Evaluation of Cisco’s Virtual Internet Routing Lab (VIRL) as a Cyber Security Research Tool

Christian Taylor

University of New South Wales at ADFA

Enterprise networks currently contain significant threats through misconfiguration and software vulnerabilities, which can be exploited. Network behaviour analysis is essential for the development of policy, technologies and configurations for the detection, identification and prevention of malware, botnets, misconfiguration and unauthorized access on a network. Currently, network emulators and simulators provide a inexpensive and effective means to test new configurations, rulesets and technologies. However, due to assumptions made in the design of models and algorithms used in these systems, often they lack the ability to scale, produce results difficult to replicate on hardware networks or are unable to connect devices to the emulated network for forensic analysis of infected hosts. Cisco’s Virtual Internet Routing Lab (VIRL) provides a viable platform to conduct large scale, cheap, flexible, repeatable and realistic network testing. This project provides three significant and novel contributions to the cyber security and network community. First, is the design, verification and release of an open source test framework to allow for repeatable and standardized network testing, utilizing existing systemic tool and technologies. Second, is the development and release of an open source baseline data set evaluating current generation cisco hardware network devices and VIRL. Third, is the evaluation and cohesive analysis of VIRL as a viable software alternative suitable network testing platform with cyber security and network research.

Contents

I Project Outline 2
A Introduction ................................................................. 2
B Aims ................................................................................ 2
C Requirements ................................................................. 3
D Motivation and Impact ...................................................... 3
E Development Process ....................................................... 3

II Literature Review 4
A Introduction ................................................................. 4
B Virtualisation ................................................................. 4
C The Need For Network Security Testing Platforms .......... 4
D Simulation, Test Beds and Emulation .............................. 5

III Test-bed 6

IV Test Harness 7

V Individual Components Baseline, Network and Architecture Testing 7

VI Experimentation Method 8

VII Results and Analysis 8

VIII Discussion 10

IX Recommendations and Future Work 11

*PLTOFF, School of Engineering & Information Technology. ZEIT4500/4501
I. Project Outline

A. Introduction

Securing enterprise networks is a growing concern to business, academia, governments and industry. Cybercrime was estimated to cost over $1 billion during a 12 month period in Australia during 2013 [1]. Cyber criminals and various threat actors leverage vulnerabilities in a network allowing the delivery of malware and botnets to achieve a malicious aims. The current volume and complexity of new malware and botnet poses a significant challenge for researchers and industry. 431 million new malware samples where detected in 2015 [2]. Adversaries are now countering current analysis tools and techniques using by detecting virtual machine environments and obfuscating binary code, increasing the time taken to analysis and characterise malware. The development of signatures and behaviours of malware and botnets normally requires isolated hardware testbeds to be created. Due to their complexity and size, replicating enterprise networks in a dedicated hardware testbed is cost prohibitive to build and maintain. A number of research tools currently exist to simulate large scale networks, however, many lack the realism of production enterprise network environments. This is due to assumptions made in the software models to reduce the complexity of the problem.

The development of new research analysis techniques and tools utilizing large scale network emulation is required to counter the advances made adversary’s in the development of malware and botnets. VIRL provides a scalable network simulator platform allowing enterprise networks to be virtualised for design and configuration testing purposes. Although, not strictly a dedicated analysis tool for malware and botnets, it has the ability to form part of an overall heuristic security process to analysis a network as an entire system. The ability to connect virtualised networks to hardware and software devices provides a greater level of flexibility and fidelity in the analysis of botnets and malware than current simulators allow. This project applies engineering best practices to evaluate what extent can VIRL, a software defined networking platform, be used for large scale automated testing and simulation for cyber security. To achieve this it must be tested against a hardware network for its functional realism, scalability, reliability, repeatability, validity of network data generated, limitations and cost effectiveness [3–5].

