

CS556 Introduction to Cryptography - Prof. Ming-Deh Huang

Scribe: Iftikhar A Burhanuddin

burhanud@usc.edu

Lecture #13, #14, #15 - October 8, 10, 15 2002

### Announcements:

Notes for Lecture #11 and #12 are online. The Pari-GP assignment is due on October 18. Dan Boneh's paper [1] is on the Announcements page. It's a good survey about state of the art attacks known till 1998. Please read chapters on DES and Politics of Cryptography from Schenier's book. Qing Luo's Office Hours: F 2-4 SAL 317.

### Topics for today:

1. Blinded Signatures contd.
2. Continued Fractions
3. Low Decryption Exponent Attack
4. Theory of Resultants
5. Low Encryption Exponent Attack

## 1. Blinded Signatures contd.

Last lecture we had asked: *How does the c.i.a (denoted by A) prevent Bob (denoted by B) from cheating while signing blinded documents?*

**Solution.** Bob wants the c.i.a to blindly sign one of the messages  $m_i, 1 \leq i \leq 100$  where  $m_i$  says "whoever uses the cover of  $c_i$  should be given diplomatic immunity. Signed Director c.i.a", where  $c_i$  is the  $i$ th covername. For example  $c_1 = \text{Ragin Bull}$ ,  $c_2 = \text{Tree Beard}$ , ... We assume that Bob will be happy with anyone of the covernames.

1. Bob computes  $m'_i = m_i b_i$  where  $b_i = E(b'_i)$  and  $b'_i$  is a random number picked by Bob.  $E_A(b'_i)$  is called the blinding factor.
2. Bob gives all the  $m'_i$  to the A.

3.  $A$  then randomly chooses 99 messages and asks Bob for the corresponding  $b'$  so that it can read the 99 messages. If all the opened messages are covername messages then the Agency gives Bob  $D_A(m'_j)$  for some  $1 \leq j \leq 100$  corresponding to the message which was not opened by the Agency. If on the other hand the Agency reads a message that makes an outrageous claim then Bob is caught cheating and then the c.i.a proceeds to take appropriate action.
4. If Bob receives  $D_A(m'_j)$ , he recovers  $D_A(m_j)$  as follows:

$$\begin{aligned} D_A(m'_j) &= D_A(m_j E_A(b'_j)) = D_A(m_j) D_A(E_A(b'_j)) = D_A(m_j) b'_j \\ \Rightarrow D_A(m_j) &= b'_j{}^{-1} D_A(m'_j) \end{aligned}$$

**Question.** Can Bob still cheat?

Say Bob generates 2 sets of messages: the covername messages  $m_i$  and the outrageous claims  $n_i$  with blinding factors  $b_i$  and  $c_i$  respectively for  $1 \leq i \leq 100$  such that  $m_i b_i = n_i c_i$ . When Alice asks Bob for the  $k$ th blinding factor he gives  $b_k$  the blinding factor corresponding to the  $k$ th covername message rather than  $c_k$  the blinding factor corresponding to the  $k$ th outrageous claim. And finally if Bob is successful in cheating uses the signed outrageous claim corresponding to the unopened message.

So what prevents Bob from cheating in such a way? Observe that  $b_i = E_A(b'_i)$  and  $c_i = E_A(c'_i)$  and Bob needs both  $b'_i$  and  $c'_i$  to be a successful con. Without loss of generality suppose he randomly picks  $b'_i$  and computes  $b_i$  and then also obtains  $c_i = m_i b_i n_i^{-1}$ . But now to compute  $c'_i$  he will have to compute  $D_A(c_i)$ , which is attacking the decryption problem!

## 2. Continued Fractions

A continued fraction looks like this and such a representation is unique:

$$x = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \dots}}}$$

and the  $a_i \in \mathbf{Z}$  can be computed as follows: First compute  $a_0 = [x]$  and let's define  $x_0 := x - a_0$ . Now  $x_0 = \frac{1}{a_1 + x_1}$ . That is  $a_1 = [\frac{1}{x_0}]$  which implies  $\frac{1}{x_0} = a_1 + x_1$ . Hence  $x = a_0 + \frac{1}{a_1 + x_1}$  and we proceed like this inductively.

**Claim.** This process will eventually terminate  $\Leftrightarrow x \in \mathbf{Q}$

**Proof.**  $\Leftarrow$  is obvious. For the  $\Rightarrow$  direction observe that if  $x \in \mathbf{Q}$   $\text{denom}(x_i) < \text{denom}(x_{i-1})$  and therefore the process eventually terminates.

