Misclassification of Game Traffic based on Port Numbers: A Case Study using Enemy Territory

Sebastian Zander
Centre for Advanced Internet Architectures (CAIA). Technical Report 060410D
Swinburne University of Technology
Melbourne, Australia
szander@swin.edu.au

Abstract—**The identification of game traffic in the Internet is very useful for a number of tasks. For trend analysis it is important to find out how much game traffic is in the Internet and how much traffic certain games contribute. To provide better than best effort QoS for game traffic in the network it is necessary to identify game traffic before it can be prioritised. Traditionally, network applications have been classified based on port numbers. It has been argued that purely relying on port numbers does result in significant number of unidentified flows for applications such as peer-to-peer file sharing and game traffic. While this has already been shown for peer-to-peer traffic no such studies exist for game traffic. In this paper we focus on one particular game and estimate how much of the traffic cannot be identified when solely relying on port number based identification. We find that the number of game flows using non-default port numbers is significant. Our evaluation is based on real traffic captured at clients and public game servers.**

Keywords- **Network Application Identification, Game Traffic**

I. INTRODUCTION

Traditionally, network applications have been classified based on port numbers but this approach is becoming more and more unreliable. The Internet Assigned Numbers Authority (IANA) [1] assigns the well-known ports from 0-1023 and registers port numbers in the range from 1024-49151. Many applications do not have IANA assigned or registered ports however and only utilise 'well known' default ports. Often these ports overlap with IANA ports and an unambiguous identification is no longer possible.

A port database [2] that lists not only the IANA ports but also ports reported by users for different applications shows that many applications have overlapping ports in the IANA registered port range. As more and more applications emerge, this overlap will increase since the port number range is not likely to increase (this would require changing the UDP and TCP protocols).

Even applications with well-known or registered ports can end up using different port numbers when users attempt to hide their existence or bypass port-based filters, or when multiple servers are sharing a single IP address (host). Furthermore some applications (e.g. passive FTP or video/voice communication) use dynamic ports unknowable in advance.

This problem has been studied extensively in relation to peer-to-peer file sharing (p2p) applications. Incentives for reliable detection are very high for p2p. Many network operators attempt to block p2p traffic because of the large volume generated and/or the legal issues arising from copyright violations. Previous studies have shown that with pure port number based identification not all peer-to-peer traffic can be detected (for example see [3]). Recently network gaming has also become more prominent.

The identification of game traffic in the Internet is very useful for a number of tasks. For trend analysis purposes it is important to find out how much game traffic is in the Internet and how much traffic certain games contribute. To provide better than best effort QoS for interactive games, game traffic needs to be identified before it can be prioritised (see [4]). Previous work has identified a number of shortfalls for pure port number based identification that could pose problems for game traffic [5].

Online games usually have only a commonly known default port (but no IANA registered port), which means other applications are potentially using the same port numbers. Often multiple online game servers run on a single physical host (IP address), which means every server must run on a different port and therefore many servers run on non-default ports. On the client-side many players are behind Network Address Translation Gateways (NATs) that may change client port numbers.

However, no empirical studies have been published yet on how efficient or inefficient port number based identification would be for game traffic. In this study we estimate the amount of game traffic that cannot be identified purely based on port numbers for a First Person Shooter (FPS) game called Enemy Territory (ET) [6] by studying ports used by clients and servers. We find that even if client and server port are utilised a significant number of flows (23%) cannot be detected.

The paper is structured as follows. Section II describes our approach including how we collected our datasets. Section III presents the results and section IV concludes and outlines future work.

II. APPROACH

We focus on ET as it is currently quite popular. The default port number for ET is 27960, which is the same as for Quake 3 (ET is actually a Quake 3 modification).

All FPS games use a very similar mechanism for letting players connect to servers. First a player's game client retrieves a list of all available game servers from a so-called master server. Then the game client probes all servers on the list to retrieve information such as the number of players on the server, current map, latency to the server etc. (see [7] for more details). Finally the player chooses a server and connects to it.

Since we are not able measure all ET traffic present in the Internet our strategy is as follows. We separately measure the port number distribution of a representative set of clients and the port number distribution of all servers. All network data is grouped into flows by using source/destination IP addresses and port numbers. Based on this we estimate the false negative rate for identifying ET traffic (ET flows not using the default port). It is impossible to estimate the false positive rate (other flows using the ET default port and therefore falsely identified as ET) because this would require knowledge about the port number distribution of all other traffic.

