FlowStats: An Ontology Based Network Management Tool

Konstantinos G. Kyriakopoulos, David J. Parish, John N. Whitley
Department of Electronic and Electrical Engineering
Loughborough University, Loughborough, LE11 3TU, U.K.
e-mail: {elkk, d.j.parish}@lboro.ac.uk.

***Abstract*—One of the problems that hinders large scale network management tasks is the number of possible heterogeneous data sources that provide network information and how to focus on a desired network segment without requiring a deep knowledge of the network structure. This work investigates how to intelligently and efficiently refine and manage a vast amount of network monitoring data sources, by using artificial intelligent reasoning through an intuitive user interface. We aim to minimise the user interaction and required user knowledge when searching for the desired network monitoring information by refining the presented information based on user choices. The concept of Ontology is utilised to create a knowledge base of multiple different aspects of our testbed: Internal Management structure, Physical Location of data sources, and network switch meta-data.**

***Index Terms*—Network Management, Netflow, Ontology, RDF, databases**

I. Introduction

With the rapid development of new communication technologies, rise in web applications, expansion in social media and growth in virtual network environments, the nature of computer networks has gradually become more complicated and vast making network management and monitoring a very complex task for network administrators.

In addition, the advances in virtual networks have created environments where sources of computer network related information get spawned and terminated rapidly and instantaneously as the topology of the virtual network adjusts to the needs of the users.

One of the practical reasons that hinders network management tasks is the large number of possible heterogeneous data sources that provide network information. In addition, an administrator might need to focus on a particular scale of the network topology. This scale might range from the top "eagles eye" view to the detailed, per switch, or even per node view. Currently, to achieve this with conventional tools in large networks a deep and thorough understanding of the network topology is required in order for an administrator to isolate a particular data source of interest in the network.

This work investigates how to quickly and efficiently manage a vast amount of network segments, by using artificial intelligent reasoning through an intuitive user interface. The purpose is to create a tool (named FlowStats) that allows users to find the right network segment using easy to understand filtering options that apply on the metadata of the network segments (for example, manufacturer and location of switch) and/or their components (such as computer host's owner, hosts' IP address).

An interesting approach in achieving this goal is to take advantage of the relation-based properties inherited in an ontological database. Ontologies, first appeared in [1] as a a means of increasing the level of automation and reducing human intervention.

This paper focuses on using ontologies to integrate meta data information regarding the network components, in particular information from network flow sources (such as Cisco's netflow [2]), with the purpose of creating a user interface for network administrators that is intuitive in refining network segment related information. We aim to minimise the user interaction when searching for the desired netflows by refining the presented information based on user choices. Therefore, using our proposed approach a network administrator would be able to identify the sources of information that are of interest quickly and without a deep knowledge of the network structure.

The paper is organised as follows. A description of background knowledge regarding Ontology and its technologies along with its advantages against relational databases is presented in Section II. Related work utilising Ontologies or Relational databases along with research in using netflow data for network management is presented in Section III. In Section IV, we present the thought process behind building the ontology for our tool. The experimental testbed structure that was used as a proof of concept to demonstrate our tool is described in Section V. In Section VI FlowStats' mechanisms, tools and components are presented and explained. Some usage scenarios of our tool are shown in Section VII. Finally, conclusions and future work are given in Section VIII.

II. Background

A. Ontologies

In computer science an ontology is "a formal explicit specification of a shared conceptualization for a domain of interest" [3]. Ontologies are basically knowledge structures of a particular domain that allow systems to automatically process the meaning of information, deduce implicit information from the explicit, and integrate heterogeneous data bases. Therefore, an ontology can be a very important tool when constructing and querying knowledge based systems [4], [5].

B. RDF, RDFS, SPARQL and OWL

Resource Description Framework (RDF) is a framework that assists in stating *simple facts* and defining relationships among resources using a format of "Resource - Property - Object" that follows the paradigm "Subject - Predicate - Object" (for example "UK is in Europe"). The terms "Subject" and "Object" are known in ontology terminology as "Domain" and "Range" respectively. The Resource and Property are always defined in a URI form, however, the Object can either be a URI or simply a literal (i.e. string, numeric, etc) [6]. RDF can be represented (serialised) in various formats and some of the most common formats are Turtle and XML serialisations .

