The 1st IEEE International Symposium on Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems
20-21 September 2012, Offenburg, Germany

Simulation Environment (SE) for mobile Virtualized Security Appliances (VSA)

Prof. Dr. Kai-Oliver Detken[1], Alexander Oberle[2], Nicolai Kuntze[3], Prof. Dr. Evren Eren[4]
[1] DECOIT GmbH, Fahrenheitstr. 9, D-28359 Bremen, detken@decoit.de, www.decoit.de
[2,3] Fraunhofer SIT, D-64295 Darmstadt, alexander.oberle/nicolai.kuntze@sit.fraunhofer.de
[4] University of Applied Sciences Dortmund, D-44227 Dortmund, evren.eren@fh-dortmund.de

The deployment of new security technologies in existing network topologies requires exhaustive testing before usage to avoid down time of productive systems. Nowadays, the required testing is omitted in many cases due to the complexity of creating test cases and experimental set ups. The VISA (Virtual IT Security Architectures) project, funded by the German Federal Ministry of Research, aims to provide a simulation environment for semi-automated deployment experiments based on system models. Building Virtualized Security Appliances (VSA) is one part of the VISA project which offers the user the possibility to build and deploy secured virtual machine(s) and services within the model to improve it. Two of the developed VSAs are based on mobile scenarios to establish secure connections from an Android smartphone to an enterprise IT infrastructure. The approach of VISA is to implement such complex security systems much more easily within the basic configuration settings for small and medium enterprises (SME).

***Keywords:** virtualization, virtual security appliances, simulation, emulation, testing, deployment, mobile security.*

I. INTRODUCTION

In small and medium enterprises (SME) IT infrastructures have already become complex. Aside from the diversity (desktop computers, laptops, servers, etc.), peripheral (i.e. multi-functional printers) and functional network components (routers, switches, etc.), the complexity is growing as a result of different security mechanisms such as firewalling, intrusion detection and prevention. Often the effects of changes in infrastructures can be seen after accomplishing the modifications. The integration of new security components often requires new hardware and network topology restructuring, without a clear idea of what impact it will have.

Since SMEs can provide only limited personnel resources and know-how for operative IT management, IT management has to be easy. This can be enabled by means of IT infrastructure virtualization. Therefore, the goal of *VISA* is to simplify and support management of IT infrastructures, especially security components, by using virtualization.

This support is based on two core technologies: simulation and evaluation of IT infrastructures in virtual realms, and realization of security applications as virtual components, so-called virtual security appliances (VSAs). The VISA framework facilitates tailor-made and simplified usage of security components based on VSAs, which can be directly integrated into an existing IT infrastructure. By combining virtual and real infrastructure components this approach will help SMEs to estimate costs of their IT and enhance security.

II. SIMULATION COMPONENTS

The envisioned simulation system aims at testing system functionality and availability after introducing new features 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)* offers the possibility to develop the model of the productive system to be evaluated. As a graphical tool it defines the topology and offers additional functionalities such as starting measurements or configuring properties for the simulation environment.

The *simulation compiler (SC)* translates the TE parameter set to a simulation definition, e.g., the designed topology or solely required measurements for the topology. Such a simulation definition may include specific data sources for automated experiments, definitions of virtual network interfaces, configuration data for virtual images as well as requests to monitor the simulation. A part of the simulation definition serves as input for the *OMF framework* [1] which is used by the VISA projects' 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 (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)* executes the OMF controlled simulation and allows the SC’s controller to directly access the SE. Furthermore, The SE processes and executes parts of the simulation definition and also offers generic (OMF independent) scenarios. The simulation facilitates measurements with respect to systems functionality. Simulation uses virtual machine images.

The workflow between these three components starts at the TE. Here, the user defines the simulation topology based on a set of existing virtual machines connected with each other. Within this topology it is possible to define specific behaviour of components involved, e.g., for an e-mail server an automated process can send e-mails to the simulated server. As a result a simulation description is created including all the information required to define an automated simulation.

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

In the final step, based upon the simulation definition the simulation is executed, comprising downloading and configuring images, establishing the intended network connections and finally executing the network and measuring relevant operation data.

III. REQUIREMENTS

The respective *VISA-SE* requirements can be structured according to the components presented in the introduction. The VISA system will include the following components in its system model:

a. **Core functionality:** These components are used to model the IT system and incorporate systems such as e-mail servers and clients.
b. **Security functionality:** One of VISA's goals is to evaluate security components with regard to their impact on the productive IT system's core functionality.
c. **Network:** All core and security components are interconnected by networks.
d. **Test data source:** Data sources are required for core and security components (might also include penetration tests) to simulate automated actors. Automated testing requires generator components (e-mail data source that can be linked to an e-mail server).

