Compare the calculation results and describe what protective measures you propose.

4.5 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, procedures
- (R): narrate (like a story), scheme for your assignment, figure of the base model with marked the risks, scheme of your proposed variant with figure a new advanced model;
- (D): answers on discovery questions, explanation of results, conclusion / summary

4.6 Test questions

1. What are the main techniques to analyze of enterprise architectures?
2. How to perform security risk analysis on enterprise architecture models?
3. What are the possible attack methods of cyber criminals?
4. How simulation of a cyber attack criminals can help the experts?
5. How a vulnerability scanner can be utilized for data collection in order to automatically create enterprise architecture models, especially covering infrastructure aspects?
6. How to interpret the results in *CySeMoL*?

4.7 Recommended literature

1. The Enterprise Architecture Analysis Tool (EAAT) / ICS KTH Royal Institute of Technology (Sweden) – <https://www.kth.se/en/ees/omskolan/organisation/avdelningar/ics/research/sa/p/eaat/about-eaat-1.387294> (accessed October 26, 2015).
2. Johnson P. Business Model Risk Analysis: Predicting the Probability of Business Network Profitability / P. Johnson, M. Iacob, M. Vålja, // Enterprise Interoperability Lecture Notes in Business Information Processing, 2013 – No. 144. – pp. 118–130.

3. EAAT User Manual: A guide how to use Class Modeler & Object Modeler –

https://www.kth.se/polopoly_fs/1.428432!/Menu/general/column-content/attachment/EAAT_manual.pdf (accessed October 26, 2015).

4. Study video EAAT from ICS KTH – <https://www.youtube.com/channel/UCWNzcSkeOi3hlGL9VRF4pVQ> (accessed October 15, 2015).

5. Svensson C. Threat modelling of historical attacks with CySeMoL / Master's Thesis at CSC – <http://www.diva-portal.org/smash/get/diva2:838530/FULLTEXT01.pdf> (accessed October 26, 2015).

6. Assessing Risks and Opportunities in Enterprise Architecture using an extended ADT Approach – <http://nmayer.eu/publis/Sousa-Marosin-Gaaloul-Mayer-EDOC13.pdf> (accessed September 28, 2015).

4.8 Assignments to laboratory work 4

Options for group assignments are given in Table 4.1, schemes for proper assignments are given below (fig. 4.13-4.15).

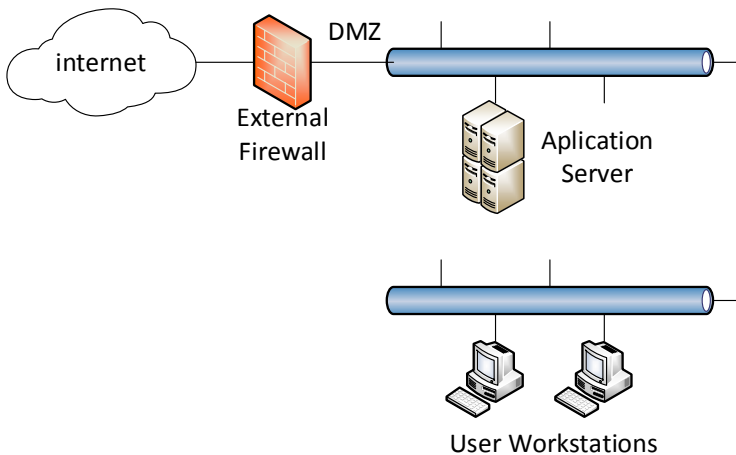


Figure 4.13 – Scheme for variants V1, V4, V7, V10

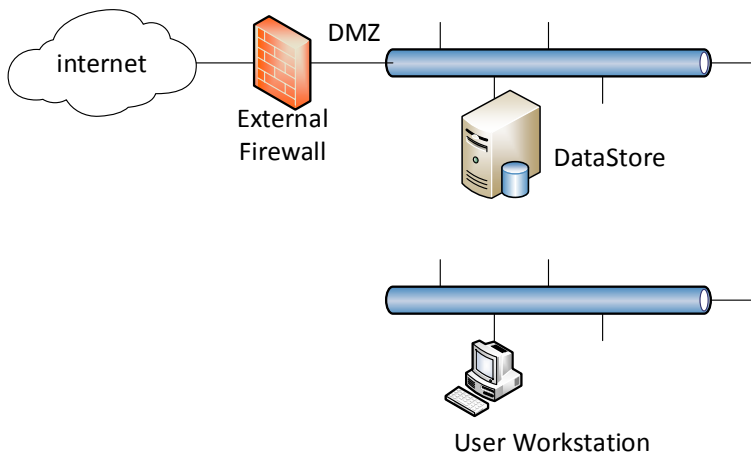


Figure 4.14 – Scheme for variants V2, V5, V8, V11

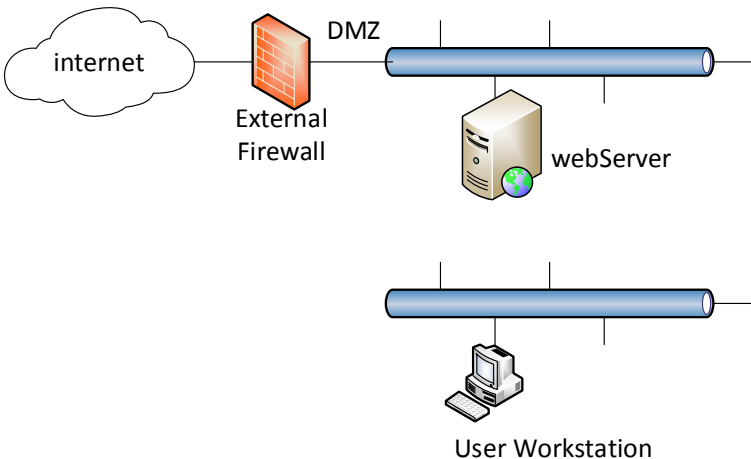


Figure 4.15 – Scheme for variants V3, V6, V9, V12

Table 4.1 - Assignments to laboratory work 4

[illegible]

Table 4.1 - Assignments to laboratory work 4 (continuation)

Variant		1	2	3	4	5	6	7	8	9	10	11	12
Application Server	Find Exploit: Likelihood: Evidence	•	•	•	•	•	•						
	Execute Arbitrary Code: Likelihood: Evidence		•	•		•	•						
	Flood DOS: Likelihood: Evidence							•	•	•			
	Secret Roaming: Likelihood: Evidence								•	•			
	Connect To: Likelihood: Evidence										•	•	•
	Access: Likelihood: Evidence											•	•
AccessControl Point_1	Interface: Likelihood: Evidence	•			•			•					
	Bypass: Likelihood: Evidence										•		
AccessControl Point_2	Bypass: Likelihood: Evidence	•			•			•					
	Interface: Likelihood: Evidence										•		
Application Client 2	Execute Arbitrary Code: Likelihood: Evidence	•			•			•			•		
Data store	Cryptographic Obfuscation: Functioning: Evidence		•			•			•			•	
	Write: Likelihood: Evidence		•										
	Delete: Likelihood: Evidence					•						•	
	Read: Likelihood: Evidence								•				
Application Client 1	Access: Likelihood: Evidence		•			•			•			•	
	Find Exploit: Likelihood: Evidence	•		•				•		•			
	Execute Arbitrary Code: Likelihood: Evidence				•		•						
	Has All Patches: Functioning: Evidence										•		•
Web Application	Exploit SQL Injection: Likelihood: Evidence			•									
	Exploit XSS: Likelihood: Evidence			•									
	Discover Vulnerability: Likelihood: Evidence						•						
	Developer Security Training: Likelihood: Evidence						•						
	Exploit Remote File Inclusion: Likelihood: Evidence									•			
	Has Public XSS: Functioning: Evidence									•			
	Black Box Testing Used: Functioning: Evidence												•
	Exploit Command Injection: Likelihood: Evidence												•
Web Application Firewall	Tuned Using Black Box Tool: Functioning: Evidence			•									
	Tuned With Significant Manual Effort: Functioning: Evidence						•						
	Tuned By Experienced Professional: Functioning: Evidence									•			
	Monitored By Operator: Functioning: Evidence												•

Laboratory work 5

RISK-DRIVEN TESTING OF THE PROGRAM SOURCE CODE

Goal and objectives: In this laboratory work, we will focus on the techniques of code quality enhancement. The idea is to use inspection and/or focus testing on the critical functions to minimize the impact a failure in this function in future production and reduce probability of faults in the system.

Learning objectives:

- study the tools to the simplifying debugging source code of specialized PLC programming language;
- study the potential problems that could affect the cost, safety, or outcome of the systems important to safety.

