Pico: No More Passwords!
(Transcript of Discussion)

Frank Stajano

University of Cambridge, Cambridge, UK

Virgil Gligor (session chair): We have a session about passwords and you will hear at least two different points of view—possibly even two contradictory points of view, which is par for the course for this workshop. Our first speaker is Frank Stajano who argues that there should be no more passwords.

Frank Stajano: My title should give you a hint about my position towards this problem. What's a password? A password is a way to drive users crazy!

Passwords were not so bad when you had only one or two of them, and when a password of eight or nine characters was considered a safe password. Nowadays computers have grown so powerful that ten character passwords can be brute-forced with the kind of computer you buy in the supermarket next to your groceries. And you don't just have one or two passwords: you have dozens of them, because there are so many more services that now require you to have a password.

If you, poor user, listen to the computer security people, they will say that your passwords must be unguessable, otherwise attackers with figure out what they are; they must be impossible to brute-force, therefore you must fill them up with special symbols, and a strong mix of upper and lower case, and put in numbers as well; and you must not write them down, so you must make sure that you don't forget the complicated passwords that you make up; and, however many passwords you have, they must all be different. This set of requirements is a problem.

If you look at what people who develop websites think, for them the password is very convenient, because every user knows how to authenticate by password, so no training is needed; it's very cheap, because you don't need any equipment at the prover end; and it's very easy to implement, because there are standard library functions for computing the hash and so on. For software developers, the password is an easy way to do user authentication. But if you ask users, then passwords are a real pain. While each developer individually thinks it's OK to use a password ("well, everybody else also requests a password, so what's wrong if I do too?"), users, instead, end up with so many passwords that remembering them all is an unmanageable problem. That's what we call the tragedy of the commons.

If you look at these requirements that we (the unreasonable computer security people) inflict on regular users, it's obvious that there's an empty intersection between them: no passwords will satisfy all of these constraints, so users are quite fed up with passwords, and rightly so. I haven't done a proper user study but I have acted (as I'm sure every one of you has too) as the informal help

B. Christianson et al. (Eds.): Security Protocols 2011, LNCS 7114, pp. 82–97, 2011.

desk for all my relatives and friends who are not into computers. And one of the first things they mention is: "How can I do all of these stupid things with passwords?" (All different, complicated, never written down etc.) I sympathize with their sentiment that the requirements they get from the computer people are contradictory, and that it's unfair for us to ask that they follow all these constraints that can't be satisfied simultaneously.

Alf Zugenmaier: This last constraint, that passwords cannot be written down: why is that? There seems to be some computer security folklore that says "you must not write down passwords"; but why?

Reply: Several pieces of advice are now folklore, as you rightly mention, including others such as "you must change your passwords every month". Despite the fact that this advice is not necessarily rational, it is still being given to users[1] and users still believe they have to comply with it.

If we could start again from zero, and if we could forget about passwords, and find another way of dealing with user authentication, what would it be? There have been a number of interesting proposals in the literature for fixing parts of this problem, web authentication in particular, and I understand that the following talk in this session is essentially about that. But I argue that that's no longer enough, because for a user a password is a password, whether it's used for web authentication or for any other purpose; and if you get users fed up with passwords, they will be fed up with *all* passwords, not just web passwords; so we should solve the problem globally, not just for the web or for online systems. I want to get rid of all passwords, and the way I want to do this is by going from "something you know" to "something you have". I propose a device that acts like a memory prosthesis that frees up the part of your brain that is now devoted to remembering passwords.

What minimum requirements would a password replacement system have to offer? First of all, on the usability side, users should not have to remember passwords: that's the whole point of the exercise. And, as far as security goes, the new method should be at least as secure as passwords (if it were possible to comply with all these contradictory requirements). Then, scalability: the new method should work even if you start having thousands of entities that request a password from you. By the way, let me call "apps" these entities that act as the verifiers of your passwords (not to be confused with cell phone apps). If we are shifting from passwords to tokens, then the new system must also offer availability: if you lose the token then it must be possible for you to regain access to the apps that were protected by those credentials. And the fact of having physical control of the token should not be enough to allow anyone to impersonate you: your token, even if stolen, should not be usable by anyone other than you. These are my baseline requirements.

