Using Multiplayer Games to Create Secure Communication

JAAK HENNO, OY Hypermeedia
HANNU JAAKKOLA, Tampere University
JUKKA MÄKELÄ, University of Lapland

Massively multiplayer online games (MMOGs) and social networks are very popular communication and entertainment formats, where millions of players from all around the world interact in a shared environment and exchange and trade different types of digital media: texts, videos, sounds and music. The communication systems implemented in these virtual environments are in increasing number encrypted to prevent fraud and user impersonation. Ubiquity of virtual forums with massive participation and participants communication systems has raised several questions about their security – gaming servers are suitable as potential exploitation tools for terrorist groups use to conduct on-line operations. The prevailing on Internet top-down security organization where trustworthiness of an object (computer, program) is established by a strong hierarchical system of security certificates does not work, since trusted high-level certificates can already be bought online in dark web marketplaces.

For many local communities (players of an online multiplayer game, local social networks etc.) is more advantageous a different, local 'sand-box' organization of a communication system where anonymous participants (initially identified only by their generated username without using even e-mails) actively interact using messages with strong encryption. For encryption they need entropy/randomness, but this they can create themselves in their interactions. In competitive interactions (e.g. they play a competitive game) all participants (trying to compete each other) behave differently, try to create thus the sequence of their actions is random and can be used as the secure key for symmetrical encryption for communication among participants.

Here is considered a class of games where expectation of payoff is the same for all moves, thus players cannot get from results any additional information about the game, thus their best strategy is to select their moves randomly (non-learnable games). It is shown, that in a sense all games of this class are similar, can be created with the same procedure and can be reduced to each other using introduced here operation of rectangular modification of the game state matrix. The best strategy for moves in this class of games is uniform randomness, thus (if they are competing and try to beat each other) in the play they create with their moves a random sequence. This sequence can be used for generating a key for symmetric encryption. In a non-local (server-based) multiplayer game players know only their own moves, about moves of other players they get only results of game (not the actual moves made by other players). Thus for generating a key server sends players sequence of all moves from which the player's own moves are removed. This sequence is different for players and contains only partial information, thus an eavesdropper (man-in-the-middle) cannot use it. Players insert into this holey sequence their own moves and get all the same sequence of moves which will be used as the common key for symmetrical encryption of communication; the procedure allows several enhancements for further randomness and/or speed of key generation. The key generation from player's moves removes need for use of public-key systems and all communication (and keys) remain inside the virtual community, whose security thus becomes self-sustainable.

1 INTRODUCTION

1.1 Communication and Communication Security

Our communication is increasingly encrypted – currently already more than 72 % of Internet communication is encrypted and the encryption is growing rapidly, with nearly 20 % increase in a year [Encrypted Traffic 2018].

Leading advisory company Gartner predicts that already in this year (2019) 80% of Internet traffic is encrypted [Cisco 2019]. Encryption is the main/only technology which enables communication privacy and data security, but for encryption is needed entropy/randomness.

Author's addresses: Jaak Henno, Hypermeedia OY, Paldiski mnt 157-8, 13518 Tallinn, Estonia, email: jaakhe@online.ee; Hannu Jaakkola, Tampere University, Pori Campus, P.O. Box 300, FI-28101 Pori, Finland, email: hannu.jaakkola@tuni.fi; Jukka Mäkelä, University of Lapland, 96101 Rovaniemi Finland, email: jukka.makela@ulapland.fi.

In: Z. Budimac and B. Koteska (eds.): Proceedings of the SQAMIA 2019: 8th Workshop on Software Quality, Analysis, Monitoring, Improvement, and Applications, Ohrid, North Macedonia, 22–25. September 2019. Also published online by CEUR Workshop Proceedings (http://ceur-ws.org, ISSN 1613-0073)

1.2 Encryption

The encrypted communication proceeds usually in two steps. In the first step, interested parties use public-key (asymmetric) encryption scheme for creating a common for both parties key for a symmetric encryption. The reason for such a two-step technology is in different properties of non-symmetric and symmetric encryption: the non-symmetric encryption system allows to send the first message using a publicly available long (currently a typical value is 2048 bits) key; the message can be decrypted only with the receiver's private key (much shorter, e.g. only 256 bits). Using a publicly available key is convenient, but the method is slow, used only for creating a 'shared secret' – the key (random string of 128/256 bits) for symmetric encryption, which is later used in the (quick) symmetric encryption (the key is secret, created in the first open-key round) of the main communication.

Using a publicly available resource (the public encryption key) is problematic - to whom does this key actually create an understandable (decipherable) message, who is the 'real' owner of this key?

1.3 The 'top-down security' organization - certificates

To authenticate keys owners, for creating new keys and managing their use has been created a hierarchical system of security certificates. These certificates are like passports which authenticate the identity of the certificate's holder and grant permissions to use encrypted communication using a public key infrastructure. The certificates are issued and managed by a strong hierarchical system of security principals using multistep validation processes – every passport/certificate should have only one valid fixed owner.

But all public secrets have shot lifetime (computers could crack any secret) and this 'top-down' security model leaks – many top-level certificates can be acquired from dark web marketplaces [Maimon at al 2019], authentication keys have been used to authenticate malware [SecurityWeek 2019] and/or to encrypt malware [Hacker News 2017], [Kaspersky 2018].

1.4 Growing need for entropy

Introduction of new Internet-connected devices - Internet of Things (IoT), virtual/cloud servers, Internet-connected mobile devices (cars, scooters etc.) increase the need for randomness used for encryption. There are already proposals for special services to serve entropy, i.e. random data [NIST 2016]. In order to deliver provided entropy to users is proposed a special new 'Entropy as a Service' protocol [EaaSP 2018]. But for delivery this entropy also should be encrypted, thus it is not clear, whether this service will reduce the need for entropy or increase.

