The 7th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications
12-14 September 2013, Berlin, Germany

Design and implementation of Virtual Security Appliances (VSA) for SME

Prof. Dr. Kai-Oliver Detken[1], Marcel Jahnke[2], Henk Birkholz[3], Christoph Dwertmann[4]
[1,2] DECOIT GmbH, Fahrenheitstraße 9, D-28359 Bremen, www.decoit.de
[3] Fraunhofer SIT, Rheinstraße 75, D-64295 Darmstadt, henk.birkholz@sit.fraunhofer.de, www.fraunhofer.de
[4] National ICT Australia, Locked Bag 9013, Alexandria NSW 1435, Australia, www.nicta.com.au

The deployment of new security technologies in existing network topologies requires exhaustive testing before usage to avoid down time of the production systems. Nowadays, the required testing is in many cases omitted due to the complexity of creating test cases and experimental set ups. The VISA (Virtual IT Security Architectures) project [1], funded by the German Federal Ministry of Research, aims to provide a simulation environment for semi-automated deployment of experiments based on system models. Building Virtualised Security Appliances (VSA) for enterprise networks is the most important part of the VISA project, which offers the user the possibility to build and deploy secured virtual machines and services within the model to improve it. The developed VSAs are based on mobile scenarios to establish secure connections from an Android smartphone to an enterprise IT infrastructure as well as a meta-data client/server system to establish a higher security level for existing infrastructures. The approach of VISA is therefore to implement such complex security systems easily within the basic environment of small and medium enterprises (SME). This paper is intended to describe the final results of the project, before the analysis phase has been started.

Keywords: virtualisation, security appliances, simulation, emulation, testing, deployment, automatic configuration

I. INTRODUCTION

In small and medium enterprises (SME) IT infrastructures have already become ubiquitous. Aside from the various types of machines (desktop computers, laptops, servers, etc.), peripherals (i.e. multifunctional printers) and functional network components (routers, switches, etc.), the complexity is continuing to grow as a result of different security devices (firewalls, intrusion detection, etc.). The effects of changes to such infrastructures are often hard to predict and can only be observed after the changes have been implemented. An integration of new security components often requires the installation of new hardware and some network topology design changes, which may have to be implemented without a clear idea on what the effects may be on the business operation.

Since SME usually only have limited human resources and know-how for operative IT management, their IT infrastructure must be simplified. This can be achieved through IT infrastructure virtualization. Therefore, the goal of VISA is to simplify and support the management of IT infrastructures, especially security components, by using virtualization technology.

This support is based on two core technologies:

a. Simulation and evaluation of IT infrastructures in virtual realms, and
b. Realization of security applications as virtual components, so-called virtual security appliances (VSAs).

Throughout the VISA framework, the tailor-made, simplified use of security applications based on VSAs will become feasible. By entirely emulating the IT infrastructure, all parameters relevant to the business as well as the VSAs' integration points can be identified and their use can be tested in the virtual realm. Successful VSAs then can be put to use directly without making changes to the rest of the infrastructure. Combining the modelling and formal description of infrastructures as well as evaluating them in virtual realms by using various defined criteria will enable SME to estimate the costs and characteristics of each IT investment better and keep their security risks low. [2]

II. SIMULATION COMPONENTS

The envisioned simulation system aims to allow tests on system functionality and its availability after a specific new feature has been introduced into a specific model of a productive system. The VISA Simulation Environment (VISA-SE) therefore offers the following components:

a. Topology Editor (TE)
b. Simulation Compiler (SC)
c. Simulation Environment (SE)

The topology editor (TE) brings the possibility to newly create or change existing formal representations (models) of the productive system that is being evaluated. It is a graphical tool that allows the definition of IT assets and their interconnected topology. Additional functionalities such as starting measurement procedures or configuring asset properties in the simulation environment are provided. The formal representation processed by the TE is stored in the Interconnected-asset Ontology – IO [3], which is a part of the simulation compiler (SC).

