HAL open science

Enabling Next-Generation Cyber Ranges with Mobile Security Components

Enrico Russo, Luca Verderame, Alessio Merlo

To cite this version:

Enrico Russo, Luca Verderame, Alessio Merlo. Enabling Next-Generation Cyber Ranges with Mobile Security Components. 32th IFIP International Conference on Testing Software and Systems (ICTSS), Dec 2020, Naples, Italy. pp.150-165, 10.1007/978-3-030-64881-7_10 . hal-03239813

HAL Id: hal-03239813

https://inria.hal.science/hal-03239813

Submitted on 27 May 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Enabling Next-Generation Cyber Ranges with Mobile Security Components

Enrico Russo[1[0000−0002−1077−2771]], Luca Verderame[1[0000−0001−7155−7429]], and Alessio Merlo[1[0000−0002−2272−2376]]

DIBRIS - University of Genoa, Genoa, Italy {name.surname@unige.it}

Abstract. The number of security incidents involving mobile devices has risen in the past years. This means that organizations must seriously consider such devices within their threat landscape and prepare their cybersecurity operators to prevent, identify, and manage security issues involving them. Nowadays, cyber ranges represent the most effective and versatile systems for training skills in the cybersecurity domain as they provide hands-on experiences in large, sophisticated infrastructures. Nevertheless, cyber ranges are capable of emulating components and interactions of the Information and Operational Technology domains, but they lack the support of mobile devices with the same effectiveness and realism. In this paper, we propose an enhancement of the architecture and the training environments of an existing cyber range, which properly integrates mobile security components. The effectiveness of such an integration is due to a thorough review of the mobile threat landscape and the functional components of a cyber range.

Keywords: Cyber Range · Mobile Security · Cybersecurity Training

1 Introduction

Cyber ranges are virtualized environments that allow defining specific training in cybersecurity for a plethora of different students. A cyber range emulates an actual cybersecurity-relevant scenarios (e.g., the ICT deployment of a corporation or a small city) in which teams of several trainees compete to reach different (and conflicting) aims. A typical (and simplified) cyber range deployment involves a simulated ICT environment in which all components of the environment suffer from specific vulnerabilities, properly inserted before the competition start. On one side, a red team is aimed at discovering and exploiting such vulnerabilities, while, on the other side, a blue team aims to discover and fix the same vulnerabilities, before the red one can exploit them.

The granularity of the exercises, as well as the kind of vulnerabilities to play with, depends on the number and kind of emulated ICT infrastructures in the cyber range. A missing brick is the mobile component of the emulated ICT infrastructure. As a consequence, a lot of high impact vulnerabilities and attack patterns cannot be tried in current cyber ranges. In fact, a vulnerable mobile node may act as a Trojan Horse to contact other connected devices, perform

harmful commands or spread malicious software. Furthermore, a compromised mobile device may allow the attacker to steal, publicly reveal, or sell any personal information extracted from the device and the installed apps, including the user's information, information about contacts, or credentials.

To try bridging this gap, in this paper we discuss the importance of adding mobile components in actual cyber ranges, by illustrating a training motivating scenario (Section 2). Then, we point out the mobile threat landscape (Section 3) in order to identify the most promising family of vulnerabilities and attack patterns that can be fruitfully ported on cyber ranges. Furthermore, we detail a taxonomy and an architectural model of a generic cyber range (Section 4), before pointing out how such architectures can be extended and integrated with a mobile component (Section 5). Finally, we discuss some related work and some future steps towards the implementation of a next-generation cyber range emulating mobile components.

2 Motivating Scenario

In this section, we introduce a motivating scenario concerning a security testing against the employees and the Information Technology (IT) infrastructure of an enterprise.

2.1 A Penetration Testing Activity

A penetration test [9], also knows as a pentest, is a simulation of an attack against a given system or environment. The purpose of a pentest activity is identifying weaknesses and vulnerabilities in the target before an attacker can identify and exploit them for malicious activities.

This scenario is inspired by [25] and refers to the pentest conducted to find possible ways to steal reserved documents from a target enterprise. The initial conditions require the actors, namely the pentesters, to have no internal knowledge of the target system and negligible privileges. In detail, the pentesters have only a time-limited access to the guest network, i.e., the wireless network granting visitors access to the Internet and isolated from corporate resources. Nevertheless, the entire attack sequence led to the gain of the admin rights and the creation of a special Kerberos ticket, namely the Golden Ticket [32], which guarantees a long-term and hard-to-detect persistence. For the sake of presentation, here we only report the relevant steps that involve the mobile components.

First, the pentesters leverage their connection to execute a quick scan of the network and discover that even the employees use the guest wireless. In particular, they discover that the receptionist accesses the guest network to connect her own personal *smartphone* to the Internet.

Then, they exploit social engineering techniques [34] against the receptionist and gather the information that (*i*) she has a young daughter and (*ii*) often uses the connected smartphone to make the daughter play with some games inside.

This information allows the pentester to conduct a spear phishing [35] campaign to invite the receptionist to download and install an Android application for children. The suggested application is a real puzzle for children but modified with an injected code to grant the attackers a shell access to the mobile device.

