A Secure and Efficient Micropayment Solution for Online Gaming

Sławomir Grzonkowski
Digital Enterprise Research Institute (DERI)
National University of Ireland Galway
Galway, Ireland
slawomir.grzonkowski@deri.org

Peter M. Corcoran
College of Engineering & Informatics,
National University of Ireland Galway
Galway, Ireland
peter.corcoran@nuigalway.ie

Abstract—**An innovative micropayment solution is proposed which incorporates both distributed security and social networking features. This has significant potential to impact on both the provision of new services and community features in online massive multiplayer online role-playing gaming (MMORPG) worlds. Initial feasibility implementation and testing of the micropayments system on both desktop and handheld device clients is presented. The system is well suited to gaming applications and our preliminary studies indicate the practical feasibility of integrating it within an operational MMORPG architecture.**

Keywords - micropayments; security; e-commenrce; authentication; MMORPGs, internet applications

I. Introduction

The recent popularity and growth in *massive multiplayer online role-playing games* (MMORPGs) has led to new online "universes" where players interact in a microcosm of the real world. Such games have led to the emergence of actual virtual economies [1] and led to spin-off industries and services [1]-[3]. One of the difficulties in such virtual economies is the barrier of offering payment for services which must be realized in the real human world and which is not generally encouraged or facilitated by the overlords of the virtual MMORPGs world [2].

Nevertheless we see a trend towards increasing development and evolution of commercial services within MMORPG worlds and this paper is directed towards offering a novel approach which not only supports micropayments but can be easily adapted to facilitate payment systems based on game waypoints or periodic elapsed-time payments in a non-intrusive manner. A further consideration, bearing in mind the scale of many MMORPGs, is that the authentication burden is placed on the client-side thus minimizing server load and the overall impact of such a system on the responsiveness of the gaming world.

Our proposed system has some additional advantages. As its origins lie in Semantic Web technologies it has been designed to be closely integrated with elements of the Dublin Core which should facilitate its integration into large-scale gaming systems. The system also employs a Zero-Knowlege-Proof (ZKP) approach which implies that the user credentials are never sent from the client to the server, thus the system is exceptionally robust and difficult to circumvent from a security perspective [26]-[29].

An extended discussion on the background and rationale to the present work is presented in section II where we discuss MMORPGs, the role of real-money-transactions (RMTs) in such virtual economies and an overview of a typical MMORPG gaming architecture. In section III we explain the basics of our Zero-Knowlege-Proof (ZKP) techniques; section IV focuses on the user authentication workflow and a detailed description of the challenge-response mechanisms we have implemented; section V discusses some of the security aspects of our system with an emphasis on the principle attacks and its potential weaknesses; in section VI we present a system evaluation and the results of some performance tests; in section VII we discuss the use and application of this technology in MMORPGs with an emphasis on ease-of-use and integration of the game system with payment mechanisms.

II. Background and Rationale

In this section we try to place our ideas within the broader framework of *authentication* and *e-commerce* systems. We begin by remarking that our system was originally designed for use in *social semantic networks* such as *friend-of-a-friend* (FOAF) as described in earlier publications [26]-[29]. Here we explore how the same techniques are equally applicable in the context of MMORPGs and how the same criteria which are relevant to *social semantic networks* imply the suitability of these techniques to large-scale gaming architectures.

With regard to our rationale to look at micropayment systems for gaming architectures it is probably sufficient to mention that real-money transactions in MMORPGs was of the order of 2B US dollars in 2007 [3] while on-line advertising revenues in the same year were of the order of 17B. Clearly this is an important and growing market sector in the Information Economy.

We begin with an overview of micropayment systems in in MMORPGs and a consideration of what the most important design criteria for such a system should be. This is followed by a consideration of a typical MMORPG architecture and how our micropayment system should be integrated into this. We also consider several similar authentication/micropayment systems that described in the literature and explain the disadvantages of these in the context of gaming platforms.

A. MMORPGs and Real-Money Transactions

Here we consider the growing role of real-money transactions (RMT) in MMORPGs and other online *virtual worlds*. Some authors [30], [31] wish to preserve the "magic

978-1-4244-4460-1/09/$25.00 ©2009 IEEE

circle" of these worlds. This is something akin to the suspension of disbelief when we watch a movie and they argue that its disruption by allowing RMTs disrupts the gaming experience. Other authors [30], [32], [33] consider the purchase of virtual assets as violating the achievement hierarcy of a game (i.e. cheating) and thus losing credibility and respect among their fellow players. Interestingly we begin to see the same social aspects emerge in MMORPG communities as are encountered more broadly in other Internet based communities.

On the other hand, Bartle [30] discusses three reasons why many MMORPG players do feel compelled to make virtual asset purchases: (i) experiencing all the content programmed into a world requires players to develop their characters to the highest level which requires time that not everyone has; (ii) some parts of the content may be so unappealing that even players with enough time would rather skip through them; (iii) MMORPGs are designed so that players must have avatars of approximately the same skill level in order to play together; thus if older players wish to spend time playing with younger ones (e.g. family members), then they must use RMTs to keep up with the avatar skills of younger family members.

