Multimed Tools Appl (2009) 45:133–161
DOI 10.1007/s11042-009-0298-4

On the impacts of human interactions in MMORPG traffic

Géza Szabó · András Veres · Sándor Molnár

Published online: 15 May 2009

Abstract Game traffic depends on two main factors, the game protocol and the gamers' behavior. Based on a few popular real-time multiplayer games this paper investigates the latter factor showing how a set of typical game phases—e.g., player movement, changes in the environment—impacts traffic on different observation levels. The nature of human behavior has such a high impact on traffic characteristics that it influences the traffic both at a macroscopic—e.g., traffic rate—and at a microscopic—payload content—level. First, by understanding the nature of this impact a *user behavior detection algorithm* is introduced to grab specific events and states from passive traffic measurements. The algorithms focus on the characteristics of the traffic rate, showing what information can be gathered by observing only packet header information. Second, as an application of our method *some results*, including a detailed analysis of measurements taken from an operational broadband network, are presented. Third, *a novel model and an algorithm* are introduced to extend the Deep Packet Inspection traffic classification method with the analysis of non-fix byte signatures, which are not considered in current methods. The model captures the variation of the dynamic byte segments and provides parameters for the algorithm. The introduced algorithm exploits the spatial and temporal correlation

G. Szabó (✉) · A. Veres
TrafficLab, Ericsson Research, Ericsson Hungary Ltd.,
Laborc u. 1., 1037, Budapest, Hungary
e-mail: geza.szabo@ericsson.com

A. Veres
e-mail: andras.veres@ericsson.com

S. Molnár
Highs Speed Networks Laboratory,
Department of Telecommunications and Media Informatics,
Budapest University of Technology and Economics,
Magyar tudsok krt. 2, 1117, Budapest, Hungary
e-mail: molnar@tmit.bme.hu

by examining and extracting the correlation structure of the traffic and constructing signatures based on the observed correlation. The algorithm is evaluated by examining proprietary gaming traffic and also other known non-gaming protocols.

Keywords MMORPG · Effects of human behavior · Traffic characteristics · DPI

1 Introduction

Increasing online game popularity causes increasing loads on game servers. It is believed that understanding player behavior can aid in the understood of their impact on Quality of Service (QoS), network traffic characteristics, and it also helps in developing efficient protocols and architecture optimization techniques. The design of an accurate player model for Massively Multiplayer Online Games (MMOGs) makes it possible for future research to predict and simulate player behavior and population fluctuations over time. Further it makes it feasible of the evaluation of existing traffic models and architecture designs.

Current papers examined several factors why macroscopic level of network traffic of online games are just like that we experience. It was found that one of its important properties is the dependence on the gaming protocol itself [20], while another one is the influence of the player actions [9].

We have carried out a comprehensive research to examine how the player behavior and ingame environment affect the network characteristics of Massively Multiplayer Online Role Playing Games (MMORPG). Based on these results, our paper has the following contributions:

- We introduce a novel user behavior detection method.
- We show analysis results using our method with measurements from an operational network.
- We present a new dynamic signature traffic identification method.
- We show how the dynamic signature method can be applied for moving packets identification.
- We introduce a model for character movements based on fractional Brownian motion.
- We also show how our method can be used for signature identification of popular applications.

Our paper discusses in details how player movement and gaming environment changes affect the traffic characteristics at a macroscopic level. Detection of player actions would make it possible to answer such interesting questions as e.g., whether the self-similar property of human behavior appears in the traffic as well due to the player actions. We need a method to collect the necessary statistics on player actions from data sources containing bulk information. As there are a number of different games with proprietary protocols, a deep packet inspection method is not generally applicable. A statistical method would be desirable as it is general and works for even new upcoming games as well. It would make it feasible to analyze vast number of gaming sessions in non-intrusive measurements obtained at high network aggregation level. In current literature there is no such method which can collect the necessary statistics, thus, we propose a method which builds on heuristics to deduce the movement or idle state of the player and its surrounding area conditions from passive measurements.

We observed the internal structure of the network traffic when the player is in moving or stalling state and the environment around him when it is crowded with other players or deserted. This observation has been examined and validated with several methods. It was found that even if a heuristic method based on statistical properties of the traffic has several fallbacks it can be applied on the recognition of game states with high hit ratio. The proposed method was thoroughly validated on network traffic of World of Warcraft [6] and Silkroad Online [25]. It was also tested on Guild Wars [41], Eve Online [7] and Star Wars Galaxies [34] to check how generally applicable the method is. Although this list does not contain every possible MMORPGs but we argue that the observed traffic characteristics are common among MMORPGs, thus the proposed player state detection method may be applicable for most MMORPGs.

We examined the similarity of ingame player and real world human behavior. It cannot be expected that the two environments would completely be the same, but some properties which were found earlier in connection with human activities are expected to appear in the gaming environment as well.

In the second part of the paper we focus on small time-scale analysis of network traffic (Section 5). We examine how the effects of human behavior can be tracked in network traffic at packet level. Player behavior and movement related packets encode the ingame location of the players. We found that this packet level information shows similar statistical properties that characterizes human motion models. We developed a method that can automatically deduce the movement related payload segments of an unknown protocol. The movement segments are always changing values—dynamic signatures—comparing to the static ones that current Deep Packet Inspection (DPI) methods apply. We present several other dynamic signature types to create dynamic signatures in general protocols for e.g., timestamps or sequence numbers. The dynamic signature is especially useful to aid in traffic recognition of gaming traffic. Wide-spread MMORPGs e.g., World of Warcraft [6] uses protocol header encryption which makes it difficult to recognize them by current DPI methods. The dynamic signature analysis makes it possible to grab other characteristic features of the gaming protocol and recognize them even if it is unfeasible to construct a well-defined static signature for the protocol headers.

2 Related work

Recent research of networked games focuses on modeling low time-scale characteristics, such as packet inter-arrival times and packet sizes, to develop efficient control methods of network traffic and optimally utilize gaming servers. Since the popularity of multiplayer online games rapidly increases there is an urgent need to develop more realistic traffic models.

The predecessor in time, look and feel and server status update mechanism of recent MMORPGs are the multiplayer First-Person-Shooter (FPS) games. Authors of [12] discovered the dependence of server traffic bandwidth on ingame events. In their later work [4] they managed to refine their measurements and divided

UT2003 [19] into the fundamental user interaction components of movement and shooting, sub-dividing movement up into simple and complex movement, and sub-dividing shooting based on the precision of the weapons being shot. In [45], the authors created a traffic model differentiating game-states focusing on such problems which will be critical to networks with shared resources. Basically they differentiated active and idle game periods which are related to the problem of proper radio resource allocation. Authors of [13] condensed the result of ingame events as a correlation in packet sizes of Quake4 [23] traffic. They proposed an ARMA(1,1) model to capture the time series behavior of the gaming traffic. In [48] authors proposed the Networked Game Mobility Model (NGMM), for synthesizing mobility in FPS networked games. NGMM utilizes application level aspects of networked game traces to statistically model FPS games.

Today a large part of the gaming traffic is generated by MMORPGs, thus several works dealing with this type of traffic appeared. In [28] authors analyzed Lineage II. It was one of the world's largest MMORPGs in terms of the number of concurrent users at that time. They found that the bandwidth usage of the server traffic is about ten times larger than the client traffic. This asymmetry is due to the fact that the server transmits all the information to construct the visual environment for the clients in the same region.

Authors of [16] provided the first look onto player behavior analysis through a measurement study on World of Warcraft [6]. Their goal was to answer the following questions: how does the population of the virtual world change over time, how are players distributed in the virtual world and how do they move in the virtual world. They found that player distribution appears to occur on a power-law distribution, and players move to only a small number of zones during each playing session. In particular, certain locations which have rare equipments tend to be popular places to visit. Further, large cities which have a wide range of services also tend to be popular. Thus, players will tend to congregate in popular locations.

In [20] the authors showed what happens when an avatar performs different actions in the virtual world, at different places and under different network conditions. The Second Life [32] client makes intensive use of network resources. They identified the reasons of high bandwidth consumption e.g., ingame music, numerous unique character clothes, etc. deriving from several game servers. Authors of [31] collected mobility traces of 80k avatars spanning more than 20 regions over two months in Second Life. They analyzed the traces to characterize the dynamics of the avatars mobility and behavior. They extracted character position information by parsing update packets from the servers. They discussed the implications of their findings in connection with the design of peer-to-peer networked virtual environments, mobility modeling of avatars. Their measurements suggest that a hybrid mobility model that incorporates both random way-point mobility model (for outdoor) and pathway mobility model (for indoor) would be suitable.

Authors of [9] found that the characteristic features of MMORPGs are the strong periodicity, temporal locality, irregularity, and self-similarity. The self-similarity property was explained by the existence of a hidden on-off process according to the authors. Their hypothesis was that on-off periods are due to the player active and idle periods. They defined a player active if the player sends packets above a threshold and idle if the packet sending rate of the player is below a threshold. Authors of [10] found that MMORPG bots traffic is distinguishable from human players by the regularity in the release time of client commands, the trend and magnitude of traffic burstiness in multiple time scales, and the sensitivity to network conditions.

Our work can extend the above studies with several aspects. Chen et al. [10] provided us with the proof that human behavior is present in the network traces at macroscopic level. We argue that the self-similarity property of the gaming traffic, which was found in [9] and was modeled with an on-off model does not fit due to the impact of human behavior. In this work we show that player behavior in the MMORPG traffic has Long-Range Dependence (LRD)[1] properties. Chen et al. [9] hints that the traffic rate changes of the game may be due to player actions, but the detection heuristic has not been validated and is not precise as will be shown later in this paper.

Our work extends the investigation of gaming traffic of [20] in the function of time gaining information about gaming environments, further we redefine the player activity applied in [9] to gain information about player behavior. Our work aims at deriving detailed character movement and gaming environment information as e.g., [20] but obtaining this information from non-intrusive measurements without the requirement of the detailed knowledge of the whole protocol as in e.g., [48] or [31].