B. Aims

The aim of this project is to evaluate to what extent can VIRL, a software defined networking platform, be used for large scale automated testing and simulation as a research tool for cyber security. To achieve this, the capabilities and limitations of VIRL will be characterised and demonstrated through experimental testing. The testing criteria will focus on the functional realism, scalability, reliability, repeatability, throughput, validity of network data generated, limitations and cost effectiveness of a virtualised network compared to a hardware network. This provides metrics against which VIRL performance will be compared. Meeting these criteria enables future research into the use of VIRL in an overall heuristic network security analysis tool. The project was divided into four distinct stages to achieve the aim. These are Test-bed Design, Development and Testing, Test Harness development and Testing, Baseline of Individual Hardware and Software Network Components and Basic Network and Underlying Hardware Infrastructure Testing. Each will be independently outlined and discussed.
C. Requirements

The development of project requirements were driven by engineering best practices through the literature review, design and requirements engineering processes. The evaluation of VIRL as a viable network security research platform is assessed via qualitative analysis of the functional realism, scalability, reliability, repeatability, ease of use, validity of network data generated, limitations and cost effectiveness. Through the key metrics of jitter, delay and throughput across a variety of network protocols, commonly utilised in enterprise networks, allows for the quantitative analysis of VIRL. These key criteria are acknowledged, within the academic community, as key traits which a network simulator should be assessed against [3–5].

D. Motivation and Impact

There is a significant need to create and evaluate new analysis tools and techniques to characterize the normal behaviour of large scale networks. This allows for the development of signatures and detection models to detect abnormal behaviour which can identify the use of malware, botnets, unauthorised network access or network faults. The increased rate of the production and complexity of malware, has created an extremely reactive approach in the development of analysis tools and models, to better understand the problem. To address this issue, network security requires a systemic approach viewing the system as a whole, rather than the individual system elements[6]. The utilisation of models, simulators and emulators to replicate the current state of a production network has numerous advantages. Simulations provide cost effective, scalable, repeatable testbeds for the analysis of malware and botnets. To achieve this the models often are simplified or designed to solve specific problems resulting in a lack of fidelity with results which are networks.

Hardware testbeds allows for repeatable experiments to be undertaken providing realistic data however physically duplicating networks, and associated devices is extremely costly and time consuming making it impractical especially for large networks in many circumstances. Additionally the ability to connect real physical hosts into the virtualised network allows for forensics analysis to be performed on the host without infecting further hosts on the production network. VIRL allows the creation of virtualised networks and connection of hardware devices to the network. This allows the reproduce hardware enterprise network for security analysis testing. This project will evaluate VIRL as a networks security analysis tool for use as an element inside a systemic level approach.

Although testing of other networks has been conducted previously, there is no standardised open source framework or data set to compare network simulators and emulators against hardware devices. Additionally there no papers or research into capabilities of VIRL as a cyber security testing platform for its use in cyber security and network research. This project has generated a advanced, nuanced understanding into the capabilities of VIRL and identified limitations in the use cases for which it would be used for. Additionally, this project has produced a test harness, utilising peer review and verified DITG, that not only mimics common C&C botnet, but also provides repeatable, standardised output which tests the VIRL against the project requirements. This test harness has been developed from first principles and is unique in its approach and will be made available as an open source tool kit for other researches to use. It provides a means by which to conduct standardised testing across all networks, for common network protocol calls, within fixed parameters of duration, packet size, file size and packet rate. This provides a metric by which individual components and the network systems as a whole can be tested, compared and verified against. Finally, this project has producing baseline data which allows for the direct comparison and analysis of current generation network devices and VIRL. This will aid future research in development of a more cohesive response in the conduct of research of malware and botnets.

E. Development Process

Currently there is no standard methodology, framework, programs or platform to compare hardware and software networks. Therefore, a unique and novel testbed and test harness was developed using engineering best practice. The project is broken into four distinct section, outline in the aims, utilising spiral model project management and experimental design methodologies. These discrete steps follow a logical design methodology allowing for the identification and mitigation of risks, ensuring the success of the project. During the development process, significant issues arouse requiring several iterations of design and development stages to be completed to ensure that the requirements of the project would be met.
II. Literature Review

A. Introduction

Network defence relies on the ability to identify, investigate and analyse vectors of attack. There are several intended use-cases for this approach, from monitoring network health through to understanding propagation patterns of malware replication.