In this paper we analyse two datasets that have been measured as part of CAIA's GENIUS research program [8]. The datasets are described in the following two paragraphs.

We have setup two public ET servers and have measured all incoming and outgoing traffic for 20 weeks. One server is connected to our university network (CAIA server) and the other is connected to GrangeNets [9] high-speed network (GrangeNet server). The dataset is described in more detail in [7]. In [7] we have shown that at a public ET server a snapshot of the global player population can be observed when taking into consideration the many probe flows caused by clients probing the server. The reason is that the ET master server always sends the complete list of servers to a requesting client and the list is not ordered by server location (distance to the client). Thus we can use the data as a sample of the client port number distribution of the global ET player community. In the following we refer to this dataset as client ports dataset.

To measure the port number distribution of all ET servers we have queried the ET master server every 30 minutes for about 22 days. We recorded all server information, including the port numbers the servers run on. The total number of servers was fairly stable during the measurement period at around 3,000 servers. This dataset and its collection are described in more detail in [10]. In the following we refer to this dataset as server ports dataset.

III. RESULTS

Figure 1 shows the Cumulative Density Functions (CDFs) of the whole port number distributions of both client ports datasets and the server ports dataset. The figure shows that client ports are widely distributed across the whole range. Only for 55%-60% of the flows was the client port equal to the default port.

We believe the client port distribution is mainly caused by Network Address Translation Gateways (NATs). The game client always uses the default port but the port number is likely to be changed if clients are behind NATs (as suggested in [11]). Some NATs however do not change the port number unless they have to because the port is already in use. But if the port is changed it is usually changed to a random number not necessarily very close to the default port.

Distributions observed at both servers are very similar. The server port distribution apparently only has about 15%-20% of the servers running on the default port.

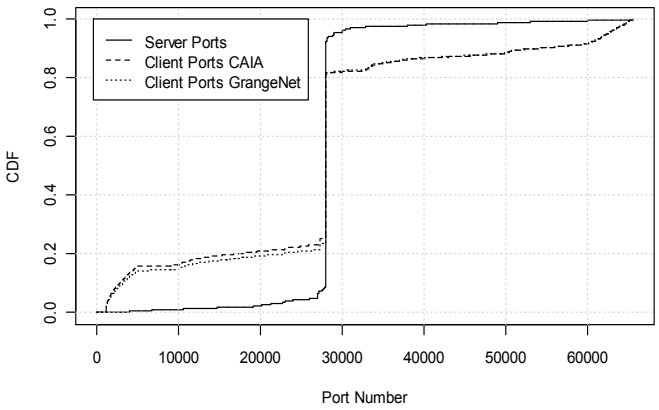

Figure 1: Cummulative distribution of port numbers

This is not the case however, as we zoom in on the default port (see Figure 2). Now we can see that in fact roughly 50% of the servers were running on non-default port numbers. 30% of the non-default ports are larger but very close to the default port.

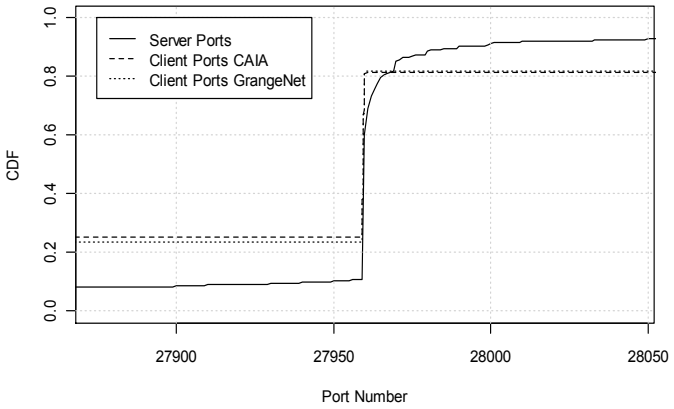

Figure 2: Cummulative distribution of port numbers zoomed in on the default port

The server port distribution is different because it is far less spread out across the port number space. Most non-default ports are higher but very close the default port. This is probably because multiple servers are running on one IP address (this could be a single physical box, a load balancer in front of multiple servers etc.). If started after each other servers automatically obtain the next available port number higher than the default port. If manually configured server administrators are likely to select port numbers that are very close to the default port. A similar effect was observed in [12] for server ports of the FPS game Half-Life.

