

Social Authentication Protocol for Mobile Phones

Bijan Soleymani and Muthucumaru Maheswaran
 Department of Electrical and Computer Engineering
 McGill University
 Montreal, QC, Canada
 {bsoley, maheswar}@cs.mcgill.ca

I. INTRODUCTION

Web applications such as online banking, online shopping carts, and so on, depend on the user authenticating himself securely. Traditionally this involves a username and password and if more security is required an electronic token is used in addition to this password. Other than these two “factors” there is also biometrics, such as fingerprints, retinal scans and voice recognition. Thus the traditional systems use some combination of these three factors: something you know (passwords), something you have (tokens) and something you are (biometrics).

Recently it has been suggested that a fourth factor: someone you know also be part of the authentication process [1]. J. Brainard et al. have applied this technique to the problem of emergency authentication, as a replacement for challenge questions or calls to a help-desk. The idea is that the user uses a token and pin to authenticate himself. If the user forgets his token, he can ask a friend who has their token to grant them a temporary password.

Thus fourth factor or social authentication is based on the process of vouching. In this method a user asks a friend to vouch for them, that is the friend must recognize the user and then issue some proof of this recognition, which the user then uses to log in to the service. In [1], this vouching was done explicitly, with the user contacting a friend and literally asking for a vouching code. In this paper we will use users’ cellphones to automate this process. Whenever a user calls a friend, a token will be issued “vouching” for this contact. These tokens if obtained in sufficient numbers can then be used to prove that a user is who he says he is.

In addition to this fourth factor we will make use of other means of authentication. These include a PIN (personal identification number) that must be entered when validating the “vouching” tokens, possibly fingerprint recognition and outputs from other biometric sensors, such as a wrist watch with heart-rate monitor, or a shoe with built-in pedometer. In this case we may want two out of three or four of these to match before authenticating the user.

The rest of the paper is organized as follows. Section 2 will review some of the related work. In Section 3, we will outline the details of our proposed scheme. In Section 4, we will test its feasibility by running simulations on data from the Reality Mining project from MIT [2]. The implementation of our authentication technique will be explained in detail in Section 5. In Section 6 we will present the results of the

implementation. In Section 7 we will examine the possible threat scenarios. The conclusion will be presented in Section 8.

II. RELATED WORK

Our work combines two techniques that have been widely used separately: using the user’s mobile phone as an authentication device and using the user’s social network as an authentication factor.

A. Mobile Phone as Authenticator

There are two ways in which to use a mobile phone for authentication, but both of them involve the user proving that they are in possession of the device. The first is to contact the user at authentication time through the telephone network. This can be achieved by either sending the user a one-time code by SMS [3], or by calling the user and requiring them to enter a PIN [4]. A problem with the first approach is that SMS traffic may be snooped.

Another way in which to use a mobile phone for authentication is to make it carry a public/private key or a certificate. At authentication time the user is asked to prove that they have the private key, thus proving they are in possession of the phone. However as we will see in section 7, without a Trusted Platform Module (TPM) that restricts access to the key, anyone with access to the phone will be able to read the key and possibly transfer it to another device.

In [5] the authors propose a scheme whereby a certificate is issued to each phone that binds a public key to the device’s Bluetooth MAC address. Thus even if an attacker obtains the user’s certificate it will only work on the phone with that particular MAC address. However if the attacker can modify his Bluetooth stack to report the user’s MAC address then the same problem occurs. The solution, securing the Bluetooth stack, is similar in nature to using a TPM.

With a TPM the phone can be used to securely store the user’s cryptographic credentials, such as private keys. Thus the phone can provide the functionality of a smartcard, saving the user the need to carry an extra piece of hardware. In addition the phone can provide an interface that helps the user update and manage their credentials [6].

A distinction can be made between the device’s identity and that of the user [7]. In this case the user proves their identity through another means (e.g. a username and password) and the private key and certificate prove the identity of the device

(e.g. the phone). Thus access can be restricted based on either the user, the device or both.

There is at least one proposed system that is a hybrid between these two approaches [8]. The idea is to generate a one time password using certain information unique to the user's phone and a PIN number. If this should fail then a one time password can be sent to the user's phone by SMS.