The success of the phishing campaign provides a more stable link to the guest network through the receptionist's smartphone. This connection enables the pentesters to obtain a more in-depth knowledge of the network configuration, as depicted in Figure 1.

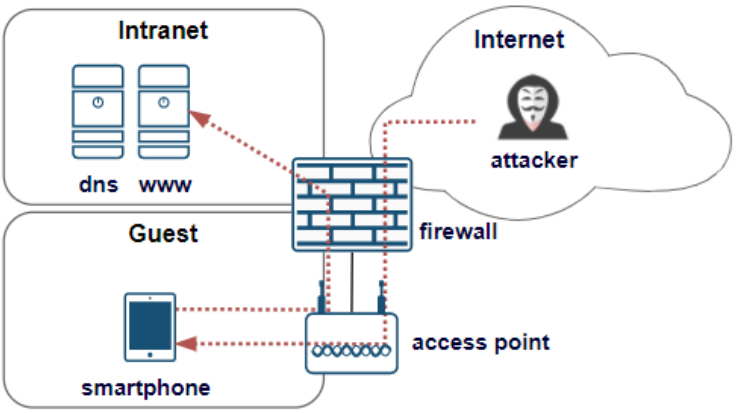

Fig. 1: The diagram of the penetration testing scenario.

Notice that the connected device can contact a name resolution server, namely *dns*, and a web server, namely *www*, hosted in a different network, probably the corporate *Intranet*. Moreover, a misconfiguration on the corporate *firewall* allows the guest hosts to access the login page of the manager application of the www server.

The pentesters use the smartphone as a gateway for attacking the www server. In particular, they obtain the admin password with a brute force attack [33], upload a web shell, and gain access to the Intranet.

A sequence of further lateral movements leads to the achievement of admin rights. We refer the interested reader to [25] for a detailed description of these steps.

3 The Mobile Threat Landscape

The involvement of vulnerable mobile devices in an attack can have severe consequences. Indeed, the attack discussed in Section 2 underlines the strategic importance of including mobile devices and applications in a cyber range exercise.

From a general standpoint, mobile devices share the same IT infrastructure of other connected devices. As shown in Figure 2, the device can connect through a Wi-fi or cellular network to the Enterprise systems and consequently to the exposed Back-end Services such as Email servers, File Shares, and Enterprise App servers. Furthermore, the mobile device may host third-party apps retrieved by either Public or Enterprise App Stores.

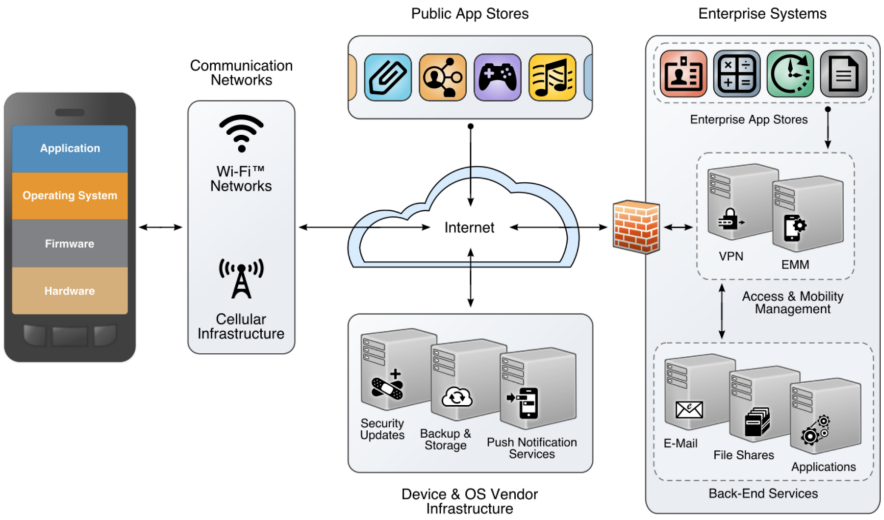

Fig. 2: Mobile ecosystem landscape.

To cope with the complexity of evaluating the security risks posed by such a mobile ecosystem, several initiatives propose frameworks and procedures for the threat modeling and the vulnerability assessment and penetration testing of mobile apps and devices. Notable examples include the Mobile Applications Security Verification Standard (MASVS) from the OWASP consortium [23] and the Mobile Threat Catalogue by NIST [19].

For instance, Figure 3 depicts the Threat Model Diagram for a mobile app that can interact with a company workstation and its back-end services [2]. The diagram details the *Assets* controlled or accessed by the app, the possible *Threat Agents* that can interact with the ecosystem and, finally, describes a set of *Controls* that enable the mitigation of security breaches.

