Practical Experiences of Building an IPFIX Based Open Source Botnet Detector

Mark Graham, Adrian Winckles, Dr. Erika Sanchez-Velazquez
Department of Computing and Technology
Anglia Ruskin University
Cambridge, United Kingdom
mark.graham@anglia.ac.uk; adrian.winckles@anglia.ac.uk; erika.sanchez@anglia.ac.uk

This paper was presented at Botconf 2015, Paris, 2-4 December 2015, www.botconf.eu
It is published in the Journal on Cybercrime \& Digital Investigations by CECyF, https://journal.cecyf.fr/ojs

DOI: 10.18464/cybin.v1i1.7

***Abstract*—The academic study of flow-based malware detection has primarily focused on NetFlow v5 and v9. In 2013 IPFIX was ratified as the flow export standard. As part of a larger project to develop protection methods for Cloud Service Providers from botnet threats, this paper considers the challenges involved in designing an open source IPFIX based botnet detection function. This paper describes how these challenges were overcome and presents an open source system built upon Xen hypervisor and Open vSwitch that is able to display botnet traffic within Cloud Service Provider-style virtualised environments. The system utilises Euler property graphs to display suspect "botnests". The conceptual framework presented provides a vendor-neutral, real-time detection mechanism for monitoring botnet communication traffic within cloud architectures and the Internet of Things.**

Keywords—IPFIX; Cloud Detection System; Botnets; Property Graphs

I. INTRODUCTION

Infrastructure-as-a-Service (IaaS) style environments provide a low cost, scalable environment from which to host botnets. Botnets have been hosted from within Dropbox [1] and Amazon AWS [2]. Furthermore, IaaS provides a botmaster with a controllable environment. As the botmaster owns this environment, anti-malware detection can be removed and the bot hosted safe in the knowledge it is not going to be detected. This infected environment can then be rapidly cloned to create a sizeable botnet. Where a Command and Control (C&C) server is hosted in a cloud virtual environment, should the server be detected, the cloud makes it easy to spin up a new C&C server elsewhere. A botnet hosted within a cloud service provider (CSP) has the option to either attack a target within the Internet, or turn its focus internally and attack the cloud itself; where potential internal targets include other cloud tenants, or the cloud infrastructure such as storage repositories. Malware has been discovered in the wild with the ability to escape from virtualised hosts [3] and virtualised guest machines [4] [17], thereby possessing the ability to propagate across a virtual environment. Crisis malware [3] was the first malware discovered to specifically attack virtualised environments, such as those that will make up the Internet of Things (IoT).

Packet capture malware detection methods, such as anti-virus software or Intrusion Protection Systems, can present issues for CSPs. Traffic measurement on high speed links requires expensive, powerful, often dedicated devices capable of performing fast packet-processing functions distributed across multiple network locations. Packet capture detection relies on payload inspection to determine the presence of malware. Within CSPs this intrusive method can impact the privacy requirements of tenant data. Additionally, in order to take down a botnet, the C&C server must be identified and eradicated; something anti-virus software is not yet capable of doing. Flow monitoring has become the prevalent method for passive traffic monitoring in high-speed networks [5]. In 2013, almost 80% of ISP and network operators surveyed had technology in their network that is capable of capturing flow [6]. NetFlow v5 and v9 being the most common implementations of flow. To date, NetFlow has been vendor proprietary. With each vendor's variant of NetFlow being slightly different, most flow export installations have been restricted to single vendor platforms. IPFIX was ratified in 2013 (RFC 7011 – RFC 7015) as the standard for flow export. Whilst IPFIX is built upon NetFlow v9, IPFIX can be considered a protocol in its own right, developed to address some of the drawbacks of NetFlow (see section II), such as the protocol's lack of security. As well as being standards based, IPFIX offers other advantages over NetFlow which may be useful for botnet detections; such as customisable templates through Information Elements, built-in IPv6 specifications and bi-directional flows.

As part of a larger project to construct an eco-system for botnet neutralisation within cloud environments such as the IoT, this paper describes the building of the data collection element of the eco-system.

The criteria for this collection element is:

- it must replicate the virtualised environments found within clouds providers who might host IoT-style applications;
- it should be built upon currently available open source solutions wherever possible, in order to allow future software developmental contributions;
- it will feed into a (planned) analysis neural network; which has the functionality for determining malicious traffic from benign traffic and quarantining infected

virtual machines (VMs); either by diverting flow communications based on Software Defined Network (SDN) architecture, or through automatic relocation and isolation of the infected machine [7];

- it should support both NetFlow and IPFIX export, to permit the academic comparison of both protocols in botnet detection.

The remainder of the paper is structured as follows. Section II of this paper reviews some of the limitations of NetFlow when used as the principal protocol to collect botnet communication traffic, and how IPFIX is designed to overcome these. Section III explains the conceptual design behind the novel detection methods presented within this paper, with section IV describing the flow monitoring architecture constructed from this design. Section V describes some of the issues encountered whilst building the environment and demonstrates the output of the built system. Section VI considers some of the limitations of flow-based detection. Finally, section VI reflects on what was learnt from this experience and how the proposed framework contributes towards protecting the IoT.