Let's look at an example:

$$\begin{aligned} x &= \frac{3}{11} = a_0 + \frac{1}{a_1+x_1} \\ a_0 &= \left[ \frac{3}{11} \right] = 0 \\ a_1 &= \left[ \frac{11}{3} \right] = 3 \\ x_1 &= \frac{11}{3} - 3 = \frac{2}{3} \end{aligned}$$

Observe how  $\frac{3}{11}$  reduced to  $\frac{2}{3}$ . The denominator drops from 11 to 3 and then to 2. Infact if  $x$  is a rational number then you are doing Euclid's gcd algorithm with Continued fraction as shorthand notation! And the *depth* of the fraction equals the number of steps in Euclid's algorithm. Here's the corresponding gcd computation.

$$\begin{aligned} 11 &= 3 \cdot 3 + 2 \\ 3 &= 1 \cdot 2 + 1 \\ 2 &= 2 \cdot 1 + 0 \end{aligned}$$

Computing each new  $a_i$  in the continued fraction us better and better approximations of  $x$ . Intutively each iteration gives a better approximation by one digit or so. In real life continued fractions are used to calculate good approximations of  $\pi, e, \dots$

<i>convergent #</i>	<i>shorthand</i>
0	$[a_0] := a_0$
1	$[a_0, a_1] := a_0 + \frac{1}{a_1}$
2	$[a_0, a_1, a_2] := a_0 + \frac{1}{a_1 + \frac{1}{a_2}}$
3	$[a_0, a_1, a_2, a_3] := a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3}}}$
...	...

Say the  $i$ th convergent  $[a_0, a_1, \dots, a_i] := \frac{b_i}{c_i}$ , where  $b_i$  and  $c_i$  follow the following recurrence  $b_i c_{i-1} - b_{i-1} c_i = (-1)^{i-1}$  for all  $i \geq 1$  with  $\frac{b_0}{c_0} = \frac{a_0}{1}$  and  $\frac{b_1}{c_1} = \frac{a_0 a_1 + 1}{a_1}$ . For  $i \geq 2$

$$\frac{b_i}{c_i} = \frac{a_i b_{i-1} + b_{i-2}}{a_i c_{i-1} + c_{i-2}}$$

Continued fractions exhibit some interesting properties one of which is:

**Thm.** Suppose  $\gcd(a, b) = \gcd(c, d) = 1$  and  $\left| \frac{a}{b} - \frac{c}{d} \right| < \frac{1}{2d^2}$  then  $\frac{c}{d}$  is one of the convergents in the continued fraction of  $\frac{a}{b}$

In other words the theorem says that that if  $\frac{c}{d}$  is quite *close* to (pretty good approximation of)  $\frac{a}{b}$  then the former appears in the continued fraction of the latter. We'll not prove this theorem.

*Remark.* Shor's quantum polynomial time algorithm for factoring uses above theorem!

### 3. Low Decryption Exponent Attack

Last couple of lectures we've seen that when implementing the RSA cryptosystem we should

1. Generate strong primes, that is,  $p$  and  $q$  should be of the same length but  $p$  and  $q$  shouldn't be close to each other, otherwise Fermat factorization will crack the system as you saw in the HW #2
2.  $p - 1, q - 1$  shouldn't be  $\log n$  smooth
3. two users shouldn't use the same modulus
4. messages shouldn't come from a small domain

Today we'll add one more to this list - the Low Decryption Exponent Attack was proposed by Wiener [2] who proved the following

**Thm.** If  $3d < n^{\frac{1}{4}}$  and  $q < p < 2q$  (this is essentially the  $p$  and  $q$  of the same length requirement) then the system is vulnerable.

So if  $\log n = 1024$  then using a decryption key  $d < 250$  bit is a bad idea. Wiener's idea was that  $\frac{e}{n}$  might contain information about  $p$  and  $q$  and he used method of continued fractions to compute  $d$ .

**Proof.** The # of divisions in computing  $\gcd(e, n)$  using the extended Euclidean gcd is  $\leq 2 \log n$  therefore the # of convergents in the expansion of  $\frac{e}{n}$  is  $\leq 2 \log n$ .

$ed \equiv 1 \pmod{\phi(n)} \Rightarrow ed = 1 + t\phi(n)$ .  $n$  is known but  $d, \phi(n)$  and  $t$  are unknown.

The intuition behind the theorem is that  $\left| \frac{e}{\phi(n)} - \frac{t}{d} \right| = \frac{1}{d\phi(n)}$  and if we replace  $\phi(n)$  by  $n$  and we'll not be too far off.