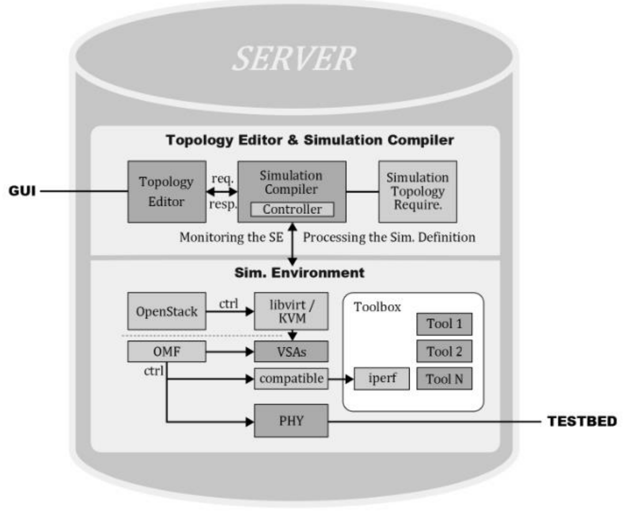

Figure 1. Topology editor and simulation compiler of the VISA simulation environment.

The IO tool-set is capable of acquiring a model from an existing productive system via agent-free automatic acquisition procedures, or it can be modelled manually via the TE. A model representing the current state of the infrastructure can then be modified according to simulation requirements (simulation description). IO is able to store different variants of an infrastructure domain to enable the evaluation of changes to the infrastructure via simulation. Multiple models of the same infrastructure domain can also be used as snapshots to visualize changes in the interconnected topology of productive IT-assets. While IO stores formal representations in OWL/XML format [4], TE and IO transfer models via the less complex RDF/XML format [5]. Both formats are based on triples and can be used to represent graph-like topologies.

In general terms, the simulation compiler (SC) allows the translation of the formal representation stored in IO combined with an additional set of simulation parameters (simulation description) into a simulation definition. Such a simulation definition may include specifics on data sources for automated experiments, or configuration data for virtual images as well as just requests to monitor the simulation. A part of the simulation definition serves as input for the OMF framework [9] that is used by the VISA project scope for some of the simulation environment deployments and services. The SC passes the simulation definition to a controller to execute all actions by the given simulation definition (which may include starting OMF). The controller delivers the results back to the GUI. *Note: This controller is defined to be part of the SC's definition.*

The simulation environment (SE) allows the execution of an OMF-controlled simulation and also the SC's controller direct access to SE, processing and executing parts of the simulation definition as well to offer more generic (OMF independent) scenarios. The simulation also allows measurements on the system's functionality, deriving a verdict on whether the intended security impact has been achieved. Simulations are based upon virtual machine images. More on OMF can be found in section VI.

The intended workflow between the three components starts with the TE. The user here defines or modifies the formal representation of the simulation topology. Based on a set of existing virtualised machines, additional IT-assets can be added to the model of the productive infrastructure. Within the model represented in the TE it is also possible to define certain behaviours of the IT-assets involved. For example, in the case of a mail server, a specific simulation description could define a data source with an automated process sending e-mails to the simulated server.

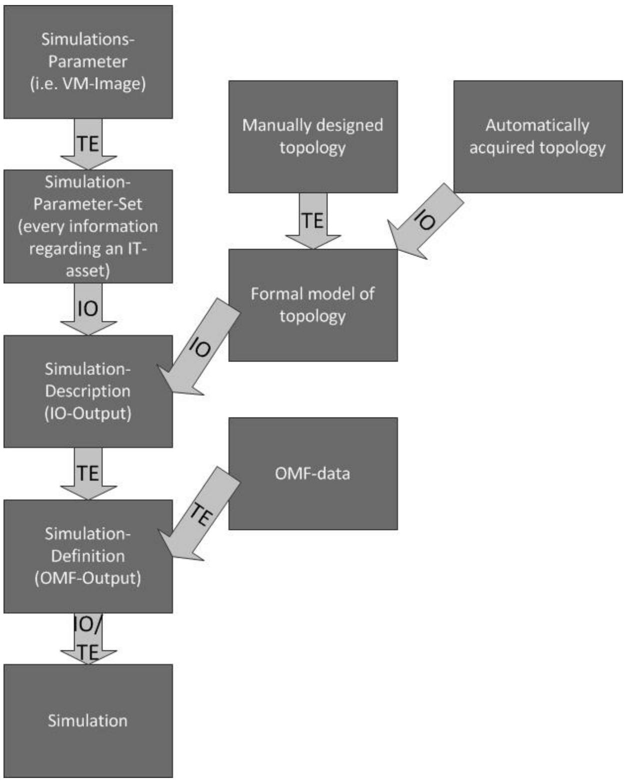
Simulations-Parameter (i.e. VM-Image)
TE
Simulation-Parameter-Set (every information regarding an IT-asset)
IO
Simulation-Description (IO-Output)
TE
Simulation-Definition (OMF-Output)
IO/TE
Simulation
Manually designed topology
Automatically acquired topology
TE
IO
Formal model of topology
IO
OMF-data
TE