The *topology editor (TE)* allows the definition of models which represent productive environments. Productive IT systems are composed of servers, clients, and network connections. The TE represents the graphical user interface (GUI) allowing the user to choose, place and connect network components to set up a topology. VMs, defined VSAs and their components/services have to be selectable and combinable as single-operating instances. Test data sources such as e-mail clients can be set by the TE, including points for sources, measurements and penetration testing within the network.

After the user has defined the simulation description it is passed to the SC (validating and creating the final simulation definition). The TE functionality also includes an abstract SE visualization, for example in presenting measurement data and results delivered by the SC. The *simulation compiler (SC)* must be able to validate the topology definition given by the simulation description against defined and secured topology requirements. E.g., if a user defines three VSAs and only two of them are behind a firewall, then the topology rules defined to fulfil the security requirements shall perform intelligent and automated checks whether the third VSA might be a security risk. In a worst case scenario the SC has to warn the TE and its user and offer solutions, if possible. After the first check has been passed the SC must be able to translate the simulation description into a simulation definition. The SC's controller then executes the simulation definition on the SE. In return, the SC also prepares and formats data that has been requested by a given simulation definition and delivered by the SE to be finally passed back to the TE for visualization.

In the *simulation environment (SE)* the different technologies are executed to load VSA virtual machine images, configure the network topology between the nodes, trigger events and collect data. The SC's controller executes the tasks according to the given simulation definition.

The measurements derived from the individual simulation need to proof system liveliness and identify possible impacts on performance metrics such as throughput, latency, and packet loss. Penetration tests and data sources address the verification of security measures. VISA aims at automating typical scenarios in the deployment of the security measures.

IV. ARCHITECTURE

In the following, the *VISA-SE architecture* with the respective basic VISA simulation environment components will be outlined. Figure 1 illustrates the architecture and controlling of the VISA-SE. The topology editor represents the GUI controlling within the simulation. The simulation compiler is responsible for validating the network topology against the defined *simulation topology requirements (STR)* and for translating the simulation description into a simulation definition, which is processed by the SC controller. The controller has access to the SE mainly via libvirt/KVM, OMF, and a set of tools defined as *toolbox* available on the host server.

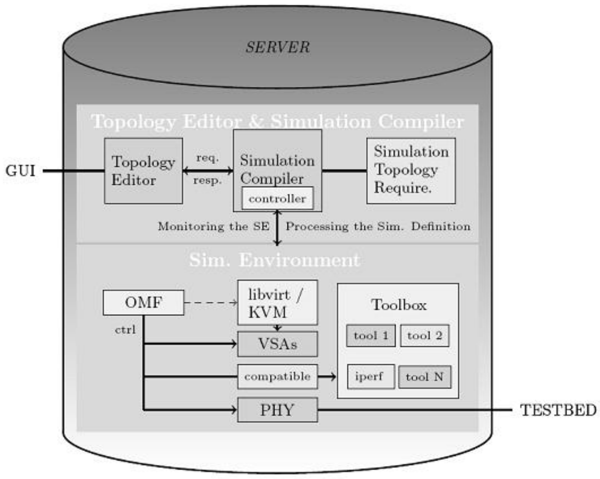

Figure 1. Topology editor and simulation compiler of the VISA simulation environment.

The simulation environment comprises the complete processing and deployment of the simulation components with all VMs, connections, services and measurements. It is controlled by the SC's controller; note that the *OMF* can control the toolbox as well if the service needed from the toolbox is compatible with the OMF framework. A physical test-bed, deployed and controlled by the OMF, with different nodes and services (e.g. WLAN/Mesh, GSM) is available to connect to the virtualized infrastructure.

V. IMPLEMENTATION STRATEGY

In this section the required functional building blocks of the VISA-SE implementation will be discussed. The *model view controller (MVC)* pattern might fit well here for the implementation design and could be chosen to structure the PHP-based implementation realizing the TE and SC/STR.

