1 Tutorial 1

1. Give real life examples of information where you'd primarily protect it for a subset of "confidentiality," "integrity," and "availability". For example, data whose integrity you care about but not its confidentiality or availability etc. etc. Some bit of contrivance is ok!

   Confidentiality only: say maybe when you want to destroy some information forever, say your medical records **forever**. Or maybe this is to destroy its availability forever.

   Integrity & Availability: You've just been married to your favorite person. You want people to know about it and you want them to know that you married the *right* person.

   Availability only: You've just started a rumor so you don't care about the information's integrity, and you don't want it to be confidential.

   Confidentiality & Integrity: say maybe the contents of a will. Or software distribution à la superdistribution.

2. Give examples of how your favorite operating system realizes (or fails to realize) one or more of Saltzer & Schroeder's design principles for secure systems.

   Complete mediation: Unix fails for file accesses but NFS4 apparently enforces them.

   Open Design: Linux probably meets it, Windows probably doesn't.

   Least Privilege: Vanilla Unix probably fails it because it's not a privilege-based OS. Processes must become root in order to do security related activities.

   Least Common Mechanism: PAM might be an example of it.

   Separation of privileges: Need both password and location of access before being allowed into the system.

3. What are "canaries" and how do they help in a specific kind of buffer overflow.

   See Section 3.4.2 from the Cowan et. al paper.

4. Compute GCD(1875, 405) showing every step in the computation. Explain why

$$a \times b \bmod n = (a \bmod n \times b \bmod n) \bmod n$$

   GCD(1875, 405) = GCD(405, 255) = GCD(255, 150) = GCD(150, 105) = GCD(105, 45) = GCD(45, 15) = 15.

5. Encrypt the following text using Vigenère's encryption with the keyword "NUS".

The moon began to rise, and I thought of the placid look at the white ceiling, which had passed away. The moon began to rise, and I thought of the pressure on my hand when I had spoken the last words he had heard on earth.

GBW ZIGA VWTUF GI JVMW, NHV V NZBOYUN GS NZR JDNWAQ FGBE SG NZR QZVNW PYAYCFT, QZVWZ UUV CUKFYV NQSL. NZR GGBH TRASA NG ECKR, UFQ C LUIMTBL BZ LUY HEYKFOJR IF ZS ZNHV JBWA C ZNX KCICRH LUY DNML JIJQM ZR BSQ BWNLV BH WNLLU.

6. Find the index of coincidence of the encrypted text in the previous question. From the IC, can you make a determination of the size of the keyword used for Vigenère encryption.

   $IC \approx 0.04108$. Keyword length ≈ 10!

7. Consider the Linear Congruential Generator

$$r_{i+1} = (a \times r_i + b) \bmod n$$

   where $n = 2^{31} - 1$, $a = 16807, b = 0$. Starting with a seed of 32 generate the next hundred (pseudo) random numbers and comment on whether the numbers are uniformly distibuted over $0 \ldots (n-1)$.

   537824, 14375175, 25108420, 48591184, 65647798, 104263761, 153351643, 166087451, 203279197, 228293697, 230611224, 239140606, 268720695, 279745341, 298396290, 310810960, 312221300, 385194808, 400687501, 440525240, 448940298, 449273380, 455934356, 457809047, 499264043, 520030653, 524918388, 529015328, 578319116, 584258316, 617222191, 628243628, 628413197, 667068796, 669970701, 699968546, 704869445, 705353886, 772130285, 773030562, 791254072, 838242904, 855763208, 912161872, 932675420, 940810486, 948348447, 992644487, 1023877877, 1052596858, 1066181259, 1081247786, 1085397761, 1090420128, 1109442088, 1114252897, 1119575216, 1133512504, 1191038039, 1200839479, 1220965263, 1229536575, 1268928922, 1294261505, 1307349127, 1327472840, 1334282928, 1388445880, 1453425998, 1483562155, 1526371937, 1531577471, 1554071509, 1556928156, 1560617032, 1583736949, 1662083261, 1683430075, 1711585032, 1722923113, 1733564591, 1819461042, 1822342580, 1850531504, 1851702990, 1861047144, 1940149762, 1943997415, 1954580925, 1966310418, 1987295956, 1993595962, 2014465249, 2024811874, 2037744988, 2040981744, 2044225328, 2072676013, 2097504821, 2110229375

2 Tutorial 2

1. Find all primes ≤ 144 using the method of sieving designed by Eratosthenes. Draw a 12×12 grid and cross out all numbers that have been "sieved out" by the method.