We can use the Threat Model Diagram to map the attack exploited in the Motivating Scenario: the attacker crafted a malicious app without specific permissions (TA04 - Malicious App (Sandboxed)) and then delivered it to the victim using a phishing message (e.g., [6]). Once installed, the app grants access to the device functionalities and the credentials of the victim (A06 - A03). After this

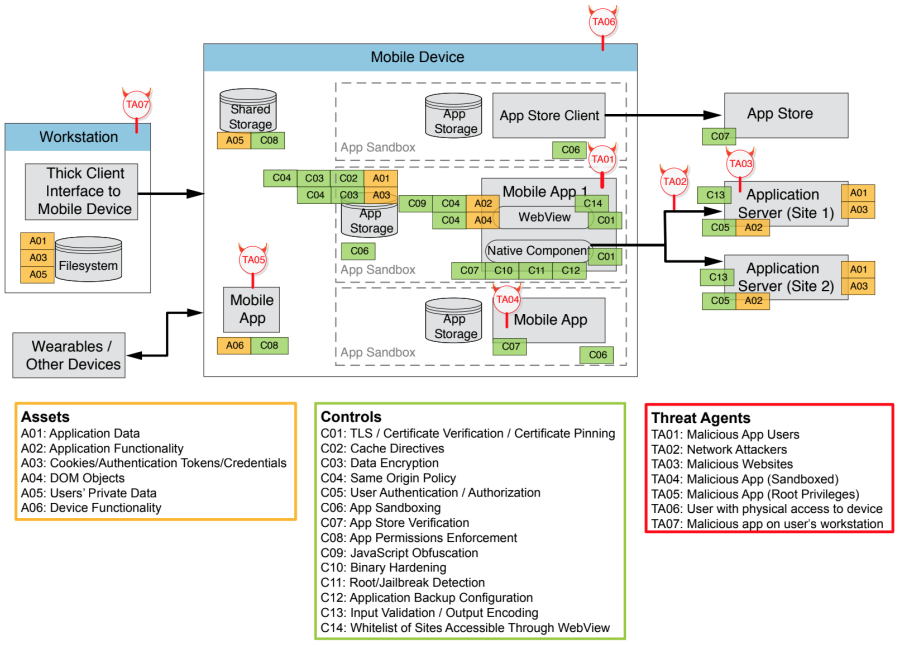

Fig. 3: Example of a Threat Model Diagram of a mobile app, taken from [2]

stage, the mobile app acted as a Trojan Horse to contact other connected devices, perform harmful commands, or spread malicious software.

4 Cyber Ranges

In this section, we provide a taxonomy and an architectural model of a generic cyber range. The overall layout and its key components, core technologies, and capabilities are inspired by [39,12].

In detail, the architectural scheme of a cyber range is depicted in Figure 4. It represents the two main levels, namely the *Management* and the *Training Environment* level, the core functionalities and related components. Two specific functionalities, i.e., the Orchestration and the Data Collection, enables interfacing between the two levels. A description of the above elements follows.

4.1 Management Level

The Management Level supports the planning and execution of the activities conducted within the cyber range. It can be accessed through a standalone *User Interface* (UI) or through an *Application Program Interface Gateway* (API GW). UI simplifies human interaction with the management functionalities. API GW provides a unified entry point for integrating external entities. This level makes

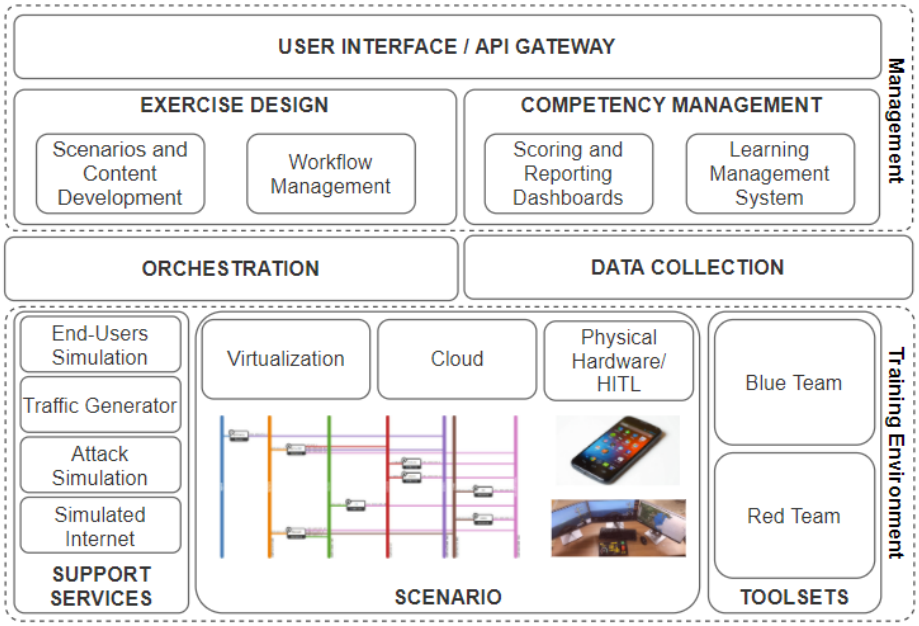

Fig. 4: The architectural scheme of a cyber range.

also available the *Exercise Design* and the *Competency Management* functionalities.

