Rational authentication protocols and their use in financial transactions

Abstract. We use ideas from game theory to improve two families of authentication protocols, namely password-based and manual authentication schemes. The protocols will be transformed so that even if an intruder attacks different protocol runs between honest nodes, its expected payoff will still be lower than when it does not attack. A *rational* intruder, who always tries to maximise its payoff, therefore has no incentive to attack any protocol run among trustworthy parties. To illustrate the use of our method, we present a case study relating to the password-based authentication stage of on-line banking, where passwords are chosen either randomly or biasedly by, e.g., humans. For the latter we use the publicly available 32 million passwords of the social gaming network website RockYou as the source of human-selected passwords.

1 Introduction

Ideas from game theory have been used to re-design a number of fair exchange protocols [3] and secret sharing schemes [6, 4] so that nodes cannot act on their own interests to bring these schemes to failure. As an example, in a fair exchange, a party accepts to deliver an item iff it receives another item in return, and hence even unmalicious but self-interested parties will be tempted to deviate from a protocol to gain advantage. This notion of players' rationality or self-interest is however not applicable to authentication and key-agreement protocols where all honest nodes cooperate to complete a protocol successfully, because it is in their mutual interest that they agree on the same data.

We instead observe that in many environments, e.g. the financial industry, the intruder can be *rational* in the sense that it always tries to maximise its payoff as in the following scenario. If the intruder has somehow obtained A's financial details such as bank statement, it will know that A can have a large amount of money. This could imply that the potential reward of having access to A's account exceeds the cost of launching many attacks on different protocol runs. The intruder might therefore be highly motivated to attack the authentication stage of online transactions carried out between A and the bank. The observations motivate us to use techniques in game theory to redesign authentication protocols to resist this kind of rational intruder.

Our first contribution presented in Section 2 is a general protocol transformation that is applicable to a variety of authentication protocols. In this transformation an honest node, who is usually the protocol initiator, will pursue some additional behaviour under some probability after each successful protocol session. The combination of the behaviour and its occurrence probability is designed to ensure that an intruder's expected payoff in attacking the protocol is lower than its expected payoff in not attacking. The intruder therefore does not have any incentive to misbehave. Since the additional behaviours

of the initiator must benefit the intruder, they vary from one to another applications but an example with respect to the above banking scenario can be given as follows. To avoid being disrupted, the account holder A can occasionally make a small payment to a third party who is the intruder in disguise after each successful transaction. The questions we therefore want to answer are: How much is the *generous* payment? and How often does A need to make such a payment to successfully discourage the intruder from attacking?

The main thrust of this paper is to demonstrate how this protocol transformation works and benefits two families of pairwise authentication protocols. They are password based authentication schemes of Section 3 and manual authentication protocols of Section 5. In Section 6, we show how the transformation can be adapted to work with group protocols. Throughout the sections, we largely abstract away from the exact details of additional behaviours which are not immediately important to our analysis until we discuss our case study.

To assess the performance of our protocol transformation, in Section 4 we present a case study on the above scenario. While we will use our computation derived in Section 3 to answer questions posed previously, our experimental results shed new light on the usability and economic security of current banking applications regarding the limit of number of consecutive failed attempts of entering the correct password. In particular, the case will be studied in light of two very different sources of password: (1) random and uniformly distributed passwords chosen by, e.g., a centralised authority; and (2) human-selected, and hence biased, passwords as seen from the leak of 32 million passwords of the social gaming network RockYou following a security breach in 2009 [12].

Our use of additional behaviours in honest parties' activities tailored for authentication protocols can be traced back to earlier work in other context of rational secret sharing schemes. To encourage an intruder to give up attacking a protocol, it is probably inevitable that we need to give something, which is less damaging than a successful attack, to the intruder in each normal run. Both rational secret sharing schemes of Gordon and Katz [6] and Fuchsbauer et al. [4] follow this strategy by allowing a trusted dealer to send invalid shares of secret to players at the beginning of some iterations, or forcing nodes to proceed in a sequence of fake runs followed by a single real one.

Also cryptographic protocols are usually designed against arbitrary behavior of a malicious intruder, adopting the "worst case" viewpoint. The game-theoretic perspective however regards parties as being rational, and hence rational authentication protocols only need to deal with a rational intruder.

2 Protocol transformation

For simplicity pairwise authentication schemes are considered, where two parties A and B want to authenticate or agree on the same data. In the schemes, it is in honest nodes' mutual interest that they follow the protocol. Among the protocol participants, there is one party who initiates a protocol by, e.g., sending the first message and hence we denote A the protocol *initiator*. No specific

Protocol transformation

The protocol initiator A pursues the following strategy to discourage a rational intruder from attacking protocol runs of honest parties.

Upon each successful protocol session, which happens when the intruder either does not interfere with or succeeds in its attack on the protocol.

