This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2007 proceedings.

Traffic Analysis and Modeling for World of Warcraft

Philipp Svoboda, Wolfgang Karner, Markus Rupp
Institute of Communications and Radio-Frequency Engineering
Vienna University of Technology, Austria
Gusshausstrasse 25/389, A-1040 Vienna, Austria
Email: {psvoboda, wkarner, mrupp}@nt.tuwien.ac.at

***Abstract*— In the last years the popularity of MMOGs (Massively Multiplayer Online Games) has grown fast. This new type of online games uses a different type of protocol and has other limitations to delay and packet loss than the classical well established games like FPSs (First Person Shooters). This paper focuses on the traffic patterns generated by a MMOG named WoW (World of Warcraft). We analyzed IP layer information like packet size and rate from network traces and user level information like session durations. Finally we will present a simple synthetic traffic generator for WoW traffic clients.**

I. Introduction

In the recent past the popularity of online games has grown fast [1]. The traffic share of online gaming traffic has reached a level of around 5% in the networks of various service providers. The relatively high load combined with the fact that this type of traffic shows significant different characteristics compared to common and well known traffic sources like WWW, e-mail or FTP (File Transfer Protocol) makes it interesting to analyze. FPS (First Person Shooters) and common strategy games like Warcraft III are well described in various papers [2–4]. In Asia MMOGs (Massively Multiplayer Online Games) are much more popular than in Europe, therefore there are already some publications based on games like ShenZhou-Online [1].

Within the last years, games like WoW made MMOGs very popular. A MMOG intends to create a virtual environment which is populated simultaneously by thousands of players. The big attraction arises from the fact that only few characters are simulated by the server, called NPCs (Non Playing Character), while the rest are humans.

The session times of MMOGs are typically higher than those of round based action games, like FPSs. FPS games join short matches in the order of 15min. Users stay for several matches in one session. Due to these tournaments the sessions have a periodical behavior. It is therefore common to model only one tournament and repeat the result [2]. MMOG gamers will play one continuous session with varying parameters, this impacts the traffic patterns. MMOGs motivate to go online regularly to interact with the virtual environment. Therefore the inter session times may show a periodic profile.

Seen by the network traffic engineer this type of application pushes a new class of service: the realtime interactive service over TCP (Transport Control Protocol). The realtime restriction is caused by the interactive nature of the application. The gamer is interacting with others via the server node. Therefore, responsiveness of the underlaying transport network is crucial for a great gaming experience. The maximum tolerable delays are much larger than those known from other online games. Common realtime systems use UDP (User Datagram Protocol), because retransmitted packets will have too high delay and will be discarded anyway. In WoW and many other MMOGs TCP serves as transport protocol. TCP is connection oriented and offers a reliable connection. This attribute is well suited for MMOGs, preventing error propagation during long sessions. The RTT (Round Trip Time) can be reduced by using small packets. In WoW we are now dealing with an application generating large numbers of very small TCP packets in one long stream. Services like web browsing generate many small connections.

We monitored a mobile core network within the METAWIN (see [5] for more details) project. There we obtained a remarkable volume share at port $TCP : 3724$, (well known for WoW). In the past, online games in mobile networks were exclusive. We were quite surprised to find WoW (World of Warcraft) in the top ten TCP services when analyzing traces taken in the mobile core network of an operator in Austria. We decided to extract the session times using these tracee. Traces from a fixed wired Internet access were used for the packet level analysis.

The paper is structured in the following way. Section II will give a short overview how the WoW client works and the measurement setup will be shown. In Section III we will analyze the traffic patterns on the IP level. Here parameters like packet-size and inter-arrival time are extracted and the user behavior including session durations and inter session times are analyzed. Section IV focuses on the simulation model. The obtained parameters are mapped to suitable simulation inputs and a simulation is performed. Section V will show an overview of related work on gaming. In the final section we will give a conclusion.

II. Measurement Setup

We used two setups for our measurements, an active and a passive approach, to obtain the different parameters.

[1] `http://www.ewsoft.com.tw/`

1-4244-0353-7/07/$25.00 ©2007 IEEE

The first setting shown in Fig. 1(a) is based on active measurements. This setup is able to monitor user session parameters, IP (Internet Protocol) packet size and inter-arrival time. The IP parameters were recorded using a monitoring PC running the WoW client, which was connected to the Internet via an 1Mbit ADSL (Asymmetric Digital Subscriber Line) link. In addition to this an ethereal[2] session running at the PC recorded the packet traces. The datasets consist of several gaming sessions with a sum of about 20 hours netto active time, recorded in May 2006.