The sale of virtual assets also divides opinions. In a survey conducted among Korean players, as many as 78% of the respondents felt that they had the right to own the items earned during play [34]. Nevertheless such real-world sale of items is generally prohibited by the commercial organizations which operate such virtual worlds and players who engage in significant levels of RMTs risk being barred from membership. For a more detailed analysis the interested reader is referred to [2]. In conclusion we can deduce that there is a significant and growing demand for the trading of *virtual assets* within virtual world communities, and despite active discouragement by the owners of such virtual worlds the value of RMTs was of the order of 2B US dollars in 2007 [3]. We argue there is a clear need for a secure, robust and distributed micropayment system for such *virtual worlds*.

B. Gaming Architectures and Micropayment Systems

We next provide an overview of a typical MMORPG gaming architecture. The core elements of this description are taken from the Java gaming platform [35] but most scalable architectures will be similar in overall design.

Most gaming architectures are very similar in design to classic 3-tier enterprise computing systems. They typically comprise: (i) a communications (or edge) layer which is responsible for connecting to local clients; (ii) a stateless game logic layer which is equivalent to the business logic layer of an enterprise computing system and (iii) a primary object store, equivalent to a central database in an enterprise system. There are, however some key differences because the needs of the gaming system differ from those of most enterprise systems.

Notably the system latencies are challenging and a system input to output of the order of 10s of ms is desirable. This implies a very tight integration between the communications servers and the game logic layer placing significant real-time computational burden on these system components. Much of the clever aspects of this architecture rely on the primary object store which looks like a cloud of persistent objects to the game logic layer. These objects are checked in and out as required by the logic layer from what is, in effect, a very fast and horizontally scalable transactional database. Object fetch and deserialization is of the order of tenths of a millisecond.

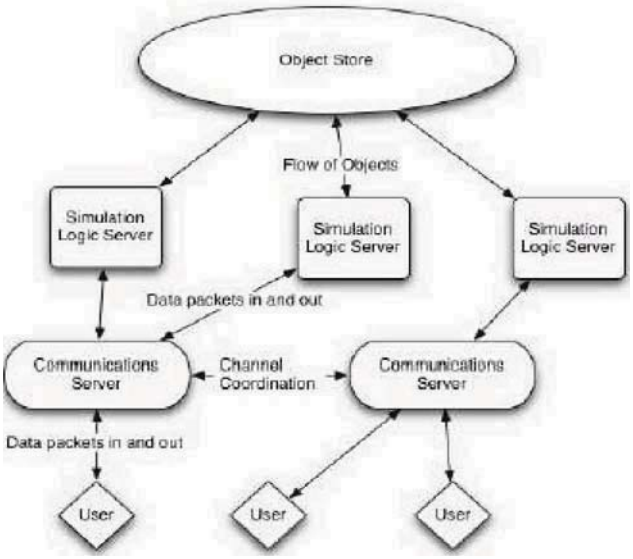

Figure 1: Gaming Architecture from ref [39]

The game logic layer itself is a stateless task processor. As user events come in from layer 1 they are converted into tasks in a set of task queues which are horizontally scaled across multiple logic servers. Because this layer is stateless if a local server dies the user is simply reconnected to an alternative logic server and can continue playing without apparent interruption. The primary object store provides deadlock detection by using timestamp ordering and communicating potential deadlocks back to layer 2 where newer tasks may be aborted and the relevant *game-level object* (GLO) is rolled back, the task surrenders its thread and transaction and is re-queued for later execution.

The lowest layer incorporates a *user manager* component at the edge of the architecture which is the only part of the system which knows that a client is connected. This component handles login and server discovery to connect the user into the main game logic at layer 2. We note that different user managers can be adopted to meet different connection strategies. We note that layer 2 just sends messages to user IDs and does not concern itself with how or where they are connected – this aspect is handled solely by the *user manager* component..Also worth noting is that the *user manager* provides a shortcut for direct client-to-client connections without a need to engage with the main gaming engine, yet providing some measure of regulation on client-to-client interactions. As we shall see shortly this aspect of a typical MMORPG architecture nicely supports our proposed micro-payments system.

C. *Related Authentication Techniques in the Literature*

A simple challenge-response solution is described by Garfinkel [20]. This idea is very similar to HTTP Digest access authentication that was proposed by Franks et. al. [4]. These solutions use, however, MD5 that is considered to be insecure. The server is supposed to store passwords in a plain-

text form or to create a hash. Even having only the hash, the server is able to impersonate the user.

Password-Authenticated Key Exchange (PAKE) is a family of protocols that affords a reasonable level of security using short memorized passwords for protecting information over insecure channels. Such protocols are also a subject of the IEEE P1363 standard working group. Encrypted Key Exchange (EKE) [21] that was introduced in 1992 combines both asymmetric and symmetric cryptography findings. The protocol has several versions and was followed by other propositions. Simplified Password- authenticated Exponential Key Exchange (SPEKE) protocol was developed by Jablon [22] for commercial purposes. The main difference in comparison with EKE is that the password is used to influence the selection of the generator parameter in the session-key generation function. Secure Remote Password (SRP) [23] was developed in 1997. Security of this protocol is dependent on the strength of the one-way hash function. The protocol was revised several times, and is currently at revision six. SRP is often applied to telnet and ftp. The protocol is more computationally intensive than EKE. It requires two modulo exponentiations, whereas EKE requires only one. Moreover, the protocol is vulnerable to offline dictionary attacks.