In the second part of the paper (Section 5) the protocols are analyzed at packet content level. To the best of our knowledge studies focusing on the structure of the packets of the game protocols do not exist yet. Authors of [48] and [31] extracted the structure of the gaming protocol from the source code of the game [48] or manually with several empirical experiments [31]. In [16] authors used a scripting language [33] which is supported by World of Warcraft to automate ingame activities and extracted the positions of every character they encountered during game.

The most related papers in literature focus on automatic signature generation with fix protocol signature subsequences. In [22] authors explore automatically extracting application signatures from IP traffic payload content. In particular, they apply three statistical machine learning algorithms to automatically identify signatures for a range of applications. In a similar manner in [27], the authors introduce a system which ranks packet content according to its prevalence, and only generates signatures as needed to cover its pool of suspicious flows. Consequently, it is designed to minimize the number of signatures it generates. Their offline evaluation on real network traces reveals that their system can be tuned to generate signature sets that perform well on worm detection. It is interesting to note that automatic signature generation can be extended for even polymorphic worms which are able to alter their form on the network to deceive pattern recognition engines. Such a method is presented in [30].

Authors of [35] present three classification techniques for capturing statistical and structural aspects of messages exchanged in a protocol: product distributions of byte offsets, Markov models of byte transitions, and common substring graphs of message strings. They compare the performance of these classifiers using real-world traffic traces from three networks in two use settings, and demonstrate that the classifiers can successfully group protocols without a priori knowledge. They analyzed common plain-text protocols like e.g., SMTP, DNS, SSL.

[1]Practical meaning: the rate of decay of statistical dependence in a time series is slower than an exponential decay. See [5] for details.

Authors of [44] used Longest Common Subsequence (LCS) for the signature extraction step. The LCS is extracted from sample flows to be the signature of the given application. The algorithm compares two samples to get the longest common subsequence between them, and then compares it with other samples iteratively to refine it. They managed to produce signatures for several P2P protocols as well e.g., LimeWire, BitTorrent, Fileguri.

Our work focuses on the creation of an algorithm which can automatically grab the dynamically changing fields of the protocols. The requirements were to test player behavior in unknown protocols in an efficient way which resulted in a simplified character movement model comparing to player movement models in e.g., [48] for FPSs and [31] for MMORPGs.

3 User behavior detection methods

We introduce methods for user behavior analysis in this section. Our constructed methods are based on the analysis of active measurements. In the term *active measurement* we mean that we measured the traffic on the client side generated by a player, thus the user actions can be controlled. Authors of [9] studied game traffic characteristics with the help of power spectrum diagrams to reveal periodicity in the traffic. However, the power spectrum does not grab the characteristics of the traffic in the function of time. We decided to apply wavelet analysis [36] on the MMORPG traffic to exploit the following features: (a) one major advantage afforded by wavelets over power spectrum analysis is the ability to perform local analysis, (b) the other advantage is the capability of revealing aspects of data like trends, breakdown points and discontinuities in higher derivatives.

We created several active measurements in a controlled environment by capturing the generated network traffic of World of Warcraft [6] and Silkroad Online [25] and manually log the time of the different states of the player. After the measurement, we compared network characteristics and the ingame activity of the player. Examples from this experiment are shown in Figs. 1 and 2. The figures depict the bandwidth characteristics of the traffic coming from the server in the function of time. The two games have similar characteristics though World of Warcraft has about the twice the network bandwidth consumption as Silkroad. Observations on the traffic

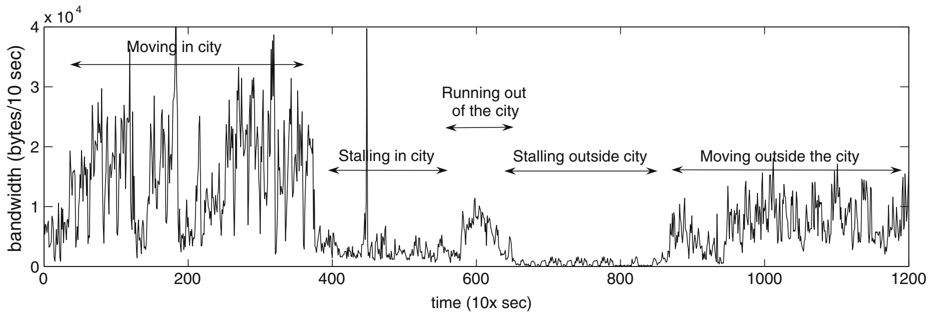


Fig. 1 World of Warcraft measurement bandwidth (bytes/10 s)

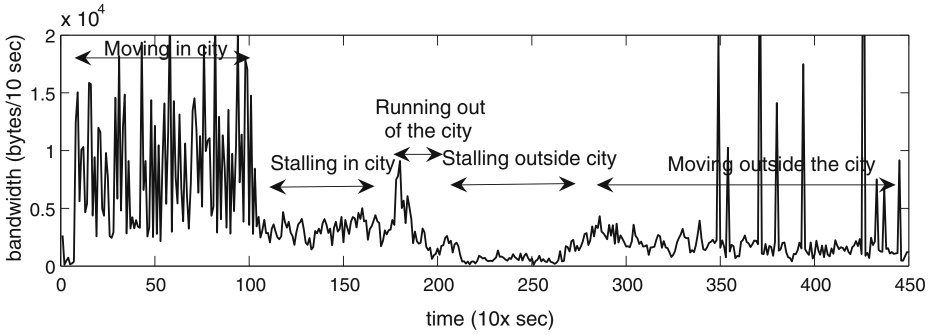


Fig. 2 Silkroad measurement bandwidth (bytes/10 s)

characteristics could be made in connection with player activity and the environment changes:

- *Moving/Stalling* The player moving actions imply constant location changes, which implies constant update of the environment information around the player. The player moves on areas with refreshed state and with obsolete information, thus update information occurs intensely but with variable frequency. During stalling, the state update regularity is independent from the player and synchronized by the server with more regularity. Our observation was that the character movements increase the noise in the traffic rate time series.
- *Inside/outside city* The density of the gaming environment of other players influences the number of independent actions which need state update at the player in their vicinity. Entering to a crowded environment induce a level-shift upward in the traffic rate, while leaving it results in a level-shift downward. Note that 'city' naming convention refers to the observation that the densely populated areas are usually in a city in the gaming environment, however it refers to any densely populated area.

To grab both of these characteristics in time we used wavelet decomposition (see Algorithm 1 in the Appendix for the Matlab [37] skeleton code of the algorithm in details):

- Reconstruction of the signal from the low frequency component gives hint about the gaming environment (densely populated city/extinct desert)
- The high frequency component gives hint about user activity (stall/run)

In Fig. 3 the wavelet decomposition of the traffic presented in Fig. 1 can be seen. It can be noted that the moving and stalling states are well-distinguishable on the high-frequency component: in case of moving, it becomes noisier. The environment changes can be well-tracked on the low-frequency component.

To improve the accuracy of the method we extended the wavelet decomposition with a state-machine which follows the changes in the environment. It tracks whether the player is inside city or outside city and does not let sudden changes in the environment if temporal transients occur in either of the frequency components. Figure 4 shows the defined states and state transmissions in the proposed detection algorithm.

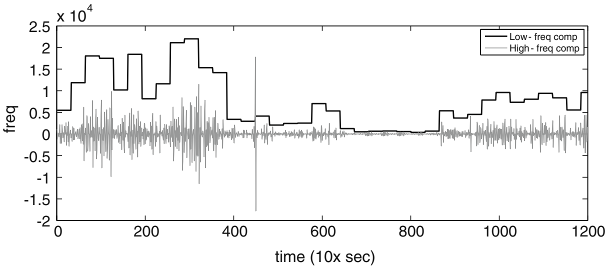


Fig. 3 Analysis of World of Warcraft traffic and the effect of different environments and character actions

The skeleton code in the Appendix follows these states and state transitions. The high frequency component change makes a state transition between stalling and moving, while the low frequency component change results in a state transition between the environments. The environment change is only possible with the player moving. This model lacks the possible state transition between stalling in and outside the city. This could occur when a player uses teleport. In this case the state machine gets into the proper state in a few additional steps, but keeps the false state hit ratio low.

The algorithm was constructed to adapt to the analyzed traffic, thus, no fix thresholds regarding the typical bandwidth have to be set. The advantage of this feature is that any MMORPG game traffic can be analyzed without previous parameter tuning.

It is important to note that the method is independent of the packet directions. We analyzed the server traffic in this section, but due to the per packet TCP ACKs, the client traffic is correlated to the server traffic. Thus client traffic also shows the presented traffic dynamics, though on a lower level. The advantage of server derived traffic is that its characteristics are more determined: the server is programmed to update player state in a designed manner that is less influenced by random network effects. The client-server traffic considering together can also fit into the presented algorithms.

Validation We collected and examined 9 h of MMORPG traffic (World of Warcraft [6] and Silkroad Online [25]) for the validation phase. The validation of the proposed method has been performed in several steps. In the **first step** we created several active measurements in a controlled environment by capturing the generated network traffic of the game at the client side and manually log the time of the different states of the player. After the measurement, the proposed method was applied on the network traces, and the states which were indicated by the proposed method and the logged

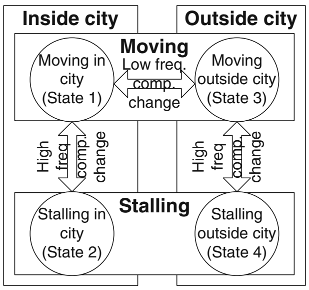


Fig. 4 States and defined state transmissions in the proposed method

states were compared. A state is the estimated action of the character in a given environment in a 10 s period. The definition of the states and their values are showed in Fig. 4. In Table 1, the 'manual logging' titled row shows that in this phase of the validation 74% of the states were exactly recognized and 14% of the states were missed with a nearby state. Nearby state means that the state can be reached within one state transition.