The user behavior was extracted using a shell script recording the gaming time of two groups with five gamers each. The script ran as a background job, dumping start and stop time of the $wow.exe$ file. These results contain the session properties for each player. To verify these values from the small sample we searched for an alternative data source.

The source found was the METAWIN[3] project. METAWIN is a project established in the core network of a mobile operator to analyze and plan the IP traffic. Fig. 1(b) sketches the system.

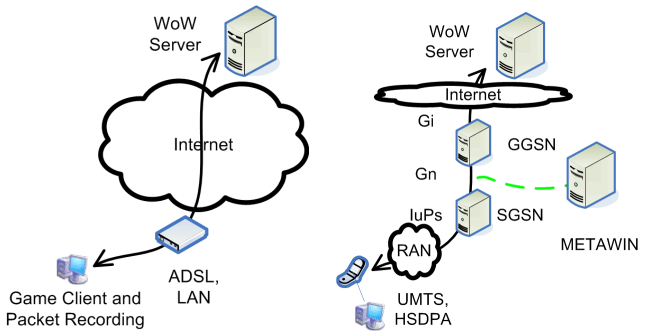
In the project we passively monitor all UMTS (Universal Mobile Telecommunications System) and GPRS (General Packet Radio Service) core network interfaces (Gi, Gn, Gb, IuPS). The results presented in this work are based exclusively on traces captured on Gn, specifically on the links near the GGSNs. The choice of Gn for this type of analysis was based on the following motivations. First, capturing on Gi links does neither allow a per mobile station discrimination nor PDP-context (Packet Data Protocol) aware analysis, since the IMSI (International Mobile Subscriber Identitys) and PDP activations are not present on Gi. Second, capturing on IuPS/Gb would involve many more physical links, without gaining any further information in our case.

Using the information gained at the Gn interface we analyzed the user behavior, e.g. via the PDP-context. The PDP-context attributes include anonymised IMSI, start time, duration, transferred volumes. The connection attributes include PDP-context identifier, start time, duration, volumes, IP addresses and ports. More information about the mobile protocol stack and the detailed location of the interfaces can be found in [6, pp. 41].

The METAWIN dataset contains one week of traffic in September 2006. Analyzing this trace we found approximatly 1000 different users playing WoW. We were surprised by the fact that these users summed up to nearly 1% of the total TCP traffic in the core network. We found out in [7] that for non web service a share of 1% is a relatively large number in mobile networks . In fact other common services like FTP had a smaller volume share.

Considering the two sources we are able to verify the session time of the small sample with the one gained by the large population. Note that the users monitored in the mobile core network most likely used a flat rate contract, therefore a similar behavior is expected. The datasets from the wired Internet access were recorded using an ethereal parser.

[2]`http://www.ethereal.org/`
[3]`http://www.ftw.at/ftw/research/projects/`



(a) Wired networks (b) Mobile networks

Fig. 1. Measurement Setups

Summarizing, we used two setups to record three different datasets. From the wired Internet access we obtained two datasets. The first dataset is a packet trace from one WoW client. This dataset was used to extract parameters at the IP level. The second dataset contains the session times of 10 users using wired Internet accesses. A shell script recorded the session duration for each user. In the second setup we obtained one dataset containing the session times of a large population.

III. Traffic Characterization

MMOGs focus on accurate execution of client inputs. This has an impact on the transport protocol used. WoW uses TCP connections, while common FPSs use UDP. The TCP ACK mechanism will protect the delivery of the data packets by the underlaying physical network.

A. Basic Analysis

Analyzing the TCP stream from WoW we observed that the two flags ACK and PSH are set for most of the packets. In near real-time usage both flags are used, although uncommon in normal TCP transmission. A TCP agent has several performance features [8], one of them is delayed acknowledgment. Instead of sending an ACK packet for each datagram, the receiver will wait for some time and then acknowledge a bunch of packets all together. On low delay links this will reduce the number of sent ACK packets. However, in near real-time applications, which do not tolerate delays, this is a bad choice as it will introduce extra delay in case of a retransmission.