Now I'm going to stick my neck out and explicitly list the benefits that my system promises to provide; then you can check against the rest of what I say

[1] Or even enforced by some operating systems, in the case of password expiration.

and see if that's true[2]. My system is called Pico, because there was a guy called Pico della Mirandola a long time ago who had a very good memory, so Pico can take care of the memory effort in your stead. The first thing Pico promises to do is to relieve the user from having to remember passwords. Another problem with passwords is that users will choose a password that is weak, easy to guess or easy to brute-force: that is not possible with Pico. Another problem is that users will recycle passwords between different apps, and then if one of the apps is malicious it can impersonate the users with the other apps; or if one of the apps is careless and has its password database stolen and cracked then, even if that app was not malicious, the user who recycled the password can now be impersonated everywhere else; again, this is not possible with Pico.

The No-Typing benefit is going to be satisfying for many users: regardless of whether you can remember it, you won't have to type the password any more. The No-Phishing benefit is a slightly subtler point: you could be persuaded to type your password into the wrong app, and that would then allow a malicious attacker to reuse your password with the real app, either in an online man-in-the-middle attack, or just offline to screw you later; but the Pico offers facilities that prevent apps from impersonating other apps.

An important point is that Pico works not just for web authentication, but for relieving your memory of *all* passwords, passphrases and PINs, even the one for your burglar alarm. Another point is that, if you have dozens or hundreds of different apps and your token has the credentials for all of them, it becomes tiresome to have to scroll through menus to get to the right one to select which credential to give to the current app. With the Pico you don't have to go through menus to decide which credentials to send to which app.

Matt Blaze: Conspicuously absent from your list[3] is the requirement that apps be allowed to work as if Pico didn't exist.

Reply: Right. I am starting from a clean slate and therefore I won't worry about backwards compatibility, at least until later.

Matt Blaze: OK, so this is not an entirely client-side design?

Reply: No, not at all: it requires changes to the apps to provide all these benefits. Besides, these are just the benefits I claim to provide, and I don't actually provide the benefit that you mention. This one here is not a list of requirements but a list of promises.

Matt Blaze: OK, so the price of your benefits is that the apps have to change?

[2] In the pre-proceedings presentation the benefits were numbered. In the following rewrites, as I kept rearranging and expanding the set, I decided instead to give them short mnemonic names. I have now translated these names in the transcript, which now reads "No-typing" where I said "Benefit 4".

[3] The list on the slide had only the promised benefits of the Pico. Following Matt's comment, I explicitly added the "non-goals" section to the table in the paper.

Reply: The price is that apps have to change, and more; but down here, CONTINUOUS, is an additional benefit I will give you for changing your apps. With passwords, you type in your password, then you have an interaction; or maybe (after you typed the password) someone else has an interaction, and the app is not really sure whether *you* are still in front of the app. Maybe you've left it, you have gone to the toilet for a minute and someone from the desk near you is using your computer, or something like that. With the Pico the authentication is continuous, it's not just a point in time, it happens all along, and therefore the app knows that you are there for all the time that you are there. So you don't have a session that is started with you there and then lasts for two hours; or, as Joe pointed out to me, you login to a website and then you have a persistent cookie for two weeks, and it's assumed that you're still the same guy who authenticated two weeks ago; with Pico, this doesn't happen. Extra benefit.

The physical design of the Pico is that of a small device that has a wireless connection to your main computer, has a camera, has some buttons and a display. It's natural to think about it in the form factor of a smartphone, which already has all these things, but Pico doesn't have to be shaped like a smartphone, nor does it have to be an application on your smartphone; it could be something small, it could be a tiny square touch-screen gadget like the latest generation iPod, very cute, or maybe something I could attach to my keychain like this key fob; or maybe a watch, an item of jewellery, or whatever. You may visualize it as a smartphone if you wish but it doesn't have to be shaped like that.

Each app will have a public and private key, and will have a visual code that is basically a certificate for the public key, a certified graphical version of the public key that can be acquired by the camera. We can thus build a multi-channel protocol around the process of selecting the app. I point the Pico at the app I want, acquiring the app's code, and then the app sends its public key to the Pico over radio, because a public key is probably too big to fit inside a visual code. So there are two channels: the camera, to acquire the visual code, and the radio, which is bidirectional, to exchange more bits and do everything else that's required in the protocol.

