The following paper was originally published in the
Proceedings of the Sixth USENIX UNIX Security Symposium
San Jose, California, July 1996.

Establishing Identity Without Certification Authorities

Carl M. Ellison
CyberCash, Inc.

Establishing Identity Without Certification Authorities

Carl M. Ellison*
CyberCash, Inc.

Introduction

> Without a KMI[1] of trusted certificate authorities, users cannot know with whom they are dealing on the network....[3]

It is commonly assumed that if one wants to be sure a public key belongs to the person he hopes it does, he must use an identity certificate issued by a trusted Certification Authority (CA). The thesis of this paper is that a traditional identity certificate is neither necessary nor sufficient for this purpose. It is especially useless if the two parties concerned did not have the foresight to obtain such certificates before desiring to open a secure channel.

There are many methods for establishing identity without using certificates from trusted certification authorities. The relationship between verifier and subject guides the choice of method. Many of these relationships have easy, straight-forward methods for binding a public key to an identity, using a broadcast channel or 1:1 meetings, but one relationship makes it especially difficult. That relationship is one with an old friend with whom you had lost touch but who appears now to be available on the net. You make contact and share a few exchanges which suggest to you that this is, indeed, your old friend. Then you want to form a secure channel in order to carry on a more extensive conversation in private. This case is subject to the man-in-the-middle attack. For this case, a protocol is presented which binds a pair of identities to a pair of public keys without using any certificates issued by a trusted CA.

The apparent direct conflict between conventional wisdom and the thesis of this paper lies in the definition of the word "identity" – a word which is commonly left undefined in discussions of certification.

*`cme@cybercash.com`

[1]Key Management Infrastructure – a hierarchy of Certification Authorities

Identity

To some people, an "identity" is just a name. To others, it is the data needed to allow one to track down, arrest and punish the violator of a contract or the passer of a bad check. In the case of an old friend who shows up on the net, "identity" closely follows the dictionary definition:

> Identity: 2a. the distinguishing character or personality of an individual.[7]

To the person verifying the identity of an old friend, that other person's identity is a body of memories: an internal representation of the distinguishing character or personality of an entity, as the verifier has come to know that entity through the relationship between them. That body of memories is labeled by a name but the name is not an identity. It is a label for one. It is also neither unique nor global[2].

Each human being is forced to name people in order to think about them. However, people typically use short internal names, of the form: "IBM", "CyberCash", "Rocky", "Alan", "Zorak", "Lolly", "Red", etc. To the person using such a name, the name is unique. To that person, no other qualification is required. However, only the first two come even close to being likely to specify the same entity to everyone else. Even in those cases there is no guarantee. Unlikely though this might be, there *could* be an "Idaho Boiler Mechanics, Inc." of special importance to someone. It is also possible for someone to change his or her name completely – perhaps to start a new life and discard psychological baggage. The identity of that person does not change but his or her generally recognized name does while someone's internal name for that person may or may not change accordingly.

Identity Certificates

The word "certificate" has been used so long to mean "identity certificate" that many people in the field of

[2]One recent attempt to deal with this fact of life can be found in the SDSI[4] proposal of Rivest and Lampson.

cryptography find it difficult to consider any other kind. For the purpose of this discussion, let us start with two definitions:

Definition 1 (Certificate) *A certificate is a digitally signed, structured message which delegates an attribute of some form to a public key.*

Definition 2 (Identity Certificate) *An identity certificate is a certificate which binds the name of an entity to a public key. Its putative meaning is to delegate all the attributes of the named entity to the public key.*

The attributes most commonly considered are **trust** and **authority**. Those words, too, are frequently used without definition when they should be defined, but for the purpose of this paper, each definition applies equally.

There are two commonly used examples of identity certificate at this time: X.509 and PGP.

X.509

X.509 certificates grew out of the X.500 global database design. Under X.500 all entities in the world (not just people) would have unique names, called Distinguished Names (DNs). These names are organized in a hierarchy, so that the name space can be distributed both for storage and for management.