The *topology editor (TE)* is the basic interface to configure, control and measure the virtualized network. To illustrate what is expected, the GUI should have an editor within a tool bar to design the infrastructure to be addressed. This may be realized by a suitable script language library such as JavaScript. A good example might be a modified version of the WiringEditor non-fullscreen example given by the JavaScript library WireIt [2]. The TE should provide an assistant (wizard) guiding the user through the VSA configuration. This may be realized by php-virt-control [3], for example, which needs to conform to the architecture requirements using a combination of PHP and libvirt to deploy virtual machines. Functionalities such as measurements, test data sources and services should be available on-demand in sub-interfaces to configure them onto the topology or its components. These menus may appear step by step. For example, after the topology has been verified successfully against the STRs (via the SC), the topology within the VSAs and all its components are shown as previously selected. The user can set additional measurement points which have been offered on fixed topology locations. The user has to specify the form of measurement. Then the simulation can be started by triggering previously set timing intervals for the data sources to generate traffic, while at the same time starting the measurements and live network visualization. Once the run has been completed or manually stopped, additional results will be presented (e.g. log files) and a security validation report is produced.

The TE will be realized by a web interface and will comprise several menus and configuration possibilities. A suitable API for generating a dynamic web interface may be XML http request (XHR), which offers the opportunity to request and load TE content asynchronously and can be controlled with standard script languages such as PHP, JavaScript (or Ajax).

The *simulation compiler (SC)* can be seen as a module (or model) of the TE written in PHP; its main focus will be to generate a simulation definition from the given user input, execute spell verifications and also further translations, for example into the experimental OEDL definition of OMF. It must also verify security issues in the context of the given topology specification by the simulation definition. Within this model a controller executes commands directly on the server, thus processing the simulation definition. Therefore, the controller module is always called by the SC after generating the simulation definition. The controller is also responsible for pointing to the returned values which have been executed (e.g. the URLs of gathered and stored data files, etc.) so that the simulation compiler can process and format it to be presented by the TE.

The deployment of the simulation should be realized by PHP and libvirt at least until OMF has a built-in support for instantiating VMs. OMF can already handle parts of the tasks given by the simulation definition to deploy the simulation. Obviously a modified version of the mentioned php-virt-control library and its interface could realize the deployment of the VMs/VSAs by integrating the library into the TE and modifying or creating its web interfaces, meeting the intended functionalities and VISA requirements.

In *OMF*, it is like defining every aspect of an experiment in its "experiment description". This is a script written in OEDL, its own experiment description language. There could be a (web) frontend that allows the assignment of the software components (e.g. select services such as FTP, SMTP etc. from a list) and the network parameters (connected to which bridge/switch, latency, loss, etc.), resulting in a simulation definition in form of a file. Since these files serve as input to OMF the simulation definition may be attached to OEDL. It also supports network emulation in OMF, meaning that the experiment can include parameters such as network loss and delay. At the moment OMF does not have built-in support for instantiating VMs, but maybe this feature will be integrated in a future release. For now, it can be written or generated as a separate OMF experiment that sets up the VMs and their connectivity/topology.

OMFs' built-in commands can configure an image containing a (previously instantiated) Linux installation to install packages and run scripts for configuring the services.

In the topology editor, the user can select certain measurement points and assign measurement tools. The following measurement and penetration testing tools may be integrated and can thus be part of the toolbox: iperf, (h)ping, nmap, mdk3 WLAN testing tool.

NICTA, the Australian partner in the VISA project, has developed OML, a measurement library which may be a useful tool to gather data. The OML library is linked against an existing application (e.g. an e-mail server). Hooks are introduced to the application code, which extract measurement points streamed to an OML server. The measured data are stored in an SQL database and can be visualized on-the-fly in various forms using OMF.

VI. MOBILE VSAs

Mobile devices are increasingly used in companies and integrated in corporate networks. These devices handle and contain security relevant business data. Smartphones operating systems such as iPhone OS, Android, Windows Mobile, RIM OS, etc. achieve capabilities of conventional PCs. Innumerable applications (apps) can be downloaded and installed. Increasing complexity and at the same time higher mobility and this ubiquitous connectivity enlarge the risk of compromising the device with malware or targeted attacks implying risks in terms of security, safety and availability for IT infrastructures.

Therefore within the *VISA project* two mobile VSAs have been defined in order to address improved securing for smartphones, especially for Android OS:

a. **VSA-SRA:** The VSA Secure Remote Access allows Android based mobile devices to access different IT systems in a trustworthy manner, for example applications spanning whole supply chains and enterprise networks. By means of the Trusted Computing (TC) technology the user device is checked by login and password requests, while the device hardware platform is analysed continuously.

b. **VSA-MAC:** The VSA Meta-data Access Control additionally uses the IF-MAP protocol of the Trusted Computing Group (TCG). The IF-MAP specification currently defines a model for meta-data which specifically addresses use-cases in the network security domain. With the correlation of other meta-data, the VSA is able to detect attacks which cannot usually discovered by security systems.