From a network defence perspective, there are three methodologies for testing the effectiveness of an attack or defence on a network; 1) simulation, and 2) test beds and 3) emulation [7, 8]. Simulation offers repeatable, cost effective testing especially on large scale networks, but at a cost of model development, fidelity and accuracy. Test beds allow for installation of real operating systems and applications to be tested, however they are cost prohibitive, require physical reconfiguration and sanitation after each test [7, 8]. Test-beds, while being hardware-based systems, are able to be used in conjunction with emulation. Emulators provide a means emulate hardware as software, allows for operating systems and applications test. This allows for the production of realistic results. Emulation may be considered an expansion of test beds, without the need for physical hardware. Emulation may be considered more accurate than simulation, as the emulated systems are real.

Within the academic body of knowledge, the performance of numerous simulators, emulators, models and algorithms is well documented. Currently most of these software packages model specific elements of networks behaviour testing for certain network protocols, topology implementations, technologies or for specific attack vectors[4]. The assumptions made within these software models can produce results which are unable to be replicated on an enterprise networks, lacking a sense of realism[4, 5]. A limited number of papers have attempted to propose emulators and simulators where users are unable to differentiate between hardware and virtualised network environment. Less have the capacity to connect physical hardware devices to the software network.

This section discussed the current state of art for simulation, emulation and test bedding of enterprise networks, virtualisation of servers and networks, from a computer network defence perspective. Critical analysis of the current research will be conducted to establish the viability of using VIRL as a network security research tool for enterprise networks.

B. Virtualisation

Virtualisation is a technique by which multiple virtual versions of operating systems or hardware platforms operate from a share pool of hardware resources [9, 10]. Logical partitions are created around each virtual machine allowing the allocation of resources required to run the virtual instances to be controlled. Virtualisation benefits include improved system security through isolation, reliability, flexibility, availability, efficiency, reduces costs and greater performance monitoring. [9, 10]. When powered on, servers without virtualisation are in use only 5% to 15% [10]. Operating multiple VMs on one hardware platform reduces the number of physical devices, infrastructure and power normally required allow for resources to be located and administered in a centralised data center. Virtualisation enables the creation of snapshots to save the current state of the VM, allowing the establishment of a configuration baselines. Snapshots reduce the deployment time of multiple instances server with the same configuration. In forensic analysis of malware, snapshots ensure that the virtual machine can be restored to a sanitised state without the need for re installation of the operating system and drivers every time a test is run. Server, network, desktop and application virtualisation are the main four categories which will be used during the engineering project.

C. The Need For Network Security Testing Platforms

Malware is malicious software designed to carry out the harmful intent of an attacker[11, 12]. Its purpose is to gain access to computer systems and network resources to gather personal information without the owner’s consent[12]. There are several variations of malware, including Spyware, Adware, Virus, Rootkit, Worm, Trojan-horse, Backdoor, and Botnet. the notable trend across all forms of malware is an increasing complexity. Current generation malware has several properties designed to maximise the efficiency of its attack and also to minimise the chance of discovery and mitigation.

Analysis of malware systems is becoming more difficult[12]. There are two forms of malware analysis; static and dynamic. Static analysis examines the software without executing the code. The code is unpacked and pattern detection is observed in the byte-sequence n-grams, syntactic library calls, string signature, control flow graph and operation code frequency distribution through the use of a disassembler/debugger and memory dumping tools to reverse engineer the code[12]. Dynamic analysis observes
the behaviour of the malicious code when it is executed in a controlled environment such as a VM to ensure isolation from the host systems. Process, file system, registry and network calls are monitored on the VM to all the function call monitoring, information flow tracking, function parameter analysis and instruction traces[12]. Malware authors have responded by developing VM identification techniques; if the malware detects it is operating in a VM it will minimise or alter its behaviour. [13]. If binary obfuscation has been used and the Anti-VM malware code is present, both passive and active analysis will be ineffective. Therefore for analysis to be conducted on a emulated network, hardware hosts will be required to be need to be connected for analysis.