Given a global, unique name for every entity on the planet, when one wanted to bind a public key to an entity, what better than use that unique name? X.509 was the result.

There are some problems with X.509. Two are relevant to this paper. The first is that X.500 has never been deployed worldwide and is likely never to be. The second is that even if some large corporation were to deploy X.500 internally and generate X.509 certificates using those DNs[3], it would make those DNs unique by addition of information of relevance to the corporation, such as operational unit, building name or mail stop. Such a unique name is not adequate to disambiguate a common name from the point of view of all possible users of the certificate. That is, Alice may have an old friend, Bob Jones, at IBM but have no idea of his mail stop, building name or operational unit. Although she has a fully trusted identity certificate from IBM for each employed Bob Jones, she can not trust any of them to be her friend. Therefore, the certificate has failed in its task of binding identity to a public key.

[3] Some organizations may build an X.500 name space solely for the purpose of creating X.509 certificates rather than for X.500's original purpose of providing a global directory service.

PGP

PGP, actively shunning the rigid hierarchy of X.509, permits each user to "certify" a binding between a public key and a UserID. By convention, the UserID consists of a common name and an e-mail address. The e-mail address serves to make the UserID unique, just as the X.509 DN is made unique.

The PGP UserID has one advantage over an X.509 DN, in that an e-mail address is an element of a currently deployed global name space. However, PGP key signatures have a disadvantage compared to X.509 certificates in that the person signing the key is not necessarily either known or trusted. PGP attempts to compensate for that by allowing multiple, presumably independent signatures to vote a binding into validity.

Other Certificates

There are some certificates which are sometimes considered to be identity certificates, possibly because the word "certificate" has traditionally been taken to imply an identity binding.

Driver's license analogy

It is common practice for a merchant accepting a paper check to ask for a driver's license for identification and to write the license number on the check. A driver's license is a physical certificate: an instrument, issued by a large, trusted organization, which binds a person's picture and physical description to his signature. That binding is relatively loose, however. The Maryland driver's license, for example, uses a digitized signature written on a data tablet which is difficult to compare to a normally written signature.

In the digital world, this would be a certificate which binds an image, birth date, height, weight and gender to a signature key. Such a certificate would be of value not only for giving meaningful evidence of identity but also for on-line dating services. It would have to be issued by a trusted third party – one trusted not to lie to the verifier of the certificate.

There is another binding provided by a driver's license. The Department of Motor Vehicles is sure to keep track of licensed drivers. In the event that a check is returned because of insufficient funds and the person passing it fails to respond to appeals to rectify the situation, the merchant can hope to employ the DMV's database to track down the person and institute legal action against him or her.

This kind of certificate also needs to be issued by a trusted third party. Specifically, that party needs to attest to the ability to locate a person in the event of criminal behavior.

Neither of these certificates is properly an identity certificate[4]. Neither needs to include a person's true name, for example. A certificate with a picture and data about a person could be issued by a dating service and have only a nickname and 1-900 telephone number to identify the person. A certificate attesting to the ability to find a person in case of misdeed could have only a serial number, to be used by the tracking service after it has been paid its fee.

Certificates for electronic commerce

One can perform electronic commerce without identity certificates at all. Certificates are needed but they are needed only to grant authority to a signature key to spend money from a given account. For example, the SET[5] certificate binds an account number to a signature key. Any card-holder name which might also be in the certificate is of no importance in the process of deciding whether to honor a given purchase request.

Specifically, if someone accepts a paper check, that check has a printed name and address as well as a signature and amount. The person depositing the check does not care about either the name or signature. He cares only about whether his bank account is augmented by the amount of the check. The person's bank (the *accepting bank*) cares only about whether the *issuing bank* transfers funds in return for the check. Both care that there is a signature, but neither can verify it. The issuing bank checks the signature and checks that there is money in the indicated account but it, too, ignores the name and address printed on the check.

