Proceedings

The Role of Software-Defined Networks for Practical Learning in the Engineering Areas [†]

Cristina Alcaraz [1,*], Antonio Ortega [2] and Rodrigo Roman [1]

[1] Department of Computer Science, University of Malaga, Campus de Teatinos s/n, 29071 Malaga, Spain; roman@lcc.uma.es
[2] Radar and Electronic Defense, Indra. Carretera de Loeches 9, Torrejon de Ardoz, 28850 Madrid, Spain; aofernandez@indra.es
* Correspondence: alcaraz@lcc.uma.es; Tel.: +34-952-139-313

† Presented at the 2nd Innovative and Creative Education and Teaching International Conference (ICETIC2018), Badajoz, Spain, 20-22 June 2018.

Published: 29 October 2018

Abstract: In this work, we have carried out a case study on the construction of a complex internetworked system that characterizes part of a Smart City within the context of Master Theses. We show that, compared to other network simulators and emulators, SDN technologies allow students to easily configure and manage large simulated communication infrastructures, and facilitate the integration of various network analysis tools.

Keywords: Software Defined Network; practical learning; virtual emulators

1. Introduction

In engineering fields related to industrial, telecommunication or computer sciences, it is difficult to manage the teaching and learning of the construction of complex internetworked systems. Yet the Software-Defined Network paradigm (SDN) [1] supplements many of these needs, by providing a tool where students can have direct contact with large simulated communication infrastructures, apply concepts of networks and engineering, and incorporate various types of protocols.

We have experimented with the use of Mininet and the Openflow protocol to represent SDN topologies within the context of Master Theses, so as to observe how the students adapt by themselves to the conditions of the environment to configure complex environments at a low cost and effort. In this paper we show that, apart from simplifying the deployment and configuration of software-based technologies [1], SDN helps and allows students to learn and develop the conceptual and technical lessons required to comply with the minimal competencies required.

2. A Practical Use Case: Smart City Applications in Engineering Disciplines

The construction and simulation of complex systems in disciplines related to the engineering is not always possible or is occasionally complicated to address. The search for specific works linked to the real world, in which a set of technologies and protocols coexist, are keys to approach the engineering students to the real world and the labor market.

Unfortunately, curricula and time restrictions limit the required time to research or prepare customizable or suitable work environments, forcing the students to concentrate all their attention, effort and time in solving specific problems, generally assuming unforeseen alternatives and/or developing the minimal goals to cover the initial expectations and overcome credits. All these handicaps, in turn, constrain or discard the exploration of other relevant factors for the learning process itself, such as the "research and the discovering" of suitable simulation environments according to the conditions and initial goals of the assigned work projects.

An example of these limitations is found when Smart Cities applications are sketched as part of a Master's end project. The management of smart meters in the home premise is a clear example of a Smart City application. The smart meters are devices in charge of automatically recollecting and processing the real consumption of a determined area and sending the consumption data to a concentrator. This concentrator has the ability to recollect evidence flows of several home premises and send this information to the billing entities to manage the final charging and the real demand [2,3]. Although the metering management process implies a significant evolution for the human being and its welfare within a society [2], numerous security and privacy breaches might arise, some of them mainly caused by the inherent vulnerabilities of the communication protocols [3–5].

In this sense, the University of Malaga (UMA) has recently led a Master's end project in order to analyze the different security weaknesses of the ModbusTCP protocol [6] in a scenario composed of eight smart meters—as illustrated in the picture to the left side of Figure 1. In this work, a man in the middle (MITM) was planned and located (as represented in the picture to the right side of Figure 1) in order to show the multiple types of cyberattacks on the path and against the confidentiality, integrity and availability of the data in the channel, among others [5]: eavesdropping, replay, data modification, false data injection, or flooding (e.g., ARP flooding, smurf or fraggle).