This part will look a bit like SSL: the app, which is a bit like a website, offers a public key; and the client, which is the Pico, recognises that it is the right one, although without using a PKI. Then enough stuff goes on that they achieve mutual authentication and set up a session key (shared secret) to protect the confidentiality and integrity of the rest of their interaction.

The app will always show a kind of front page which displays the visual code: that's equivalent to having the fields where you type in your username and password. At that stage, the Pico can do two things. One is the equivalent of typing in the password, and the other one is registering with that app for the first time; one or the other happens depending on which of two main buttons you press on your Pico. So the two principal actions on the Pico are: "offer credentials", which is like typing in your password, or "initial pairing", which is like creating a new account.

Later I will mention systems that already do a number of things that the Pico does, but none of them does the same subset. One of them is the password manager that Firefox has. You visit a website, you type in your password, it offers to remember it and then pre-types it for you when you next visit that same site (with everything encrypted under a master password). Firefox is second-guessing what you're doing and sometimes it gets confused, for example on some occasions where you have to define your password on a different page from the one where you enter it later (or when you change passwords). With the Pico system it's very clear that there is just one place, the place where the visual code of the app is displayed, where you can do both of these actions.

I'm now going to describe those two actions, "initial pairing" and "offer credentials". The first one is done only once per application, while the second one is done much more frequently, at each authentication. First, the user points the Pico at the chosen app, to acquire the visual code of the app. This is how the user communicates her intent of interacting with that particular app, without the Pico getting confused by other apps nearby that might advertise their public key over the radio at the same time. This is initial pairing so we expect that the app might be new to the Pico; but there's actually a subtlety because you might register with the same app again if you want to have another identity, such as when you open a second Gmail account. For that reason you are still allowed to do an initial pairing with an app that is already registered in your Pico, and you would get a different account for it. Conversely, if the app is not already known to the Pico, then the other button for "offer credentials" won't work; in fact, instead of working, it might give you a warning that you're probably facing a phishing attack because, if you have never paired with that app, it would make no sense for you to log into it.

The Pico gets the public key of the app through the radio channel and checks that it matches the acquired visual code, because otherwise it could be some other malicious app that is sending the public key instead of the one that you want to interact with. Then the Pico talks to this app by encrypting messages to the app's public key: the Pico sends the app an ephemeral public key, one that's just made up for this temporary interaction, and it does so to preserve the user's privacy in case that the app is fake. If the app cannot prove that it owns the secret key matching the public key in the visual code, then there is no reason for Pico to disclose the identity of the user to that app. Therefore, only after the Pico is convinced that the app knows the secret key of its public key will the Pico give the app its permanent public key for that account. Inside, the Pico has a different key pair for every account.

Bruce Christianson: Is that limited to asymmetric keys or may it just be a symmetric key at this stage?

Reply: Admittedly, there aren't that many advantages in using the public key crypto for the client side. You do need the public key crypto for the app side, but on the client side you could do almost exactly the same stuff just by having some secret bit string that is then transferred through the channel that you have

established with the app's public key[4]. And in fact this is one getaway trick that I can use later for saying, in cases where I need to be more backwards compatible, then I will have the Pico just remember a secret string and send it to the app that way. Note that, if I take this route, I still have to change the app so that it can receive something over the public-key-protected channel; but, compared to just typing in the password, I avoid all the attacks of eavesdropping, keylogging, hijacking, phishing and so forth.

The username (which identifies the user to the app) is defined on the application itself and is then sent back to the Pico. There is little point in sending this username to the Pico because, to the app, the ultimate identifier of the user is its public key; however, it's nice to have some human-readable name just for the benefit of the human user, to browse accounts in the Pico, when he has to select which account to activate of the several that he has with that particular app: if I have three Gmail accounts and I want to send my credentials, I have to say which ones in that case.

If we are talking of initial pairing with something like a Gmail account, where anyone can get a new account whenever they like, then there's not much else that the app needs to do. But if this initial pairing connects to some resources that already belong to the human user, even before doing the pairing (as in the case where I register for online banking and my existing bank account needs to be attached to this online persona that I am making now), then in that case the app at the back-end needs to check that the user who's registering is the one linked to whatever resources are available (in this case the bank account) before doing this linking. And so, offline, before any of this interaction starts, the back-end will send the user a letter containing two visual codes: one is the visual code of the app, the same one that the app will present on its start page; and another one is a visual code that encodes some authenticator that says "OK, you are the Pico public key that is going to be linked to the resources of your bank account". The Pico acquires these two codes from the letter, and its behaviour will be: "I will only spit back this authenticator to the app whose public key matches the other visual code I acquired in this atomic transaction". And then, as part of the "initial pairing" protocol, the Pico sends that authenticator to the app, and the app checks that it's the right one, and then it links that newly created account to those existing resources.