1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72
73	74	75	76	77	78	79	80	81	82	83	84
85	86	87	88	89	90	91	92	93	94	95	96
97	98	99	100	101	102	103	104	105	106	107	108
109	110	111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130	131	132
133	134	135	136	137	138	139	140	141	142	143	144

2. Compute $(345^{28567} \times 23^{567} + 1078/65) \bmod 29$ given that 29 is a prime.

$$\begin{aligned} &= 345^7 \times 23^7 + 5 * (-4) \text{ use Fermat's} \\ &= 26^7 \times 23^7 - 20 \\ &= 17 \times 1 - 20 \\ &= -3 \\ &= 26 \end{aligned}$$

   Note: $345^{28567} = 345^{1020*28+7}$. So by Fermat's theorem we get $345^{1020*28} \bmod 29 = 1 \bmod 29$.

3. Use modular exponentiation to calculate $97^{5678} \bmod 101$.

   $5678 = 1011000101110_2$. So we need to find $97^2, 97^3 \ldots 97^{12} \bmod 101$. These values are $16, 54, 88, 68, 79, 80, 37, 56, 5, 25, 19, 58$.

   So, $97^{5678} \bmod 101 = 16 \times 54 \times 88 \times 79 \times 5 \times 25 \times 58 \bmod 101 = 37$.

4. Find $45623457^{-1} \bmod 2389511$ using the EGCD method.

   It should be 354745.

$$
\begin{array}{c}
(1, 0, 45623457) \\
(0, 1, 2389511) \\
\hline
(0, 1, 2389511) \\
(1, -19, 222748) \\
\hline
(1, -19, 222748) \\
(-10, 191, 162031) \\
\hline
(-10, 191, 162031) \\
(11, -210, 60717) \\
\hline
(11, -210, 60717) \\
(-32, 611, 40597) \\
\hline
(-32, 611, 40597) \\
(43, -821, 20120) \\
\hline
(43, -821, 20120) \\
(-118, 2253, 357) \\
\hline
(-118, 2253, 357) \\
(6651, -126989, 128) \\
\hline
(6651, -126989, 128) \\
(-13420, 256231, 101) \\
\hline
(-13420, 256231, 101) \\
(20071, -383220, 27) \\
\hline
(20071, -383220, 27) \\
(-73633, 1405891, 20) \\
\hline
(-73633, 1405891, 20) \\
(93704, -1789111, 7) \\
\hline
(93704, -1789111, 7) \\
(-261041, 4984113, 6) \\
\hline
(-261041, 4984113, 6) \\
(354745, -6773224, 1)
\end{array}
$$

For each of these questions you may write a program to solve it. You must show (and demonstrate understanding) of all the steps needed to arrive at the answer.

3 Tutorial 3

1. Convert the super increasing sequence

$$
\begin{aligned}
A = \; & \{81, 162, 322, 572, 1167, 2386, 4702, 9469, 18888, 37766, 75593, 151190, \\
& 302324, 604627, 1209249, 3926579\}
\end{aligned}
$$

into a "random" looking one by transforming it using $w = 279$ and $m = 7936729$.

This looks to be

$$A = \{22599, 45198, 89838, 159588, 325593, 665694, 1311858, 2641851, 5269752, 2599985, 5216989, 2498365, 4981106, 2019624, 4037853, 246939\}$$

Encode the message "Sammy Cheng is cool la" by taking two characters at a time and using their ASCII bit pattern to make a selection in the "random" knapsack.

Encoding "Sa" makes the selection 0x5361 into the knapsack. This is the bit sequence 0101001101100001

$$= 45198 + 159588 + 1311858 + 2641851 + 2599985 + 5216989 + 246939 = 12222408$$

Decrypt the encrypted message using the trapdoor (w, m, A) to recover the message.

$w^{-1} = 3470541$. So $12222408 \times 3470541 \bmod 7936729 = 4054843$. Solving for it in the super increasing knapsack yields the selection 0101001101100001.

And so on...

2. For the RSA cryptosystem where $p = 8663835841$ and $q = 802360858343257$, find d for $e = 65537$. Then encrypt the message "**attack@2**" by using the concatenation of the ASCII encoding of each character in the message as a large integer. Then decrypt the cipher text using the private key to verify that you've recovered the original message.