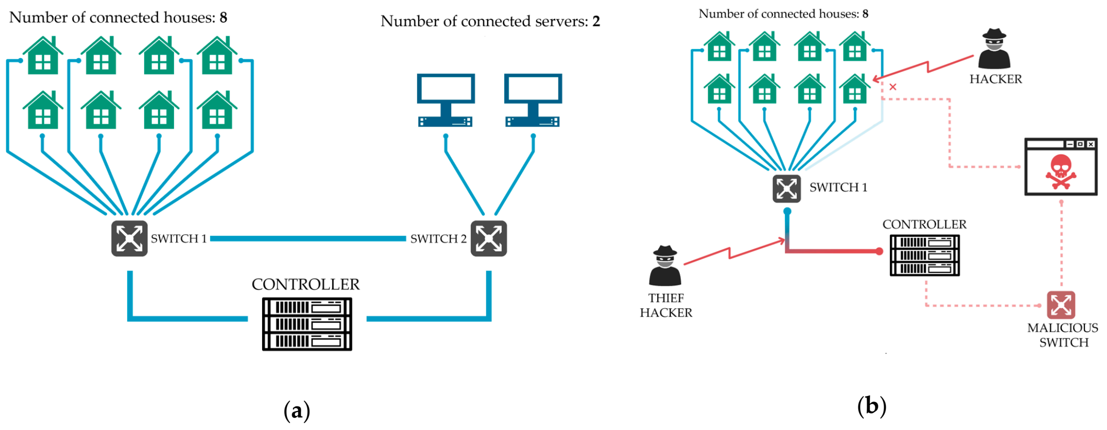
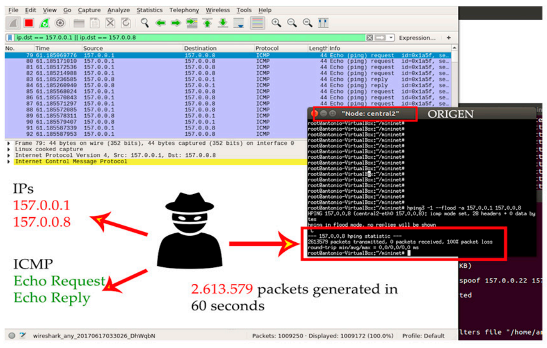

Number of connected houses: 8
Number of connected servers: 2
SWITCH 1
SWITCH 2
CONTROLLER
(a)
Number of connected houses: 8
HACKER
SWITCH 1
CONTROLLER
THIEF HACKER
MALICIOUS SWITCH
(b)

Figure 1. Use case: (**a**) Topological illustration of the application scenario with 8 smart meters and one controllers; (**b**) Representation of the practical scenario taking into account the presence of a MITM.

For the scenario construction, SDN together with Mininet [7] and OpenFlow were determinant due to the specificity of the communication protocol and the complexity of the work environment. The majority of simulators support the typical TCP/IP protocols (e.g., SSH, SSL/TLS, IPSec, etc.) but do not cover any communication protocol, and, more specifically, those related to the industry, such as ModbusTCP, DNP3 or IEC-104. In addition to this, Mininet allows for the integration of many of the hacking tools which are based on Linux distributions, facilitating the interception and manipulation tasks. Such tools include Wireshark (for dissertation of the packets and sniffing [8]), Ettercap (suite for MITM [9]), Hping3 (for flooding), Medusa (for brute force) or Scapy (packet manipulation software [10]), and are represented in Figure 2a,b.