What happens during "offer credentials", the normal day-to-day interaction where I would usually type the password? The beginning of the operation is practically the same as initial pairing: the Pico requires the visual code of the app the user wants to interact with, but then finds this time that it has a public key memorized for it. (As I said before, if the app was not known then you cannot offer credentials but maybe there's a flag that says "you are perhaps being phished".) An ephemeral public key is sent by the Pico as before and, once the Pico is happy that that app really knows its secret key, then it sends

[4] The pre-proceedings version of the paper discussed the trade-offs betwee symmetric and asymmetric cryptography in greater detail but this was dropped from the final paper for brevity, having decided not to offer symmetric keys as a variant.

its long-term public key which can be challenged by the app—and that's the part where, after authenticating the app to the user, we authenticate the user to the app. Then they do whatever else they would normally do after verifying passwords. Except that, as I anticipated, the Pico system offers the possibility for the app to keep checking whether the user is still there throughout the session, instead of just during the verification at the beginning.

Alf Zugenmaier: You mean that the app can check if the *device* is still there?

Reply: Good point. Indeed the app can only check through radio whether the Pico is still there, and the question is: how does it know that the Pico is still attached to the user as opposed to having been left on the desk? Let me jump ahead and talk about that right now, even though it was going to come a few slides later. This is the topic of Pico locking and unlocking.

As I said earlier, it's not appropriate for a token to be available to anyone who holds it: your token should only work when it's in *your* possession. The token will thus have a locked mode and an unlocked mode, and when it's in locked mode then you cannot use it, you cannot retrieve its credentials or operate with it in any other way. This implies some kind of tamper protection of the device, which is very important, but at this stage is still just an engineering detail. The whole contents of the Pico should be encrypted all the time.

But the more difficult question is: when is a good time to unlock the Pico? I have been working on authentication throughout my computer security career; regulars of this workshop will remember that, the first time I was here, I talked about the Resurrecting Duckling for pairing up two devices. The following year my talk also had to do with the Resurrecting Duckling and one thing came up in that discussion which I am re-using now: using family feelings between various ubiquitous computing devices to decide whether they are in a safe situation or whether they have been kidnapped. We thus have some Picosiblings, which are siblings of the Pico, and they could be your glasses, your watch, your keyring, your belt, things that you tend to have with you all the time. They also talk through radio; when the Pico is within range of all these Picosiblings it feels safe, but if it's left on the desk because you go to the toilet, then it's not with its Picosiblings any more, and from that it deduces that it is no longer with the user. Besides these we have two special items, which I will describe in a moment, but let's just stick with the standard Picosiblings for now. Each sibling holds a share, and all these shares together[5] make the key that unlocks the Pico. These shares are pinged periodically by the Pico. There is a decay counter for each share, which is refreshed (topped up) every time you find the share in these pings; if shares are missing or expire, they cannot be used to reconstruct the full key that unlocks the Pico, so the Pico is unusable. That way, if you leave your Pico on the desk and go to the toilet, it cannot authenticate. If you come back from the toilet before another timeout, the Pico will find the Picosiblings at the next ping and it will work again. If that longer, second timeout is also exceeded,

[5] The design proposed in the pre-proceedings version was based on n-out-of-n secret sharing, later changed into k-out-of-n for greater flexibility and usability.

then the Pico will switch itself off and you will have to restart from zero with all the shares present.

Alf Zugenmaier: What is my incentive not to leave all Picosiblings in one big pile on the desk?

Reply: Your incentive is so that you don't lose all your keys.

Alf Zugenmaier: I could have one Pico on my person all the time, or I could put all the Picosiblings on one keyring.

Reply: You could.

Alf Zugenmaier: And then, if I leave, the Pico and all the Picosiblings are still together and it all looks the same to them. What's the incentive for me? What benefit do I get out of not putting all my Picosiblings on the same keyring such that, if I lose it, I lose all of them?

Reply: Well, the benefit is exactly what you're saying!