   $n = p \times q = 6951522761929833877274137, \phi(n) = 6951522761127464355095040$. $d = e^{-1} \bmod \phi(n) = 13858071757042635732382 73$.

   attack@2 is the hex sequence $61747461636B4032 = 7022365680606068786$. Its encryption is

   $$7022365680606068786^{65537} = 10725221375141273616099 26$$

   Its decryption is

   $$10725221375141273616099 26^{13858071757042635732382 73} = 7022365680606068786$$

3. Find primes p and q so that 12-bit plaintext blocks can be encrypted with RSA.

   Twelve bits means that we must find primes p, q such that $p \times q = n \geq 4096$. Say $p = 43$, $q = 419$.

For each of these questions you may write a program to solve it. You must show (and demonstrate understanding) of all the steps needed to arrive at the answer.

4 Tutorial 4

1. Use the Miller-Rabin primality test to determine whether the following numbers are likely to be prime. Try with bases $b = 2, 3, 4$. Show all the steps in the computation i.e., the value of b^y where $n - 1 = 2^e \cdot y$ and then the values in each iteration of the computation of $b^{y^{2^e}}$.

   - 125.
     $2^{31} = 23, 23^2 = 29, 29^2 = 91$. So 125 fails the Fermat test.
   - 561.
     $2^{35} = 263, 263^2 = 166, 166^2 = 67, 67^2 = 1!$. 561 fails the Miller-Rabin primality test.
   - 3017049239485840259629.

     $$\begin{aligned} 2^{7542623098714600649 07} &= 1041102915096788007582 \\ 1041102915096788007582^2 &= 2822113103589653422290 \\ 2822113103589653422290^2 &= 669058088867085399090 \end{aligned}$$

     So 3017049239485840259629 fails the Fermat test for primality.
   - 5472011961261553846573217.

     $$\begin{aligned} 2^{1710003737894235577054 13} &= 4152402373042979414844561 \\ 4152402373042979414844561^2 &= 297268502766585781027689 \\ 297268502766585781027689^2 &= 129715058485234280459524 9 \\ 129715058485234280459524 9^2 &= 5472011961261553846573216 = -1 \end{aligned}$$

     We are finished with the test because we know that Fermat's result will be 1. So this base seems to indicate that the number is a prime.

2. For a DES key of $0x0123456789ABCDEF$ and a plaintext of $0x0123456789ABCDEF$, show all the intermediate values of the computation in getting from L_0, R_0 to L_1, R_1. You can verify the final values using the CGI script des.cgi. Note that I have removed the part of the script that expands the F function.

3. Show that the following property holds for DES encryption

   $$\overline{DES_k(X)} = DES_{\overline{k}}(\overline{X})$$

   That is, encrypting the complement of a block with the complement of the key results in the complement of the cipher text.

For each of these questions you may write a program to solve it. You must show (and demonstrate understanding) of all the steps needed to arrive at the answer.

5 Tutorial 5

1. Some proposals suggest making the RSA modulus a product of three primes, $n = pqr$ of equal size. Describe the RSA system in this case. That is, explain how e and d are chosen.

   Sketch:
   Look at the proof of the RSA algorithm in Sec 3.5 of the text. Encryption and decryption with RSA makes use of the property that $(P^e)^d \bmod n = P$. This holds when $ed = 1 \bmod \phi(n)$. With 3 primes where $n = pqr$ then RSA works the usual way to determine e and d. If $n = pqr$ then $\phi(n) = n(1 - \frac{1}{p})(1 - \frac{1}{q})(1 - \frac{1}{r}) = (p-1)(q-1)(r-1)$.

2. Show how you can construct a permutation from a one-way function. Think about how DES does the same thing! In particular, show how you can construct a block cipher using SHA1 (you can pattern it after how DES does it). What is the block length of your cipher? What is its key length? How does the block cipher encrypt and decrypt?

   Sketch:
   A block cipher is one means of doing a permutation, i.e. C_i is permutation of $E(K, P_i)$. So one way of doing this is to use a use a one way function to construct a block cipher. There a number of possible block cipher constructions. One way is to use a Feistel network like DES. For the F function, we will use SHA1, choosing appropriately some number of bits. For example, $F_k(x) = SHA1(k.x)$ which is k concatenated with x. The block length and key length can be chosen appropriately within the parameters of the input and output size of SHA1. After some rounds, 3 is sufficient, you will have a reasonable block cipher.

3. Suppose that a cryptanalyst discovers a message P that is not relatively prime to the enciphering modulus $n = pq$ used in a RSA cipher. Show that the cryptanalyst can factor n.

   Sketch:
   Either p or q is $gcd(n, P)$ giving a factorization of n.