In the following both VSA types are described in detail.

A. *VSA-SRC*

For the first VSA prototype the core element of the platform is represented by the *VPN gateway*. Additionally, a management server (e.g. RADIUS), a directory server (e.g. LDAP), and a certification authority server is necessary. In the first step, the user has to be identified accessing the VPN gateway. All criteria are available on the directory server and assign the user to different profiles and user groups.

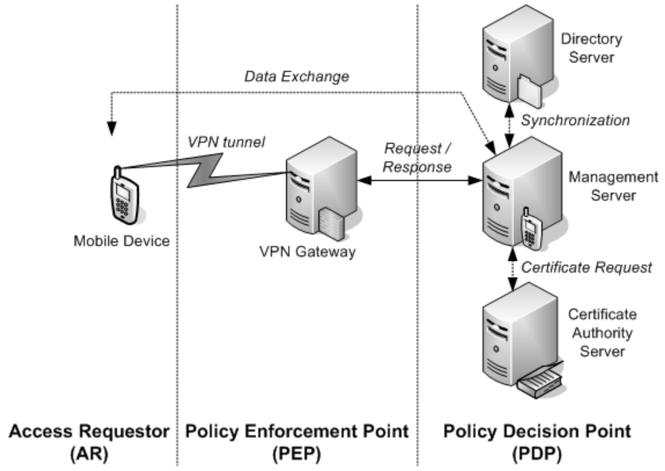

Figure 2. Platform overview of the virtual security appliance – secure remote Access (VSA-SRC).

In companies each user group has different security policies for different access rights. The management system synchronises in intervals continuously user information with the directory server. That includes that user from the directory server with VPN access rights, if they are not yet available on the management server, will synchronize with all user group membership automatically after one interval.

As an option a public certification authority (CA) can be adapted. If a new user is created on the management server, a certificate will be applied. In this case, the management server functions as registration authority. The VPN gateway has to be configured that all requested clients will be authenticated via the management server. Therefore, the gateway site does not need adaptations for a new user. This will be done automatically by the communication with the management server.

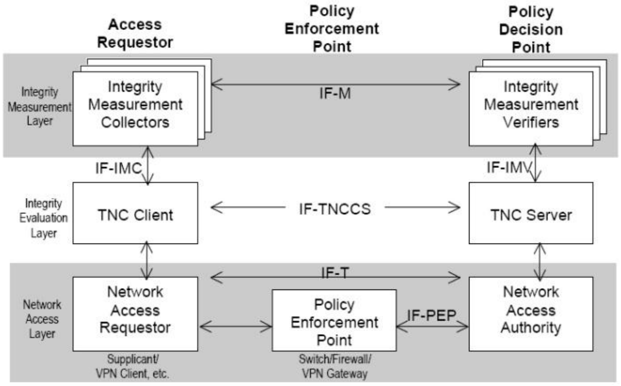

Figure 3. Architecture overview of TNC components. [9]

Next to the authentication of the user, the smartphone platform (hardware and software configuration) is checked according to the enterprise TNC requirements. *TNC* is a specification of the *Trusted Computing Group (TCG)* [8]. With the TNC specification, the TCG developed an open and vendor-neutral specification for the integrity check of communication end-points, requesting access to a resource (e.g. a network). The TCG's TNC offers 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. As built in desktop PCs and notebooks this integrated chip protects data on a hardware level. Together with 802.1X, it guarantees the TNC architecture, so that solely certificated (digitally signed) application software may be used. Furthermore, this technology uses an authorization token (e.g. a X.509 certificate), which is communicated together with the client status information. These are being validated at the target system against policy conformity. Access management relies on client identity and system status.

After the establishment of a VPN connection, the network access of the mobile device is limited to a quarantine zone. Within this area, it is only possible to update software components of the mobile device like anti-virus-software or operating system patches. Access to other network areas of an enterprise network is prohibited. Mobile device status information is available by the access requestor (AR) on the client-site. This entity includes the network requestor (as a component of the VPN client), the TNC client (as an interface between the network access requestor and plug-in software), and the integrity measurement collector (describes the plug-ins which allows different software products like anti-virus software to communicate with TNC).