In cyberspace, the same applies – with the exception that the person accepting an electronic check is able to determine whether the signer of the check is authorized by the issuing bank – given the proper certificates. There is still some trust involved that the indicated account has funds, but there is always that issue unless one has on-line clearing. To determine validity of an electronic check, one needs to verify:

- the digital signature on the check
- a certificate from the issuing bank giving check-writing authority to the signature key
- a certificate from the accepting bank, acknowledging the issuing bank's signature key as belonging to a bank with which it deals (or more likely a chain of certificates through various clearing houses)
- a certificate issued by the person accepting the check (the ultimate authority) verifying the accepting bank's key. That certificate would be generated when the person opened his electronic checking account.

In none of those certificates does a name or other generally recognized mark of identity appear – although each party may file the certificate under a local name.

Local and global names

The internal name each human is forced to use in order to think about a person can be thought of as a local name, as opposed to a global name in the form of an X.500 DN or a fully qualified Internet e-mail address. Even the DN in an X.509 certificate can be thought of as a name local to the CA which issued the certificate.

Local names – nicknames – are both unique and meaningful, but only to the one owner of the local name space.

For a (name,key) certificate actually to bind a key to an identity, there needs to be a secure mapping from the name space of the issuer to that of the verifier. Since the issuer and verifier are free to change their name spaces at will, the two should be linked for periodic update. The simplest way to achieve that linked mapping is for the issuer and the verifier to be the same person. If the verifier makes his own certificates, he can use his own choice of names in those certificates. However, that requires him to assure himself of the validity of the binding without relying on certificates from a third party.

The preferred means for performing that verification depends on the relationship between the subject and the verifier.

Relationships

A verifier can issue his own certificate for someone's public key, binding the key to the verifier's name for that person, in one of a number of ways, depending on the relationship between the two parties: personal contact; major corporation; network acquaintance; stranger; or old friend, out of touch.

[4] The picture certificate could be of use in binding identity of an old friend unless that friend is old enough to have become unrecognizable.

[5] Secure Electronic Transactions – a standard for electronic commerce being developed jointly by VISA and MasterCard

Throughout this section, it must be kept in mind that the certificates being issued are for the verifier's use only. The names involved are the verifier's own nicknames assigned to the corresponding entities. This certificate structure can be extended through SDSI[4] to affect other people, but under no circumstances are the certificates mentioned here pretending to deal in universal, global names. To the contrary, it is clear that there is no such thing as a universal, global name space with names meaningful to all possible users and there never will be. There are too many names for one human being to remember and attach meaning to each one of them.

Personal contact

If the two parties are in personal contact periodically, they can exchange their public keys in person (or alternatively, they can exchange secure hashes of their keys). There are three variants of this exchange, however, and the certificate verifier needs to keep them in mind and note which form of exchange was employed for a given certificate:

- **Alice gives her key to Bob**

  Bob is sure Alice claims to own the key he is about to build into a certificate. As long as that key is used only for confidentiality, Alice would gain nothing by lying about ownership of the key. Similarly, if the key is for signatures but the only thing being signed is a key exchange for confidentiality, Alice would have no reason to lie.

  However, if the key were used to sign lab notebooks, later to be used in patent applications, for example, Alice might have a reason to claim ownership of a key she can not actually use.

- **Alice gives her key to Bob; Bob gives a challenge to Alice**

  This variant can take place over an extended period of time. During the personal meeting, Bob gives a challenge value to Alice to take home and process. Later, Bob receives an electronic communication from Alice completing her half of the protocol. Bob gains all the assurance in this case which he received in the previous but holds off generating a certificate until Alice responds correctly to the challenge.

  If Alice's is a signature key, Alice demonstrates the ability to sign the random challenge which had been given to her in the clear. If Alice's is a confidentiality key, Bob hands Alice a message with the random challenge encrypted under her key and generates a certificate only after Alice returns that challenge value.

  In this variant, Bob is assured that Alice has access to the private key, although that access might be by duping the true owner of the key into acting as an oracle for Alice.

- **Alice gives her key to Bob; Bob witnesses Alice's processing of his challenge**

  In this case, Bob is assured of everything the previous variant assured him and also knows that Alice is in true possession of the private key. For the truly paranoid, Alice can be sent into a sealed room to perform the signature or decryption of the challenge.