A second technique used by malware authors to minimise exposure during propagation is the use of selective network paths to propagate. This may be a timing technique to search for vulnerable computers over a long period of time, or to only search for and infect computers that are likely to yield the highest chance of success [12]. An understanding of network propagation is therefore necessary to develop an understanding of how a malware sample propagates through a network and any internal restrictions it has.

Botnet detection is another use of network security testing platforms; used as a means of monitoring and developing network traffic samples of C&C traffic and attacks. Botnets are a network of compromised computers (bots) controlled by an intruder (botmaster)[14]. Commonly they are used by cyber-criminals to conduct malicious activities such as DDoS attacks, conducting click-fraud scams, stealing sensitive information, hijacking/exploiting computational power, spreading spam email and to infecting other computers[14, 15]. Once established, Botnets are rented out through the cyber criminal black market for a range of attacks. In 2009, prices ranged from $50 to thousands of dollars dependent on the size of the botnet required for DDoS attack over a 24 hour period[16]. The main architecture for C&C of botnets is centralised, decentralized or a hybrid meshed combination. Communication to the C&C server occurs over legitimate channels such as IRC, P2P and HTTP. Detection is typically accomplished through either passive network traffic monitoring and analysis or honeypots. Passive monitoring relies on signature based and anomaly based detection on network traffic flows. Honeypots are deliberately vulnerable machines attached to a network with no production value which are monitored for suspicious traffic[15].

D. Simulation, Test Beds and Emulation

Understanding how physical hardware networks operates is a fundamental requirement in the development of technologies for network defense. Simulation is an attractive option as it allows for scalable, controlled, cost effect, repeatable testing of a problem.

Simulation, however, relies on the interaction between developed models. When developing models, there is an inherent bias in the fidelity and accuracy of a produced model. By its nature, a model is an approximation, not the real artefact [4]. In practice, simulators focuses on one type of problem or protocol with c-BGP [17], OverSim [18, 19], Network HTTP Simulator(NHS)[7] and Nessi[20] exemplifying this approach. Data produced from each of these types of simulators designed to understand a specific system and cannot be made more generic without rewriting underling code. None of these systems are currently able to effectively model any of the potential use-cases outlined above. They are therefore unsuited for use as cyber-security tools in this context.

Hardware testbeds provide functional realism, but are cost prohibitive to purchase and maintain for large scale networks. Each new experiment undertake requires re-cabling adding to time and potential errors [4]. There is both a large infrastructure cost and an maintenance overhead. When coupled with specific tests that may involve the need to reset systems to default configurations between testing runs, the outcome is a slow, expensive operation.

Emulation provides a trade off between the extremes of simulation and test beds by allowing hardware devices, such as routers, switches, servers and network security appliances, to be run as VM on physical servers. [4, 5, 21]. The time and cost of creating a VM is significantly reduced compared with configuring and cabling physical network devices[22]. As emulators use real operating systems, results and configurations developed during testing can be implemented on a production network. Additionally, results are valid, as emulation uses the same underlying system.

The limitations of emulation are often found in performance, as emulation imposes additional layers of complexity and processing requirements [5]. Additionally emulators may be unable to utilise specific vendors operating systems due to compatibility or legal issues. For example, GNS3[23] and Mininet HI-6[5] may be capable running the Cisco IOS, to do so would break the terms and conditions of that licence[24]. Cisco have, in 2015, released their own emulator environment, Cisco released VIRL for use by academics, researchers and students and CML for corporate customers. A current limitation of VIRL and CML is that only a selected number of Cisco IOS's are able to be emulated[25, 26].
The low cost, volume and complexity of malware and botnets is significantly increasing pressure on research and network security experts to develop novel approaches to analysis, identify, detect and classify them in a timely fashion. The use of simulators, emulators and purpose-built testbeds is a common practice used within research for the analysis of network security. The utilisation of virtualisation enables isolation from the hardware hosts however malware is not being designed to counter this technique. Although they allow security issues to be analysed, the data produced by these systems lacks a level of fidelity and realism due to the assumptions and simplifications used in the models. The development of a systemic approach to the system as a whole is required to embed security within the network. The characterisation of VIRL aims to form a baseline for the effective use within a systemic approach. To date, VIRL has only been used as a teaching tool. There is an opportunity to understand to what extent it may be a suitable cyber-security network platform.

