Solutions to Exercises

Fundamentals

1. $X = \{(-1,0),(-1,1),(0,0),(0,1),(1,0),(1,1)\}$, $|X| = 6$ and $Y = \{1,2,3,4,5,6\}$. The map $f : X \to Y$ given by $f(-1,0) = 1$, $f(-1,1) = 2$, $f(0,0) = 3$, $f(0,1) = 4$, $f(1,0) = 5$ and $f(1,1) = 6$ is bijective.

2. a) f_1 is injective, $im(f_1) = \{3,5,7,\dots\}$, not surjective.
   b) f_2 is not injective, since for example $f_2(-1) = f_2(1) = 2$, but surjective.
   c) f_3 is bijective, $f_3^{-1} = f_3$.
   d) f_4 is neither injective nor surjective.

3. If f is injective, then $|im(f)| = |X| = |Y|$ and hence $im(f) = Y$, so that f is surjective.
   If f is surjective, then $|im(f)| = |Y| = |X|$. Hence f must be injective.

4. a) If $x \in f^{-1}(B)$, then $f(x) \in B$ which shows $f(f^{-1}(B)) \subset B$. Let $y \in B$. If f is surjective, then there exists $x \in X$ such that $f(x) = y$. Furthermore, we have $x \in f^{-1}(B)$, which gives $B \subset f(f^{-1}(B))$.
   b) Let $x \in A$. Then $f(x) \in f(A)$ and $x \in f^{-1}(f(A))$. If f is injective and $x \in f^{-1}(f(A))$, then $y = f(x) \in f(A)$. Since x is the only element with $f(x) = y$, we have $x \in A$.

5. $\mathbb{Z}_{26} = \{\overline{0},\overline{1},\dots,\overline{25}\}$. The standard representatives of the given integers modulo 26 are: 14, 22, 25, 15, 25 and 9.

6. $f_1 = O(n^3)$, polynomial.
   $f_2 = O(2^n)$, exponential.
   $f_3 = O(n^{1/2})$, polynomial.
   $f_4 = O(2^{-n/2})$, negligible.
   $f_5 = O(1/n)$, not negligible.
   $f_6 = O(2^{n/3})$ exponential.
   $f_7 = O(n)$, polynomial.