The current hierarchical, top-down security scheme introduces many unnecessary checks which slow down encrypted traffic. If I want to send an encrypted message to my co-worker sitting in the next room then with the current hierarchical top-down security system would be evoked many upper-level security authorities (up to the Heavenly God Microsoft – we are using Outlook). Specialists estimate, that the share of network traffic that really should be checked by security measures is still relatively small [Effect of Encryption 2017] and local traffic should be secured locally – local communication participants (the communication's local context) know best what are the possible dangers and thus also can invent best measures to eliminate them. This is also the main idea of the behavioural security measures which popularity is rapidly growing [Terada 2017], [Leventhal 2018].

1.5 BYOK – Bring Your Own Keys

The growing in popularity trend to move security concerns down, closer to communication participants is also the 'Bring Your Own Keys', where the user manages the encryption keys for their data. This is offered (or proposed) by many global data/communication services: Google Compute Engine, Amazon, Adobe Creative Cloud, Microsoft's Key Vault and the number is growing.

One of areas where local communication could be secured with local measures and locally generated encryption keys is communication in (limited) on-line communities – chat/exchange for players of online multiplayer games and (local) on-line social forums.

2 GAME AND GAME COMMUNICATION (CHAT, EXCHANGE/TRADE)

2.1 Economy of Video Games

Video games have become a significant part of the global entertainment economy and their significance is constantly growing.

The first video games were products – you paid and got a game, either stored on disk (CD, DVD) or right to download/play it from some (streaming) server. But with the global marketing trend „tie your customer forever“ video games are increasingly creating their own economic values and trading space for these values. A game could be rather cheap, but then appear the „extreme edition“, „additional downloads“, new items could be bought in-game etc. You can enrich your game with DLC (DownLoadable Content) from a virtual marketplace and add to your game various digital content: songs, skins, characters, modes, levels, weapons, cars, expansions, etc. Value of DLC-s already exceed value of games, with DLC video game industry has become a multi-billion dollar-a-year business [WePC 2019]. Such an on-line market needs good security.

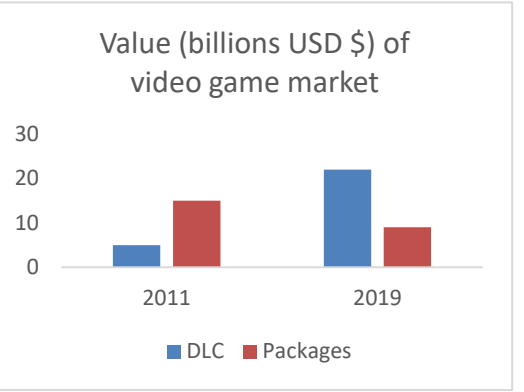

Fig. 1. Emergence of new 'game economy' with in-game trading, DLC and communication.

Multiplayer Online Games (MOG) introduce a virtual in-game economy where players can find or earn virtual objects – coins, arms, skins for weapons etc. Games supports in-game trading of these virtual items and even have their own virtual currency that can be used in trading within the game ("Ultima Online", "Second Life", "League of Legends" etc.), but many players want to trade game items also outside the game. This economy is driven by virtual supply and demand, involves the exchange of real world money and is usually not regulated by game developers. However, many game creators support 'outside-game' economy, e.g. player of MMOG 'Counter-Strike: Global Offensive' (created by Valve) get sometimes after finishing the game a crate which could contain several skins; some (e.g. gold tier skins) are extremely rare, thus many gamers want to get one, but to open the crate gamer should buy from Valve (for real 2.49 USD) a key. In last year this game had more than 4 million players [SteamCharts 2019], thus many potential byers and this 'in-game' purchases (for real money) strategy is increasingly used. Demand for these virtual objects has created alternative economy for enterprising players who have enough time to play these games to "farm" the game by earning in-game currency and rare items. For farming, level up player characters and other repetitive, no-skill based and time-intensive tasks have appeared many bots, which perform these tasks while player sleeps [Hackerbot 2019], [MPGH 2019].

These virtual goods are then converted to real money using Real-Money Trading (RMT) - sold to players using chat rooms or dedicated forums or on online auction sites, e.g. eBay; profits for this business are growing in thousands of dollars [TrendMicro 2016].

Besides virtual values are for cybercriminals attractive also player's 'real-world' values - working e-mail, country/language, mobile phone's number. Although many on-line communities (Fortnite, Google Play, EA) have introduced double verification of personality, statistics shows that theft of player's personal information is constantly growing. Thus it is better not to use player's personal information and make the game world and game communication closed. Game designers should not hope for profits from selling players information, but make their games 'inner world' with DSL better

When participants of online multiuser communities (multiplayer games, social networks) want to establish a direct communication with fellow participants (a chat system), which allows to exchange text messages and game's virtual values (media files- text, images, video), then this new communication system should be closed and not to burden game servers which are already busy with the game communication. To ensure security of game (game players and game server) the communication system should be 'sand-boxed'.

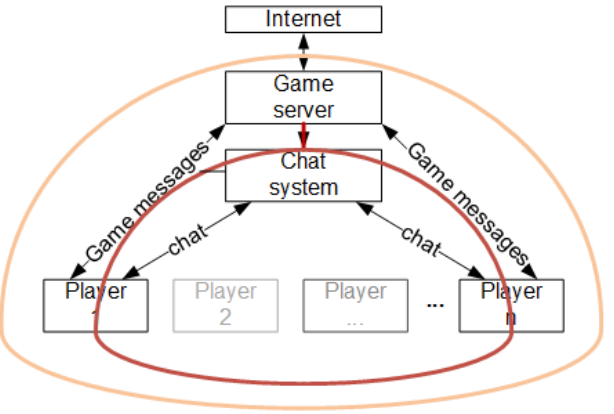

Fig. 2. Security surfaces of information flows in a multiplayer game with communication (chat system).