An extension of RDF is RDF Schema (RDFS), that allows for defining classes, properties, property restrictions and hierarchies of properties and classes in order to represent more *complicated knowledge* models. RDFS is used for defining the *Terminological Knowledge*, also called Ontology, which represents the *generalised knowledge*, i.e. classes, properties, relationships between classes and relationships between classes and properties. *Specialised* knowledge is represented in the *Assertional Knowledge* which is adequately described by the RDF language and represents knowledge about instantiations of classes and their associated properties [7].

The SPARQL (Sparql Protocol And RDF Query Language) query language is used to query RDFS knowledge databases. It is to ontology databases what SQL is to relational databases.

The OWL language enriches RDFS with semantic information. OWL does not define the structure of the knowledge model as RDFS does, but deals with the semantic relationships between classes and their properties. In other words, OWL provides Artificial Intelligence or inference capabilities to the ontology [8]. For example if we have the statements "London is in UK" and "UK is in Europe" then with the help of OWL it can be inferred that "London is in Europe".

C. Advantages of Ontologies

Ontologies have the following advantages in comparison to conventional relational databases:

- *Dealing with NULL values:* Ontologies deal better with the meaning of null values in comparison to relational databases. They can handle the conceptual difference between "existing but not known" and "does not exist" more efficiently due to the usage of blank nodes [9]. One of the purposes that blank nodes are used in knowledge representation is for statements of existence of unidentified elements [7].
- *Relating, merging, and integrating heterogeneous data:* Because Ontology schema is an open world assumption and uses URI's, it allows the answering of queries that require information outside the central database by integrating (linking) data stored in remote locations. On the other hand, in a relational database, because it is based on a closed world assumption, there is no method to reuse the schema outside this database. In contrast to Ontologies, federalised queries can not be answered and the administrator needs to extend the database schema by loading the extra information which is stored remotely [9], [6].
- *Consistency of SPARQL:* SPARQL, the querying language used in Ontologies, is more standardised than SQL. SQL has multiple dialects (MySQL, Oracle, DB2, PostgreSQL) and users can get confused when moving from one dialect to another. In contrast, SPARQL remains the same no matter the implementation[9].
- *Software maintenance:* In SQL applications, whenever we have to extend or modify the data information and structure, the application's source code, database schema and SQL queries need to be adjusted appropriately. In contrast, in SPARQL/RDF only the applications' code need to be modified, leading to much less software code maintenance [9].

III. Related Work

The authors in [10] propose a method for managing network devices of different terminologies, interfaces and vendors, through a single gateway. Ontologies are used primarily for the creation of a unified application information base and as a mapping algorithm for translating configuration commands and terms between different vendors.

In [5], the authors present a network monitoring system that manages heterogeneous network segments composed of different vendors equipment. The actual data measured from the network come from SNMP and pcap tools. However, the ontology, in contrast to our work, gathers the typical 5-tuple information: computer network protocol definitions (TCP, UDP, etc), source and destination IP addresses and port numbers. Therefore, the ontology, is not designed with the purpose of identifying netflow sources or other network switch related meta data but to hold the 5-tuple data taken from network packets.

Regarding the use of relational databases for storage, searching and processing of network related data (most commonly netflow), there have been several studies [11], [12], [13] comparing the performance of relational databases against flat file systems, i.e. binary files locally stored on the hard disk of the netflow collector. Most studies agree that, at least for smaller scale networks a flat file based system (such as nfdump tool discussed later) is faster for querying and retrieving information. It could be argued that in a distributed database scenario, a relational database would scale better without reduction in the performance. The focus of this paper is the usage of artificial reasoning for identifying appropriate netflow sources through intuitive filtering rather than analysing the drawbacks and benefits of using relational databases for storing the actual netflow data.

In [14], the authors present Gestalt, a system that takes advantage of the distributed capabilities of ontologies to create a federated query interface to simplify access to network security data that would otherwise require manual inspection. Specifically, Gestalt uses an ontology to infer types of data sources that are useful for replying to a specific query, identify these sources in order to access them and natively query them,

and, finally, to semantically integrate the results and return them to the user.

IV. Concept of Ontology

One of the most important aspects for our work was to design a well thought ontology, i.e. the Terminological Knowledge that would hold knowledge in such a way as to facilitate a network administrator to manage the network infrastructure.

We designed an ontology combining three different aspects:

- The Internal Management structure of our organisation, Loughborough University.
- The Physical Location of the University's buildings in the University campus.
- Switch Information regarding the users and their computers associated with the switch.