III. Zero Knowledge Proof Authentication

The zero-knowledge proof term has been formalized by Goldwasser, Micali, and Rackoff [5]. It describes challenge-response authentication protocols, in which parties are required to provide the correctness of their secrets, without revealing these secrets. Such protocols exist for any NP-set, provided that one-way functions exist.

During the authentication procedure, a user, for instance Alice, must respond to a number of challenges issued by the server. If the protocol is repeated t times, all t rounds must be answered successfully to prove Alice's identity. The server is always convinced with probability $1-2^{-t}$. In zero- knowledge-proof protocols, the verifier cannot learn anything from the authentication procedure. Moreover, the verifier is unable to cheat the prover because of having always only one value from two possible challenge responses; this is not sufficient to calculate the prover's secret. Furthermore, the verifier cannot cheat the prover because the protocol is repeated as long as the verifier is not convinced; due to random challenge selection, the verifier cannot pretend to be the prover to a third party. We next discuss various approaches to zero-knowledge protocols. Each of them uses different data structures, hence has unique properties that we briefly describe.

A. Conventional Authentication Strategies

The classical way of authenticating a user in a web application is to ask the user for a login and password. Then, the login and password are sent (see Figure 1) directly to the server; and the server responds to the request. If security is needed, an HTTPS protocol is used; the credentials, however, are still sent through the Internet and servers are able to read them, even if the password is stored in a hashed and salted form. For example, very often a user visits a web site whose certificate cannot be verified by the user's browser. However, the browser usually enables the user to accept the certificate although its verification was unsuccessful. Furthermore, to take the full potential of HTTPS, a user should buy a certificate issued by a well-known certificate authority and then the certificate should be installed on the user's browser. Installing a certificate to a web browser happens rarely and is prone to various attacks [9].

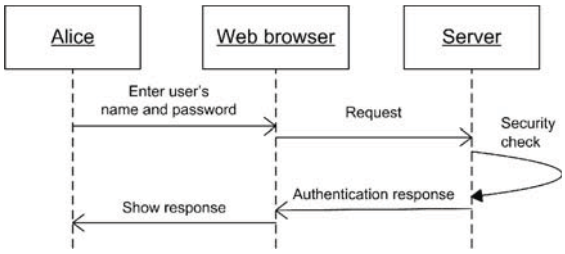

Figure 2: Conventional Authentication on Web

B. Eliptic Curve Cryptography

There are two main classical problems that enable zero-knowledge-proofs: discrete logarithm [6] and square-root [6]. These problems strongly depend on generating prime numbers. Moreover, the security of a key that is 1024-bits-long is comparable with the security of a 160-bit key in Elliptic Curve Cryptography (ECC) [7], [8]. Therefore, the classical approach is considered to be replaced with other solutions.

ECC is an efficient and attractive alternative to the classical public key cryptosystems. The main operation involved in ECC is point multiplication. The computations tend to be slow and inaccurate because of round-off error. In order to make the computations fast and accurate, they are performed using finite fields; this involves finding a multiplicative inverse of a number, which is the main problem for making an efficient implementation. To perform computations faster and avoid many multiplicative inverse operations, points can be represented in projective coordinates. It eliminates the need to compute many multiplicative inverses, but it requires more scalar multiplication than with affine coordinates.

C. Graph Isomorphisms

Assuming that V1, V2 are vertices and E1, E2 are edges. Two graphs G1 = (V1, E1) and G2 = (V2, E2) that have the same sets of vertices V1 =V2 ={1, 2, … n} are isomorphic, if there exists a permutation π on vertices {1, 2, … n} so that $(u, v) \in V1 \leftrightarrow (\pi(u), \pi(v)) \in V2$. An example is depicted on Figure 2. The problem is not likely to be NP-complete [24], but it is NP. There is no known polynomial-time algorithm that solves it. If we apply this problem to ZKP, a public key is composed of two isomorphic graphs G1 and G2, whereby the permutation πp is given as: $G2 = \pi p\ (G1)$

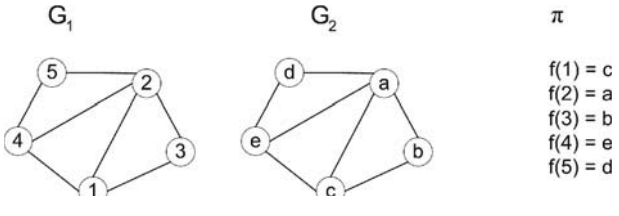

Figure 3: An example of graph isomorphism G2 = π(G1)

This work was supported (in part) by Science Foundation Ireland under Grant No. RSF0844 and Enterprise Ireland under Grant No. REI 1005.

is a private key. A prover generates a random permutation πR, and sends a graph GR = πR (G1) to the verifier. Then, depending on the verifier's challenge, the prover sends back πR or πR2 such that,

$$\pi R2 = \pi R \circ \pi p$$

Thus, the verifier is able to check one of the conditions:

$$GR = \pi R\ (G1) \text{ or } GR = \pi R2\ (G2)$$

Knowledge of only one parameter πR1 or πR2 does not let the verifier compute the prover's private keys.