II. NETFLOW AND IPFIX

In the late 1980s Simple Network Management Protocol (SNMP) was the standard for network management. SNMP was designed to give basic information on a device status such as up/down status and common error alerts. The information available via SNMP was somewhat limited, so syslog was often used alongside SNMP to provide more granular and detailed event information. Unlike SNMP which pulls information off a device, syslog is a push technology which means devices send information without being polled. This makes syslog ideal for logging information about devices, but its unstructured data format makes it slow for querying and reporting. Today, flow export protocols send the same types of data as SNMP traps and syslog, with the advantages that flow export is a push technology which reports highly structured information that is ideal for reporting and querying. The IETF (Internet Engineering Task Force) first published the idea of aggregating packets into flows using packet header information for Internet accounting back in 1991, but this work group was disbanded in 1993 due to a lack of vendor interest. In 1996, Cisco patented a technology based on flow export [8]. In 2002, Cisco released their first commercially available version of this flow export – NetFlow v5. With the release of NetFlow v9 (or Flexible NetFlow) Cisco enhanced NetFlow v5 with support for templates, IPv6, MLPS and VLANs. But NetFlow has some limitations. In 2008 the IETF proposed to create a standard to address these shortcomings. In 2013, the IPFIX standard was ratified as RFC 7011 – RFC 7015 [9] with some improvements over NetFlow.

A. Template Extensions

Although NetFlow v5 and v9 are the most common implementations of flow today, there is no support for any extensions in the template set which allow new field types to be created as needed. Sometimes threat detection needs to understand information that is not available in the NetFlow v9 template. IPFIX was designed as an extensible data model to be used in network security applications with flexibility and customisation front of mind. NetFlow v5 has a fixed template of 18 fields, which confines the NetFlow v5 PDU to 48 bytes. RFC 3954 [10] defines 79 fields that are available in NetFlow v9. Cisco's version of NetFlow v9 defines 104 fields, however these are proprietary and will not necessarily interoperate with other vendor's definition of NetFlow v9. IPFIX enterprise elements can be used to gain more information about flows by introducing new field types, known as Information Elements (IE). RFC7012 [11] does not define the IEs, but states that IANA is responsible for maintaining Private Enterprise Numbers for defining the enterprise elements. IANA recognises 433 official fields in the IPFIX standard, with fields 433 - 32767 available for vendor specific assignment. IPFIX allows a vendor to export whatever layer 2 to layer 7 information they want in a standardised template compatible between vendors. Support of enterprise elements will mean IPFIX will be superior to NetFlow in next generation network monitoring, supporting higher performance for collection and use in analysis tools [12]. IPFIX also allows for variable length fields. NetFlow can achieve similar but with fixed columns sizes, which usually ends up with wasted space with every flow thereby increasing storage requirements [13].

B. Transport Protocol

The default transport protocol for NetFlow v9 is UDP [10]. Unlike TCP, UDP is not reliable, secure or congestion aware. This can lead to UDP flooding when a device is down or experiencing a DDoS attack. Additionally, transmitting flow statistics from the observation point over unreliable UDP can induce loss of measured data. IPFIX allows the transport protocol to be selected from SCTP (Stream Control Transmission Protocol), TCP or UDP. SCTP is congestion aware and is the recommended transport protocol as SCTP allows graceful degradation by selectively dropping exported datagrams under high load rather than overloading buffers. Additionally, SCTP mandates a cookie-exchange mechanism designed to defend against DoS attacks [9]. SCTP can be difficult to transmit over the Internet as some devices will drop these packets due to unrecognised protocol numbers, hence IPFIX supports TCP and binds well to TLS for secure transport, especially over the Internet. Cisco's NetFlow v9 also allow for SCTP support to provide TCP like sequential packet delivery reliability and congestion awareness over UDP.

C. Bi-directional Flow

NetFlow defines a flow as a uni-directional sequence of packets with some common properties that pass through a network device [10]. RFC 5103 [14] allows IPFIX to be extended for the many applications where flow analysis benefits from associating the upstream with the downstream flows of a bi-directional communication. This is particularly useful for separating unanswered from answered TCP requests, such as when a botnet is searching for a peer that is offline. Bi-flow can also determine which party initiated the conversation, which may be useful when studying P2P traffic.

D. Security

When NetFlow was designed it was believed that flow records would be confined to private networks, with collectors and exporters in close proximity. Hence, NetFlow did not impose confidentiality, integrity or authentication requirements on the protocol as this reduced the efficiency of the implementation [10]. This leaves NetFlow v5 and v9 open to Man-in-the-Middle attacks, packet tampering, packet forgery and attacks on the collector. IPFIX implementations must address confidentiality, integrity and authentication; including data obfuscation, for example, through encryption (as outlined in RFC 7011 [9] section 11).

E. Next Generation Networking

NetFlow has a fixed key structure that lacks the ability to monitor more complex network protocols such as IPv6, MPLS and multi-cast [15]. NetFlow v5 and many non-Cisco NetFlow v9 vendors do not support IPv6, whilst IPFIX and Cisco's NetFlow v9 do. IPFIX was chosen over NetFlow v9 when looking at anomalous traffic in IPv6 flows [15], as the ease of extending IPFIX to include additional Information Elements allowed the design of new templates for detecting ICMPv6-DoS attacks, IPv6 extension headers and monitoring IPv6-over-IPv4 tunnelling.