B. Social Network as Fourth Factor

It seems that making use of a user's social network to facilitate authentication hasn't been explored as fully. This may be because unlike the previous case this isn't a simple extension of existing techniques and technologies. Rather it is a completely new approach to computer authentication.

There are two different ways this can be used: one involves contacting the members of the user's social network in order to securely authenticate the user while the other involves making use of the user's account on a social networking site to securely contact members of the user's social network.

The first approach is the one proposed in the RSA paper [1]. As described in the introduction it involves the user contacting a friend when he has forgotten his token. The friend logs in with her own token, requests a "vouching" code for the user and relays this to the user, who can use this to log in temporarily. Our approach also falls under this category. The user obtains "vouching" tokens from their friends, whenever they have a phone conversation or a Bluetooth sighting. And these tokens are used to log in. The important feature of this approach is that the user's friends are contacted directly (by phone or Bluetooth), and this contact is used to prove the user's identity when logging in to a site or service.

The second approach contacts the user's friends through the social networking site. Social network sites provide peer-discovery (finding friends) and secure messaging (instant messages). Therefore it is possible to set up a secure communication channel with a friend by sending them a key through the social network, and using that key to encrypt subsequent transmissions that will travel through the open Internet. This can be done manually by the user, or it can be implemented in the application that needs to send the encrypted data. A special API can be built to facilitate the interaction between applications and social networking sites [9].

C. Social Authentication on Mobile Phones

Finally we can consider the combination of mobile phone and social networking for authentication. One of the goals of the Reality Mining project is to measure users' social networks using mobile phones. This can be done based on call patterns (which indicate who the user was talking to) and Bluetooth sightings (which indicate who the user was close to). To apply this to authentication one can take these measurements and compare them to typical values for the user. For example one can measure the devices (friends) in the user's Bluetooth range and compare this to the value for a typical day [10]. Our scheme considers both call patterns and Bluetooth sightings, but does so in a slightly different way.

III. PROPOSED SCHEME

In this section we will outline the details of our authentication scheme. First we will focus on the social authentication based on telephone conversations and Bluetooth sightings. We will then consider additional factors that can be used in order to minimize the risk of intrusion.

A. Social Authentication

The goal of this scheme is to leverage the functionalities of a Bluetooth capable cellphone in order to automate the process of vouching. The user will obtain vouching tokens from friends and will use them together with a PIN to log in to a secure service.

The user starts by declaring a list of friends that will "vouch" for him. This list is stored on a central server. After a phone call with one of these friends the user will receive a token indicating that this communication took place. A token is only issued after a phone call that is longer than a minimum duration. This duration is determined by analyzing the distribution of the user's call durations. The idea is that it is unlikely that an intruder will be able to make a phone call long enough to receive a token, without alerting the other side that something is wrong.

Similarly after seeing a friend using Bluetooth a token will be received confirming this sighting. Bluetooth sightings are trickier because the long range (10m) doesn't imply that the users actually made contact. Thus the Bluetooth sightings are augmented in two ways. First a rough estimate of the distance of the other user is made by measuring the time it takes to establish a Bluetooth connection. Secondly the user is prompted to confirm the sighting of the other party. After both of these take place the vouching messages are exchanged.

The user authenticates himself to the central server by sending the required number of fresh tokens, along with entering his PIN number. After repeated errors in the PIN number the tokens become invalid and new tokens must be obtained.

B. Supplementary Factors

In addition to the "vouching" tokens which are "someone you know" (fourth factor authenticators), the PIN which is "something you know" and the private keys (see section 4) which are "something you have" we can also make use of "something you are" (biometrics).

Just as it is common for a user to carry a cellphone with them, we would like to make use of other devices within the user's reach. For example it is likely that when using this system to authenticate to an online banking website, the user will be using her laptop, which may be equipped with a fingerprint scanner. A fingerprint scan can then be used in conjunction with the "vouching" tokens as an additional factor.

Use of a special-purpose wearable authentication device in the form of a wristband has been suggested [11]. This device would take fingerprint scans, but would also monitor the user's vital signs, including: heart rate, skin temperature, body capacitance and acceleration. Using these readings the

person wearing the device is transparently authenticated once they are within radio range of the target computer.