In detail, the following points will initiate for a mobile device communication (also depicted in figure 2):

1) A VPN connection is established.
2) The management server (TNC server) initializes an integrity check.
3) The mobile device (TNC client) collects integrity measurements (IM) information using the local integrity measurement clients (IMC) on the mobile device.
4) The management server (TNC server) forwards the IM information for a check to the integrity measurement verifier (IMV).
5) The integrity measurement verifier (IMV) checks the IMs and sends the results with a recommendation to the management server (TNC server).
6) The management server (TNC server) takes access decision und forwards this information to the VPN gateway (PEP) and the mobile device (AR).
7) The VPN gateway (PEP) allows or does not allow access to the network for the mobile device (AR).

Summarizing, the integration of the TPM allows a further check of the software components on the mobile device. This simplifies the detection of rootkits. Furthermore it is possible to sign and encrypt messages with key material of the TPM, affecting strong security check of the origin of the information. As a conclusion, the TNC approach within the VSA-SRC is a viable solution to raise the security level in mobile networks. [5]

B. *VSA-MAC*

A further specification of the TCG is *IF-MAP*, a protocol 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 also can subscribe to specific meta-data and provided with information when new meta-data is published.

The specification in its actual 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 environment can be specified without changing the specification for the underlying protocol.

Based upon a threat analysis of another research project [10] the following desirable key features for the *VSA-MAC* solution have been identified:

a. **Anomaly Detection:** Consolidation of meta-data created by different components in order to detect outliers, indicating potential fraud activities. Furthermore, smartphone driven attack patterns like sensory malware approaches will be analysed.
b. **Smartphone Awareness:** Identification of smartphones within the business environment, enabling provisioning services that are specifically tailored or to make policy decisions based upon the smartphone type.
c. **Single Sign Off:** Immediate and global de-provisioning of user accounts, ensuring that revoked credentials cannot be used anymore within the respective environment, no matter which service or device is used.
d. **Secure Evidence:** Generation and integration of evidence records proofing the integrity of meta-data objects within the MAP server, thus increasing the trustworthiness of the IF-MAP data set itself.
e. **Identity Awareness:** Making the user’s authenticated identity available within a business environment beyond the scope of the authenticating entity, thus enabling use-cases like automated, identity-based configuration of low-level security tools (e.g. packet filters).
f. **MalApp Detection:** To defend against the spread of potentially malicious applications and to limit the amount of damage they can cause to the respective business environment. This also implies the development of new means in order to assess the security state of a smartphone including its installed applications and their respective privileges that go beyond well-known approaches like Trusted Computing or application certification.
g. **Location-based Services:** To provision services based upon the smartphone’s location as well as to support detection capabilities (anomaly and MalApp detection components) by providing location information on users and devices.
h. **Real-time Enforcement:** To enable immediate reaction on identified anomalies by any component that can help to mitigate the potential damage (like flow controllers and net-work enforcement points).

In order to realize the key features mentioned above, the *VSA-MAC* approach relies on the concept of trustworthy meta-data correlation based upon the IF-MAP protocol. Any security relevant data, whether it stems from a smartphone or a service that is provided by the IT infrastructure (such as an IDS, a firewall, or an AAA service), is expressed according to a well-defined meta-data model. In addition, a trust model has been defined that enables to reason about the trustworthiness of the meta-data instances. Both the meta-data model and the trust model are based on the IF-MAP protocol.

The meta-data model is the fundamental basis for any further analysis and correlation approaches that are performed. The challenge is to encapsulate both smartphone specific features as well as other meta-data of interest that might be generated by arbitrary services in the network in a common model. In addition, it is also necessary to model the relationships between the relevant meta-data concepts.

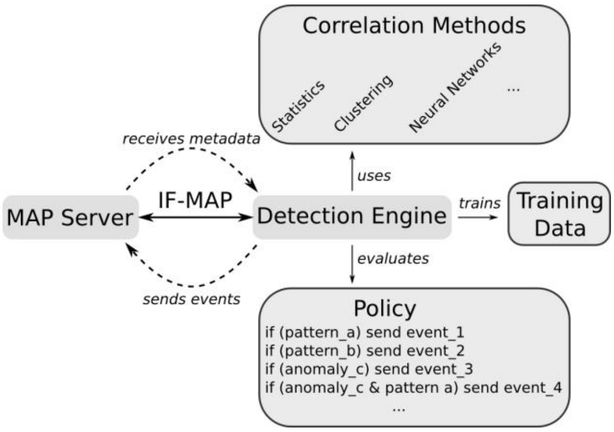

Figure 4. Overview about the architecture of the detection engine.