Another throughput optimization is done by the gathering of small chunks of data. The Nagle algorithm in the TCP agent uses a lower threshold to send blocks of data [9]. As TCP is connection oriented, the way programs transmit data has fundamental differences compared to UDP. In UDP the program has to take care of packetizing the data whereas in TCP the program hands over its payload to the agent without preprocessing, e.g. there is no problem for the program to hand over more than MTU (Maximum Transmission Unit) bytes because the TCP agent will split them later. Due to this behavior the TCP agent has no detailed information on

a timestamp to send a datagram. To avoid the transmission of packets with very small payload that leads to a large overhead, a threshold was established. In an application with relative low data-rates this will introduce a high delay. Known applications like telnet suffer from this fact. Not constant bandwidth will add jitter. The PSH flag forces the TCP agent to send the output buffer immediately, bypassing the threshold.

B. Packet Traces

In this paragraph we will analyze packet traces recorded during several sessions. Our datasets contain $\approx$ 20hours of gaming. Client patch level was 1.10.

Figures 2(a) and 2(b) give a snapshot of the down- and uplink bandwidth time series. To plot the bandwidth, we used a bin size of $1sec$ and no running average. The median downlink bandwidth is $6.9kbit/sec$. In uplink we observe less traffic with a median of $2.1kbit/sec$. From the figure we also learn that there are some high peaks ($\geq 64kbit/sec$) in downlink direction. Comparing the traces with a recorded video we can correlate the peaks to scenes with high environment interaction (e.g. many players nearby). We did not use this information in our model.

Fig. 2(c) presents the CDF of the up- and downlink bandwidth. Compared to the snapshot in Figure 2(a) we observe some not neglectible part of high downlink bandwidth. A detailed analysis of the packet trace time series showed that these peaks occur mainly in the beginning of the sessions.

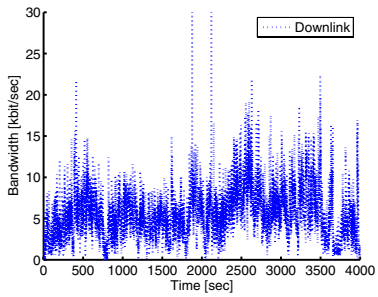
Fig. 2(d) illustrates the empirical CDF of the packet sizes in up- and downlink direction. The figure displays that in up- and downlink 28% and 57% respectivly, of the packets have no payload. This is due to the fact that these packets are only acknowledgments (ACK).

Fig. 2(e) shows the empirical CDF of the inter-arrival time of packets in up- and downlink direction. From this we learn that the client has a target update interval of approximatly 220 ms indicated by a step in the CDF. The maximum values range up to 1.5sec. Up- and downlink curves are similar.

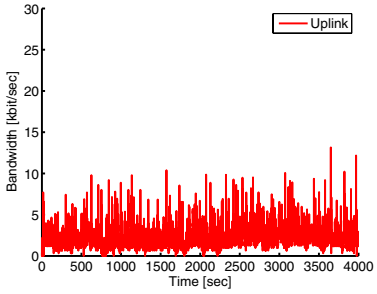
C. User Behavior

With the METAWIN system we recorded the session times for a large population ($\approx$ 1000 gamers). In a next step we focused on the question if using a mobile network had a negative impact on the active time (ontime) or not. Therefore, we also extracted session times from two groups of users, each consisting of five gamers, connected via fixed line access. Ten persons were monitored by the shell script for one month. We split the users into two groups according to their Internet access speed. Group one used a standard ADSL link with $1M/256kbit/sec$ and group two had a link speed of $1.5M/512kbit/sec$.