III. Test-bed

The design and development of the hardware and software testbed focused on the functional requirements imposed by VIRL and the hardware available. For the physical hardware network testbed, Cisco provided a 2900 and 3900 Integrated Service Router and a 3560G Catalyst Layer 3 switch. All devices are capable of 1Gb/s through each of the Ethernet interfaces.

For an accurate measurement of performance of VIRL, the software testbed requires that the underlying physical hardware can be separated from the results. To accomplish this, a single testbed was constructed with the test host VMs on one HP Z600 workstation connected to a Cisco SG300-28 port Gigabit Managed switch through an Intel Quad Port Gigabit PCIe Ethernet Card NIC HP NC364T allowing for a total bandwidth of 4Gb/s backbone. The VIRL VM was run on an initial on Testbed 1 listed in table 1. VIRL requires a large number of logical processor cores and memory to allow for a significant number of nodes to be simulated. To ensure that the only load present on process during the test was from VIRL, all of the test hosts VMs were allocated to the single. The test host VMs where built on a Ubuntu 16.04 image and allocated the same resources as the physical test hosts. A limitation of VIRL is that it is purpose built for VMware virtualisation and require specific builds to operate correctly[25].

As the project aim is to evaluate VIRL for cyber and network security testing, network separation is primary requirement. vCenter allows physical Ethernet ports to be assigned to individual virtual switches and virtual networks, as per Figure 2. The simplified logical block diagram for the test bed is shown in Figure 3. Two key concepts are demonstrated in this figure. First is the use of VLANs, which virtual segregated traffic on a switch into separate broadcast domains. Simply, computers on different subnets are only able to communicate to other computers with ports assigned the same VLAN id, effectively creating virtual barriers on a physical switch. Second is the use of FLAT, a Openstack pseudo-acronym with defined additional meaning, allows external connectivity of layer 2 devices to the internal VIRL network devices. Each of the FLAT networks are assigned to separate subnets and virtual switches, with a predefined VLAN id. The FLAT networks allow the connectivity of VMs and physical devices into the VIRL simulated network.

<table>
<thead>
<tr>
<th>Testbed Name</th>
<th>CPU type</th>
<th>No. of Cores (logical)</th>
<th>CPU Speed (GHz)</th>
<th>Memory (GB)</th>
<th>Memory Speed (MHz)</th>
<th>HDD Speed (MB/s)</th>
<th>Hypervisor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testbed 1</td>
<td>2 x Intel Xeon x5660</td>
<td>24</td>
<td>2.8/3.2</td>
<td>64</td>
<td>1600</td>
<td>300</td>
<td>vSphere 6.0 (ESXi)</td>
</tr>
<tr>
<td>Testbed 2</td>
<td>Intel i7 5820K</td>
<td>12</td>
<td>4.5 Over Clocked</td>
<td>32</td>
<td>2400</td>
<td>540</td>
<td>Workstation 12.5</td>
</tr>
<tr>
<td>Testbed 3</td>
<td>2 x Intel Xeon e5-2660</td>
<td>32</td>
<td>3.40/3.90</td>
<td>256</td>
<td>1600</td>
<td>540</td>
<td>vSphere 6.0 (ESXi)</td>
</tr>
</tbody>
</table>

