University of Maryland
CMSC456 — Introduction to Cryptography
Professor Jonathan Katz

Lecture 27

1 Generating Primes and Testing Primality

See notes for Lecture 26.

2 Public-Key Cryptography

Thus far in the class, we have only discussed the private-key (also known as "secret-key" or "symmetric-key") setting. Here, two parties who wanted to communicate securely had to somehow share in advance a random key; furthermore, the complete secrecy of this key needed to be maintained at all time. This approach to cryptography has a number of disadvantages which we sketch here:

1. The parties must be able to somehow share a random key in a secure manner. But if they can do this, they might just as well share their message in the same secure manner! (Of course, this is not always possible. For example, the parties may be able to securely share a key *now* but will want to communicate tomorrow when secure transmission is no longer possible.)

2. Along the same lines, note that in some cases even meeting a single time to securely agree upon a key is not possible. What if our sender and receiver are in a crowded room (or are always monitored) so that they cannot share a secret key?

3. In the private-key model, you always need to know (in advance) the parties with whom you want to communicate. Accepting communication from anybody is essentially impossible in the secret-key setting. (Examples of where you might want to accept communication from anyone might include: a company that wants to accept credit card numbers from anyone, or a government agency that wants anyone to be able to submit suggestions)

4. Finally, if we want to secure a network with n parties (representing, say, n people working for the same company) then we need everyone to share a key with everyone else, for a total of $\binom{n}{2} = O(n^2)$ keys overall.

This motivates the introduction of *public-key cryptography*. Since we will begin with a discussion of public-key encryption, we discuss this example now. Here, a party who wishes to receive communication generates two keys: a public-key and a corresponding secret key (or private key). We will denote these by PK and SK, respectively. The public-key is widely distributed: everyone (including an adversary) is assumed to know about this

person's public key. Now, anyone can communicate with this party using only this public key; the receiver can decrypt using only his secret key.

Note that we can view this encryption in two different, but equivalent, ways. First, imagine that A and B want to communicate, but have not set up a shared key in advance. They can do the following:

1. B generates a public-key/private-key pair (PK, SK).
2. B sends PK to A. The secret key remains secret.
3. A obtains PK and encrypts message M as: $C \leftarrow \mathcal{E}_{PK}(M)$ (note that encryption might be randomized). A sends C to B.
4. B obtains C and decrypts it as: $M = \mathcal{D}_{SK}(C)$ (typically, decryption will be deterministic).

Now (informally), the security we desire from such a scheme is the following: an adversary who passively eavesdrops on *all* communication between A and B (in particular, who sees PK and C) should be unable to determine anything about the message M. (We will rigorously formalize a notion of security for this setting over the next few lectures.) Note carefully what this means if we can achieve this: it means that two people in a crowded room — who have not even shared any information in advance — can shout information to each other and keep the contents of their conversation hidden from everyone else in the room!

We may also view the encryption as happenning in two, distinct stages. In stage one:

1. B generates a public-key/private-key pair (PK, SK).
2. B widely publicizes PK to anyone who might ever want to communicate with him. He makes sure to keep SK secret.

At some later point in time, A wants to communicate with B; she does so as follows:

1. A finds that she indeed knows PK, the public key of B.
2. A encrypts message M as above: $C \leftarrow \mathcal{E}_{PK}(M)$. A sends C to B.
3. B obtains C and decrypts it as: $M = \mathcal{D}_{SK}(C)$.

Here (again, informally), the security we desire from such a scheme is that an adversary eavesdropping on all communication between A and B should be unable to determine M. But in this setting, the only communication between A and B is the ciphertext C, so this seems to be a weaker requirement than before! In fact, we should have more carefully stated our security goal: an adversary *who is assumed to know* PK and who eavesdrops on all communication between A and B should *still* be unable to determine M. We stress that we assume the adversary knows PK because B makes no effort to prevent its widespread dissemination; it is only fair to assume the adversary knows PK. It should be clear that this definition of security is now equivalent to the previous one.

Let's briefly see how public-key cryptography can address some of our concerns with private-ley cryptography.

1. Parties wishing to communicate do *not* have to share keys in a secret manner. Note, however, that they do need to exchange information in some "authenticated" manner (we assumed a passive adversary above but it should be clear that an adversary who can interfere with the communication can completely ruin the security of the scheme). We will return to this issue (briefly) below, and may discuss this point later in the semester as well.

2. The party publishing his public key does *not* need to have any idea who will use the key to communicate with him. Anyone who learns the public key (and B tries to publicize it so that as many people as possible leanr it) can communicate with B.

3. In a network of n people, each party now needs only distribute his public key to the other $n-1$ people in the network. The total number of keys is now n (rather than $O(n^2)$ as before).

Having given these arguments in favor of public-key encryption, what are the drawbacks? (If there are no drawbacks then why bother learning about private-key cryptography at all!) We mention a few here:

1. From a practical point of view, public-key cryptography is much less efficient than private-key cryptography. To give a rough example, encrypting short messages is about 1000 times faster using private-key cryptography than using public-key cryptography. Of course, a lot of research has been and continues to be focused on making public-key operations as efficient as possible (we discuss some of these methods later in the semester).

2. From a theoretical point of view, public-key cryptography sometimes requires stronger assumptions than private-key cryptography. As an example, we noted that indistinguishable private-key encryption can be achieved under the assumption of one-way functions. Public-key encryption cannot be achieved under this same assumption.

3. Public-key cryptography relies strongly on some system for reliably distributing public keys. Note that it is not always so easy to publicize a public key in a reliable manner. As an example, one way of publicizing a public key is to list it in the newspaper. Well, what if I put my public key (for which I know the associated secret key) in the newspaper and claim that this is really someone else's (say, Amazon's) key? In this case, I will be able to read all messages (including credit card numbers, etc.) which were intended for Amazon. Resolving this issue turns out the be — perhaps — the biggest drawback to using public-key cryptography in practice.

4. Finally, we note that in the case of private-key cryptography both parties "know" (in some sense) exactly to whom they are speaking (if two parties share an encryption key, then [informally speaking] messages sent by any other party will be unintelligible anyway). On the other hand, if someone publishes their public key and receives a message, they have no way of reliably telling who sent that message. (Of course, the sender can claim some identity in the message itself, but how do we know he is not lying?)