Second, we focused on Silkroad Online and World of Warcraft to check the accuracy of the method thoroughly and in bulk measurement in an automated way.

The following different methods can be collected to create game play logs:

– Capture network traffic, and parse the protocol e.g., [48]. Our aim was to analyze wide-spread MMORPGs. In the case of MMORPGs, the wide-spread ones are all commercial games meaning that the protocols are proprietary and practically reverse engineering and complete reimplementation of the game server would be required to catch the ingame events.
– Use ingame scripting languages e.g., [16]. There is no wide-spread ingame scripting language which is supported by most of the MMORPGs. This is not a general applicable method to analyze several MMORPGs with the same technique.
– Use key and mouse movement loggers. The advantage of this method is its completely game independence. On the other hand it is difficult to track the ingame events with this technique as the key and mouse events can refer to other commands than character movement. Therefore a complex parsing mechanism would be required to find out the player actions. Further it is unfeasible to extract the ingame environment from the key and mouse logs.
– We decided to capture ingame video. The ingame video provides straightforward information about the character movement and the surrounding environment of the player. This technique is game independent and the required code to extract the necessary information is simple. One drawback of this method is that neither space nor processor efficient as the live measurement generates huge video files that have to be processed later.

Several measurements were taken in Silkroad Online and World of Warcraft by capturing the network traffic and the ingame video during game play. The recorded video files were processed by a heuristic algorithm to create automatically player action and game environment logs to replace the manual work for bulk measurements. Our ingame video analysis code and an example ingame video can be found at [15].

The heuristic algorithm grabs the characteristics of the ingame screens of both the environment and the moving activity. As an example outside the city, the characters march through grassy fields and forests where majority of the pixels has green color or a gradient of green. Our prototype implementation counts the number of greenish

Table 1 The difference of the MMORPG state recognition algorithm and the ingame video processing algorithm

Difference	0	1	2	3
Manual logging	74%	14%	8%	4%
Video processing	68%	19%	10%	3%

pixels of every frame of the recorded ingame video file. (We are aware of that the implementation is not completely general in its current status, as e.g., a desert area where the environment is not greenish would mislead the algorithm. With a more complex color processing mechanism, our idea would be not restricted to specific gaming environments.) The ratio of the greenish pixels can be seen in the function of time in Fig. 5. We used the HSL color space (hue, saturation, lightness) to check if a pixel is greenish. HSL color space makes it possible to test whether the color falls into a given hue range, but its saturation and lightness can take different values which is due to e.g., the day/night visual effect in the games. Figure 5 shows that as the character started inside the city with running around for 15 min, the green pixels ratio—the green and non-green pixels ratio on a given screenshot—is low and remains low for that period even if small fluctuations occurred. As the character stops, the green pixel ratio remains relatively constant. After the 30th minute the player leaves the city and there is a significant rise of the green pixel ratio as in enters to the nature. The varying number of trees, grassy fields, roads, monsters, etc. results in fluctuations in the green pixel ratio, but the mean value remains high.

The player activity—whether player stalls or runs—is tracked by the comparison of the pixels around the legs of the player in two neighboring frames. The ratio of the differing pixels is low in case of stalling and high in case of moving. Figure 5 shows that in the first 15 min the character was running around, thus this ratio remained high. As the player stopped the ratio dropped and the mean of this value remained low during stalling. The above two metrics make it possible to track the activity of players and the game environments by creating simple limits. It is evident from the results that as the metrics can temporary fluctuate, it is advisable to smooth them e.g., by calculating their sliding mean value.

In Table 1 'video processing' titled row shows that 68% of the states were identified exactly and 19% of the states were missed by a nearby state. The validation showed that the algorithm misses mainly the city/outskirts transitions. It is clear that the term city is not perfectly proper as the buildings in the city and walls are just visual effects, thus the high bandwidth is in connection with the densely populated areas not the exact location if it is inside the city or not. It would be also possible to analyze the client generated traffic to aid in state recognition. This can be a task of further work.

In the **third phase** of the validation process, we created several active measurement traces of the following MMORPG games beside World of Warcraft [6] and Silkroad Online [25]: Guild Wars [41], Eve Online [7] and Star Wars Galaxies [34] to check how generally applicable the method is. We found that the method performs well in cases where the player population is high and the character actions generate

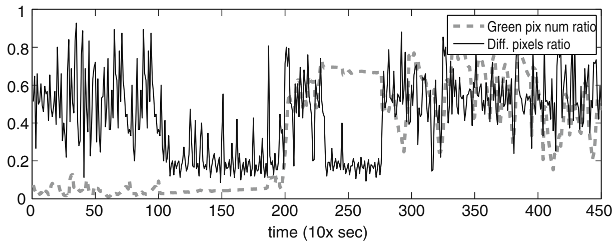


Fig. 5 Analysis of a Silkroad ingame video

considerable traffic. This states well for World of Warcraft, Silkroad Online, Star Wars Galaxies and Guild Wars as well. In the case of Guild Wars one can ask whether the method is affected by the game rule that leaving a city results in that only the members of the guild and the non-player-characters remains around the player. We measured that the traffic characteristics does not alter considerably. Our algorithm is inaccurate in the case of Eve Online and the explanation for this needs further investigation. We observed low traffic rate in our traces which makes it difficult to differentiate between the states.

4 Analysis of gaming traffic in an operational broadband network

Barabási [3] showed that the human activity patterns display bursty dynamics with interevent times following a heavy-tailed distribution. Crovella [14] showed that self-similar property of network traffic is caused by user 'think time'. In this section we show that human behavior can result in a non-heavy-tailed but strong-correlated process causing the same LRD effect.

We measured the network traffic of an operational third generation (3G) mobile broadband network. The measurement point was a middle-point interface, containing the subscribers outgoing and incoming traffic. Current 3G networks are fully capable of supporting broadband applications e.g., P2P, VoIP, gaming, streaming, etc. having the measured average RTT in [2, 42] far below 1250 ms which was measured to provide acceptable user experience in MMORPGs [21]. These applications exist in 3G mobile environment as it was shown in [46]. We preprocessed the measurement with filtering out the MMORPG traffic from the total traffic with the method introduced in [47]. Then we analyzed the filtered traffic with the methods presented in the previous sections. The examined traffic is a three-day-long passive measurement, which contained about 200 World of Warcraft flows and 100 other MMORPG flows.

In Fig. 6 we can see the cumulative distribution function of the duration of filtered MMORPG traffic. It can be seen that though the average game length is about 90 min, we can find even 5-h-long games. It is interesting to compare the session durations to e.g., [16] where authors found that 50% of the population remains online for 10 min or less. Their hypothesis was that players login to see if other friends or guild members are online and then leaving if not. Probably our measured player population was more determined to play and stay ingame.

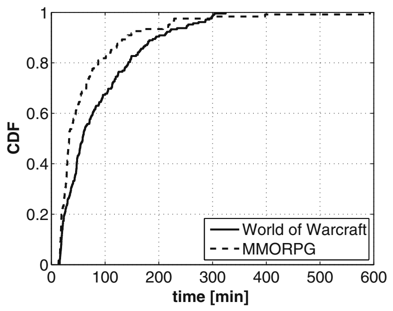


Fig. 6 Distribution of the duration of filtered MMORPGs

Table 2 The ratio of total time spent in a specific state by the players

	WoW		MMORPG	
	< 1 h (%)	> 1 h (%)	< 1 h (%)	> 1 h (%)
City run	23	20	19	19
City stall	35	42	29	35
Out run	30	27	25	30
Out stall	12	11	27	16

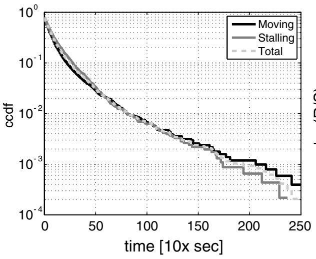
Table 2 shows the ratio of time spent in different states in MMORPGs in the case of the flows less than 1-h-long and more than 1-h-long. With this differentiation it is possible to check if longer MMORPG sessions are the consequence of longer 'away from computer' status of the players. We found that the state ratio do not significantly altered which means that these players take some rest during game play but remain active in most of the time. A possible explanation of the high ratio of stalling state can be explained by the socialization activity of players according to [49]. Unreal Tournament [49] says that a major part of the players like to use MMORPGs to chat and catch up with friends and not only play.

We intended to compare the observed player behavior characteristics in the gaming environment with **real world data**. Authors of [9] suggested that gaming traffic has Long-Range Dependence (LRD) and possibly self-similar property. They constructed an on-off model for the traffic and their hypothesis was that it may fit due to the impact of human behavior with the changes of active and idle states of the players. We argue that to unfold the origin of LRD property of the traffic is a difficult and often impossible task [14]. LRD property can be the result of a superposition of heavy-tailed on-off periods [52], TCP propagation property [50], the competition of TCP sources but it may also be caused as an artifact by non-stationarity.

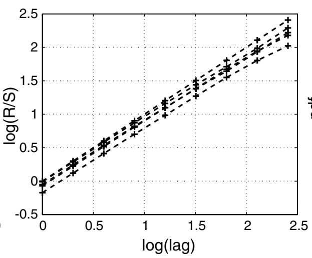
Figure 7a shows the sojourn time[2] ccdf of the moving and stalling periods in Wow traffic. It can be seen that the ccdf resembles to an exponential distribution, meaning that the distribution is not heavy-tailed. This excludes the explanation that the LRD property is the result of the heavy-tailed distributions of the movement state durations.