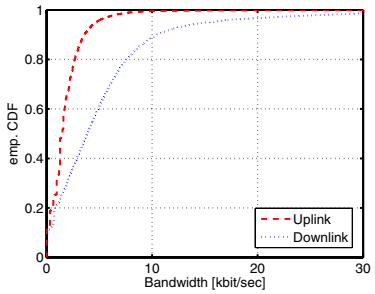
Fig. 3 visualizes the empirical CDF for the session times obtained by the METAWIN system. The ontime exceeds the value of 2.75 hours in more than 20% of the samples. It is interesting to see such high values of active service times in a mobile core network. This indicates that the mobile Internet access is used to replace wired Internet access links. The tail



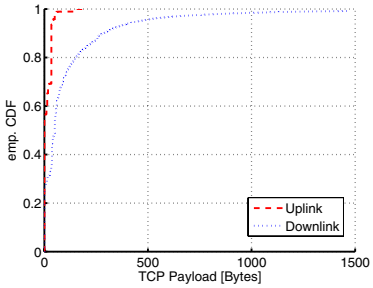
(a) Snapshot of a downlink time series (bin=1sec)



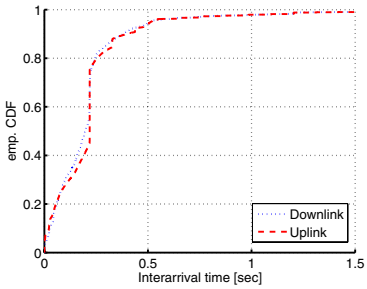
(b) Snapshot of a uplink time series (bin=1sec)



(c) Bandwidth (CDF)



(d) Packet sizes (CDF)



(e) Packet inter-arrival time (CDF)

Fig. 2. Traffic snapshot and summary of the packet-level analysis (CDFs)

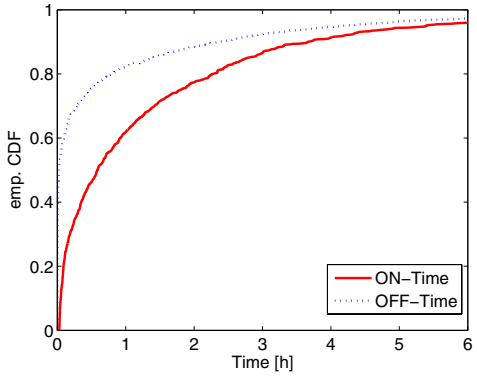


Fig. 3. Session times wired

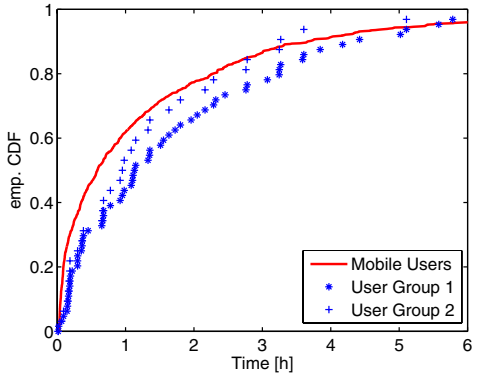


Fig. 4. Session times: mobile vs. wired Internet access

of the offtime curve indicates that the players are online on a regular time frame.

The stars and crosses in Fig. 4 show the values obtained by the second dataset extracted from wired Internet accesses. We obtained that the results of all three datasets were similar. Therefore we fitted our model parameters according to the METAWIN dataset.

IV. Simulation Model

Based on our findings in Section III we developed a ns-2 script to model WoW like client sever connections. The following paragraph discusses the introduction of two new parameters that adapt the measured values to inputs for the simulation. In contrast to UDP, using a TCP connection between server and client introduces interaction at the protocol level. Therefore, it was necessary to transform the input parameters obtained in the packet traces into suitable parameters to drive a simulation. In other words we could not make direct use of the packet sizes measured, but had to find a data unit from the view of the application. We then fitted these new parameters and implemented them into the ns-2 simulation environment.

A. Filtered Packet Trace Analysis

In the previous paragraph we have shown that there is a high fraction of ACK packets in the trace. The first straight forward approach was to filter all packets transmitting no payload.