4. For a hypothetical 5-DES encryption scheme with 5 independent keys used as E-E-E-D-D, what is the effective key length for a KPA (known plain-text attack), CPA (chosen plain-text attack) given that storage is not a consideration.

   Sketch:
   For a KPA, a more efficient attack than pure enumeration is the *meet in the middle* attack. 5-DES gives $C = D_{k_5}(D_{k_4}(E_{k_3}(E_{k_2}(E_{k_1}(P)))))$. The middle attack is to find a value M roughly in the middle of the cipher cascade which corresponds to the given P_0 and C_0 for a particular key $k_1, \cdots, k_5$. Choose M to be after k_3. We need to guess/find the keys which lead to P_0 and C_0 from M. This can be done by constructing two tables. ¿From P_0 construct one table for all possible keys k_1 to k_3, i.e. $M_i = E_{k_3}(E_{k_2}(E_{k_1}(P_0)))$. Similarly from C_0, encrypt to obtain

a table for possible values of M_i' for the possible keys k_4 and k_5. A matching value, $M_i = M_i'$, indicates that the corresponding keys produces a consistent ciphertext C_0 for plaintext P_0. There may be a number of possible keys which are consistent, this can be verified with other cipher-plaintext pairs. The time complexity assuming $O(1)$ hashing is $O(2^{k_1+k_2+k_3})$, thus the effective keylength is $k_1 + k_2 + k_3$.

A CPA attack means that we can generate as many P_i, C_i pairs as we like. Using the meet in the middle table construction, the worst case time complexity is unchanged.

6 Tutorial 6

1. Encrypting the message "singapore" using a mono alphabetic substitution cipher created using the keyword "secure" results in the ciphertext:

   (a) PFKBSMLOR (b) ODJASLKNR (c) RHMFSONQB (d) QGLDSNMPA (e) None of the above

   Ans: I get the translation map

   ```
   ABCDEFGHIJKLMNOPQRSTUVWXYZ
   SECURABDFGHIJKLMNOPQTVWXYZ
   ```

   which makes for the translation "PFKBSMLOR".

2. When the Miller-Rabin primality test fails for a number n, n is definitely composite. But if it succeeds for n, n is not necessarily a prime.

   (a) True (b) False (c) Can't really tell (d) None of the above.

   Ans: True

3. Is $\log n$

   (a) $O(n)$ (b) $O(n^2)$ (c) $O(2^n)$ (d) all of the above (e) none of the above

   Ans: (d)

4. Consider the equality $(a + b)^p \equiv a^p + b^p \bmod p$. Is it

   (a) True (b) False (c) Depends on a, b, p (d) None of the above.

   Ans: (c). It's not specified that p is a prime.

5. For $n = 101$, will the table $3 \times x \bmod 101$ $(0 \leq x < 101)$ generate a permutation of $(0 \ldots 100)$?

   (a) True (b) False (c) Can't really tell without generating the whole table (d) Damn!

   Ans: (a) because $GCD(3, 101) = 1$.

6. Approximately how many samples will one need to find a prime around 10000000000000000. That is, if one were to randomly generate numbers close to 10^{16}, how many might you need to run the Miller-Rabin test on before you accept it as a probable prime.

   (a) 5000000000000000 (b) 37 (c) 19 (d) Can't really say.

   Ans: (c) One would make approx $\log 10^{16} = 37$ samples before finding a prime close to 10^{16}. But approx half of them will be even and wouldn't need to have the Miller-Rabin applied to them.

   Because of the ambiguity in the question (b) is also OK.

7. What can be said about the DES initial permutation IP?

   (a) DES would be substantially weakened without it (b) The strength of DES is equivalent to its strength without the IP (c) Without IP the keyspace to be searched to mount a known plaintext attack could be reduced substantially (d) It's hard to say anything definitive.

   Ans: (b) See [KPS02] for details.

8. In DES, suppose that $F(R, K) = 0$, i.e., for any input the F function output 0. What function does DES compute?

   (a) Inverse (b) Identity (c) Some other permutation (d) None of the above.

   Ans: (c) DES has a gratuitous swap at the end, so what you get before IP^{-1} is (R_0, L_0) not (L_0, R_0). So what you get is $IP^{-1}(SWAP(IP(input)))$ which is just a permutation.

9. In DES, how many bits in (L_1, R_1), i.e., the 64 bits of the result of the first round, are related to bit 1 in (L_0, R_0)? I.e., if the value of bit 1 in L_0 changes, how many bits of (L_1, R_1) may be changed? Assume key is the same in both cases.

