PortableVN: A Generic Mobile Application for Network Security Testbeds

Medha Pujari[1], Jhaghapratha Narayanamoorthy[1], Weiqing Sun[1], Farha Jahan[1], Bill McCreary[1], Quamar Niyaz[2], and Vijay K Devabhaktuni[2]
[1]College of Engineering, University of Toledo, OH, USA
[2]ECE Department, College of Engineering and Sciences, Purdue University Northwest, Hammond, IN, USA

Abstract— *Network security testbeds are instrumental for researchers and students to perform various network security related experiments in a controlled environment. The rapid growth of smartphone devices may bring a significant convenience for the users to let them access multiple testbeds from their mobile devices through a single interface. This paper presents a lightweight, user-friendly, and platform-independent smartphone application, titled as PortableVN, to access various testbeds. It serves as a generic entry point to connect with the testbeds and enables users to access their networks hosted on different testbeds with small configuration efforts. Evaluation is conducted to monitor its performance with two existing network security testbeds.*

Keywords: network security education, generic smartphone app, testbeds

1. Introduction

There have been many significant works toward the academic testbed development for network security education. These testbeds consist of virtual networks to carry out network security related experiments. In general, access to these testbeds is limited due to security reasons. They can be accessed only from the systems present inside the institutional network. A few of them are accessible both from the institutional and the public networks [1]. However, irrespective of their network accessibility, there is one major limitation about accessing these testbeds - *device constraint*. The users often experience difficulty to stay connected to their testbeds for several reasons, such as inconvenience to frequently go to the machines that connect to the testbeds, cannot carry their personal computers while traveling, and so on.

In addition, accessing a testbed using a compatible network protocol is a vital factor to be considered. Some testbeds provide web-based access to their virtual networks. The users need only a supportive web browser to access their testbeds [2]. However, most of the testbeds strictly accept connections from a specific network protocol, e.g., Virtual Network Computing (VNC), Remote Desktop Protocol (RDP), and Secure Shell (SSH) protocols. This constraint often urges the users, who want to access their testbeds from personal computers (or from work computers), to install the client software for the specific protocol to remotely access the testbed networks. It causes inconvenience for accessing and working on the testbeds. Also, as mentioned earlier the users may not always prefer carrying their personal computers with them.

In this work, we develop a generic smartphone application, *portableVN*, to address these inconveniences and make testbed-access comfortable. We make the application adaptable for various remote access network protocols. PortableVN will work for multiple testbeds upon providing their remote access protocol information and other configuration parameters. One of the primary objectives of portableVN is to allow users to access their testbeds from their smartphone devices. It will increase the testbeds accessibility so that users can conduct hands-on experiments more frequently and smoothly. The application consists of an interface that happens to be a generic point of entrance and abstracts the modules that implement the main logic to facilitate a user to connect to various testbeds. This approach eliminates the necessity of sticking to the work computers in the institution, or personal computers for most of the time the users spend on their testbeds. In addition, PortableVN relieves the users from any software installation overhead to access the testbeds.

When network security testbeds are spoken about, the most common visuals that scroll in mind about accessing those testbeds are terminal interfaces, remote connections, and so on. A user-friendly approach to access testbeds certainly improves the interaction of users with them, giving more convenience than what conventional terminal-based approach does. Having explained that being able to connect to a testbed from a smartphone adds more flexibility, the platform the smartphone has is equally essential to be considered in the process. Limiting the users to a specific mobile platform to work on their security testbeds does not really overcome the device constraint. The users should not be concerned about the underlying platforms of their smartphones and still be able to to work on their testbeds with ease. PortableVN is designed and developed to accomplish it. Security matters everywhere, and even while accessing security testbeds. Since this application stands like an additional layer in the process of connecting to a security testbed, it needs to maintain itself securely as well.

ISBN: 1-60132-509-6, CSREA Press ©

The ability to let the users connect to multiple testbeds through a single interface makes this application “hybrid". Although the user can work on one testbed at a time, portableVN gives comfort to switch from one testbed to another whenever the user wishes to. Additionally, it is hybrid in terms of its support to smartphone platforms.

The paper covers the application design, technological aspects that make the implementation possible, an overview of the operational flow within the application, and mechanisms employed that enable it to set-up connections with various testbeds, its evaluation process, and future enhancements in the line.