On the other hand, we checked the autocorrelation function of the time series of the states and it was found that the decrease of the correlation vs the lag is slow resulting in strong autocorrelation. Cross-checking this statement, we reconstructed the analysis performed in [9] by examining the time-series of the states in the function of time with the R/S plot [5] method. This time-series is such abstraction of the original traffic in which every effect—e.g., transport layer characteristics: TCP properties, packet sending rate, packet size—is eliminated and let us focus on the state change events of the player. Authors in [9] performed similar analysis except that they defined a player active if the player sends packets above a threshold and idle if the packet sending rate of the player is below a threshold. This property was found to occur due to the environment changes in our experiments. In our analysis we used our more sophisticated state definition. Figure 7b shows examples of the R/S plot of several WoW time-series constructed by the analysis of the filtered WoW traffic with

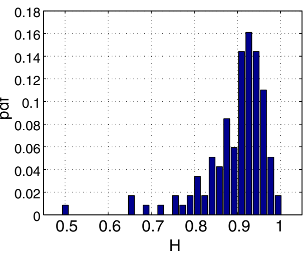
[2]The amount of time the player remains in the same state.



(a) Sojourn time ccdf in moving or stalling periods in Wow traffic



(b) R/S plots of the player states in several Wow sessions



(c) Estimated H-parameter pdf of th eplayer states in the Wow traffic

Fig. 7 Parameters of World of Warcraft sessions (**a**–**c**)

our proposed method. The R/S plots also support the high correlation property of the time-series of the player states. The estimated H-parameter distribution for all the filtered and analyzed WoW traffic can be seen in Fig. 7c with an average of 0.89.

Summarizing our findings of this section, we revealed the LRD property of the gaming traffic and we also showed that the popular explanation for the presence of LRD by heavy-tailed periods cannot be held.

5 'Dynamic signature' for traffic identification and its application for MMORPGs

Traffic identification has several approaches: one is traffic classification based on protocol header information, e.g., (a) used ports, based on (b) traffic characteristics based on e.g., packet length or packet interarrival times, (c) connection pattern based methods based on connection graph of the hosts and (d) Deep Packet Inspection (DPI).

Focusing on gaming traffic the following issues arises with the methods. *Port based classification* is becoming less and less accurate [40]. Current MMORPGs use dedicated ports e.g., World of Warcraft uses TCP port 3724, but their authentication protocol includes port negotiation for game server communication. The dedicated port usage could be changed from one day to another and this method becomes obsolete.

Several papers reported successful classification of several application classes including gaming traffic with *traffic characteristics based classification* e.g., [54, 55]. Nonetheless, most of the papers focused on FPS games which server traffic rate is constant. This characteristic can be captured well. The traffic characteristics of MMORPGs are influenced by several factors as it was shown in Section 3 and would be more difficult to find identifiable features. However, there is a lack of papers with exact measurements and experiences in this topic yet.

Connection pattern based methods e.g., [24, 26] assumed that gaming applications e.g., FPS games use P2P architecture to communicate. In this case P2P means distributed small scale client-server system. MMORPGs use strict client-server architecture which would make it difficult to differentiate the game server from any common server e.g., a web server based solely on connection pattern. On the other hand, the centralized architecture eases to collect the IP addresses of the game servers and recognize game traffic going to and coming from these IP addresses. The shortcoming of this technique is that gaming traffic of pirate or private servers[3] remain in the dark.

The other approach is the *Deep Packet Inspection* (DPI). In current DPI methods specific structures of the composition of fix byte signatures and 'do not care' segments are searched in the packet payloads (see Section 2 for details). There are several cases when there is a variable in the packets at a fix position. Our idea was to focus on the 'do not care' segments and analyze them. This dynamic signature can be recognized by statistical methods.

The dynamic signature proved to be especially useful applying on game traffic. Wide-spread MMORPGs e.g., World of Warcraft [6] uses protocol header encryption which makes it difficult to recognize them by current DPI methods. The dynamic signature analysis makes it possible to grab other features of the gaming protocol and recognize them even if it is unfeasible to construct a well-defined signature for the protocol headers. Games which transmit location information in coordinates were well recognizable by the proposed method. The payload segment which encodes the location of the players in the gaming environment shows such statistical properties that characterize human motion models. Our presumption was that such variables which statistically fit to motion models but actually containing not motion related information are rare.

It is important to note that some games are evolving basic encryption or obfuscation of payloads. Payload obfuscation or a payload compression may modify the statistical nature of different fields in the payload. This is not the scope of our current work.

6 Definition of 'dynamic signature'

We define the following three signature types: In a packet (see Fig. 8 as an example) there can be the usual fix signatures—flags (F)—, 'do not care' segments (?) and dynamic signatures on every byte position. We define the following field types as a dynamic signature:

- *Strong correlation* (C): There is a strong temporal correlation in the specific byte segment among the sequence of the packets. The increments of the packet values are random.
- *Walk* (W): There is a strong correlation in the specific byte segment among the sequence of the packets. Further, the values in the sequence of packets are random. These two properties make the time series similar to real world human motion models, thus it is considered as walking. In our method this field is considered as a fractional Brownian motion process.
- *Sequence number* (S): These fields are used in protocols typically for e.g., sequence numbers or timestamps.
- *Fix* (F): These fields are used in protocols as fixed headers or flags.

[3] It is difficult to add reference to sites collecting these servers as they are always on the move to avoid legal consequences.

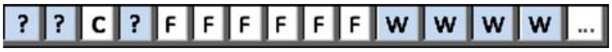


Fig. 8 Dynamic signature in a packet

In the case of every dynamic signature the length of representation is important. In the example packet in Fig. 8 the value in the 'walk' field is represented on four-byte-long.

An MMORPG example We created several experiments with active measurements in World of Warcraft [6] to examine its traffic. We measured the traffic of World of Warcraft and created an active measurement in such a way that the player actions were planned before the measurements to let us synchronize the network traffic with user actions. The analysis showed the high occurrences of 34-byte-long packets suggesting the candidate packets responsible for character movement information. We performed several experiments to find out how the packet is partitioned and which fields of the packets refer to a specific data. In one of the active measurements we moved with the character from a well-defined starting point, for 1 min to north (according to the ingame compass), 1 min to south, 1 min to west and 1 min to east. In another active measurement we run around in a big circle and finally we checked if we take a bigger wandering and return to the staring point whether the coordinates are in agreement with our expectations.

Figure 9 shows the reconstructed original path of the player. One derivation of this experiment is that by starting and ending at the same coordinates with the character, we can be sure that the coordinates are non relative, thus not only the movement vectors are transferred to the server but they are absolute coordinates. The other gained information is the relation of the packet sending time with the moving coordinates and with the ingame timestamps.

The assumed partition of the packet fields and their function:

- *1–6 byte* This is the protocol header encrypted part of the packet. This part contains the movement packet identification flag, which would ensure us that this is a movement packet.
- *7–10 byte* These are some kind of flags, which rarely change, and the values they take are either 0 or 1.
- *11–14 byte* This part seems to be an ingame sequence number or timestamp as it constantly increases.
- *15–18 byte, 19–22 byte, 23–26 byte*: One of the 3D coordinates used by the game.

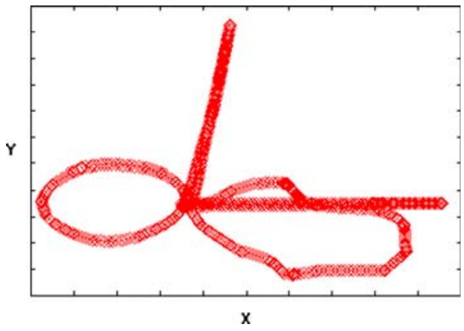


Fig. 9 The coordinate values in the active measurement depicted in 3D and projected to XY plane

– *27–34 byte*: We guess that this field is a character facing value.

7 Discussion on the 'walk' signature

Based on the main dependence characteristics of the character movement process our presumption was that fractional Brownian motion (fBm) can describe the changes of character movements. We have chosen two independent fBm models, where one fBm would be fitted for the X and another would be fitted to the Y coordinates. The modeling of the Z coordinate was neglected due to the fact that the terrain serves as a constraint in this case. The intuition behind the fBm model was our expectation that the character position coordinates exhibit high positive correlations. Therefore a player keeps going in the same direction with high probability, thus a simple random walk (e.g., Brownian motion) cannot describe its behavior. We think that our 'walk' model applies for a broad range of games because our fBm based model captures generally both the randomness and correlations of the players. Our model has a great potential for generalizations because tuning the parameter H (see Section 7.1) we can set different correlations.

7.1 Basic modeling definitions

A continuous time process W_t is called *fractional Brownian motion* (fBm) with parameter H, $0 < H < 1$, if W_t is Gaussian and self-similar. The increment process of fBm is called *fractional Gaussian noise* (fGn) [51]. The fGn exhibits positive correlations if $H > 0.5$ and we want to capture the important positively correlated character movement property by this process. This property means that if there is an increasing pattern in the previous 'steps', then it is likely that the current step will be increasing as well.

In order to validate our model assumption we have carried out a time series scaling analysis. We have used several tests but in this paper we show only the R/S test [51] and wavelet analysis [1] results. The R/S test can be applied in the following way: given an empirical time series of length N ($X_k : k = 1, ..., N$), the whole series is subdivided into K non-overlapping blocks. Now, the rescaled adjusted range $R(t_i, d)/S(t_i, d)$ can be computed for a number of values d, where $t_i = \lfloor N/K \rfloor (i-1)+1$ are the starting points of the blocks which satisfy $(t_i - 1) + d \leq N$.

$$R(t_i, d) = max\{0, W(t_i, 1), ..., W(t_i, d)\} - min\{0, W(t_i, 1), ..., W(t_i, d)\}$$

where $W(t_i, k) = \sum_{j=1}^{k} X_{t_i+j-1} - k\left(\frac{1}{d}\sum_{j=1}^{d} X_{t_i+j-1}\right)$, $k = 1, ..., d$. $S^2(t_i, d)$ denotes the sample variance of $X_{t_i}, ..., X_{t_i+d-1}$. For each value of d one obtains a number of R/S samples, which decreases from K for larger values of d, i.e., $d_{l+1} = md_l$ with $m > 1$, starting with d_0 of about 10. Plotting $\log R(t_i, d)/S(t_i, d)$ vs $\log d$ results in the R/S plot. Finally, a least squares line is fitted to the points of R/S plot, where both the R/S samples of the smallest and largest values of d are omitted. The slope of the regression line is an estimate for H.