   (a) 1 (b) 2 (c) 4 (d) 32 (e) 64 (all bits in L_1 and R_1)

   Ans: (a)

10. Suppose DES is modified so that the high order 44 key bits are set to 0 so that only the 20 low order bits are used. What is the effective key length of this system?

    (a) 20 (b) 18 (c) 17 (d) 15 (e) 12

    Ans: (c). Bits 48, 56 & 64 are parity bits which will be ignored.

11. What is $\phi(p^2)$ where p is an odd prime?

    (a) $p(p-1)$ (b) $(p-1)(p-1)$ (c) There isn't a closed form expression just involving p (d) $p^2 - 1$

    Ans: (a). $(p^2 - 1) - (p - 1) = p^2 - p = p(p - 1)$.

12. For how many primes p is $17p + 1$ a square?

    (a) None (b) One (c) Two (d) Infinitely many (e) Can't really tell.

    Ans: (b). Let $x^2 = 17p+1$. Then $\frac{x^2-1}{17 \cdot p} = 1$. Because both 17 and p are prime, 17 must divide either $(x-1)$ or $(x+1)$ exactly. This works out for $x = 18, p = 19$.

13. Consider building a message digest function (similar to SHA) using DES. In particular, consider using the CBC mode of DES. Let K be a fixed known encryption key and let IV denote a fixed public initialization vector. The message could be padded appropriately using zeros to make it of block length suitable for encrypting with DES.

    To generate a hash, encrypt the (possibly padded) message in CBC mode of DES using key K and initialization vector IV. Output the final block C_n of the resulting ciphertext as the message digest of the message. What can be said about this setup:

    (a) This hash scheme is not collision resistant. (b) This hash scheme is not preimage resistant. (c) This hash scheme is not second preimage resistant. (d) All of the above. (e) a & b only.

    Ans: (d). It is not (a) because given a 64-bit hash value, say c, one can find a single 64-bit block, say m such that $h(m) = c$. This is $D_k(c) \oplus IV$. Actually an arbitrary length message can be constructed that wil hash to a given value c. That is, you easily generate an arbitrary number of messages that hash to c. Because of this it is neither (c) nor (a).

14. What is 2579^{2579} mod 19?

    Ans: 10.

15. For the RSA system in which $n = 852337$, $e = 3$, $d = 566091$, find the factors of n. Indicate the smaller factor in the answer.

    Ans: $ed = k\phi(n) + 1$ for some integer k. But because $e = 3$, $d < \phi(n)$, and $ed - k\phi(n) = 1$ we must have $k = 1$ or 2.

    So $ed - 1 = 1698272 = 2 \times 849136$. This gives us the equations $p + q = n + 1 - \phi(n) = 3202$ and $p - q = \sqrt{6843456} = 2616$ which gives $p = 2909$ and $q = n/p = 293$.

7 Tutorial 7

1. Let $a = 1234, b = 4321$ and $m = 42$. Demonstrate how you can use the Chinese Remainder Theorem (crt) to compute $a + b$ mod m and b^{769} mod m using a representation with moduli $m_i = \{2, 3, 7\}$.

   Let $m_1 = 2, m_2 = 3, m_3 = 7$. In the reduced representation We have 1234 and 4321 represented by $(0, 1, 2)$ and $(1, 1, 2)$ respectively. $(0, 1, 2) + (1, 1, 2) = (1, 2, 4)$ which is 11 mod m by crt. $(1, 1, 2)^{769} = (1, 1, 2)$ which is 37 mod m by crt.

2. Suppose Pablo and Renee share a symmetric secret key and generate a fresh session key every hour whenever they need to communicate. Assume that Pablo and Renee use different machines on different networks on the Internet. Under this scheme, Pablo sends to Renee the message to "pay John $1000". How can an intruder exploit this scheme? Explain any improvements.

   See what happens with a replay message attack. Replaying the encrypted message plays havoc with P's account. Improvements to try – transaction ids, timestamps, authentication of sender, etc.

3. The symmetric key exchange protocol using a trusted server (page 3 of notes) uses a nonce I_p. Suppose we simplify the protocol to remove the use of I_p, we may argue for example that I_p is not used by P or R. This new protocol is less secure than the original one. Explain a possible attack. What assumptions do you need?

