Challenges in Network Application Identification

Alok Tongaonkar
Narus, Inc.
atongaonkar@narus.com

Ram Keralapura
Narus, Inc.
rkeralapura@narus.com

Antonio Nucci
Narus, Inc.
anucci@narus.com

ABSTRACT

The evolution of the Internet in the last few years has been characterized by dramatic changes to the way users behave, interact and utilize the network. This has posed new challenges to network operators. To deal with the increasing number of threats to enterprise networks, operators need greater visibility and understanding of the applications running in their networks. In years gone by, the biggest challenge in network application identification used to be of providing real-time classification at increasing wire speeds. But now the operators are facing another challenge - the ability to keep pace with the tremendous rate of development of new applications. This problem can be attributed largely to the explosive growth in the number of web and mobile applications. This combined with application hiding techniques like encryption, port abuse, and tunneling have rendered the traditional approaches for application identification ineffective. In this paper, we discuss the challenges facing the network operators and the limitations of current state of the art approaches in both the commercial and the research world in solving these problems.

1. INTRODUCTION

A critical aspect of network management from an operator's perspective is the ability to understand or classify all traffic that traverses the network. This ability is important for traffic engineering and billing, network planning and provisioning as well as network security. Rather than basic information about the ongoing sessions, all of the aforementioned functionalities require *accurate* knowledge of what is traversing the network in *real-time* in order to be effective.

2. STATE OF ART

Traditionally, network operators used *port numbers* to identify and classify network traffic. For example, this technique labels all traffic on TCP port 80 to be HTTP traffic, all traffic on TCP port 25 to be SMTP, and so on. This approach was successful because many traditional applications use port numbers assigned by or registered with the Internet Assigned Numbers Authority (IANA). Moreover, this approach is extremely simple to implement and introduces very little overhead on the classifier. Although very effective a decade ago, today, this approach will result in high false positive and false negative rates. There are three main reasons for this: (1) **Port abuse:** Applications (like P2P networks) have started using non-standard ports for communication [11, 16, 19]. For example, BitTorrent [1] can run on TCP port 80 if all the other ports are blocked. (2) **Random port usage:** Applications can (ab)use random ports for communication. For example, BitTorrent can communicate on any TCP or UDP network port that is configured by the user. (3) **Tunneling:** Applications can tunnel traffic inside other applications to prevent detection and/or for ease of implementation. For example, BitTorrent can send all its data inside a HTTP session. These strategies at the application-level have essentially made port number based traffic classification inaccurate and hence ineffective [16].

To overcome the above issues with port-based approach, the research community has focused on a family of techniques called "flow-features-based analysis". Common goal of these techniques is to identify which application-class a traffic flow belongs to when using traffic flow information only. These techniques achieve the flow-application class mapping by extracting and analyzing hidden properties of the flow either in terms of "social interaction" of hosts engaged in such a flow [10, 12, 13] or the spatial-temporal behavior of several flow features such as flow duration, number and size of packets per flow, and inter-packet arrival time [9, 14, 15, 17, 18, 21, 23]. Many of these techniques suffer from low detection rate and low accuracy. A large variety of more and more sophisticated data mining algorithms have been proposed on top of such framework, such as supervised and un-supervised machine learning, clustering and graph-theoretical approaches, to increase the detection rate while decreasing the false positive rate. Nevertheless, these techniques lack one fundamental attribute which makes them impractical from an operational perspective, i.e., precise identification of the application responsible for the observed flow. Many of these techniques provide coarse-grained "application class" detection like *p2p* and *game application*. This is clearly not sufficient for many applications like security that need to know more fine-grained information about the applications in the network traffic.

Commercial world has responded to these challenges by relying on inspection of application payload for identification. This technique, called as deep packet inspection (DPI), is agnostic to application port usage and usually very accurate (very low false positive and false negative rates). It matches predefined application signatures against incoming traffic. The signatures for a given application are developed by reverse engineering the application/protocol. However, this approach faces the problem of *scalability*. For example, reverse engineering the several hundred new p2p and gaming protocols that have been introduced over the last few years requires a huge manual effort. As a consequence, keeping a comprehensive and up-to-date list of application signatures is infeasible.