Table 1: Hardware specification of each testbed used across all experiments

Figure 2: vCenter Virtual Switch
IV. Test Harness

The developed test harness is a Linux bash script which utilises DITG heavily for the transmission, receiving and analysis of the network traffic. The design of the test harness is a unique and novel approach, which required the utilisation of engineering best practices, experimental design and an in-depth knowledge of networks and botnets, obtained through extensive research. Iterative design and development was employed, building up from testing one-to-one communication to many-to-one communications, advancing to complete automation of protocol testing for an unlimited number of senders. At each of development the test harness reviewed to ensure it meet the project requirements. The test harness operates in a similar fashion to a current generation botnet whereby a central controller listens for senders (bots) to transmit their IP address. The controller then waits for a predetermined time, generates the scripts to be executed and transmits commands to the senders through SSH. In this case the controller uses \texttt{ITGRecv} and the sender \texttt{ITGSend} from DITG to generate and log the network flow. The test harness automates the collection of the log files from the sending hosts, develops a logical map of the testing hierarchy based on the network setup, allows single or multiple host throughput tests and controls the testing sequence. The test parameters for the protocol, duration, packet size, packet rate, and file size are drawn from a configuration file which are assigned through a configuration file. Significant testing and development went into the creation of the test harness, to ensure that the results from obtain during the experiments were accurate. Each function inside the test harness was tested against a known quantity (hardware cisco switch with 2 physical hosts connected) and compared to each test parameters required of the specific test. The test harness fulfills the key metrics to be observed of throughput, delay and jitter allowing the quantitative analysis of the system. Additionally, the Test Harness performs incremental increases the load placed on the device being tested allows for the failure modes of the devices to be identified. This provides a means by which the qualitative analysis can be conducted utilising the error logs for further analysis.

V. Individual Components Baseline, Network and Architecture Testing

To allow for an accurate comparison between hardware and software, each individual component must undergo testing to provide a baseline for comparison. The baseline consists of testing the throughput, delay and jitter of each component. This is achieved through testing TCP, UDP and ICMP traffic with increasing packets sizes and packet rates. The selection of the three protocols was based on their fundamental properties. TCP ensures that the transmitted packet is received and will retransmit the data if corrupted or lost in transit. This provides a reliable jitter and delay to be established for each device. UDP transmits as best effort and ensures that a steady throughput is maintained without error correction. This provides a reliable throughput to be established for the device. ICMP is used to interrogate a device to test connectivity and is commonly used in denial of service attacks. The number of VM test hosts was restricted to three to allow for an accurate comparison. This is due to the number of Ethernet ports available on the physical routers. The analysis of this data drove intelligent design and engineering decisions in the optimization of both the test harness and test bed.
VI. Experimentation Method

The experiment method was designed to ensure that the project requirements can be tested against the defined set of metrics. As such a baseline must be developed of both current generation network equipment and VIRL. The metrics that these test will measure will be throughput, delay, jitter, reliability and stability. The test also test the limits of VIRLs to identify limitations not previously encountered. The three experiments listed below

Experiment 1: Baseline Network Components

To provide a baseline data test set to allow the comparison of hardware to software the throughput, jitter and delay across TCP, UDP and ICMP was tested. Each network devices was tested individually to provide a baseline characteristic. The device test order consisted of the physical and virtual hosts, switches and routers. The hosts are directly connected to one another utilising one-to-one communication. The switches and routers are tested in a similar fashion initially one-to-one then multi-to-one configuration. The maximum number of hosts connected is limited to three hosts in total due to the physical port limitations of the routers. This testing methodology reduces the background noise and unwanted traffic types on the network. At the conclusion of this experiment, each hardware and software network device will have a baseline data for the significant network protocols.

Experiment 2: Basic Network Topology in VIRL

The basic network topology testing, in Figure 4, builds from the basline network components test. This test is design to place further strain on to the VIRL VM and evalutes the effect on the TCP, UDP and ICMP throughput as additional network nodes are added. At the conclusion of this experiment, a base line dataset for the basic network topology for both hardware and software networks will be produced. From this, the analysis and comparision can be conducted.

Experiment 3: Underlying Hardware Architecture for VIRL

VIRL is heavily based on the ability of the CPU to run the simulation. As such, disconnecting the throughput results from a single CPU architecture is essential. This experiment utilises Testbed 2 and Testbed 3 to run the VIRL VM using the same basic network topology in Experiment 2. Each throuhput test is conducted allow for the development of a baseline dataset for comparision and analysis.