Exercise Design. The outcomes of the activities carried out in a cyber range are highly correlated to the quality of the training scenario. This means that the exercise designers must be able to build complex and heterogeneous infrastructures and reproduce events that characterize them, e.g., the users' activity. The Exercise Design functionality includes all the facilities to help designers to create training environments complying with the above constraints.

The definition of the infrastructure and its initial configuration is handled through the *Scenario and Content Development* (SCD) component. The primary function of SCD is allowing exercise designers to compose scenarios without having to know the underlying technologies and the complexity to configure the running environment. A common solution is to provide SCD with a domain-specific language (see, for example, [29,11]) for detailing the training environments of a cyber range. In particular, the above language, also identified as the *Scenario Definition Language* (SDL), allows the declarative specification of the scenario elements (e.g., a server, a firewall, or an IoT device), and their (mis)configurations (e.g., the installed software, the configured users, or the affected vulnerabilities).

A further component, namely the *Workflow Management* (WM), is in charge of scheduling and programming events, i.e., scenario injections, that are performed during the simulation. The WM supports the configuration of scenario

injections through an extension of SDL, namely *extended SDL* (eSDL), that resembles the languages for business process modeling. Briefly, an exercise designer can use eSDL to configure (a combination of) activities, e.g., a scenario element that generates Internet traffic, and the conditions allowing their executions, e.g., a time constraint.

Finally, the *Orchestrator* functionality receives the outputs from the Exercise Design components. First, it interacts with the *Scenario* components and creates the sandbox defined with the SDL specification. Then, it handles tasks and condition defined with eSDL and, when required, execute a specified activity interacting with the *Support Services* component.

Competency Management. The *Competency Management* functionality includes systems used to manage a competence program. In particular, the above systems allow an organization to perform the skill gap analysis, user profiling, and, consequently, define learning paths and competence assessment.

A first component, i.e., the *Scoring and Reporting Dashboards* (SRD), scores trainee during their interaction with the cyber range. The main outcome of the scoring system is monitoring the progression of activities and showing a timeline of performances of individual users or teams. SRD also reports the real-time situational awareness,e.g., the impact of tools used, and action taken by the users during the exercise.

A further core component that integrates the Competency Management functionality is the *Learning Management System* (LMS). LMS is required for the administration, documentation, tracking, reporting, and delivery of learning and assessment content.

The *Data Collection* functionality extracts from the training environment most of the data used by these components, e.g., services status, generated traffic, memory dumps, logs, or other artifacts.

4.2 Training Environment

The Training Environment level represents the sandbox in which teams and individual users carry out their activities during the exercise execution. It includes the *Scenario* along with the *Support Services* and *Toolsets* functionalities.

Scenario. The Scenario functionality includes the digital infrastructure where all the activities are staged as well as all the game-specific aspects such as targets, vulnerabilities and rules of engagement. To this aim, it mainly leverages *Virtualization* and *Cloud Computing* components to host the virtual instance of all the required components.

Nevertheless, the Scenario may also require the inclusion of physical devices such as network equipment or IoT systems. The *Physical Hardware* (PH) component manages the interfacing with these hardware items.

PH also enables the *Hardware-in-the-Loop* (HITL) capability and allows real systems to interact with virtual stimuli received from the training environment.

Support services. Actions performed by the *Support Services* functionality take place during the execution of the exercise and help to improve the realism of the training experience. Support Services interact directly with the Scenario and include the following components.

- *End-User Simulation* (EUS). It simulates the presence and behavior of benign users in the training environment. User simulation may refer to both internal users or fictitious clients accessing exposed services of the simulated environment. Examples of such user activity can include browsing the Internet, sending emails, or interacting with cloud or external services.
- *Traffic Generator* (TG). This component is capable of automatically generating both benign and malicious traffic. The function of traffic generators is crucial as the ability to identify and react to attacks by defense systems and operators lies in the precision with which traffic can be distinguished.
- *Attack Simulation* (AS). It refers to the ability to simulate attacks within and to the simulated environment. This component made available an attack library which contains a list of pre-defined attacks together with the ability to import/create custom operations. Attacks are performed interacting with the *Red Team Toolset*, and they may aim at exploiting vulnerabilities, executing lateral movements, and injecting malware or backdoors.
- *Simulated Internet.* It provides services outside the main simulated environment that are required for the realization of specific use cases. For example, this component can simulate social media (e.g., Twitter, Facebook, or LinkedIn), internet routing protocols, global services such as name resolution, remote API endpoints, or update and software repositories for various operating systems.

Toolsets. A cyber range is also equipped with state-of-the-art attack and defense tools made available to attackers (red team) and defenders (blue team) by the *Red Team* (RTB) and *Blue Team Toolsets* (BTT), respectively.

RTB consists of a collection of tools that aid in attacker operations, e.g., reconnaissance, weaponization, command-and-control, escalate privileges, lateral movements, or data exfiltration.

BTT includes services for the identification and prevention of intrusions, e.g., anti-malware or intrusion detection systems [18,17], and tools for carrying out analysis and managing security incidents, e.g., a System Information and Event Management (SIEM) software or digital forensic tools.