Scaling properties of traffic can also be efficiently investigated by multifractal analysis via wavelet-based methods [1]. The discrete wavelet transform represents a data series X of size n at a scaling level j by a set of wavelet coefficients $d_X(j,k)$, $k = 1, 2, ..., n_j$, where $n_j = 2^{-j}n$. Define the q^{th} order Logscale Diagram (q-LD) by the log-linear graph of the estimated q^{th} moment $\mu_j(q) = 1/n_j \sum_{k=1}^{n_j} |d_X(j,k)|^q$ against the octave j. Linearity of the LDs at different moment order q suggests the scaling property of the series, i.e. $log_2\mu_j(q) = j\alpha(q) + c_2(q)$ where $\alpha(q)$ is the scaling exponent and $c_2(q)$ is a constant. In our test results we plot $y_j = log_2\mu_j(q)$ for $q = 2$ which is called the second-order logscale diagram (LD).

7.2 Results of the statistical tests

The filtered gaming traffic of an operational broadband network (see Section 4 for details) was analyzed with the methods presented in the previous sections (Section 7.1). An important assumption in statistical analysis and modeling is the stationarity. The character movement coordinates contain jumps which are the characteristics of non-stationarity processes. With the inspection of the movement packets of the World of Warcraft flows we found that there are several jumps in the coordinate values which suggest that the character changed its location from one place to another with very high pace. This can happen in case of the character used teleports. To create time series holding the stationarity property for analysis, we constructed first the time series of increments from the original (cumulant) time series of the coordinates. For this increment time series we applied a filter and removed the outliers from the time series. Therefore we created a pre-processed stationary increment time series from our original data.

7.2.1 Hurst parameter estimation

First we applied an R/S test [51] for the increments. The test resulted in that the average estimated H parameter is 0.73 for both the X and Y coordinates calculated from 200 independent time series. The cumulative density function of the H parameter is depicted in Fig. 10. As the linear fitting to the R/S plot is difficult to tune and need a lot manual refinement, we decided to apply the wavelet based method [1] because it is more robust than the R/S estimator. During the examination of the time series of the increments, the wavelet method estimates H by the $H = \frac{\alpha+1}{2}$ formula. The results for the different time series can be seen in Fig. 10 where the

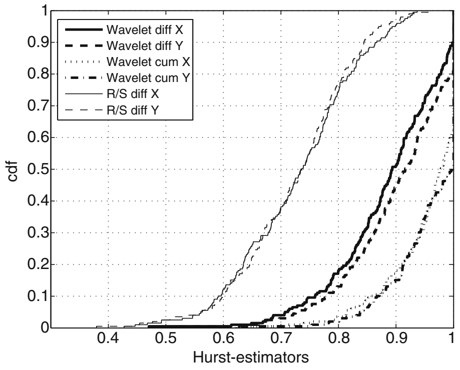


Fig. 10 The cdf of the Hurst estimators for the different traces

average H-parameter is 0.95 for X coordinate and 0.96 for the Y coordinate time series.

An additional validation step has been performed by applying the wavelet based H-estimators on the original cumulant time series, which were constructed by summarizing the pre-processed increments. These pre-processed increments were obtained by the above mentioned filtering process in order to keep the stationarity property of the time series. The formula for the estimation of H changes to $H = \frac{\alpha-1}{2}$ during the examination of the cumulant data. The average H estimation resulted in 0.89 for the X coordinate and 0.9 for the Y coordinate time series. (The logscale diagram is depicted in Figure 11). It can be clearly seen from the figure that the cumulative character movement process is consistent with statistical self-similar process and supports the fBm model assumption.

Therefore our presumption that the character movements can be well modeled with two independent fractional Brownian motion for the X and Y coordinates has been validated by the statistical test results of real data. An appropriate H parameter is the average of all the estimation results from R/S, increment logscale and cumulant logscale analysis over 200 samples. It resulted in $H = 0.9$.

7.2.2 Filtering out non-random processes

An important thing to note in connection with the estimated Hurst parameter is that, such high values are typical of non-random processes e.g., in the case of a process which has fix increments. To significantly differentiate this case from the one where randomly distributed values occur in the increments, the randomness of the values in the timeseries is examined. In the case of a moving packet, if the difference of two neighboring coordinates were examined it is likely that the size of the character step would occur most of the cases. Therefore a sliding window has been defined and the difference of the coordinates of the first and last element of the sliding window is calculated. These difference values construct a time series, which has been grouped into bins to make it possible to properly estimate the *Relative Uncertainty* with the method which was used in [29, 53]. Suppose we randomly observe X for m times, which induces an empirical probability distribution on X, $p(x_i) = \frac{m_i}{m}$, $x_i \in X$, where m_i is the frequency or number of times we observe X taking the value x_i. Empirical

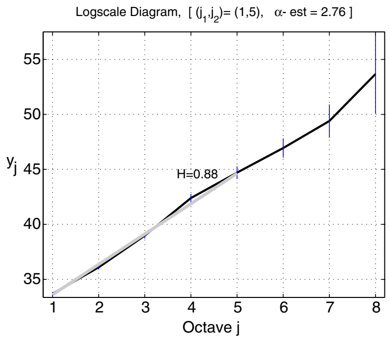


Fig. 11 An example logscale diagram of the time series of X coordinates

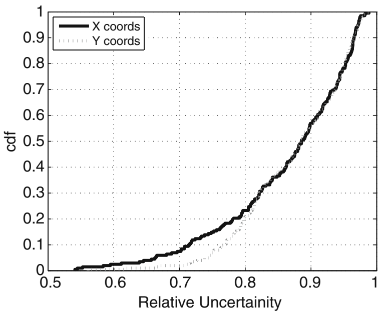


Fig. 12 The cdf of the estimated Relative Uncertainty of the difference of the moving packets

entropy of X can be calculated by $H(X) = -\sum_{x_i \in X} p(x_i) log p(x_i)$. Let N_X denote the number of discrete values X can take. The maximum entropy is $H_{max}(X) = \log \min\{N_X, m\}$. *Relative Uncertainty* can be defined as $RU(X) = \frac{H(X)}{H_{max}(X)}$. With this normalization, the value of Relative Uncertainty is in the [0, 1] range. The higher values mean less predictable behavior, the lower values mean more deterministic behavior. In particular $RU = 0$ means that X has only one possible value, while $RU = 1$ means that X is uniformly distributed.

In Fig. 12 it can be seen that the average of the RU values is 0.87, and there is not any occurrence of the RU below 0.5 meaning that the high value of the estimated RU is a strong proof of the fact that differences of the coordinates in the moving packets are randomly distributed.

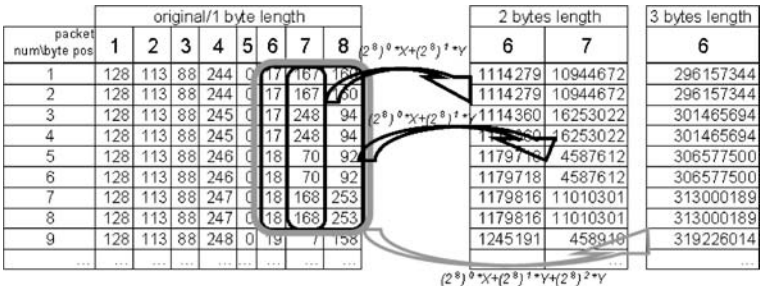
8 'Dynamic signature' construction algorithm

In the previous section, a model for the character position has been constructed. This model makes it possible to introduce a method in this section to identify the moving packets based on time series analysis in MMORPGs. The presented method is not restricted for this purpose. It can be applied more generally, where the same packet structure is used for transmitting the same kind of information.

8.1 Temporal time-series construction

The working mechanism of the constructed algorithm (Algorithm 2) is presented on a World of Warcraft traffic measurement. The preconditions during the examination of a set of possible moving packets are that it is neither known if they are really moving packets nor the fields where the different values take place. During the examination of the values it is critical to know on how many bytes the values are represented. The presented algorithm steps through all byte positions and creates time series of all possible byte lengths (Algorithm 2 Steps 3,4).

In Fig. 13 an example can be seen describing how the timeseries are constructed from the raw packet data for the analysis (Algorithm 2 Steps 5,7): when the number is considered to be represented in 2 bytes, regarding one row, the values in the 2 bytes representation output in the 6th column can be calculated by the $(2^8)^0 * A + (2^8)^1 * B$ formulae where A is the value in the 6th column and B is the value from



	original/1 byte length								2 bytes length		3 bytes length
packet num\byte pos	1	2	3	4	5	6	7	8	6	7	6
1	128	113	88	244	0	17	167	160	1114279	10944672	296157344
2	128	113	88	244	0	17	167	160	1114279	10944672	296157344
3	128	113	88	245	0	17	248	94	1114360	16253022	301465694
4	128	113	88	245	0	17	248	94	1114360	16253022	301465694
5	128	113	88	246	0	18	70	92	1179718	4587612	306577500
6	128	113	88	246	0	18	70	92	1179718	4587612	306577500
7	128	113	88	247	0	18	168	253	1179816	11010301	313000189
8	128	113	88	247	0	18	168	253	1179816	11010301	313000189
9	128	113	88	248	0	19	7	158	1245191	458910	319226014
...	...	...	...	...	...	...	...	...	...	...	...