- With probability $\alpha \in [0, 1)$: A is *generous* and pursues an additional behaviour that benefits the intruder. The exact behaviour depends on the intruder's goals in different scenarios, but an example described in our case study of Section 4 is as follows. To avoid being disrupted, an account holder A makes a small payment to a third party who is the intruder in disguise after each successful transaction.

  The intruder will get payoff U when it does not attack or U_1^+ when it successfully attacks the protocol.

- With probability $1 - \alpha$: A is ungenerous and pursues no further activity. There is no payoff for the intruder if it behaves honestly, but if the intruder attacks and succeeds it will still get a payoff U_2^+.

Upon each unsuccessful protocol session, which usually happens when the intruder fails in its attack. The initiator A will not pursue any additional behaviour, and the intruder receives a negative payoff U^- due to, e.g., the cost of launching an attack on a protocol run.

Since the intruder much benefits from a successful attack regardless of whether A is generous or not, we arrive at:

$$min\{U_1^+, U_2^+\} > U > 0 > U^-$$

The following table summarises the payoff for the intruder according to different protocol outcomes and the initiator's strategy.

Strategy of intruder	**Protocol session outcome**	**Strategy of initiator**	**Payoff of intruder**
No attack	Succeed	Ungenerous	0
No attack	Succeed	Generous	U
Attack	Succeed	Ungenerous	U_1^+
Attack	Succeed	Generous	U_2^+
Attack	Fail	Ungenerous	U^-
The lower half is the worst case scenario of the upper half.			
No attack	Succeed	Ungenerous	0
No attack	Succeed	Generous	U
Attack	Succeed	Any	$U^+ = max\{U_1^+, U_2^+\}$
Attack	Fail	Ungenerous	U^-

Table 1. Protocol transformation.

protocol is given until multiple-run attacks are considered in subsequent sections, because for single-run attacks our suggested changes in the behaviour of the initiator A are independent of the type of authentication protocols whether they are based on passwords [1] or human interactions [8, 10]. Our analysis will be generalised to deal with group authentication scenarios in Section 6.

Prior to proceeding to the next paragraph, we would strongly recommend the readers to study the protocol transformation provided in Table 1, which also introduces the notation for the intruder's payoffs, i.e. U, U_1^+, U_2^+, U^-, with different combination of parties' strategies and protocol outcomes. The payoffs, which are often quantified in terms of money, depend on a number of factors, including the cost of launching attacks (computation or energy consumption) and financial reward of a successful attempt. As in many rational secret sharing schemes introduced to date [6, 4], we assume here that the payoffs are known to both protocol participants and the intruder. Moreover our analysis in this section as well as Sections 3, 5 and 6 does not require us to specify the additional behaviour of the initiator, because its abstract form in terms of the corresponding payoff for the intruder is sufficient. We will justify the assumptions when a case study is provided in Section 4.

From Table 1, we observe that the difference between the payoffs U_1^+ and U_2^+ for an attacking intruder can vary, e.g. they can be far apart or roughly the same. We therefore will tackle the worst case scenario here: regardless of whether A is generous or not the intruder's payoff is $U^+ = max\{U_1^+, U_2^+\}$ when it launches a successful attack as seen in the bottom of Table 1. A solution for the worst case scenario applies to every other scenario where $U_1^+ \neq U_2^+$.

Using the protocol transformation of Table 1, we arrive at this theorem.

Theorem 1. If an intruder can only attack up to a single run of an authentication protocol and succeed with probability ϵ, then to discourage the intruder from attacking protocol runs between honest nodes, this inequality must hold:

$$\alpha > \frac{\epsilon U^+ + (1-\epsilon)U^-}{U}$$

Proof. If the intruder does not misbehave, his expected payoff in each run is αU. If the intruder misbehaves, his expected payoff of a single-run attack is $\epsilon U^+ + (1-\epsilon)U^-$.

So as long as $\alpha U > \epsilon U^+ + (1-\epsilon)U^-$ or $\alpha > [\epsilon U^+ + (1-\epsilon)U^-]/U$, it is in the intruder's interest not to attack any protocol runs of honest nodes. □

Although U^+ is usually significantly bigger than U, ϵU^+ can still turn out to be less than U. This can be done by, e.g., choosing a password of a reasonable length so that the probability of a successful attack ϵ is small.

Suppose that there are a number of strategies regarding different values of α that node A can pursue, then α is selected big enough to meet the requirements of Theorem 1. In other words, α is big enough that the intruder's expected payoff is higher if it behaves honestly, as honest parties always prefer not to give the intruder too much benefit.

The above analysis only takes into account single-run attacks, in practice a rational intruder as defined in Section 1 would attack multiple protocol runs. For this reason, it is desirable that we consider the case of multiple-run attacks on authentication protocols.