The game communication system is also a marketplace for exchange of game's virtual values, thus all the communication here should be encrypted. In order to close the game communication (chat) system from outside it should generate the encryption keys itself (i.e. this is another example of BYOK). For this it could use the entropy/randomness, created by player's themselves and not to use any outside sources of entropy/randomness.

3 ENTROPY FROM GAMES

For players of on-line multiplayer games keys for symmetric encryption can be generated as a part of gameplay, thus eliminating the need for higher-level security authorities and by-passing the use of public-key encryption step in key generation.

"Randomness has been part of games since their earliest inception -- and when I say 'earliest inception,' I mean deep into the unwritten Neolithic past" stated game design/author /classic Greg Costikyan [Costikyan 2009], [A. Chalk 2009].

Randomness in games appears from two sources – players decisions (actions/moves) and game setup, e.g. in a shooting game – movement/appearance of targets, precision of calculating hits (collisions of player's bullets with targets) etc. The properties of the second source (randomness caused by game setup) is nearly impossible to measure exactly, game designers always want to create random outcomes within

parameters that can be influenced to fit the game design. Thus also the randomness appearing in game mechanics is difficult (in a good game - nearly impossible) to evaluate and estimate its dependence on game algorithms (e.g. showing/moving/hiding targets) – this is just a noise injected between a player's choice and the result.
Therefore in the following is presented a setup, where this second source is eliminated - game is a simple deterministic algorithm and the randomness is created only by player's actions.

Randomness created by humans is an arguable topic – many game designers consider humans very predictable [Costikyan 2013]. Statement "Multiple competing players create randomness" is not a provable mathematical statement. This statement is similar to Gödel's incompleteness results – it can't be proved, but since nobody has presented a counter-example, we believe it. Humans are playing (video) games more and more; if the process of playing does not introduce randomness (surprise, entropy), if this were just a process of repeating game's moves over and over, playing were not fun and the number of players would not increase. But more than one third of human population is already playing video games [Video Gaming 2019], thus this should be true – players create randomness in the process of gameplay, this creates the fun of playing.

3.1 Non-local simultaneous games

A game is non-local, if the only information what players get about actions of other players are the results of those actions and they do not get information about actions/moves of other players what caused these results. This is the typical situation in server-based on-line video games. The situation is more complicated with e.g. direct Peer-To-Peer (P2P) communication [House 2018] which is here not considered.
A (simultaneous) game is a 5-tuple

$$G=< P, M, A, R, \bar{\rho} >$$

Here

P is the set of players, $|P| = n,\ n \geq 2$ is the number of players; e.g. for two-player game $P = \{p^1, p^2\}$;

$M = \{0,1,...,m\text{-}1\},\ m \geq 2$ is the set of player's moves (legal actions, the same for all players), here they are labelled as m-ary natural numbers; all players move simultaneously, i.e. select an element from the set M; denote by $m_t^i \in M$ the move of player $p^i \in P$ in gameplay at moment (move) $t,\ t = 0,1,...,l$, (l - the length of gameplay); $\bar{m}_t = (m_t^1, m_t^2, ..., m_t^n) \in M^n$ is the (global) move - vector of all (simultaneous) moves made by all players in move t;

A is a deterministic finite automaton, which on players move $\bar{m}_t = (m_t^1, m_t^2, ..., m_t^n)$ decides the rewards (payoff) $\bar{r}_t = (r_t^1, r_t^2, ..., r_t^n) \in R^n$ for all players after the move and changes its state (i.e. players rewards can/will change), thus automaton A implements n-ary mapping

$$\bar{\rho} : M^n \to R^n$$

The mapping $\bar{\rho}$ could be different for different states of automaton A (during gameplay);

R is the finite set of rewards. In 'real' video games rewards can be of several different types, numeric and/or symbolic (badges, ammo, health etc., see e.g. [Phillips 2018] for typology); they represent the utility, thus should be comparable; here rewards R are shown as integers. In 'real' video games number of available actions (thus also rewards) may change in different states of the game (players can

get e.g. new weapons or skills/spells), in the following number m of available actions and rewards remains constant.

Players make moves (create vectors $\bar{m}_t = (m_t^1, m_t^2, ..., m_t^n)$) simultaneously and after a move get information only about their own result, i.e. player $p^i \in \mathrm{P}$ gets the value r_t^i, but not information about moves what were made by other players, i.e. they do not get moves $m_t^j \in \mathrm{M}$, $j \neq i$.

In order to extract randomness from players moves the mappings $\bar{\rho}$ should be resilient to game-bots – programs which play the game on behalf of human players or help player to advance in game. They try to extract information from players' moves and predict play; producers of on-line games are developing methods to discover and disable them [Ah Reum Kang et al 2013].

Development of game-bots was started by dishonest players, but currently their development has become a direction of study in machine learning [Luzgin, R. 2018].

However, not every game can be 'cracked' by machine learning. Games where expectation of payoff is the same for all actions of player are non-learnable even for Tensorflow [Shai Ben-David et all 2019].