2. Design, Architecture, and Implementation

A cross-platform application can support multiple underlying platforms. There are two types of smartphone application development frameworks: Hybrid and Native. The native framework provides the application development environment for a single native platform, e.g., XCode [3] and Android Studio [4] for iOS and Android applications, respectively. A downside with the native apps is that they only support the platform for which they are developed. The code base needs to be changed or modified to use them on a different platform. This approach is not suitable for applications that are expected to be used on multiple platforms. Hybrid applications take advantage in this respect. They serve as a one-stop solution for a wide range of smartphone platforms. Only one code base is needed to develop and maintain applications for multiple platforms such as Windows, Android, and iOS, making the approach cost-effective, time-saving, and easier to scale. The application proposed in this paper falls in the hybrid category.

PortableVN when integrated with a testbed forms a three-layered architecture as shown in Figure 1. This section discusses the internal details of the application including the components that contribute to its generic behavior, and the operational flow.

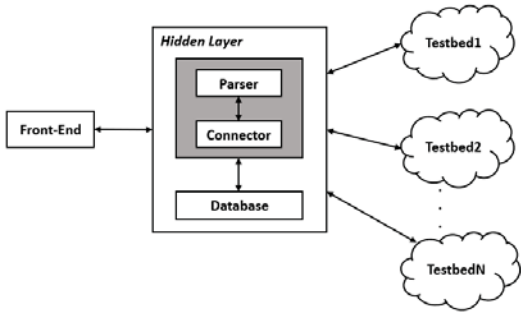

Fig. 1: The PortableVN architecture after integration with testbeds.

2.1 The Front-End

The front-end implementation uses Ionic framework, a popular hybrid framework for cross-platform smartphone application development. It is open-source and built on HTML5, Angular, and TypeScript. Apache Cordova [5], a framework in itself is another component of Ionic. Cordova gives Ionic most of its power, making it deployable on various platforms including iOS, Android, and Windows. Ionic offers the best web and native application components to build highly interactive applications. A few more features of the framework are: i) fully cross-platform ii) premier native plugins iii) comprehensive documentation, and iv) application management with a single code base. Figure 2 shows a basic overview of Ionic architecture and interaction among its components.

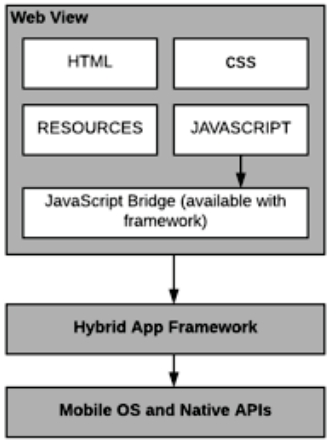

Fig. 2: Basic Ionic architecture representation [6]

The framework has many built-in typescript [7] extensions and a variety of features including fingerprint, Bluetooth, health kit, and Cordova or PhoneGap [8] plugins. A developed application can be uploaded, shared, and tested on native devices through a platform called Ionic View [9].

The front-end of portableVN consists of a user interface that launches with a registration/log-in page where a user can sign-up or log-in to the application. After a successful log-in, the user can select a testbed (in case the user had earlier connected to one or more) among the list of testbeds displayed on the screen. To connect to a new testbed, the user must provide a configuration file after log-in. The configuration file consists of information to connect with the testbed. The front-end has nothing much to do with the configuration file, and only forwards it to the next layer of the application, which is the hidden layer. Figure 3 shows an interaction flow chart between the user and the front-end. The registration/log-in screen of portableVN is the same to all kinds of users irrespective of their privileges on their testbeds.

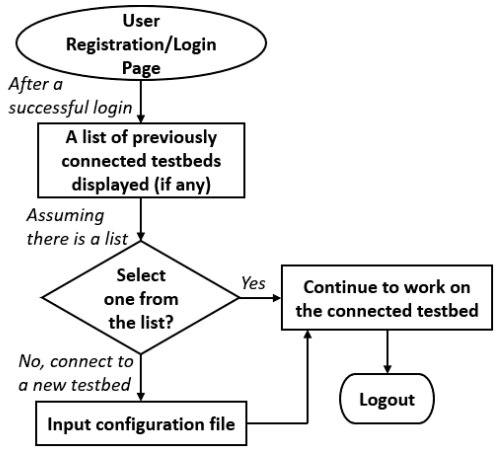

Fig. 3: Flow chart for interaction between a user and the front-end.

2.2 The Hidden Layer

The hidden layer implements the core functionalities of the application. The parser and the connector are the two essential modules in this layer. The functionality of this layer is implemented in NodeJS, an integration server for the Ionic framework that forms the front-end. The hidden layer is responsible for authenticating and authorizing the users of the application. It includes a database to which the NodeJS server is associated. The database stores the details of the registered users and the configuration files of the testbeds connected.