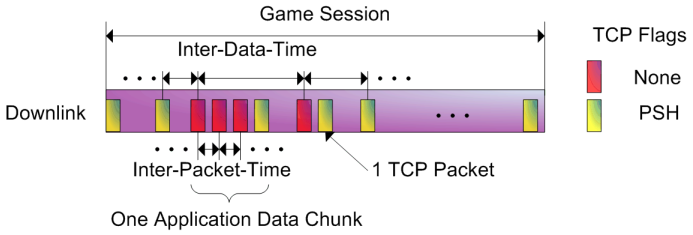


Fig. 5. Definition of application-data-size and inter-data-time

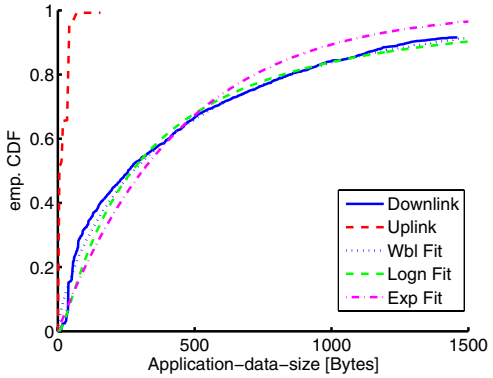


Fig. 6. Empirical CDF of the application-data-size

(Note: removing all ACK packets would also remove payload packets, as the client can mark a normal payload packet to confirm previous traffic.)

Analyzing the resulting packet trace we realized that not all of the packets carry a PSH flag. Extracting these packet sizes we obtained mainly full MTU size. We assume that in this case the application sends datagram units larger than the MTU. The TCP service then splits the units down to MTU in order to transmit them. The last packet of this block is assigned with a PSH flag. Based on this idea we created two artificial values. Fig. 5 illustrates a detailed picture of the parameter mapping. The inter-data-time replaces the interarrival time for packets and the application-data-size substitutes the packet size.

The following figures visualize the empirical CDFs for the two parameters. Fig. 6 displays the empirical CDF of the application-data-size in up- and downlink direction. The uplink curve shows several high discrete steps. We conjecture that these values represent simple commands that are often used, like walk, attack or open. More complex commands, represented by a larger data size in the CDF, have a low probability. As we will find out later the update rate is around $220msec$. Using scripts, which execute multiple commands, may change the distribution.

The downlink curve ranges up to 3000 bytes. In the downlink the application-data size is larger than in the uplink. We applied a continuous distribution for the smooth downlink curve. A Weibull distribution fitted best. A comparison between Fig. 2(d) and 6 points out that only the downlink stream is affected by the chosen add up rule.

Fig. 7 shows the empirical CDF of the inter-data-time. The filter process reduces the step size at approximately $220msec$.

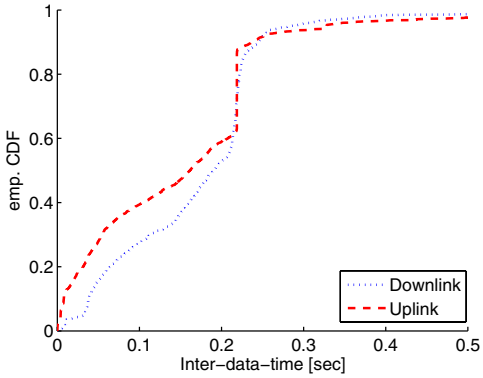


Fig. 7. Empirical CDF of the inter-data-time

The step occurs in up- and downlink. We assume that this is some threshold coded to the software. Both curves have a relative linear part followed by a step at around $220msec$, indicating a uniform distribution. The tail of the curves shows that the threshold is sometimes exceeded by network delays.

B. Simulation Model

In ns-2 standard TCP agents only support unidirectional communication. Although client and server were modeled separately from each other, a full featured ACK process needs a bidirectional TCP implementation on the ns-2 level. A bidirectional agent permits a realistic communication between two nodes, e.g. a payload packet that carries an ACK information. We used the $FullTCP$ agent implemented in ns-2 to model the two way connection, the agent generates ACK packets according to the TCP algorithms. Therefore we directly implemented the parameters extracted in the last paragraph to drive our simulation. The following paragraphs discuss the used parameter fits. The parameters were fitted using the matlab curve fitting function tool.

1) Server → Client: We fitted a Weibull distribution to the dataset as shown in Eq.(1). The resulting parameters for the distribution were $\lambda = 426$ and $k = 0.8196$. The real distribution has an upper limit $l = 3010$ bytes, we implemented a cutoff to model this limit.

$$f(x;k,\lambda) = \frac{1}{n} \cdot \frac{k}{\lambda} \cdot \left(\frac{x}{\lambda}\right)^{k-1} \cdot e^{-\frac{x}{\lambda}^k} \qquad (1)$$

$$n = f_{wbl}(l;k,\lambda)$$

The inter-data time was harder to synthesize. We decided to use a joint distribution of three random variables. All three processes are modeled uniformly distributed. The resulting equation of the PDF is given in Eq.(2). All parameters were fitted by regression to the datasets ($a = 218,3$, $b = 251.2$, $c = 1500msec$). Hence the curves for inter-data time in up- and downlink direction are similar, we applied Eq.(2) to both.