Practical tasks:

- acquaintance with the static analysis tool to detect safety-relevant errors in PLC source code;
- provide automatically unit testing for software components of microprocessor system.

Exploring tasks:

- examine and interpret information about the software product and classify it to determine the amount of potential risk in the test phase;
- investigate how a risk-driven testing approach can be used ensure quality in systems important to safety.

Setting up

In preparation for laboratory work it is necessary:

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1–3];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources: interpreter *AutoTestDFB*

5.1 Synopsis

In order to ensure the quality of safety-critical software, testing procedures is of great importance. Further work is associated with the

automation of the testing process that will identify errors in the algorithm of the program in an automatic testing as well as the selection of the metrics and static code attributes for defect prediction.

5.2 Brief theoretical information

Sometimes we are not able to decide where to start testing and how much testing should be done of a software system on which we are working because there are too many functional and complex scenarios in the system. At that time then we should use a risk-based approach to testing the system. If we want to decide how much software testing should be done for such systems then we should consider the project and/or product risk identifiable approach, this is risk-based software testing.

There are two types of risks in developing software products: project risk and product risk. In this laboratory work, we make an emphasis on the second one. This risk is related to the functionality of the product. The existence of a major functionality failure or a missing critical functionality in the software then it is called as a product risk. These also include performance issues, security issues, crash scenarios, etc.

Risk-based or risk-driven testing is a testing approach that aims at focusing the testing process on the aspects of features of the system under test that are most exposed to risk.

Risk-based testing approach follows certain steps as mentioned:

1. Identify potential risks that may occur if particular modules/functions of the application are not tested completely.
2. Determine the likely impact of each risk as well as the probability of its occurrence.
3. Take up testing activities that would eliminate risks with the highest numeric values.

To meet safety requirements, unit tests should include 100% code coverage. It can be achieved by using special software testing tools.

5.2.1 General information about software testing in a railway software development lifecycle

For software development of safety-critical systems in the rail industry apply regulations IEC 62279:2002, EN 50128, and Ukrainian state standard ДСТУ 4178-2003. According to the European standard

EN 50128 “Railway applications - Communication, signaling and processing systems - Software for railway control and protection systems” the software for railway applications must be analyzable, verifiable, testable, and maintainable (paragraph 7.5.1.1) and static analysis of source code is highly recommended for all security integration levels. The functional software for railway microprocessor interlocking system has a fourth level of functional safety in accordance with the classification ДСТУ 4178-2003 “Hardware components set of control and traffic arrangements systems: Functional safety and reliability. Requirements and test methods”.

All stages of the life cycle of functional software include a set of appropriate measures to ensure the required level of safety, reliability and quality.

Verification of functional software at the coding stage is required to guarantee the compliance of results to requirements and restrictions stated on previous design stage and consist of methods of (a) unit testing and (b) integration testing.

As it mentioned in [4], the tests of microprocessor software are relatively simple but expensive, fatiguing and is performed by humans on volumes of code.

The problem is motivated also by usage of special programming language that is vary considerably from general purpose languages C, C++, C# or Java and as a consequence lack of appropriate analysis tools.

And at last, software for railway microprocessor system consists of different modules or units: points control, signal manipulation, traffic control, etc. And different groups of coders involved in programming to achieve sufficient software diversity what raises additional difficulty in software quality control.

All this call forth development technical solutions that allow, on the whole, to increase software quality and also to reduce the time and costs of software development.

Our goal is to provide automatically unit testing for software components of microprocessor system.

In this laboratory work, we use static analysis tool Autotest DFB to assess software important to safe railway operation.

Static analysis tool detects safety-relevant errors in PLC source code, detects faults in semantic units of language in the program source

code, finds the operators in sections, positions of key macro commands, the sequence order analysis, and their correctness, as well as writes data into the relevant database.

5.2.2 Brief information about testing environment

Development of software components that implement control algorithms in systems important to safety in terms of technological programming is done using the specialized programming language text applications.

To simplify debugging source code are encouraged to use specialized software interpreter, based on the method of static testing, which performs analysis of commands, processing and execution of a given source code and display parameters during testing.

5.2.2.1 Appointment of the interpreter AutoTestDFB

The initial data for the interpreter are:

- program text (file *.dfb);
- section calling the application program (file *.itx).

5.2.2.2 Running the of the interpreter AutoTestDFB

Starting the interpreter for execution is carried out by one of the batch files, depending on the input data.

Command file consists of a character string, the format of which is as follows:

<name of the program> <command-line options>.

The values of command-line options for verifiable source files listed in the table 5.1. If there are no errors in the source code under test DFB the user will see the following message on the screen:

*Analysis fo file <Name DFB>.dfb has been wrapped up.
DFB source code is in the file - <Name DFB>.log*

5.2.2.3 Control Panel of interpreter AutoTestDFB

Control Panel of interpreter *AutoTestDFB* is shown in figure 5.1 The concise description of the fields main dialog box in the program interpreter *AutoTestDFB* is given in Table 5.2.

Table 5.1 – Command Line Options

Verifiable files (*.dfb, *.itx)	Command line options	Name of the command file
PG_AB	DFB=PG_AB ITX=PG_AB LOG=ON	PG_AB.bat
PG_CH	DFB=PG_CH ITX=PG_CH LOG=ON	PG_CH.bat
PG_EC	DFB=PG_EC ITX=PG_EC LOG=ON	PG_EC.bat
PG_KG	DFB=PG_KG ITX=PG_KG LOG=ON	PG_KG.bat
PG_NAGAD	DFB=PG_NAGAD ITX=PG_NAGAD LOG=ON	PG_NAGAD.bat

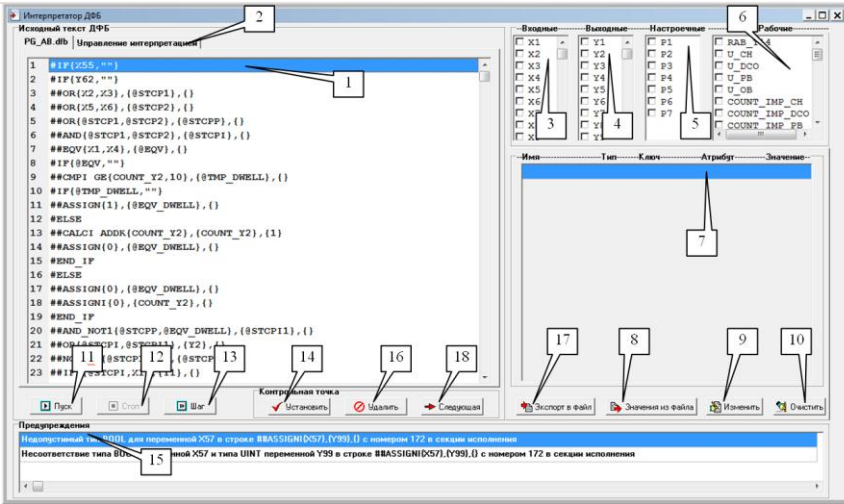


Figure 5.1 – Control Panel of interpreter *AutoTestDFB*

Table 5.2 – Field mapping of a dialog box interpreter *AutoTestDFB*

Number	Type of the field/element	Purpose and accepted value / state
1	Visualization of the source code	This field is used to render the program source code
2	Tab	The tab is used to control the mode of interpretation
3	Selection list box for input variables	List of input variables - name, type, a key name, and the value of the parameter is displayed in field 7 current dialog box
4	Selection list for output variables	List of output variables - name, type, a key name, and the value of the parameter is displayed in field 7 current dialog box
5	Selection list of tuning variables	List of tuning variables - name, type, a key name, and the value of the parameter is displayed in field 7 current dialog box
6	Selection list of operating variables	List of operating variables - name, type, a key name, and the value of the parameter is displayed in field 7 current dialog box
7	Visualization field	This field is used to visualize of names, types, the key names and values of parameters selected in the fields 3-6
8	Button	By pressing this button, the values of the variables are assigned values from a user-specified file on your hard drive
9	Button	Pressing on this button, user can change the current value for the selected variable
10	Button	The button is used for deselecting in fields 3-6
11	Button	Button to start interpretation of the source code in accordance with the predetermined values
12	Button	Button to terminate the process of interpretation of the program source code
13	Button	Button to transition from the current macro function to the next one in the list box 1 during interpretation
14	Button	Button for checkpoints setting
15	Text field	Displays the list of warning messages
16	Button	Button to remove checkpoints
17	Button	By pressing this button, the values of all variables will be saved to a file on your hard drive
18	Button	Button to jump to the next checkpoint

5.3 Execution order and discovery questions

1. Familiarize yourself with the interpreter AutoTestDFB and basic theory.
2. Refer to the assignments to laboratory work 5 and read the description of the following algorithms: PG_NAGAD; PG_KG; PG_AB; PG_CH; PG_EC.
3. Run the interpreter AutotestDFB through one of the batch files, depending on the checked file (*.dfb). Values taken command-line parameters are given in Table 5.1.
4. Detect and correct the errors in the assignment file (*.dfb) with the help of an interpreter.
5. Do some thinking about how a risk-driven testing approach can be used ensure quality in systems important to safety?