Since  $p < q < 2p$  we have  $p < \sqrt{n}$  and hence  $q < 2\sqrt{n}$ . So  $p+q-1 < 3\sqrt{n}$ . Now  $ed - tn = 1 + t\phi(n) - tn = 1 + t(\phi(n) - n) = 1 - t(p+q-1) < 3\sqrt{nt}$ .  
 $\left| \frac{e}{n} - \frac{t}{d} \right| = \left| \frac{ed-tn}{nd} \right| < \frac{3\sqrt{nt}}{nd} = \frac{3t}{\sqrt{nd}} < \frac{n^{1/4}}{\sqrt{nd}} = \frac{1}{n^{1/4}d} < \frac{1}{3d^2}$ .

As  $ed = 1 + t\phi(n)$  and  $0 < e, d < \phi(n)$  we have  $t < d < \frac{n^{1/4}}{3}$  and the using the theorem in the previous section we are done.  $\diamond$

Since  $ed - t\phi(n) = 1$ , we have  $\gcd(t, d) = 1$  and hence  $\frac{t}{d}$  is a reduced fraction. So the recipe to crack the system using this attack is to calculate the  $\log n$  convergents of  $\frac{e}{n}$  iteratively and check whether the  $d$  in each convergent is the right one.

*Remark.*

1. A natural question to ask is how big can the exponent get? The best record is if  $d < n^{0.29}$  then the system is vulnerable. Though it has been conjectured that *if  $d \not\leq n^{0.5}$  then the system is safe.*
2. To make encryption cheap encryption small exponents are used and to make signing cheap small decryption exponents are used. To make signing possible in smart cards small  $d$ 's are used. And hence the above attack is bad news for practioners.

## 4. Theory of Resultants

**Definition.** If  $f(x) = \sum_{i=0}^m a_i x^i$  and  $g(x) = \sum_{i=0}^n b_i x^i$  are polynomials of degrees  $m$  and  $n$  respectively then the *Resultant* of  $f$  and  $g$  denoted by  $Res(f, g)$  is the determinant of an  $m+n$  square matrix.

$$Res(f, g) = \begin{vmatrix} a_m & a_{m-1} & \dots & a_1 & a_0 & 0 & \dots & 0 \\ 0 & a_m & a_{m-1} & \dots & a_1 & a_0 & \dots & 0 \\ \dots & & & & & & & \\ 0 & \dots & 0 & a_m & a_{m-1} & \dots & a_1 & a_0 \\ b_n & b_{n-1} & \dots & b_1 & b_0 & 0 & \dots & 0 \\ 0 & b_n & b_{n-1} & \dots & b_1 & b_0 & \dots & 0 \\ \dots & & & & & & & \\ 0 & \dots & 0 & b_n & b_{n-1} & \dots & b_1 & b_0 \end{vmatrix}$$

For example say  $f(x) = a_3x^3 + a_2x^2 + a_1x + a_0$  and  $g(x) = b_2x^2 + b_1x + b_0$

then

$$\text{Res}(f, g) = \begin{vmatrix} a_3 & a_2 & a_1 & a_0 & 0 \\ 0 & a_3 & a_2 & a_1 & a_0 \\ b_2 & b_1 & b_0 & 0 & 0 \\ 0 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & b_2 & b_1 & b_0 \end{vmatrix}$$

**Thm.** Let  $\mathbf{k}$  be a field<sup>1</sup>. If  $f(x) = \sum_{i=0}^m a_i x^i = a_m \prod_{i=1}^m (x - \alpha_i)$  and  $a_i \in \mathbf{k}$  then  $\alpha_i \in \overline{\mathbf{k}}$ , the algebraic closure of  $\mathbf{k}$ .

$f(x) = a_m \prod_{i=1}^m (x - \alpha_i)$  says that  $\alpha_i$  are the roots of  $f$ , that is  $f(\alpha_i) = 0$

**Fact.** If  $a_i \in \mathbf{R}$  (or  $a_i \in \mathbf{C}$ ) then  $\alpha_i \in \mathbf{C}$ .

Say  $f(x) = \sum_{i=0}^m a_i x^i = a_m \prod_{i=1}^m (x - \alpha_i)$  and  $g(x) = \sum_{i=0}^n b_i x^i = b_n \prod_{i=1}^n (x - \beta_i)$ . Observe that if  $a_i, b_i \in \mathbf{k}$  then  $\text{Res}(f, g) \in \mathbf{k}$ .