2.2.1 The Parser Module

The configuration files provided by the users need to be systematically analyzed. The parser module carries out this task. It works closely with both the front-end and the connector module. The parser reads from a configuration file, organizes and passes the required information to the connector module. After the analysis, the configuration file is stored in the database. When a user wants to access the same testbed at a later point of time, its configuration file is fetched from the database based on the identity of the testbed. For each testbed, the application creates a record for the corresponding user with the testbed's configuration file. Therefore, it needs to be provided with a separate configuration file for each testbed.

A configuration file is an XML file that stores the necessary testbed access information for a user. An administrator of a testbed provides this information to its users. At its current stage, portableVN needs the user to input a file that is manually set up in accordance with the parsing abilities of the hidden layer. The file should have the tags and parameters set in cohesion with the terms detected and understood by the parser. After future enhancements, the application should be able to provide a generic template of a configuration file which a user can either use as a reference to make another file or can fill in the same template to add a testbed. In general, a testbed configuration file should contain the following information:

- Remote access network protocol: The user should provide the type of network protocol used to access the targeted testbed. The application uses this information to make itself clear about what kind of connection to be established with the testbed.
- The Gateway IP address or Hostname: The IP address or hostname is used in constructing a gateway URL which is taken care of by the application's parser module. The URL is required to launch a log-in page for the testbed.
- Port number: The port number on which the server listens to the connection requests is another vital detail required to create the connection. This information is also used in constructing the URL and passed to the connector module.

If a user (probably an administrator) has access to multiple testbeds, the details for each testbed should be provided in a separate file and passed to the application. Once the application launches a login page, the user needs to enter the credentials on the login screen and gain access to the testbed after a successful login. Figure 4 shows a sample testbed configuration file for a user.

```
<testbed name = "Testbed1">
    <connection-url>
        <protocol>ssh</protocol>
        <param name = "ip">131.100.100.100</param>
        <param name = "hostname">testbed1_gateway</param>
    </connection-url>
</testbed>
```

Fig. 4: A sample configuration file in PortableVN.

In a case where a user has access to multiple virtual networks of a testbed and has mentioned all the necessary details in the configuration file, the hidden layer performs the following tasks:

- The parser reads the information of the virtual networks one after another.
- If they have separate gateways, it frames separate URLs for them respectively. In such a case, the parser sends the names of those virtual networks to the front-end, to display them to the user. They are displayed as clickable links on the front-end. The user can then select a virtual network to log-in to.
- If all of those virtual networks have a common gateway to enter through, the parser frames a single URL which is later used to launch the login page. In this case, the names of virtual networks are not sent to the front-end.

2.2.2 The Connector Module

This module attempts to establish a connection with a testbed, by taking the necessary information from the parser. The hidden layer of the application has web clients of various networking protocols (SSH, VNC, RDP) installed, so as to give users web-based access to their testbeds. The connector module selects a web client based on the protocol information present in the configuration file. If the application fails to launch a connection with a testbed, the hidden layer sends appropriate error messages to the front-end to notify the user about the failure. In case the protocol supported by the targeted testbed is not provided to the application, it launches an SSH connection with the testbed.

Figure 5 is an overview of the tasks handled by the parser and the connector modules inside the hidden layer.

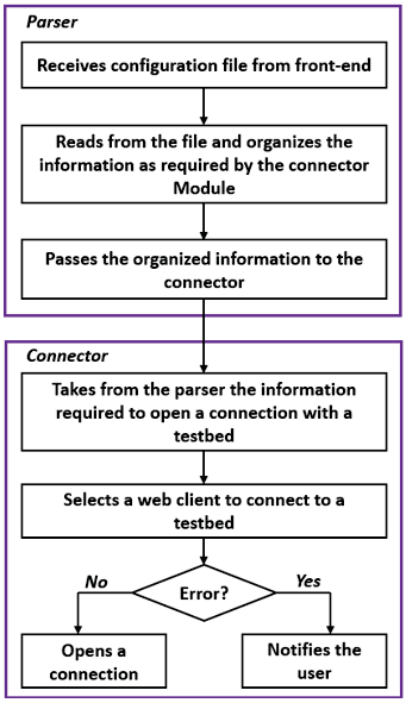

Fig. 5: Operation flow inside the Hidden Layer.