5.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary.

5.5 Test questions

1. What is risk-based software testing?
2. The purpose of testing. Classification tests.
3. Unit testing. The concept of the unit.
4. The V-shaped model. Static and dynamic testing.
5. Validation and verification.
6. What are the differences between testing methods of "black" and "white" box?
7. Test case and test coverage.
8. How much software testing should be done for the systems important to safety?

5.6 Recommended literature

1. Michael, C. C. Risk-Based and Functional Security Testing [web] / C. C. Michael, K. van Wyk, W. Radosevich – Available at: <https://buildsecurityin.us-cert.gov> – 23.02.2014 г.
2. Livshits, B. Improving Software Security with Precise Static and Runtime Analysis : PhD dissertation / B. Livshits; Stanford University – USA, 2006. – 229 p.
3. Dusting, E. Implementing Automated Software Testing: How to Save Time and Lower Costs While Raising Quality / E. Dustin, T. Garrett, B. Gauf . – Addison-Wesley Professional, 2009. – 368 p.
4. Churchley, A.R. Microprocessor Based Protection Systems / A. R. Churchley – Springer, 1992. – 290 p.

5.6 Assignments to laboratory work 5

V1. Description of the algorithm of reminders for a single-track open line (file PG_NAGAD.dfb) set a reminder for open line (<name of open line>.NAGAD) is defined as follows:

```
{reminder is set (<name of open line>.NAGAD = 1)},  
if  
{command setting a reminder (<name of open line>.R_KOM =200)}.
```

reset reminder for open line (<name of open line >.NAGAD) is defined as follows:

```
{reset reminder (<name of open line>.NAGAD = 0)},  
if  
{command reset reminder (<name of open line>.R_KOM =201)}.
```

activation of reminder for open line (<name of open line>.NAGAD) is defined as follows:

{activation of reminder (<name of open line>.NAGAD = 2)},
if

{ reminder is set (<name of open line>.NAGAD = 1)
and
route departure is set (<name of open line>.M_SOSTO =1) }.

confirmation of the command with the reminder for open line (<name of open line>.NAGAD) is defined as follows:

{confirmation of the reminder (<name of open line>.NAGAD = 3)},
if

{ command confirmation of the reminder (<name of open line>.R_KOM =202)
and
reminder activated (<name of open line>.NAGAD = 2) }.

prohibition command with the reminder for open line (<name of open line>.NAGAD) is defined as follows:

{prohibition command with the reminder (<name of open line>.NAGAD = 4)},
if

{ command of prohibition for reminder (<name of open line>.R_KOM =203)
and
reminder is actively (<name of open line>.NAGAD = 2) }.

initialization of the reminder for open line (<name of open line>.NAGAD) is defined as follows:

{initialization of the reminder (<name of open line>.NAGAD = 1)},
if

{ route departure is not set (<name of open line>.M_SOSTO =0)
and
{ activation of reminder (<name of open line>.NAGAD = 2)
or
confirmation of the reminder (<name of open line>.NAGAD = 3)
or
prohibition of the reminder (<name of open line>.NAGAD = 4) } }.

V2. Description of the algorithm interaction with key-token (file PG_KG.dfb)

Veracity of signal status of key-token(<name of open line>_KG.STCP) determined by the state of the relay signal №1 KG (SR_<name of open line>KG1.SD) and signal №2 KG (SR_<name of open line>KG2.SD), and veracity of signal (SR_<name of open

line>KG1.STCP and SR_<name of open line>KG2.STCP) proceeds as follows:

```
{ signals of key-token is not veracity (<name of open line>_KG.STCP=1)},
if
{
  (relay contacts of signal №1 KG unworkable (SR_<name of open line>KG1.SD=1)
  or signal №1 KG is veracity (SR_<name of open line>KG1.STCP=1)
  and
  (relay contacts of signal №2 KG unworkable (SR_<name of open line>KG2.SD=1)
  or signal №2 KG is not veracity (SR_<name of open line>KG2.STCP=1)
  or
  (both signals are valid (SR_<name of open line>KG1.SD=0 and SR_<name of open line>KG1.STCP=0)
  and SR_<name of open line>KG2.SD=0 and SR_<name of open line>KG2.STCP=0)
  and
  match (SR_<name of open line>KG1.CVDI = SR_<name of open line>KG2.CVDI) бoлее 1 s
  )
}
else
{signals of key-token is veracity (<name of open line>_KG.STCP=0)}.
```

position control of key-token (<name of open line>_KG.SOST) determined by the state of the relay signal №1 1 KG (SR_<name of open line>KG1.CVDI) and signal №2 KG (SR_<name of open line>KG2.CVDI), and veracity of signal status of key-token (<name of open line>_KG.STCP) proceeds as follows:

```
{ state of key-token equally for value №1 KG inverse duplicate signals from key-token contacts }
{ (<name of open line>_KG.SOST = SR_<name of open line>KG1.CVDI) }
if
{ signal №1 KG is veracity (@stcp1=0)
  and generalized state key-token is veracity (<name of open line>_KG.STCP=0) },
else
{ state of key-token equally for value №2 KG inverse duplicate signals from key-token contacts }
{ (<name of open line>_KG.SOST = SR_<name of open line>KG2.CVDI) }
if
{ signal №2 KG достоверен (@stcp2=0)
  and generalized state key-token is veracity (<name of open line>_KG.STCP=0) },
else
{ state of key-token is not changed }.
```

V3. Description of the algorithm interaction with open line (file PG_AB.dfb).

Formation of the generalized current state of nodes and elements of the circuit for interfacing with open line the following algorithm:

For open line determined by the current combination of signals to determine the status (internal variable @combi, where i – number of path 1 or 2) and sign allowable combinations (internal variable @comb_dopi, where i – number of path 1 or 2) in table **Ошибка! Источник ссылки не найден.** of the following set of signals:

1) reception status (<name of open line>_PG.SOST) with two-way traffic;

2) installation state of reception (<name of open line>_PGM.SOST) with two-way traffic;

3) state of departure (<name of open line>_OZ.SOST) with two-way traffic;

4) installation state of departure (<name of open line>_OZM.SOST) with two-way traffic;

5) direction of automatic lock (<name of open line>_CH.SOST);

Table 5.3 – Combinations of signals for determining the state of open line

number combination @combi	flag of allowable combinations @comb_dopi	reception status *PG.SOST	installation state of reception *PGM.SOST	state of departure *OZ.SOST	installation state of departure *OZM.SOST	direction of automatic lock *CHO.SOST
1	0	0	1	0	0	0
2	0	1	0	0	0	0
3	0	0	0	0	1	0
4	0	0	0	1	0	1

Note – A sign of allowable combinations (@comb_dopi) has the value 0, if the installed combination of admissible, 1 - otherwise.

out the combination signals for determining the state of open line (internal variable @combi) determined direction (<name of open line>_NAPR.SOST) and its veracity (<name of open line>_NAPR.STCP) proceeds as follows:

{state of direction is not veracity (<name of open line>_NAPR.STCP = 1) by default}.