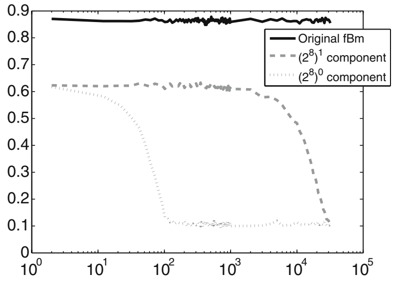
Fig. 13 An example describing how the timeseries are constructed from the raw packet data

the 7th column. The result of our model (see Section 7) was that we can consider the coordinate values as the result of a fractional Brownian motion with H close to 1. Therefore there is a strong correlation between the values of the same fields in the time series. The same test should be performed during the search for this structure in an unknown traffic. We knew that the original fBm in the examined World of Warcraft traffic was represented on four bytes. We examined how the test results altered if the original $fBm \sim \sum_{n=0}^{i}(2^8)^n X_n$, where i is the length of byte representation of the value, is decomposed to the X_n components.

8.2 H-parameter examination

The algorithm searches for fBm-process candidates (Algorithm 2 Step 9). We made several simulations regarding the Hurst-parameter test results of the fBm process decomposing it into different byte-representation ranges. Figure 14 shows the result of such simulation. The original timeseries is an fBm process with $H = 0.8$ parameter. It can be seen that the variance of the fBm process determines the range where the fBm properties can be observed after the decomposition. As the original process was tested on its original representation range, the H-parameter testing method finds the same H-parameter independently from the variance. The tested H-parameter of the $(2^8)^0$ component falls below 0.5 in the case of $\sigma > 30$, while the $(2^8)^1$ component tested H-parameter falls below 0.5 approximately in the case of $\sigma > 10^4$.

Fig. 14 The H-test results of an fBm process in the function of variance

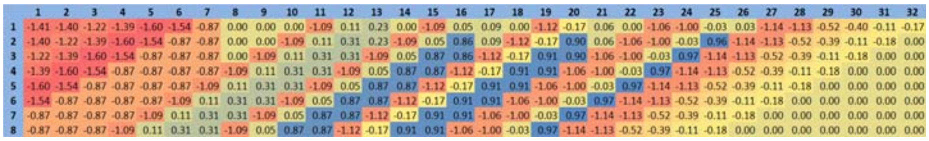


Another important thing to note is that the H-value of the original process is always higher than the decomposed one. This means that even if the variance of the tested fBm process is so low that it falls in the range of a smaller component, considering all components the test results in a better fit for the original process meaning a higher H-value.

If we examine the H-test results on the different byte positions with different length of representation of the World of Warcraft movement packets (see Fig. 15) it can be seen that considering the values as one-byte long numbers there is no proper H-parameter ($0.5 < H < 1$). Considering the values at least two-bytes long, the proper H-parameter appears at the 16, 20, 25 positions, meaning that the 16–17, 20–21, 25–26 bytes are proper candidates of walking fields. Checking the H-parameter in the next row, meaning that considering the representation longer in 3 bytes we can see the same H-parameter for the 16–18, 20–22 positions but a non-proper value for the 25–27 bytes. This means that considering them longer does not deteriorate their fBm property. World of Warcraft represents the coordinate values in the little-endian form, thus the least significant byte is in the left most position. If we take into consideration that the coordinates are represented in four-bytes-long, we can conjecture that the least significant byte varies the most, and the others far less, going toward the most significant value. Considering the byte representation longer in our case means that:

1. If the variance of the process falls into a more proper component, the H-value will fit better to the original process
2. If the new byte value does not alter the H-parameter that means a constant level-shift in the considered time-series
3. The most-significant value is not an fBm process any more thus the H-parameter will drop.

In the 16–18, 20–22 positions the case 2. occurs, while in the 25–27 byte positions, the case 3. Moving one column left, checking what happens if the longer byte-representation is considered from a former position, we can see the increase of the H-parameter falling into the case 3. Considering them longer (15–4), the case 2. occurs, but for even longer byte representations case 3. occurs for (15–5), (15–6). When the most-significant bytes of the considered time series is again a real fBm process—which is the next walking field of the World of Warcraft packet—then the H-value becomes proper again. These considerations are implemented in the proposed algorithm in Algorithm 2 Steps 22–30.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	-1.41	-1.40	-1.22	-1.39	-1.60	-1.54	-0.87	0.00	0.00	0.00	-1.09	0.11	0.23	0.00	-1.09	0.05
2	-1.40	-1.22	-1.39	-1.60	-1.54	-0.87	-0.87	0.00	0.00	-1.09	0.11	0.31	0.23	-1.09	0.05	0.86
3	-1.22	-1.39	-1.60	-1.54	-0.87	-0.87	-0.87	0.00	-1.09	0.11	0.31	0.31	-1.09	0.05	0.87	0.86
4	-1.39	-1.60	-1.54	-0.87	-0.87	-0.87	-0.87	-1.09	0.11	0.31	0.31	-1.09	0.05	0.87	0.87	-1.12
5	-1.60	-1.54	-0.87	-0.87	-0.87	-0.87	-1.09	0.11	0.31	0.31	-1.09	0.05	0.87	0.87	-1.12	-0.17
6	-1.54	-0.87	-0.87	-0.87	-0.87	-1.09	0.11	0.31	0.31	-1.09	0.05	0.87	0.87	-1.12	-0.17	0.91
7	-0.87	-0.87	-0.87	-0.87	-1.09	0.11	0.31	0.31	-1.09	0.05	0.87	0.87	-1.12	-0.17	0.91	0.91
8	-0.87	-0.87	-0.87	-1.09	0.11	0.31	0.31	-1.09	0.05	0.87	0.87	-1.12	-0.17	0.91	0.91	-1.06

	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	0.09	0.00	-1.12	-0.17	0.06	0.00	-1.06	-1.00	-0.03	0.03	-1.14	-1.13	-0.52	-0.40	-0.11	-0.17
2	0.09	-1.12	-0.17	0.90	0.06	-1.06	-1.00	-0.03	0.96	-1.14	-1.13	-0.52	-0.39	-0.11	-0.18	0.00
3	-1.12	-0.17	0.91	0.90	-1.06	-1.00	-0.03	0.97	-1.14	-1.13	-0.52	-0.39	-0.11	-0.18	0.00	0.00
4	-0.17	0.91	0.91	-1.06	-1.00	-0.03	0.97	-1.14	-1.13	-0.52	-0.39	-0.11	-0.18	0.00	0.00	0.00
5	0.91	0.91	-1.06	-1.00	-0.03	0.97	-1.14	-1.13	-0.52	-0.39	-0.11	-0.18	0.00	0.00	0.00	0.00
6	0.91	-1.06	-1.00	-0.03	0.97	-1.14	-1.13	-0.52	-0.39	-0.11	-0.18	0.00	0.00	0.00	0.00	0.00
7	-1.06	-1.00	-0.03	0.97	-1.14	-1.13	-0.52	-0.39	-0.11	-0.18	0.00	0.00	0.00	0.00	0.00	0.00
8	-1.00	-0.03	0.97	-1.14	-1.13	-0.52	-0.39	-0.11	-0.18	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Fig. 15 The H-test results on the different byte positions with different length of representation of the World of Warcraft movement packets

8.3 Analysis of randomness

High H-parameter can be obtained by testing the timeseries with fix increments which are part of the gaming protocol message like sequence numbers or timestamps. The H-parameter testing has to be extended with the examination of the randomness of the candidate byte positions to differentiate a fBm-process and a sequence number. To do this, the Relative Uncertainty (for details see Section 7) of the differences of the coordinates in a sliding window is calculated, and if the value is less than 0.8 then it is a sequence number with fix increments otherwise if RU is high it is considered as a moving packet (Algorithm 2 Steps 13–17). The sliding window is necessary as neighboring values can be fix in case of walking behavior—size of the step—, but the coordinate distance differences in a sliding window show randomness due to the movement in several dimensions. To differentiate the sequence numbers from the fix values in a specific position, the original time series is grouped into bins—the number of bins is selected in the function of the examined range of values—and the relative uncertainty is calculated for the binned values. If the RU is low then the examined values can be regarded as fix values (Algorithm 2 Steps 18–20).

Finally, the still empty fields are filled with the longest non-walking type fields from the state and H-parameter test tables (Algorithm 2 Steps 31–36).

9 Signatures of some popular applications

Several active traces of different applications have been examined by the introduced algorithm. The list of results can be seen in Table 3. The meaning of the columns of Table 3 are the following:

1. *Application*—The first column refers to the examined application
2. *Size*—The second column refers to the total length of the signature
3. *Signature*—The third column shows the constructed dynamic signature
4. *Endianess*—The fourth column shows whether the value is represented in **B**ig or **L**ittle endianess
5. *Ratio*—The fifth column shows the ratio of this packet comparing to the total number of packet in our traces of the given application

First, several games from the FPS, RTS and MMORPG genres have been examined. World of Warcraft [6] was the first target of our inspection and beside the 34-byte-long movement packet which was thoroughly examined in Sections 6–8, another packet of 71-byte-length showed correlation structures. Our active measurements showed that this packet is in connection with the visit of mission-giving non-player characters in the gaming world. When the player approaches to such a character and clicks on it, the mission which can be given by that character is sent by the server. The correlation is due to the closeness of these mission givers around a specific location e.g., in a city, and as the players visit them one by one, it induces a correlation in the transmitted coordinates. Other MMORPGS were also examined by the algorithm, and Silkroad [25] was found to contain a fBm structure. It is interesting to note that the values of the byte positions which were found to be fBm-like do not show such characteristics as an absolute position value which was

Table 3 Application signature patterns /C-strong correlation, F-flag, S-sequence number, W-walk, ?-not specified/