3 Multiple-run attacks on password-based protocols

Any secure password-based (authentication or key-agreement) protocol usually need to resist off-line searching, i.e. the only way to find out a guess of a password is correct is to interact with the protocol participants. Our analysis here applies to a variety of password-based protocols, but for clarity we give the definition of the Diffie-Hellman-based Encrypted Key Exchange scheme of Bellovin and Merritt [1]. This protocol establishes a shared private key g^{xy}, where g^x and g^y are Diffie-Hellman keys of A and B, from a short password pw using an encryption scheme $E_{pw}()$ and a cryptographic hash function $hash()$.

Encrypted Key Exchange Protocol [1]
1. $A \longrightarrow B \ : \ A \parallel E_{pw}(g^x)$
2. $B \longrightarrow A \ : \ E_{pw}(g^y) \parallel hash(sk \parallel 1)$
where $sk = hash(A \parallel B \parallel g^x \parallel g^y \parallel g^{xy})$
3. $A \longrightarrow B \ : \ hash(sk \parallel 2)$

Passwords are selected from a set of n elements. While password distribution will vary depending on the population of users, we assume that it is completely known to the intruder. p_i denotes the probability of the i^{th} most common password, and hence we have $1 \geq p_1 \geq p_2 \geq \cdots \geq p_n \geq 0$ and $\sum_{i=1}^{n} p_i = 1$.

If the intruder decides to attack and proceeds in an optimal order of guessing then the chance of correctly guessing the password the first time is $\epsilon_1 = p_1$. If the first guess is incorrect, then the second guess succeeds with probability $\epsilon_2 = p_2/(\sum_{i=2}^{n} p_i)$. For all $t \in \{1, \ldots, n\}$ we have $\epsilon_t = p_t/(\sum_{i=t}^{n} p_i)$. It is worth to observe that $\epsilon_t \geq p_t$ for any $t \in \{1, \ldots, n\}$.[1] We refer to $\{\epsilon_i\}_{1 \leq i \leq n}$ the successive probabilities of correctly guessing the password.

In practice we usually limit the number of failed attempts an intruder can make, e.g. three wrong guesses and the protocol will stop running, and thus we denote k the limit of number of attacks an intruder can launch on a protocol.

In order to be precise in our arguments, we need to be clear about the attacking strategy of the intruder that our protocol transformation of Table 1 seeks to resist. If the intruder decides to attack a protocol up to k runs, then the intruder only terminates its attack following the end of the t^{th} attempt for $t \in \{1, \cdots, k\}$ if any of the following three conditions is met:

- The intruder succeeds in the t^{th} attempt.[2]

[1] Although $p_t \geq p_{t+1}$, it is not always true that $\epsilon_t \geq \epsilon_{t+1}$.

[2] For example, A is running an authentication protocol with a bank where A has an account. If the intruder successfully guesses the password in the t^{th} attempt, then it will take all of A's money. The intruder does not have any incentive to continue its attack because the account balance is zero and there is a cost of launching an attack.

– The intruder fails in the t^{th} attempt, but $t = k$ and hence the intruder has reached the limit of number of attempts.
– The intruder fails in the t^{th} attempt and $t < k$, but the expected gain of the next attempt is not positive or $\epsilon_{t+1}U^+ + (1-\epsilon_{t+1})U^- \leq 0$. Consequently there is no further incentive for the intruder to continue.

It is clear that this is the optimal strategy for any rational intruder who wants to launch up to k attacks on the protocol.[3] These k attempts do not need to be consecutive and can be interleaved with any number of protocol runs which are not attacked by the intruder. The readers might question what if the intruder blocks communication of a protocol run, but then it will get nothing, i.e. neither the prospect of a successful attack nor the benefit from additional behaviours of a generous initiator.

Without loss of generality we assume that $\epsilon_t U^+ + (1-\epsilon_t)U^- > 0$ for all $t \in \{1, \cdots, k\}$,[4] and thus the intruder will attack until it either reaches the k^{th} attempt or has succeeded before reaching that point. We summarise the intruder's cumulative gain $\{g_i\}_{1\leq i\leq k}$ and cumulative probability $\{\theta_i\}_{1\leq i\leq k}$ that it is successful up to k attempts in Table 2.

No. of attempts	**Outcome**	**Probability**	**Payoff of intruder**
1	Succeed	$\theta_1 = \epsilon_1$	$g_1 = U^+$
2	Succeed	$\theta_2 = (1-\epsilon_1)\epsilon_2$	$g_2 = U^- + U^+$
3	Succeed	$\theta_3 = (1-\epsilon_1)(1-\epsilon_2)\epsilon_3$	$g_3 = 2U^- + U^+$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
t	Succeed	$\theta_t = \epsilon_t \Pi_{i=1}^{t-1}(1-\epsilon_i)$	$g_t = (t-1)U^- + U^+$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
k	Succeed	$\theta_k = \epsilon_k \Pi_{i=1}^{k-1}(1-\epsilon_i)$	$g_k = (k-1)U^- + U^+$
k	Fail	$\theta'_k = \Pi_{i=1}^{k}(1-\epsilon_i)$	$g'_k = kU^-$

Table 2. This tables shows the cumulative payoff and probability of the intruder's success and failure when (s)he attacks a password-based protocol up to k runs.