   The two protocols: ticket $T = E_{K_R}(K_{PR}, P)$

OLD	NEW
$P \to S : (P, R, I_P)$	$P \to S : (P, R)$
$S \to P : E_{K_P}(R, I_P, K_{PR}, T)$	$S \to P : E_{K_P}(R, K_{PR}, T)$
$P \to R : T$	$P \to R : T$

   One possible attack, assume K_R is compromised and R and S have changed their key to K'_R. Assume attacker M has already seen protocol session under old key K_R. M impersonates server so does not need to know new K'_R:

$$\begin{aligned} &P \to M : (P, R) \\ &M \to P : E_{K_P}(R, K_{PR}, T) \\ &P \to R : T \end{aligned}$$

   At step 2, M replays a previous message, he cannot decrypt under K_P. At step 3, P can decrypt message from step 2, so he thinks M is the real server. M watches step 3 and pretends to be R. M also knows K_{PR} since the ticket T can be decrypted with K_R.

   Another weakness, if the session key is ever reused, attacker will know this.

4. Consider a session key exchange scheme with RSA using the following following protocol where K_{PR} is the session key generated by P, K^P_{pub} denotes the public key of P and K^P_{priv} denotes the private key of P, and encryption and decryption with those RSA keys is denoted by E_P and D_P respectively:

$$\begin{aligned} &P \to R : (P, K^P_{pub}) \\ &R \to P : (R, K^R_{pub}) \\ &P \to R : (P, I_P, E_R(K_{PR})) \\ &R \to P : (R, E_P(I_P)) \end{aligned}$$

Show how an adversary who has control of the network can obtain the session key K_{PR}.

Consider the adversary M who has control of all network traffic. He will intercept and substitute messages between P and R so that P thinks he is talking to R and vice versa with R.

$$
\begin{aligned}
&P \rightarrow M : (P, K^P_{pub}) \\
&M \rightarrow R : (P, K^M_{pub}) \\
&R \rightarrow M : (R, K^R_{pub}) \\
&M \rightarrow P : (R, K^M_{pub}) \\
&P \rightarrow M : (P, I_P, E_M(K_{PR})) \\
&M \text{ now knows } K_{PR} \\
&M \rightarrow R : (P, I_P, E_R(K_{PR})) \\
&R \rightarrow M : (R, E_M(I_P)) \\
&M \rightarrow P : (R, E_P(I_P))
\end{aligned}
$$

8 Tutorial 8

Questions 1 and 2 are due the week of October 14, while questions 3, 4 and 5 are due the week of October 7th.

1. Consider $n = p \times q$ where both p, q are (probable) primes. Let

   $$p = 4703713376517472005806418066555779293457873398157752389958 09$$

   and let

   $$q = 4499844734117731401670236904640172639714995487317761872083826 1$$

   Find

   ```
   659798681212804331928725342070000524862635920743474257017255732454895236384572 62\
   351946261146698247560345
   ```

   raised to itself (mod n) using (a) modular exponentiation and (b) the Chinese Remainder Theorem. You must implement both of them yourself. Time both measurements by running your program 10 times and reporting the average and standard deviation in both instances.

   You may use a big integer package and use its multiplication, division, inverse, and other operations except for its modular exponentiation and chinese remainder theorem implementations.

   The answer you should get (if I've got it correct) is

- Using modular exponentiation.
  Decimal: **765385769101002231322357756496673841127590558343652908611841-8613982850185838405413953588038245207580313446648868475771207**,
  Hex: **0x2F6CA74DDC1D1A023184F6C12446322ABC8811678D15697D3EECF99-9213294AAAA32F75201EFC0D3C695B26F0E3F84B981947**
  and it should be faster computed using CRT.
  The mean time to compute the exponentiation for my tests was 19.6s, while the standard deviation was .27s.
- Using CRT
  The # in CRT pair is (**463314701768235875479102945779611968027772044705666443683387**, **296437249292846513087103327847946732297615015366131388972-39307**).
  After exponentiation, we get the pair (**370461059912961164789989749941846849236520396035595750752143**, **134576259453294711191479428142232856319590767-52927579530958682**).
  Converting it back to a number we get **765385769101002231322357756496673841127590558343652908611841861398285018583840541395358803824520758031344664886847577120 7**.
  The mean time to compute the exponentiation for my tests was 19s, while the standard deviation was .9s.