Alf Zugenmaier: No, I am not talking of security benefits now: outside of this room, nobody cares.

Bruce Christianson: Security benefits are not real!

Reply: The reason for having Picosiblings like your glasses, your watch or your earrings is precisely because they will be "attached to you" without you even thinking about it. The Pico can sense the presence of other devices that are even more closely attached to you than the Pico itself: some people find it fashionable to wear a piece of hardware attached to their nose—it could well be a Picosibling for them, and it's unlikely that they would undo it and leave it on the desk when they go to the toilet.

Ross Anderson: Then the gentleman in Oakland with the shotgun, instead of saying "please give me your phone", will say "please give me your Picosiblings, and here I have a pair of bolt cutters to assist you with removing your nose jewellery".

Reply: This is the reason why, besides the Picosiblings, I have two other items, one of which is the biometric (which might be a liability if it means the Oakland gentleman will want to chop off your finger).

Bruce Christianson: A link to your Pico-pacemaker will probably be OK.

Alf Zugenmaier: But then why do you have any siblings? I mean, you could stick everything into one, especially if they work with an n-out-of-n secret sharing scheme and thus you need to have all of them together. It's not like, if you forget your watch, your glasses will do as well; it's instead, if you forget one of these things, you are out of luck, nothing will work! Then why don't you just have it as one piece, and that can be your pacemaker, or your nostril jewellery, or whatever?

Reply: It's rather harder to have a sensible user interface on your pacemaker or your nostril jewellery than on a handheld Pico, so I think it's better if you have the Picosiblings just as a proof that you are there, and have something else as a user interaction device.

Bruce Christianson: Well, it would seem sensible to have a Pico and to have something that it has to be in the proximity of.

Reply: Exactly.

Bruce Christianson: But then they're kind of not siblings, it's a different sort of relationship.

Reply: You mean there is a hierarchy between them?

Matt Blaze: I don't want to give people an incentive to steal my pacemaker.

Bruce Christianson: They'll just wait till you go swimming.

Omar Choudary: I think in the Oakland scenario the guy comes over and demands all of them.

Reply: I have a later slide about coercion resistance, so let me show it to you now. First, when you see that Oakland guy with the shotgun, you take your nostril jewellery and you just destroy it, or even throw it far away so that it loses contact with the Pico, and then the Pico after a while switches off. But I have an additional protection against that attack: one further share comes from a network server. (It's somewhat debatable whether you do want your Pico to be re-enabled only after talking to a network server.) The biometric share and the network server share have a much longer timeout than the others: the latter may be of the order of minutes while the former may be of the order of hours or even a day, which means that once a day you have to show your eye to the Pico, and once a day the Pico has to talk to a network server.

The nice thing about a network server (your own network server) is that, if someone has stolen all these Picosiblings from you because you were in the swimming pool and they were all in your locker (even your glasses, though perhaps not your pacemaker), then you still have a chance, after having lost everything, to login to your network server and say "don't give out your share any more"; so, from tomorrow, it would no longer be possible for anybody to use your Pico.

Jonathan Anderson: Can you login to your network server though?

Ross Anderson: The point is that the semantics of the network server may be quite different. The network server is your R&R (Revoke and Re-provision) facility. If it's also vulnerable to legal coercion, and that's not just coercion by the government, but also coercion by you, then if all else fails you can go to the county court, pay the £30, and get your life back.

Reply: I like two things about the network server: first of all, it gives me the ability of revoking the Pico after I have lost control of it. It's easy to say "when you see the guy with the shotgun approach, just break your Pico, you can recover

it later from the backup"; but you can't do that if he mugs you before you realise you're being mugged. If you can revoke the Pico after he's left, and he cannot coerce you on the spot, that's more secure. The other nice thing is that the network server will keep an audit trail of any shares that it sends out, so you can see all the instances in which your Pico has been woken up from a switched-off state.

If you are worried about the paranoid threat model that, if you know the password to something very valuable, they could kidnap and torture you until you tell them the password, then with a Pico you are in a better situation because there's nothing you know or have that can help them. And if you just break the Pico, or break the connection between Pico and Picosiblings so that the Pico becomes unusable, then there's nothing you can do to let them re-enable it which they cannot already do themselves.