5 Extending Cyber Ranges with Mobile Ecosystems

As previously discussed, one of the main strengths of cyber ranges is the execution of exercises with a high degree of realism. Unfortunately, state-of-the-art cyber ranges cannot easily cope with a scenario that comprises mobile devices and apps. Indeed, current cyber ranges threat models do not include mobile ecosystems specifically, and thus the corresponding Assets, Threats, and Controls are coarse-grained and inaccurate. This implies that designers cannot create

mobile-first exercises - like the one presented in Section 2 - and thus, red and blue teams cannot train their attack and defense skills in a mobile ecosystem.

To this aim, we define and present here an extension of the cyber range architectural model to support mobile ecosystems.

5.1 Mobile-Aware cyber range Architecture

The support of the mobile ecosystem requires a revision of the components of a cyber range (see Section 3) following the analysis of their functionalities that need to be updated or added. In table 1, we summarize the results of the above analysis presenting the required integration tasks for revising each component.

Table 1: Integration tasks for mobile security components.

Management level (M)	
1. Exercise Design	
a. Scenario and Content Development	– M.a1.1: extend SDL with new components. – M.a1.2: add configuration script for new components.
b. Workflow Management	– M.b1.1: extend SDL+ with new actions and conditions.
2. Competency Management	
a. Scoring and Reporting	– M.2a: extend goals and metrics with ones related to mobile components.
b. Learning and Management System	– M.2b: introducing learning material and hands-on specific to mobile security.
Training environment (T)	
1. Scenario	
a. Virtualization/Cloud	– T.1a: extend automation tool for supporting mobile-specific connectivity.
b. Physical Hardware	– T.1b: support the interfacing with physical mobile devices.
2. Support services	
a. End-Users Simulation/-Traffic Generator	– T.2a: execute mobile-specific actions.
b. Attack Simulation	– T.2b: supporting the interaction with mobile-specific attack tools.
c. Simulated Internet	– T.2c: introducing the mobile-specific components.
3. Toolsets	
a. Blue Team	– T.3a: extending the weapon rack with mobile-specific tools for BTs.
a. Red Team	– T.3b: extending the weapon rack with mobile-specific attack tools.

Scenario and Content Development. A major set of updates concerns functionalities providing facilities for personnel involved in designing exercises, i.e., the SCD and WM components.

A first update consists of extending SDL languages to support the addition of mobile components and their configurations in the scenario specification (Task M.a1.1). For instance, cyber ranges can use easily expandable languages, e.g., *CRACK* SDL [29], to reproduce mobile elements and configurations through typed nodes. Figure 5 depicts the CRACK SDL diagram for the receptionist's smartphone as described in the motivating scenario.

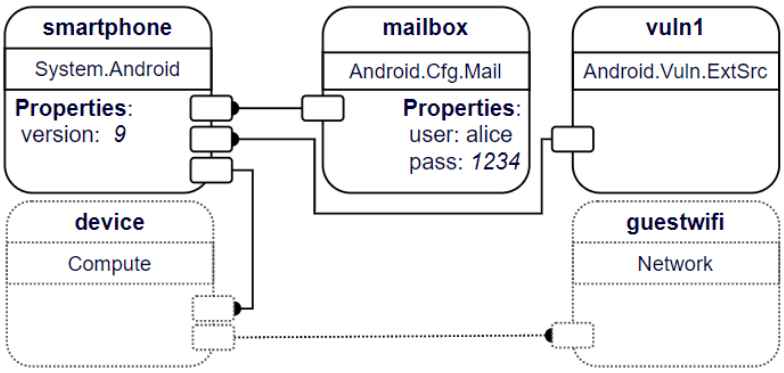

Fig. 5: An excerpt from the CRACK SDL diagram for the motivating scenario.

In detail, the *device* and *guestwifi* nodes represent components of the existing *Compute* and *Network* types, respectively. They represent the compute node, i.e., a virtual machine, connected to the guest network. The *System.Android*, *Android.Cfg.Mail*, and *Android.Vuln.ExtSrc* types are an example of an extension of the existing CRACK SDL. The *smartphone* node is the instance of the Android system running on the compute node. Its type allows the designer to specify the Android version through the *version* property. The *mailbox* node represents an Android configuration. In particular, it configures the mailbox used by the receptionist for reading emails on the smartphone. Its type allows the designer to specify the credential for accessing the mailbox through the *user* and *pass* properties. The *vuln1* node is a vulnerability that arises from a misconfiguration of the smartphone.

In addition to the support of an extended SDL language, the SCD needs to associate a corresponding configuration script to each new component (Task M.a1.2). For example, the Android.Vuln.ExtSrc type presented in Figure 5 must be translated into a sequence of operations required on Android for allowing the installation of apps from unknown sources.

Scenario. Hence, the scenario deployment is handled by an SDL Orchestrator that (*i*) parses the specification, (*ii*) selects all the configuration scripts associated with the scenario, and (*iii*) executes the script on the related System node.