D. Implications for Micropayment Systems

To authenticate a user, ZKP protocols require more than two steps: a user's request, a server's challenge, a user's response, and a server's response. Due to the ZKP nature, the server's challenges cannot be delivered to the user with the login form. Therefore, such protocols differ from the classical approaches, and require more flexible communication means. In addition, the authentication process is more computationally expensive and consumes more bandwidth. To satisfy these requirements, we decided to combine Ajax (Asynchronous JavaScript and XML) that is an asynchronous web technology with a graph isomorphism protocol. We chose this NP problem since it does not require user's browsers to find co-prime numbers nor multiplicative inverses.

Computations are performed using natural numbers and matrices; therefore, such arithmetic is easier to implement than, for example, elliptic curves. Our technology choice did not require plugins and was supported by default settings of the vast majority of web browsers. Hence, integration with existing applications was straightforward. Furthermore, their start-up time was negligible.

IV. System Design and Main Components

To authenticate a user, ZKP protocols require more than two steps: a user's request, a server's challenge, a user's response, and a server's response. To satisfy this requirement, we decided to combine Ajax (Asynchronous JavaScript and XML) that is an asynchronous web technology with a graph isomorphism protocol. We chose this NP problem since it does not require user's browsers to find co-prime numbers nor multiplicative inverses. Computations are performed using natural numbers and matrices; therefore, such arithmetic is easier to implement than, for example, elliptic curves.

A. Authentication Procedure

A sequence diagram presented on Figure 3 shows our approach in detail. Alice types her username and password, but the password never leaves her browser. In contrary to existing approaches like HTTP MD5 digest, the server does not have any information that would allow for impersonation Alice at another server. The browser uses the password to calculate her public-private key pairs, as described in the next section, and then executes the ZKP protocol.

The browser is responsible for a number of new tasks: calculating private keys from passwords, generating challenge graphs, and responding to the challenge. A server has only one more responsibility: generating random challenges. There is also one more interaction between a browser and a server in comparison with classical approaches Thus, the main question is if the new approach is feasible and will not require long waiting times for users.

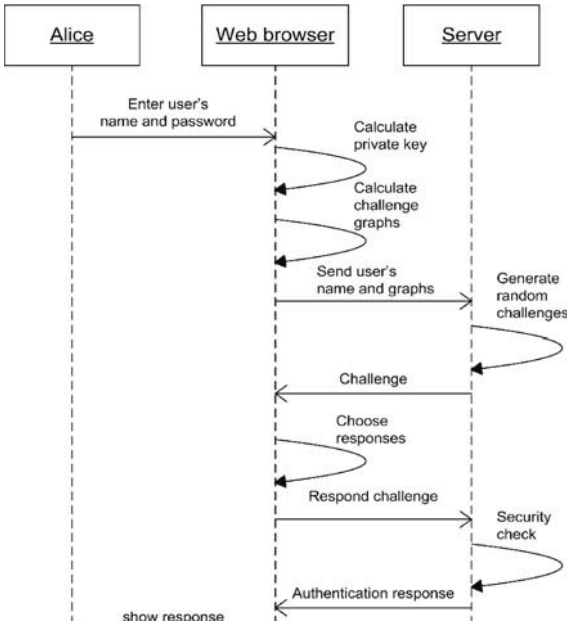

Figure 4: ZKP authentication workflow using a Web browser as client.

B. Private Key Algotithm

A user's private key is a permutation. Since we want to keep users using login and password pairs, we transform passwords to corresponding permutations using a one-way function. Such a transformation must always generate the same-size permutations.

1) Transformation

To compute a hash from a password we use Secure Hash Algorithm - Version 1.0 (SHA1). Such a hash is composed of numbers 0-9, a-f; and therefore, we can interpret the hash as a hexadecimal number. All the generated hashes have the same size, i.e. 160 bits and so their corresponding index numbers when sorted in lexicographical order in a table are from 38!+1 to 40!. If the hash alone were used as an index into a table of permutations stored in lexicographical order, different hashes would be mapped to permutations of varying lengths. Thus, we extend all the hashes with a hexadecimal character b at the beginning of each obtained string. This modification causes that the new values of hashes point to index positions between 41!+1 and 42! ;therefore, all the generated permutations have the same length. The key size can be changed, if another hash generation algorithm is used.

2) Calculating a permutation at a certain position

The web-application calculates the permutation indicated by the extended hash in two steps, the first being based on the observation that all the natural numbers can be represented as:

$$a0\ 0! + a1\ 1! + a2\ 2! + \ldots + ak\ k! \text{ for any natural number } k$$

Moreover, such a representation is unambiguous. There is a need to find the $a1\ 1! + a2\ 2! + \ldots + ak\ k!$ representation of

the extended hash. This can be done using Algorithm 1 (see Section IV), which converts a given decimal number to vector a[]. The GreatestFactorial() function returns the smallest factorial that is greater than the given number.