Application	Size	Signature	Endianess	Ratio
Age of mythology [39]	10	?{1} CCCCCCC{7} SS{2}	B	19%
Command & conquer 3 [18]	12	FFFFCCCCCCC{11} ?{1}	L	25%
Command & conquer 3	16	???W{4} CCCWW{5} CC{2} ?CCC?{5}	L	18%
C&C Generals [17]	23	WW{2} ?C{2} SSSSSSS{7} FCCCCC{6} CCCCC{5} ?{1}	L	19%
C&C generals	31	WW{2} ?C{2} WCCCC{5} FFCCC{5} ??CCW{5} FC{2} C{1} ?{1} CCC{3} CCCC{4} ?{1}	L	16%
Cossacks - back to war [8]	32	CCCCC{5} FFCC{4} FC{2} FC{2} F{1} ?{1} CW{2} CC{2} ?CCC??CC?CC??{13}	L	4%
Cossacks - back to war	18	W{1} CCWWWWWWWC{10} SSS{3} CC{2} ??{2}	L	93%
Gnutella [43]	23	?{1} SSSS{4} CCCFFFFF{8} ?{1} F{1} ????????{8}	B	80%
MSN Messenger [38]	Varies	??{2} SS{2} CCCW{4} CCCCCCC{7} ?{1} C{1} ???{3}	L	100%
Silkroad [25]	6	CCCC{4} ??{2}	L	5%
Silkroad	20	?{1} CC{2} ????W???C{9} C{1} ?{1} CC{2} CC{2} ??{2}	L	4%
Silkroad	30	??????W{7} WC{2} ??????W{7} C{1} ??????{6} C{1} ??C{3} C{1} ??{2}	L	11%
UT2003 [19]	20	W{1} ??{2} C{1} ???{3} WC{2} ???CC{5} CC{2} C{1} ???{3}	L	17%
UT2003	22	WWWWC{5} ?W???W?WW??C?{13} CCC{3} ?{1}	L	18%
World ot Warcraft [6]	34	??{2} C{1} ?{1} CCCWWW{6} WWWW{4} WWWW{4} WWWW{4} WWWW{4} WWWWCCC{7} ?{1}	B	25%
World ot Warcraft	71	????{4} CCCCC{5} WCCC{4} ??? {3} C{1} ????W?{6} C{1} CC{2} CCCFFFFFFF{10} ??{2} C{1} CC??{4} CFFFF{5} CC?{3} CCW{3} ???{3} CC{2} C?{2} CW{2} ???{3} CCC{3} ?{2}	B	4%

found in World of Warcraft. We conjecture that these packets contain the movement vectors not the actual positions.

Games from the First Person Shooter (FPS) genre have been also examined by the introduced algorithm. FPS games are typical of transmitting movement vectors rather than absolute positions. A good example was found during the examination of the packets of UT2003 [19] (traces are available at [11]).

Games from the Real Time Strategy (RTS) genre have been also examined by the introduced algorithm. Age of Mythology [39], C&C Generals [17] and Cossacks [8] contain packets containing timestamps or sequence numbers which are responsible for maintaining the synchronization among the gaming clients. Beside this there are other type of messages in the packets which seem to be in direct connection with the gaming environment itself: probably the number of active units in the gaming environment as it shows an increase overall the game session.

Our method can examine non-gaming applications as well. The traffic of a Gnutella [43] client has been examined and it was found that there are similar packets to the gaming protocols in terms of having the same size and same packet structure. Two UDP packets of Gnutella were found to contain the correlation characteristics in the time series of the packets.

Other applications generating varying packet sizes are also examined. MSN Messenger [38] VoIP traffic can be analyzed by selecting one direction of the flow, than the longest packet of the flow is selected and the smaller packets byte values are filled with uniformly distributed random values. In this way the dynamic signature search can be executed on the packets having varying size. MSN Messenger works with RTP messages thus it is also a good candidate to validate our algorithm. The algorithm finds the first two bytes of the RTP message where several flags take place. The 3–4 byte is the sequence number in the RTP message and it is found to be fBm-like according to the algorithm. The next 4 byte is also found to be fBm-candidate: the 5–8 byte is the place of the timestamp. The next 7 bytes are considered as coherent: the 9–11 byte is the place of the synchronization source identifier, which is a fix values on its own, but 3 additional bytes are added to this by the algorithm. Examining the payload of the MSN RTP packets, the first byte of the RTP payload seems to be fix, and the second and third byte varies in small steps. Interesting to note that there are other two byte fBm-like values at the 34–35, 36–37, 58–59, 82–83, 84–85 bytes of the MSN RTP packet. We conjecture that this property is due to the used codec.

10 Conclusion

In this paper, the traffic of the MMORPGs is examined with special focus on the effects of player behavior at a macroscopic level on the gaming traffic. By understanding how the gaming traffic is influenced by the activity of the players and the changes in the gaming environment, detection methods based on time series analysis are introduced to grab specific events and states during the game play.

The proposed methods were applied on traffic traces obtained from live operational broadband networks in order to analyze the player behavior in real network situation. Both the user actions and the environment around the player were recognized in the MMORPG network traffic.

The collected data in the gaming environment was compared to data obtained from the real world environment. We revealed the LRD property of the gaming traffic and we also showed that the popular LRD explanation by heavy-tailed periods cannot be held. We also demonstrated that the player behavior affects the traffic characteristics at a macroscopic level.

An additional contribution of the paper is the introduction of a novel model and algorithm to extend the Deep Packet Inspection traffic classification method. We focus on the analysis of variable length byte signatures. The model captures the variation of the dynamic byte segments and provides parameters for the algorithm. The proposed algorithm considers the packets of the same flow as a time series and examines the H-parameter of the values of the different byte positions. The algorithm constructs protocol specific signatures of the examined traffic, which makes it possible to identify the protocol in a traffic measurement even in those cases when the signature in a protocol is not a fix value. As a proof-of-concept several proprietary gaming traffic and known traffic types have been examined with the introduced algorithm and the existence of the fBm-like structure is proved to exist in several cases.

Appendix

Algorithm 1: MMORPG state recognition algorithm

```
   Input: rate
   Output: out
1  nx = length(ina); out = zeros(size(ina));
2  [C5,L5]=wavedec(ina,5,'db1');A5 = wrcoef('a',C5,L5,'db1',5);
3  [C1,L1]=wavedec(ina,1,'db1');A1 = wrcoef('d',C1,L1,'db1',1);
4  medF=mean(abs(A1));maxF=max(A1);
5  phase_length=30 /depends on the wrcoef paratmeteres/;
6  state=0 /init=stalling, just where?/;
7  if (abs(A5(1)-min(A5)) < abs(A5(1)-max(A5))) then
8      state=4;
9  else
10     state=2;
11 for i=2:nx, do
12     frek_min=abs(abs(A1(i))-0);
13     frek_max=abs(abs(A1(i))-medF);
14     p=polyfit([1:phase_length*3],ina(i-phase_length:i+phase_length),1);
15     if (state==4) AND (frek_max<frek_min) then
16         state=3 /stalled outside, started to move/;
17     else if (state==2) AND (frek_max<frek_min) then
18         state=1;
19     else if (state==3) AND (frek_min<frek_max) then
20         state=4 /stalled and started/;
21     else if (state==1) AND (frek_min<frek_max) then
22         state=2;
23     if (state==1) AND (A5(i)<A5(i-1) AND p<0.5) then
24         state=3 /it moved out to the desert/;
25     else if (state==3) AND (A5(i)>A5(i-1) AND p>0.5) then
26         state=1;
27     out(i)=state;
```

Algorithm 2: Dynamic signature construction

```
Input: $M[i][b]$:=packets for examination, $i$: the $i^th$ packet, $b$: the byte position in the packet
Output: $Out_{state} = \oslash$ /Output field type/, $Out_{length} = \oslash$ /Output field length/
1  $H_o = \oslash$ /Hurst-test results/;
2  $S = \oslash$ /Guess for the field/;
3  for $w = 1,\ w < 8,\ w++$ do
4      for $k = 0,\ k < b,\ k++$ do
5          $X = zeros(size(M[:]))$;
6          $T = M[:][k : k + w - 1]$;
7          for $q = 1,\ q < w,\ q++$ do
8              $X = X + T(:, q) * ((2^8)^{q-1})$;
9          $H_o[w][k] = waveletestimation(diff(X))$;
10         $S_{temp} := 0$;
11         if $0.5 < H_o[w][k] < 1$ then
12             $S_{temp} := 1$ /fBm candidate H/;
13         $RU_{diff,window} = calc_{RU}(diff_{window}(X))$ (Section 7);
14         if $S_{temp} := 1\ AND\ RU_{diff,window} > 0.8$ then
15             $S_{temp} := 4$ /walking/;
16         else if $RU_{diff,window} < 0.5$ then
17             $S_{temp} := 2$ /sequence number/;
18             $RU_{diff} = calc_{RU}(create_{bins}(X, (2^8)^w/100))$;
19             if $RU_{diff} < 0.2$ then
20                 $S_{temp} := 1$ /fix/;
21         $S[w][k] := S_{temp}$;
22 for $w = 1,\ w < 8,\ w++$ do
23     for $k = 0,\ k < b,\ k++$ do
24         if $S[w][k] == 4$ then
25             if $S[w-1][k+1] < 4\ OR\ (S[w-1][k+1] == 4\ AND$
               $S[w-1][k+1] < S[w][k])$ then
26                 $Out_{state}[k] := 4$;
27                 $Out_{length}[k] := w$;
28                 for $j = k + 1,\ j < k + 1 + w,\ j++$ do
29                     $Out_{state}[j] = 0$;
30                     $Out_{length}[j] = 0$;
31 for $k = 0,\ k < b,\ k++$ do
32     for $w = 8,\ w > 0,\ w--$ do
33         if $0 < S[w][k] < 4$ then
34             if $!conflict(Out_{state}, Out_{length})$ then
35                 $Out_{state}[k] := S[w][k]$;
36                 $Out_{length}[k] := S[w][k]$;
```