Jonathan Anderson: I'd rather be on the other side of this debate. If I suspect that a man is going to walk up to me in Oakland with a shotgun, I want to be in a position where I can help him, and make him feel good and go away! I don't want to be the guy who says "oh sorry, I've just broken the thing that would let you have my money" and now he's in a bad mood! I like the revoke-after-he's-taken-my-stuff story, but in fact I'm quite happy for him to take the Pico and go away.

Joseph Bonneau: I'm not sure about that.

Reply: There's a series of answers to this one, some of which I have already written up in the pre-proceedings. First of all is that (as I said when we were discussing the previous talk, in fact), if I have something that's really valuable, when I'm going to somewhere dodgy like Oakland or Afghanistan...

Matt Blaze: Can I please speak for the dangerousness of Philadelphia?

Reply: We can have an auction!

Bruce Christianson: That's W C Fields's epitaph.

Reply: If you're planning a trip to Oakland, or Afghanistan, or Philadelphia, or other similar places, you might decide *not* to take the Pico that has all your most valuable passwords. Nothing prevents you from having two or three Picos, and there's one where you just have the stuff that you plan to use on that trip, and then another one you just leave at home in your safe.

Alf Zugenmaier: You aren't allowed to write down your Picos either?

Reply: What do you mean, write them down?

Alf Zugenmaier: This way you'll get to the 37 Picos situation eventually, and then I think we are not much better off than with written down passwords.

Bruce Christianson: (You could write on each Pico which one it was!)

Reply: I don't quite buy that argument. This is just like having two security levels, with different classes of accounts in them.

Sandy Clark: But I'd argue that the common people can't really function that way: they're going to put them all together into one for convenience. It's only paranoid people like us that think about having more than one.

Reply: Well, whenever I travel I never take all my home keys. It's like Alf: if the hotel wants me to have a credit card I carry a credit card, and I will take one key just to get back home, but I'm definitely not going to take all my other keys because I'm scared of losing them.

Omar Choudary: I disagree with the claim that ordinary people are not as paranoid as us. I am not as paranoid as my father, for instance, who is a mathematician, but he will never type the number on his credit card: he's very careful about this kind of things.

Sandy Clark: Mr and Mrs Average Citizen are not bad.

Jonathan Anderson: If Mr and Mrs Average Citizen don't have anything more valuable to steal than banking credentials, well, who cares! Because they can get that back if they talk to their bank within 48 hours or so.

Ross Anderson: . . . if they're in North America.

Reply: So Sandy, what is your recommendation that would suit the average citizen?

Sandy Clark: I don't have a solution for this. I was imagining combining your idea with Bruce's and having an embedded system that was actually physically embedded, so you'd have one wallet that is your Pico that is embedded with you and you carry it around all the time. And then are you going to lose appendages every time you get attacked?

Ross Anderson: There might be something to be said, in the case of an organisation like Cambridge University, for having a Pico, perhaps an iPad with our bank authorisation software on it, bolted into a half-a-ton slab of concrete in the Old Schools, so it can only function in one precise physical location. Or perhaps we could do a better implementation than half a ton of concrete.

Reply: One of the things I rely on is the existence of a secure location for you. You can assume you're not going to be in trouble in that place: it could be your home, it could be the headquarters of the spy centre if you're a spy, or something like that. There's a place you can go back to and do things safely: that's where you do your backups, that's where you have your revocation server, all this stuff.

Joseph Bonneau: If you assume that, then you assume the attacker won't say "don't go home and disable the Pico or else I'll do <whatever bad thing> to you".

Reply: That's why I said that, before I go to Oakland, I can set the decay time for that share of the network server to be long enough to cover my trip; but if

I ever switch off the Pico, that's it, I won't be able to authenticate to anything during that trip because I won't have a second chance. And that's the price I pay: availability. In exchange, if someone mugs me, then they're stuffed, they can do nothing with it, it's as if I didn't have the Pico with me.

Mike Burmester: Did you say that, if it's physically separated from you, after an hour or so it will just die? It needs your warmth, your presence?

Reply: Yes.

Mike Burmester: So if I steal it from you, take it from you, it will die?

Reply: Yes, that's it.

Mike Burmester: A natural death?

Reply: That's part of the design, but I can revive it by having all the other Picosiblings there and by reactivating the network server.

Mike Burmester: Then it is a threat to the system because I will force you to take me to the other places.