F. Vendor Neutrality

NetFlow comes in many flavours; Cisco's NetFlow, Juniper's JFlow, Alcatel-Lucent's CFlow, Citrix's AppFlow and Huawei's NetStream, to name but a few. Some vendors adhere to the NetFlow v5 structure, others adhere to a NetFlow v9's structure. Cisco further extends NetFlow v9 with their own propriety solutions. As a ratified standard IPFIX allows interoperability between vendors. This is especially important when defining additional IEs for advanced flow-based monitoring systems.

III. CONCEPTUAL DESIGN

The criteria for the built detection element was outlined in section II:

1) Replicating cloud provider virtualised environments

The system design has to incorporate two principles of CSP design: a) tenant isolation and b) tenant privacy. To achieve tenant isolation CSPs utilise virtualised infrastructures which additionally reduces the hardware footprint whilst increasing equipment utilisation. The three most common hypervisor platforms amongst CSP are Citrix's Xen, Microsoft's Hyper-V and VMware's ESXi. Of these Xen Hypervisor is open source. Xen is also very common amongst CSPs and is used by Amazon AWS, OpenStack and Apache's Cloudstack. Xen, Hyper-V server and ESXi are all bare-metal hypervisors which boot directly from BIOS without additional operating system requirements between them and the hardware. Xen also offers *para-virtualisation*. Whilst full HVM (Hardware-assisted Virtualised Machine) more closely resembles the complete hardware isolation of a physical server, CSPs, like Amazon, tend to run para-virtualisation as it is faster when running on Linux OS.

Signature-based packet capture techniques, as used in anti-virus software and Intrusion Detection Systems, rely on packet payload inspection to detect malware. Packet inspection in CSP networks is not an option as it jeopardises tenant privacy requirements. Additionally, forensic detection techniques such as anti-virus serve to protect an individual host (or network) through disinfection. Botnet elimination requires that the C&C server(s) must be detected and taken down. This is not possible through forensic techniques, making traffic-based techniques more suited to botnet detection systems. Traffic monitoring using PCAP captures both traffic header information and payload, thereby beaching the privacy requirement. Flow export captures only packet header information, thereby retaining tenant privacy.

2) Open Source

Already mentioned is the desire to use open source equipment wherever possible as future software will need to be written to ensure interoperability between each element in the overall neutralisation eco-system.

3) Feed into a Botnet Analysis and Recognition AI

The detection test bed will be used to capture network traffic which will undergo a pre-processing function (such as filtering and correlation) before being passed onto a neural-network AI element which hunts for malicious traffic signatures. Should the AI detect malicious traffic, the AI will notify an isolation system which will either dynamically reconfigure the infected network segment via SDN, or undergo automatic VM relocation placement. The SDN network is being developed around OpenFlow protocols supported by Open vSwitch.

4) Support both NetFlow and IPFIX export

One function of the test bed will be to compare NetFlow against IPFIX functionality. As such, the system has to be capable of supporting both. Juniper is one of the few hardware vendors who support both NetFlow and IPFIX. Citrix's AppFlow supports IPFIX and Huawei only support NetFlow. Cisco supports NetFlow on all of its hardware products, and is starting to increase support for IPFIX. As the environment replicates a CSP, virtual switches will be utilised. One of the few virtual switches that supports both NetFlow and IPFIX export is Open vSwitch (OVS) [16]. OVS has the added benefits of pairing well with Xen Hypervisor, is supported in OpenStack and Apache CloudStack, whilst being open source.

IV. FLOW MONITORING ARCHITECTURE

Malware attacks on virtual environments can be categorised [18] as:

1) Intra-VM Attacks

- *Cross VM Side-Channel Attacks* where malware gains information about a neighbouring VM via information leakage through convert signalling channels [19] [20]
- *VM Hopping* allows malware to jump to another VM on the same host [21] [22]

2) Inter-VM Attacks

- *VM Hyper-jacking* where a rogue module, such as a compromised hypervisor, is inserted between the physical hardware and machine operating system allowing an attacker to control the virtual machines on that host [23]
- *Guest VM Escape* allows malware to escape from the guest VM onto the host operating system. Typically the host has root privilege, thereby allowing the malware access to other virtual machines or the network [21]. Cloudburst [4] and Venom [17] were both capable of VM Escape
- *Host Escape* allows malware to jump from the host operating system into a guest VM. Crisis malware (OSX.Crisis / W32.Crisis) [3] in 2012 was the first known malware to do this

The flow monitoring process is a complete chain of events comprising of four stages: packet observation, flow export, data collection and data analysis [5].

A. Packet Observation & Flow Export

The correct citing of flow exporters to detect attacks on virtual environments is important, as all exporters tested for the architecture did not support capture data in promiscuous mode.

This meant that to detect intra-VM attacks, an observation point is required within the tenant's virtual environment. This will not be possible within CSP networks due to tenant privacy expectations. Exporters located in the LAN, for example a hardware exporter on a switch, can detect traffic destined for that switch port, but are blind to intra-VM attacks and requires an observation point on each LAN segment. Likewise, for a software exporter cited on a server. This would capture traffic between VMs distributed across that server, but would also miss intra-VM communication.

All possible locations to cite an exporter were evaluated to understand the visibility at each collection point. It was found that an exporter cited on a hypervisor would export both intra-VM and inter-VM traffic providing maximum traffic visibility for the least number of probes (see figure 1). This does however necessitate an exporter on each server under observation.