VII. Results and Analysis

The analysis of TCP throughput, from Experiemnt 1, is representivity of the perform each device type for UDP and ICMP. In Figure 5a, the TCP throughput for the physical and VM hosts are almost identical. The virtual switch back plane performes slightly better and can be attributed with the optimised method vSphere controls network traffic with the physical host. Figure 5b compares the TCP throughput of the cisco 2900 router with the VIRL IOSV router. The performance of the VIRL router was several orders of magnitude lower with the average TCP throughput of 2.5MB/s compared to 938MB/s for the physcial router. The TCP jitter and delay, Figure 5c, was significantly higher for the VIRL IOSV router compared to with the cisco 2900 router and the virtual hosts. As the simulation is completely CPU based, the increased delay would account for the lower throughput. The delay performance of the cisco 2900 was significantly better and is unsurprising as the hardware is optimised for network traffic. From these results, it is evident that a direct quantitative comparision of VIRL, against hardware network devices, is not achievable.

The basic network testing of TCP, UDP and ICMP throughput, Experiment 2, for VIRL are shown in Figure 5d and Figure 5e. The addition two extra nodes into the simulation reduce the average throughput by 4% (0.1Mb/s) for TCP and 5% (0.125Mb/s) for UDP. These figures, however, are within a margin of error of 5% and would require significantly more test iterations to provide a better sample. The ICMP test remained stable compared to the Experiment 1 results, however, significant and repeated errors
occurred when the packet rate was increased over 1 million packets per second. The error resulted in the VIRL IOS router to stop passing traffic and reported memory errors in the terminal window. This results suggests a possible vulnerability in the Openstack software though buffer overflow.

The performance of VIRL was marginally improved through using modern CPU architecture. Figure 5f, Figure 6a and Figure 6b show the results from Experiment 3. ICMP showed the most significant gain with the throughput an almost doubling of throughput. UDP averaged performance was not significant improved or diminished. An increase in both logical processors and memory to the VIRL virtual box did not see an over increase in performance however it does allow for more virtual nodes to be run at the same time.

Overall, VIRL provides a reasonable level of scalability and realism when used for networks requiring low throughput. Repeatability, realibility and stability are compromised with very high packet rate, due to the ICMP buffer flow issue, however, for low packet rate applications VIRL is a viable option. Although not an open course tool, VIRL does a significant functionality with its autonetkit feature which allows for the development with large complex networks without the need to program the routing protocols on each router. This functionality is not available in other simulators.
VIII. Discussion

There are two components of this work that require discussion; the testing harness developed to perform the experimentation, and the results of the experimentation as they pertain to the suitability of VIRL as a cyber security platform. The test harness created for this work operated well. This represents a novel development as there is no other testing harness designed to operate across both hardware and virtualised systems. There is opportunity to expand this work, and this is discussed in section IX. The testing harness met the required goals, and this can be verified with the produced datasets.

There is no simple metric that can determine the suitability of VIRL as a cyber-security platform. Suitability of the product for the system itself, its intended use-case, and the nuances of a deployment, is too variable to be reproduced to such a generic response. These experiments, however, highlight the types of work for which VIRL would be beneficial, and types of experimentation where it would be unsuitable.

The experiments outlined in section VI have discovered several limitations within VIRL that would limit its suitability in some contexts. The most significant outcome was the throughput of 2.5MB/s for VIRL was significantly less than the physical devices 938MB/s. This strongly limits the potential use-cases for VIRL in circumstances where any meaningful throughput is necessary, or even probable. For the lower test ranges of packet rate and size, VIRL jitter and delay matched the physical devices. This strongly suggests that the emulated devices are similar to their hardware counterparts, and it would be unlikely that even a VM-aware malware system would be able to detect the emulated devices as such.

This work also identified several undocumented bugs that present themselves during the initial setup and load testing of VIRL. The layer 2 connectivity of the VM hosts to VIRL compared to the physical device to a switch has additional layers of overheads and does not accurately reflect the physical network. VIRL requires that the host VMs connect through the FLAT interfaces which are connected through the virtual switch in vCenter. When communicating between hosts on the same subnet, the traffic does not go through the VIRL switch, when directly connected, instead is switched inside of vSphere virtual switch. Apart from degrading performance, these bugs impose network connectivity limitations on how networks are constructed. This limits the possibilities of what networks and devices can be connected and tested together.