2.2.3 The Database

PortbleVN uses MySQL database primarily to store the details of the application users and their testbeds. The details are stored when users register to portableVN. Sensitive information (the login credentials) pertaining to user-login is stored in an encrypted format for security reasons. Each time a user signs-in, the user is authenticated by checking against the record present in the database. When the user supplies a configuration file of a testbed to connect to, the file is stored in the database, and thereafter, the hidden layer can carry out other tasks to process the file and launch connection(s). The access URLs of the testbeds, when framed, are appropriately stored in the database.

2.2.4 Security Practices

Security is a key concern of this work. The application needs to make sure the configuration files of testbeds are not only transported securely to the hidden layer but are also stored safely thereafter. The same applies to the details of the users of this application. The portableVN is neither supplied with the user credentials required to log-in to a testbed nor does it validate them while the testbed login. The validation is taken care of by the targeted security testbed.

As the hidden layer forms the main medium in between a user and a testbed which is already secured by its own means, the security of the hidden layer becomes pivotal in making portableVN rigid and unyielding. To secure the hidden layer which NodeJS is a major part of, we follow certain conventional security best practices [10]. A web token authentication technique, JsonWebToken (JWT) Authentication, is followed to authenticate the users of the application when they log-in to it.

3. Evaluation

The performance of portableVN has been assessed by connecting it to two existing testbeds: LightVN and V-NetLab. Figure 6 portrays the integration of the portableVN with these testbeds.

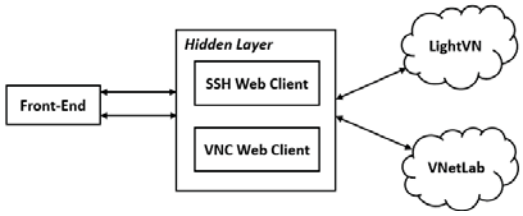

Fig. 6: Overview of PortableVN integration with LightVN and VNetLab.

3.1 LightVN testbed

LightVN is an academic testbed setup to conduct hands-on experiments in the computer and network security related courses [11]. It was built as a light-weight and flexible testbed which later adapted to web-based interaction with its users [2]. The term web-based implies that this testbed acts like a web application, and all a user needs to do is log-in to the testbed from a web browser that supports JavaScript and CSS. The URL of the login page happens to be the gateway URL for LightVN.

The user needs to provide the following details about the LightVN testbed in the configuration file to get connected with it. One of the important details is that it allows SSH connections. Hence, the application connects to the testbed through an SSH channel, for which it utilizes a web-based

terminal emulator which is in turn supported by and is configured on the testbed. The gateway URL is another essential detail to be put in the file. Through the gateway, the portableVN connects to the emulator of the testbed and can provide users entry into the virtual network's hosts. LightVN testbed internally consists of multiple virtual networks each of which has a group of users assigned. Therefore, for a user to log-in to the LightVN, he/she must have log-in credentials to access the network. Connection to the testbed, in case of the LightVN, means to get connected to an internal virtual network the user intends to work on. After submitting the configuration file with all the required information, the following is the operational flow that takes place for the user to gain access to the testbed:

- The parser module reads the configuration file and processes its information to pass to the connector.
- The parser does not have to explicitly construct a gateway URL as it is already read from the file.
- The connector module is given the parsed information.
- In case the testbed admin uses the application, he/she can provide the details of multiple internal virtual networks in the file.
- The connector module launces the testbed gateway after which the user can log-in using his/her credentials for that virtual network.
- The user can then continue to work on the virtual network.

The user can work on specific hosts within the virtual network through the terminal emulator of the testbed. Once the emulator launches, the user can log-in to the host with his/her user name and password specific to the host. Figure 7 is the configuration file used to connect to LightVN testbed.

```
<testbed name = "LightVN">
    <connection-url>
        <protocol>ssh</protocol>
        <param name = "ip">131.183.222.81</param>
        <param name = "gateway-url">http://131.183.222.81/LightVN/</param>
    </connection-url>
</testbed>
```

Fig. 7: PortableVN configuration file for LightVN.

The web-based terminal emulator used by this testbed is Shell-In-A-Box [12]. It provides a browser-based terminal login interface and hence, keeps its users free from typing commands in terminals to remote login to and SSH the virtual hosts. The student user groups can access their respective subnetwork of the testbed by logging in using their credentials and SSH into different virtual hosts from a web page. The testbed includes other components such as Linux Containers and OpenVSwitch in the back-end for the testbed implementation.

Figure 8 is a screenshot of the mobile view when connected to LightVN using portableVN from an android phone. The figure shows a terminal log-in prompt for a host in the testbed network.