Figure 2. Workflow of VISA architecture.

In the next step, the simulation compiler translates this simulation description into a specific simulation definition. The definition contains specific information about the actual deployment of virtual machines on the simulation hardware, the configuration of these virtual machines and their interconnectivity. In later steps, virtual machine deployments in a simulation environment may be further optimized, allowing for the incorporation of more complex scenarios.

In the final step, the simulation will be started based upon the simulation definition. The simulation is comprised of the steps of downloading the images, configuring the images, establishing the intended network links and finally running the network and measuring relevant data on the operation [2]. Figure 2 describes the architecture workflow and the interaction between the topology editor (TE) and the Interconnected-asset Ontology (IO) tool.

In the next chapters, the components topology editor (TE), the simulation compiler/environment, the handling of two VSA and the experiment and measurement conduction are described more in detail.

III. TOPOLOGY EDITOR

The topology editor (TE) provides the primary user interface (UI) via an http-based client/server architecture. Figure 3 presents the web interface that enables the user to delete, replace or add new IT assets or predefined VSAs in a topological layout. IT-assets are visualized as black boxes.

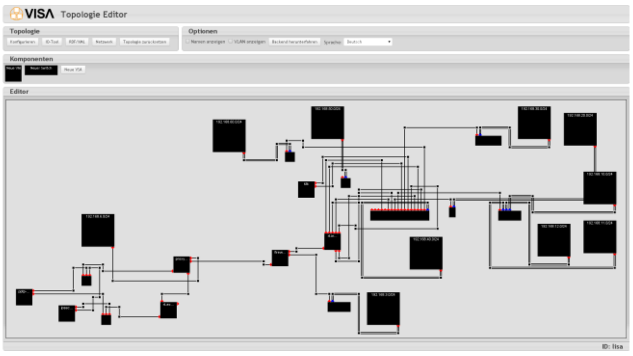

Figure 3. Web interface of the topology editor

It is also possible to setup the network definitions manually. Therefore a "dummy" VM with a Linux operating system can be used to establish a realistic enterprise network.

The following visual options within the TE are available for the "black boxes" after IT-assets have been added into the infrastructure via the TE:

a. Hostname
b. Width/height
c. Number of network interfaces
d. Location of the network interfaces on the visualized IT-asset box

To store the network components, the TE uses a Resource Description Framework (RDF)-based data model. The RDF data model doesn’t have a class structure like an object-oriented programming language. Figure 4 shows the structure and which information is stored in this data model in a similar manner.

The classes with marker "C" represent non-anonymous resources. Their attributes are statements of the model. Classes with marker "E" represent a fixed value range for the literals. This data model is also used for the communication with the SC.

For transmitting the data between SC and TE an additional protocol has been developed within the VISA project. The protocol is based on serialized objects and uses an SSL socket. All string values have been encoded as Base64 to prevent failures while parsing the serialized objects. The following additional instruction set has been implemented in the protocol:

a. Get topology-list (get_topo_list)
   Arguments: None
   Data: Every available topology with its ID
b. Get topology (get_topo)
   Arguments: topology-ID
   Data: The topology ID and the infrastructure data of the newest version as RDF/XML
c. Write topology (write_topo)
   Arguments: Topology ID, the new infrastructure data as RDF/XML and an optional description
   Data: None
d. Collect topology (collect)
   Arguments: Topology ID
   Data: None
   Collects the topology configured in the SC and stores it under the provided ID
e. Replicate topology (replicate)
   Arguments: Topology ID
   Data: None
f. Clean up replication environment (cleanup)
   Arguments: None
   Data: None
g. Reset SC (reset)
   Arguments: None
   Data: None
   Calls ‘cleanup’ and drops all stored topologies
h. Drop topology (drop)
   Arguments: Topology ID
   Data: None