This work also discovered stability issues and undocumented default settings that cause instability within the platform. These findings are qualitative, but strongly suggest that the management overhead of using VIRL is more significant than previously expected. VIRL itself is complex, under active development and the system itself is often at odds with outdated documentation. Stability issues where also encountered during high packet rate testing of ICMP. The sheer flood of packets overloaded the memory on the VIRL IOSV router and caused a buffer overflow to occur. As the underlying infrastructure of VIRL is Openstack, the testing has uncovered a potential exploitable vulnerability. This is obviously undesirable in a security product. the actual mechanisms of the crash is also not graceful degradation or the dropping of packets but a complete system crash.

Currently, the limitations that VIRL currently has in the Layer 2 connectivity does not allow the platform to effectively and completely emulate large computer networks such as this found in government or enterprise. These are based on limitations within the platform itself. However, these limitations do not preclude VIRL’s use as a cyber-security tool; merely that there are limitations to what it can achieve. VIRL can be an effective network virtualisation platform for malware propagation, or in instances where
large-scale throughput is not necessary. VIRL is unsuited to environments emulating enterprise-scale systems, where there is a possibility of large amounts of ICMP traffic, or where high throughput is necessary.

**IX. Recommendations and Future Work**

VIRL provides an array of functionality and would suit selected applications in as a cyber security and network research tool. Although this work has highlighted several areas where it is both suitable and also limited, the platform itself is not a static system. Currently VIRL is undergoing extensive development and it is recommended that a re-evaluation occurs in 18 months, against the same test harness developed for this work. It would be expected that additional development time on this relatively new product would potentially improve the stability and documentation. Additional development may also improve performance, especially regarding throughput. Use of the same test harness will ensure that comparisons are meaningful and comprehensive.

There are also several areas of this project where there is opportunity for additional engineering research. Given that one of the outcomes of this work has been a test harness designed specifically to evaluate hardware and software defined networks, an immediate opportunity would be to expand testing to other, similar or related products. The comparison of VIRL against GNS3 and Mininet would provide a more in-depth understanding of the ability and limitations of the current generation network simulators against hardware. Additionally the evaluation and comparisons of clustered VIRL implementations against Cisco Modelling Labs would further allow the cost effectiveness against capability of VIRL to be assessed, removing the requirement for high end or specialised computer resources required to conduct network testing. The final recommendation would be more specialised, comprehensive testing of botnet and malware across more complex network within VIRL. This testing harness could easily be adapted to allow for more specific forms of testing to assess suitability for niche requirements.

The developed test harness also requires additional engineering work. At the moment, the system is functional but not easy to use or user-friendly. Porting the harness to a modern programming language, such as Python, would allow for rapid integration of existing library modules, making for a more scalable and modular system. There are trade-offs with such an approach, such as added installation complexity, but the increase in functionality and ease of software development would be advantageous.

**X. Conclusion**

This project evaluated VIRL to be a suitable platform for use in cyber security and network research. Experimentation focused on the key metrics of throughput, jitter and delay and the qualitative requirements of scalability, repeatability, reliability, and the validity of network data generated. Qualitatively, use-case limitations, stability and cost effectiveness were also discussed. In performing these experiments, it discovered several undocumented limitations in the platform that place significant caveats on potential use-cases for which it is suited. Specifically, this work has discovered several undocumented limitations within VIRL that preclude its use for either large throughput networks, or in emulating large-scale networks such as those found in enterprise and government.

This work also found that no suitable testing harness existed that could be used across both hardware and virtual machines. This project then had the secondary aim of developing this test harness. This output is also significant as it allows for the testing of additional systems and provides a unique artefact. The outcome of these tests is a dataset. This dataset will be available for researchers wishing to validate results and also the VIRL developers as a means of verifying both the limitations and the instabilities testing has caused. This dataset can also be used to compare similar systems. Significant effort was placed on the development and verification of the test harness to ensure that accurate, automated and repeatable network testing was conducted.

The outcomes of this research indicate that there is a strong need to evaluate and understand software-defined networks, as they are likely to become more prevalent in both corporate networks but also in the field of research testing.

**References**