The second step of the conversion takes as input the table a[], and returns as output, a table permutation[] that contains the created permutation. It can be done using Algorithm 2. The transformation requires a temporary structure *s*, on which the following functions can be performed:

- insert(x) - adds a number x to the structure
- remove(x) - removes a number x from structure
- elementAt(j) - returns a number i that satisfies the condition: there are exactly j-1 smaller numbers than i in the structure
- full(n) – returns a structure with integers 0, 1, ... (n−1)

For example, with *s = 0, 1, 2, 3* by removing an element at second position we get *s = 0, 1, 3* and if we repeat the operation we have *s = 0, 1*. The structure s could be a simple table, however, the complexity of operating the table would be O(n). In the preferred embodiment an *AVL tree*, which is a special case of a *B-Tree*, is used for the structure s. In such structures, the computational complexity of removing elements is *O(log n)*.

```
Algorithm 1: Convert(number)
var i := 0
var factor := 0
var a[] := newArray()
while number > 0 do
        factor := GreatestFactorial(number)
        a[i] := number/factor
        number := number − a[i] * factor i = i + 1
end while
return a
```

```
Algorithm 2: 2ndConvert(a[])
var n := lengtha
var s := full(n) {returns a structure with integers 0,1,....(n-1)}
var a[] := newArray()
for i = 0 to n − 1 do
s.insert(i) {Initializing our temporary structure}
end for
for i = 0 to k − 1 do
        permutation[i] := s.elementAt(a[i])
            {getting an element at a certain position}
        s.remove(a[i])
            {removing the taken element}
end for
return permutation
```

3) Preservation of Authentication and Key States

After the authentication procedure is complete, there is no need for the web browser to store the entered user's credentials. The time necessary for the authentication procedure depends on the user's browser and server parameters, such as the amount of challenges. Our evaluation (see Section VI) shows that the private key is stored in memory for 1 second in case of 10 challenges.

During authentication, the server is not given any details that could be used to compromise the user's account. The result calculated by the server is a boolean value: *true* or *false*. In case of *true*, the server continues the session with the user assuming that the user has confirmed the claimed identity. If *false* is obtained, the authentication process is interrupted and the session with the user should be terminated.

V. System Robusness and Security Threats

The implementation of the algorithms can be done as a browser extension or as a script. In case of a script, for example JavaScript, two parties of the protocol are controlled by the server. Hence, if the server cannot be trusted, such implementation gives no benefits. If the implementation is done as a browser extension, then it is browser dependent.

In our considerations we also assume that the website is free from Cross-site scripting (XSS) vulnerabilities and that the communication between Alice and her browser is secure. Hence, the user's computer does not contain any viruses or malicious software. In addition, the human factor is involved: she can write down her credentials; during authentication procedure, somebody can see her password looking at the keyboard and then re-use it. Moreover, our prototype requires JavaScript. A user who turns it off will not be able to use ZKP approach. Another problem is phishing, we do not consider it in this paper since our solution can be easily combined with existing approaches, for example sign-in seal [11].

Our algorithm is also dependent on SHA-1, which we use in the private-key algorithm. It was supposed that 280 operations are needed to find a collision in SHA-1. Note that in August 2005 an attack [19] on the full version of SHA-1 was presented that lowers the required computational complexity to 263. Security of our application depends also on the NP problem and a dictionary attack feasibility:

A. Attacks on Graph Isomorphism

Efficient algorithms for some very specific graphs were proposed; Babai [13] discussed probabilistic algorithms; Corneil et al. [10] described algorithms for determining if two graphs are isomorphic. However, in our approach the problem is to find the permutation, not to determine if two given graphs are isomorphic. Thus, this work can only increase the speed of brute-force attacks. Our protocol uses square matrices of size 41; therefore there are 41! possibilities of such matrices. The generated graphs are completely random, and thus none of the proposed algorithms are useful to find the secret.

B. Dictionary Attacks

Florencio and Herley [14] performed research about password habits. They gathered a lot of statistical data and shown that most users use plenty of passwords every day, but the passwords are low quality and the users tend to re-use and forget them. If a user's public key is available or if the server is compromised, an attacker may check password candidates offline. Such attacks can be very effective [15], especially if users tend to choose low entropy passwords. Hence, guessing attacks were topics of many studies [16], [17]. If the communication between the server and Alice is encrypted, the only feasible way to attack the protocol is to run a trivial online dictionary attack and attempt authentication with password candidates. This attack can be, however, detected and slowed down by using captchas [18].

VI. System evaluation and performance testing

As a proof of the concept we implemented a prototype. Its server side was a Java™ servlet that was deployed on a Jakarta™ Tomcat™ server. The code was not platform specific to make it easily portable to other languages. The client side was implemented in Ajax, which requires users to trust the server.

A. Performance tests

An important challenge for our application was to satisfy the existing response-time standards. Such a standard was proposed in 1968 by Miller [25]. He defined three main time constraints: 0.1 second is for keeping the user attention attracted; 1 second is for keeping a user flow though uninterrupted; finally, 10 seconds for keeping user's attention on the dialog. Three decades after his findings, those rules are still applicable for web applications [12].