As mentioned before, the current version of the meta-data model of the VSA-MAC is based upon the *IF-MAP meta-data model*. That includes the basic components as identifiers, meta-data objects, and links. However, smartphone specific features are currently not part of the IF-MAP protocol. In order to realize the described key features, it is necessary to name smartphone features of interest and to integrate those features into the IF-MAP protocol.

The extended meta-data graph forms the basis for any further correlation approaches. Those approaches can now benefit from both, the ability to consider network generated meta-data and smartphone specific meta-data, in addition to the trust tokens that vouch for the trustworthiness of the participating entities.

The currently *correlation approaches*, which the VSA-MAC used are feasible and perform best for the desired key features. The current candidates include rule- and case-based reasoning, neuronal networks, and dependency graphs. But, an evaluation of these approaches is subject of the future work.

By the use of the *detection engine* component, pattern and anomaly detection is possible for the VSA-MAC environment, based on real-time collected meta-data. If the detection engine recognizes a pattern such as a signature or an unregistered app, an event will be sent to the MAP server. This event can be analysed by other MAP clients, which can react (e.g. with a real-time enforcement) according to their policy definitions, if necessary. The detection engine works as MAP client and is able to get any subscriptions from the MAP server directly.

The recognition of anomalies implies a higher complexity. First of all, the normal behaviour has been trained or defined by expert knowledge. For the recognition the detection engine can be supported by many statistical algorithms (average, median, clustering) and automatic learning (neural networks). The evaluation of the different algorithms will be one task of the next steps. [8]

VII. CONCLUSIONS

The VSA-SRA consists of several security components such as *VPN gateway*, *TNC client/server*, *RADIUS server*, *LDAP*, and *PKI server*. The complete VSA-MAC consists of several MAP clients like *DECOIT IF-MAP-Client* (Android, iptables, Snort, Nagios), *macmon*, *NCP*, *irondhcp*, and *irondetect*. Additionally the MAP server *irond* is a central component, which is necessary to collect all MAP data. All components work with each other and have to be configured.

The modelling framework developed in VISA allows IT system security testing at the modular and overall system level more easily and efficiently. This would be an important step in the direction of end-to-end security, which is an essential concept in IT landscapes.

A further advantage of comprehensive IT infrastructure planning via the VISA framework is the tailor-made, simplified use of security applications based on VSA. Through the extensive emulation of IT infrastructures, parameters relevant for businesses as well as VSA integration points can already be identified transparently and tested in use in the virtual realm. VSAs tested in this way can then be implemented without changing the existing infrastructure.

Acknowledgment

The project VISA [4] is funded by the Federal Ministry of Education and Research (BMBF) of Germany. The project started in August 2011 and will end 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 activities presented in this paper.

References

[1] OMF Framework: *http://omf.mytestbed.net*
[2] Javascript Library Wirelt: *http://neyric.github.com/wireit/*
[3] Php-Virt-Control: *http://php-virt-control.org/screenshots.html*
[4] VISA Project: *http://www.visa-project.de*
[5] Detken, Fhom, Sethmann, and Diederich: *Leveraging Trusted Network Connect for Secure Connection of Mobile Devices to Corporate Networks*, Communications: Wireless in Developing Countries and Networks of the Future, IFIP World Computer Congress (WCC) 2010, Ana Pont, Guy Pujolle, S.V. Raghavan (Eds.), Springer-Verlag, Brisbane, Australia 2010
[6] Trusted Computing Group, *TNC IF MAP Binding for SOAP*, Version 2.0, Revision 36, 2010.
[7] Trusted Computing Group, *TNC IF-MAP Meta-data for Network Security*, Version 1, Revision 25, 2010.
[8] Bente, von Helden, Hellmann, Vieweg, Detken: *ESUKOM: Smartphone Security for Enterprise Networks*, Securing Electronic Business Processes, Vieweg+Teubner Verlag, Springer Fachmedien Wiesbaden GmbH, 13. Information Security Solutions Europe Conference (ISSE) conference in Prague, 22.-23. November, Wiesbaden 2011
[9] Trusted Computing Group, *TCG Specification Architecture Overview*, Revision 1.4, August 2007
[10] BMBF research project ESUKOM: *http://www.esukom.de*