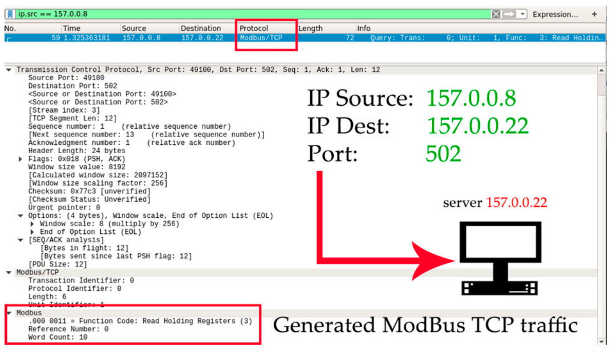
One of the attacks that can affect these scenarios is the TCP SYN flood cyberattack. In this attack, the hacker will send a significant number of TCP packets with the SYN flag activated. This TCP traffic, generated using the tool Hping3, forces the server to start thousands of connections per second, but none of these connections finish the 3-way TCP handshake, meaning that the server will eventually run out of resources, eventually causing a critical failure.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...
ORIGEN
IPs
157.0.0.1
157.0.0.8
ICMP
Echo Request
Echo Reply
2.613.579 packets generated in 60 seconds

```
ip.dst == 157.0.0.1 || ip.dst == 157.0.0.8
No.  Time          Source      Destination  Protocol  Length Info
  79 61.185069776  157.0.0.1   157.0.0.8    ICMP      44 Echo (ping) request  id=0x1a5f, se…
  80 61.185171010  157.0.0.1   157.0.0.8    ICMP      44 Echo (ping) request  id=0x1a5f, se…
  81 61.185172536  157.0.0.1   157.0.0.8    ICMP      44 Echo (ping) request  id=0x1a5f, se…
  82 61.185214988  157.0.0.1   157.0.0.8    ICMP      44 Echo (ping) request  id=0x1a5f, se…
  83 61.185236585  157.0.0.8   157.0.0.1    ICMP      44 Echo (ping) reply    id=0x1a5f, se…
  84 61.185260940  157.0.0.8   157.0.0.1    ICMP      44 Echo (ping) reply    id=0x1a5f, se…
  85 61.185568024  157.0.0.1   157.0.0.8    ICMP      44 Echo (ping) request  id=0x1a5f, se…
  86 61.185570843  157.0.0.1   157.0.0.8    ICMP      44 Echo (ping) request  id=0x1a5f, se…
  87 61.185571297  157.0.0.1   157.0.0.8    ICMP      44 Echo (ping) request  id=0x1a5f, se…
  88 61.185572085  157.0.0.1   157.0.0.8    ICMP
  89 61.185578311  157.0.0.8   157.0.0.1    ICMP
  90 61.185579407  157.0.0.8   157.0.0.1    ICMP
  91 61.185587339  157.0.0.1   157.0.0.8    ICMP
  92 61.185587953  157.0.0.1   157.0.0.8    ICMP
▸ Frame 79: 44 bytes on wire (352 bits), 44 bytes captured (352 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 157.0.0.1, Dst: 157.0.0.8
▸ Internet Control Message Protocol
wireshark_any_20170617033026_DhWqbN    Packets: 1009250 · Displayed: 1009172 (100.0%)    Profile: Default
```

```
"Node: central2"
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet#
root@antonio-VirtualBox:~/mininet# hping3 -1 --flood -a 157.0.0.1 157.0.0.8
HPING 157.0.0.8 (central2-eth0 157.0.0.8): icmp mode set, 28 headers + 0 data by
tes
hping in flood mode, no replies will be shown
^C
--- 157.0.0.8 hping statistic ---
2613579 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@antonio-VirtualBox:~/mininet#
```

(**a**)



```
ip.src == 157.0.0.8
No.  Time         Source     Destination  Protocol    Length  Info
  59 1.325363181  157.0.0.8  157.0.0.22   Modbus/TCP  72      Query: Trans:    0; Unit:   1, Func:   3: Read Holdin
▾ Transmission Control Protocol, Src Port: 49100, Dst Port: 502, Seq: 1, Ack: 1, Len: 12
    Source Port: 49100
    Destination Port: 502
    <Source or Destination Port: 49100>
    <Source or Destination Port: 502>
    [Stream index: 3]
    [TCP Segment Len: 12]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 13    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    Header Length: 24 bytes
  ▸ Flags: 0x018 (PSH, ACK)
    Window size value: 8192
    [Calculated window size: 2097152]
    [Window size scaling factor: 256]
    Checksum: 0x77c3 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▾ Options: (4 bytes), Window scale, End of Option List (EOL)
    ▸ Window scale: 8 (multiply by 256)
    ▸ End of Option List (EOL)
  ▾ [SEQ/ACK analysis]
      [Bytes in flight: 12]
      [Bytes sent since last PSH flag: 12]
    [PDU Size: 12]
▾ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 6
    Unit Identifier: 1
▾ Modbus
    .000 0011 = Function Code: Read Holding Registers (3)
    Reference Number: 0
    Word Count: 10
```

(**b**)

Figure 2. Cyber-attacks: (**a**) Representation of a denial of service using Hping3; (**b**) Illustration of a false injection ModbusTCP packets using Scapy.

On the other hand, false traffic can be generated using tools such as Scapy. For example, a Modbus TCP packet could be crafted (field by field) and injected into the network. These actions might allow hackers to operate different elements, just by using different TCP payloads, as well as different origin/destination IP directions.

Here, SDN offers greater capacities than simulators in general, and differs from other powerful emulators such as GNS3 by its easy network management without requiring the configuration and dependence on multiple virtual machines running as individual entities of the network. SDN offers a straightforward environment to produce simple or complex topologies and allows the student to build networks from a cheap and effective manner. Many open source routers available to be configured and included as part of GNS3, such as Quagga [11] or VyOS (compatible with Vyatta) [12], demand additional resources (more virtual space), hindering the development of practical works; whereas SDN centralizes all the network control in programmable controllers (see Figure 1 on the right side) where students can be able to interact with the own network at a low implementation cost and hardware resources.