**Thm.**

$$\text{Res}(f, g) = a_m^m b_n^n \prod_{1 \leq i \leq m} \prod_{1 \leq j \leq n} (\alpha_i - \beta_j)$$

**Cor.**  $\text{Res}(f, g) = 0 \Leftrightarrow$  there exists  $\gamma$  such that  $f(\gamma) = g(\gamma) = 0$

We can calculate the resultant by a straight forward determinant computation in time  $(n + m)^3$  field operations. We next turn to defining the resultant for bivariate polynomials by treating polynomials in  $x$  and  $y$  as polynomials in  $x$  with coefficients as polynomials in  $y$ .

Say  $F(x, y) = a_m(y)x^m + \dots + a_0(y)$  and  $G(x, y) = b_n(y)x^n + \dots + b_0(y)$  with  $a_i(y), b_i(y) \in \mathbf{k}[y]$ . Let  $r(y) := \text{Res}_x(F(x, y), G(x, y))(y)$  where the subscript  $x$  denotes that we've eliminated  $x$  and the matrix has as entries polynomials in  $y$ .

$$\text{Res}_x(F(x, y), G(x, y)) = \begin{vmatrix} a_m(y) & a_{m_1}(y) & \dots & a_1(y) & a_0(y) & 0 & \dots & 0 \\ 0 & a_m(y) & a_{m_1}(y) & \dots & a_1(y) & a_0(y) & \dots & 0 \\ \dots & & & & & & & \\ 0 & \dots & 0 & a_m(y) & a_{m-1}(y) & \dots & a_1(y) & a_0(y) \\ b_n(y) & b_{n_1}(y) & \dots & b_1(y) & b_0(y) & 0 & \dots & 0 \\ 0 & b_n(y) & b_{n_1}(y) & \dots & b_1(y) & b_0(y) & \dots & 0 \\ \dots & & & & & & & \\ 0 & \dots & 0 & b_n(y) & b_{n-1}(y) & \dots & b_1(y) & b_0(y) \end{vmatrix}$$

---

<sup>1</sup>to be defined in the next lecture

By the above corollary when  $r(\alpha) = Res(F(x, \alpha), G(x, \alpha)) = 0 \Leftrightarrow$  there exists  $\beta$  such that  $F(\beta, \alpha) = G(\beta, \alpha) = 0$ . Intuitively if  $f, g$  are two curves we project them along the  $y$ -axis and the projected points are captured by  $r(y)$  and the resultant captures the common zeros of  $f$  and  $g$ .

## 5. Low Encryption Exponent Attack

Though mathematically encryption and decryption are symmetric, in reality there is non-symmetry as  $e$  is public and  $d$  is private. Hence the low encryption exponent attack is more subtle and far less a crack compared to the low decryption exponent attack.

If a message is yes or no and if the adversary knows this, the message can be recovered by brute force. Ideally we don't want to leak any information. If we send 2 identical messages, the adversary can say that they are the same messages if the ciphertexts are the same if we use a deterministic encryption function like RSA. This is 1 bit of information leak. For critical applications we would like to avoid this. So we add random bits so that the eavesdroppers cannot differentiate between the 1st and the 2nd ciphertext. So padding the message with random 0's and 1's seems to be a good idea.

Say message  $m$  is padded with  $r$  random bits and  $m' = m||r$  is sent for encryption. It is not difficult for receiver to recover message. He or she computes  $D(E(m')) = m||r$  and the first chunk the receiver will be able to read and the rest will be unreadable junk which the receiver will discard.

Suppose the same message  $m$  is sent twice in encrypted form over the network but each time it is padded with a different set of  $k$ -random bit pads  $r_1, r_2$ . Say  $m_1 = m||r_1$  that is  $m_1 = m2^k + r_1$  and  $m_2 = m||r_2$  that is  $m_2 = m2^k + r_2$  and  $c_1 = m_1^e \bmod n$  and  $c_2 = m_2^e \bmod n$

**Thm.** Say  $e$  the encryption exponent is upper bounded by a constant and  $k \leq \frac{\log n}{e^2}$  and given  $c_1, c_2, e, n$  then  $m$  can be recovered (without factoring  $n$  or computing the decryption key.)

Let  $\Delta := m_2 - m_1 = r_2 - r_1$ . So  $|\Delta| < 2^k$ . Please bear in mind that the operations happen in  $\mathbf{Z}/n\mathbf{Z}$  unless otherwise specified.

Let  $F(x, y) := x^e - c_1$  and  $G(x, y) := (x + y)^e - c_2$ . These functions by construction share a common zero  $(m_1, \Delta)$ , that is  $F(m_1, \Delta) = G(m_1, \Delta) = 0$ . Hence  $Res_x(F, G)(y) = 0$  has  $\Delta$  as it's solution. This is true because  $Res_x(F, G)(\Delta) = 0 \bmod p$  and  $Res_x(F, G)(\Delta) = 0 \bmod q$  implies

$Res_x(F, G)(\Delta) = 0$  in  $\mathbf{Z}/n\mathbf{Z}$ .