Game payoffs (in current state) can be described by the state's payoff mapping, which for players p^1, p^2 is as in the economy-based game theory a matrix, where in rows are moves and corresponding rewards for player p^1, in columns – moves and rewards for player p^2; rewards for move m^i by player p^1 and move m^j by player p^2 are r_{ij}^1, r_{ij}^2 :

Table. 1. Payoffs table for a two-person game (in some state)

Player p^1 \ Player p^2	0	1	...	m-1	
0	$r_{00}^1,\ r_{00}^2$	$r_{01}^1,\ r_{01}^2$		$r_{0m-1}^1,\ r_{0m-2}^2$	$\sum_i r_{0i}^1, \sum_i r_{0i}^2$
1	$r_{10}^1,\ r_{10}^2$	$r_{10}^1,\ r_{10}^2$		$r_{1m-1}^1,\ r_{1m-1}^2$	$\sum_i r_{1i}^1, \sum_i r_{1i}^2$
...					
m-1	$r_{m-10}^1,\ r_{m-10}^2$	$r_{m-11}^1,\ r_{m-11}^2$		$r_{m-1m-1}^1,\ r_{m-1m-1}^2$	$\sum_i r_{m-1i}^1, \sum_i r_{m-1i}^2$
	$\sum_i r_{i0}^1, \sum_i r_{i0}^2$	$\sum_i r_{i1}^1, \sum_i r_{i1}^2$		$\sum_i r_{im.1}^1, \sum_i r_{im-1}^2$	

Game is non-learnable, if all sums for all rows and columns are equal:

$$\sum_i r_{i0}^1 = \sum_i r_{i1}^1 = ... = \sum_i r_{im-1}^1 = \sum_i r_{0i}^1 = \sum_i r_{1i}^1 = ... = \sum_i r_{m-1i}^1 =$$
$$\sum_i r_{i0}^2 = \sum_i r_{i1}^2 = ... = \sum_i r_{im-1}^2 = \sum_i r_{0i}^2 = \sum_i r_{1i}^2 = ... = \sum_i r_{m-1i}^2 = \mathrm{V} \quad (1)$$

The value V/m is the (average) expected payoff of a move.

Games satisfying (1) have been considered in several earlier studies, e.g. in [Shapley 1963], [Plan 2017], [Shih-Fen CHENG et al 2004], but mostly only for their symmetry properties and not for the aspect considered here – extracting randomness from gameplay.

In the (economic) game theory publications are in the game payoff (for the current state) matrices usually shown only payoffs and not labels for player's moves/strategies (every player can relabel them). Usually only values for r_{ij}^1 are shown (especially if the game is zero-sum, i.e. $r_{ij}^1 = -r_{ij}^2$), as e.g. a game from [Shapley 1963]:

Table. 2. Payoff table for a non-symmetric non-learnable game

0	1	2	-1
-1	0	1	2
2	-1	0	1
1	2	-1	0

(2)

3.2 Mappings of games

A game is non-learnable if the (expected) value of all moves for all players is the same, i.e. (1) holds in every state of the game. Such a non-learnable games have many properties in common and already in [Shapley 1963] was shown, that the payoff matrices for such a game could not be symmetric, but could be made skew-symmetric with re-arrangement (re-labelling) of player's moves using blocks of size power of two. Here it is shown, that the matrix of such games could be transformed to totally non-symmetric form.

In the following is considered the class of all such games (for the fixed number m of player's actions) and is shown, that all these games can be constructed from one vector of payoffs, using transformations of blocks of 2×2 elements, thus all these games belong to a class closed for mappings of zero-sum vectors and 2×2 rearrangements.

In the 'real' (i.e. economy-based) theory of games mappings (isomorphism, endomorphism) are considered only for games with the same number of players. For video games this is nonsense – in a (server-based) multiplayer game new players could appear at any moment and just the same way, already playing players could drop the game or just miss a number of moves. Nobody considers that (dis)appearing players introduce a new game, the game remains the same, thus we need a definition of games same-ness (similarity).

A game $G=<\mathrm{P},\mathrm{M},\mathrm{A},\mathrm{R},\overline{\rho}>$ is simpler (a homomorphic image) of a game $G_1=<\mathrm{P}_1,\mathrm{M}_1,\mathrm{A}_1,\mathrm{R}_1,\overline{\rho}_1>$ iff:

1. $|\mathrm{P}|\leq|\mathrm{P}_1|$ - it is defined for a smaller set of players (but still $|\mathrm{P}|\geq 2$ - otherwise this is not a multiplayer game), let $\iota:\mathrm{P}\leftarrow\mathrm{P}_1$ be an (arbitrary) mapping (function, renaming) of players;
2. $|\mathrm{M}|\leq|\mathrm{M}_1|$ - players have less actions (but sill $|\mathrm{M}|\geq 2$ - otherwise this is not a game) ;
3. $|\mathrm{R}|\leq|\mathrm{R}_1|$ - the number of (different) rewards is smaller (but still $|\mathrm{R}|\geq 2$ - otherwise this is not a game)
4. there exist mappings (with component wise application to vectors)

$$\mu:\mathrm{M}\leftarrow\mathrm{M}_1$$
$$\nu:\mathrm{R}\leftarrow\mathrm{R}_1$$

such that

$$\nu(\overline{\rho}(m^1,m^2,...,m^n))=\overline{\rho}_1(\mu(m^1),\mu(m^2),...,\mu(m^n))$$

A game $G=<\mathrm{P},\mathrm{M},\mathrm{A},\mathrm{R},\overline{\rho}>$ is similar (nearly isomorphic) to a game $G_1=<\mathrm{P}_1,\mathrm{M}_1,\mathrm{A}_1,\mathrm{R}_1,\overline{\rho}_1>$, if in the above definitions ι,μ are 1-1 (i.e. renaming). The mapping ν could not be 1-1, but for a games for humans who want to see differences in their actions it should preserve entropy, i.e. satisfy

$$(\forall r_1,r_2,r_3\in\mathrm{R}_1)((r_1\neq r_2\neq r_3\neq r_1)\rightarrow(\nu(r)_1\neq\nu(r_2)\neq\nu(r_3)\neq\nu(r_1))$$

Under this definition instances of a multiplayer game with different number of players are similar even with different dispersion of expectations of player awards.