Typically, the execution phase is managed using automation IT tools, e.g., *Ansible* [27], that use Secure Shell (SSH) daemons to access each compute node and execute the scripting operations to configure the scenario. However, mobile OSes do not support by default the SSH connectivity, and thus, it is not the best method to configure them. For example, the *Android debug bridge* [3] (ADB) can run over a TCP connection and facilitates a variety of device configuration and actions.

To this aim, the Orchestrator must be extended to support specific connectivity methods (i.e., an ADB plugin) for configuring mobile devices (Task T.1a).

The operations described above apply to mobile OSes running inside a virtual machine. Unfortunately, in some cases, mobile OSes can not be executed inside a virtual environment. For instance, this is due to a licensing problem, virtual hardware incompatibility, or because designers want to include a physical mobile device in the exercise scenario. For this reason, the PH component must be extended to support the interfacing with physical mobile devices (Task T1.b). A suitable solution for this extension could be (*i*) having the management interface through a USB cable and (*ii*) providing the connectivity to the training environment using a wireless access to the scenario networks.

Workflow Management. As it happened for the SDL language, a further update consists in extending eSDL in the Workflow Management to support activities and conditions related to a mobile ecosystem (Task M.b2.1). As an example, Figure 6 depicts the eSDL diagram of the spearphishing attack described in the motivating example. The diagram represents an injection event that is programmed to possibly occurs during the running exercise.

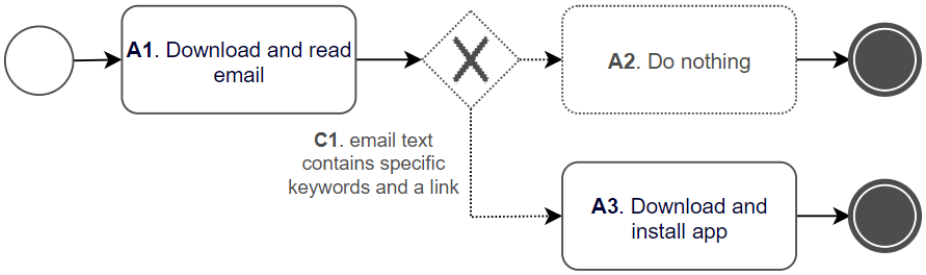

Fig. 6: A eSDL diagram for the spearphishing attack of the motivating scenario.

Differently from the traditional eSDL language, we introduced two activities, A1 and A2, to mimic the user operations on Android devices. The A1 activity download and read an email using the Android mail app. The A2 activity download and install an app from a HTML link in input and install it on Android.

Support Services. As detailed in Section 4, eSDL activities are executed in the running environment by the Orchestrator but through the EUS and TG com-

ponents of the Support Services. Therefore, EUS and TG must be extended to execute mobile-specific actions on running devices (Task T.2a). To do so, the Support Services need to include specific software to interact and simulate the user interaction, such as the monkeyrunner tool [4] and DroidBot [16] for Android OS.

Another fact that features the mobile ecosystem is its close relationship with Internet services. As described in Section 3, apps installed on mobile devices are typically retrieved from stores hosted on the public Internet. Moreover, many mobile apps implement their functionalities interacting with remote API services exposed by servers from the Internet. For this reason, the SI component must be integrated with services that simulate App Stores and remote API services (Task T.2c). For instance, the SI can include a customized application market based on the F-Droid [10] open-source marketplace and a system to mock remote API services based on the Mocky [13] tool.

Toolsets. Concerning the Toolsets component, the cyber range needs to include mobile-specific tools for both the Red and Blue teams (Tasks T.3a and T.3b).

For example, as described in the motivating scenario, the red team could use an injection tool, namely Metasploit framework [26], to generate a malicious payload and pack it into a legitimate Android application as a Trojan to compromise the user device. Other red team tools can include Owasp ZAP Proxy [21], zAnti [40], and iNalyzer [7].

Notice that some of these attacking tools need to be integrated with the AS component (Task T2.b) if they are used to perform operations during automatic attacks.

On the other hand, blue teams can use vulnerability assessment tools, e.g., apktool [37], jadx [31], and Mobile Security Framework (MobSF) [1], to detect vulnerabilities and patch mobile apps [5].

Competency Management. Mobile-aware cyber ranges requires also to extend the Competency Management. In particular, trainees must practice on topics and hands-on specific to mobile security before tackling an exercise on a complex and mixed scenario. These new resources need to be added to the LMS component (Task M2.b). For example, the contents hosted by LMS could be enriched with documentation and related exercises about the security of mobile ecosystems, such as fundamentals of decompilation of mobile apps, static analysis techniques, fundamentals of app repackaging and instrumentation, how to intercept the network traffic of an app, or forensic techniques for mobile devices.

Similarly, the rating of the trainees' performances must be extended according to their activities related to the mobile environment. In detail, the SRD score must include the goals that involve mobile assets (Task M.2a). For example, a red team's goals could include taking control of a mobile device, exfiltrate data from smartphones and apps, or threaten the overall service availability shutting them down. On the other side, new blue team goals could be reporting how they hardened flaws in mobile devices' configuration or how they analyzed and patched an installed app.