Footstep characteristics have been used for personal identification [12]. Many users carry an iPod or palmtop device with them. These may be equipped with a pedometer that measures the user's footsteps. The characteristics of the footsteps can be analyzed and compared to the user's baseline measurements. In the event of a match the user is authenticated. The use of this feature further improves the reliability of the authentication.

In our system the primary means of authentication are the "vouching" tokens and the PIN. For additional security the user may opt to require one of the above biometric factors to authenticate herself. For example consider a user with a laptop and pedometer system. In addition to the social authentication if either the laptop recognizes the user's fingerprint or the pedometer's readings match the user's gait, then the user will be authenticated.

IV. SIMULATION ON REALITY MINING DATASET

The Reality Mining project at MIT's Media Laboratory follows 100 cellphone users over the 2004-2005 academic year. These 100 students, faculty and staff were each given a Nokia 6600 smartphone with an installed application that would log their usage. Data collected includes: call logs, Bluetooth devices within range, cell tower information, application usage and phone status [2].

Among the data collected by the Reality Mining project, the readings of most interest to us are the call logs and the Bluetooth device sightings as these would lead to the issuing of "vouching" tokens.

Looking through the data, we see that there are many days over the course of the year, when there is no call data nor any Bluetooth sightings. So the first thing we did was determine the number of days that each phone logged any relevant data. This is shown in Figure 1 below. As can be seen the number of days with data varies between 0 and a little over 250 for the different users. Looking more closely we see that 8 users never logged any data.

In order to simulate the issuing of a "vouching" token based on phone conversations. We tried to establish a list of friends for each user. We used the ten most popular numbers that they talked to, which we assume to be their friends. This is the best case scenario, in actuality the user may call a certain number very often and yet this number may not correspond to a friend. So the actual results may be slightly worse than these simulation would indicate.

We then established the minimum duration of a call before tokens are sent or received. When choosing the minimum duration there is a trade-off between security and convenience. If we choose it too small then an impostor can make random calls to the user's phonebook and then hang up after receiving the token. On the other hand if we put the threshold too high then very few tokens will be generated, and it will take a long time for a user to have enough tokens to be authenticated. In this case we tallied the durations of each user's calls and took the 25th percentile as this minimum duration. We do this to eliminate the effect of wrong numbers and such very short

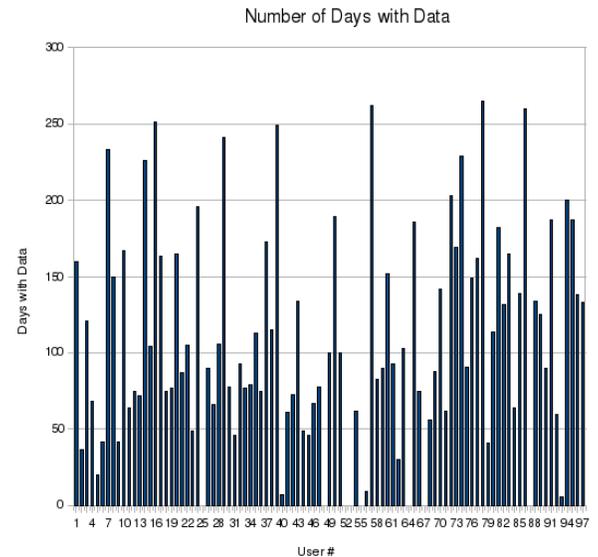


Figure 1.
Days with Data

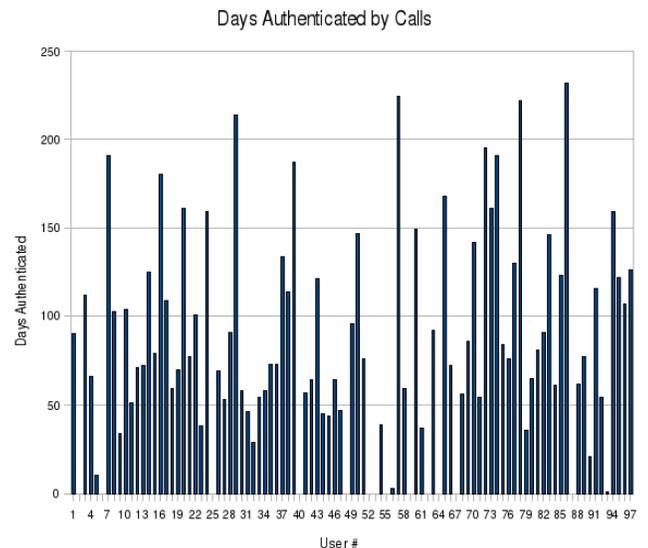


Figure 2.
Days Authenticated by Calls

calls, and at the same time keeping the probability of false rejection small.