These personal exchanges are appropriate not only for employee/employer relationships where the employer might run a traditional CA but also for individual acquaintances and relatives. The personal meeting can be in the flesh or it could be by telephone or video conference, depending on the extent to which the verifier trusts those connections.

Major corporation

A major corporation or other entity with access to broadcast media (radio, TV, satellite, newspaper, magazine) which is difficult or impossible to intercept and modify, can publish its key in an advertisement, binding it to the corporate name. A verifier can trust this binding to a certain extent on the assumption that the corporation will perform a random sampling of the broadcast and detect any tampering. This has little value however if the verifier is being targeted specifically for spoofing[6].

For this relationship, a traditional certificate hierarchy provides a benefit. However, if a verifier is being targeted, the attacker can presumably substitute a root certificate and certificate chain. Anyone wanting to provide substantial assurance via certificate hierarchies in this case needs to assure the delivery of the correct root key to every individual verifier.

Fortezza/Tessera provides that assurance by storing the root key at time of card programming and prohibiting later modification of that key.

A less heavily structured option is to gather some keys for yourself in person, by visiting the companies in question, and then passing your knowledge around to your friends, in the manner of the PGP web of trust, assuming of course that SDSI[4] naming is employed.

[6] Even a radio or TV broadcast can not be trusted totally, as the old "Mission Impossible" television series was fond of pointing out.

Network acquaintance

Perhaps the easiest person for whom one can generate a certificate is a network acquaintance. Alice has never met Bob in person and likely never will. She got to know him via the network – exchanging e-mail or, someday, digital voice or video.

Provided all of these exchanges were digitally signed in the same key (or encrypted under the same confidentiality key), Alice can immediately generate a certificate tying her name for Bob to that key. In fact, she can generate that certificate the first time she encounters the key.

This requires no mathematics. It relies on the definition of "identity".

A public key is a surrogate presence in cyberspace for some entity in physical space. It acts directly in cyberspace, just as the associated entity can act in physical space.

The public key is bound tightly to the physical space entity, assuming that the latter maintains physical possession of the associated private key and sole knowledge of the pass phrase under which it is encrypted. Once there are devices which actively monitor living contact with the proper thumb, for example, instead of merely accepting a pass phrase or PIN, the bond between key and person will be even stronger.

If the bond between key and person is broken, no layer of certificates will strengthen it. On the contrary, in this case certificates merely provide a false sense of security to the verifier.

Alice can therefore generate an immediate local-name certificate for this person. She can not testify to that person's global name. She can not know even if he's a dog, much less what gender. However, to the extent that she knows a person, she knows the cyberspace surrogate for that person and in cyberspace, the person's key is the final authority.

This comes back to the word "identity". Alice's total knowledge of Bob – and therefore Bob's total identity, from Alice's point of view – is derived from digital communications strongly tied to a public key. That key is therefore tied more strongly to this person's identity (personality; operation of his mind) than would be a key presented by someone in the flesh.

Alice can issue her certificate for Bob even before reading Bob's first signed message. She can generate that certificate and choose her own nickname for this new person at will. The real person is yet to reveal himself but even with no information, everything Alice knows about Bob (namely an as-yet unread message) is digitally signed in the same key and Bob's total personality is still bound to that key, from Alice's point of view.

Stranger

A stranger provides perhaps the biggest challenge to the thesis that no global identity certificates are needed. There is not even any shared knowledge with which to perform one of the protocols given below and build a local identity certificate.

On the other hand, as the network acquaintance example showed, this might be the easiest case. If someone is a total stranger, then there is no presumed relationship and therefore no trust and therefore nothing to risk by binding his key to a still-meaningless nickname. All trust of a network acquaintance will be acquired over time through digitally signed interactions so the verifier has assurance that all these interactions came from the same person.

Let us differentiate the stranger case from the network acquaintance artificially. Let us assume that we need to extend some trust to this stranger without building a relationship first. The stranger might be a shopper, wanting to spend digitally signed checks on our web site. The stranger might want to gain FTP or telnet access to an ISP's computer. The stranger might want physical access to an electronically controlled door.

