Disclaimer. Draft note. No guarantee on completeness nor soundness. Read with caution, and shoot me an email at fsong@pdx.edu for corrections/comments (they are always welcome!)

Logistics. Typo in HW 4 Problem 4: $E'_{pk}(m) = E_{pk}(m_1)\ldots$. Remarks on QUIZ 3. Solution posted on D2L. Quiz 4 next class.
Last time. Public-key encryption.
Today. Public-key encryption cont'd, CCA

1 PubKE constructions cont'd

Diffie-Hellman's original idea, i.e., using a TDP directly as a PubKE is not CPA since it's deterministic. Instead, we combine a hardcore predicate of a TDP, and construct a PubKE for single-bit messages. In principle, we are done. We can use this construction to encrypt long messages bit-wise, which is still CPA-secure. But in practice, we would like to encrypt long messages more efficiently. Our good friend, RO, will help us out.

1.1 PubKE in RO

Bellare-Rogaway CPA. Let (G, F, I) be a TDP, and $\mathcal{O} : \{0,1\}^* \to \{0,1\}^\ell$ be a Random Oracle. Construct $\Pi = (G, E, D)$ with message space $\{0,1\}^{\ell(n)}$ in Fig. 1. Intuitively, as long as F is hard to invert, y would be totally random which acts as a one-time-pad on plaintext m.

KeyGen G:	**Encrypt** $E_{pk}(m)$:	**Decrypt** $D_{sk}(c)$:
$(pk, sk) \leftarrow G(1^n)$.	• $r \leftarrow \{0,1\}^\ell$. Get $y := \mathcal{O}(r)$, • Output $c := (f(r), y \oplus m)$.	• Parse c as (c_1, c_2). • Compute $r \leftarrow I_{sk}(c_1)$ and output $m := c_2 \oplus \mathcal{O}(r)$.

Figure 1: CPA PubKE in RO from trapdoor permutations

FS NOTE: Good to know, but not essential for this course.
Efficiency improvement: OAEP. One shortcoming of the constructions above is the efficiency overhead (e.g. longer ciphertexts). Bellare and Rogaway proposed another transformation - *optimal asymmetric encryption padding* (OAEP) based on any trapdoor permutations, which achieves CPA-security (an even stronger security against *Chosen-ciphertext-attacks* (CCA) is possible. It's more tricky and we discuss it in the latter part). The basic idea is applying random padding and a two-round Feistel network to "mix" the plaintext before encrypting. Our building blocks are:

- (G, F, I): a trapdoor permutation on $\{0,1\}^{n+k_0+k_1}$.
- $\mathcal{O}_1 : \{0,1\}^{k_0} \to \{0,1\}^{n+k_1}$: random oracle 1.
- $\mathcal{O}_2 : \{0,1\}^{n+k_1} \to \{0,1\}^{k_0}$: random oracle 2.

FS NOTE: Draw encryption diagram

KeyGen G: $(pk, sk) \leftarrow G(1^n)$.	**Encrypt** $E_{pk}(m)$: $m \in \{0,1\}^n$ • Sample $r \leftarrow \{0,1\}^{k_0}$. $m' := m \| \bar{0}$ denotes message m appended with k_1 bits of 0. • Compute $s := \mathcal{O}_1(r) \oplus m'$, and $t := \mathcal{O}_2(s) \oplus r$. • Output $c := f(s\|t)$.	**Decrypt** $D_{sk}(c)$: • Compute $I_{sk}(c)$ and parse it as $s\|t$. • Compute $r := \mathcal{O}_2(s) \oplus t$ and $m' := \mathcal{O}_1(r) \oplus s$. Output the first n bits of m' as m.

Figure 2: CPA from OAEP

1.2 Factoring, RSA and TDP

Now that we have many constructions using TDPs, we need a candidate TDP.

Recall that $\mathbb{Z}_N^*$ under multiplication modulo N is a group of order $\phi(N)$.

Example 1. $N = 15 = 3 * 5$. Then $\mathbb{Z}_{15}^* = \{1,2,4,7,8,11,13,14\}$.

- $|\mathbb{Z}_{15}^*| = \phi(15) = \phi(3) \cdot \phi(5) = 8$.
- $8^{-1} = 2$ in $\mathbb{Z}_{15}^*$ because $8 \cdot 2 \bmod 15 = 1$.

Here comes the famous Fermat's little theorem and its generalization known as Euler's theorem.

Theorem 2 (KL-Corollary 8.21)**.** *Let $N > 1$ and $a \in \mathbb{Z}_N^*$. Then*

$$a^{\phi(N)} = 1 \bmod N. \quad \textit{(Euler's theorem)}$$

In the special case that $N = p$ and $a \in \{1,2,\ldots,p-1\}$, we have

$$a^{p-1} = 1 \bmod p. \quad \textit{(Fermat's little theorem)}$$

Now we introduce the famous problems and assumptions related to integer factorization.