Figure 1 and Figure 2 show the distributions of the

overall datasets. To investigate how the percentage changes over time we have grouped all flows measured at both servers into hourly time intervals and computed the percentage of flows not having default client ports for each hour. Figure 3 shows the CDFs of the percentages.

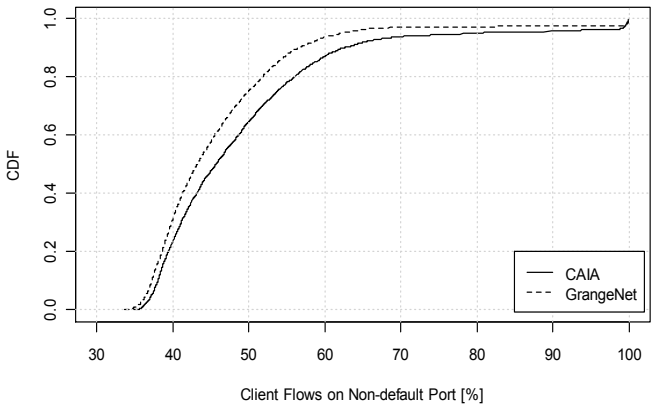

Figure 3: CDFs of hourly percentage of client flows using non-default ports

The figure shows that the percentage is not constant but changes between 35% and 70% for most hours. The distributions are quite similar for both servers.

Figure 4 shows the CDF of the per 30 minutes percentage of servers that are not using the default port. Above we found that if servers are not running on the default port, often their port number is very close to the default. Therefore, we introduced a distance parameter *d* that simply specifies the range (27960-*d*, 27960+*d*). We treat all ports within the range as if they would be the default ET port. This means for *d*=0 we only accept 27960 as default port, whereas for larger *d* we accept a whole range of ports.

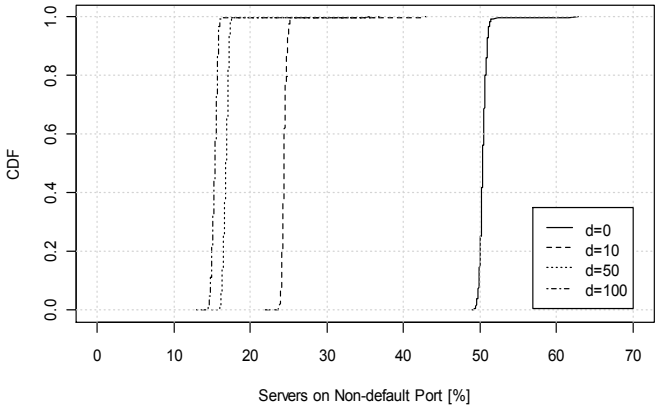

Figure 4: CDFs of half hourly percentage of servers running on non-default port

The CDFs are very steep as we encounter only little variation of the percentage during the measurement period. The distance parameter has a large impact on the percentage for smaller *d* values, but the impact is rapidly decreasing for larger *d*. For identification based on port numbers only, it would be obviously beneficial to set *d* between 10 and 50 to greatly reduce the number of unknown ports (from 50% down to ~17%). However, if any other applications use ports in these ranges the number of false positives would also increase.

If only the server port is used for identification the false negative rate would be expected to be between 15% and 50% depending on *d* (*d*>0 is likely to cause false positives though). If the client port is also used we estimate the false negative rate based on both port distributions. However, because many clients are behind NATs that potentially map ports of other application to the default ET port, this would also lead to an increased false positive rate.

Assuming the percentages in the CDFs reflect the probabilities of clients and servers to use non-default ports, both probabilities are independent and clients choose servers randomly we can simply estimate a joint probability using the product of the two probabilities. On average 45% of flows have non-default client ports. With *d*=0 the false negative rate could be decreased to 23% (50% non-default server ports) whereas *d*=50 would reduce the false negative rate further down to 8% (17% non-default server ports). Again, this decrease in false negatives is very likely to come at the price of an increased number of false positives.