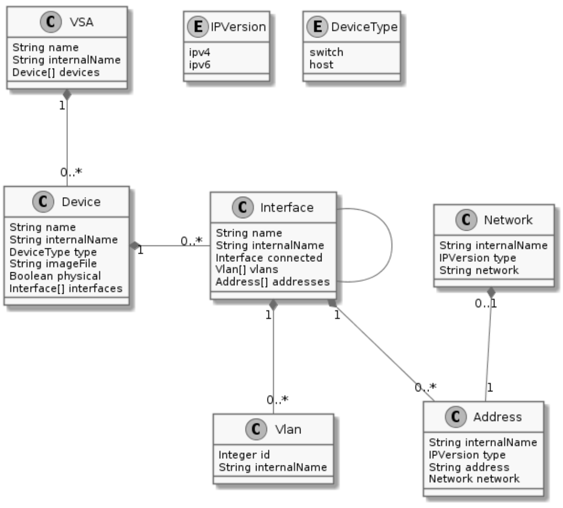

Figure 4. UML model of the data structure

The TE now supports adding complete VSAs to the topology by using templates. These templates consist of a RDF/XML file containing the infrastructure data of the VSA and an additional XML file for defining additional information, e.g. required connections to the existing topology.

To add a new VSA, the user selects the template by using the web interface and defined the devices which will be connected to the corresponding VSA components. If the target device is a switch, the user may select a VLAN for the connection. Finally the infrastructure data of the VSA has to be imported into the topology and the defined connections will be established.

Furthermore, it is possible to execute predefined network tests. For example:

a. Availability tests (e.g. redundancy checks)
b. Load tests (e.g. packet loss, incomplete packets)
c. Stress tests (e.g. DDoS attacks)

Tests are performed automatically by the cOntrol and Management Framework (OMF) [9]. Required measurement points and other information about the test cases are defined in the TE. After the information is transferred, the tests can be started. The results can be displayed using OMF utilities.

In a future version of the topology editor it will be possible to create VMs on an OpenStack cloud directly without the use of the Interconnected-asset Ontology (IO) tool.

IV. IO TOOL-SET

The Interconnected-Asset Ontology (IO) tool-set aggregates heterogeneous infrastructure information and meta-data in an ontological representation with a high level of detail. This includes static IT-asset information, such as manually assigned address configuration, vendor IDs, serial numbers or software versions, and volatile information, such as dynamic address configuration, neighbourhood relationships or MVRP-state. In essence, all data that can be extracted from managed network components (e.g., network equipment or network endpoints) can be processed by the IO tool-set. For this task, IO is composed of four types of modules that provide interfaces for acquisition, storage, search and retrieval procedures. Information is acquired from producers of information via modules supporting protocols such as SSH, SNMP, IF-MAP or SOAP and is made available to consumers of information through customizable query modules supporting, e.g. SCAP-AI, IIOX (the Inter-IO-exchange protocol, utilizing RDF or OWL), CSV, or SQL.

Figure 5 represents two corresponding data flows regarding the IO tool-set: 1) Acquisition of IT-asset information from productive IT-infrastructure and utilization of the resulting formal representation by the Simulation Compiler (SC). 2) Processing of a manually composed infrastructure topology saved and loaded by the Topology Editor (TE) via IIOX. IIOX functions as a platform independent exchange protocol that is based on RDF (optionally OWL if more complexity is required by consumers or producers of information).