{state of direction is veracity (<name of open line>_NAPR.STCP = 0)},

if


```

{original signals is veracity (<name of open line>_PG.STCP=0 and
<name of open line>_OZ.STCP=0 and<name of open line>_CHO.STCP=0 and
<name of open line>_PGM.STCP=0 and <name of open line>_OZM.STCP=0)
and
set a valid combination of signals (@comb_dopi=0)
}
{state of direction is veracity (<name of open line>_NAPR.STCP = 0)},
if
{original signals is veracity (<name of open line>_PG.STCP=0 and
<name of open line>_OZ.STCP=0 and<name of open line>_CHO.STCP=0 and
<name of open line>_PGM.STCP=0 and <name of open line>_OZM.STCP=0)
and
set a valid combination of signals (@comb_dopi=0)
}
{installation on reception for open line (<name of open line>_NAPR.SOST = 1)}
if
{combination of signals №1 is set (@combi=1)
and
state of direction is veracity (<name of open line>_NAPR.STCP = 0)}
{open line set on reception (<name of open line>_NAPR.SOST = 2)},
if
{combination of signals №2 is set (@combi=2)
and
state of direction is veracity (<name of open line>_NAPR.STCP = 0)}
{installation on departure for open line (<name of open line>_NAPR.SOST = 3)},
if
{combination of signals №3 is set (@combi=3)
and
state of direction is veracity (<name of open line>_NAPR.STCP = 0)}
{open line set on departure (<name of open line>_NAPR.SOST = 4)},
if
{combination of signals №4 is set (@combi=4)
and
state of direction is veracity (<name of open line>_NAPR.STCP = 0)}

```

V4. Description of the algorithm change of direction on open line with Auto Block system (file PG_AB.dfb)

Formation of the command change of direction (<name of open line>.U_CH) for managing interface relays commands to change

direction CH (UR_<name of open line>CH.U_KOM) proceeds as follows:

```
{command change of direction formed to supply power to the relay CH}
{(<name of open line>.U_CH = 1)}
if
{command change of direction (<name of open line>.R_KOM=111)},
else
{command change of direction has not been formed (<name of open line>.U_CH = 0)},
if
{(command change of direction is not received (<name of open line>.R_KOM ≠ 111))}
or
{30 seconds has elapsed since the start of the formation command}
```

V5. Description of the algorithm interaction with relay centralization (PG_EC.dfb)

Formation of state departure for open line (<name of open line>.U_O) for managing an indicator of departure (UR_<name of open line>O.U_KOM) proceeds as follows:

```
{command of state departure for open line is set (<name of open line>.U_O = 1)}
{for managing an indicator of
departure in a pulsed mode (UR_<name of open line>O.U_KOM=0/1)}
if
{installation on reception for open line (<name of open line>_NAPR.SOST = 5}
{and <name of open line>_NAPR.STCP = 0)}
else
{command of state departure for open line is not set
{(<name of open line>.U_O = 0) and indicator of departure is
off (UR_<name of open line>O.U_KOM=0)}
```

Formation of state departure for open line (<name of open line>.U_CH1) for managing relays of departure CH1 (UR_<name of open line>CH1.U_KOM) proceeds as follows:

```
{ command of state departure for open line is set  
{(<name of open line>.U_CH1 = 1) to supply power to the relay CH1  
{(UR_<name of open line>CH1.U_KOM=1) for managing an indicator of  
departure (UR_<name of open line>O.U_KOM=1)  
}  
}  
if  
{ open line set on departure (<name of open line>_NAPR.SOST = 4 and  
{<name of open line>_NAPR.STCP = 0)  
}  
else  
{ command of state departure for open line is not set  
{(<name of open line>.U_CH1 = 0) and the power for managing the relay CH1 is not filed  
{(UR_<name of open line>CH1.U_KOM=0) for managing an indicator of  
departure (UR_<name of open line>O.U_KOM=1)  
}  
}
```

PROGRAM FOR THE SEMINARS

The major topics covered in the seminars are shown in Table.

No.	Topic	Details
1	Cyber crimes & Ethics	Cyber crimes (CC) in industry; CC in governance; CC against property, against person; financial CC, Intellectual property crimes, Cyber criminals (kinds, organized hackers), corporate espionage. Ethics in information security police.
2	IT security risk analysis methods	Quantitative vs. qualitative risk analysis. Qualitative risk analysis basics. Improving quantitative risk analysis with security ontologies. Quality risk analysis and QRM tools. Case studies.
3	Security risk management methodologies	OCTAVE, CORAS, CRAMM, FRAP, COBRA, ISRAM, CORA, IS Risk Analysis Based on a Business Model, RiskWatch, Australian Risk Management Standard AS/NZS 4360:2004, Dutch A&K Analysis, Ebios, ISO/IEC IS 13335-2, ISO/IEC IS 17799, IT-Ground-schutz, Mehari. Integrating HAZOP and SIL/LOPA Analysis
4	ICS security	Security risk metrics in industry. ICS Characteristics, Threats and Vulnerabilities. Major ICS Security Objectives. Comparing ICS and IT Systems. Cyber Attack Modeling and Security risk management methodologies for ICS.
5	A cyber security risk assessment for the design of ICS	Assessments during the system design phase. System Identification and Cyber Security Modeling during design. Asset and Impact Analysis. Threat Analysis. Vulnerability Analysis. Security Control Design. Penetration testing.

Appendix A. Short installation manual for MulVAL attack paths engine

Attack-Paths Engine up & running

The following steps need to be performed to get the Attack-Paths Engine up & running:

1. You need to install the XSB logic engine from <http://xsb.sourceforge.net/>
2. You will also need to check whether GraphViz is already installed on your system by typing "dot".
3. Make sure both the program "xsb" and "dot" reside in your PATH.
4. You need to install MySQL: <http://dev.mysql.com/downloads>
5. Basic Setup: The environmental variable MULVALROOT should point to this package's root folder Include \$MULVALROOT/bin and \$MULVALROOT/utlis in PATH
6. You can run the MulVAL-Attack-Paths Engine directly, if you already have an input file: `graph_gen.sh INPUT_FILE [OPTIONS]`

Installing

MulVAL needs Java and a C++ compiler to be built. Use:

```
apt-get install g++ openjdk-7-jdk
```

Duplicating *MulVAL* from GitHub repository.

```
git clone --depth=50 --branch=master git://github.com/fiware-cybercaptor/mulval.git fiware-cybercaptor/mulval
```

Before building *MulVAL*, the MULVALROOT environment variable has to be set to the folder on which *MulVAL* will be built.

```
export MULVALROOT=/path/to/mulval
export CXX=g++
export CC=gcc
cd fiware-cybercaptor/mulval
sudo apt-get update -qq
```

MulVAL also needs bison and flex dependencies. This can be installed for example on Debian-like distributions using:

```
sudo apt-get install -y bison flex
```

Then, *MulVAL* can be build with `make`

Download XSB. Uncompress XSB package. To do this use:

```
Tar xzf xsb.tar.gz
```

Go to xsb folder, use:

```
cd /to/xsb-src/XSB
```

Use for install XSB:

```
java -jar InstallXSB.jar
```

Install GraphViz .

```
apt-get install graphviz
```

Install MySQL.

```
sudo apt-get install mysql-server  
sudo apt-get install mysql-client  
sudo apt-get install libmysql-java
```

Additional Links

1. *MulVAL* original documentation (installation and user guide) can be found in doc/README - <https://github.com/fiware-cybercaptor/mulval/blob/master/doc/README>

2. Requirements to *MulVAL* installing can be found at <http://people.cis.ksu.edu/~xou/argus/software/mulval/readme.html>

3. The full version of *MulVal* Attack Paths Engine - Installation and Administration Guide available at <http://cordis.europa.eu/fp7/ict/netinnovation/deliverables/fi-ware/fi-ware-d831.pdf> (see pp. 9-16).

Appendix B. Attack graphs generating from Nessus reports

To generate attack graph directly from Nessus report you can employ a technique described below.

1. In unpacked MulVAL package open a folder `'utils'`.
2. Find the file `nvd_sync.sh`
3. Open it by text editor.
4. Modify the line `http://` to the `https://` ... (append the letter "s").

```
java -cp $CLASSPATH mysqlConnectionChecker

if [ -r connectionSucc.txt ]; then
    echo 'connection tested successfully'
else
    # echo 'connection cannot be established'
    exit 1
fi

rm -f connectionSucc.txt

if [ ! -d $NVDPATH ]; then
    mkdir $NVDPATH
fi
cd $NVDPATH
rm -f nvdCVE*
i=2002

year=`date +%Y`
while [ $i -le $year ]; do
    wget https://nvd.nist.gov/download/nvdCVE-$i.xml
    i=`expr $i + 1`
done
cd ..
java -cp $CLASSPATH -Xmx512m InitializeDB $year
echo "NVD update finished. You can remove the temporary NVD files in $NVDPATH."
```

5. On default MySQL is set it with encoding `latin1`. To start server with prerequisite coding, you should edit the file `/etc/mysql/my.cnf`:

```
sudo nano -w /etc/mysql/my.cnf
```

6. Add the following lines to the section [mysqld]:

```
skip-character-set-client-handshake
character-set-server = utf8
init-connect = 'SET NAMES utf8'
collation-server = utf8_general_ci
```

7. It is also desirable to set the encoding for the client and mysqldump. To do this you should add the line in the sections [client] and [mysqldump]:

```
default-character-set = utf8
```

8. Restart MySQL server:

```
sudo service mysql restart
```

9. Create a database and name it 'text':

```
>mysql
>create database test;
```

Create a user with access to that table. Replace the items in square brackets by the database name, and the chosen user and password.

```
grant all privileges on [database].* to [user]@localhost
identified by "[password]";
flush privileges;
```

Example:

```
grant all privileges on test.* to root@localhost identified
by "pass";
flush privileges;
```

You can test this by:

```
mysql -u [user] -h 127.0.0.1 -p
```

For more information about MySQL operation visit <http://www.linuxjournal.su/rabota-s-mysql-iz-komandnoj-stroki/>.