These aspects were chosen to represent possibilities which ontologies could assist with; they are not a definite list.

Internal management structure provides information pertaining to the hierarchical functional structure of our University, but has no information regarding the physical location of the described entities (schools, departments, etc). As can be seen in Fig. 1, there are several hierarchical classes describing the entities in the University's structure: School, Department, Group. There are also properties assigned to link each of the classes. The property "hasSchool" for example has as subject the class "University" and for object the class "School". The inverse of property "hasSchool" is defined as "inUni".

In contrast to the Internal Management structure, the Physical Location ontology is a knowledge model of the physical topology of the University campus without modelling any knowledge from the functional information. As seen in Fig. 1, there is only one class, the "Location" that "contains", or inversely "isWithin", another "Location" class. The property "contains" has been defined as a *transitive* property, which means that if $LocationA$ contains $LocationB$ and $LocationB$ contains $LocationC$ then it is implied that $LocationA$ contains $LocationC$.

Finally, the knowledge base relating to the network serviced by a particular switch, as seen in Fig. 1, includes information regarding the model and manufacturer of the switch, the devices (or nodes) attached to the switch, the associated person with that device, and the device's IP address. Note that the property "hasNetflow" can be applied on anything as it does not have any restriction on its domain. This was designed on purpose in order to have flexibility to assign the property on any class, i.e. School, Department, Group etc. As was mentioned earlier, it is easy with ontologies to extend the knowledge base with additional properties and classes. The ontology was built from the ground up using Protégé [15]. Protégé is a well known tool in the ontology community that assists in building knowledge structures and relationships between the defined entities.

V. Test Bed

Even though we have built an ontology with the aim of managing the University's computer network, we initially tested this in a controlled environment without interfering with the organisation's actual production network. For this purpose, we built a test-bed (Fig. 2) in our laboratory as a first step, that aims to showcase the usefulness of our conceptual tool.

We gather the network traffic information from three netflow sources, each one dedicated to its own associated network. The three networks set up in our test-bed are: a Virtual network, an internal local network in our test-bed, and an external network, which is actually our office's connection to the Internet.

A. Description of the Virtual network

For setting up the virtual network, ESXi from VMware [16] was used as a hypervisor. ESXi runs directly on the hardware without requiring any operating system as it includes essential drivers for booting and interacting with the hardware. ESXi was installed on two distinct physical hosts composing our physical infrastructure that will host virtual machines. In total, there were three virtual machines set up for generating traffic. One virtual machine, acting as a client, was installed on one physical host, and two other virtual machines, acting as client and server, were installed on the second physical host. That ensures that there was traffic both internally in the physical host and between two distinct physical hosts. Fig. 2 shows the logical topology of the virtual network.

For managing the virtual network, vCenter Server was installed as a virtual machine. The administrator can access vCenter Server through a web browser and manage most properties of the virtual network. A distributed virtual switch was also deployed, namely Nexus 1000v [17]. Distributed virtual switches are particularly useful for interconnecting virtual machines hosted on distributed physical hosts. The virtual switch was set up to send netflow data regarding the virtual network to a netflow collector. In order to automatically and periodically start and stop processes to generate data, and therefore make the validation of FlowStats easier, three cron (built-in Linux tool) schedules were scripted: downloading a webpage (wget), securely copying a file over the virtual network (scp) and finally streaming video (VLC) [18].

B. Description of Internal LAN

An internal LAN has been set up using a Netgear GS724T (referred as GS) switch. Similarly to the virtual network, there are two clients communicating with a server and following the same cron job as described above. Because the GS switch did not export Netflow, the switch ports where the clients and server are connected were monitored and softflowd [19] was tasked to compose the netflows and send them to the collector.

C. Description of External Network

The external network is basically the wired network of our research group. The computers connected to this switch belong to real PhD students and staff and do not follow any cron job but represent normal usage with the seasonal and time of day characteristics.

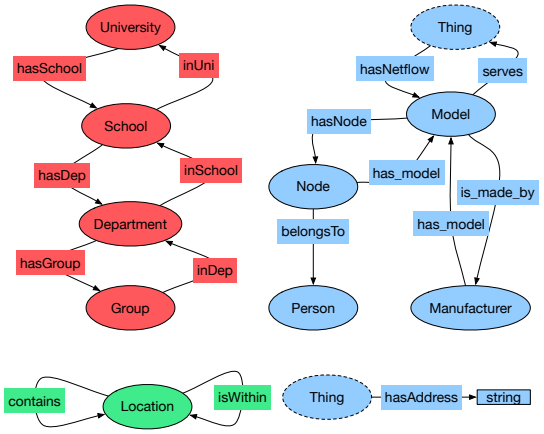


Fig. 1. Terminological Knowledge. In red: Internal Management structure, In blue: Switch related information, In green: Physical Location.