Many techniques were developed in the research world that tackle the problem of automatically reverse engineering the network protocols. Early works in this area focused on inferring the message formats for network applications [6, 8, 22]. These approaches can be broadly classified into: (*i*) Host based techniques - these techniques run and monitor the application on a host and derive application signatures by performing dynamic data analysis [6, 22]. (*ii*) Network traffic based techniques - these techniques rely on generating signatures based on observing the network traffic [8]. Later works extend these ideas for protocol state machine inference both on host based [7] and network traffic sides [5]. Although these techniques are very useful in extracting signatures for known network applications, there are many challenges in using such techniques in a real-world setting. One of the biggest challenges that these techniques face is in being able to identify the so-called "0-day applications", i.e, applications that have not been seen before. These techniques work with the application binary or application network traffic. For network traffic containing flows from applications for which there is no prior information, there is no way of extracting signatures.

3. CHALLENGES

The limitations of current approaches for automated reverse engineering of protocols are further exposed due to the changes in the way that networks are being used. (*i*) Recent years have seen a tremendous increase in the number of applications tunneled inside other applications. For example, Zynga [4] offers several games that are played on the Facebook platform. All of this network traffic is carried over HTTP. New exploits are being discovered which target such applications. This means that the network operators need to know not just that there is HTTP traffic in the network but also that there is Zynga poker within Facebook traffic being carried over HTTP. (*ii*) There has been a dramatic shift in the network traffic mix in recent years. Mobile traffic now accounts for a major chunk of network traffic within enterprises [2]. This can be attributed to the tremendous increase in the number of mobile devices being used in enterprises along with an exponential growth in the number of mobile

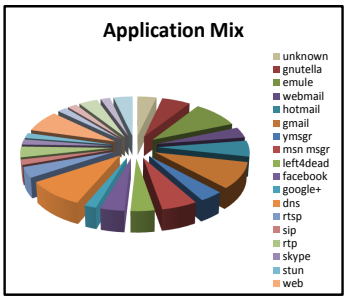

Figure 1: Ideal Scenario

applications (also called mobile apps). A few years ago, most of the mobile devices in an enterprise, such as Blackberries, were controlled by the organization. These devices had integrated security and very few applications. However, now iOS and Android based devices account for a majority of the devices in the enterprise. Each of these platforms have more than 500K apps in their official app markets along with thousands of apps available in third-party app markets. The ease of development of mobile apps means that this growth in the number of mobile apps is likely to continue in the foreseeable future. Another interesting change in mobile device usage within enterprises is that more and more personal mobile devices are being used within enterprise networks. Enterprise network operators have no control over the apps that a user downloads on her device and runs within the enterprise network. This combined with the increasing threat of mobile malware [3], make these devices and the applications running on them soft targets for attackers. This fact is reflected in the increasing number of attacks targeting mobile apps [20]. Hence, network operators need greater visibility into the mobile application traffic in the network.

The combined effect of the above trends is that network operators can no longer have a clear view into the network by using signatures developed by reverse engineering the few top protocols or applications that account for the majority of the traffic. Hence, there is an increasingly growing disparity between the expectation that network operators have from network application identification tools and what they really produce. Figures 1 and 2 illustrate this gap between what the operators *need* and what they *get*. What the operators need is a fine grained application classification (mobile apps and tunneled applications) with very low number of *unknown* flows as shown in Figure 1. What they typically end up with is a coarse grained application classification with large number of unknown flows as shown in Figure 2.

4. CONCLUSIONS

In this paper we discussed the current state of art approaches for network application identification and the reasons why they fail to provide clear visibility into network traffic. Clearly, there is a great need for development of semi-supervised or unsupervised learning techniques that can help in detecting

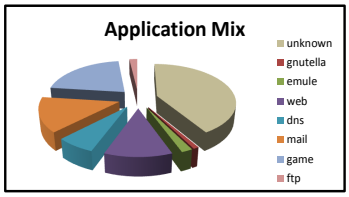

Figure 2: Real Scenario

unknown or new applications with little or no manual intervention. Such techniques should be able to handle mobile apps and tunneled applications to be effective in providing complete visibility into the network traffic.

5. ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their valuable inputs and recommendations. We would also like to thank Shuaifu Dai for his input regarding current trends in mobile security.

6. REFERENCES

[1] Bittorrent. http://www.bittorrent.com/.
[2] Flurry report. http://techcrunch.com/2012/01/09/flurry-mobile-app-usage-up-to-94-minutes-per-day/.
[3] A survey of mobile malware in the wild. http://www.cs.berkeley.edu/ afelt/mobilemalware.pdf.
[4] Zynga. http://company.zynga.com/.
[5] J. Antunes, N. Neves, and P. Verissimo. Reverse Engineering of Protocols from Network Traces. In *Working Conference on Reverse Engineering (WCRE)*, 2011.
[6] J. Caballero, Z. Liang H. Yin, and D. Song. Polyglot: Automatic Extraction of Protocol Message Format using Dynamic Binary Analysis. In *14th ACM Conference on Computer and Communications Security*, 2007.
[7] P. Comparetti, G. Wondracek, C. Kruegel, and E. Kirda. Prospex: Protocol Specification Extraction. In *IEEE Symposium on Security and Privacy*, 2009.
[8] W. Cui, J. Kannan, and H. J. Wang. Discoverer: Automatic Protocol Reverse Engineering from Network Traces. In *16th USENIX Security Symposium*, 2007.
[9] J. Erman, M. Arlitt, and C. Williamson. Offline/Realtime Traffic Classification Using Semi-Supervised Learning. In *IFIP Performance*, October 2007.
[10] M. Illifotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese. Network monitoring using traffic dispersion graphs. In *ACM Internet Measurement Conference (IMC)*, October 2007.
[11] T. Karagiannis, A. Broido, M. Faloutsos, and K. Claffy. Transport Layer Identification of P2P Traffic. In *Proceedings of ACM Sigcomm Internet Measurement Conference*, 2004.
[12] T. Karagiannis, A. Broido, M. Faloutsos, and K. Claffy. Transport Layer Identification of P2P Traffic. In *ACM Internet Measurement Conference (IMC)*, October 2004.
[13] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. Blinc: Multilevel traffic classification in the dark. In *ACM Sigcomm*, 2005.
[14] Z. Li, R. Yuan, and X. Guan. Accurate Classification of the Internet Traffic Based on the SVM Method. In *ICC*, June 2007.
[15] A. McGregor, M. Hall, P. Lorier, and J. Brunskill. Flow clustering using machine learning techniques. In *ACM Passive and Active Measurements Conference (PAM)*, April 2004.
[16] A. Moore and K. Papagiannaki. Toward the Accurate Identification of Network Applications. In *Passive and Active Measurements*, March 2005.
[17] A. Moore and D. Zuev. Internet traffic classification using bayesian analysis techniques. In *ACM Sigmetrics*, June 2005.
[18] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield. Class-of-service mapping for qos: a statistical signature-based approach to ip traffic classification. In *ACM Internet Measurement Conference (IMC)*, October 2004.
[19] S. Sen, O. Spatscheck, and D. Wang. Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures. In *WWW2004*, May 2004.
[20] T. Vidas, D. Votipka, and N. Christin. All your droid are belong to us: A survey of current android attacks. In *Proceedings of the 5th USENIX conference on Offensive technologies*, WOOT'11. USENIX Association, 2011.
[21] N. Williams, S. Zander, and G. Armitage. A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification. In *ACM Sigcomm CCR*, October 2006.
[22] G. Wondracek, P. Milani Comparetti, C. Kruegel, and E. Kirda. Automatic Network Protocol Analysis. In *15th Symposium on Network and Distributed System Security (NDSS)*, 2008.
[23] S. Zander, T. Nguyen, and G. Armitage. Automated traffic classification and application identification using machine learning. In *IEEE LCN*, November 2005.