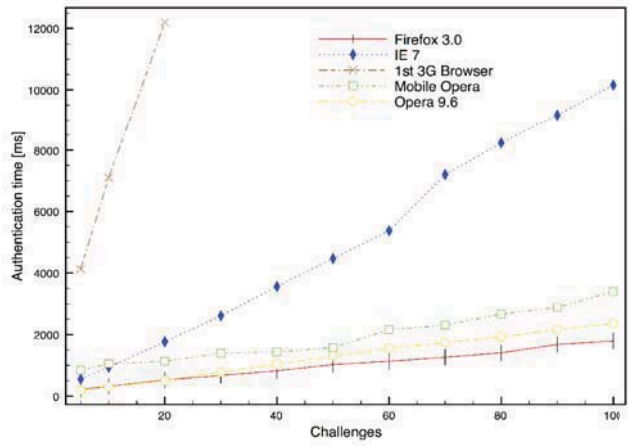

Figure 4: Authentication time for various browsers up to 2000 challenges

To indicate if our approach was feasible for web authentication, we evaluated our implementation in two ways: we measured authentication time for a given amount of challenges, and we counted the data that was exchanged to achieve certain security levels. Figure 4 shows results obtained for 5 configurations that we have tested:

- MacBook Pro™ 2.5 GHz, Intel Core 2 Duo, Firefox 3
- MacBook Pro™ 2.5 GHz, Intel Core 2 Duo, Opera 9.6
- Xeon(TM) CPU™ 2.8 GHz, 2 GB of Ram, Internet Explorer 7
- Nokia 5800™, Opera™ browser
- iPhone 3G™, Safari™ browser

For the Firefox™ 3 browser, 1 second, and therefore a user's impersonation risk level at $1/2^{50}$, was crossed for more than 50 challenges. Then, we also measured and compared our implementation in two other browsers:

1) Internet Explorer™ 7 and Opera™ 9.

Both browsers needed more than 1 seconds to achieve security risk level $1/2^{10}$. To process 100 challenges, Opera, which performed better, needed 3.4 seconds. When the computation lasted longer than 7 seconds, the user's thought flow was always interrupted by Internet Explorer which was displaying a message: 'A script on this page is causing Internet Explorer to run slowly. If it continues to run, your computer may become unresponsive. Do you want to abort the script? yes/no'. If the computation were not canceled, the browser returned correct results up to 1000 challenges. Therefore, the authentication process was transparent from the user's perspective up to 70 challenges. The additional calculations we performed on the obtained results showed that one challenge in Firefox 3 cost approximately 18 ms, 34 ms for Opera, and 101 ms for Internet Explorer.

When the computation lasted longer than 3.5 seconds, the user's thought flow was always interrupted by Internet Explorer™ which was displaying a message: 'A script on this page is causing Internet Explorer™ to run slowly. If it continues to run, your computer may become unresponsive. Do you want to abort the script? yes/no'. If the computation were not canceled, the browser returned correct results up to 1000 challenges. Therefore, the authentication process was transparent from the user's perspective up to 70 challenges. We also calculated the time cost of a single challenge for a web browser: in Firefox™ 3 a challenge cost approximately 32 ms, 84 ms for Opera™, and 82 ms for Internet Explorer™. Finally, we have also evaluated popular mobile browsers:

2) Mobile Safari™ and mobile Opera™.

The mobile Safari™ browser performed much worse than its competitors. The browser did not support the amount of challenges greater than 20. This is probably the lack of optimization for applications using java script so extensively. However, 20 challenges that were achieved by the mobile Safari™ browser are secure enough for an authentication procedure with high confidence level. We note that another mobile browser that we tested, Opera™, behaved correctly up to 800 challenges. In our tests mobile opera performs better than Internet Explorer™ 7 installed on a standalone computer.

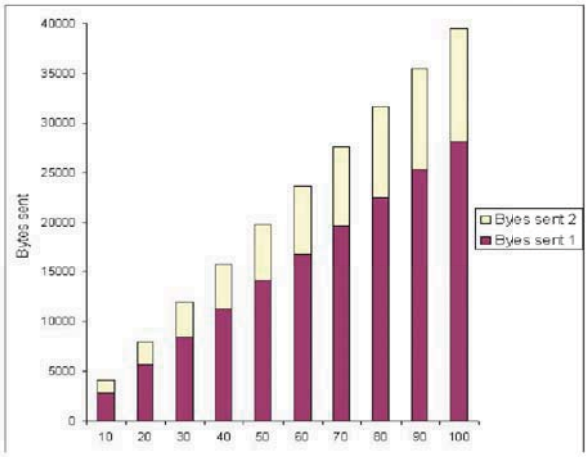

Figure 5: Bytes sent form a user's browser to a server during an authentication request and responding to the challenge

B. Data Measurements

Our other objective was to count data that is exchanged between a user's browser and a server. In the classical approaches, we cannot set the security level, and thus we always send a login and a password pairs. In zero-knowledge

proof approaches, we sent more data, the more security we need. We removed the HTTP frame size from our considerations since this parameter has a fixed length. Figure 5 shows that the total amount of data sent from a user's browser to a server is mainly determined by the first stage of the protocol performed on the user's browser side. The ratio between data sent in protocol step 4 and 8 is 71% to 29%. This parameter is neither browser nor hardware dependent and a single challenge costs approximately 394 bytes. Moreover, the data transmitted by our server in step 6 was equal to the amount of challenges, whereas in step 10 the server always sends 4 or 5 bytes depending on the authentication result.