3.3 Symmetric group-like games

Here are considered games, where randomness is generated by player's decisions and not by game decision mechanism and structure. This randomness is similar to biological physically unclonable functions [Wali et al 2019] – it is the result of built-in hardware, first of all limits of player's brains.
Information is not created *en masse*, but in interactions of individuals. Therefore the game decision function

$$\bar{\rho} : (m_t^1, m_t^2, ..., m_t^n) \to (r_t^1, r_t^2, ..., r_t^n)$$

decides rewards $r_t^1, r_t^2, ..., r_t^n \in \mathrm{R}$ after considering all interactions between all players, i.e. it uses a sub-function ρ to decide local interactions

$$\rho : (m^i, m^j) \to (\tilde{r}^i, \tilde{r}^j)$$

which uses actions (m^i, m^j) of players $p^i, p^j \in \mathrm{P}$ and decides their (local) rewards $\tilde{r}^i, \tilde{r}^j$; the mapping ρ is deterministic, thus instead of mapping more suitable is the functional notation $\rho(m^i, m^j) = (\tilde{r}^i, \tilde{r}^j)$
The global rewards function $\bar{\rho}$ is cumulative, i.e.

$$r_t^i = \sum_{1 \le i \le n} \rho(m_t^i, m_t^j)$$

In games of chance nobody wants to have worst chances by design of the game and all actions of players should be significant, i.e. could change the result (have maximal entropy – players want to have maximal fun!), thus the local decision function ρ should satisfy the following conditions.

1. For any actions $m^i, m^j \in \mathrm{M}$ if $\rho(m^i, m^j) = (\tilde{r}^i, \tilde{r}^j)$, then $\tilde{r}^i + \tilde{r}^j = 0$ - the game is zero-sum
2. For any actions $m^i, m^j \in \mathrm{M}$ if $\rho(m^i, m^j) = (\tilde{r}^i, \tilde{r}^j)$, then $\rho(m^j, m^i) = (\tilde{r}^j, \tilde{r}^i)$ - the game is symmetric (in some publications is used different symmetry $\rho(m^i, m^j) = \rho(m^j, m^i)$ - payoffs do not depend on who made a move, such a game is e.g. 'Zero-One' considered in [26])
3. For any actions $m^i, m^j \in \mathrm{M}$, if $\rho(m^i, m^j) = (\tilde{r}^i, \tilde{r}^j)$, then there exists an action $\tilde{m}^i \in \mathrm{M}$ such that $\rho(\tilde{m}^i, m^j) = (\tilde{r}^j, \tilde{r}^i)$, i.e. game is group-like – any player could reverse the result of local interaction (if player had information about opponents move m^j - but players make moves simultaneously and even after the move they have information only about result, i.e. reward – not about the move what caused it)
4. For any actions $m^{i1}, m^{i2} \in \mathrm{M}$ there exists an action $m^j \in \mathrm{M}$ such that $\rho(m^{i1}, m^j) \neq \rho(m^{i2}, m^j)$, i.e. all actions are efficient and one player can always change the reward of the second player.
5. For any actions $m^{i1}, m^{i2} \in \mathrm{M}$ there exists 1-1 cyclically monotone mapping (see e.g. [CalTech 2004]) between sequences of rewards $\rho(m^{i1}, 0), \rho(m^{i1}, 1), ..., \rho(m^{i1}, m-1)$ and $\rho(m^{i2}, 0), \rho(m^{i2}, 1), ..., \rho(m^{i2}, m-1)$; from the property 2. it follows, that 1-1 mapping exists also between the above sequences and the sequences $\rho(0, m^j), \rho(1, m^j), ..., \rho(m-1, m^j)$ for any $m^j \in \mathrm{M}$.

Let NL be the class of all games with properties 1-5.
From the properties 1-4 of the local decision function ρ follow several properties of this function.

(1) From 1. it follows, that $\rho(m^i, m^j) = (\tilde{r}^i, \tilde{r}^j)$ could be presented as $\rho(m^i, m^j) = (\tilde{r}^i, -\tilde{r}^i) = (-\tilde{r}^j, \tilde{r}^j)$. The expressions, e.g. $\rho(m^i, m^j) = (\tilde{r}^i, -\tilde{r}^i)$ could create impression, that ρ depends only on one argument (here – on the first, in the second expression $\rho(m^i, m^j) = (-\tilde{r}^j, \tilde{r}^j)$ - on the second) thus in the following we will sometimes use notations $\rho(m^i, m^j) = (\tilde{r}^i, -\tilde{r}^i) = \tilde{r}^{ij}$, $\rho(m^i, m^j) = (-\tilde{r}^j, \tilde{r}^j) = -\tilde{r}^{ji}$

(2) $\rho(m, m) = (0, 0)$

Proof. By 2. for any $m^i, m^j \in \mathrm{M}$, $\rho(m^i, m^j) = \tilde{r}^{ij} = -\tilde{r}^{ji}$, thus if $m^i = m^j$ we have $\tilde{r}^{ii} = -\tilde{r}^{ii}$, thus $r^{ii} = 0$

(3) The elements of the set R can be renamed so that R has cyclic structure.