From Table 2, the expected (average) number of protocol runs the intruder intervenes is

$$N = \theta_1 + 2\theta_2 + 3\theta_3 + \cdots + (k-1)\theta_{k-1} + k(\theta_k + \theta'_k)$$

[3] As a part of an attack on a password-based protocol the intruder will interact with honest nodes to check the accuracy of its guess of the password. Obviously the intruder can manipulate protocol messages without guessing the password, but this does not reduce the size of password guessing domain and hence is not optimal. Also there is no harm in modifying exchanged data and guessing the password at the same time.

[4] For otherwise the intruder will always terminate at the smallest $t < k$ such that $\epsilon_t U^+ + (1-\epsilon_t)U^- \leq 0$.

Similarly, the expected cumulative payoff of the intruder's multiple-run attack can be computed as follows

$$P = g_1\theta_1 + g_2\theta_2 + \cdots + g_k\theta_k + g'_k\theta'_k$$

Since the expected payoff an intruder gets from not attacking a protocol in each run is αU, in order to discourage the intruder from attacking a password-based protocol up to k runs, we must have

$$\alpha UN > P \text{ or } \alpha > \frac{P}{UN}$$

An important comment is that this strategy is "one size fits all" as it can be used to deal with arbitrary password distribution. To illustrate the use of our analysis, we will consider two scenarios in Section 4 where passwords are either

- uniformly distributed or randomly chosen by a centralised authority or
- biasedly selected by humans as in the social gaming network RockYou.

4 Case study

Let us suppose that A has perhaps accidentally revealed his financial details, e.g., bank statement to someone whom A later distrusts. A then wants to discourage that person or the attacker from interfering with online transactions carried out between him and the bank because (1) A can have a large amount of money in his account and (2) A wants to have the freedom to carry out transactions with other parties without being disrupted by the attacker. In the following scenario, the attacker plays the role of a lending company, but our work is applicable to other situations where the above condition applies.

- Party A has borrowed some money from a lending company (who can be the mafia in disguise).
- A however delays making the payment because of, e.g., further investment, and this goes against the interest of the lender who does not trust A.
- To make a loan, the lender must have seen A's financial proofs such as bank statements, and therefore knows that A potentially has a bank account of up to 30 thousand US dollars. With this information the lender will be tempted to break into A's account and get all of A's money.

Whenever A carries out an online transaction, he authenticates himself to the bank by typing in his password on the bank's website.

Let us suppose that it costs the lender 0.1 US dollar[5] to interfere with the authentication stage of the online banking protocol. Upon a successful attack the payoff for the lender is thus $U^+ =$ (30,000 - 0.1) US dollars. If the lender fails, it gets a negative payoff $U^- = -0.1$ US dollar due to the running cost.

[5] According to `www.csgnetwork.com/elecenergycalcs.html` the cost of running a home computer system for one hour is 0.08 US dollar.

To discourage the lender from misbehaving, A will pursue the following additional behaviour. Each time after A authenticates himself to the bank successfully, with probability $\alpha \in [0, 1)$ the account holder or borrower will make a payment of U US dollar to the lender. The value of U will be dependent on password distribution as discussed in the next two subsections. The payoff for the lender after a successful online transaction which it does not interfere is therefore U US dollar, i.e. of course the lender does not get this payment when it decides to attack but fails.

In our description so far, neither have we specified how passwords are chosen nor the value of a generous payment U. In the next two subsections, we consider two different cases where passwords are selected randomly by, e.g., a machine or biasedly by humans. In both cases password distribution is known to the lender who plays the role of a rational intruder, as this will determine the lender's optimal order of password guessing.

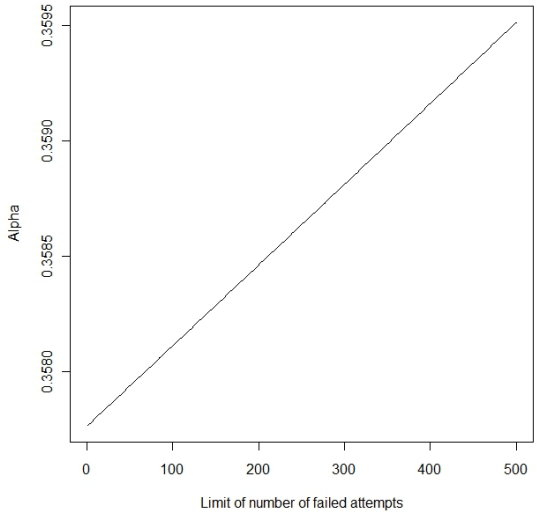

Fig. 1. Randomly chosen 16-bit passwords.