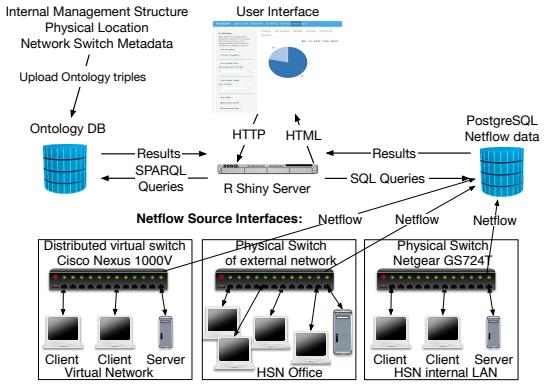


Fig. 2. Network Diagram and FlowStats set up

VI. FlowStats Set Up

A. Assertional Knowledge

As was mentioned earlier, the assertional knowledge describes the specific instances of the general classes in the ontology (terminological knowledge). As can be seen in Fig. 3 (c) we have populated the general knowledge with specific instances to describe our test bed. For example, the instance of Cisco Nexus switch "1000v" (a "Model" class) has the property is_made_by, and the range of the property is "Cisco" (a "Manufacturer" class). Similarly, more information is instantiated for 1000v, including IP address and associated devices. Note that for the Netgear GS and the External switch only the IP and the manufacturer name are assigned, and this will have an effect as will be explained in our case scenarios.

Regarding the Internal Management structure (Fig. 3 (a)), two big Schools have been assigned under "Loughborough University", each School has a Department and so on. Obviously, this is a small segment of the real structure designed just for experimental purposes and to demonstrate the capabilities of artificial inference. Similarly, the physical location knowledge, Fig. 3 (b), maps a segment of our University's buildings into instances of the classes defined in the ontology.

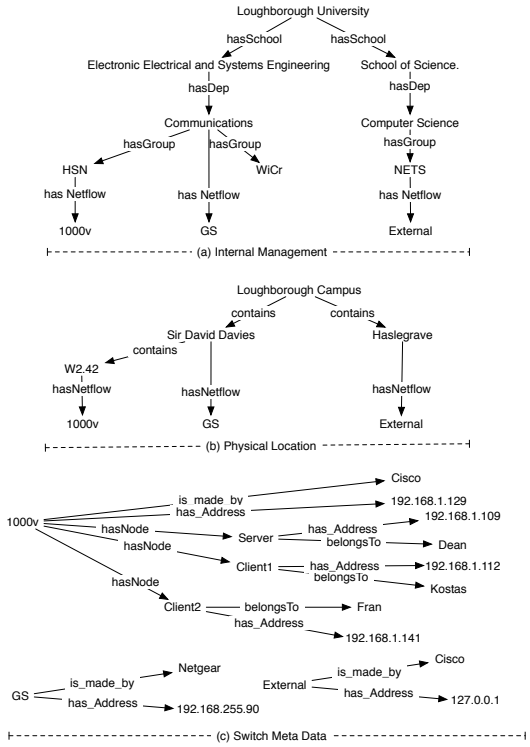


Fig. 3. Assertional Knowledge

The entities HSN, Communications, NETS in the Internal Management structure and W2.42, Sir David Davies and Haslegrave in the Physical Location structure have been assigned a netflow source from our test-bed that hypothetically represent the source of network monitoring data. These associations do not necessarily reflect the real nature of our structure but allow us to illustrate the capabilities of ontology inference without interfering with the University's production network.

B. Netflow Set Up

A netflow architecture is composed of three distinct tools: a sensor, a collector, and a reporting tool. Usually the sensor needs to be attached either to a spanned ("mirrored") port of a switch or be connected to a hub in order to be able to listen to all network traffic that needs to be monitored and construct flows. Once the flow has ended, the sensor transmits this flow data, usually over UDP, to a specific IP address (or more than one address), where a netflow collector listens. In our testbed scenario, softflowd was used as a netflow sensor in the cases where the switch didn't export netflows or where we didn't have access to configure the switch. Network switches and routers are usually able to export netflow data but will need to sacrifice CPU resources in doing so. Using a third party option can be a cost efficient way of generating the netflow data without sacrificing the performance of the switch/router [20].