2. Consider a 512 bit Diffie-Hellman system in which

```
p = 0x00B074C48A962CDF1EB3895DA6DBE20A7AFBADE32ED9AF48CC7FFE378BBCE063848ECD57CCF\
      90D4184E0E91836F156D0D2C8063B948EC179CE54B179C7DADD8B45
q = 0x00A612AB9B9E27938F402C38BF6464BA1BFCA8C1B3
g = 0x373CBDEAFF2C44FE1EA25B500E112383F7E41F6278DA39E9347A640E9C95702A65E6BA2BD15\
      4DA6ABDFCB8E73107EB5CA9118DA79406EE2E7DEDC7B4157D15B7
```

If Alice uses the random integer 6325782345 and Bob uses the random integer 629851207 for key exchange and use the least significant 64 bits of the resulting shared secret as a DES key for encrypting traffic, what is the key they agree on?

$g^{629851207 \times 6325782345} \bmod p = ?$.

g = **289301072601089759860426688300467869347149557283748316981220336785659810764296239791339171062015613828077741084344508480769594195298331335246216752090872 7**.

p = **924175710323063009114860052762013168173384029246356304214850533944146180940019220414903971862624680335627265274310847050413089794519373153175020853038573 3**.

ab = **3984301645217540415**.

g^{ab} = **13738052515239702827697395226721522123182236867228885701076949639723571402988838585272784450921030980368212065515642264896545353222956043756933116565342 44**.

The least 64 bits of this number are $g^{ab} \bmod 2^{64} = 7216447536848380132$, i.e., 0x6425F8CCE61720E4.

3. Given that the Lamport hash value is sent in the clear over the network, why is it more secure than a password?

   Because any given hash value cannot be reused (it's one time use only) as a password, and it is computationally infeasible to derive the "next" hash value from a given hash value.

4. Is the Lamport hash protocol vulnerable to dictionary attack by an eavesdropper? Can someone impersonating the server do a dictionary attack?

   Yes. If Alice chooses a poor password p initially, then given n and $h^n(p)$, one can easily determine p using a brute force attack.

   Yes, as in above. The impersonating server can can send an n to Alice to get the corresponding $h^n(p)$ and use it for a dictionary attack. No, in the sense that being a server does not offer any further advantages.

5. As described in the text, the mental poker protocol has a flaw. If there are only ten cards, as soon as Alice sees her hand, she knows from set difference what cards Bob must have. Suggest an alternative protocol.

   We assume that in poker, the objective of the protocol is to prevent a player from knowing the other's cards. Assuming only 10 cards, for example card values from 1-10, then from your hand with 5 cards, you will know the other cards immediately. Here are some approaches to solving it.

   - One way would be to introduce a "fictitious" player Charlie who has his own keypair and in the enhanced protocol is played by Bob. On being presented with ten cards, Bob encrypts his selection with his public key and selects additional cards to serve as Charlie's hand and encrypts them with Charlie's public key. Now the protocol proceeds as before and both Alice and Bob can decrypt their cards. Bob doesn't know Charlie's hand so he can't deterministically predict Alice's, and Alice can't deterministically predict Bob's.
   - Consider a possible protocol with 52 cards, each player gets 5 cards and has no more information about the other player's cards except by guessing (at least initially, we make no assumption about how the game proceeds and poker variations).

     Let Alice be player **A** and Bob be player **B**. The protocol proceeds by choosing one card at a time as outlined below:

(a) Player A locks all the unchosen cards with k_A. Sends all these cards to player B.

(b) Player B cannot see the cards which are locked. Chooses 2 cards and locks with k_B. Sends the 2 cards to player A.

(c) Player A cannot tell which cards was chosen by B. He selects his card c_A. B's card is now c_B which A unlocks. Sends $< c_A, c_B >$ to B.

(d) Player B can unlock his card c_B. He unlocks c_A and sends it back to player A.

(e) Player A can unlock his card c_A.

One can either generalize this to 5 cards for each player or proceed with one card at a time.

9 Tutorial 9

1. Discuss how you would implement the Chinese Wall access control policy in your favorite operating system.

   Consider a Unix-like operating system. The Chinese wall model is history dependent and an access control decision cannot be made simply by examining the identity of the subject and the access permissions of the object. It depends on what objects a subject has accessed in the past. That history would have to be maintained persistently and used in the access control decision.

   One might say that, in effect, accessing an object belonging to a certain dataset results in an automatic modification to the access control list of all objects that belong to a different dataset but same conflict class such that the user is denied access to them.

2. Can you fit the Chinese Wall model into the Bell-La Padula framework?

   Yes and no. Yes in the sense that the abstract model can be mapped, no in the sense that it's hard to implement deterministically with a static set of à priori assignments of need-to-know compartments. See the paper by Brewer and Nash on the Chinese wall security policy.