With these two pieces of data in hand we then looked at each day and considered the two days immediately preceding it. If two friends or more friends were called in the current day or those two preceding days and each at least two of those calls were longer than the minimum duration, we assumed that two or more tokens were generated. In this case we only require two tokens to authenticate and thus the user can be authenticated on that day. Figure 2 illustrates the number of days where the user could be authenticated based on phone conversations.

On the average the users are able to authenticate themselves on 74% of the days.

The next step was to investigate the effect of adding Bluetooth device sightings to the authentication process. Again

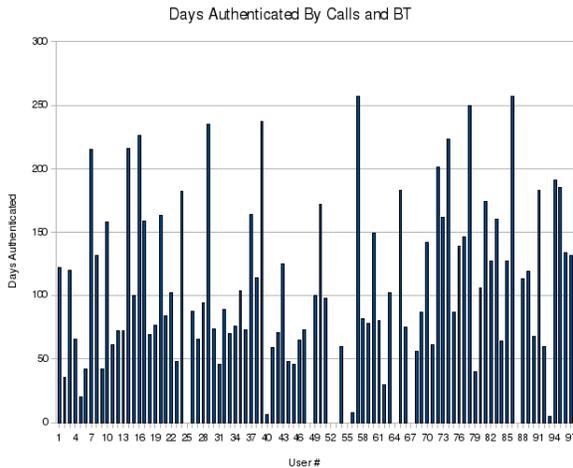


Figure 3.
Days Authenticated by Calls and BT

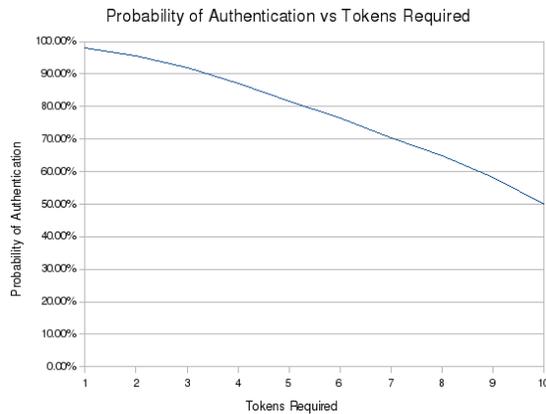


Figure 4.
Probability of Authentication vs Tokens Required

for each one of the top ten devices that the user sees in the current day or the previous two days, the user receives a token. If the user has two tokens, either two from phone calls, or two from Bluetooth sightings or one of each, then the user is assumed to be authenticated for that day. Figure 3 displays the number of days where the user could be authenticated based on both factors.

We see that the numbers are much closer to those of Figure 1. In fact on the average users can authenticate on 95% of the days.

In the above we required two tokens for authentication. Obviously there is a trade-off between the number of tokens required and the probability of successful authentication. We varied the number of tokens required between 1 and 10, and calculated the probability of authentication. As Figure 4 shows, the probability of authentication varies between 98% and 50%. We chose 2 tokens above to have a high probability of authentication.

There is a similar trade-off between the number of days a token remains valid and the probability of successful authentication. We required two tokens and then varied the number of days between 1 day and seven days. As can be seen in Figure 5, the probability of authentication varies between 82%

