Schneier Chapter 4: Intermediate Protocols

Chris Brown
University of Rochester Computer Science Department

August 14, 2001

There are lots of topics in this chapter. We shan't cover them all. They include timestamping, subliminal channels (cool!), undeniable digital signatures, group signatures, computing on encrypted data, or hiding information from an oracle (cool!), committing to predictions (say stocks you believe will rise) without revealing what they are AND without the possibility that you can lie later about your prediction. I guess we have to get into *bit commitment to get into fair coin flips and mental poker.

1 Bit commitment with symmetric cryptography

- Bob generates random string R sends to alice: R.
- Alice creates message consisting of the bit she wants to commit to, b (and in fact it can be several bits), and B's random string. She encrypts with random key K and sends result to bob: $E_K(R, b)$.
- This is her commitment: B can't encrypt, so he doesn't know b.
- NOW it is time for A to reveal her bit: A sends K to B.
- B decrypts the message to get b and he checks R to make sure that this is the bit associated with his request.

Why does the protocol have A send R back to B? without it, she can secretly decrypt the message with different keys until she finds a key that yields the desired result (say a 0 instead of a 1 or a 01 instead of 10). Very likely she can find such a key for a few bits, very unlikely if she has to reproduce all of R.

OK here's another. There is a protocol using one-way functions that doesn't require B to send a message to get things started, but... This next one uses (pseudo) random numbers

- B genrates random string and sends to A.
- A generates a random seed for a pseudo-random bit generator. Then, for every bit of B's random bit string, she sends him either a) the output of her generator if his bit is 0, or b) the XOR of the output of the generator and her bit if Bob's bit is 1. Call this message M.
- NOW it is time for her to reveal her bit. She sends B her random seed and the bit.

- B repeats the calculation she did to create M; if his result matches M she's telling the truth.

A cannot cheat if the numbers and the generator are random enough.

So finally we can define a *blob as the strings A sends B to commit to a bit. Blobs have four properties:

- Alice can commit to blobs, by which she is committing to a bit.
- Alice can open any blob she's committed to. In so doing she can convince Bob of the value of its bit....i.e. she canNOT choose to open a blob as a 0 and also open it as a 1.
- Bob cannot learn how A opens blobs she's committed to, even after she has opened other blobs.
- Blobs do not carry other information than the committed bit: they are not correlated with any other info A wants secret from B.

2 Fair Coin Flipping

One way to do mental flipping. Same as XOR of two random bits. B decides on his bit at random,,writes it down, puts in envelope. A announces her bit, they open the envelope, they're done. The envelope is a commitment blob.

So...Blum's scheme:

- A commits to random bit by one of previous methods.
- B guesses which it is (announces what he thinks A's bit is).
- A reveals bit: B wins if he guessed correctly.

In general,

- A must flip before B guesses
- A can't change or re-flip after hearing B's guess
- B must not know how coin landed before he guesses.

3 Coin Flips with One-Way Functions

A and B agree on one-way function $f(x)$.

- A chooses random number x and computes $y = f(x)$.
- A sends y to B.
- B guesses whether x is even or odd, sends his guess to A.

- if B is correct, result of coin flip is heads. If B is wrong, it's tails. A announces the result and sends x to B.
- B confirms that $y = f(x)$.

Important that Alice can't find x and x' such that x is even, x' is odd, and $y = f(x) = f(x')$, else she can win every time. Also the least significant bit of $f(x)$ must not be correlated with x. If $f(x)$ produces even results more than half the time when x is even, now Bob has an advantage. (In fact it might be best not to use the least significant bit).

You can do coin flips with public key cryptography or symmetric key cryptography, but let's generalize it to mental poker. We need a *commutative encryption scheme, such that $D_{K_1}(E_{K_2}(E_{K_1}(M))) = E_{K_2}(M)$. This is not generally true for symmetric systems , but can be for RSA ciphers with identical moduli, for instance.

4 Mental Poker

A makes and encrypts 52 messages, one for each card. Bob chooses 5 of them at random, encrypts them with his public key and sends them back to Alice. A decrypts the messages and sends them back to Bob, who decrypts them (with his private key) to determine his hand. Bob chooses 5 more messages at random and sends them back to Alice as he received them. She decrypts, and they become her hand. Repeat procedure for additional cards dealt. At end A and B reveal cards and keys so that each can be assured other did not cheat.