The collector listens to a specific UDP port where it expects the netflow data to arrive after being transmitted by the sensor. The collector is tasked with writing these netflow data to a

storage device for further analysis if required. In our case,we used as a collector the nfcapd tool (part of nfdump suite of tools [21]), which has the ability to automatically rotate files at the end of an interval (default is 5 minutes). The information in the files is in binary mode and requires special software to read [20].

Some of the software that read netflow files for reporting purposes are nfdump and flowdumper (built-in Linux tool). However, our approach is to sequentially insert the incoming netflow data into a relational database for further analysis and interaction with our website framework.

C. Relational Database Set Up

The netflow capturing tool, nfcapd, has the option to run a particular command at the end of the specified interval, i.e. when a new file becomes available. We run a dameon script, *capture.pl*, as a wraper to nfcapd, to simply execute nfcapd with the above option enabled in order to run a second script, *inserter.pl*, every time the specified interval lapses. In other words, the daemon at boot runs the following command $/usr/local/bin/nfcapd$ -D -x '$/usr/local/bin/inserter.pl$ ... ' (command shortened for clarity).

The second script, *inserter.pl*, takes care of reading the flows from the raw binary files and inserting each flow as a row into the relational database. As a relational database, we used PostgreSQL [22].

D. Website Set Up

The R language [23] was used for creating a script to interact between the user interface, the relational database and the ontological database. For the interaction with the ontology, the R script calls a Python script using the "rdflib" library [24] due to a lack of a native library in R. For building a website to present the graphs, and filtering options, R Shiny was utilised [25]. R Shiny is a web application framework for R that allows for quick development of dynamic web applications.

VII. Test Case Scenarios

FlowStats currently provides 5 main tabs on the top menu (Fig. 4 top). "Dates and Time", "Netflow Filter", "Port Filtering", "IP Filtering", and "Actions". The selections on the first four tabs will affect the results presented in the Actions tab. The Actions tab (Fig. 4 left hand pane) offers some of the basic capabilities of a network management application: Plotting of throughput and showing the top n (where n is user customisable) utilised ports and nodes in the network for the selected time period. The time period is selected in the first tab of "Dates and Time". By default the last 24 hours is selected. However, the benefits of utilising an ontology knowledge base are apparent in the "Netflow Filter" and "IP filtering" tabs.

Netflow Filter is actually a graphical interface to query the ontology and its assertional knowledge. Netflow Filter allows the combination of selections for each of the three distinct knowledge structures, discussed in Section IV, in order to limit choices to Netflow sources that meet the user selected criteria.

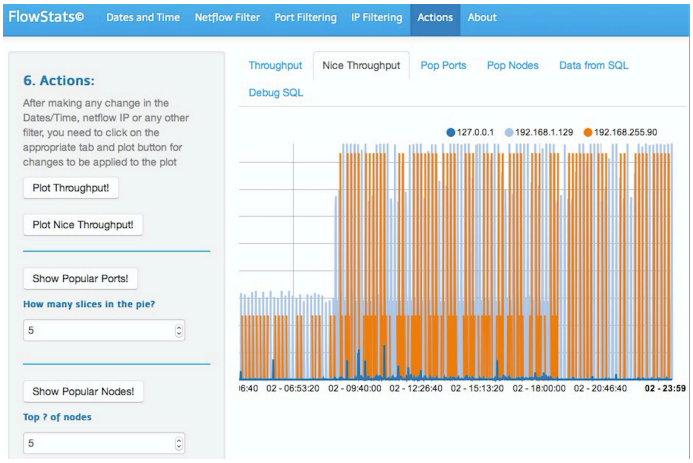
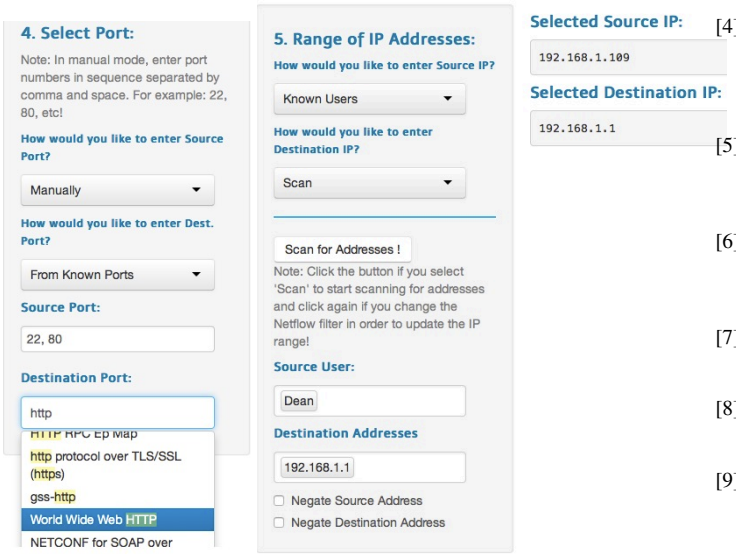