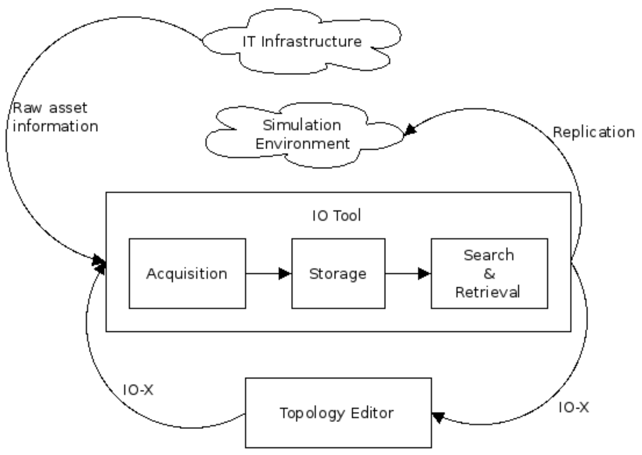

Figure 5. Data flows regarding the IO tool-set

IV. VIRTUAL SECURITY APPLIANCES

A virtual security appliance (VSA) can be a single virtual machine (VM) or a combination of multiple VMs. A VSA is able to offer several services within IT infrastructures, especially for IT security.

The VSA of VISA consists of virtual IT security modules and services. The goal is to improve IT security of a typical SME network topology. In this chapter, two VSA examples are described shortly: VSA-MAC (Virtual Security Appliance – Meta-data Access Control) and VSA-SRA (Virtual Security Appliance – Secure Remote Access).

The *VSA-MAC* is based on the components IF-MAP server and IF-MAP clients for Android, Snort, iptables and Nagios. IF-MAP is an open and vendor-independent client/server protocol to exchange meta-data. The central component is the IF-MAP-server, which stores the collected meta-data from the clients and also provides the data for the clients. With the help of the collected meta-data, anomalies can be detected easily through correlation of all information. [8]

An important specification of the VSA-MAC is IF-MAP [6], a protocol of the Trusted Computing Group (TCG) for exchanging meta-data in a client/server based environment. Its main purpose is to achieve interoperability for security-related data exchange between components in a network. So-called MAP clients (MAPC) can publish new meta-data to a MAP server (MAPS) and also search for meta-data. They can also subscribe to specific meta-data and provide notifications when new meta-data is published.

The specification in its current version 2.0 is separated into several documents. The basic communication protocol based on SOAP is specified in [6], and meta-data definitions for network security are defined in [7]. Thus, new meta-data definitions for non-security environments can be specified without changing the specification of the underlying protocol.

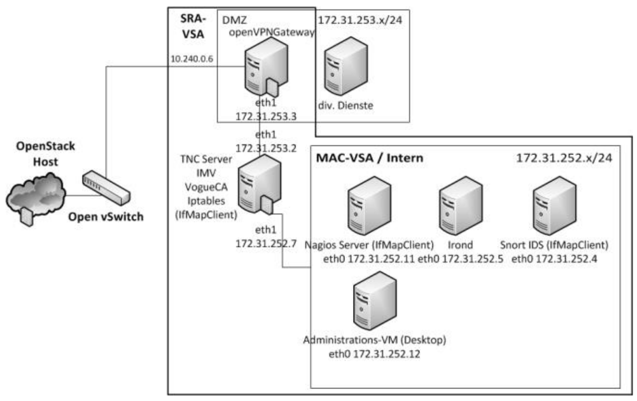

Figure 6. Topology of both VSAs

The *VSA-SRA* allows a secure dial-in to a SME network via an Android-based mobile phone. The VSA contains an Android client, TNC server, and VPN gateway. The mobile phone connects to the SME-network through the VPN gateway. But the mobile phone isn't trustworthy, because only the user credentials have been used and the software and hardware haven't been checked. Therefore, to reach a higher security level it is necessary to send additional meta-data from the Android mobile phone as well. The metrics include the installed application, version number and policies that are applied to the mobile phone. The TNC server compares the metrics with those in its database. If all policies are fulfilled, the mobile phone is granted access to the internal resources, if not, it is rejected. [4]

The VSA-SRC allows Android-based mobile devices to access different IT systems in a trustworthy manner, e.g. applications spanning whole supply chains and enterprise networks. By means of the Trusted Computing (TC) technology, the mobile device is checked through login and password requests of the user credentials, while the device soft- and hardware platform is analysed in the background continuously.