5 One-Way Accumulators

How identify members of your covert group? Could carry a membership list around, but that's bulky and risky. Digitally signed ID cards? Needs trusted card-issuer. The *one-way-accumulator is like a one-way hash function except that it is commutative. So it returns the same number no matter what order its inputs are given to it.

So Alice computes the hash HA of all members except herself, and so does Bob (=HB). When they meet, she gives him HA and her name, he gives her HB and his name. By the property H(HB +B) = H(HA+A), they can verify that they are indeed members.

Nonmembers can be given the accumulation of everyone HE; now they can identify you're a member of a covert group even though they can't find the other members, since HE = H(HA +A).

New members get added just by sending around their names. Can't delete people easily.

6 Key Escrow

Key Escrow is the idea of surrendering your secret key to the Feds ("We're from the Government and we're here to help"). It arises when you start believing that "criminal elements", or whoever your political or ideological enemies are, will gain some advantage over you by using strong cryptosystems. The other idea is to force everyone to use weak cryptosystems (DES, yes?).

In fact key escrow is the basis of Our Government's Clipper program and its Escrowed Encryption Standard. We want individual privacy and court-authorized wiretaps. An EES chip has

tamper-proof hardware with a unique ID and a secret key. The key is split in two and stored with two escrow agencies, along with the ID number of course.

In an session, the EES chip encrypts the session key with the chip's secret key, then transmits the encrypted session key and its ID over the comm. channel. To wiretap and decrypt, listen in for the ID number, collect the keys from the agencies, XOR them together, decrypt the session key, and decrypt the traffic. There are complications when you have cheaters, and it can be done with PK as well [Schneier].

Micali, the guy who came up with this, calling it a *fair cryptosystem, is supposed to have collected a million bucks of our taxpayer's money for the use of his patents.

As an escrow agency, you'd like to know your chunk of key was actually a valid part of the private key, not just a random string. Without reconstructing the key! With N chunks, each of N trustees knows he has a valid chunk of key, the feds need a court order to reconstruct the key from N chunks, and Mallory has to corrupt N trustees to do the same thing.

- A creates her (private-key, public-key) key pair. She splits the private key into several public pieces and private pieces.
- A sends a public piece and corresponding private piece to each of the trustees. These messages must be encrypted. The public key goes to the KDC.
- Each trustee verifies that the private piece and public piece work together and are correct. Each trustee stores the private piece and sends the public piece to the KDC.
- The KDC performs another calculation on the public pieces and the public key to verify that everything is copasetic. If so, it signs the public key and either sends it back to A or posts it in a public database.

Any PK algorithm can be made fair (Algorithms in Schneier Ch. 23). Also you can do threshold schemes (previous notes) so that some subset of the trustees can reconstruct the private key. Scheme is vulnerable to subliminal channels, which will defeat the wiretapping, so *failsafe key escrow was developted to defeat that.

Schneier has a couple of pages on the politics of key escrow. Bottom line is that there is no advantage for the user and lots of possibilities for big problems on the national and international fronts.

7 Subliminal Channels

I was going to ignore this but Key Escrow motivates it, so...

How to two people communicate secretly when Mallory can read what they send, it can't be encrypted, and when M wants to send misleading messages to either of them? *Subliminal channels to the rescue. So maybe the number of words in a sentence is the channel. Even for 0, odd for 1. This is a steganography, of course, with no key. Security depends on the secrecy of the algorithm. You can put a subliminal channel into a conventional digital signature. This is a form of obfuscation, since the signature appears normal. Mallory can't read or even detect the subliminal message.

- A generates some random innocuous message.

- Using key shared with B, she signs message in such a way that she hides her subliminal message in the signature (the interesting part.. see Schneier Chapter 23.3).
- A sends signed message to B thru M.
- M reads msg, checks signature, looks OK, passes it to B.
- B checks the signature, verifying it's from A
- B ignores the innocuous message and extracts the subliminal message using the secret key.

M can't fake A's signature. Cheating between A and B is different, because in some algorithms the key to read is the same as the key to sign, so B can impersonate A. But other implementations avoid this problem. A could safely sign a document under threat, simply embedding "I am being coerced" into the signed confession or whatever. Companies can embed marks in documents that lets them be tracked, the govt can mark digital cash...

SO, guess what? There are *subliminal-free signature schemes!