In all of these cases, what is needed by the verifier is not an identity certificate. The person being certified is by definition a stranger and therefore his or her identity is meaningless to the verifier. Rather, what the verifier needs is an **authorization certificate**: a certificate which authorizes the holder of a key (therefore, authorizes the key) to spend money from a given bank or charge account, to get access to a given FTP server, etc. These certificates need to be issued by an authority with the power to delegate the authorization in question – a bank, a corporation owning the FTP server, etc. However, these are not identity certificates and this paper will leave the issue of authorization certificates to be addressed elsewhere[2].

Old friend, out of touch

Various protocols are presented later in this paper for using shared knowledge between the two parties involved mutually to verify each other's identity and bind those identities to their respective public keys. Because these depend on shared, common knowledge, each person is assured of binding a key to an identity by the dictionary definition given earlier – that is, to a distinguishing character or personality.

These protocols are complicated by the need to prevent various attacks, some due to the low entropy of individual pieces of shared common knowledge. This is especially a concern because the case of an old friend who shows up on the net is tailor-made for the "man in the middle" attack.

Man In The Middle Attack

The Man In The Middle attack assumes that there are two people, Alice and Bob, who hope to establish a secure communication channel. Sitting between them and controlling the only physical communications channel connecting them is an active eavesdropper, Mallet, who is capable not only of intercepting all traffic but also of deleting, inserting and modifying messages en route.

Alice and Bob try to establish a secure channel by sending an RSA public key to each other over this insecure channel or by engaging in an exponential key exchange protocol such as Diffie-Hellman.

RSA

Using RSA, Alice and Bob each generate an RSA key pair and each transmit their public key to the other. If there is no active eavesdropper, then all other eavesdroppers are foiled.

1. A→: (Alice transmits) K_A, Alice's public key
2. B→: K_B, Bob's public key
3. →B: (Bob receives) K_A
4. →A: K_B

At this point, Alice and Bob have each other's public keys and can use them for a normal secure mail interchange. Alternatively, one of them can generate a key for a symmetric link-encrypting cryptosystem and transmit it to the other, after which they can use that key and have an open secure channel.

With the active eavesdropper, Mallet, in the channel, the situation changes. Mallet generates two RSA key pairs, with public keys K_{MA} and K_{MB}. Alice and Bob engage in the same protocol listed above, or so they think, but in fact:

1. A→: K_A
2. B→: K_B
3. →M: K_A
4. →M: K_B
5. M→B: (Mallet transmits to Bob) K_{MA}
6. M→A: K_{MB}
7. →B: K_{MA}
8. →A: K_{MB}

Now, Alice and Bob believe that they have each other's RSA keys, but instead they have Mallet's versions of each other's key. For all future traffic, whether plaintext or a symmetric key, Mallet will intercept each message, decrypt it, possibly modify it and re-encrypt it in the appropriate destination key. Alice and Bob are unaware of this interference, except possibly for performance anomalies.

In other words, Mallet has created two secure channels, one between himself and Alice over which he impersonates Bob, and the other between himself an Bob over which he impersonates Alice. This general attack applies no matter what public key exchange mechanism is employed.

Diffie-Hellman

Using Diffie-Hellman, Alice and Bob determine a new secret quantity, allegedly known only to the two of them. In the normal protocol:

1. Alice and Bob agree to a large prime, p and a generator g
2. Alice generates a random, secret quantity, x_A
3. Bob generates a random, secret quantity, x_B
4. Alice computes $y_A = g^{x_A} mod p$
5. Bob computes $y_B = g^{x_B} mod p$
6. A→: y_A
7. B→: y_B
8. →B: y_A
9. →A: y_B
10. Alice computes $z = y_B^{x_A} mod p$
11. Bob computes $z = y_A^{x_B} mod p$