10. In folder `/to/path/mulval/utils` create config.txt where specify your DB on the local server (login and password).

Example of the config file:

```
jdbc:mysql://localhost:3306/test
root
pass
```

11. Enter in command line the environment variables.

```
export CLASSPATH=$CLASSPATH:/usr/share/java/mysql.jar
export MULVALROOT=/root/fiware-cybercaptor/mulval
export PATH=$PATH:/to/path/XSB/bin
export                                NVDPATH=/root/fiware-
cybercaptor/mulval/utils/nvd_xml_files
export                                CLASSPATH=$CLASSPATH:$MULVALROOT/lib/dom4j-
1.6.1.jar:$MULVALROOT/lib/jaxen-1.1.1.jar:$MULVALROOT/lib/mysql-
connector-java-5.1.8-bin.jar:$MULVALROOT/bin/adapter
```

12. On the base of NVD DB create your local vulnerability DB. To do this enter next command in the command line (in folder `/to/path/mulval/utils`):

```
nvd_sync.sh
```

As a result, you will obtain a new folder `nvd_xml_files` in the folder `utils` where NVD DB with all CVE will be hosted.

Appendix C. Bayesian Network Simulation Tools

Name	Graph types	Learn parameters	Learn structure	Has a GUI	Platform (Windows, Linux, Mac OS)	Limitations/ Concerns
AgenaRisk	All	•	•	•	Windows, Linux, Mac OS	Price full version is £2,000. 30 days trial. Simulation by Dynamic iscretionisation
BayesBuilder	Directed			•	Windows, Linux	Doesn't learn parameters or structures
Bayesialab	Directed	•	•	•	Based on Java. any OS with installed JRE	30 days trial version is free of charge
Bayesian Knowledge Discoverer / Bayesware Discoverer	Directed	•	•	•	Windows, Unix, Mac OS	A 30-days trial version is free of charge
Bayesian Network tools in Java	Directed		•	•	any OS with installed JRE	Free
Bayesware	Directed	•	•	•	Windows	A free student edition for 30 days use. Cannot handle DB larger 700 records
Bayes Net Toolbox for Matlab	Directed	•	•	•	Matlab	Concerns does not found
B-Course	Directed	•	•	•	Runs on their server accessed using a web browser	Size of the model is limited
Belief Network Constructor	Directed	•	•	•	Windows	Last updated in 2001
Belief Network Power Constructor	Directed	•	•	•	Windows	Free software

Appendix C. Bayesian Network Simulation Tools

Belief and Decision Networks 5.1.10	Directed	•		•	Java applet	Concerns not found
Elvira	Directed	•	•	•	any OS with installed JRE	Free
FastInf	Undirected	•	*		Linux, Mac OS	No Windows support * - for template Markov Network
Flexible Bayesian Modeling and Markov Chain Sampling	Directed	•	•		Unix	Open Source
GeNIe & SMILE	Directed	•	•	•	Windows, Linux, Mac OS	Free software
GMTK	Directed	•	•		Linux, Solaris, Cygwin	Closed source. Freely available
Hugin	Directed	•	•	•	Windows, Linux, Solaris, Mac OS	The demo version is limited to 50 variables and 500 cases.
JAGS (Just Another Gibbs Sampler)	Directed	•			Unix	Port/clone of OpenBUGS. Open source
JBNC Toolbox-Weka	Only BN	•	•	•	It can run on any OS installed with JRE	Weka provides an open source platform for data mining.
JProGraM - PRObabilistic GRaphical Model in Java	Directed	•	•		Linux	Released under the GNU
MALLET + GRMM package (Java)	Directed, Undirected	•	•		It can run on any OS installed with JRE	"GRMM can handle continuous features, but only discrete variables."
MensXmachina	Directed	•	•	•	Windows	Toolbox of Matlab
MIM	All	•	•	•	Windows	Not free
MSBNX	Directed	•	•	•	Windows	Concerns not found

Appendix C. Bayesian Network Simulation Tools

Netica	Directed	•		•	Windows, Mac OS	The full version is 285\$ for students. The demo version is limited to model size.
Octave	It can run Matlab code, and has some package itself as well.	•	•		Linux	No Windows support
OpenBUGS	Directed	•	•		Windows	Open Source
Pulcinella	Based on the framework of valuation systems				Solaris, Mac OS	Requires a Lisp system
RISO	Directed			•	any OS installed with JRE	Distributed implementation
SamIam	Directed	•	•	•	Windows, Linux, Mac OS, Solaris	Free
Tetrad	Directed	•	•	•	any OS installed with JRE	Free beta version
UGM	Undirected	•			Matlab	It just provides some Matlab functions
WinMine	Undirected, Directed	•	•		Windows	Only for Windows platforms and doesn't have a GUI

COURSE PROGRAM

DESCRIPTION OF THE MODULE

TITLE OF THE MODULE	Code
Risk Analysis of Infrastructure Security and Resilience	

Teacher(s)	Department
Coordinating: Prof. I. Skarha-Bandurova Others: Prof. V. Kharchenko Dr. E. Brezhnev	Computer Engineering Computer Systems and Networks,

Study cycle	Level of the module	Type of the module
Master	A	Full-time tuition

Form of delivery	Duration	Langage(s)
Full-time tuition	One semester	English

Prerequisites	
Prerequisites: Probability Theory and Foundations of Mathematical Statistics; Foundation of Modeling; Computer Networks; Information-Networking Technologies	Co-requisites (if necessary):

Credits of the module	Total student workload	Contact hours	Individual work hours
4	108	36	72

Aim of the module (course unit): competences foreseen by the study programme		
<p>The aim of module is to create a knowledge base for multidisciplinary research on SoS risk management and to provide prerequisites for practical use of risk assessment theories. The study also expands the current research on SoS Security and Resilience by combining system approach and holistic people risk management in the context of the study of security management approach.</p>		
Learning outcomes of module (course unit)	Teaching/learning methods	Assessment methods
At the end of course, the successful student will be able to:		
1. employ the functional representation of system interconnections for structural and resilience analysis in framework of resilience and risk assessment.	Interactive lectures, Learning in laboratories, Just-in-Time Teaching	Module Evaluation Questionnaire
2. select appropriate risk analysis method for different tasks of IT security.	Interactive lectures, Learning in laboratories, Just-in-Time Teaching	Module Evaluation Questionnaire
3. prepare test scenarios designed to validate the performance of security systems and protective force in detecting, interdicting and countering threats.	Interactive lectures, Learning in laboratories, Just-in-Time Teaching	Module Evaluation Questionnaire
4. utilize different statistical tools to compile and analyze data to identify trends regarding security assurance activities and create reports to support customer requirements and/or compliance related requirements.	Interactive lectures, Learning in laboratories, Just-in-Time Teaching	Module Evaluation Questionnaire
5. perform risk analyses and process analyses related to security data manipulation and management.	Interactive lectures, Learning in laboratories, Just-in-Time Teaching	Module Evaluation Questionnaire
6. analyze of PCS technical security risk assessment methods	Engagement with enterprises resources	Module Evaluation Questionnaire

Themes	Contact work hours						Time and tasks for individual work	
	Lectures	Consultations	Seminars	Practical work	Laboratory work	Placements	Total contact work	Individual work
1. System of Systems as an object of security analysis 1.1. Definitions of Infrastructure / System of Systems (SoS). 1.2 SoS classifications. 1.3 Characterization of SoS.	2						7	1.4. Risk representation Probability versus Impact. 1.5. Risk assessment and risk mitigation.
2. Introduction in risk analysis of SoS structures 2.1 Architecture and attributes of SoS. 2.2 Interdependencies in SoS. 2.3 What components of SoS are at cybersecurity risk. 2.4 Analysis of security auditing tools	2				4		7	2.5. Security auditing tools and methodology
3. Models for risk management of SoS. 3.1 Risk management framework. 3.2 Risk analysis process. Risk assessment. Risk mitigation. 3.3 Software Risk Management								3.4 Relevance of ICT Risk Assessment Methods.
4. Overview of models for risk analysis Quantitative and Qualitative risk analysis. 4.1. Basic concepts associated with security risk in SoS. 4.2 Classification of the main risk analysis and assessment methodologies.	2						8	4.6 Qualitative risk analysis basics. 4.7. Resilience analysis: quantitative 4.8. Modify topology in order to reduce