Reply: Yes, and that is why I said: if I expect to be under that coercion threat, then I will make it impossible for me to revive it after it switches off. But in normal circumstances, when it switches off, I just have to pay the small penalty of doing some authentication dance that includes contacting the server, iris scan, Picosiblings and so on.

Mike Burmester: So how do you control the separation? What does "separation from your body" mean for Pico? What would trigger the break?

Reply: Separated would mean out of radio range of the Picosiblings, which you can define from signal strength—or, to be fancy and more secure, through distance bounding.

Ross Anderson: One of the things that I was thinking about when thinking about revocation was how does the upper class do it, because often you design services well by looking at the services that you get if you are a software billionaire or a duke, and then seeing whether we can do 90% of that for 10% of the cash. Now if you're a billionaire you'll have an account at somewhere like Coutts (there are branches here in Cambridge for the software billionaires) and you will have a young man, let's call him Rupert, who knows you by name and does your banking. Rupert knows your dogs' names and your kids' birthdays, and stuff. Now, if you get mugged in Oakland and you're a Coutts customer, it's no problem: you phone Rupert and Rupert makes the world right! Rupert will helicopter a new mobile phone, Rupert will send a messenger round from the nearest bank with $2000 in cash to tide you over, Rupert will fix your bill at the Claremont, Rupert will pick up your medical bill at the Alto Medical Centre. How close can we get to that and still make it economical?

Sandy Clark: But at that level you're not carrying any money anyway. you're relying on someone else to do all the paying.

Reply: Perhaps, if you are that rich, you wouldn't even have the Pico: Rupert would have your Pico and would do all of this kind of low level authentication for you.

Ross Anderson: Yes, but Rupert is sitting in Cambridge, you're in Oakland, you've been mugged, all you have to do is phone Rupert.

Bruce Christianson: Rupert is a server somewhere and you just have to get some message to Rupert and that's going to download all the stuff you need into some other Pico.

Reply: I think the practical issue is that, if you are at that galactic level of wealth, then you just have other minions do things for you anyway. But the philosophical problem, which is interesting, is how do you authenticate to Rupert? Because you're falling back on human authentication anyway, which of course is the thing that always works.

Ross Anderson: But you know Rupert, he comes to your parties, he buys presents for your children.

Reply: Yes, that's human authentication!

Bruce Christianson: We share the password.

Reply: I mean, it would be good if we could implement something as good as humans recognising each other in a way that worked for all this other stuff.

Ross Anderson: An interesting thing about this is if Pico can somehow recognise you, or if a Pico can be conjured into existence that would recognise you, even in a strange place. That can be a sort of Pico Rupert.

Reply: There is the old David Wheeler quote: "every computer science problem can be solved with another level of indirection". And here we're basically taking one extra step away: authentication is not between me and the app, it's between my Pico and the app; and then I have to authenticate to the Pico, and I do that through the Picosiblings and so on. And if there's Rupert, then there's yet another layer: Rupert authenticates to my Pico, I authenticate to Rupert and so on.

Jonathan Anderson: But the thing that Rupert gives you is that he is sitting in the city of London somewhere and a mugger in Oakland has absolutely no sway over him.

Bruce Christianson: No chance of getting to him at all.

Jonathan Anderson: And it's like a time-locked vault, and you tell the bank robbers: "sorry, there's nothing I can do".

Reply: Indeed. Without that extra level of Rupert, that's the whole point of my network server construction: nobody can mess with the network server, even when they're in front of me with a shotgun.

Ross Anderson: Now let me give you another example of a threat model. In São Paulo and Rio de Janeiro, if you look too rich and you go out in a bad street, then somebody will kidnap you and take to a favela, where you will be milked. They will sit you down at a PC and get you to transfer all your money to them; they'll take you around ATMs at gunpoint and get you to empty your accounts; and the police don't go there except with military backup.

Reply: Then the Pico is perfect for this. When you are getting kidnapped you destroy one of your Picosiblings: you just let it fall out of the window when they get you. And then you go there and you say "oh shit, it doesn't work, there's nothing I can do".

Ross Anderson: Boom.

Bruce Christianson: You want it to work a bit.

Reply: In your dying breath you will tell them: "you don't understand how this works!"

Bruce Christianson: In really dangerous places like New York I understand you're supposed to carry two wallets, one in your hip pocket for when you get mugged, and one somewhere else.

