RSA and Padding

This Video Covers:

- How padding is done in RSA
- Examples with OpenSSL

RSA and Padding

- Secret-key encryption uses encryption modes to encrypt plaintext longer than block size.
- RSA used in hybrid approach (Content key length << RSA key length)

- To encrypt:
  - **short plaintext:** treat it as a number, raise it to the power of e (modulo n)
  - **large plaintext:** use hybrid approach; treat the "content key" as a number and raise it to the power of e (modulo n)

Treating plaintext as a number and directly applying RSA is called plain RSA or textbook RSA

Attacks Against Textbook RSA

- RSA is a ***deterministic*** encryption algorithm
  - The same plaintext encrypted using the same public key gives the same ciphertext
  - secret-key encryption uses randomized IV →different ciphertexts for same plaintext

- For ***small*** e and m
  - if $m^e <$ modulus n
  - e-th root of ciphertext gives plaintext

- If same plaintext is encrypted e times or more using the same e but different n, then it is easy to decrypt the original plaintext message via the Chinese remainder theorem

Padding Schemes: **PKCS#1 v1.5** and **OAEP**

- The simple fix to defend against previous attacks is to ***add randomness*** to the plaintext ***before encryption*** → ***padding!***

- **Types of padding:**
  - **PKCS#1** (up to version 1.5); weakness discovered since 1998
  - Optimal Asymmetric Encryption Padding (**OAEP**); prevents attacks on PKCS

- `rsautl` command provides options for both types of paddings
  (PKCS#1 v1.5 is the default… why? IDK…)

PKCS Padding

```
$ cat msg.txt
This is a secret.

$ openssl rsautl -encrypt -inkey public.pem -pubin -in msg.txt -out msg.enc -pkcs
```

PKCS Padding

```
$ cat msg.txt
This is a secret.

$ openssl rsautl -encrypt -inkey public.pem -pubin -in msg.txt -out msg.enc -pkcs

$ openssl rsautl -decrypt -inkey private.pem -in msg.enc -out newmsg.txt -raw
Enter pass phrase for private.pem: csci476
```

PKCS Padding

```
$ cat msg.txt
This is a secret.

$ openssl rsautl -encrypt -inkey public.pem -pubin -in msg.txt -out msg.enc -pkcs

$ openssl rsautl -decrypt -inkey private.pem -in msg.enc -out newmsg.txt -raw
Enter pass phrase for private.pem: csci476

$ xxd newmsg.txt
00000000: 0002 a6dc c092 9a2e 4a8e 3849 c14f cf0b  .........J.8I.O..
00000010: b036 de51 b222 28ab 1b98 6018 5e04 b084  .6.Q."(...`.^...
00000020: 31fc c2ef 680f a4f7 07c9 2b04 8d84 089d  1...h.....+.....
00000030: a2f3 5bbc 2f82 2969 18a1 6c09 2762 82a6  ..[./.)i..l.'b..
00000040: 7d26 b7e0 1a41 077b 86a8 4459 9a0d 6b61  }&...A.{..DY..ka
00000050: af55 a61d 0101 8f26 1ed1 cc3b 33c9 74db  .U.....&...;3.t.
00000060: bad1 38a4 dd0e 59b5 8097 4d93 a400 5468  ..8...Y...M...Th
00000070: 6973 2069 7320 6120 7365 6372 6574 2e0a  is is a secret..

$ ls -al msg.enc
-rw-rw-r-- 1 seed seed 128 Mar 18 14:29 msg.enc
```

OAEP Padding

- Original plaintext is not directly copied into the encryption block
- Plaintext is first XORed with a value derived from random padding data

```
$ openssl rsautl -encrypt -inkey public.pem -pubin -in msg.txt -out msg.enc -oaep
```

OAEP Padding

- Original plaintext is not directly copied into the encryption block
- Plaintext is first XORed with a value derived from random padding data

```
$ openssl rsautl -encrypt -inkey public.pem -pubin -in msg.txt -out msg.enc -oaep

$ openssl rsautl -decrypt -inkey private.pem -in msg.enc -out newmsg_oaep.txt -raw
Enter pass phrase for private.pem: csci476

$ xxd newmsg_oaep.txt
00000000: 00cd 119c 1376 6ea4 bb17 cd2e 5462 52a1  .....vn.....TbR.
00000010: 4dd1 2031 f446 c3ea f000 55b2 785d 86ba  M. 1.F....U.x]..
00000020: 97af dba7 4ee1 cd02 5fa3 4752 488d f523  ....N..._.GRH..#
00000030: 9d7c c69b f1a8 dba2 c4d1 9c14 f0f1 4abe  .|............J.
00000040: 3c1c e904 711d 0944 2f0b 8b72 7f82 06dc  <...q..D/..r....
00000050: 50af bf94 cac1 b402 7522 7d17 6fc8 699d  P.......u"}.o.i.
00000060: e4ab fff9 952a fb47 673e 7bf5 729f 96bb  .....*.Gg>{.r...
00000070: c282 b678 15c5 2a22 5ae6 bcf1 51be 1a2e  ...x..*"Z...Q...
```

OAEP Padding

- Original plaintext is not directly copied into the encryption block
- Plaintext is first XORed with a value derived from random padding data

```
$ openssl rsautl -encrypt -inkey public.pem -pubin -in msg.txt -out msg.enc -oaep

$ openssl rsautl -decrypt -inkey private.pem -in msg.enc -out newmsg_oaep.txt -raw
Enter pass phrase for private.pem: csci476

$ xxd newmsg_oaep.txt
00000000: 00cd 119c 1376 6ea4 bb17 cd2e 5462 52a1  .....vn.....TbR.
00000010: 4dd1 2031 f446 c3ea f000 55b2 785d 86ba  M. 1.F....U.x]..
00000020: 97af dba7 4ee1 cd02 5fa3 4752 488d f523  ....N..._.GRH..#
00000030: 9d7c c69b f1a8 dba2 c4d1 9c14 f0f1 4abe  .|............J.
00000040: 3c1c e904 711d 0944 2f0b 8b72 7f82 06dc  <...q..D/..r....
00000050: 50af bf94 cac1 b402 7522 7d17 6fc8 699d  P.......u"}.o.i.
00000060: e4ab fff9 952a fb47 673e 7bf5 729f 96bb  .....*.Gg>{.r...
00000070: c282 b678 15c5 2a22 5ae6 bcf1 51be 1a2e  ...x..*"Z...Q...

# decrypt without -raw to recover the original data (need -oaep flag!)
$ openssl rsautl -decrypt -inkey private.pem -in msg.enc -out newmsg_oaep.txt -oaep
Enter pass phrase for private.pem: csci476
```