4.3 Quantitative vs Qualitative risk analysis. 4.4 Improving Quantitative Risk 4.5 Hybrid risk analysis in SoS.								criticality and vulnerability
5. Integrated safety and security risk assessment methods. 5.1 Security-Aware Hazard Analysis and Risk Assessment (SAHARA). 5.2 Combined Harm Assessment of Safety and Security for Information Systems (CHASSIS). 5.3 Failure-Attack-Countermeasure (FACT) Graph. 5.4 FMVEA. 5.5 Unified Security and Safety Risk Assessment.	2						7	5.6 Extended Fault Tree. 5.7 Extended Component Fault Tree.
6. SoS resilience analysis. 6.1 Essential Resilience Capabilities. 6.2 Classification of SoS resilience based on when and how failures are addressed. 6.3 Resilience assessment methods for cyber systems. 6.4 Frameworks to assessing resilience.	2				4		7	5.5. Security risk analysis of networks using graphs.
7. Risk assessment for testing purposes. 7.1. Risk-based testing of safety-critical or security-critical systems. 7.2. Risk-based test process optimization. 7.3. Risk-based test planning, design, prioritization or selection. 7.4. Model-based approaches to risk-based testing. 7.5. Support of standard-compliant quality assurance by risk-based testing (e.g. IEC 61508, ISO 26262, Common Criteria).	2				4		7	7.6. Application of risk-based testing in an industrial environment. 7.7. Tools and languages for risk-based testing. Static and Dynamic Analysis Tool.

8. Security risk metrics in industry. 8.1. The Process Control Systems (PCS) technical security risk assessment problem. 8.2. Cross-domain information sharing (CDIS), in the context of the Process Control System (PCS) community. 8.3. Cyber Attack Modeling. 8.4. Control Security risk analysis.	2				4			7	8.5. Security risk management methodologies for Process Control Systems (PCS).
9. Network Security Risk Analysis for Process Control Networks 9.1. Infrastructure Decomposition. 9.2. Failure Modes and Effects. 9.3. Process Specification. 9.4. Process Disruption Modes.	2							8	9.5. Construct Attack Scenarios. Network Security Structure.
Iš viso	20				16			72	

Assessment strategy	Weight in %	Deadlines	Assessment criteria
Lecture activity, including fulfilling special self-tasks	10	7,14	<p>85% – 100% Outstanding work, showing a full grasp of all the questions answered.</p> <p>70% – 84% Perfect or near perfect answers to a high proportion of the questions answered. There should be a thorough understanding and appreciation of the material.</p> <p>60% – 69% A very good knowledge of much of the important material, possibly excellent in places, but with a limited account of some significant topics.</p> <p>50% – 59% There should be a good grasp of several important topics, but with only a limited understanding or ability in places. There may be significant omissions.</p> <p>45% – 49% Students will show some relevant knowledge of some of the issues involved, but with a good grasp of only a minority of the material. Some topics may be answered well, but others will be either omitted or incorrect.</p> <p>40% – 44% There should be some work of some merit. There may be a few topics answered partly or there may be scattered or perfunctory knowledge</p>

Course Program

			<p>across a larger range.</p> <p>20% – 39% There should be substantial deficiencies, or no answers, across large parts of the topics set, but with a little relevant and correct material in places.</p> <p>0% – 19% Very little or nothing that is correct and relevant.</p>
Learning in laboratories	30	7,14	<p>85% – 100% An outstanding piece of work, superbly organised and presented, excellent achievement of the objectives, evidence of original thought.</p> <p>70% – 84% Students will show a thorough understanding and appreciation of the material, producing work without significant error or omission. Objectives achieved well. Excellent organisation and presentation.</p> <p>60% – 69% Students will show a clear understanding of the issues involved and the work should be well written and well organised. Good work towards the objectives.</p> <p>The exercise should show evidence that the student has thought about the topic and has not simply reproduced standard solutions or arguments.</p> <p>50% – 59% The work should show evidence that the student has a reasonable understanding of the basic material. There may be some signs of weakness, but overall the grasp of the topic should be sound. The presentation and organisation should be reasonably clear, and the objectives should at least be partially achieved.</p> <p>45% – 49% Students will show some appreciation of the issues involved. The exercise will indicate a basic understanding of the topic, but will not have gone beyond this, and there may well be signs of confusion about more complex material. There should be fair work towards the laboratory work objectives.</p> <p>40% – 44% There should be some work towards the laboratory work objectives, but significant issues are likely to be neglected, and there will be little or no appreciation of the complexity of the problem.</p> <p>20% – 39% The work may contain some correct and relevant material, but most issues are neglected or are covered incorrectly. There should be some signs of</p>

Course Program

			appreciation of the laboratory work requirements. 0% – 19% Very little or nothing that is correct and relevant and no real appreciation of the laboratory work requirements.
Module Evaluation Quest	60	8,16	The score corresponds to the percentage of correct answers to the test questions

Author	Year of issue	Title	No of periodical or volume	Place of printing. Printing house or internet link
Compulsory literature				
K. Jones	2014	Management of Information Security and Risk: Full 2014 programme specification		City University London, UK http://www.city.ac.uk/_data/assets/pdf_file/0005/178691/PSINS-R-MSc-Management-of-Information-Security-and-Risk.pdf
A. Behnia, R. A. Rashid, J. A. Chaudhry	2012	Survey of Information Security Risk Analysis Methods	vol. 2, no. 1.	Smart Computing Review
K. Bingham, P. Goteti	2004	Integrating HAZOP and SIL/LOPA Analysis: Best Practice Recommendations		http://www.keesofferman.nl/calculators/Integrating_Hazop_and_SIL.pdf
R. E. Bloomfield, K. Netkachova, R. Stroud.	2013	Security-Informed Safety: If it's not secure, it's not safe.	5th International Workshop on Software Engineering for Resilient Systems	Kiev, Ukraine. http://openaccess.city.ac.uk/3097/1/Bloomfield_serene_2013.pdf
G. Giannopoulos, R. Filippini	2012	Risk Assessment and Resilience for Critical Infrastructures		http://www.moi.gov.cy/moi/cd/cd.nsf/5D9E4DBCF6DBB062C2257A3000294D18/\$file/RISK%20ASSESS

Course Program

				MENT%20AND%20RESILIENCE%20PROCEEDINGS.pdf
W. G. Gulland	2004	Methods of Determining Safety Integrity Level (SIL) Requirements – Pros and Cons		http://4-sightconsulting.co.uk/Current_Papers/Determining_SILs/Methods_of_Determining_Safety_Integrity_Level.pdf
A. Ekelhart, S. Fenz, M. Klemen, E. Weippl	2007	Security Ontologies: Improving Quantitative Risk Analysis		https://www.sba-research.org/wp-content/uploads/publications/2007%20-%20Ekelhart%20-%20Security%20Ontologies%20Improving%20Quantitative%20Risk%20Analysis.pdf
Y. Y. Haimes	2008	Models for risk management of systems of systems	Vol. 1, Nos. 1/2	Int. J. System of Systems Engineering
O. Netkachov, P. Popov, K. Salako	2014	Quantification of the impact of cyber attack in critical infrastructures.	pp. 316-327	International Conference on Computer Safety, Reliability, and Security.
T. L. Norman	2010	Risk analysis and security countermeasure selection		CRC Press, Taylor & Francis Group
P. Kertzner, J. Watters, D. Bodeau, A. Hahn	2008	Process Control System Security Technical Risk Assessment Methodology & Technical Implementation	Research Report no. 13	http://www.thei3p.org/docs/publications/ResearchReport13.pdf
T. R. Peltier	2005	Information security risk analysis		CRC Press, Taylor & Francis Group

P.T. Popov, L. Strigini	2010	Assessing Asymmetric Fault- Tolerant Software	doi: 10.1109/I SSRE.2010.1 0	http://openaccess.city.ac.uk/277/
T. Tagarev, V. Georgiev, P. Ivanova	2012	Analytical Support to Critical Infrastructure Protection Policy and Investment Decision- Making	Vol. 28(1). – 2012. – pp. 13-48.	Information & Security. An International Journal
M. Zhivich, T. Leek, G. Baker, R. Lippmann, R. Cunningham	2009	Improving Software Security and Robustness using Automated Testing		U.S. Department of Homeland Security, the I3P – http://www.thei3p.org/docs/publications/90.pdf
Additional literature				
О.В. Поморова, Д.Н. Медзатый, Д.А. Иванчишин	2013	Safety Case методы и средства получения и анализа данных		Х. : Нац. Аэрокосм. Ун-т. Им. Н.Е. Жуковского «ХАИ»
M. Samner	2000	Risk factors in enterprise-wide/ERP projects	Vol.15	Journal of Information Technology
Ernst & Young's IT Risk Management Survey, EMEIA FSO IT Risk Management Survey	2013	Managing IT risk in a fast-changing environment		http://www.ey.com/Publication/vwLUAssets/Managing_IT_risk_in_a_fast_changing_environment/\$FILE/IT_Risk_Management_Survey.pdf
E. Zio, E. Ferrario	2013	A framework for the system-of-systems analysis of the risk for a safety-critical plant exposed to external events	Vol.114 doi: 10.1016/j. ress.2013.01. 005.	Reliability Engineering and System Safety

ABSTRACT

UDK 004.7.056.5:005.334](076)=111

Skarga-Bandurova I.S., Velykzhanin A.Y., Kovalenko Y.P. **Risk Analysis of Infrastructure Security and Resilience. Practicum** / Edited by Kharchenko V. S. – Kyiv: 2017. – 112 p.