Fig. 4. FlowStats Menu and Actions tab

For example, a user could ask: Show me the most popular nodes in Netgear switches from the School Of Electronic and Electrical and Systems Engineering (EESE). The user simply chooses EESE in the "School" field and Netgear in the "Manufacturer" field and then proceeds in the "Actions" tab to click on "Show Popular Nodes". In the backend of our web application, the system first queries the Ontology in order to first find out which IP address(es) are applicable to be queried with the appropriate Action. As a result only 192.168.255.90 (IP of GS switch) is queried, which corresponds to the Netgear switch belonging in EESE. If the Manufacturer criteria was not set, IP addresses from all manufacturers of switches in the School of EESE would appear, i.e. IP addresses of 1000v and GS (192.168.255.90 and 192.168.1.129) as can be confirmed by combining the information from the tree diagrams in Fig. 3 (a) and (c).

Port Filtering allows the user to specify which network port is of interest. This can either be selected by manually typing port numbers in sequence (Fig. 5 (a): Source Port field) or the user can select port number from a list of known ports (Fig. 5 (a): Dest. Port field). In this tab we can also filter by protocol.

Finally, IP filtering allows filtering by node/IP address. There are two ways to do that, and the user can select which method applies for Source and Destination fields independently (Fig. 5 (b) top menus). "Scan" method, scans through the available addresses that exist in the relational database (Fig. 5 (b) Destination Address field). "Known Users" method, is using the Switch knowledge base, whereby the appropriate fields of Source and/or Destination addresses (see Fig. 5 (b) Source Address field) will automatically list the available hosts by user name, and then proceed to identify this hosts' IP address (Fig. 5 (b) right pane). If instead, netflow sources "External" or "GS" are selected, because no relevant information is entered into the Assertional Knowledge, no available address will be inferred and, therefore, neither listed in the respective fields without, though, causing any problem to the operation.



(a) Port Filtering (b) IP Filtering

Fig. 5. FlowStats Menu

VIII. Conclusion - Future Work

A web based tool for managing networks has been presented in this work that tackles the issue of managing a large number of multiple network management netflow sources using artificial intelligent reasoning and intuitive filtering. Even though there is a plethora of open source or licensed network management tools available on-line, they do not focus on using a knowledge base for the purpose of refining netflow sources in order to proceed with querying those networks for network management statistics.

In our approach, we automatically insert netflow statistics in a relational database, using netflow sources as a method for measuring network traffic. Three different aspects of our infrastructure have been mapped into the knowledge base: a functional structure for internal management, a physical location relationship and switch related meta data.

For future work, we will compare the performance of implementations that natively query for netflows and implementations that use big data databases. We are also thinking of ways to tackle the problem of automatically populating the ontology with relevant information. Ways to do that include accessing log files, e.g. dhcpd.conf files in the DHCP server to retrieve information for the topology of a network and remotely connecting to existing databases that hold key metadata information.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web: A new form of web content that is meaningful to computers will unleash a revolution of new possibilities," *Scientific American*, vol. 284, no. 5, pp. 1–5, 2001.

[2] "Cisco IOS netflow," Website, http://www.cisco.com/.

[3] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge acquisition*, vol. 5, no. 2, pp. 199–220, 1993.

[4] M. Trifan, B. Ionescu, D. Ionescu, O. Prostean, and G. Prostean, "An ontology based approach to intelligent data mining for environmental virtual warehouses of sensor data," in *Virtual Environments, Human-Computer Interfaces and Measurement Systems, 2008. VECIMS 2008. IEEE Conference on*. IEEE, 2008, pp. 125–129.