Fig. 8: A LightVN host interface connected via PortableVN.

To connect to the URLs, the Ionic framework uses a device-level browser plugin called InAppBrowser. This plugin is associated with Apache Cordova and needs to be installed as a dependency.

3.2 V-NetLab testbed

The application has also been assessed by connecting it to another testbed, V-NetLab [13]. The testbed uses VMware for virtualization and contains a number of virtual machines distributed over a set of physical hosts [13][14]. This testbed allows VNC connections, as specified in the configuration file. Hence, portableVN uses the VNC web client, namely, noVNC [15], which opens a remote session with the VNC server (TightVNC recommended) installed on the hosts in the testbed. The web client launches the gateway URL, framed by the parser, through an HTML5 enabled browser. The integration of portableVN with this testbed is similar to the steps mentioned for the LightVN testbed.

The configuration file for this testbed has details of the gateway, and the port number on it that listens to the incoming connection requests. The following are the steps that take place after the user selects this testbed to connect with:

- The application launches a log-in page of the gateway.
- User enters the testbed after a successful login attempt.
- As the user is now inside the testbed, all user actions hereafter are handled by the testbed. The user can choose to work on various hosts in the testbed network.

4. Related Works

There have been several research works performed in developing academic testbeds, and there are a good number

of such platforms built which are in use till date. This section quotes a few examples of educational or research-oriented testbed platforms in use, and a state-of-the-art application which serves as a mediator to enable remote access using multiple protocols.

Xen Worlds, developed at Iowa State University, was designed to build a virtualized lab environment for information assurance. It provides access to a huge number of students to log in to the lab remotely. In order to keep up the information assurance, the system needs strict security. The testbed can be accessed both by on-campus and off-campus students simultaneously. The two most important design characteristics of this system are being consistently accessible and being easy to use. The team made the virtual lab environment system available for all days in a week, 24 hours every day. The access permissions remain the same for both on-campus and off-campus students. Various customized configurational set-ups were made on the platform for the students to perform lab experiments, in accordance with the curriculum requirements [2][16].

DETER was a collaborated work by the University of UC Berkeley, USC/ISI, UC Los Angeles, George Mason University, Lehigh University, Colorado State University, University of South California. This testbed is built upon Emulab technology [17]. The project was developed to perform cyber defensive experiments. An in-built web interface allows interaction to the testbed, and strict security measures are in place. Any intermediary data and traffic generated during experiments and research are confined to its dedicated experimental network. No traffic is allowed out of the experimental network. The testbed is designed to be portable as well [18][19].

A research team of the University of Illinois at Urbana-Champaign worked on a research project for the development of a testbed that can analyze the level of vulnerabilities that occur while using publicly accessible communication networks. The motto behind this project was to overcome the vulnerabilities present in the cyber power systems [20] which were primarily implemented with the recent generation SCADA protocols [21].

There is a product available in the software market, by name Apache Guacamole [22]. It is a clientless HTML5 web application that can be used to access various remote servers and desktops using just a web browser. All that is needed is a web browser without any additional software or plugins to be installed. The software is completely free and open source [23]. It offers its native support to Linux based systems.

The research and development works made so far, have been undoubtedly profound in addressing respective problems [22][24], producing efficient testbeds to conduct various research activities like attack detection [25], data preservation, and so on, concentrated in the field of security. The application presented in this paper aims at overcoming the barriers of not only the platform adaptability but also those with networking protocols to access more than one testbed. Therefore, users, from a wide range of mobile platforms, can connect to multiple testbeds that they have access to and can switch to work from one to another. In other words, a user selects one among the list of testbeds connected so far with the user's account; to switch to another testbed, the user comes back to the menu and chooses another one from the list to continue with it; if the testbed to be switched to is not in the list, the user needs to add the testbed to the application by providing its configuration file. Supporting a variety of underlying platforms and the ability to connect to multiple testbeds without letting the users install any additional software are the two major challenges for an application, and this paper intends to address them.

5. Conclusion and Future Work

As an effort to make the existing network-security related educational testbeds more conveniently accessible, the paper proposes a generic solution which can help users connect to their testbeds using their mobile devices irrespective of the underlying platforms. The solution intends to expand the possibilities of access to the academic or educational testbeds in existence, and also eliminate the need for the users to install various remote accessing client software. The paper discusses the evaluation of portableVN with two existing testbeds.