ISBN 978-617-7361-23-6

Practical materials of study MSc course “Risk Analysis of Infrastructure Security and Resilience” given in this book are developed within project TEMPUS “Modernization of Postgraduate Studies on Security and Resilience for Human and Industry Related Domains” (543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR).

The course is devoted to the to creating a knowledge base for multidisciplinary research on critical infrastructure risk management and develop a security curriculum of suitable and recognized industry and academic experts as well. In terms of methodology the following were given: syllabus, description of laboratory works and recommendations for independent learning of course.

The book is mainly devoted to MSc students of universities in such fields as computer security, computer and program engineering when studying methods and tools for safety critical systems. It could be useful for lecturers and professors who conduct classes on corresponding courses.

Ref. – 29 items, figures – 43

CONTENTS

ABBREVIATIONS	3
INTRODUCTION	4
Laboratory work 1. Analysis of Security Auditing Tools	6
1.1 Synopsis.....	6
1.2 Brief theoretical information	7
1.3 Execution order and discovery questions	8
1.4 Requirements to the content of the report.....	13
1.5 Test questions	13
1.6 Recommended literature.....	13
Laboratory work 2. Network Security Risk Analysis using Attack Graphs Approach.....	14
2.1 Synopsis.....	15
2.2 Brief theoretical information	15
2.2.1 General information about MulVAL.....	16
2.2.2 Modeling for network topology analysis.....	22
2.3 An example of MulVAL code for multi-host attack.....	25
2.4 Execution order and discovery questions	30
2.5 Requirements to the content of the report.....	31
2.6 Test questions	31
2.7 Recommended literature.....	31
2.8 Assignments to laboratory work 2.....	33
Laboratory work 3. Cyber Threats Risks Modeling with Bayesian Networks	36
3.1 Synopsis.....	37
3.2 Brief theoretical information	37
3.2.1 How to use BN to represent cyber security probabilistic metrics.....	38
3.2.2 Initial information about AgenaRisk tool.....	40
3.2.3 Creating new model in AgenaRisk environment	41
3.3 Execution order and discovery questions	45
3.4 Requirements to the content of the report.....	50
3.5 Test questions	50
3.6 Recommended literature.....	50

3.7 Assignments to laboratory work 3	52
--	----

Laboratory work 4. Assessing Risks and Opportunities in Enterprise Architecture	55
4.1 Synopsis	56
4.2 Brief theoretical information	56
4.2.1 General information about EAAT Object modeler	56
4.2.2 General information about CySeMoL	57
4.2.3 Setting up modeling environments	59
4.3 Modeling and simulating with CySeMoL	60
4.4 Execution order and discovery questions	65
4.5 Requirements to the content of the report	66
4.6 Test questions	66
4.7 Recommended literature	66
4.8 Assignments to laboratory work 4	67

Laboratory work 5. Risk-Driven Testing the Program Source Code	71
5.1 Synopsis	71
5.2 Brief theoretical information	72
5.2.1 General information about software testing in a railway software development lifecycle	72
5.2.2 Brief information about testing environment	74
5.3 Execution order and discovery questions	77
5.4 Requirements to the content of the report	77
5.5 Test questions	77
5.6 Recommended literature	78
5.7 Assignments to laboratory work 5	78

PROGRAM FOR THE SEMINARS	85
---------------------------------------	----

Appendix A. Short installation manual for MulVAL attack paths engine	86
---	----

Appendix B. Attack graphs generating from Nessus reports	88
---	----

Appendix C. Bayesian Network Simulation Tools	91
--	----

Course Program	94
-----------------------------	----

АННОТАЦИЯ

УДК 004.7.056.5:005.334](076)=111

Скарга-Бандурова И.С., Великжанин А.Ю., Коваленко Я.П. **Анализ рисков безопасности и резильентности инфраструктур. Практикум** / Под ред. Харченко В.С. МОН Украины, Восточноукраинский университет им. Владимира Даля, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», 2017. - 112с.

ISBN 978-617-7361-23-6

Изложены материалы практической части учебного курса "Анализ риска безопасности и резильентности инфраструктур" ("Risk Analysis of Infrastructure Security and Resilience"), подготовленного в рамках проекта TEMPUS SEREIN "Modernization of Postgraduate Studies on Security and Resilience for Human and Industry Related Domains" (543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR).

Курс посвящен методологии и практике оценки, анализа рисков и устойчивости масштабных объектов критической инфраструктуры к кибер вмешательствам. Программа практической части курса охватывает широкий круг вопросов от выбора средств аудита безопасности, практики анализа и моделирования рисков безопасности компьютерных сетей, и оценки кибер рисков корпоративных архитектур, до вопросов тестирования критических функций программного обеспечения управляющих систем для минимизации возможных сбоев.

Приводится учебная программа курса, описание лабораторных работ, практикумов и тренингов, методические рекомендации по самостоятельному изучению курса.

Книга предназначена для магистрантов и докторантов университетов, специализирующихся в сфере кибербезопасности, компьютерной и программной инженерии, компьютерных наук при изучении методов и средств оценки рисков и устойчивости критических инфраструктур. Издание может быть полезным для инженеров и преподавателей, которые проводят занятия по соответствующим курсам.

Библ. - 29, рисунков - 43, таблиц - 12.

СОДЕРЖАНИЕ

СПИСОК СОКРАЩЕНИЙ.....	3
ВВЕДЕНИЕ	4
Лабораторная работа 1. АНАЛИЗ ИНСТРУМЕНТОВ АУДИТА БЕЗОПАСНОСТИ	6
1.1 Краткое содержание	6
1.2 Краткая теоретическая информация	7
1.3 Порядок выполнения и вопросы исследования	8
1.4 Требования к содержанию отчета	13
1.5 Контрольные вопросы	13
1.6 Рекомендуемая литература	13
Лабораторная работа 2. АНАЛИЗ РИСКОВ БЕЗОПАСНОСТИ СЕТЕЙ С ИСПОЛЬЗОВАНИЕМ ГРАФОВ АТАК	14
2.1 Краткое содержание.....	15
2.2 Краткая теоретическая информация	15
2.2.1 Общие сведения о MulVAL.....	16
2.2.2 Моделирование для анализа топологии сети	22
2.3 Пример кода MulVAL для атаки с несколькими хостами.....	25
2.4 Порядок выполнения и вопросы исследования	30
2.5 Требования к содержанию отчета	31
2.6 Контрольные вопросы	31
2.7 Рекомендуемая литература	31
2.8. Задания на лабораторную работу 2	33
Лабораторная работа 3. МОДЕЛИРОВАНИЕ РИСКОВ КИБЕРУГРОЗ С ПОМОЩЬЮ СЕТЕЙ БАЙЕСА	36
3.1 Краткое содержание.....	37
3.2 Краткая теоретическая информация	37
3.2.1 Как использовать сети Байеса для представления вероятностных метрик кибербезопасности.....	38
3.2.2 Исходная информация об инструменте AgenaRisk	40
3.2.3 Создание новой модели в среде AgenaRisk	41
3.3 Порядок выполнения и вопросы исследования	45
3.4 Требования к содержанию отчета	50
3.5 Контрольные вопросы	50