The factoring problem and assumption. Let GMod be a polynomial-time algorithm that on input 1^n, output (N, p, p) where $N = pq$ and p, q are n-bit primes, $(N, p, q) \leftarrow \mathsf{GMod}(1^n)$. The factoring problem is defined as finding p, q given N, where $(N, p, q) \leftarrow \mathsf{GMod}(1^n)$. Define $\mathsf{Factor}_{\mathscr{A},\mathsf{GMod}}(n) = 1$ if $\mathscr{A}$ succeeds in finding p and q.

Definition 3 (KL-Definition 8.45)**.** Factoring is hard relative to GMod, if for all PPT $\mathscr{A}$,

$$\Pr[\mathsf{Factor}_{\mathscr{A},\mathsf{GMod}}(n) = 1] \le \mathrm{negl}(n).$$

The Factoring assumption
there exists a GMod relative to which the **factoring** problem is hard.

The study of factoring has a long history and yet the best factoring algorithm known still requires running time $\sim \exp(n^{1/3} \log n^{2/3})$ based on general number field sieve.

The RSA problem and assumption. Consider group $\mathbb{Z}_N^*$. Let $e > 2$ and $\gcd(e, \phi(N)) = 1$. Define

$$f_e : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$$
$$x \mapsto [x^e \bmod N]$$

Then f_e is a permutation on $\mathbb{Z}_N^*$ [**TS**: Verify on your own]. The inverse permutation is actually $f_d(y) := y^d \bmod N$, i.e., the same function with a different exponent, where $ed = 1 \bmod \phi(N)$ [KL: Corollary 8.22].

$$(x^e)^d = x^{ed} \overset{WHY?}{=} x^{ed \bmod \phi(N)} = x \bmod N.$$

The RSA problem is basically inverting f_e, i.e. computing e-th root modulo N.

More formally, let GRSA be a PPT algorithm $(N, e, d) \leftarrow \mathsf{GRSA}(1^n)$, where N is the product of two n-bit primes, and $\gcd(e, \phi(N)) = 1$ and $ed = 1 \bmod \phi(N)$. TheRSA game:

FS NOTE: Draw RSA-INV diagram

1. CH runs $(N, e, d) \leftarrow \mathsf{GRSA}(1^n)$.
2. Choose uniform $y \in \mathbb{Z}_N^*$.
3. $\mathscr{A}$ is given N, e, y and outputs $x \in \mathbb{Z}_N^*$.
4. Define $\mathsf{RSA}_{\mathscr{A},\mathsf{GRSA}}(n) = 1$ iff. $x^e = y \bmod N$.

Definition 4 (KL-Definition 8.46)**.** The RSA problem is hard relative to GRSA, if for all PPT $\mathscr{A}$, $\Pr[\mathsf{RSA}_{\mathscr{A},\mathsf{GRSA}}(n) = 1] \leq \mathrm{negl}(n)$.

The RSA assumption
there exists a GRSA relative to which the **RSA** problem is hard.
i.e., the function defined by $F_e(x) := x^e \bmod N, I_d(y) := y^d \bmod N$ is a trapdoor one-way permutation (referred to as RSA-TDP hereafter).

Relationship btween RSA and factoring. RSA $\leq$ Factoring clearly. [**TS**: Why?] Does hardness of factoring imply hardness of RSA? This remains an open question. We do know that finding d from N, e is as hard as factoring N. In your homework, you need to show that computing $\phi(N)$ is as hard as factoring N as well.

1.3 Instantiating PubKE with RSA-TDP

- **"Textbook" RSA**. Again NOT CPA secure.

- **RSA TDP + hard-core predicate**. Can we find a HCP for RSA TDP? There is a very simple one:

$$\mathsf{lsb}(x) := \text{least significant bit of } x, x \in \mathbb{Z}_N^*.$$

Fact 5 (KL-Theorem 11.31). *$\mathsf{lsb}(\cdot)$ is a hard-core predicate of RSA TDP F.*

Hence we obtain a CPA-secure encryption for single-bit messages.

- **RSA-OAEP**. Plugging the RSA-TDP into the OAEP construction, we immediately get a CPA-secure PubKE. In fact, RSA-OAEP is CCA-secure, which we will discuss soon. A version of it has been standardized as part of RSA PKCS #1 V2.0[1].

Using RSA correctly is more tricky than you might think. Many pitfalls in implementations and other vulnerabilities had open route to various attacks. Read Dan Boneh's survey [Bon99].