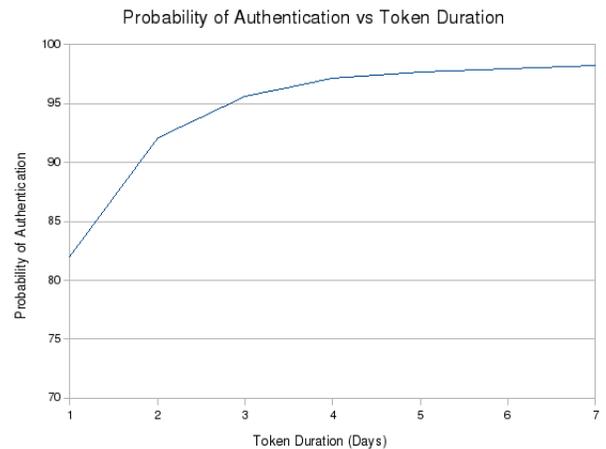


Figure 5.
Probability of Authentication vs Token Duration

and 98%, and levels off after 3 days. We again chose 3 days (the current day plus the two preceding ones) to have a high probability of authentication.

V. IMPLEMENTATION

Our implementation uses PKI (public key infrastructure) to provide confidentiality and integrity. In particular we use an implementation of the RSA encryption algorithm written in Python (Pys60, Python for Nokia S60 phones). We chose to code for Pys60 because it has modules that allow easy access to a number of phone features that we need, including: call logs, SMS inbox, SMS messaging and Bluetooth. Additional modules can be written in Symbian C++, in order to either speed up critical sections of the code or to provide access to low-level hardware not accessible from the standard modules. It is important to note that our system could be implemented using other programming languages such as C or Java for phones without a Python interpreter.

A. Issuing Tokens

Each node has a public/private key pair. A copy of the public key resides on the server. When a user Bob (B) wants to give a token to user Alice (A), Bob signs (encrypts with his private key) A 's name and the current time T_b : $K_{BS}(A, T_B)$, and then encrypts this with A 's public key $K_{AP}(K_{BS}(A, T_B))$ and sends it to Alice. Only A can decrypt this to retrieve $K_{BS}(A, T_B)$.

Tokens are issued when a phone conversation with a friend occurs and it is over the minimum duration, or a friend is sighted over Bluetooth and is within a small distance.

Our software continuously scans the call logs and looks for calls that are to or from friends. It then checks the length and if the call is long enough a token is issued as above.

Similarly our software continuously scans for friends over Bluetooth. To determine whether the friend is close or not we time the duration of the Bluetooth obex discovery call. The Bluetooth obex discovery function, takes the target device's MAC address and returns a list of obex (Object Exchange) services available on that device. It turns out that the weaker

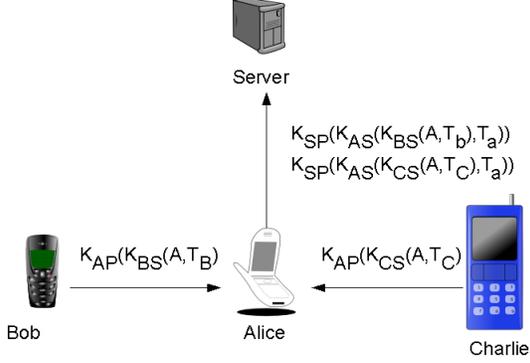


Figure 6.
Using Tokens to Authenticate

the signal between the phone and the device it is trying to discover the longer this function takes. We call the function four times and time the last three. If all three calls took less than 0.06 seconds, then we ask the user if they would like to send a token to the corresponding friend. 0.06 seconds was chosen because it was determined experimentally to be a typical value when the device and the phone are within line-of-sight of each other.

B. Using Tokens to Authenticate

When A wants to use a token from B , she concatenates the current time T_a and signs with her private key K_{AS} : $K_{AS}(K_{BS}(A, T_b), T_a)$. And then encrypts with the server's public key K_{SP} : $K_{SP}(K_{AS}(K_{BS}(A, T_b), T_a))$. This is illustrated in Figure 6. The server can decrypt this to recover T_a , A and T_b . The server checks that A matches user A . That T_a is close to the current time. And T_b is within the allowed lifetime of a token.

Then the server issues a challenge for the PIN. If A responds correctly then she is authenticated.

If not the challenge is repeated. If A fails repeatedly then the tokens is invalidated and new tokens are required. This is done by requiring tokens with timestamp newer than T_b .

VI. IMPLEMENTATION RESULTS

Having implemented the system we can now consider its practicality. One major factor is the battery usage. Both public-key cryptography and Bluetooth scanning are considered big battery drains. We will consider each in turn.