4.1 Random passwords

We consider an ideal scenario where 16-bit (or 4-hexadecimal-digit) passwords are randomly selected by, e.g., a centralised authority such as a bank. This means that $n = 2^{16}$ and $p_1 = p_2 = \cdots = p_n = 2^{-16}$. The probabilities of successive password guesses of being correct is therefore

$$\epsilon = \epsilon_1 = \frac{1}{n} < \epsilon_2 = \frac{1}{n-1} < \epsilon_3 = \frac{1}{n-2} < \cdots < \epsilon_k = \frac{1}{n-k+1}$$

and hence this inequality $\epsilon_t U^+ + (1-\epsilon_t)U^- > 0$ holds for any $t \in \{1, \cdots, k\}$, where $1 \leq k \leq n$. This implies that there is always an expected positive gain when the lender launches an k-run attack on the authentication stage of online transactions.

By substituting the values of $\{\epsilon_i\}_{1 \leq i \leq k}$ into our calculation of Section 3, the cumulative probabilities of correctly guessing the password at different attempts throughout a k-run attack can be shown to be $\theta_1 = \theta_2 = \cdots = \theta_k = 1/n$ and $\theta'_k = (n-k)/n$. Using information of Table 2, we can compactly derive the expected number of protocol runs the intruder intervenes as follows:

$$N = \theta_1 + 2\theta_2 + 3\theta_3 + \cdots + (k-1)\theta_{k-1} + k(\theta_k + \theta'_k) = \frac{k(2n-k+1)}{2n}$$

and its corresponding expected cumulative payoff:

$$P = g_1\theta_1 + g_2\theta_2 + \cdots + g_k\theta_k + g'_k\theta'_k = \frac{kU^+}{n} + \frac{k(2n-k-1)U^-}{2n}$$

To discourage the lender from attacking, we must have $\alpha > \frac{P}{UN}$. We therefore arrive at the following condition for α.

$$\begin{aligned}
\alpha &> \frac{kU^+}{nUN} + \frac{k(2n-k-1)U^-}{2nUN} \\
\alpha &> \left(\frac{1}{n} + \frac{k-1}{n(2n-k+1)}\right)\frac{U^+}{U} + \left(1 - \frac{1}{n} - \frac{k-1}{n(2n-k+1)}\right)\frac{U^-}{U} \\
\alpha &> \frac{\epsilon U^+ + (1-\epsilon)U^-}{U} + \left(\frac{U^+ - U^-}{U}\right)\Delta, \text{ where } \Delta = \frac{k-1}{n(2n-k+1)}
\end{aligned}$$

In this experiment, we set the value of the generous payment to be $U = 1$ US dollar. As seen from Figure 1, we use the above calculation to plot α against $k \in \{1, \cdots, 500\}$ where k is the maximum number of attacks the lender can launch. In this experiment ϵ, n, U, U^+, U^- are fixed as defined earlier.

It is very clear from both Figure 1 and the above condition for α that as k increases so does α but very slowly, i.e. α is around 0.35 for any $k \in \{1, \cdots, 500\}$. This is because as k increases so do Δ and hence α. However since $\epsilon > \Delta \geq 0$, the difference between the bounds for α with respect to single-run (see Theorem 1) and n-run attacks is $\Delta(U^+ - U^-)/U < \epsilon(U^+ - U^-)/U$, which can be very small given that the password is of a reasonable length.

We argue that this can have a significant impact on many banking applications which usually set $k = 3$ and so can be inconvenient to use especially after one comes back from a holiday and there are too many passwords to remember. What this experiment shows is that when the quality of passwords is strong as in this case, the number of consecutive attempts of entering a wrong password can be increased significantly without compromising the economic security of online banking protocols.

The readers might question what if A does not pay the lender the small amount of money, even though A had agreed to it. The answer is as follows:

the lender will regularly monitor its own account to check whether this small payment occurs with probability α with respect to the total number of successful transactions carried out by A. If this agreement were violated, the lender would change its mind and re-launch its attack immediately.

4.2 Human-selected passwords – RockYou

In this section, we study the effect of our protocol transformation when passwords are selected by humans. To the best of our knowledge, the only reliable and publicly available source of human-selected passwords comes from the social gaming website RockYou following the leak of 32 million passwords in 2009. The security breach also revealed passwords to outside websites, including Facebook and MySpace who run software applications developed by RockYou. The data have proved invaluable for password research since then [2, 12]. In our studies, we only need the distribution of the human-selected passwords, which has been kindly provided to us by an expert in the field. The same information can be extracted from the password list publicly available on the Internet.[6] The exact passwords are irrelevant to our studies, but for information we provide the top 15 most popular passwords of RockYou here.