1.4 El Gamal PubKE

We do not know of a simple construction of a trapdoor one-wy permutation based on discrete-log related assumptions. Hence we cannot port the PubKE designs we have seen directly. However, it is actually not difficult to adapt the DH key-exchange protocol and obtain a PubKE known as the *El Gamal* scheme.

Recall $\mathscr{G}$ denotes a PPT group sampling procedure, $(G, q, g) \leftarrow \mathscr{G}(1^n)$.

Let $\mathscr{G}$ be as above. Construct $\Pi = (G, E, D)$

- G: run $(G, q, g) \leftarrow \mathscr{G}(1^n)$, choose uniform $x \leftarrow \mathbb{Z}_q$ and compute $h := g^x$.

$$pk := (G, q, g, h), \quad sk := (G, q, g, x).$$

Message space is group elements of G.

- E: on input pk and message $m \in G$, choose uniform $y \leftarrow \mathbb{Z}_q$ and output

$$c := (c_1 = g^y, c_2 = h^y \cdot m).$$

- D: on input sk and ciphertext $c = (c_1, c_2)$, output

$$\hat{m} := c_2 / c_1^x$$

Figure 3: The El Gamal PubKE scheme

Correctness of the El Gamal

$$\hat{m} = \frac{c_2}{c_1^x} = \frac{h^y \cdot m}{(g^y)^x} = \frac{g^{xy} \cdot m}{g^{xy}} = m.$$

To gain some intuition about the security, note that the critical information that an adversary obtains is $(h = g^x, g^y, g^{xy} \cdot m)$. But under DDH assumption, g^{xy} would be indistinguishable from an independent uniform h'. Then the encryption is essentially one-time-pad in group G with fresh "key" form each message. Read KL book for the detailed security proof.

[1] RSA Laboratories Public-Key Cryptography Standard. https://tools.ietf.org/html/rfc2437.

Theorem 6. *Under DDH assumption, El Gamal is CPA-secure.*

1.5 Hybrid Encryption

All PubKE schemes we have seen so far rely on problems in number theory. They are usually much slower than private-key encryption schemes. Encrypting every message with a PubKE, especially when they are long, is not very efficient in practice. A common practice in the real world is to combine PubKE with PrivKE as a *hybrid encryption* scheme. Basically, Alice (sender) uses Bob's (receiver) public key to encrypt a priv-key (session-key), and uses PrivKE to encrypt the actual message m. Bob would first decryt using secret-key in PubKE to recover the session key with which he recovers the message.

Let $\Pi^{pub} = (G^{pub}, E^{pub}, D^{pub})$ be a PubKE and $\Pi^{priv} = (G^{priv}, E^{priv}, D^{priv})$ be a PrivKE. Construct **PubKE** $\Pi = (G, E, D)$

- $G = G^{pub}$: run $(pk, sk) \leftarrow G^{pub}(1^n)$.
- E: on input pk and message m, run $k \leftarrow G^{priv}(1^n)$. Output ciphertext
$$c := (c_1 = E^{pub}_{pk}(k), c_2 = E^{priv}_k(m))\,.$$
- D: on input sk and ciphertext $c = (c_1, c_2)$, compute $k := D^{pub}_{sk}(c_1)$ and output
$$m := D^{priv}_k(c_2)\,.$$

Figure 4: Hybrid Encryption

Notice Π is a PubKE, but benefits from the efficiency advantage of a PrivK.

Theorem 7 (KL-Thm. 11.12)**.** *If Π^{pub} is CPA-secure and Π^{priv} is computationally secret, then Π is CPA-secure.*

Proof idea. A hybrid argument bridging computational indistinguishability.

$$\begin{tikzcd}
\langle E^{pub}_{pk}(k), E^{priv}_k(m_0)\rangle \arrow[rr, leftrightarrow, "\text{3: transitivity}"] \arrow[d, leftrightarrow, "\text{1: Security of }\Pi^{pub}"'] & & \langle E^{pub}_{pk}(k), E^{priv}_k(m_1)\rangle \arrow[d, leftrightarrow, "\text{1': Security of }\Pi^{pub}"] \\
\langle E^{pub}_{pk}(0^n), E^{priv}_k(m_0)\rangle \arrow[rr, leftrightarrow, "\text{2: Security of }\Pi^{priv}"] & & \langle E^{pub}_{pk}(0^n), E^{priv}_k(m_1)\rangle
\end{tikzcd}$$

□

2 Security against Chosen-Ciphertext-Attacks

We have discussed before that encryption itself does not necessarily provide data integrity. In fact, if a ciphertext can be modified, i.e. *malleable*, it may comprise secrecy sometimes. Consider the following scenario:

Suppose Alice sends Bob an encrypted email starting with `From:Alice@mail.com` under Bob's public key, call the ciphertext c. Now an adversary Charlie intercepts c and modifies the underlying message so that it starts with `From:Charlie@mail.com`, call the new ciphertext c'. Then c' is sent to Bob, who decrypts and gets Alice's actual message m. Now Bob replies the email, but to Charlie, instead of Alice, and *quote the decrypted text* m (the entire message may be encrypted under Charlie's public key). Charlie hence learns m.