A. Public-key cryptography and battery life

We use public-key cryptography when generating tokens and when using tokens to authenticate. To test the effects of generating tokens on the phone's battery life we ran the code to generate a token in a continuous loop. We ran this on a phone with a full charge and let it run until the battery ran out. The result was that 7140 tokens were generated in 18710 seconds (or 5 hours 11 minutes and 50 seconds). This means that it takes 2.62 seconds to generate a token. This includes signing and then encrypting the message.

How this will affect battery life depends on how often the phone will be required to generate tokens. Even generating as much as 700 tokens per day would only drain 10% of the phone's battery.

B. Bluetooth Scanning

Our program continuously searches for friends within Bluetooth range. Therefore to measure Bluetooth scanning battery usage we can simply run our program and see how long it takes to drain the battery. Scanning for Bluetooth connections every 10 to 20 seconds, drained the battery in about 32 hours. This is a little over 3% battery usage per hour.

Assuming the above 700 tokens per day, and 16 hours of daily use between charges the program as it stands would drain 60% of the phone's battery in a typical day (50% Bluetooth scanning, 10% token generation). This leaves 40% of the battery for talk-time and other applications. Obviously the frequency of the scanning could be decreased to increase the battery life. But it seems that the program as it stands is still usable, if a little battery hungry.

VII. THREAT SCENARIOS

Let's consider the case where an intruder Trudy wants to authenticate herself as legitimate user Alice. Let us consider the case with the vouching tokens and a simple PIN (no biometrics). There are many possible combinations of scenarios, depending on the intruder's possession of the phone, the pin, and the tokens.

A. Intruder does not have phone.

In this case as in all cases the intruder needs to obtain enough tokens and also obtain the PIN. Even if Trudy were to obtain the PIN, without access to Alice's phone Trudy can't get tokens directly from Alice's friends. She can't generate false tokens without the friends' private keys. But she can still snoop the SMS and Bluetooth traffic and grab the tokens as they are transmitted. However without the Alice's private key she can't decrypt the tokens to make use of them. Therefore the security of the tokens depend on the security of Alice's private key.

B. Intruder has phone

Now the problem is that without a TPM (Trusted Platform Module), Alice's private key is stored as a file or other accessible structure on her phone. Anyone with access to Alice's phone can copy the key, or send it to themselves over the network. Thus if Trudy can gain physical access to Alice's phone all bets are off. We can encrypt the key with a PIN, but that leads to a new problem. Either the user will constantly have to enter the PIN, since the private key is needed whenever a token is issued, or the user will enter the PIN once and then the private key will be stored in memory unencrypted, where Trudy can get to it if she has access to the phone.

Once Trudy has the phone there are three possibilities. If Trudy is lucky the phone already contains enough tokens, in that case she can try to authenticate using those tokens. If she

also has the PIN it is game over for Alice. If there are not enough tokens on the phone, Trudy has two options: either return the phone to Alice and snoop tokens off the network, or keep the phone and try to obtain enough tokens directly.

1) *Return phone and snoop*: This approach is rather straightforward. The only problem is that Trudy has to return the phone before Alice notices it has disappeared and reports it as stolen. This is a problem if Trudy is a stranger, but if Trudy and Alice are friends then Trudy would simply have to borrow Alice's phone. Once the phone is returned Trudy waits, grabs enough tokens, and then all she needs is the PIN.

If the phone is equipped with a TPM then this approach is impossible, as there should be no way to extract the private key from the TPM. Thus Trudy will be forced to use the phone and get the tokens directly.

2) *Keep phone and get tokens directly*: There are several ways Trudy can go about trying to obtain tokens directly. For voice calls she can impersonate Alice to her friends, but this may backfire as the friends might find out and contact Alice or report suspicious activity on Alice's account. If Trudy and Alice are friends then she can call mutual friends and say that she is borrowing Alice's phone. Another tactic is for Trudy to call Alice's friends, say that she has found the phone and wants to return it to its owner. She may be able to drag out the conversation long enough for a token to be issued.