Rank	Password	Probability	Rank	Password	Probability
1	123456	$p_1 = 0.00892$	9	12345678	$p_9 = 0.00063$
2	12345	$p_2 = 0.00243$	10	abc123	$p_{10} = 0.00051$
3	123456789	$p_3 = 0.00236$	11	nicole	$p_{11} = 0.00050$
4	password	$p_4 = 0.00182$	12	daniel	$p_{12} = 0.00047$
5	iloveyou	$p_5 = 0.00153$	13	jessica	$p_{13} = 0.00047$
6	princess	$p_6 = 0.00102$	14	monkey	$p_{14} = 0.00045$
7	1234567	$p_7 = 0.00067$	15	lovely	$p_{15} = 0.00044$
8	rockyou	$p_8 = 0.00064$	500	xavier	$p_{500} = 0.000067$

It is not difficult to check that $\epsilon_i U^+ + (1 - \epsilon_i)U^- > 0$ for $1 \leq i \leq 500$,[7] and hence there is no harm for the lender to attack up to 500 protocol runs.

Since $\epsilon_1 = p_1 = 0.0089 \gg 2^{-16}$ of the previous case regarding random passwords, we will need to significantly increase the generous payment U to have a chance of discouraging a rational intruder from misbehaving. We therefore set $U = 30$ US dollar.

Since there is no uniform pattern in the distribution of human-selected passwords, it is not possible to derive a compact formula for α as in Section 4.1. It is however possible to use the method of Section 3 to calculate and then plot α against $k \in \{1, \cdots, 500\}$. Again U, U^+, and U^- are fixed in this experiment.

[6] Website: `http://www.skullsecurity.org/wiki/index.php/Passwords`

[7] Observe that $p_1 \geq p_2 \geq \cdots \geq p_{500}$ and $\epsilon_i \geq p_i$ for all i, we therefore have $\epsilon_i \geq p_{500}$ for $1 \leq i \leq 500$. Since $p_{500}U^+ + (1 - p_{500})U^- = 1.9 > 0$, we arrive at the above condition.

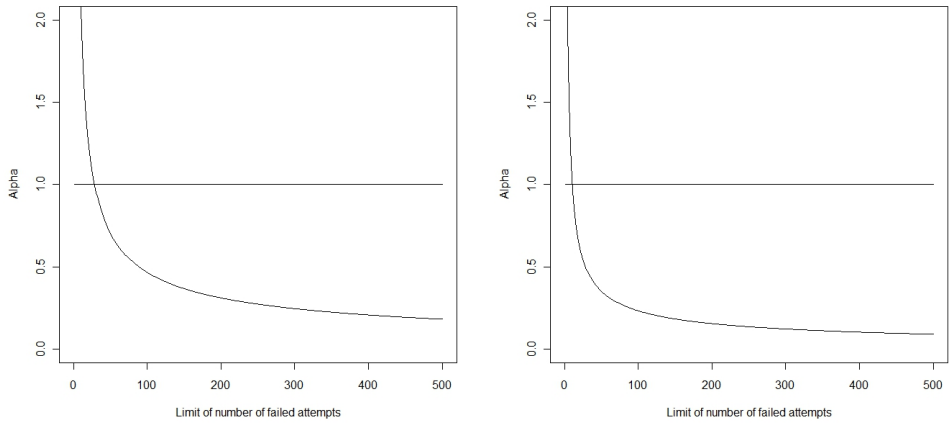

Fig. 2. Human-selected passwords taken from the social gaming network RockYou. The first and second graphs correspond to $U = 30$ and $U = 60$ US dollar respectively.

To our surprise, it is very clear from Figure 2 that as k increases the value of α decreases, which is the opposite of what can be observed from the previous experiment regarding randomly selected passwords. This is because of the strong bias of a small number of easy-to-predict passwords, whose occurrence probabilities are much higher than every other in the list. Consequently, as the lender launches more attacks on the protocol, the expected gain per attempt reduces due to significantly less popular passwords, and hence so does α.

This has an important impact on the effectiveness of our protocol transformation. The first graph of Figure 2 indicates that even setting $U = 30$ US dollar is not enough to discourage the lender from launching up to 28 attempts, because α would need to be greater than 1 when $k \leq 28$. Of course we can increase U to 60 US dollar as seen in the second graph of Figure 2 to reduce the value of α, but that would be too expensive for A to afford.

What we therefore can conclude from this experiment is as follows. If passwords are poorly chosen as in RockYou:

- It is not financially feasible to discourage a rational intruder from launching a small number of attacks. This justifies the 3-time restriction on entering wrong password successively currently set in many banking applications.
- On the other hand, if party A is confident that his or her password is strong and random then A can certainly resist the potential danger of the lender launching a small number of initial attempts targeting weak passwords. The 3-time restriction can be lifted and setting $\alpha = 0.47$ would fence off the lender from making 100 or more attacking attempts.

We note that human-selected passwords for banking applications are probably stronger than RockYou passwords because of the financial importance and also, especially after the security breach at RockYou, many weak pass-

words will have been blacklisted. In some cases, e.g., Chase Bank in the USA self-service PIN change is not possible and changes must be made in person, making it easier for the bank to strengthen the quality of users' passwords. Nevertheless our studies further highlights the danger of using weak passwords.