Denote $k = (|\mathrm{M}| - 1)/2$ if $|\mathrm{M}|$ is odd and $k = |\mathrm{M}|/2 - 1$ if $|\mathrm{M}|$ is even and let $\bar{\boldsymbol{r}} = \{0, r_1, ..., r_{k-1}, r_k\}$ be a (strictly) monotony increasing sequence, i.e. from $i < j$, $0 \le i < j \le k$ follows $r_i < r_j$. Set

$\bar{\mathrm{R}} = \{0, r_1, ..., r_{k-1}, r_k, -r_k, -r_{k-1}, ... - r_1\}$ for k -odd

$\bar{\mathrm{R}} = \{0, r_1, ..., r_{k-1}, r_k, 0, -r_k, -r_{k-1}, ... - r_1\}$ for k – even

3.4 Constructing a non-learnable game from a vector of rewards

In both cases above the sequence $\bar{\mathrm{R}}$ has *m* elements, thus to simplify presentation denote $\tilde{\mathrm{R}} = \{\tilde{r}_0, \tilde{r}_1, ..., \tilde{r}_{m-1}\}$ and let $\pi : \tilde{\mathrm{R}} \to \tilde{\mathrm{R}}$ be any substitution of the sequence $\bar{\mathrm{R}}$ without sub-cycles, i.e. for any $\tilde{r}_i, \tilde{r}_j \in \tilde{\mathrm{R}}$ $\exists t < \mathrm{m}$ such that $\pi^t(\tilde{r}_i) = \tilde{r}_j$; then for any $\tilde{r} \in \tilde{\mathrm{R}}$ elements $\tilde{r}_0$, $\pi(\tilde{r}_0), \pi^2(\tilde{r}_0), ..., \pi^{m-1}(\tilde{r}_0)$ .cover the whole set $\bar{\mathrm{R}}$.

A game matrix for selected vector $\tilde{\mathrm{R}}$ can be the following:

Table. 3. Payoff table of a game from a vector $\tilde{\mathrm{R}}$ of payoffs

$\tilde{r}_0$	$\tilde{r}_1$	...	$\tilde{r}_{m-1}$
$\pi(\tilde{r}_0)$	$\pi(\tilde{r}_1)$	...	$\pi(\tilde{r}_{m-1})$
$\pi^2(\tilde{r}_0)$	$\pi^2(\tilde{r}_1)$	...	$\pi^2(\tilde{r}_{m-1})$
...	...	...	
$\pi^{m-1}(\tilde{r}_0)$	$\pi^{m-1}(\tilde{r}_1)$	...	$\pi^{m-1}(\tilde{r}_{m-1})$

(3)

The conditions (1), (2) hold – in all rows and columns are just the elements of the set $\bar{\mathrm{R}}$ (but in different order), i.e. this is a matrix for a non-learnable game constructed from the sequence $\tilde{\mathrm{R}}$.

3.5 Modifications of games from NL

The class NL of non-learnable games allows modifications of their matrixes preserving condition (1):

(a) Multiplication of all elements with a constant (even with 0 – the property (1) will hold, but the game will not have any entropy – nobody would like to play);

(b) Adding a constant to all elements the – in 'real' video games players rewards are thousands or even bigger – this seems to make the game more fun;

(c) Manipulating the corner values of any rectangular block: if $x_{i,j}, x_{i,k}, x_{l,j}, x_{l,k}$ are corner elements of a rectangle in a game matrix, then they could be replaced with $x_{i,j}+c, x_{i,k}-c, x_{l,j}-c, x_{l,k}+c$ for any constant c (the rectangle adjustment); similar property was considered already in [Shapley 1963]).

Transformations (a), (b) change the value of V, i.e. the average expected payoff V/m, thus can be used for transforming/comparing non-learnable games with different value of V. Transformation (c) does not change V/m and with induction on greatest difference in matrix it is easy to see, that with transformation (c) all games from the class NL with the same V could be reduced to a constant game (all elements of the matrix are the same - V/m). Since the rectangle adjustment transformation (c) is reversible it follows, that all games of the class NL with the same number of actions m can be transformed to each other, i.e. are all similar (nearly isomorphic).

3.6 Examples

Below is the decision table for the function ρ for a 3-ary game: this is the classical rock-paper-scissors game with encoding of player's actions *rock*=0, *paper*=1, *scissors*=2 and the set of rewards $R=\{-1,0,1\}$ (but using property (b) it could be changed to $R=\{-100000,0,100000\}$ - human players would like this much more!); in the table are indicated rewards r^i of player $p^i \in P$ (rewards of player $p^j \in P$ are $p^j=-p^i$); in the last row/column is indicated direction of monotonicity and the sum for the corresponding row/column.

Table. 4. Decision table for a rock-paper-scissors game with encoding of actions rock=0, paper=1, scissors=2

	p^2			
ρ	0	1	2	Σ
0	0	1	-1	$\rightarrow 0$
1	-1	0	1	$\rightarrow 0$
2	1	-1	0	$\rightarrow 0$
Σ	$\downarrow 0$	$\downarrow 0$	$\downarrow 0$	

(p^1) (4)

The same for a 4-ary game (m – even) with $R=\{-2,0,2\}$:

Table. 5. Decision table for a 4-ary group like game with $R=\{-2,0,2\}$.

	p^2				
ρ	0	1	2	3	Σ
0	0	-2	0	2	$\leftrightarrow 0$
1	2	0	-2	0	$\leftrightarrow 0$
2	0	2	0	-2	$\leftrightarrow 0$
3	-2	0	2	0	$\leftrightarrow 0$
Σ	$\updownarrow 0$	$\updownarrow 0$	$\updownarrow 0$	$\updownarrow 0$	

(P^1) (5)

From conditions 2.-4. in 3.3 it is clear, that if m is odd then these games are fair ([Aten 2019]), i.e. $|\rho^{-1}(r)| = |\rho^{-1}(r')| = m$ for any $r, r' \in \mathrm{R}$; if m is even, then $|\rho^{-1}(r)| = |\rho^{-1}(r')| = m$ for any $r, r' \in \mathrm{R}$, $r \neq 0 \neq r'$, $|\rho^{-1}(0)| = 2*m$ (every row/column in the table has 0 twice).