3. What would happen if you used the Bell-La Padula model to model confidentiality and the Biba model to model integrity simultaneously?

   BLP and Biba use opposite policies when it comes to writing. BLP allows writes from the subject's security level upwards while Biba allows writes from the level downwards. Thus the only intersection for both is for writes to objects at the subjects security level.

   Read Fred Cohen's paper on Computer Viruses for diagrammatical representation of the combination of the two policies.

10 Tutorial 10

1. Q1 on page 333 of [Pfl96].

   If you didn't have **tranquility** in the BLP model, then while reading an object at a lower clearance level, its clearance level could change (of course in a manner consistent with the rules in the security policy). And while you are reading it, its clearance could change to become higher than yours. At the end of the change, you'd be violating the BLP model. With tranquility, reads and writes are atomic w.r.t to such changes so such kinds of violations would not occur.

2. Q4 on page 334.

   Yes. The relation $\subseteq$ is reflexive, transitive and anti-symmetric. Given any two sets A and B, the set $A \cup B$ is dominated by both A and B, and the set $A \cap B$ dominates both A and B.

3. Q5. This deals with secrecy because of the clearance labels. You are allowed to read from an object if your clearance is $\geq$ the clearance of the object, and the set of your need-to-know compartments $\supseteq$ the set of the need-to-know compartments of the object.

   (a) No (b) Yes (c) No (d) No (e) Yes (f) No.

4. Q6. If signals can be interpreted as sending encoded communication then a process at a higher clearance level should not be permitted to signal a process at a lower clearance level.

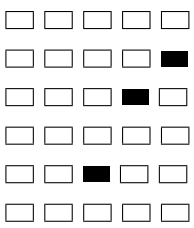
5. Q10. No because the owner of an object can always make the object world readable thereby permitting information in it to be read by anyone.

11 Tutorial 11

1. Considering the performance of your instructor in his last video taped lecture, what are his chances of making it as a leading protagonist in an epic Hollywood saga? Shade the appropriate box considering that the answer is 0/10.

   What would be his chances making a cameo appearance in Bollywood as a confused computer scientist eking a living in a tropical country. Shade the appropriate box assuming that the answer is 43/100.



2. Suppose that end-to-end (in this case machine to machine) authentication and confidentiality are required for communication *between two hosts*. If IPSec is used, show what an IPv4 datagram may look like that leaves one end point for the other. Consider one end point to be 137.132.90.23 and the other to be 137.132.87.29. Fill in as many fields in the datagram as you can.

   We are only considering tunnel mode. ESP provides both confidentiality and authentication.

First word of IP header
Second word of IP header
Third word of IP header with **protocol=ESP**
saddr=137.132.90.23
daddr=137.132.87.29
SPI
Sequence #
Tunneled IP datagram with saddr=137.132.90.23, daddr=137.132.87.29 + padding
Authentication data

   Discuss at a high level what a selector at host 137.132.90.23 for traffic to 137.132.87.29 looks like.

   At a high level, the selector matches the entire destination and maps it to an SA. So the selector looks like "137.132.87.29".

3. Suppose that you only wanted to connect to a service on a remote machine if it's running as a specified user, for example, you only want to connect to the ftp daemon on suna.comp if it's running as user root. Can you do that easily using IPSec?

   No easy way to specify it and no implementations that I know of that will invalidate SA's at the destination should ftp restart as different user id.

4. Suppose that on a Unix machine there are three users a, b, and c. User a is cleared at the "confidential" level, b is cleared at the "secret" level, and c is cleared at the "top secret" level. User a's home directory is /home/a, similarly for b and c.

How can you implement the MAC-style BLP policy in this setting using groups and ACLs. Consider only the SS property of BLP. In other words, c should be able to read everything under /home/{a,b,c}, b should be able to read everything under /home/{a,b} and so on.

One point to note to have MAC, the files cannot be owned by either of a,b,c to ensure that that the policy is always applied.

The groups for read access could be: files for c: g_c, files for b: g_{bc}, and files for a: g_{abc}.

Membership of groups can be structured as: $g_c : c$, $g_{bc} : b, c$, $g_{abc} : a, b, c$. That is the group g_c has only one member c, the group g_{bc} has two members–b and c and so on.

References

[KPS02] Charlie Kaufman, Radia Perlman, and Mike Speciner. *Network Security—Private Communication in a Public World*. Prentice Hall PTR, 2nd edition, 2002.

[Pfl96] Charles P. Pfleeger. *Security in Computing*. Prentice Hall, iind edition, 1996.