With Mallet in control of the raw channel, however, he gains access by forming two independent secure channels – one between himself and Alice and the other between himself and Bob. He then translates all messages between them, modifying some as necessary. For example, Alice and Bob may try to verify security of the channel by transmitting the resulting z or a secure hash of it to each other. Mallet would need to modify the content of such messages, in order to continue his deception[7]

Protocol for Binding Identity to Keys

Let there be Alice and Bob, old friends who want to communicate securely. They have only one channel between them and it is possible for Mallet to intercept and modify all messages between Alice and Bob. Alice and Bob can perform the following protocol in order to determine whether Mallet has captured their secure channel. If they determine that he has, then they record the shared secret knowledge which they had revealed to Mallet and mark it as suspect. If they determine that the channel is truly secure, then the shared secret knowledge which they used to verify each other's identity and the security of the channel can be reused in the future – and additional shared knowledge can pass between them for possible future use.

1. Alice generates a key pair and sends the public key, K_A, to Bob.
2. Bob generates a key pair and sends the public key, K_B, to Alice.
   (a) If Mallet has captured the channel, he generates keys and sends his keys on to Alice and Bob – sending K_a instead of K_A to Bob, K_b instead of K_B to Alice. In this case, Alice has keys (K_A, K_b) while Bob has keys (K_a, K_B). The allegedly secure channel they create with these keys includes Mallet, able to read all traffic and also to modify it.
   (b) If Mallet is not present in the channel, then Alice and Bob both have keys (K_A, K_B) and the secure channel they create with those keys is private to them.
3. Alice and Bob engage in one or more rounds of protocol. Each round returns either "failure" or an entropy value, E. Further rounds are executed until there are K failures or the sum of returned E values exceeds a security parameter S.
4. If K rounds failed, the protocol failed. Alice and Bob conclude that the channel may have been compromised by Mallet.
5. If $\sum E > S$, the protocol succeeded. Alice and Bob are assured that the probability that Mallet has captured the channel is less than 2^{-S}.

Once the channel has been verified secure with Alice and Bob having proved to their own satisfaction that the other end of the channel holds their old friend, a side effect of the protocol is that Alice and Bob can issue personal identity certificates for their old friend's key – either the key used to create the channel or a signature key exchanged over that secure channel.

Protocol round

For each round of the protocol, Alice and Bob use the presumably secure channel they have set up with the keys they exchanged. They could be correct in assuming that channel to be secure and private to the two of them or there could be an active eavesdropper, Mallet, who has created two secure channels – one between himself and Alice and one between himself and Bob. Each protocol round presents a test to the person on the other end of the immediate secure channel – either the desired party or Mallet. The protocol uses interlock in order to attempt to insure that only the person on the end of the immediate secure channel can provide answers. Alice therefore tests either Mallet or Bob to determine which one he is.

Over the secure channel:

1. Alice and Bob each formulate and transmit a question which only the other should be able to answer, such as, "what was the first name of the woman you kept telling me about in June, 1979?" R_A and R_B are Alice's and Bob's answers, respectively, to each other's questions.
2. Alice and Bob each list the possible answers the other might give, accounting for spelling variations. Alice forms the list $X_i, 0 < i \leq I$, of answers she would accept from Bob and Bob forms the list $Y_j, 0 < j \leq J$, of answers he would accept from Alice. (Each has presumably chosen a question designed to minimize the number of acceptable answers.)
3. Alice computes $U_i = H(X_i|K_A|K_1)$ and $T_A = H(R_A|K_1|K_A)$ where H() is a cryptographically strong hash and K_1 is the key Alice believes belongs to Bob.
4. Bob computes $V_j = H(Y_j|K_B|K_2)$ and $T_B = H(R_B|K_2|K_B)$, where K_2 is the key Bob believes belongs to Alice.

[7] A secure telephone can present problems to Mallet in this desire. He would be forced to be able to imitate the voices of the participants if he wanted to change any messages. Assuming the parties involved knew each other's voices, this could be significantly difficult.

If there is no eavesdropper and if $R_B \in \{X_i\}$ and if $R_A \in \{Y_j\}$, then $T_B \in \{U_i\}$ and $T_A \in \{V_j\}$.