[5] S.-Y. Yang and Y.-Y. Chang, "A new network management system with ontology-supported multi-agent techniques," in *2010 International Symposium on Parallel and Distributed Processing with Applications (ISPA)*. IEEE, 2010, pp. 275–282.

[6] S. Albahli and A. Melton, "ohStore: Ontology Hierarchy Solution to Improve RDF Data Management," in *In 9th International Conference for Internet Technology and Secured Transactions (ICITST-2014)*, December 2014.

[7] D. H. Sack, "Lecture 2: Semantic Web Technologies, in Open HPI Course: Knowledge Engineering With Semantic Web Technologies," Website, April 2014, http://open.hpi.de/courses/semanticweb2014/.

[8] ——, "Lecture 5: Knowledge representations part 2, in Open HPI Course: Knowledge Engineering With Semantic Web Technologies," Website, April 2014, http://open.hpi.de/courses/semanticweb2014/.

[9] "Advantages of RDF over relational databases," Website, Accessed January 2015, http://answers.semanticweb.com/questions/19183/advantages-of-rdf-over-relational-databases.

[10] A. K. Y. Wong, A. C. Chen, N. Paramesh, and P. Rav, "Ontology mapping for network management systems," in *Network Operations and Management Symposium, 2004. NOMS 2004. IEEE/IFIP*, vol. 1. IEEE, 2004, pp. 885–886.

[11] A. Kobayashi, D. Matsubara, S. Kimura, M. Saitou, Y. Hirokawa, H. Sakamoto, K. Ishibashi, and K. Yamamoto, "A proposal of large-scale traffic monitoring system using flow concentrators," in *Management of Convergence Networks and Services*. Springer, 2006, pp. 53–62.

[12] M. Siekkinen, E. Biersack, G. Urvoy-Keller, V. Goebel, and T. Plagemann, "Intrabase: integrated traffic analysis based on a database management system," in *End-to-End Monitoring Techniques and Services, 2005. Workshop on*, May 2005, pp. 32–46.

[13] R. Hofstede, A. Sperotto, T. Fioreze, and A. Pras, "The network data handling war: Mysql vs. nfdump," in *Networked Services and Applications-Engineering, Control and Management*. Springer, 2010, pp. 167–176.

[14] M. Atighetchi, J. Griffith, I. Emmons, D. Mankins, and R. Guidorizzi, "Federated access to cyber observables for detection of targeted attacks," in *MILCOM*, 2014.

[15] S. U. Stanford Center for Biomedical Informatics Research, "Protégé: A free, open-source ontology editor and framework for building intelligent systems," Website accessed on January 2015, http://protege.stanford.edu.

[16] VMware, "vSphere Hypervisor," Website accessed on 1/2015, http://www.vmware.com/products/vsphere-hypervisor. [Online]. Available: http://www.vmware.com/products/vsphere-hypervisor

[17] C. Systems, "Cisco Nexus 1000V Switch for VMware vSphere," Website accessed on 1/2015, http://www.cisco.com/c/en/us/products/switches/nexus-1000v-switch-vmware-vsphere/index.html. [Online]. Available: http://www.cisco.com/c/en/us/products/switches/nexus-1000v-switch-vmware-vsphere/index.html

[18] VideoLAN, "Vlc media player," Website accessed on January 2015, http://www.videolan.org/vlc/index.html. [Online]. Available: http://www.videolan.org/vlc/index.html

[19] "Softflowd: A flow-based network traffic analyser," Website, Accessed January 2015, http://www.mindrot.org/projects/softflowd/. [Online]. Available: http://www.mindrot.org/projects/softflowd/

[20] M. W. Lucas, "Monitoring network traffic with netflow," Website, http://www.onlamp.com/pub/a/bsd/2005/08/18/Big_Scary_Daemons.html.

[21] "Nfdump," Accessed January 2015, http://nfdump.sourceforge.net/. [Online]. Available: http://nfdump.sourceforge.net/

[22] "PostgreSQL," Accessed January 2015, http://www.postgresql.org. [Online]. Available: http://www.postgresql.org

[23] "The R Project for Statistical Computing," Accessed January 2015. [Online]. Available: http://www.r-project.org

[24] "rdflib: A python library for working with RDF," Website, Accessed January 2015, https://code.google.com/p/rdflib/. [Online]. Available: https://code.google.com/p/rdflib/

[25] "Shiny by RStudio," Accessed January 2015, http://shiny.rstudio.com. [Online]. Available: http://shiny.rstudio.com