IV. Conclusions and Future Work

In this paper we investigate if game traffic can be identified purely based on port numbers. Our study is built around a currently popular first person shooter game. We have measured representative client port distribution and server port distributions. Based on these distributions the percentage of game flows not identifiable is between 50% and 8%. However, small false negative rates can only be achieved with a very likely increase in false positives. We conclude that a significant fraction of the traffic cannot be detected purely based on port numbers.

Obviously it is interesting to expand this work to different games and/or different game types (e.g. real-time strategy. We believe results for other first person shooter games could be similar since all FPS games are based on the same client, server and master server architecture. However, for other game types that use different mechanisms (e.g. peer-to-peer) results could be totally different.

Since port numbers seems to be inaccurate for detecting game traffic (as we demonstrated for one popular game) the question is how game traffic could be more reliably detected?

Payload-based identification provides very high accuracies. It can be further divided into protocol decoding and signature-based identification. With protocol decoding the classifier actually decodes the application protocol, while signature-based methods search for application specific byte sequences in the payload. However, payload-based classification relies on specific application data, making it difficult to detect a wide range of applications or stay up to date with new applications. Since most game protocols are not openly specified this would require at lot of reverse engineering. In addition, the process of creating rules for signature-based classification must often be done by hand, which can be very time consuming.

Another promising alternative proposed more recently is to use machine learning [13] and classify flows based on payload-independent statistical flow attributes such as packet length or inter-arrival time distributions. In CAIA's DSTC project [14] we currently evaluate the effectiveness of this approach.

ACKNOWLEDGEMENTS

We thank Carl Javier for providing the server port dataset and David Kennedy who helped preparing the client port datasets. We thank GrangeNet and specifically Chris Meyers for letting us run a game server in their network. We also thank Grenville Armitage and Nigel Williams for their feedback that helped improve the paper.

This paper has been made possible in part by a grant from the Cisco University Research Program Fund at Community Foundation Silicon Valley.

REFERENCES

[1] IANA, http://www.iana.org/assignments/port-numbers (as of February 2005).

[2] Ports database, http://www.portsdb.org/ (as of February 2005).

[3] Thomas Karagiannis, Andre Broido, Nevil Brownlee, kc claffy, "Is P2P dying or just hiding?", Proceedings of Globecom 2004, November/December 2004.

[4] L.Stewart, G.Armitage, P.Branch, S.Zander, "An Architecture for Automated Network Control of QoS over Consumer Broadband Links," IEEE TENCON 05 Melbourne, Australia, 21 - 24 November, 2005.

[5] S.Zander, T.T.T.Nguyen, G.Armitage "Automated Traffic Classification and Application Identification using Machine Learning," IEEE 30th Conference on Local Computer Networks (LCN 2005) Sydney, Australia, 15-17 November, 2005.

[6] Enemy Territory: http:// www.enemy-territory.com (as of January 2005)

[7] S. Zander, D. Kennedy, G. Armitage "Dissecting Server-Discovery Traffic Patterns Generated By Multiplayer First Person Shooter Games, ACM NetGames 2005, Hawthorne NY, USA, 10-11 October, 2005.

[8] Game Environments Internet Utilisation Study (GENIUS): http://caia.swin.edu.au/genius/ (as of February 2006)

[9] GrangeNet: http:// www.grangenet.net (as of February 2006)

[10] G.Armitage, C.Javier, S.Zander, "Measuring a Wolfenstein Enemy Territory Master Server's Response to Game Client Queries," CAIA Technical Report 060410A, April 2006

[11] G.J. Armitage, "Inferring the Extent of Network Address Port Translation at Public/Private Internet Boundaries," CAIA Technical Report 020712A, July 2002.

[12] S. Zander, N. Williams, G. Armitage, "Internet Archeology: Estimating Individual Application Trends in Incomplete Historic Traffic Traces", Technical Report, http://caia.swin.edu.au/urp/dstc/dstc-papers.html, Centre for Advanced Internet Architectures, Swinburne University of Technology, Melbourne, Australia, Feb 2006.

[13] Tom M. Mitchell, "Machine Learning", McGraw-Hill Education (ISE Editions), December 1997.

[14] Dynamic Self-learning Traffic Classification based on Flow Characteristics (DSTC): http://caia.swin.edu.au/urp/dstc/ (as of February 2006)