References

1. Abry P, Veitch D (1998) Wavelet analysis of long-range-dependent traffic. IEEE Trans Inf Theory 44(1):2–15
2. Arjona A, Westphal C, Ylä-Jääski A, Kristensson M (2008) Towards high quality VoIP in 3G networks—an empirical study. In: AICT '08: proceedings of the 2008 fourth advanced international conference on telecommunications. IEEE Computer Society, Washington, DC, pp 143–150
3. Barabási A-L (2005) The origin of bursts and heavy tails in human dynamics. Nature 435:207
4. Beigbeder T, Coughlan R, Lusher C, Plunkett J, Agu E, Claypool M (2004) 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. ACM, New York, pp 144–151
5. Beran J (1994) Statistics for long-memory processes. Chapman & Hall, London, pp 71–86
6. Blizzard Entertainment (2004) World of Warcraft. http://www.worldofwarcraft.com

7. CCP (2003) Eve online. http://www.eve-online.com/
8. CDV Software (2001) Cossacks. http://www.cossacks.com/
9. Chen K, Huang P, Huang C, Lei C (2005) Game traffic analysis: an MMORPG perspective. In: NOSSDAV '05, New York
10. Chen K, Jiang J, Huang P, Chu H, Lei C, Chen W (2006) Identifying MMORPG bots: a traffic analysis approach. In: ACM SIGCHI ACE'06, Los Angeles
11. Claypool M (1995–2009) Mark Claypool's homepage. http://web.cs.wpi.edu/~claypool/
12. Claypool M, LaPoint D, Winslow J (2003) Network analysis of Counter-Strike and Starcraft. In: IEEE international performance, computing, and communications conference (IPCCC), April 2003
13. Cricenti A, Branch P (2007) ARMA(1,1) modeling of quake4 server to client game traffic. In: NetGames '07: proceedings of the 6th ACM SIGCOMM workshop on network and system support for games. ACM, New York, pp 70–74
14. Crovella ME, Bestavros A (1997) Self-similarity in world wide web traffic: evidence and possible causes. IEEE/ACM Trans Netw 5:835–846
15. Crysys (1993–2008) MMORPG analysis code. http://www.crysys.hu/~szabog/MMORPG/
16. Daniel DP, GauthierDickey C (2007) A measurement study of virtual populations in massively multiplayer online games. In: NetGames '07: proceedings of the 6th ACM SIGCOMM workshop on network and system support for games. ACM, New York, pp 25–30
17. Electronic Arts (2003) Command and conquer generals. http://www.ea.com/official/cc/firstdecade/us/generals.jsp
18. Electronic Arts (2007) Command and conquer 3. http://www.commandandconquer.com/default.aspx
19. Epic Games (2002) Unreal tournament 2003. http://www.unrealtournament2003.com/
20. Fernandes S, Antonello R, Moreira J, Kamienski C (2007) Traffic analysis beyond this world: the case of second life. In: NOSSDAV '07, Urbana, June 2007
21. Fritsch T, Ritter H, Schiller J (2005) The effect of latency and network limitations on mmorpgs: a field study of everquest2. In: NetGames '05, New York
22. Haffner P, Sen S, Spatscheck O, Wang D (2005) ACAS: automated construction of application signatures. In: MineNet '05, New York
23. id software (2005) Quake4. http://www.idsoftware.com/games/quake/quake4/
24. Iliofotou M, Pappu P, Faloutsos M, Mitzenmacher M, Singh S, Varghese G (2007) Network traffic analysis using traffic dispersion graphs (TDGs): techniques and hardware implementation, technical report, 2007. http://www.cs.ucr.edu/~marios/Papers/UCR-CS-2007-05001.pdf
25. JOYMAX (2005) Silkroad online. http://www.silkroadonline.net/
26. Karagiannis T, Papagiannaki K, Faloutsos M (2005) BLINC: multilevel traffic classification in the dark. In: Proc ACM SIGCOMM, Philadelphia
27. Kim H, Karp B (2004) Autograph: toward automated, distributed worm signature detection. In: SSYM'04, Berkeley
28. Kim J, Choi J, Chang D, Kwon T, Choi Y, Yuk E (2005) Traffic characteristics of a massively multi-player online role playing game. In: NetGames '05, New York
29. Lakhina A, Crovella M, Diot C (2005) Mining anomalies using traffic feature distributions. In: Proc ACM SIGCOMM, Philadelphia
30. Li Z, Sanghi M, Chen Y, Kao M, Chavez B (2006) Hamsa: fast signature generation for zero-day PolymorphicWorms with provable attack resilience. In: SP 2006, Washington, DC
31. Liang H, Tay I, Neo MF, Ooi WT, Motani M (2008) Avatar mobility in networked virtual environments: measurements, analysis, and implications. CoRR abs/0807.2328 (informal publication)
32. Linden Research (2003) Second life. http://secondlife.com/
33. Lua (1993–2008) Lua programming language. http://www.lua.org/
34. LucasArts (2003) Star wars galaXies. http://starwarsgalaxies.station.sony.com/
35. Ma J, Levchenko K, Kreibich C, Savage S, Voelker GM (2006) Unexpected means of protocol inference. In: IMC '06: proceedings of the 6th ACM SIGCOMM conference on internet measurement. ACM, New York, pp 313–326
36. Mallat S (1989) A theory for multiresolution signal decomposition: the wavelet representation. IEEE Pattern Anal Machine Intell 11(7):674–693
37. Mathworks (1983–2009) Matlab. http://www.mathworks.com/
38. Microsoft (1999–2009) MSN Messenger. http://join.msn.com/messenger/overview2000
39. Microsoft (2002) Age of mythology. http://www.microsoft.com/games/ageofmythology/
40. Moore AW, Papagiannaki K (2005) Toward the accurate identification of network applications. In: Proc PAM, Boston

41. NCsoft (2005) Guild Wars. http://www.guildwars.com/
42. Netlab (2005) Measuring online game application in GPRS and UMTS. http://www.netlab.hut.fi/opetus/s38310/04-05/Kalvot_04-05/H%E4m%E4l%E4i%nen_070605.ppt
43. Nullsoft (2000–2009) Gnutella. http://www.gnutella.com
44. Park B, Won YJ, Kim M, Hong JW (2008) Towards automated application signature generation for traffic identification. In: NOMS, pp 160–167
45. Svoboda P, Rupp M (2005) Online gaming models for wireless networks. In: Internet and multimedia systems and applications (IASTED)
46. Szabo G, Orincsay D, Gero BP, Gyori S, Borsos T (2007) Traffic analysis of mobile broadband networks. In: WICON '07: proceedings of the 3rd international conference on wireless internet. ICST (Institute for computer sciences, social-informatics and telecommunications engineering). ICST, Brussels, pp 1–5
47. Szabó G, Szabó I, Orincsay D (2007) Accurate traffic classification. In: Proc IEEE WOWMoM, Helsinki
48. Tan SA, Lau W, Loh A (2005) Networked game mobility model for first-person-shooter games. In: NetGames '05: proceedings of 4th ACM SIGCOMM workshop on network and system support for games. ACM, New York, pp 1–9
49. Yee N (2003–2009) The Daedalus Project.http://www.nickyee.com/daedalus/archives/001586.php
50. Veres A, Kenesi Z, Molnár S, Vattay G (2000) On the propagation of long-range dependence in the internet. In: ACM SIGCOMM, Stockholm
51. Willinger W, Taqqu M, Erramilli A (1996) A bibliographical guide to self-similar traffic and performance modeling for modern high-speed network
52. Willinger W, Taqqu MS, Sherman R, Wilson DV (1997) Self-similarity through high-variability: statistical analysis of ethernet LAN traffic at the source level. IEEE/ACM Trans Netw 5:71–86
53. Xu K, Zhang Z, Bhattacharyya S (2005) Profiling internet backbone traffic: behavior models and applications. In: Proc ACM SIGCOMM, Philadelphia
54. Zander S, Nguyen T, Armitage G (2005) Automated traffic classification and application identification using machine learning. In: Proc IEEE LCN, Sydney
55. Zuev D, Moore AW (2005) Traffic classification using a statistical approach. In: Proc PAM, Boston

Géza Szabó received the degree of Master in Computer Science in 2006 from the Budapest University of Technology and Economics, in Budapest, Hungary. His main interests include internet traffic classification and modeling. He works as a research engineer in Trafficlab of Ericsson Research Hungary and also pursues a PhD degree in the High Speed Networks Laboratory of the Budapest University of Technology and Economics.

András Veres is a senior researcher at Ericsson's Traffic Analysis and Network Performance Laboratory. His research interests include analysis of fixed and mobile networks, applications and protocols.

Sándor Molnár received his M.Sc. and Ph.D. in electrical engineering from the Budapest University of Technology and Economics (BME), Budapest, Hungary, in 1991 and 1996, respectively. In 1995 he joined the Department of Telecommunications and Media Informatics, BME. He is now an Associate Professor and the principal investigator of the teletraffic research program of the High Speed Networks Laboratory. Dr. Molnár has been participated in several European research projects COST 242, COST 257, COST 279 and recently in COST IC0703 on "Traffic Monitoring and Analysis: theory, techniques, tools and applications for the future networks". He is a member of the IFIP TC6 WG 6.3 on "Performance on Communication Systems". He is participating in the review process of several top journals and is serving in the Editorial Board of the Springer Telecommunication Systems journal. He is active as a guest editor of several international journals like the ACM/Kluwer Journal on Special Topics in Mobile Networks and Applications (MONET). Dr. Molnár served on numerous technical program committees of IEEE, ITC and IFIP conferences working also as Program Chair. He was the General Chair of SIMUTOOLS 2008. He is a member of the IEEE Communications Society. Dr. Molnár has more than 130 publications in international journals and conferences (see http://hsnlab.tmit.bme.hu/~molnar/ for recent publications). His main interests include teletraffic analysis and performance evaluation of modern communication networks.