6 Related Work

The increasing popularity of Cyber Ranges is pushing the research community to design and build advanced Cyber Ranges architectures. More in detail, most of the attention is devoted to the building and deployment of complex and heterogeneous training scenarios to reduce the learning curve of students. The KYPO cyber range [38] is a cyber exercise and research platform where security experts can carry out exercises and experiments in a secure and isolated environment. KYPO simulates complex scenarios through virtual environments hosting different types of infrastructural components. These scenarios can also include mobile devices running virtual machines with Android images.

ADLES [15], CyRIS [8], CRACK [29] are tools for creating cyber range scenarios and use virtualization for deploying them. Like [38] they can potentially emulate mobile devices running Android in the instantiated virtual machines.

Unlike [38,15,8,29], a proficient training on mobile security can not be limited only to the integration of Android virtual machines in the running scenario. It requires at least the support of specific configurations and vulnerabilities, the emulation of their strong relationship with the user activity through ad-hoc runtime injections, and the support of targeted tools like the preliminary proposal we discussed here.

Many other initiatives offer hands-on solutions for training security experts on mobile security. Among them, the MSTG Playground [20], the Android Security Sandbox [36], DIVA Android [24] and InsecureBankv2 [30] provide vulnerable apps to be used as the target of VAPT sessions. Other similar forms of hands-on activities are Capture the Flag (CTF) challenges like UnCrackable Mobile Apps [22], KGB Messenger [14] and Mobile CTF [28].

However, although these hands-on solutions represent an effective way to learn mobile security, they are unable to cover the comprehensive process of cybersecurity defense like Cyber Ranges and our proposal do.

7 Conclusion

This paper is an early step towards enhancing generic cyber ranges with the full integration of the mobile security ecosystem in their training environments. We briefly reviewed the mobile threat landscape along with the component and functionalities of the current state-of-the-art cyber ranges. The above analysis allowed us to detail a list of tasks and examples to simplify and carry out such an integration. This project is still on-going and we identify the following main next steps.

First, a complete implementation of the proposal on an existing and running cyber range. During this activity, we strongly require to focus our effort on extending the SDL and its orchestrator engine with the support of a rich catalog of configurations, vulnerabilities and runtime actions for mobile components.

Then, after a survey of real-world case studies involving the use of mobile components in complex IT infrastructures, we need to identify key aspects to be

reproduced in training exercises. This effort should lead us to proficiently extend the current training scenarios with mobile components and make the most of the new features.

Finally, we need to validate our proposal by using an extended scenario and evaluating the training outcome after an exercise session.

Acknowledgement

This work was partially funded by the Horizon 2020 project "Strategic Programs for Advanced Research and Technology in Europe"(SPARTA) and by the Italian Ministry of Defense PNRM project "UNAVOX".

References

1. Abraham, A.: Mobile Security Framework. https://github.com/MobSF/Mobile-Security-Framework-MobSF, (Accessed on September 2020)
2. Amit Sethi, Neil Bergman, J.K.C.G.J.S.: The Developer's Guide to Securing Mobile Applications: Threat Modeling. https://www.synopsys.com/software-integrity/resources/ebooks/securing-mobile-applications-threat-modeling.html, (Accessed on September 2020)
3. Android: Android Debug Bridge. https://developer.android.com/studio/command-line/adb (2020), (Accessed on September 2020)
4. Android: monkeyrunner tool. https://developer.android.com/studio/test/monkeyrunner (2020), (Accessed on September 2020)
5. Aonzo, S., Georgiu, G.C., Verderame, L., Merlo, A.: Obfuscapk: An open-source black-box obfuscation tool for android apps. SoftwareX **11**, 100403 (2020). https://doi.org/https://doi.org/10.1016/j.softx.2020.100403, http://www.sciencedirect.com/science/article/pii/S2352711019302791
6. Aonzo, S., Merlo, A., Tavella, G., Fratantonio, Y.: Phishing attacks on modern android. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. p. 1788–1801. CCS '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3243734.3243778, https://doi.org/10.1145/3243734.3243778
7. AppSec Labs: iNalyzer. https://appsec-labs.com/inalyzer/, (Accessed on September 2020)
8. Beuran, R., Pham, C., Tang, D., Chinen, K.i., Tan, Y., Shinoda, Y.: Cybersecurity education and training support system: CyRIS. IEICE Transactions on Information and Systems **101**(3), 740–749 (2018). https://doi.org/10.1587/transinf.2017EDP7
9. Bishop, M.: About Penetration Testing. IEEE Security Privacy **5**(6), 84–87 (2007). https://doi.org/10.1109/MSP.2007.159
10. F-Droid Limited: F-Droid. https://www.f-droid.org/, (Accessed on September 2020)
11. Fite, B.K.: Simulating Cyber Operations: A Cyber Security Training Framework. https://www.sans.org/reading-room/whitepapers/bestprac/simulating-cyber-operations-cyber-security-training-framework-34510 (2018), (Accessed on September 2020)
12. Gonzalez, C.P.: Cyber Range The Future of Cyber Security Training. Tech. rep., SANS (2020)