$$f(x) = \begin{cases} 0.620 \cdot \frac{1}{a-0}, & x = [0 \dots a) \\ 0.257 \cdot \frac{1}{b-a}, & x = [a \dots b) \\ 0.123 \cdot \frac{1}{c-b}, & x = [b \dots c] \end{cases} \qquad (2)$$

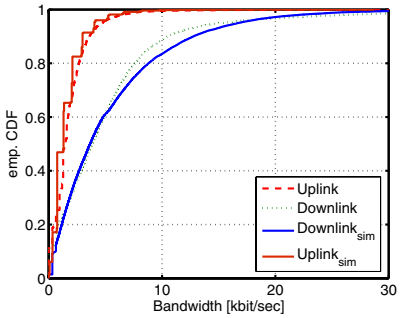


Fig. 8. Simulated Bandwidth (CDF)

2) Client → Server: The application-data size in the uplink had discrete steps (Fig. 6). We implemented a source generating three different sizes of packets (Eq.(3) with $a = 6$, $b = 19$, $c = 43bytes$). The parameters were estimated using the average over the different datasets. The original probability of those steps adds up to 98% in the CDF.

$$f(x;a,b,c) = 0.52 \cdot \delta(a) + 0.14 \cdot \delta(b) + 0.34 \cdot \delta(c) \qquad (3)$$

To model the session times we compared several fittings using Weibull, lognormal and neg-exponential distributions. The best fit for the on times was obtained using a Weibull distribution with the parameters $\lambda = 4321$ and $k = 0.7813$ in Eq.(1). The off time fitted best a lognormal distribution Eq.(4). The resulting parameters were $\mu = 5.512$ and $\sigma = 2.434$.

$$f(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} \cdot e^{-\frac{(\ln x-\mu)^2}{2\sigma^2}} \qquad (4)$$

3) Simulation Output: These distributions were implemented in an ns-2 script[4]. The simulation setup consists of a server client pair using two way TCP agents. The bandwidth recorded in 10 hours of simulation time is plotted in Fig. 8. For a direct comparison we presented the CDF and added the original values given in Fig. 2(c). We notice that the simulation produces a similar distributed bandwidth. However a closer look to the time series reveals the weakness of the simple modeling, as there are no activity bursts like in Fig. 2(a).

V. Related Work

In the recent past online gaming has become a hot research topic. In the very beginning of online gaming mainly FPS and strategy games were played, therefore the first papers covered these games. Borella and Faerber started to develop the first model for gaming traffic [4, 10]. The research group of G. Armitage has published models for more recent games also including the XBox platform [2, 3, 11]. All these models have in common that they are based on UDP traffic. Therefore they support no interaction at the protocol level.

A next step in research was the analysis of user behavior in various networks. In several papers Chang et al. reported on this topic discussing the interaction of delay, user satisfaction and the online session times [12]. As the service *online gaming* itself was a new topic researchers also started to analyze basic

[4]The simulation code is available at `http://www.nt.tuwien.ac.at/about-us/staff/philipp-svoboda/`.

properties as the impact of losses to the service quality [11, 13, 14].

In the last year several research groups started to focus on MMOGs. Especially in Asia this type of games is very popular. Huang et al. modeled the system performance of MMOG regarding the server load [15]. The paper analyzes the load a server has to handle given a number of attached users. Chen et al. presented a paper analyzing the MMOG ShenZhou-Online [16]. Although they only analyzed the packet level, it is interesting to note that the curves for up- and downlink packet sizes are similar to our traffic patterns in WoW.

As far as we know up to now this is the first paper analyzing WoW traffic patterns and showing an explicit ns-2 simulation model.

VI. Summary and Conclusions

In this paper we have analyzed traffic patterns generated by a popular MMOG called World of Warcraft and extracted parameters for a model implemented in ns-2. In contrast to common online games, it relies on TCP to transmit data, which introduces an interaction between server and client at the protocol level. Therefore this traffic differs from common gaming traffic that is based on UDP.