3. Comparison of Simulators and Emulators for Education

As we have seen, most simulation tools (e.g., OMNet++, CISCO Packet Tracer, NS3) and emulation tools (e.g., GNS3, NetSim, CORE) could not completely replace the real environment in a learning context, as they did not have support for tools to analyse, for example, network traffic with Wireshark, interact directly with the environment, and perform cyber-attacks through specific Linux commands and applications. A summary of this analysis can be found on Table 1.

Table 1. Comparison of network simulators and emulators in a learning context.

	SIMULATORS			EMULATORS			
	OMNeT++	Cisco Packet Tracer 7.0	NS3	GNS3	NetSim	CORE	Mininet
Open Source	✓	✗	✓	✓	✗	✗	✓
Language	C++	JavaScript/CSS	C++/Python	Python	Java	Python	Python
Last Updated	2017	2018	2018	2018	2018	2018	2017
Suitable for Wireless Networks	✓	✓	✓	✓	✓	✓	✓
Supports 6lowPAN traffic for IoT devices	✗	✗	✓	✓	✓	✗	✓
Supports SDN traffic	✗	✗	✓	✓	✗	✗	✓
Scientific Community Support	✓	✗	✓	✓	✗	✓	✓
Emulates real network interfaces	✗	✓	✗	✓	✓	✓	✓
User-Friendly Interface	✓	✓	✗	✓	✓	✓	✓
✓✗	+1	−1	+3	+7	+1	+1	+7

On the other hand, Mininet, a real-time network emulator based on SDN and OpenFlow, was able to provide an easy use of the environment and configuration of software components, amongst other benefits. MiniNet allowed the dynamic creation of virtually connected networks of host, switches, links and controllers. These emulated hosts worked with Linux kernels, and as the switches support the OpenFlow protocol, it was possible to create a customizable SDN topology with a high level of flexibility. Moreover, Mininet is not limited to wired environments. A fork of Mininet, Mininet WiFi, allows users to emulate wireless networks based on WiFi technologies. Mininet WiFi provides the possibility of building complex WiFi networks using Python and it also enables the configuration of different parameters within the Wireless nodes.

4. Conclusions and Future Work

The main conclusions of this study focus on the beneficial characteristics of SDN as opposed to other existing technologies, and on the potential capabilities to build practical scenarios at a low infrastructural cost, configuration time and effort, in addition to favoring the students to concentrate their time in learning the essential goals of the subject—exploiting their competences properly.

Author Contributions: C.A. and A.O. conceived the study, and developed the simulated networked scenario. All authors participated in the analysis and comparison of the simulators and emulators. C.A. and A.O. wrote the paper, and R.R. edited it.

Acknowledgments: This article also has been partially supported by the "I Plan Propio Integral de Docencia, Universidad de Málaga".

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kreutz, D.; Ramos, F.M.V.; Veríssimo, P.E.; Rothenberg, C.E.; Azodolmolky, S.; Uhlig, S. Software-Defined Networking: A Comprehensive Survey. *Proc. IEEE* **2015**, *103*, 14–76.
2. Chakraborty, A.K.; Sharma, N. Advanced metering infrastructure: Technology and challenges. In Proceedings of the 2016 IEEE/PES Transmission and Distribution Conference and Exposition (T&D), Dallas, TX, USA, 2–5 May 2016; pp. 1–5.
3. Rubio, J.E.; Alcaraz, C.; Lopez, J. Recommender system for privacy-preserving solutions in smart metering. *Pervasive Mob. Comput.* **2017**, *41*, 205–218.
4. Zeadally, S.; Pathan, A.; Alcaraz, C.; Badra, M. *Towards Privacy Protection in Smart Grid, Wireless Personal Communications*; Springer: Cham, Switzerland, 2012; Volume 73, pp. 23–50.
5. Alcaraz, C.: Lopez, J.; Wolthunsen, S. OCPP Protocol: Security Threats and Challenge. *IEEE Trans. Smart Grid* **2017**, *8*, 2452–2459.
6. MODBUS Application Protocol Specification V1.1b. Available online: http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf (accessed on 1 May 2018).
7. Mininet. Available online: http://mininet.org (accessed on 1 May 2018).
8. Wireshark. Available online: https://www.wireshark.org (accessed on 1 May 2018).
9. Ettercap. Available online: https://www.ettercap-project.org (accessed on 1 May 2018).
10. Scapy. Available online: https://scapy.net (accessed on 1 May 2018).
11. Quagga. Available online: https://www.quagga.net (accessed on 1 May 2018).
12. VyOS. Available online: https://vyos.io/es/ (accessed on 1 May 2018).