3.6 Рекомендуемая литература	50
3.7 Задания на лабораторную работу 3	52
Лабораторная работа 4. ОЦЕНКА РИСКОВ И ВОЗМОЖНОСТЕЙ В АРХИТЕКТУРЕ ПРЕДПРИЯТИЯ	55
4.1 Краткий обзор.....	56
4.2 Краткая теоретическая информация	56
4.2.1 Общие сведения об ЕААТ моделирования объектов	56
4.2.2 Общие сведения о CySeMoL.....	57
4.2.3 Настройка среды моделирования	59
4.3 Моделирование с помощью CySeMoL.....	60
4.4 Порядок выполнения и вопросы исследования	65
4.5 Требования к содержанию отчета	66
4.6 Контрольные вопросы	66
4.7. Рекомендуемая литература	66
4.8 Задания на лабораторную работу 4	67
Лабораторная работа 5. РИСК-ОРИЕНТИРОВАННОЕ ТЕСТИРОВАНИЕ ИСХОДНОГО КОДА ПРОГРАММ.....	71
5.1 Краткий обзор.....	71
5.2 Краткая теоретическая информация	72
5.2.1 Общая информация о тестировании в жизненном цикле разработки программного обеспечения для железных дорог	72
5.2.2 Краткая информация об окружении.....	74
5.3 Порядок выполнения и вопросы исследования	77
5.4 Требования к содержанию отчета	77
5.5 Контрольные вопросы	77
5.6 Рекомендуемая литература	78
5.7. Задания на лабораторную работу 5	78
ПРОГРАММА СЕМИНАРОВ	85
Приложение А. Краткое руководство по установке путей атаки MulVAL	86
Приложение В. Графы атаки, генерируемые отчетами Nessus.....	88
Приложение С. Средства моделирования байесовской сети	91
Программа курса	94

УДК 004.7.056.5:005.334](076)=111

Скарга-Бандурова І.С., Великжанін А.Ю., Коваленко Я.П.
**Аналіз ризиків безпеки та резиль'єнтності інфраструктур.
Практикум** / За ред. В.С. Харченка. – МОН України,
Східноукраїнський університет ім. Володимира Даля, Національний
аерокосмічний університет ім. М.Є. Жуковського «ХАІ», 2017. – 112 с.

ISBN 978-617-7361-23-6

Викладені матеріали практичної частини начального курсу
“**Аналіз ризиків безпеки та резиль'єнтності інфраструктур**” (“Risk
Analysis of Infrastructure Security and Resilience”), підготовленого в
межах проекту TEMPUS SEREIN “Modernization of Postgraduate Studies
on Security and Resilience for Human and Industry Related Domains”
(543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR).

Курс присвячений методології та практиці оцінювання, аналізу
ризиків та резиль'єнтності масштабних об'єктів критичної інфраструктури
до кібервтручань. Програма практичної частини курсу охоплює широке
коло питань від вибору інструментів аудиту безпеки, аналізу та
моделювання ризиків кібер безпеки мереж, та оцінки кібер ризиків
корпоративних архітектур, до питань тестування критичних функцій
програмного забезпечення керуючих систем для мінімізації можливих
збоїв. Приводиться навчальна програма курсу, опис лабораторних
робіт, практикумів та тренінгів, методичні рекомендації щодо
самостійного вивчення курсу.

Книга призначена для магістрів університетів, та фахівців що
спеціалізуються у сфері кібербезпеки, комп'ютерної та програмної
інженерії, комп'ютерних наук при вивченні методів і засобів оцінки ризиків
і стійкості великих складних систем. Видання може бути корисним для
викладачів, які проводять заняття з відповідних курсів.

Бібл. - 29, рисунків – 43, таблиць – 12.

ЗМІСТ

СПИСОК СКОРОЧЕНЬ	3
ВСТУП	4
Лабораторна робота 1. АНАЛІЗ ІНСТРУМЕНТІВ АУДИТУ БЕЗПЕКИ	6
1.1 Короткий зміст	6
1.2 Коротка теоретична інформація	7
1.3 Порядок виконання та питання дослідження	8
1.4 Вимоги до змісту звіту	13
1.5 Контрольні питання	13
1.6 Рекомендована література	13
Лабораторна робота 2. АНАЛІЗ РИЗИКІВ БЕЗПЕКИ МЕРЕЖ З ВИКОРИСТАННЯМ ГРАФІВ АТАК.....	14
2.1 Короткий зміст	15
2.2 Коротка теоретична інформація	15
2.2.1 Загальні відомості про MulVAL	16
2.2.2 Моделювання для аналізу топології мережі.....	22
2.3 Приклад коду MulVAL для атаки з кількома хостами	25
2.4 Порядок виконання та питання дослідження	30
2.5 Вимоги до змісту звіту	31
2.6 Контрольні питання	31
2.7 Рекомендована література	31
2.8. Завдання на лабораторну роботу 2	33
Лабораторна робота 3. МОДЕЛЮВАННЯ РИЗИКІВ КІБЕРЗАГРОЗ ЗА ДОПОМОГОЮ МЕРЕЖ БАЙЄСА.....	36
3.1 Короткий зміст	37
3.2 Коротка теоретична інформація	37
3.2.1 Як використовувати мережі Байєса для подання імовірнісних метрик кібербезпеки	38
3.2.2 Вихідна інформація про інструмент AgenaRisk	40
3.2.3 Створення нової моделі в середовищі AgenaRisk.....	41
3.3 Порядок виконання та питання дослідження	45
3.4 Вимоги до змісту звіту	50
3.5 Контрольні питання	50

3.6 Рекомендована література.....	50
3.7 Завдання на лабораторну роботу 3	52
Лабораторна робота 4. ОЦІНКА РИЗИКІВ В АРХІТЕКТУРІ ПІДПРИЄМСТВА	55
4.1 Короткий огляд.....	56
4.2 Коротка теоретична інформація	56
4.2.1 Загальні відомості про ЕААТ моделювання об'єктів	56
4.2.2 Загальні відомості про CySeMoL	57
4.2.3 Налаштування середовища моделювання	59
4.3 Моделювання за допомогою CySeMoL	60
4.4 Порядок виконання та питання дослідження	65
4.5 Вимоги до змісту звіту	66
4.6 Контрольні питання	66
4.7. Рекомендована література.....	66
4.8 Завдання на лабораторну роботу 4	67
Лабораторна робота 5. РИЗИК-ОРІЄНТОВАНЕ ТЕСТУВАННЯ ВИХІДНОГО КОДУ ПРОГРАМ.....	71
5.1 Короткий огляд.....	71
5.2 Коротка теоретична інформація	72
5.2.1 Загальна інформація про тестування в життєвому циклі розробки програмного забезпечення залізниць	72
5.2.2 Коротка інформація про оточення.....	74
5.3 Порядок виконання та питання дослідження	77
5.4 Вимоги до змісту звіту	77
5.5 Контрольні питання	77
5.6 Рекомендована література.....	78
5.7. Завдання на лабораторну роботу 5	78
ПРОГРАМА СЕМІНАРІВ	85
Додаток А. Посібник зі швидкого встановлення шляхів атаки MulVAL	86
Додаток В. Графи атаки, які генеруються звітами Nessus	88
Додаток С. Засоби моделювання байєсівської мережі	91
Програма курсу	94

Скарга-Бандурова Інна Сергіївна
Великжанін Артем Юрійович
Коваленко Ян Павлович

**АНАЛІЗ РИЗИКІВ БЕЗПЕКИ ТА РЕЗИЛЬЄНТНОСТІ
ІНФРАСТРУКТУР**

Практикум
(англійською мовою)

Редактор Харченко В.С.

Комп'ютерна верстка
Я.П. Коваленко

Зв. план, 2017
Підписаний до друку 20.02.2017
Формат 60х84 1/16. Папір офс. №2. Офс. друк.
Умов. друк. арк. 21,89. Уч.-вид. л. 22,31. Наклад 150 прим.
Замовлення 2/1. Ціна вільна

Національний аерокосмічний університет ім. М. Є. Жуковського
"Харківський авіаційний інститут"
61070, Харків-70, вул. Чкалова, 17
<http://www.khai.edu>

Випускаючий редактор: ФОП Голембовська О.О.
03049, Київ, Повітрофлотський пр-кт, б. 3, к. 32.

Свідцтво про внесення суб'єкта видавничої справи до державного реєстру видавців,
виготовлювачів і розповсюджувачів видавничої продукції
серія ДК № 5120 від 08.06.2016 р.

Видавець: ТОВ «Видавництво Юстон»
01034, м. Київ, вул. О. Гончара, 36-а, тел.: +38 044 360 22 66
www.yuston.com.ua

Свідцтво про внесення суб'єкта видавничої справи до державного реєстру видавців,
виготовлювачів і розповсюджувачів видавничої продукції
серія ДК № 497 від 09.09.2015 р.