To sum up our evaluation, we have shown that our approach can be performed satisfying existing response- time standards for keeping a user flow thought uninterrupted (1 second) and for keeping user's attention on the dialog (10 seconds). Excluding mobile Safari™, all of the tested browsers and configurations were able to execute 10 challenges below 1 second, and 100 challenges for less than 10 seconds. We also observed that the first communication step performed by the user's browser is very crucial since it is the most expensive both from the computational and bandwidth perspective. Furthermore, it determines the whole interaction from users' point of view. To perform the second communication step, a browser searches and selects previously created data according to server's choice. Finally, we note that in all ZKP protocols together with linear growth of amount of challenges (t), we have an exponential drop of cheating probability (2^{-t}).

VII. Conclusions

In section II.B we provided an overview of a typical MMORPG gaming architecture after a discussion of the scope, role and scale of real-money-transactions (RMTs) in such virtual gaming worlds. Our conclusions are that RMTs are an emerging industry which already generate a large amount of real-world econominc activity and which are likely to grow significantly over the next few decades. Thus one can see the underlying motivation to investigate practical, secure authentication systems such as our solution that takes advantage of a Zero-knowledge proof (ZKP) authentication approach based on isomorphic graphs. Such approach causes computational and bandwidth overhead in comparison to classical solutions. However, the results obtained by our prototype implementation demonstrate feasibility of this idea, including embodiments on mobile devices.

With respect to the gaming architecture described in section II.B, what is interesting is the role of the lowest layer in the gaming architecture hierarchy of servers, where a *user* manager component can offer means for direct user to user connections. This suggests that many of the techniques currently applied to social, semantic peer-to-peer networks can be equally applied and adopted within current gaming architectures. The potential to develop additional social interactions and for user-centric services derived from current research efforts in semantic Web technologies [26]-[29] is thus high.

Now by implementing our micropayment system at this lowest level we do not need to interfere with the core game logic or to impede real-time elements of the main game-play. Furthermore, because the computationally intensive elements of the authentication process are offloaded from the *user manager* server component onto the user client our system will have very little impact on system performance after the initial user log-in and creation of user keys to generate server-side challenges.

We also tested our client software on a number of mobile devices, extending previous work [26]-[29] and showing the practicality of our system even on hand-held smart-phone devices. This opens possibilities in mobile gaming as well as more conventional MMORPGs. We presented our solution that takes advantage of Zero-knowledge proof (ZKP) authentication approach based on isomorphic graphs.

Finallly, we remark that this work is simply and initial feasibility study and that a full integration of our micropayment system with an operating MMORPG gaming world would require significant additional development and testing. Nevertheless we are now convinced that this work is both feasible and practical and that the ensuing benefits would be well worth the required efforts.

References

[1] Vili Lehdonvirta: "Virtual Economics: Applying Economics to the Study of Game Worlds", *Proceedings of the 2005 Conference on Future Play (Future Play 2005)*, Michigan State University, Lansing, MI, USA, October 13-15, 2005.

[2] Vili Lehdonvirta: "Real-Money Trade of Virtual Assets: Ten Different User Perceptions", Proceedings of Digital Arts and Culture (DAC 2005), IT University of Copenhagen, Denmark, December 1-3, 2005, pp. 52-58.

[3] Vili Lehdonvirta: "Virtual Item Sales as a Revenue Model: Identifying Attributes that Drive Purchase Decisions", Electronic Commerce Research vol. 9, no. 1, pp. 97-113.

[4] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen and L. Stewart. HTTP Authentication: Basic and Digest Access Authentication, 1999.

[5] Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In STOC '85: Proceedings of the seventeenth annual ACM symposium on Theory of computing, p291–304, New York, NY, USA, 1985. ACM Press..

[6] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot. Handbook of Applied Cryptography. CRC Press, Inc., Boca Raton, FL, USA, 1996.

[7] V. S. Miller, "Use of elliptic curves in cryptography", In Lecture notes in computer sciences; 218 on Advances in cryptology—CRYPTO 85, pages 417–426, New York, NY, USA, 1986. Springer-Verlag New York, Inc.

[8] N. Koblitz, " Elliptic curve cryptosystems", Mathematics of Computation, Vol. 48, No. 177. (Jan., 1987), pp. 203-209

[9] H. Xia and J. C. Brustoloni, "Hardening web browsers against man-in-the-middle and eavesdropping attacks", In WWW '05: Proceedings of the 14th international conference on World Wide Web, pages 489–498, New York, NY, USA, 2005. ACM Press.

[10] D. G. Corneil and C. C. Gotlieb. An efficient algorithm for graph isomorphism. J. ACM, 17(1):51–64, 1970.

[11] Naveen Agarwal, Scott Renfro and Arturo Bejar. Current Anti-Phishing Solutions and Yahoo's Sign-in Seal. In W2SP: Workshop on Web 2.0 Security and Privacy 2007 held in conjunction with the 2007 IEEE Symposium on Security and Privacy, May 2007.

[12] Anna Bouch, Allan Kuchinsky and Nina Bhatti. Quality is in the eye of the beholder: meeting users,Äô requirements for Internet quality of service. In CHI'00: Proceedings of the SIGCHI conference on Human factors in computing systems, pages 297-304, New York, NY, USA, 2000. ACM Press.

[13] L. Babai, P. Erdos, and S. M. Selkow. Random graph isomorphism. SIAM Journal on Computing,

[14] D. Florencio and C. Herley. A large-scale study of web password habits. In WWW '07: Proceedings of the 16th international conference on World Wide Web, pages 657–666, New York, NY, USA, 2007. ACM Press.