The following powerful result due to Coppersmith (which we'll assume is true without proof tells us that we can quickly calculate small roots of a modular polynomial) will help us retrieve  $\Delta$ .

**Thm.** Suppose  $f \in \mathbf{Z}[x]$  and of degree  $d$  then we can find solutions  $x_0$  with  $|x_0| < N^{\frac{1}{d}-\epsilon}$  to  $f(x) \equiv 0 \pmod{N}$  in time polynomial in  $\log n$  and  $\frac{1}{\epsilon}$  and  $d$ .

Now observing that  $\deg Res_x(F, G)(y) \leq e^2$  and  $|\Delta| < 2^k$  and using Coppersmith's result we can compute the *small* root  $\Delta$  from our polynomial  $Res_x(F, G)(y)$  provided  $k < \frac{\log n}{e^2}$ .

Let's assume we've computed  $\Delta$  using Coppersmith's algorithm, now we'll proceed to recover  $m$ .

**Claim.**  $\gcd(f, g) = x - m_1$ , where  $f(x) = x^e - c_1$  and  $g(x + \Delta)^e - c_2$

**Proof.**  $f$  and  $g$  share a common root  $m_1$  by design as  $f(m_1) = g(m_1) = 0$ . Say there exists  $\alpha \neq m_1$  and  $\alpha \in (\mathbf{Z}/n\mathbf{Z})^*$  such that  $\alpha^e = c_1$ . This implies  $\alpha^{ed} = c_1^d \Rightarrow \alpha^{1+\phi(n)} = \alpha = c_1^d$  but  $m_1 = c_1^d$ . So there cannot be another root of  $x^e - c_1$  in  $(\mathbf{Z}/n\mathbf{Z})^*$ . Hence  $m_1$  is the unique common root of  $f$  and  $g$  in  $(\mathbf{Z}/n\mathbf{Z})^*$ .  $\diamond$

Once we've computed  $m_1$ , the part of  $m_1$  which makes sense is  $m$  and the rest of the message is made up of the  $k$ - random bits. Also it's very unlikely that we won't be able to compute the gcd. But the good news is that in this very unlikely case we'll actually find a factor of  $n$ .

If  $e = 3$  and  $\log n = 1024$ , then this attack says that even padding  $k = \frac{1024}{9} \sim 100$  random bits is not a good idea.

Hence we see that the Low Encryption exponent attack is a much weaker and subtler result compared to the Low Decryption exponent attack because

1.  $e$  is a constant or near constant and
2. The system only gets weakened and not broken

RSA is based on the presumed difficulty of integer factoring. This is not to say unless you cannot factor quickly you cannot crack the RSA system. Security of RSA is not just about how difficult factoring or computing  $\phi(n)$  as we've seen. Weaknesses of RSA come in all shapes and sizes.

Are most integers equally hard to factor? Can we find a set of integers with non-zero density which are easy to factor? When  $p$  is prime it's most

likely that  $p-1$  is not smooth. Ruling out primes  $p, q$  such that  $p-1, q-1$  are not smooth numbers is ruling out only some of the non-zero density primes.

We can say that Primality Testing is easy in the practical sense since we can rule out composites easily. Given  $n$  answering Yes  $\Leftrightarrow n$  is prime in random polynomial time was known in 1992. Doing it deterministically is a recent (2002) result.

The Prime number theorem tells us that the density of primes is  $\frac{1}{\log n}$ . So working with 512 bit numbers, roughly 1 in every 500 numbers is prime. So prime number generation can easily be done by a PC. Hence RSA is based on the fact that primality testing is easy and but more importantly that factoring is presumably hard, which is a \$100 million question!

Does a witness to the compositeness of number  $n$  have to be a factor of  $n$ ? No! FLT tells us  $a^{p-1} \equiv 1 \pmod p$ , when  $p$  is prime. When  $p$  is not prime it's very likely that the congruence doesn't hold.  $a^{99} \equiv 1 \pmod{100}$  if it doesn't hold we have a witness to the compositeness of 100 and the witness is not a factor.

### **Bibliography:**

1. Dan Boneh. Twenty years of attacks on the RSA cryptosystem. Notices of the American Mathematical Society (AMS), Vol. 46, No. 2, pp. 203-213, Feb., 1999.
2. M. Wiener. Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory, 36:553-558, 1990.