13. Julien Lafont: Mocky.io. https://github.com/julien-lafont/Mocky (2020), (Accessed on September 2020)
14. Lambert, T.: KGB Messenger. https://github.com/tlamb96/kgb_messenger, (Accessed on September 2020)
15. Conte de Leon, D., Goes, C.E., Haney, M.A., Krings, A.W.: ADLES: Specifying, deploying, and sharing hands-on cyber-exercises. Computers & Security **74**(C), 12–40 (May 2018). https://doi.org/10.1016/j.cose.2017.12.007, https://doi.org/10.1016/j.cose.2017.12.007
16. Li, Y., Yang, Z., Guo, Y., Chen, X.: Droidbot: a lightweight ui-guided test input generator for android. In: 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C). pp. 23–26. IEEE (2017)
17. Migliardi, M., Merlo, A.: Modeling the energy consumption of distributed ids: A step towards green security. In: 2011 Proceedings of the 34th International Convention MIPRO. pp. 1452–1457 (May 2011)
18. Migliardi, M., Merlo, A.: Improving energy efficiency in distributed intrusion detection systems. J. High Speed Netw. **19**(3), 251–264 (Jul 2013)
19. NIST: Mobile Threat Catalogue. https://pages.nist.gov/mobile-threat-catalogue (2018), (Accessed on September 2020)
20. Open Web Application Security Project: MSTG Hacking Playground. https://github.com/OWASP/MSTG-Hacking-Playground, (Accessed on September 2020)
21. Open Web Application Security Project: Zed Attack Proxy. https://owasp.org/www-project-zap/, (Accessed on September 2020)
22. OWASP: UnCrackable Mobile Apps. https://github.com/OWASP/owasp-mstg/tree/master/Crackmes, (Accessed on September 2020)
23. OWASP: OWASP Mobile Security Testing Guide. https://owasp.org/www-project-mobile-security-testing-guide/ (2020), (Accessed on September 2020)
24. Payatu Software Labs LLP: DIVA Android. https://github.com/payatu/diva-android, (Accessed on September 2020)
25. Pierini, A., Trotta, G.: From APK to Golden Ticket (February 2017), https://www.exploit-db.com/docs/english/44032-from-apk-to-golden-ticket.pdf
26. Rapid7 LLC: Metasploit Framework. https://www.metasploit.com/ (2020), (Accessed on September 2020)
27. Red Hat: Ansible. https://www.ansible.com/ (2020), (Accessed on September 2020)
28. Rodriguez, I.: Mobile CTF. https://ivrodriguez.com/mobile-ctf/, (Accessed on September 2020)
29. Russo, E., Costa, G., Armando, A.: Building Next Generation Cyber Ranges with CRACK. Computers & Security **95**, 101837 (2020). https://doi.org/10.1016/j.cose.2020.101837
30. Shetty, D.: InsecureBankv2. https://github.com/dineshshetty/Android-InsecureBankv2, (Accessed on September 2020)
31. Skylot: jadx. https://github.com/skylot/jadx, (Accessed on September 2020)
32. Soria-Machado, M., Abolins, D., Boldea, C., Socha, K.: Kerberos golden ticket protection. Mitigating Pass-the-Ticket on Active Directory, CERT-EU Security Whitepaper **7**, 2016 (2014), https://cert.europa.eu/static/WhitePapers/UPDATED%20-%20CERT-EU_Security_Whitepaper_2014-007_Kerberos_Golden_Ticket_Protection_v1_4.pdf
33. The MITRE Corporation: T1110: Brute Force. https://attack.mitre.org/techniques/T1110/ (2020), (Accessed on September 2020)

34. The MITRE Corporation: T1268: Conduct social engineering. https://attack.mitre.org/techniques/T1268/ (2020), (Accessed on September 2020)
35. The MITRE Corporation: T1566.002: Phishing: Spearphishing Link. https://attack.mitre.org/techniques/T1566/002/ (2020), (Accessed on September 2020)
36. Toledo, R.: Android Security Sandbox. https://github.com/rafaeltoledo/android-security, (Accessed on September 2020)
37. Tumbleson, C.: Apktool. https://ibotpeaches.github.io/Apktool/, (Accessed on September 2020)
38. Vykopal., J., Oslejsek., R., Celeda., P., Vizvary., M., Tovarnak., D.: Kypo cyber range: Design and use cases. In: Proceedings of the 12th International Conference on Software Technologies - Volume 1: ICSOFT,. pp. 310–321. INSTICC, SciTePress (2017). https://doi.org/10.5220/0006428203100321
39. Yamin, M.M., Katt, B., Gkioulos, V.: Cyber ranges and security testbeds: Scenarios, functions, tools and architecture. Computers & Security **88**, 101636 (2020). https://doi.org/https://doi.org/10.1016/j.cose.2019.101636, http://www.sciencedirect.com/science/article/pii/S0167404819301804
40. Zimperium: zAnti. https://www.zimperium.com/zanti-mobile-penetration-testing, (Accessed on September 2020)