B. Data Collection & Data Analysis

Even on a simple test infrastructure, a flow exporter located on each hypervisor captures a considerable amount of traffic. When flow capture is compared with PCAP traffic capture, flow captures exactly the same information as PCAP, minus payload, albeit with a considerably reduced cost of data storage. Under test conditions a 2.9GB file was transferred between two servers. Over this transfer period WireShark collected nearly 3.1GB of traffic (including both the file transfer and background network traffic). Similarly, IPFIX exported the same information, minus payload, in as little as 43KB.

Exported data will be collected on a central server where it will be collated according to export time stamp. Hence, every IPFIX exporter needs to synchronise their internal clocks to ensure time stamp consistency across the monitoring infrastructure. As software IPFIX exporters take their timestamp from their parent server, NTP on the hypervisor host server can be used to synchronise exporter clocks. In addition, as the IPFIX collector takes it's time source from the server, the internal clock in each VM need not be synchronised. Once collated, data unrelated to botnet detection will be filtered (primarily network layer 2 and 3 information such as ARP, IPv6, STP, broadcast traffic) in order to improve the performance of the analysis phase. Filtered flows will be collected in CSV comma delimited format. When viewed as a spreadsheet, the data could be filtered on any of the captured IEs. Source & destination IP address were found to be the primary fields to determine conversation, with other template data adding context to these conversations.

In order to visualise botnet communication, the flow data will be represented as a property graph (see figure 2); where nodes represent IP addresses and edges represent the data flow relationships. Creating relationships from the IPFIX template data allows inter-nodal conversations to be viewed by protocol, port number, direction, etc. Property graphs were chosen primarily because graph databases provide the structure and ability to query relational data. However, property graphs lend themselves to analysis of clustered data and data correlation. Experiments have shown that when utilising property graphs to understand botnet communication, analysing the impact of graph deformation from removing significant nodes is more accurate than analysing the vertices with the most edges [33]. Likewise, the number of edges per node can give an indication to P2P traffic [34]. Property graphs will be manually analysed for indications of botnet communication, however work is planned to implement a neural network-based AI to interpret the data and recognise malicious traffic emanating from infected nodes.

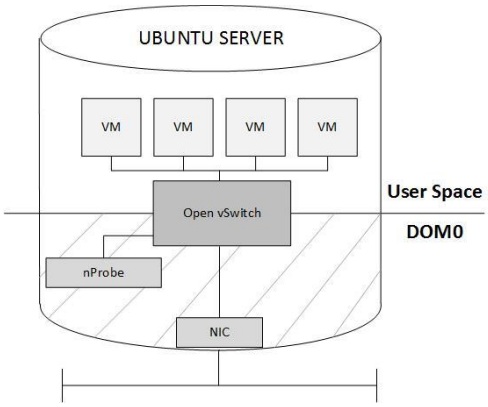

Fig. 1. IPFIX Probe in DOM0

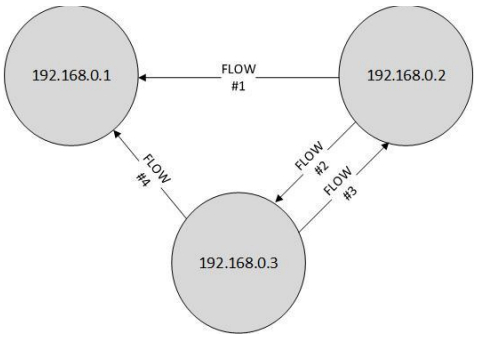

Fig. 2. Property Graph of Botnet Communication

V. RESULTS: LEARNING FROM THE SYSTEM BUILD

A. XenServer

The conceptual framework specifies how the botnet detector will be built upon Xen hypervisor with Open vSwitch as the software switch between tenant VMs. Citrix XenServer is an open source project available as a free download [24]. XenServer 6.2.0 is a self-contained package that includes a Linux-based operating system (CentOS v5.5) and the Xen Hypervisor (v4.1.5) which provides the secure control domain (DOM0) on top of the underlying OS. The Xen Server 6.2.0 package also includes Open vSwitch v1.4.6 and Citrix's XenCentre; a VM management GUI based on the XAPI tool stack. Upon install, the self-contained package boots a simple install wizard which installs CentOS, Xen hypervisor, XAPI tool stack and Open vSwitch. A further manual installation of Citrix XenCentre allows the import of guest OS .iso files into a central repository, after which XenCentre can create and configure guest VMs. Using the XenCentre management GUI, XenServer 6.2.0 provides the control for NetFlow v5 export and "Flows v Time" visualisation graphs. XenCentre would not support IPFIX collection, so an alternative IPFIX analysis tool was required. Open vSwitch supports IPFIX export after release v1.10. The upgrade to OVS v1.10 requires CentOS 5.6 i686 RPM or above. XenServer v1.4.6 shipped with CentOS 5.5 i386 RPM. During installation XenServer partitions DOM0 into 4GB partition of which about 3.8GBs is used by Xen hypervisor. Before any upgrades can be undertaken, this partition must be manually enlarged to 8GB with a re-installation of XenServer. Whilst it may be possible to upgrade CentOS 5.5 i386 to CentOS 5.6 i686, the limited functionality of the cut-down CentOS OS and lack of documentation meant CentOS refused to be upgraded beyond