We have illustrated the use of the protocol transformation in banking, we however have to regconise that the case study considered here has limitations. In particular, we have only considered banking scenarios where the identity of the intruder is known to or suspected by the protocol initiator, even though such a restriction reduces the number of potential attackers and hence makes the protocol transformation feasible. Investigation of how other financial applications may also benefit from this study will be a subject of future work.

5 Multiple-run attack on manual authentication protocol

In contrast to password-based schemes, the chance of a successful attack ϵ on a manual authentication protocol run remains unchanged regardless of how many times an attack is launched. This property applies to all secure protocols of this type, e.g. oneway, pairwise or group authentication [5, 8, 11].

Our analysis here applies to many manual authentication protocols, but for clarity we give a pairwise protocol. In this scheme, parties A and B want to authenticate their public data $m_{A/B}$ from human interactions to remove the need of passwords, private keys and PKIs. The single arrow ($\longrightarrow$) indicates an unreliable and high-bandwidth link (e.g. WiFi or the Internet), whereas the double arrow ($\Longrightarrow$) represents an authentic and unspoofable channel. The latter is not a private channel (i.e. anyone can overhear it) and it is usually very low-bandwidth since it is implemented by humans, e.g., human conversations, text messages or manual data transfers between devices. $hash()$ and $uhash()$ are cryptographic and universal hash functions. Long random keys $k_{A/B}$ are generated by A/B, and k_A must be kept secret until after k_B is revealed in Message 2. Operators $\|$ and $\oplus$ denote concatenation and exclusive-or.

A pairwise manual authentication protocol [8]
1. $A \longrightarrow B$: $m_A, hash(k_A)$
2. $B \longrightarrow A$: m_B, k_B
3. $A \longrightarrow B$: k_A
4. $A \Longleftrightarrow B$: $uhash(k_A \oplus k_B, m_A \parallel m_B)$

To ensure both parties share the same data, the human owners of devices A and B have to compare a short universal hash value of 16–32 bits manually. Since the universal hash key $k_A \oplus k_B$ always varies randomly and uniformly from one to another protocol run, the chance of a successful attack on each protocol run ϵ equals the collision probability of the universal hash function.[8]

[8] We note that our protocol transformation of Table 1 and the analysis of this section also apply to other manual authentication protocols, including schemes of Vaudenay [10] and Hoepman [7] which do not use a universal hash function.

Definition 1. [9] An ϵ-almost universal hash function, $uhash : R \times X \rightarrow Y$, must satisfy that for every $m, m' \in X$ and $m \neq m'$:

$$\Pr_{\{k \in R\}}[uhash(k, m) = uhash(k, m')] \leq \epsilon$$

For a b-bit universal hash function the best-possible ϵ is 2^{-b} and there are various constructions that achieve close to this [9].

To discourage the intruder from attacking a manual authentication protocol in multiple runs, we use the protocol transformation of Table 1. Upon a successful protocol session, with probability α the initiator A pursues additional behaviours that benefit the intruder.

Since the chance of a successful single-run attack ϵ is unchanged, the value of α required to discourage a multiple-run attack is the same as in a single-run attack of Theorem 1. We thus arrive at the following theorem.

Theorem 2. When an intruder is allowed to attack a manual authentication protocol up to k runs for any $k \geq 1$, then to discourage the intruder from attacking protocol runs between honest nodes, this inequality must hold:

$$\alpha > \frac{\epsilon U^+ + (1 - \epsilon)U^-}{U}$$

6 Extended protocol transformation

We have so far focused on the use of the protocol transformation of Table 1 on pairwise authentication protocols where there is a designated role for the protocol initiator who is generous with probability α after each successful protocol session. Although the same protocol transformation is applicable to group authentication schemes, it might be difficult for multiple protocol participants to agree on who will be the initiator. We therefore would like to remove the need for such an initiator by assuming that every honest node can be generous with probability α independently after a successful protocol run. In other words, we extend our protocol transformation and apply it to the behaviour of every group protocol participant, and hence the *extended protocol transformation*.

In a group authentication protocol, there are always two or more nodes in a group $\mathbf{G}$ who want to authenticate or agree on the same public data. For the types of considered protocols not all of the protocol participants have to be honest, and this means that these compromised principles are not obliged to follow our protocol transformation.[9] We therefore denote p the number of honest parties out of all protocol participants and $p \geq 1$.

When the intruder does not attack (and thus a protocol runs successfully), there are two main possibilities that affect the payoff of the intruder:

[9] In a password-based group protocol, all parties share a common and private password pw. Though some of these parties can be dishonnest, neither will they reveal pw to the intruder nor use their knowledge of pw to fool other honest protocol participants into agreeing on different and corrupt data. The conditions must hold for otherwise it is impossible to resist an intruder who possesses the password.

– With probability $(1-\alpha)^p$, every honest node is ungenerous and there is no payoff for the intruder.
– With probability $1-(1-\alpha)^p$, there is at least one generous node. The payoff for the intruder might vary according to the number of generous parties, but as in pairwise schemes we consider the worst-case scenario where the intruder's payoffs is always the same under this condition.