As future work, we plan to perform its evaluation by accessing more testbeds, and from more smartphone platforms. Another future task is to give users an ability to update the configuration files of the connected testbeds in the application. In addition, we plan to support users who might be interested in viewing the topology of the virtual network of the connected testbed. Attempts will be made to provide a view-only (initially) access to the network topology of the connected testbed. A dynamic topology generator is also an idea of future work which can facilitate the users to see a dynamic network topology based on the number of hosts and other information provided as input.

References

[1] B. Daya, "Network security: History, importance, and future," *University of Florida Department of Electrical and Computer Engineering*, vol. 4, 2013.

[2] W. Liu, Q. Niyaz, W. Sun, and A. Y. Javaid, "A web-based lightweight testbed for supporting network security hands-on labs," in *2018 IEEE International Conference on Electro/Information Technology (EIT)*. IEEE, 2018, pp. 0498–0503.

[3] "Xcode," https://developer.apple.com/support/xcode/ Accessed May 12, 2019.

[4] "Android Studio," https://developer.android.com/studio Accessed May 12, 2019.

[5] "Apache Cordova," https://cordova.apache.org/ Accessed May 12, 2019.

[6] "Ionic Architecture," https://mentormate.com/bg/blog/what-is-ionic-getting-started-guide-for-beginners/ Accessed May 12, 2019.

[7] "Typescript," https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html Accessed May 12, 2019.
[8] "Phonegap," https://www.npmjs.com/package/phonegap Accessed May 12, 2019.
[9] "Ionic Framework," https://ionicframework.com/ Accessed May 12, 2019.
[10] "Security Best Practices with Node.js," https://tinyurl.com/y7fanjnq Accessed May 12, 2019.
[11] Q. Niyaz, W. Sun, R. Xu, and M. Alam, "LightVN: A Light-Weight Testbed for Network and Security Experiments," in *2015 12th International Conference on Information Technology-New Generations*. IEEE, 2015, pp. 459–464.
[12] L. Morell and C. Jiang, "Using shellinabox to improve web interaction in computing courses," *Journal of Computing Sciences in Colleges*, vol. 30, no. 5, pp. 61–66, 2015.
[13] K. Krishna, W. Sun, P. Rana, T. Li, and R. Sekar, "V-netlab: a cost-effective platform to support course projects in computer security," in *Proceedings of 9th Colloquium for Information Systems Security Education*, 2005, p. 4.
[14] W. Sun, V. Katta, K. Krishna, and R. Sekar, "V-netlab: An approach for realizing logically isolated networks for security experiments." *CSET*, vol. 8, pp. 1–6, 2008.
[15] "VNC Web Client," https://novnc.com/info.html Accessed May 12, 2019.
[16] B. R. Anderson, A. K. Joines, and T. E. Daniels, "Xen worlds: leveraging virtualization in distance education," in *ACM SIGCSE Bulletin*, vol. 41, no. 3. ACM, 2009, pp. 293–297.
[17] "Emulab," http://www.emulab.net/ Accessed May 12, 2019.
[18] T. Benzel, R. Braden, D. Kim, C. Neuman, A. D. Joseph, K. Sklower, R. Ostrenga, and S. Schwab, "Experience with deter: A testbed for security research." in *Tridentcom*, 2006.
[19] "Deter Lab," http://www.isi.edu/deter Accessed May 12, 2019.
[20] "Systems Research Group," http://srg.cs.illinois.edu/research/ Accessed May 12, 2019.
[21] J. Edmonds, M. Papa, and S. Shenoi, "Security analysis of multilayer scada protocols," in *Critical Infrastructure Protection*, E. Goetz and S. Shenoi, Eds. Boston, MA: Springer US, 2008, pp. 205–221.
[22] "Apache Guacamole," https://guacamole.apache.org/ Accessed May 12, 2019.
[23] B. K. Ram, S. A. Kumar, S. Prathap, B. Mahesh, and B. M. Sarma, "Remote laboratories: For real time access to experiment setups with online session booking, utilizing a database and online interface with live streaming," in *Online Engineering & Internet of Things*. Springer, 2018, pp. 190–204.
[24] W. D. Armitage, A. Gaspar, and M. Rideout, "A uml and mln based approach to implementing a networking laboratory on a scalable linux cluster," *Journal of Computing Sciences in Colleges*, vol. 23, no. 2, pp. 112–119, 2007.
[25] M. Little, "Tealab: a testbed for ad hoc networking security research," in *MILCOM 2005-2005 IEEE Military Communications Conference*. IEEE, 2005, pp. 936–942.