In some sense, Charlie managed to access the decryption algorithm corresponding to Bob's secret key. This motivates considering a stronger attacking model, *chosen-ciphertext-attacks* (CCA), and defining CCA-secure encryption.

> **FS NOTE**: Draw CCA game
>
> 1. CH runs $(pk, sk) \leftarrow G(1^n)$.
> 2. $\mathscr{A}$ is given pk and access to a decryption oracle $D_{sk}(\cdot)$. $\mathscr{A}$ outputs a pair of messages (m_0, m_1) of the same length.
> 3. CH picks uniform $b \leftarrow \{0,1\}$, compute $c^* \leftarrow E_{pk}(m_b)$ and give it to $\mathscr{A}$.
> 4. $\mathscr{A}$ continues to interact with $D_{sk}(\cdot)$, but may not request decrypting c^* itself. Finally $\mathscr{A}$ outputs b'. We say $\mathscr{A}$ succeeds if $b' = b$.
> 5. Define $\mathsf{PubK}^{\mathsf{cca}}_{\mathscr{A},\Pi}(n) = 1$ iff. $\mathscr{A}$ succeeds.

Figure 5: CCA indistinguishability game $\mathsf{PubK}^{\mathsf{cca}}_{\mathscr{A},\Pi}(n)$

Definition 8 (KL-Def. 11.8)**.** Π is CCA-secure if for all PPT $\mathscr{A}$, $\Pr[\mathsf{PubK}^{\mathsf{cca}}_{\mathscr{A},\Pi}(n) = 1] \le \frac{1}{2} + \mathrm{negl}(n)$.

Likewise we can define CCA security for private-key encryption, just noting that we need to provide encryption oracle $E_k(\cdot)$ explicitly [KL: Section 3.7].

2.1 Constructing CCA-secure encryption schemes

CCA-secure PrivKE: Combining ENC & MAC [KL: Section 4.5]. A natural idea in the private-key setting is to combine PrivKE and MAC:

- Encrypt-**and**-MAC: may be completely insecure
- Encrypt-**then**-MAC: CCA-secure for any CPA-secure ENC and eu-cma-secure MAC.
- MAC-**then**-Encrypt: Not always secure. CCA-secure with CBC and Randomized Counter mode.

CCA-secure PubKE from OAEP in RO. Bellare and Rogaway claimed that OAEP with any TDP achieves CCA [BR94]. However, a bug in their proof was later identified, and only CPA (& CCA-1) can be achieved. Nonetheless OAEP with the RSA permutation is indeed CCA as shown by [Sho01, FOPS04]. [FOPS04] actually showed that any trapdoor permutation with the special *partially one-way* security property gives CCA under OAEP. Shoup [Sho01] also gave a variant of OAEP called OAPE+ which achieves CCA with any trapdoor permutation (standard one-way).

Hybrid Encryption. In fact, hybrid encryption gives a generic way of designing new CCA-secure PubKE schemes. If Π^{pub} and Π^{priv} are both CCA-secure, then hybrid scheme Π is also CCA-secure. There are more efficient transformations in the RO model that converts weaker encryption (e.g. CPA) to CCA-secure encryption (e.g. [FO99]).

Direct CCA constructions based on number-theoretic assumptions. The first efficient CCA PubKE without RO based on the DDH assumption is due to Cramer and Shoup [CS03]. Subsequently a CCA-secure scheme based on the RSA assumption was shown by Hoffheinz and Kiltz [HK12].

References

[Bon99] Dan Boneh. Twenty years of attacks on the rsa cryptosystem. *Notices of the AMS*, 46(2):203–213, 1999. PDF at `http://crypto.stanford.edu/~dabo/pubs/papers/RSA-survey.pdf`.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *Advances in Cryptology–EUROCRYPT 1994*, pages 92–111. Springer, 1994.

[CS03] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Advances in Cryptology–CRYPTO 1999*, pages 537–554, 1999. Full version in Journal of Cryptology 2013.

[FOPS04] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the rsa assumption. *Journal of Cryptology*, 17(2):81–104, 2004. Prelim in CRYPTO 2001.

[HK12] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. *Journal of Cryptology*, 25(3):484–527, 2012.

[Sho01] Victor Shoup. OAEP reconsidered. In *Advances in CryptologyâĂŤ-CRYPTO 2001*, pages 239–259. Springer, 2001.