5.5. It was therefore impossible to enable IPFIX export on the XenServer 6.2.0 platform.

B. Xen Creedence

In 2014, Xen released Creedence Alpha v6.4.94 [25] (with the official release called XenServer v6.5). Another self-contained package, Creedence includes CentOS v5.10 and OVS v2.1.2 which supports IPFIX. Again the DOM0 is partitioned to 4GB but can be extended to 8 GB during installation. Once this network was configured, OVS was set to export the IPFIX format. The IPFIX template was not recognised by XenCentre, nor several other open source collectors and commercial tools. WireShark confirmed that IPFIX traffic was indeed being exported from OVS, but the collectors were not recognising the flow timestamps. OVS was set to sample IPFIX at a 1:1, but appeared to be exporting every single flow without aggregation, resulting in huge amounts of flow data collected in a very short timeframe. All the probes trailed were able to detect OVS exporting NetFlow v5, NetFlow v9 or sFlow, but not IPFIX. It was decided to retain OVS as the vSwitch, but to install an IPFIX exporting probe into the hypervisor. A number of open source probes were tested but failed to install correctly into the hypervisor, possibly due to the restricted functionality of the cut down CentOS operating system. At this stage it was decided to build a bespoke cloud stack.

C. Bespoke Build

A clean Ubuntu 14.04 operating system was installed as the server OS. Into this Xen Hypervisor 4.4 64-bit [26] was installed as DOM0. By default this ships with the XEND toolstack which has limited functionality, so this was upgraded to the XAPI toolstack. Open vSwitch v2.0.2 was installed as the virtual switch. Again OVS appeared to present timestamp and aggregation issues with IPFIX. After testing a number of open source exporter probes, nTop's nProbe [27] was found to export IPFIX templates from the hypervisor. OpenXenManager is the recommended VM management software for Xen hypervisor, however this proved to be buggy when creating new VMs and closing existing VMs. XenCentre proved a more successful management system but some configuration was needed to allow VM consoles to be viewed. Once the final configuration was tested and confirmed to work as expected, repeat installations of the framework onto new servers were taking almost 30 hours. Optimisation of the installation and configuration process of framework was undertaken such that the entire framework would be installed and VMs could be up and running within 3 hours.

D. C&C Botnet

Zeus is a popular malicious botnet that is typically used to steal banking information by man-in-the-browser, key logging and form grabbing. Zeus was first discovered in 2007. Despite its age, new active Zeus C&C servers are still being discovered on a daily basis [28]. One reason for the popularity of the Zeus bot is the easily accessible DIY construction kit, which allows someone with malicious intent to create, deploy and manage a Zeus botnet. The Zeus botnet created for this work was created from the Zeus crime-wave toolkit v2.0.8.9. When a device is infected with Zeus, the bot runs silently in the background giving no tell-tale signs of the compromise. As a C&C botnet, the Zeus bot must periodically communicate with its C&C server for updates, attack instructions or to report back with stolen information. With Zeus, this communication takes place over HTTP (TCP port 80). It is this communication traffic that this research intends to capture and analyse, in order to monitor the progression of Zeus across the virtual infrastructure.

E. IPFIX Template

Section II explained how IPFIX's extensible IE template allows exportation of any layer 2 to layer 7 information in a standardised template compatible between vendors. Figure 3 details the IPFIX template constructed for this architecture, in order to detect botnet traffic in a CSP environment. This IPFIX template contains 9 of the original 18 fields found in NetFlow v5; decreasing the PDU size from 48 bytes in NetFlow v5, to 24 bytes, thereby increasing the efficiency of the PDU. This frees PDU space for the addition of extra fields which might aid in the detection of botnets. In this template *protocol mapping, bi-flow direction, source MAC address & destination MAC address* were added at a cost of 18 bytes. Bringing the IPFIX template to 42 bytes in total compared to NetFlow v5's 48 bytes. Study into creating an optimised IPFIX template specifically for botnet detection has started. This may take advantage of IPFIX's ability to support plugins for additional IE export, such as HTTP, DNS and SMTP fields.

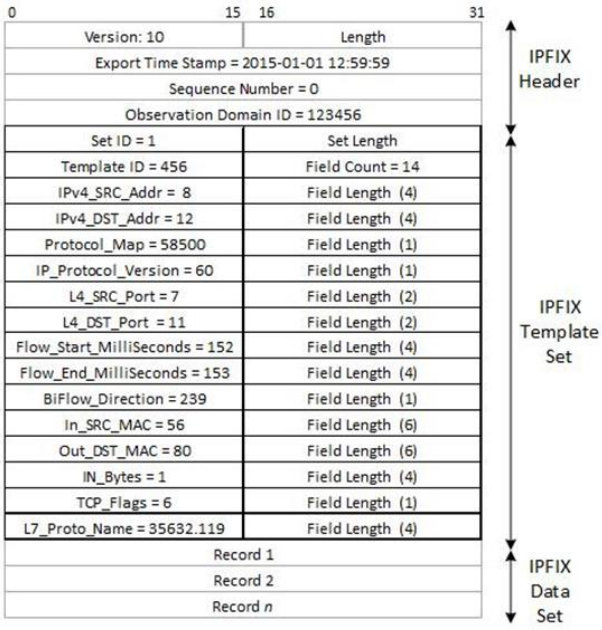

Fig. 3. IPFIX Template

F. Botnests

Figure 4 shows a property graph of nodal communication across multiple ports. Figure 5 shows the same plot of nodal communication, but confined to HTTP (Port 80) traffic only; where the size of the relationship connection is proportional to traffic sent and received. Figure 5 shows some HTTP traffic back to a server, as would be expected. Figure 5 also shows PC's #3, #4, #5 and #8 connecting to PC #7 via HTTP. As PC #7 is a standard PC rather than a server, this suggests irregular behaviour indicating a potential botnest hosting a C&C server. In this instance, PC #7 hosted the Zeus C&C, whilst #3, #4, #5 and #8 were infected with Zeus bot executable. Further granularity can be added to these flows when displayed as in and out traffic captured using the BiFlow_Direction field.