If there is an eavesdropper who has replaced either K_A or K_B then the probability that $T_A \in \{V_j\}$ is roughly $J/2^S$, where $S = \min(|H|, |K|)$. Since $|K| > |H|$ in common practice, this probability becomes $J2^{-|H|}$, for large $|H|$ and small J, reflecting the assumption that one is unable to find a collision for a cryptographically strong hash function with probability greater than $2^{-|H|}$.

5. Alice and Bob exchange answers T_A and T_B, using an interlock protocol detailed below. If the received $T'_A \in \{V_j\}$ and $T'_B \in \{U_i\}$, then the round is considered successful. Otherwise, the round has failed. (Failure of the round might be detected in the middle of the interlock protocol, in which case the interlock exchange is terminated early, depriving Mallet of information.)

6. If the round succeeds, Alice computes the entropy, $E_{A,i}$ of answers X_i, given the transmitted question, and computes

$$P_A = \sum_{i=1}^{I} 2^{-E_{A,i}}$$

and $E_A = -\log_2(P_A)$, assuming $I \ll 2^{E_A}$. Bob computes E_B similarly. Alice and Bob exchange their E_z values and each computes $E = \min(E_A, E_B)$. That value, E, is the numeric result of the successful round. The individual entropy computation must be conditional on all demonstrated knowledge so far in the protocol. That is, if Mallet has succeeded in guessing prior question/answer pairs, then the fact that the protocol continues gives him confirmation of his guess(es).

The probability of a successful guess by Mallet is, to a first order approximation, 2^{-E}. Since these entropies are computed conditional on previous rounds, the sum, X, of E values from all rounds so far gives the the probability of an eavesdropper's guessing the whole protocol so far to be 2^{-X}.

Answer entropy magnitude

Common-knowledge answers which are short enough to type uniquely (or nearly so) include names of people as possibly the simplest class. One sample of names from the employee list of a medium size company was analyzed to get an estimate of the entropy of such common knowledge answers. That analysis produced the results in the table below:

Items	N	$\log_2(N)$	Entropy
People	2507	11.292	
Full names	2501	11.288	11.287
Last names	2012	10.974	10.796
First names	802	9.647	8.376

Interlock protocols

If Alice and Bob were merely to transmit T_x to each other, Mallet would receive each. Since we are dealing with low entropy individual answers, Mallet could perform a dictionary attack, given T_x, and determine R_x. Given R_x, he could form the correct response, using keys he had provided to the other party, and allow the protocol to succeed.

Rivest and Shamir designed and published an interlock protocol[5] to prevent such attacks. Instead of transmitting the entire message, Alice and Bob each transmit one half of the message[8], waiting to receive the first half before transmitting the second half. Unfortunately, this works only when large entropy secrets are exchanged. If the answers are of small entropy, as in our case, Mallet can do a dictionary attack based on half the message as easily as on the whole message.

The interlock protocol can be blinded. Torben Pedersen[6] proposed one such scheme. If the secret is a, one chooses a high-entropy random u and computes $x = g^a h^u \bmod p$ where p is prime, g and h are generators of the group mod p, and $\log_g(h)$ is unknown. The first exchange carries x while the second exchange carries (a, u).

As a variant on Pedersen's interlock, one can use $x = H(a|u)$ where $H()$ is a presumed one-way function (e.g., a strong cryptographic hash).

Bellovin and Merritt[1] have presented an attack on the Rivest and Shamir interlock protocol which permits access to one side of the conversation and, if shared secrets are re-used, eventually to both sides. That attack works as well for the Pedersen commitment mechanism.

If communication is cheap compared to computation, there is a variant of the interlock protocol which is less vulnerable to the Bellovin and Merritt attack. Alice and Bob can release T_x to each other one bit at a time. Under this interlock protocol, once an accumulating partial result no longer matches any

[8] assuming the message is a single unit, such as a single RSA encryption or a single hash function output block

of the legal possible results, the protocol round is terminated in failure.