The property (c) of the class NL allows to modify the presented above example (2) with rectangle adjustment to totally non-symmetric form:

Table. 6. Modified with transformation $x_{0,1} - 2, x_{0,2} + 2, x_{2,1} + 2, x_{2,2} - 2$ matrix (3)

0	-1	4	-1
-1	0	1	2
2	1	-2	1
1	2	-1	0

(6)

Particular cases of this class of games were considered in [Henno et al 2018], [Henno et al 2019]; games belonging to this class were (for odd values of m) considered also e.g. in [Akin 2018].

The above minuscule examples could be combined to create a 'real' game, adjoining the matrices to a game (automaton) transitions graph, e.g. using as the transition table the graph of a 4-bit digital clock from [Henno 2017]; for rewards are applied matrixes from above (modified with transformations (a), (b)):

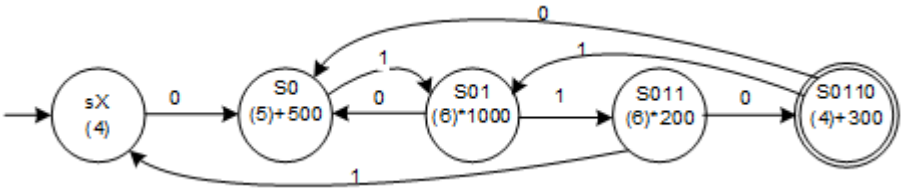

Fig. 3. A non-learnable game from a 4-bit lock automaton; inputs without transformations keep state.

4 CREATING ENCRYPTION KEY

In the above was presented a class on non-learnable games, i.e. where competing players always have to select their moves randomly and presented some examples of such games; some of such games ('Odd-Even', 'Matching Pennies') were considered in [Henno et al 2017] and above is shown, that such a game could be created for any vector of rewards. In the following is described a procedure for creating a shared secret (encryption key) using the randomness created by players when playing such a game where the best strategy for players is total randomness.

The server records sequence of player's moves, e.g. for a game with two players Alice and Bob with l moves this could be $m_{11}m_{12}m_{21}m_{22},...,m_{1t}m_{2t},...,m_{1l}m_{2l}$; here m_{1t}, m_{2t} are respectively moves of Alice and Bob in gameplay move t. Both players know only their own moves and from every result the probability any move of opponent which caused this result is the same (the above property (4)), thus players cannot guess moves of other players; if the situation is local, players e.g. can see each other moves, then they do not need any elaborated communication – they can speak directly with each other.

To generate a key server send to players the sequence of all moves from which the player moves are removed, e.g. server sends to Alice the sequence $*m_{12} * m_{22},...,*m_{2t},...,*m_{2l}$ - this information with holes does not give to an eavesdropper any information (it is assumed, that the game server communication with players is secure, so here is the only time when the game communication is used for the chat system). When players replace in the received sequences with holes their own moves they all get the same random sequence which could be used as the secure random key for symmetric encryption.

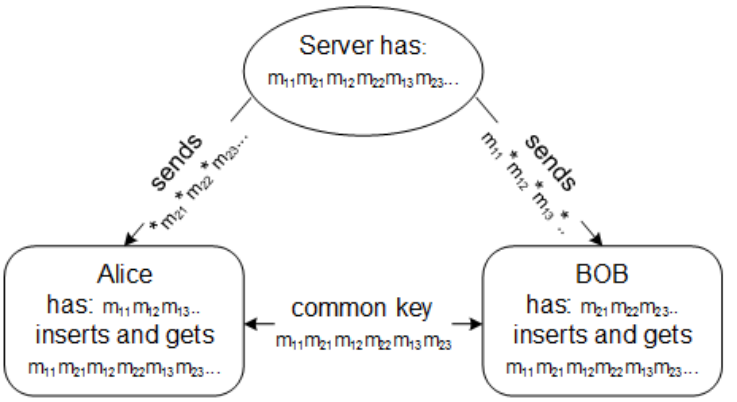

Fig. 4. Key generation combining a sequence with holes from server with sequence of player moves.

If all the following messages are secured/encrypted with the means used in game for sending player's moves and rewards, players store their moves in the order they made them and when player enters the game (logs in) he receives together with his gameplay client also (a link to) a cryptographic library, e.g. some member from the W3C list or (when using node.js) Node.js Crypto module, then the procedure to create a secret key could be described by following steps:

1. *Player1* sends to server message: *chat(playerList)*; here *playerList* is a list of players with whom he/she would like to communicate or the word *all* - player wants to start global chat
2. Server checks whether there is already enough entropy (at least e.g. 12-20 moves have been made) and *Player1* does not look like a bot - has comparable rewards
3. Server sends invitations to start chat to all members of *playerList* (with time limit for answer)
4. After time limit server removes from *playerList* all players who did not answer the invitation
5. If the remaining list contains some players, server adds to list also the *Player1* and sends to all members the global list of moves from which the moves of player himself are removed (replaced by '*') - with time limit for answer
6. Players replace the '*' in received list with their own moves (in the correct order); the corresponding procedure could also be delivered to all players already together with the game client
7. To check, players send to server (encrypted with the new key) message "Encryption OK"
8. Server removes from *playerList* the players, whose messages were not decipherable
9. If there still remain players (>1), server sends them all list of addresses of other members, thus they can start a chat forum.

To increase security of the key server could use some filters, e.g. for a randomly-selected value $r \in \mathrm{R}$ remove all moves which produced result r; to speed up the game could be used multi-moves, i.e. participants send in every move a fixed-length sequence of moves etc.; several test implementations are in work.