When the intruder attacks a protocol run, there are also two possibilities that affect the intruder's payoff:

– With probability ϵ, the attack succeeds and hence the protocol also terminates successfully. The payoff for the intruder also can vary according to the number of generous protocol participants, but again we only consider the worst-case scenario where the intruder's payoff is the same here.
– With probability $1-\epsilon$, the intruder fails and so does the protocol run. The intruder gets a negative payoff.

We summarise the payoffs for the intruder in different scenarios in Table 3.

Strategy of intruder	Outcome of protocol	Strategies of honest nodes	Payoff of intruder
No attack	Succeed	All ungenerous	0
No attack	Succeed	≥ 1 generous node	U
Attack	Succeed	Any	U^+
Attack	Fail	All ungenerous	U^-

Table 3. A summary of the extended protocol transformation.

Based on the damages an intruder might cause to honest parties, it is clear from Table 3 that we always have:

$$U^+ > U > 0 > U^-$$

The same analysis as provided in Sections 3 and 5 can be used to show how both password-based and manual authentication protocols can similarly benefit from our extended protocol transformation.

7 Conclusions and future research

We have used ideas from game theory to transform two families of authentication protocols, namely password-based authentication and manual authentication protocols, to make them resilient against a rational intruder. In these protocols, only the intruder is self-interested and all other trustworthy protocol participants should cooperate to complete a protocol run successfully, since it is in their mutual interest to agree on the same data.

At the heart of our protocol transformation is the introduction of some additional behaviours protocol participants can pursue to discourage the intruder. In addition to making a generous payment that might only be suitable to and practical in banking applications, another possibility of an additional behaviour is that the initiator would occasionally exchange and authenticate random data of no use. Such a decision would have to be made by the initiator probabilistically and discreetly at the start of each protocol run instead of at the end of each successful protocol session. The consequence, which is in the interest of any anonymous intruder, is that it is a waste of time for other honest protocol participants who are not aware of the uselessness of the authenticated data. We intend to investigate this possibility further in our future studies.

While we have explored the notion of rational intruder in two types of authentication protocols, our work reported here opens the way to a number of new problems. For example, it would be interesting to investigate how relevant the notion of a rational intruder is to other types of authentication protocols which are based on PKIs or long private keys. Also our studies on password-based protocols hopefully would lead to further attempts in improving the usability and economic security of many banking applications based on passwords which are currently quite inconvenient to use regarding the limit on the number of consecutive failed attempts of entering the correct password.

References

1. S.M. Bellovin and M. Merritt. *Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks.* Proceedings of the IEEE Symposium on Research in Security and Privacy (Oakland): 72.
2. J. Bonneau, S. Preibusch and R. Anderson. *A birthday present every eleven wallets? The security of customer-chosen banking PINs.* FC '12: The 16th International Conference on Financial Cryptography.
3. L. Buttyán, Jean-Pierre Hubaux, and S. Čapkun. *A Formal Analysis of Syverson's Rational Exchange Protocol.* Proceedings of the 15th IEEE CSF 2002.
4. G. Fuchsbauer, J. Katz and D. Naccache. *Efficient Rational Secret Sharing in Standard Communication Networks.* TCC 2010: 419-436
5. C. Gehrmann, C. Mitchell and K. Nyberg. *Manual Authentication for Wireless Devices.* RSA Cryptobytes, vol. 7, no. 1, pp. 29-37, 2004.
6. S.D. Gordon and J. Katz. *Rational secret sharing, revisited.* In Proceedings of Security and Cryptography for Networks. LNCS vol. 4116, 229-241, 2006.
7. J.-H. Hoepman, *Ephemeral pairing problem/* Proceeding of the 8th International Conference on Financial Cryptography, LNCS, Vol. 3110, A. Juels, ed., Springer, 2004, pp. 212-226.
8. S. Laur and K. Nyberg. *Efficient Mutual Data Authentication Using Manually Authenticated Strings.* LNCS Vol. 4301, pages 90-107, 2006.
9. D.R. Stinson. *Universal Hashing and Authentication Codes.* Advances in Cryptology - Crypto 1991, LNCS vol. 576, pp. 74-85, 1992.
10. S. Vaudenay. *Secure Communications over Insecure Channels Based on Short Authenticated Strings.* Crypto 2005, LNCS vol. 3621, pp. 309-326.
11. J. Valkonen, N. Asokan and K. Nyberg. *Ad Hoc Security Associations for Groups.* In Proceedings of the Third European Workshop on Security and Privacy in Ad hoc and Sensor Networks 2006. LNCS vol. 4357, pp. 150-164.
12. M. Weir, S. Aggarwal, M. Collins and H. Stern. *Testing metrics for password creation policies by attacking large sets of revealed passwords.* Proceedings of the 17th ACM Conference on Computer and Communications Security. ACM, 2010, pp. 162-175.