In addition to the authentication of the user, the mobile phone platform (hard- and software configuration) is checked according to the enterprise Trusted Network Connect (TNC) requirements. TNC is also a specification of the Trusted Computing Group (TCG) [11]. With the TNC specification, the TCG developed an open and non vendor-specific description for the integrity check of communication endpoints requesting access to a resource (e.g. a network). The TCG's TNC offers also hardware support by means of the trusted platform module (TPM), so that e.g. the accuracy of the platform integrity information used in the network access control process is guaranteed. Deployed inside desktop PCs and notebooks, this integrated chip protects data on a hardware level. Together with 802.1X, the TNC architecture setups a higher trustworthy platform, so that solely certified (digitally signed) application software may be used. Furthermore, this technology uses an authorization token (e.g. a X.509 certificate), which is transmitted together with the client status information. These are being validated at the target system for conforming to the policy. Access management relies on client identity and system status.

Figure 6 shows both described VSAs in one topology. Both VSAs work together within one OpenStack server. Through this platform, a very flexible topology can be created and configured to handle different security requirements for SMEs. To set up such a complex architecture, automated configuration mechanisms are necessary (see next chapter).

V. AUTOMATIC CONFIGURATION

The presented VSAs will be delivered into an existing IT infrastructure with a basic configuration. Therefore, an automatic configuration of the VSA components is necessary. However not all configuration parameters are available when the VSA will be established, such as:

a. IP address of the default gateway
b. IP address of the irond server
c. IP address of the OpenVPN server
d. Nagios server needs IP addresses of the monitored virtual machines

Additionally, some services have to be started manually if the configuration changes. To avoid this, the VISA project uses the configuration management tool *puppet* (http://puppetlabs.com). This management tool based on Ruby and mainly developed to manage UNIX Systems. Puppet is able to manage different configurations of several computers. The state of a system can be described using templates, which include e.g. services, packets, files or command line instructions.

Within VISA a client-/server model is used. This includes a puppet server on the OpenStack host and a puppet client on every virtual machine (VM) instantiated from the VSAs. To reserve an IP address range in real-time, an additional script was written and is used on the OpenStack host. These IP addresses can be used for the puppet templates. Next, the new default gateways will be set and the IF-MAP clients and the server start. The gateway VM activates the IP forwarding and loads the iptables policies. Also the TNC server and client start up. A detailed workflow of the automatic configuration process is shown in figure 7.

When a VSA has been started, the VMs try to reach the puppet server and ask it for the configuration. The VMs are identified by their fully qualified domain names (FQDN). If a configuration for a client exists, the puppet server transfers the necessary content (files, commands, etc.) to the client. The puppet client compares the existing configuration with the required one. If there is a difference, the new configuration is applied. The puppet clients are configured to ask the server for the configuration every five minutes. The communication between the puppet server and the clients is encrypted with SSL.

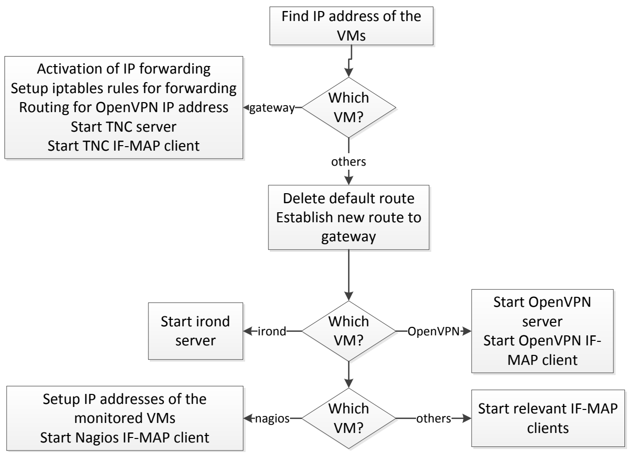
Find IP address of the VMs
Activation of IP forwarding
Setup iptables rules for forwarding
Routing for OpenVPN IP address
Start TNC server
Start TNC IF-MAP client
gateway
Which VM?
others
Delete default route
Establish new route to gateway
Start irond server
irond
Which VM?
OpenVPN
Start OpenVPN server
Start OpenVPN IF-MAP client
Setup IP addresses of the monitored VMs
Start Nagios IF-MAP client
nagios
Which VM?
others
Start relevant IF-MAP clients

Figure 7. Workflow of an automatic configuration

VI. ORCHESTRATION & MEASUREMENT

VISA uses the OMF Framework [9] to control the flow of experiments inside the VSA. Each VSA runs an OMF Resource Controller (RC), which allows a remote experimenter to execute instructions, e.g. starting programs or setting up measurements. The experimenter's tool is the OMF Experiment Controller (EC), which steers the experiment through an experiment description file. The file format is OEDL (OMF Experiment Description Language [9]), which is an output of the VISA simulation compiler.