Probability of guessing a round

Using the basic Rivest-Shamir interlock protocol, it is possible for Mallet to perform a dictionary search over all possible answers, compute the hash for each and match half the resulting hash to the one received. The probability of collision, that two different dictionary entries would result in the same hash value, is effectively 0 assuming $E \ll |H|$, so that Mallet's probability of success is nearly 1.

Using Torben Pedersen's interlock, a dictionary attack is thwarted through blinding by the unique random value, u. Mallet can still perform the Bellovin and Merritt attack. Assume in the example below that of the two challenges offered, Bob's is the easier for Mallet to guess and therefore has the lesser entropy.

1. Alice generates u_A at random to mask secret a_A and computes $x_A = g^{a_A} h^{u_A} mod p$
2. Bob generates u_B at random to mask secret a_B and computes $x_B = g^{a_B} h^{u_B} mod p$
3. A→: x_A
4. B→: x_B
5. →M: x_A
6. →M: x_B
7. Mallet generates u'_B at random to mask secret a'_B (derived from his guess for Bob's answer combined with the keys Mallet has established with Alice) and computes $x'_B = g^{a'_B} h^{u'_B} mod p$
8. M→A: x'_B
9. A→: (a_A, u_A)
10. →M: (a_A, u_A)
11. Mallet performs a dictionary attack on a_A in order to learn the underlying secret and uses that secret to form x'_A to send on to Bob.
12. Mallet and Bob finish authenticating one another
13. M→A: (a'_B, u'_B), which will be accepted by Alice with probability 2^{-E_B} where E_B is the entropy of Bob's answer.

If $E = \min(E_A, E_B)$, the probability of Mallet's success is 2^{-E} using bit-at-a-time interlock, just as it was under the Pedersen interlock. That is, Mallet chooses one of the two answers to guess and constructs his reply accordingly. If he guesses correctly, then he can learn from that participant the full answer for the other participant – through a dictionary attack on a. He will guess correctly with probability 2^{-E}. Once he has the correct answer from one participant, he can engage in the protocol with the other participant.

However, should Mallet fail to guess correctly, the round will have failed and he will have learned only a limited number of bits of a legitimate answer. This would ease his completion of the other side of the protocol, but that round has failed and he has no purpose for finishing the other side. The note in the previous section that this variant is less vulnerable to the Bellovin and Merrit attack refers to this reduction of the information leakage to Mallet.

Adjustments to entropy

It is envisioned that this protocol would be used only once, when old friends meet on-line and share their first signature key. Assuming they were successful in having a secure channel, they can rest assured that the small secrets they exchanged to verify one another's identity and the security of the channel remain their secrets.

However, if they discover that the protocol failed, they must assume that Mallet has intercepted some number of secrets (or bits about secrets, if the bit-interlock protocol was used), and record the compromise of those secrets, adjusting their entropy (to 0, for cases where Mallet correctly guessed or appeared to guess; reduced by the number of released bits, for cases where Mallet failed). Those adjusted entropies must be used for any subsequent execution of this protocol.

References

[1] Bellovin and Merritt, "An Attack on the *Interlock Protocol* When Used for Authentication", IEEE Trans. Inf. Theory 40:1, Jan 1994.

[2] Ellison, "Generalized Certificates", manuscript, http://www.clark.net/pub/cme/html/cert.html

[3] McConnell and Appel, "Enabling Privacy, Commerce, Security and Public Safety in the Global Information Infrastructure", report of

the Interagency Working Group on Cryptography Policy, May 12, 1996; [quote from paragraph 5 of the Introduction]

[4] Rivest and Lampson, "SDSI – A Simple Distributed Security Infrastructure", manuscript, http://theory.lcs.mit.edu/~rivest/sdsi.ps

[5] Rivest and Shamir, "How to Expose an Eavesdropper", CACM, Vol. 27, April 1984, pp. 393-395.

[6] Torben Pryds Pedersen, "Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing", Advances in Cryptology - CRYPTO '91, LNCS 576, pp. 129-140.

[7] *Webster's Ninth New Collegiate Dictionary*, Merriam-Webster Inc., Springfield MA, 1991.