For Bluetooth sightings, she can trail Alice and receive tokens whenever Alice crosses a friend. Whenever they cross one of Alice's friends, Alice's phone will ask Trudy if she wants to send a token. Trudy will do so. The friend will then assume he is receiving a token from Alice and will send back a token in return. However this is dangerous, because Trudy has to be within line of sight of Alice for this to work, so only a particularly daring intruder would pull this off. However if Trudy and Alice are friends this may be a bit easier to do.

C. Stealing multiple phones

Of course we hope that the intruder is unable to obtain enough tokens either indirectly, because the phone has a TPM, or directly, because the intruder is unable to impersonate the user or otherwise fool her friends. However there is still a way for Trudy to get the required tokens: steal multiple phones, from the user's friends. This way Trudy can use these phones to generate tokens, by going through the call or Bluetooth token generation process.

Of course to succeed this requires the theft of N phones, where N is the number of tokens from different friends that are required to authenticate. The greater N the more secure the system is.

D. Once the intruder has enough tokens

Even if Trudy can obtain enough tokens, she can't authenticate without the PIN. Trudy can try randomly guessing the PIN. With a 4 digit PIN and 3 attempts, that gives Trudy a 3 in 10000 (0.03%) chance of success, and a 99.97% chance of failure, which will invalidate the tokens she obtained. Furthermore the system could be made to detect repeated

failed attempts (say 100 failed attempts in a row), and then require a longer PIN or a manual reset.

On the other hand if Trudy can obtain the PIN, by for example observing Alice enter it while logging in, then she has all she needs to login. Thus if Trudy has only the tokens, the security of the system depends on the security provided by the PIN. While if Trudy has only the PIN, the security of the system depends on the security provided by the tokens.

VIII. CONCLUSION

With a standard security token, the intruder needs the token and the PIN to masquerade as the user.

In our protocol the theft of a single phone would not necessarily result in a security breach, even if the PIN is known to the intruder. To generate enough tokens the intruder needs to have N phones. Where N is the number of tokens required for authentication. Having a large N , maintains high security. However it makes authentication more difficult.

REFERENCES

- [1] J. Brainard, A. Juels, R. Rivest, M. Szydlo and M. Yung, "Fourth-Factor Authentication: Somebody you know", CCS'06, October 30-November 3, 2006, Alexandria, Virginia, USA.
- [2] N. Eagle and A. Pentland, "Eigenbehaviors: Identifying Structure in Routine", Behavioral Ecology and Sociobiology.
- [3] "MobileKey (Mobile Authentication Server)", MobileKey http://www.visualtron.com/products_mobilekey.htm.
- [4] PhoneFactor "Tokenless Two-Factor Authentication", PhoneFactor <http://www.phonefactor.com/how-it-works/overview/>.
- [5] R. Ghosh and M. Dekhil, "I, Me and My Phone: Identity and Personalization using Mobile Devices", HP Technical Reports, HPL-2007-184.
- [6] M. Mont, B. Balacheff, J. Rouault and D. Drozdowski, "On Identity-Aware Devices: Putting Users in Control across Federated Services", HP Technical Reports, HPL-2008-26.
- [7] M. Mont and B. Balacheff, "On Device-based Identity Management in Enterprises", HP Technical Reports, HPL-2007-53.
- [8] F. Aloul, S. Zahidi and W. El-Hajj, "Two Factor Authentication Using Mobile Phones", IEEE International Conference on Computer Systems and Applications (AICCSA), Rabat, Morocco, May 2009.
- [9] A. Ramachandran and N. Feamster, "Authenticated out-of-band communication over social links", Proceedings of the first workshop on Online social networks, Seattle, 2008.
- [10] A. Frankel and M. Maheswaran, "Feasibility of a Socially Aware Authentication Scheme", Consumer Communications and Networking Conference, Las Vegas, 2009.
- [11] S. Ojala, J. Keinanen and J. Skytta, "Wearable authentication device for transparent login in nomadic applications environment", 2nd International Conference on Signals, Circuits and Systems, 2008. SCS 2008. 7-9 Nov. 2008, pp. 1-6.
- [12] A. Itai and H. Yasukawa, "Footstep classification using wavelet decomposition", International Symposium on Communications and Information Technologies, ISCIT 2007, 17-19 Oct. 2007 pp.551 - 556.