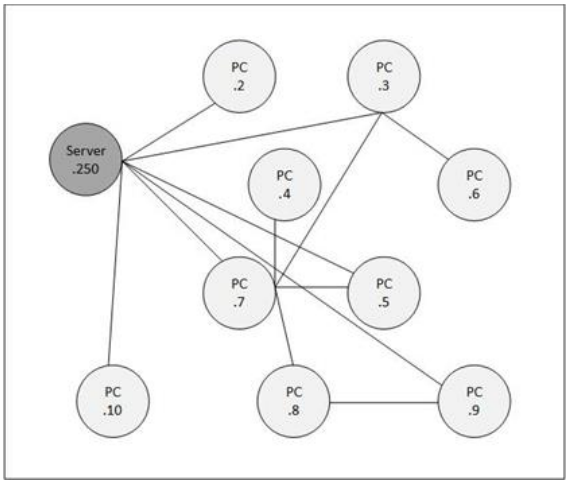

Fig. 4. Property graph of all flows

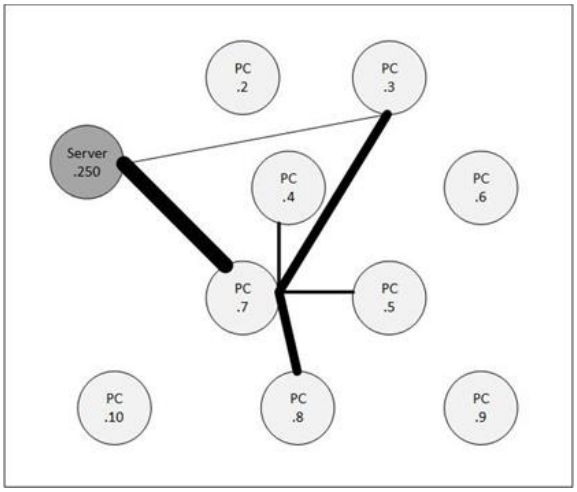

Fig. 5. Property graph of HTTP only flows

VI. DISCUSSION

The template in figure 3 was constructed by exporting Zeus botnet data over the LAN and VM infrastructure and selecting IE fields that either a) changed between malicious and benign traffic, or b) would provide a useful pointer to indicate malicious traffic. Testing is underway to create a template that contains fields that are optimal for detecting a range of botnet malware, which can be fed into a detection algorithm. One build criteria for this architecture was to mimic the privacy requirements found within CSP networks, and thereby assuming that data from within the payload would not be available to such a detection algorithm. A challenge in flow-based detection is obtaining sufficient characteristic data from the packet header. Whilst discarding the payload is well-suited for privacy preservation, it comes at a cost. Namely that many distinguishing malware characteristics are hidden in the payload, such as HTTP URL and HTTP POST/GET information. Many IPFIX collectors support extension plug-ins that are able to capture payload information. Testing has shown that it is possible to export bot characteristic data such as HTTP and DNS information using IPFIX extensions. In turn, this raises the question of where privacy starts and ends, and whether it is acceptable to sacrifice some degree of privacy in order to detect malicious activity. Further testing needs to be undertaken to establish the minimum set of payload data required for botnet detection, and how payload encryption impacts such detection mechanisms.

Section II highlights the protocol transport features and security mechanisms built into the IPFIX standard in order to preserve and protect the data flows between exporter and collector. These include congestion awareness protocols such as SCTP to protect against DoS attacks, and packet obfuscation to protect against flow tampering and MITM attacks. In any detector-based system, the detectors themselves present an attack surface. Whilst this architecture provides partial protection for the probes from their citing outside of the virtual environment, a mechanism is needed to protect the detection system from attack from the LAN, or from malware escaping the virtual environment.

VII. CONCLUSION

Botnet detection using NetFlow protocol is an established concept. Since the ratification of IPFIX as a standard in 2013 academic study has begun to evaluate IPFIX against NetFlow. No academic work could be found that harnesses IPFIX for botnet detection. Several papers claim to do this, but upon closer inspection use proprietary NetFlow v9. This paper contributes to the state of the art by being the first to develop a flow-based IPFIX export framework for use in the academic study of virtualised environments.

This framework was designed specifically for application within cloud provider and virtual environment networks, but is flexible enough to be applied to any LAN environment detection system. As the system is built upon flow export, rather than packet inspection, it is not impacted by payload encryption techniques used by botnets for detection evasion. This framework meets the build criteria as follows:

1) Replicate cloud provider virtualised environments

Xen Hypervisor and Open vSwitch were used to replicate cloud provider virtualised environments. Locating the flow export element within the hypervisor allows collection of intra-VM and inter-VM traffic. This study came across obstacles when trying to utilise the XenServer eco-system for IPFIX capture in virtual environments, choosing to replace some of the elements of the XenServer stack. Table 1 summaries this framework and draws comparisons against the XenServer eco-system for capture and display of traffic communication. In August 2015, XenServer underwent an upgrade to bring the CentOS operating system up to current specifications. It will be interesting to understand whether the update removes some of the obstacles encountered in this study.