[15] A. Narayanan and V. Shmatikov. Fast dictionary attacks on passwords using time-space tradeoff. In CCS '05: Proceedings of the 12th ACM conference on Computer and communications security, pages 364–372, New York, NY, USA, 2005. ACM.

[16] L. Gong, M. A. Lomas, R. M. Needham, and J. H. Saltzer. Protecting poorly chosen secrets from guessing attacks. IEEE Journal on Selected Areas in Communications, 11(5):648–656, 1993.

[17] M. Baudet. Deciding security of protocols against off-line guessing attacks. In CCS '05: Proceedings of the 12th ACM conference on Computer and communications security, pages 16–25, New York, NY, USA, 2005. ACM.

[18] B. Pinkas and T. Sander. Securing passwords against dictionary attacks. In CCS '02: Proceedings of the 9th ACM conference on Computer and communications security, pages 161–170, New York, NY, USA, 2002. ACM.

[19] http://www.iacr.org/conferences/crypto2005/rumpSchedule.html

[20] S. Garfinkel. "Fingerprinting your files" in MIT technology review technical report, August 2004.
(also at: http://beta.technologyreview.com/computing/13718/)

[21] S. M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks", in Research in Security and Privacy, 1992. Proceedings., 1992 IEEE Computer Society Symposium, Pages 72–84.

[22] D. P. Jablon. Strong password-only authenticated key exchange. SIGCOMM Comput. Commun. Rev., 26(5):5–26, 1996.

[23] T. Wu. The secure remote password protocol. In Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium, pages 97–111.

[24] U. Schoning. Graph Isomorphism is in the Low Hierarchy. In STACS'87: Proceedings of the 4th Annual Symposium on Theoretical Aspects of Computer Science, pages 114-124, London, UK, 1987. Springer-Verlag.

[25] R. B. Miller. Response Time in Man-Computer Conversational Transactions. In Proc. AFIPS Fall Joint Computer Conference Vol. 33, pages 267-277, San Francisco, Calif, 1968.

[26] S. Grzonkowski and B. McDaniel, "FRM: Towards a Semantic Platform for Fair Content Distribution. In AXMEDIS '08: Proceedings of the Fourth International Conference on Automated Production of Cross-Media Content for Multi-Channel Distribution. IEEE Computer Society, November 2008.

[27] S. Grzonkowski, A. Gzella, H. K. Krawczyk, S. R. Kruk, F. J. Martin, R. Moyano and T. Woroniecki., "D-FOAF - Security Aspects in Distributed User Management System", In Proceedings of IEEE International Conference of Technologies for Homeland Security and Safety. (TEHOSS 2005), 2005.

[28] S. Grzonkowski, W. Zaremba, M. Zaremba and B. McDaniel, "Extending Web Applications with a Lightweight Zero Knowledge Proof Authentication", Fifth IEEE International Conference on Soft Computing as Transdisciplinary Science and Technology (CSTST'08), October 2008.

[29] S. Grzonkowski, S. R. Kruk, A. Gzella, J. Demczuk and B. McDaniel. "Community-aware Ontologies", In S. R. Kruk and B. McDaniel, editors, "Semantic Digital Libraries", pages 125–139. Springer, 2008. ISBN: 978-3-540-85433-3.

[30] R. A. Bartle, "Pitfalls of Virtual Property", The Themis Group, 2004. <http://www.themis-group.com/uploads/PitfallsofVirtualProperty.pdf>

[31] Castronova, E. "The Right to Play", State of Play Conference Proceedings, New York Law School, 2004.

[32] Burke, T. Rubicite Breastplate Priced to Move, Cheap: How Virtual Economies Become Real Simulations (2002) . <http://www.swarthmore.edu/SocSci/tburke1/Rubicite Breastplate.pdf>

[33] T. L.Taylor, "Whose game is this anyway? Negotiating corporate ownership in a virtual world", In Frans Mäyrä (Ed.), Computer Games and Digital Cultures Conference Proceedings. (pp. 227-242). Tampere: Tampere University Press, 2002

[34] I. MacInnes, Y. J. Park,. and S. M. Whang, "Virtual World Governance: Digital Item Trade and its Consequences in Korea". TPRC Conference Paper, 2004.

[35] J. Kesselman, "Server Architectures for Massive Multiplayer Online Games", Sun Worldwide Developer Conference 2004.

Slawomir Grzonkowski is a researcher at DERI, National University of Ireland, Galway. He is the leading person of the DRM lab within the eLearning Cluster in Digital Enterprise Research Institute (DERI Galway), in which he is the creator and the main developer of SeDiCi and FRM. His research interests and scientific publications are related to semantic web, security, digital rights and social networks. His work on security and digital identity resulted in over 20 scientific articles at international conferences and workshops.

Peter Corcoran received the BAI (Electronic Engineering) and BA (Math's) degrees in 1984 and a Ph.D. for research work in the theory of Dielectric Liquids in 1987 from Trinity College Dublin . He is currently Vice-Dean of research in the College of Engineering & Informatics, National University of Ireland Galway. His research interests include embedded systems, home networking, digital imaging, security techniques and wireless networking technologies.