Alf Zugenmaier: And of course the crooks know that as well.

Bruce Christianson: Yes, but it's a social convention, there's a convention about how much it's polite to have in the hip pocket wallet.

Sandy Clark: How civilised!

Bruce Christianson: If the protocols involve the identity for the particular Pico that's being used, you could revoke from another Pico. The second Pico could just say to everyone you did business with: "if Pico number 37 comes online to you, please just give it these digits"—digits that mean to Pico number 37 "switch yourself off forever"[6].

Sandy Clark: Couldn't it incorporate the two wallet idea and have just two levels, where one set of digits gives them a little bit of money, and the rest gives you access to the remainder?

Ross Anderson: But there's a problem with that: how can you convince them that you've given them all the digits that are available? How do you get them to stop torturing you?

Matt Blaze: Fundamentally, it seems that this system works well if I can create the impression that the Pico is necessary, but in fact the Pico isn't necessary.

[6] These digits would have to be signed to prevent crippling denial of service attacks.

If I really want to get the money, I want to be able to get it, I just don't want anybody to know that I can get it.

Reply: The point is that, in order to really get the money if I don't have the Pico, I may have to go to the secure place in person, which may not be done with the kidnapper holding me at gunpoint, because when he takes me to the headquarters of the spy centre with a gun he will be stopped by people with many more guns.

Alf Zugenmaier: That's why you don't give them to your wife and children. I think for coercion resistance, you may want to have it as a limited risk, like cash: you get asked for money, you hand it over and that's it, you walk away.

Omar Choudary: You might make two sets of passwords with this website, let's say the bank website; once you get just $n-1$ of the Picosisters, your Pico issues the password for the low level, where you can get at most £100; with all n you give the other password that gives access to all.

Reply: Similar to Sandy's suggestion. I guess that the danger with all these tricks is that, if you make the solutions too institutionalised, then the bad guy will know; then, as Ross said, they'll say: "OK, that was the first level; now open up the second level or I keep on torturing you".

Ross Anderson: But institutions sometimes help. There's no point in you demanding my house at gunpoint because surely you would have to sign the transfer document as well, and the social conventions are that you have estate agents, and searches, and you can't even sell a house nowadays without an energy certificate. I can't lawfully sell you my house even if Shireen and I both wanted to, until you've got some geezer crawl over it and count the radiators. Does that not add some value?

Reply: Isn't that in the same spirit as having that server in a bunker somewhere that nobody can mess with?

Matt Blaze: But Ross, I can hold you at gunpoint until you give me as much money as your house is worth.

Ross Anderson: Then I would have to raise a mortgage against it which would require Shireen to attend at the Cambridge Building Society and sign various documents.

Matt Blaze: Well I'm sure there are phone calls that you can make that would persuade Shireen to do this. Like: "help, I'm being held at gunpoint!"

Bruce Christianson: But one of the effects of protocols is just to slow things down.

Jonathan Anderson: I just worry that (aside from the property of not having to type passwords any more) strengthening the coercion resistance properties of the Pico to this degree is only useful against attackers who are already so

committed that they'll do whatever it takes to get around even this level of coercion resistance.

Joseph Bonneau: I think we're imagining attackers that are much more powerful than actually exists in Oakland. Or Planet Earth.

Bruce Christianson: I think alien abduction is the real threat.

Sandy Clark: The probes will find out where your Pico is.

Joseph Bonneau: Criminals aren't stupid and they know the difference between the charges that they'll take for different attempted crimes. Mugging won't get them nearly as much as going to your house with a gun and kidnapping somebody. And we have to push them into things they don't want to be doing.

Jonathan Anderson: Against mugging, you don't need an n-out-of-n threshold sharing scheme blah, blah, blah; all you need is: it times out.

Bruce Christianson: Joe's is a very good point, and it's the same one Ross made this morning: a good countermeasure is to raise the stakes for the attacker.

Ross Anderson: So we bring in the death penalty for muggers.

Bruce Christianson: Yeah!

Jonathan Anderson: Well no, because then you have the perverse incentive that there used to be: if you're going to steal bread, you might as well kill people, because the penalty is the same.

Bruce Christianson: But there's also going to be the death penalty for littering. So, as soon as you drop that earring out of the taxicab, you're gone!

Virgil Gligor: OK; on that note, perhaps we should go on to the second talk.