2) Utilise currently available open source solutions

This was achieved with the exception of nProbe. nProbe is not strictly open source, but code can be made available to research institutions upon request. Additionally, plugins can be constructed for nProbe that extend the IE template. This framework has been constructed to allow alternative probes or collectors to be inserted in place of nProbe. The suitability of alternative exporters, such as YAF [32], is being studied. Further work is needed to understand why Open vSwitch was not exporting IPFIX timestamps that could be recognised by other collectors and why Open vSwitch was not aggregating IPFIX flows. Overcoming these issues and designing Open vSwitch to support extensible IE template customisation may simplify the framework by enabling the vSwitch to become the flow export/collection element.

	IPFIX Framework	**XenServer v6.4.94**
DOM-0 OS	Ubuntu 14.04	CentOS 5.10
Hypervisor	Xen 4.4 (64 bit) [26]	Xen 4.4 (64 bit)
Hypervisor API	XAPI Toolstack [29]	XAPI Toolstack
Virtual Switch	Open vSwitch v2.0.2[16]	Open vSwitch v2.1.2
Flow Exporter	nProbe v6.15 [27]	Open vSwitch v2.1.2
Flow Collector	nProbe v6.15	XenCentre v6.5
VM Management	XenCentre v6.5 [30]	XenCentre v6.5
Flow Protocol Support	NetFlow v5 NetFlow v9 IPFIX	NetFlow v5 (NetFlow v9)[1]
Flow Traffic Presentation	Neo4J [31]	XenCentre v6.5

Table 1 Bespoke IPFIX Detector compared with XenServer

3) Feed into a neural network analysis element

The aim of this framework is to become the data collection mechanism for part of a larger eco-system for botnet neutralisation within cloud environments. The output data from the exporter is currently converted to CSV format before being filtered and collated in Python, then exported to Neo4j for display and manual analysis. Further work is needed to customise the collector output data to form the input mechanism to a neural network detection system capable of dynamically determining botnet traffic profiles from background network traffic, before feeding into a containment mechanism.

4) Support both IPFIX and NetFlow export

NetFlow v5 exports a fixed template of 18 fields. Whilst these fields are sufficient for capture of network management data, IPFIX (and proprietary NetFlow v9) was designed to allow customisation of the template to capture additional criteria for new applications, such as malware detection. This framework permits export of both NetFlow v5 and v9 as well as IPFIX. Furthermore, the exporter (and vSwitch) utilised can additionally export sFlow and support port mirroring should academic study of these against IPFIX warrant. This work devised a simple IPFIX template (see figure 3) that exports sufficient information to confirm the presence of C&C botnet traffic. Further work is required to design an IPFIX template that is optimised for botnet detection. This may include additional exporter plugins to capture characteristics of the botnets under test, and should be designed for detection of C&C, P2P and IRC botnets. Within a CSP, such a template not only needs to respect tenant privacy, but the template size should be kept to a minimum due to the large number of flows captured over high-speed multi-gigabit environments; thereby reducing storage requirements and improving analysis performance.

CSPs are a crucial building block in the IoT, providing centralised storage and accessibility of IoT data. As the "intelligence" of more IoT devices is migrated into the cloud, dumb endpoints will further reduce the cost of the IoT [35]. As CSP infrastructure is built around virtualised architectures, citing malware detection mechanisms within tenant architectures is difficult if tenant privacy is to be respected. This work demonstrates how flow export detection systems overcome these restrictions by allowing detection functions to be located upon a hypervisor to capture both intra-VM and inter-VM traffic without packet inspection. Now that the IETF has ratified IPFIX as a fully workable standard, flow solutions based upon IPFIX are starting to emerge. As more vendors support IPFIX, CSPs will replace their NetFlow based network management solutions with IPFIX capable next generation networking technology. This standards based approach to flow export allows vendor neutrality and, over time, IPFIX will overtake NetFlow to become the prevalent protocol. This work demonstrates that open source IPFIX technology is of sufficient maturity to create a botnet detection function for real world application.

REFERENCES

[1] Trend, "Dropbox Used in Delivering UPATRE Malware", June 2014. [Online]. https://www.trendmicro.com/vinfo/us/threat-encyclopedia/spam/566/dropbox-used-in-delivering-upatre-malware.

[2] SERT, "The SERT Q2 Quarterly Threat Intelligence Report", July 2014. [Online]. https://www.solutionary.com/resource-center/blog/2014/07/sert-q2-quarterly-threat-intelligence-report/.

[3] Symantec, "Crisis for Windows Sneaks onto Virtual Machines", August 2012. [Online]. https://www.symantec.com/connect/blogs/crisis-windows-sneaks-virtual-machines.

[4] K., Kortchinsky, "CloudBurst", Black Hat USA 2009 Conference, Las Vegas, 2009.

[5] R. Hofstede, P. Celeda, B. Trammell, I. Drago, R.Sadre, A. Sperotto and A Pras, "Flow Monitoring Explained: from packet capture to data analysis with NetFlow and IPFIX," in IEEE Communications Survey and Tutorials, vol. 16, no. 4, pp. 2037-2064, 2014.