We extracted three datasets for the analysis. The first was a packet trace recorded on wired Internet accesses. A second dataset obtained the session times from a larger population using the METAWIN system, recording one week of traffic in a mobile core network in September 2006. The last dataset recorded the session times using a shell script monitoring two groups of five wired Internet users each.

The analysis at packet level showed that the packet size in the downlink direction was quite smooth and could be modeled with a Weibull distribution, the uplink packet sizes had large discrete steps. We concluded that this was due to some common player actions that were reported to the server. Delta impulses were applied at the discrete packet values to simulate this traffic. Further investigation revealed that not all the packets carried a PSH flag. Therefore we decided in Section III to merge the packets into application datagrams. One application datagram was defined as the payload transfered between two consecutive PSH flags. This mapping introduced parameters which could directly feed the ns-2 simulation.

The session duration was extracted from the second and third dataset. These were based on two contrary sources, a wired and a mobile Internet access. Comparing the two traces revealed a strong similarity in the user behavior. The population was much larger in the mobile set, therefore we used it to fit the model-parameters.

Finally we presented the output of a simulation using the fitted parameters defined in Section IV. The resulting CDFs for the bandwidths are similar to the measured ones. However, the obtained burstiness of the original time series cannot be modeled with this simple approach.

Acknowledgements

We thank mobilkom austria AG for technical and financial support of this work. The views expressed in this paper are those of the authors and do not necessarily reflect the views within mobilkom austria AG.

References

[1] An analysis of MMOG subscription growth. `http://www.mmogchart.com/`.

[2] T. Lang, P. Branch, and G. Armitage. "A synthetic traffic model for Quake3". *ACE '04: Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 233–238, 2004.

[3] S. Zander and G. Armitage. "A traffic model for the Xbox game Halo 2". In *NOSSDAV '05: Proceedings of the international workshop on Network and operating systems support for digital audio and video*, pages 13–18, New York, NY, USA, 2005. ACM Press.

[4] J. Färber. "Network game traffic modelling". In *NETGAMES '02: Proceedings of the 1st workshop on Network and system support for games*, pages 53–57, New York, NY, USA, 2002. ACM Press.

[5] F. Ricciato et al. "Passive Tomography of a 3G Network: Challenges and Opportunities". In *INFOCOM*, 2006.

[6] G. Heine. *GPRS - signaling & protocol analysis*. Inacon, 2002.

[7] P. Svoboda and F. Ricciato. "Composition of GPRS and UMTS traffic: snapshots from a live network". *IPS MoMe 2006, Salzburg*, pages 42–44, Feb 2006.

[8] Information Sciences Institute University of Southern California. *RFC 793: Transmission Control Protocol*. IETF.

[9] J. Nagle. *RFC 896: Congestion control in IP/TCP internetworks*. IETF.

[10] J. Färber. "Traffic Modelling for Fast Action Network Games". *Multimedia Tools Appl.*, 23(1):31–46, 2004.

[11] G. Armitage and L. Stewart. "Some thoughts on emulating jitter for user experience trials". In *NetGames '04: Proceedings of 3rd ACM SIGCOMM workshop on Network and system support for games*, pages 157–160, New York, NY, USA, 2004. ACM Press.

[12] F. Chang and W. Feng. "Modeling player session times of on-line games". In *NETGAMES '03: Proceedings of the 2nd workshop on Network and system support for games*, pages 23–26, New York, NY, USA, 2003. ACM Press.

[13] T. Beigbeder et al. "The Effects of Loss and Latency on User Performance in Unreal Tournament 2003". In *NetGames '04: Proceedings of 3rd ACM SIGCOMM workshop on Network and system support for games*, pages 144–151, New York, NY, USA, 2004. ACM Press.

[14] P. Svoboda and M. Rupp. "Online Gaming Models for Wireless Networks". In *Internet and Multimedia Systems and Applications*, pages 417–422, 2005.

[15] G. Huang, M. Ye, and L. Cheng. "Modeling System Performance in MMORPG". In *Globecom 2004 Workshop*, pages 512–518. IEEE, 2004.

[16] K. Chen, P. Huang, C. Huang, and C. Lei. "Game traffic analysis: an MMORPG perspective". In *NOSSDAV '05: Proceedings of the international workshop on Network and operating systems support for digital audio and video*, pages 19–24, New York, NY, USA, 2005. ACM Press.