All OMF components communicate via XMPP protocol. One benefit here is that OMF entities can be in different networks and behind NAT or firewalls, but can still communicate as long as they can all reach an XMPP server.

To measure the impact of an experiment on the VMs and network components, VISA uses the OMF measurement library (OML) [10]. This C library can be linked to existing programs where measurement points have been defined in the code. OML transports these runtime measurements to an OML server, where data from all experiment entities is stored in a database. A library of existing applications, tutorials and language bindings are available on the OML website (*http://oml.mytestbed.net*).

A common scenario in VISA is to deploy a couple of VSA that mimic a SME and then introduce an “attacker” VSA to the network. OMF is used to start up some applications (e.g. a mail server) and shortly after run the attack (e.g. a flood of spam mails to drown the server). Additionally, OMF can start some OML enabled measurement tools to monitor network throughput (e.g. *iperf*) and system load (e.g. *nmetrics*) during the attack. After analysing the data and securing the VSAs, the exact same attack can be repeated by OMF and measured by OML to check whether the security measures were successful. OMF & OML provide a range of live visualization options for the measurements, or they can simply be interpreted in raw format.

VII. CONCLUSION

The goal of VISA is to establish more security mechanisms in SME infrastructures. That also means to make the configuration itself more secure, which is a difficult task in general. Through the VISA project it is now possible to setup a virtual IT infrastructure with different security components and automatic configuration mechanisms without extensive knowledge about the individual VMs. Additionally, it is feasible to simulate the infrastructure and the configuration of the VMs and analyse the security afterwards (by using the visualization tools of OMF). Using the topology editor, an administrator can design and re-design the IT infrastructure in a simple-to-use graphical tool. Another feature is to analyse the existing infrastructure and simulate it on this virtual platform. Thus a complete simulation cycle can be established, which helps to institute more security in SME environments.

ACKNOWLEDGEMENT

The VISA project [1] has been funded by the Federal Ministry of Education and Research (BMBF) of Germany. The project started in August 2011 and ended in July 2013. The authors would like to thank the BMBF for their support. We also wish to express our gratitude and appreciation to all VISA partners for their strong support and valuable contributions during the various developments presented in this paper.

REFERENCES

[1] VISA Project: http://www.visa-project.de

[2] Detken, Oberle, Kuntze, Eren: *Simulation Environment (SE) for mobile Virtualized Security Appliances (VSA)*. 1st IEEE International Symposium on Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems, 20.-21. September, University of Applied Sciences Offenburg, Offenburg 2012

[3] H. Birkholz, I. Sieverdingbeck, K. Sohr, C. Bormann: IO: *An interconnected asset ontology in support of risk management processes*. ARES 2012 Security Ontology workshop

[4] W3C, *“OWL 2 Web Ontology Language Document Overview,”* Tech.Rep., 2009

[5] D. Beckett and B. McBride, *RDF/XML syntax specification (revised)*. W3C recommendation, vol. 10, 2004.

[6] TCG Trusted Network Connect: *TNC IF-MAP Binding for SOAP*. Specification V. 2.1, rev 15, Mai 2012.

[7] Trusted Computing Group, *TNC IF-MAP Meta-data for Network Security*, Version 1, Revision 25, 2010.

[8] Detken, Eren, Steiner: *Konfigurationsunterstützung bei der Virtualisierung*. D A CH Security 2012 ISBN: 978-3-00-039221-4

[9] Thierry Rakotoarivelo, Max Ott, Guillaume Jourjon, Ivan Seskar, *OMF: a control and management framework for networking testbeds*. ACM SIGOPS Operating Systems Review 43 (4), 54-59, Jan. 2010

[10] Jolyon White, Guillaume Jourjon, Thierry Rakotoarivelo, and Max Ott, *Measurement architectures for network experiments with disconnected mobile nodes*. TridentCom 2010, May 2010

[11] Trusted Computing Group, *TCG Specification Architecture Overview*, Revision 1.4, August 2007