5 CONCLUSIONS

In above is considered a class of non-learnable games, where competing players should be maximally random and presented a framework for generating from players moves a secure key for symmetric encryption. The process of generating a key (shared secret) uses randomness generated in gameplay and does not use public-key step; thus it also does not have to distribute any personal information of players nor need any higher-level security authorities, i.e. the use of randomness and securing communication is self-sustained.

REFERENCES

Encrypted Traffic 2018. Encrypted Traffic Reaches A New Threshold. https://www.networkcomputing.com/network-security/encrypted-traffic-reaches-new-threshold

Cisco 2019. Encrypted Traffic Analytics. Cisco white paper.

D.. Maimon, Y. Wu, M. McGuire, N. Stubler. 2019. SSL/TLS Certificates and Their Prevalence on the Dark Web (First Report). https://www.venafi.com/sites/default/files/2019-02/Dark-Web-WP.pdf

SecurityWeek 2019. Study Finds Rampant Sale of SSL/TLS Certificates on Dark Web. https://www.securityweek.com/study-finds-rampant-sale-ssltls-certificates-dark-web

Hacker News 2017. The Rise of Super-Stealthy Digitally Signed Malware—Thanks to the Dark Web. https://thehackernews.com/2017/11/malware-digital-certificate.html

Kaspersky 2018. LuckyMouse Group is back and using a legitimate certificate to sign malware. https://www.kaspersky.com/about/press-releases/2018_luckymouse-group-is-back-and-using-a-legitimate-certificate-to-sign-malware

Effect of Encryption 2017. The Effect of Encryption on Lawful Access to Communications and Data.https://ec.europa.eu/home-affairs/sites/homeaffairs/.../encryption/csis_study_en.pdf

Terada et al 2017. Security Measures Based on Human Behavior ... - Fujitsu Global. https://www.fujitsu.com/global/documents/about/resources/.../fstj/...3/paper12.pdf

Leventhal 2018. Monitoring Users' Behaviors to Better Secure Data: One Health Plan's Story. https://www.hcinnovationgroup.com/cybersecurity/article/13029889/monitoring-users-behaviors-to-better-secure-data-one-health-plans-story

WePC 2019. 2019 Video Game Industry Statistics, Trends & Data. https://www.wepc.com/news/video-game-statistics/

Costikyan 2009. Randomness: Blight or Bane? http://www.costik.com/randomness-blight-or-bane.htm

A. Chalk 2009. Randomness in Gaming: Good or Bad? https://v1.escapistmagazine.com/forums/read/7.145275-Randomness-in-Gaming-Good-or-Bad

G. Costikyan 2013. Uncertainty in Games. MIT Press 2013, ISBN 0262018969 (ISBN13: 9780262018968)

Video Gaming 2019. WePC. 2019 Video Game Industry Statistics, Trends & Data, https://www.wepc.com/news/video-game-statistics/

House 2018. Evolving multiplayer games beyond Unet. https://blogs.unity3d.com/2018/08/02/evolving-multiplayer-games-beyond-unet/

C. Phillips 2018. Video Game Reward Types & The Player Experience. PhD Thesis, Queensland University of Technology, https://eprints.qut.edu.au/119100/1/Cody_Phillips_Thesis.pdf

Ah Reum Kang et al 2013. Online game bot detection based on party-play log analysis. Computers & Mathematics with Applications, Vol 65:9, pp 1384-1395

Luzgin R. 2018. Video Games as a Perfect Playground for Artificial Intelligence. https://towardsdatascience.com/video-games-as-a-perfect-playground-for-artificial-intelligence-3b4ebeea36ce

Shai Ben-David et all 2019. Learnability can be undecidable. Nature Machine Intelligence vol. 1, pp 44–48, https://www.nature.com/articles/s42256-018-0002-3

Shapley 1963. SOME TOPICS IN TWO-PERSON GAMES. The US Department of Defence Rand Corporation Memorandum RM-3672-PR

A. Plan 2017. Symmetric n-player games. www.asafplan.com/files/SymmetricGames.pdf

Shih-Fen CHENG et al 2004. Notes on Equilibria in Symmetric Games. Proceedings of the 6th International Workshop On Game Theoretic And Decision Theoretic Agents GTDT 2004 pp71-78

Akshay Wali et all 2019. Biological physically unclonable function. Communications Physicsvolume 2, Article number: 39 (2019), https://www.nature.com/articles/s42005-019-0139-3

CalTech 2004. Monotonicity and cyclical monotonicity. www.its.caltech.edu/~kcborder/Notes/CyclicalMonotonicity.pdf

C. Aten 2019. Multiplayer Rock-Paper-Scissors. https://web.math.rochester.edu/people/grads/caten2/documents/ALH_RPS.pdf

Henno et al 2018. Henno, J., Jaakkola, H., Mäkelä, J. Using Games to Understand and Create Randomness. Seventh Workshop on Software Quality Analysis, Monitoring,Improvement, and Applications. SQAMIA 2018, Novi Sad, Serbia, 27–30.08.2018, CEUR-WS, CEUR workshop proceedings vol. 2217, p. 1-9.

Henno et al 2019. Henno, J., Jaakkola, H., Mäkelä, J. Creating Randomness with Games. To appear in Acta Polytechnica Hungarica, http://uni-obuda.hu/journal

Akin E. 2018. Rock, Paper, Scissors, Etc - The Theory of Regular Tournaments. https://arxiv.org/abs/1806.11241

Henno 2017. Information and Interaction. Information Modelling and Knowledge bases XXVIII, IOS Press, pp 426-449