[6] J. Steinberger, L. Schehlmann, S. Abt, and H. Baier, "Anomaly Detection and mitigation at Internet Scale: A Survey", in Proceedings of the 7th International Conference on Autonomous Infrastructure, Management and Secuirty, AIMS'13, Berlin, 2013.

[7] R. I. Dinita, G. Wilson, A. Winckles, M. Cirstea and T. Rowsell, "A novel autonomous management distributed system for cloud computing environments", in IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society, IEEE, 2013.

[8] D. R. Kerr and B. L. Bruins, "Network flow switching and flow data export". Washington, DC Patent US6243667 B1, 5 June 2001.

[9] B. Claise, B. Trammell and P. Aitken, "RFC 7011: Specification of the IP Flow Information Export (IPFIX) Protocol for Exchange of Flow Information", 2013.

[10] B. Claise, "RFC 3954: Cisco Systems NetFlow services export version 9", 2004.

[11] B. Claise and B. Trammell, "RFC 7012: Information Model fo IP Flow Information Export (IPFIX)", 2013.

[12] P. Velan, "Practical experience with IPFIX flow collectors", in 2013 IFIP/IEEE International Symposium on Integrated network Management, IEEE, 2013.

[13] M. A. Patterson, "Unleasing the Power of NetFlow and IPFIX", Sanford, Maine: Plixer International, Inc, 2012.

[14] B. Trammell and E. Boschi, "RFC 5103: Bidirectional Flow Export Using IP Flow Information Export (IPFIX)", 2008.

[15] Y. Lee, S. Shin, S. Choi and H. G. Son, "IPv6 Anomaly Traffic Monitoring with IPFIX", in ICIMP 2007 Second International Conference on Internet Monitorin and Protection, IEEE, 2007.

[16] OpenvSwitch, "Open Virtual Switch", June 2015. [Online]. https://www.openvswitch.org.

[17] P. Duckin, "The VENOM 'virtual machine escape' bug - what you need to know", May 2015. [Online]. https://nakedsecurity.sophos.com/2015/05/14/the-venom-virtual-machine-escape-bug-what-you-need-to-know/

[1] Open vSwitch will export NetFlow v9, but XenCentre collects this a NetFlow v5

[18] M. Graham, A. Winckles and E. Sanchez-Velazquez, "Botnet Detection within Cloud Service Provider Networks using Flow Protcols", in INDIN 13th IEEE International Conference on Industrial Informatics, IEEE, 2015.

[19] T. Ristenpart, E. Tromer, S. Shacham and S. Savage, "Hey, you, get of my cloud: exploring information leakage in third-party compute networks", in Proceedings of the 16th ACM Conference on Computer and Communications Security, ACM, 2006.

[20] Z. Wang and R. B. Lee, "Covert and side channels due to processor architecture", in Computer Security Applications Conference, IEEE, 2006.

[21] J. Amarnath, P. Shah, R. Nararaj and R. Pendse, "Security in multi-tenancy cloud", in International Carnhan Conference on Security Technology, IEEE, 2010.

[22] T. Hsin-Yi, M. Sienbenhaar, A. Miede, Y. Huang and R. Steinmetz, "Threat as a Service? Virtualization's Impact on Cloud Security", IT Professional, IEEE, vol. 14, no. 1, pp. 32-37, 2012.

[23] H. Yu-Lun, B. Chen, M. Shih and C. Lai, "Security Impacts of Virtualization on a Network Testbed", in Sixth International Conference on Software Security and Reliability, IEEE, 2012.

[24] Citrix Systems Inc, "XenServer Open Source Virtualization", February 2015. [Online]. https://xenserver.org.

[25] XenServer, "Download XenServer", April 2015. [Online]. https://xenserver.org/open-source-virtualization-download/11-product.html.

[26] Linux Foundation, "Xen Project", February 2015. [Online]. https://xenproject.org.

[27] nTop, "nProbe: An extensibve NetFlow v5/v9/IPFIX Probe for IPv4/v6", March 2015. [Online].
https://www.ntop.org/products/netflow/nprobe/.

[28] Zeus Tracker, "Zeus Tracker", August 2015. [Online]. https://zeustracker.abuse.ch/monitor.php

[29] Linux Foundation, "XAPI", March 2014. [Online]. https://www.xenproject.org/developers/teams/xapi.html.

[30] Citrix Systems Inc, "How to Download and Install a New Version of XenCentre", March 2014. [Online].
https://support.citrix.com/article/CTX118531.

[31] Neo Technology Inc, "Neo4j", April 2015. [Online]. https://neo4j.com.

[32] C. Inacio and B. Trammel, "YAF: Yet Another Flowmeter", in LISA, 2010.

[33] P. M. Collins and M. K. Reiter. "Hit-list worm detection and bot identification in large networks using protocol graphs". In Recent Advances in Intrusion Detection, Springer Berlin Heidelberg, 2007.

[34] M. Iliofotou, H. C. Kim, M. Faloutsos, M. Mitzenmacher, P. Pappu and G. Varghese. "Graption: A graph-based P2P traffic classification framework for the internet backbone." *Computer Networks*, *55*(8), pp.1909-1920, 2011.

[35] M. Dillon and T. Winters. "Virtualization of Home Network Gateways". *Computer*, 47(11), pp.62-65, 2014.