7. The exponent is significantly reduced by the cubic root. $f(n)$ is sub-exponential, but grows faster than any polynomial. Set $n = 2^b$ and compute the effective key length:
$$\log_2(f(2^b)) = \frac{2\ln(2^b)^{1/3}}{\ln(2)} = \frac{2\ln(2)^{1/3} \cdot b^{1/3}}{\ln(2)}$$
For $b = 128$ and $b = 1024$ this gives 12.87 and 25.74 effective key bits, respectively.
8. The number of ones (or zeros) in a uniform random byte follows a binomial distribution with $n = 8$. One has $\binom{8}{4} = 70$ and $Pr[Y = 4] = 70\left(\frac{1}{2}\right)^8 \approx 0.27$.
9. Since $Y = X_1 + \cdots + X_n$ (with $X_i = X$) and $E[X_i] = p$ one obtains $E[Y] = np$. The Bernoulli tries are independent, and from $V[X_i] = p(1-p)$ one concludes that $V[Y] = np(1-p)$.
10. The output of the new generator is uniformly distributed, but the bits are not independent since after 100 bits each output bit depends on the preceding 100 bits. So it is not a Random Bit Generator.
11. $\overline{0} = \overline{104}$, $\overline{3} = \overline{-49}$.
12. $\mathbb{Z}_{22}^* = \{1, 3, 5, 7, 9, 13, 15, 17, 19, 21 \bmod 22\}$. The corresponding inverses are $1, 15, 9, 19, 5, 17, 3, 13, 7, 21 \bmod 22$.
13. $897 : 32 = 28$, remainder 1. Hence $897 = 28 \cdot 32 + 1$ and $1 = 897 - 28 \cdot 32$. This gives $32^{-1} \equiv -28 \equiv 869 \bmod 897$.
14. The multiples of p and q are not invertible modulo pq. There are $q + p - 1$ such multiples in $\{0, 1, \ldots, pq - 1\}$, and so $\varphi(pq) = pq - (q + p - 1) = (p-1)(q-1)$.
15. $\varphi(2p) = p - 1$, $\varphi(2^m) = 2^{m-1}$, $\varphi(p^m) = (p-1)p^{m-1}$.
16. We use an algorithm for problem B to solve the factorization problem A. We can therefore assume that N and $\varphi(N)$ is given. Note that $\varphi(N) = N - p - q + 1$, and rearranging gives $p + q = N - \varphi(N) + 1$. Hence we can compute the sum $s = p + q$. We have $N = pq = p(s - p)$. This yields a quadratic equation in the unknown variable p, which can be easily solved in polynomial time.
$$p^2 - sp + N = 0$$
17. a) Fast exponentiation:
$$2^{55} \bmod 61 \equiv 2^{32} \cdot 2^{16} \cdot 2^4 \cdot 2^2 \cdot 2^1 \bmod 61 \equiv 57 \cdot 22 \cdot 16 \cdot 4 \cdot 2 \bmod 61 \equiv 21$$
b) Square-and-Multiply: SQ, MULT, SQ, SQ, MULT, SQ, MULT, SQ, MULT
$$2^{55} \bmod 61 \equiv ((((2^2 \cdot 2)^2)^2 \cdot 2)^2 \cdot 2)^2 \cdot 2 \bmod 61 \equiv 21$$
Therefore, 5 squarings and 4 multiplications modulo 61 are necessary.
18. For size (n) = size (k) = 2048, at most 2047 modular squarings and 2047 multiplications are required to compute $a^k \bmod n$.
19. $128 \cdot (-\frac{3}{4}\log_2(\frac{3}{4}) - \frac{1}{4}\log_2(\frac{1}{4})) \approx 103.8$.
20. Each octal number has 3 bits of entropy, so $\frac{80}{3} \approx 27$ octal numbers are required.
21. $1.2\sqrt{1000} \approx 38$ samples are likely to be sufficient for a collision.

Encryption Schemes and Definitions of Security

1. Suppose the plaintext and the key length is n. Hence, for any given plaintext m and ciphertext c of length n, there is exactly one key k such that $\mathcal{E}_k(m) = c$. The key is given by $k = m + c \bmod 26$. Hence $Pr[\mathcal{E}_k(m) = c] = \frac{1}{26^n}$ for all m and c, if the key is chosen uniformly at random. The cipher is therefore perfectly secure.
2. Arguments: a) It is difficult to keep an algorithm secret.
   b) The algorithm should be available for investigation by a large community.
   c) The designers may not be aware of weaknesses.
   Counter-arguments: a) The algorithm is not easily available to attackers.
   b) Vulnerabilities are more likely to remain unknown.
3. Suppose a key k of length n is used twice to encrypt plaintexts of length $2n$, i.e., the plaintext is XORed with $k\|k$. Let m_0 and m_1 be two different strings and c any string of length n. Then
   $$Pr[\mathcal{E}_k(m_0\|m_0) = (c\|c)] = \frac{1}{2^n}, \text{ but } Pr[\mathcal{E}_k(m_0\|m_1) = (c\|c)] = 0.$$
   If the ciphertext is $c\|c$, then a plaintext $m_0\|m_1$ with $m_0 \neq m_1$ is impossible. This implies that the cipher does not have perfect secrecy.
4. Let c be a given ciphertext. Perfect secrecy implies that any plaintext is possible (i.e., with probability > 0) for the given ciphertext c. Hence for every $m_0 \in \mathcal{M}$, there must exist a key $k \in \mathcal{K}$ such that $\mathcal{D}_k(c) = m_0$. Furthermore, different plaintexts must have different keys since the ciphertext is fixed. So there are at least as many keys as plaintexts.
5. A bit permutation is not perfectly secure: the number of zeros and ones is preserved by a bit permutation. Hence there exist many plaintext-ciphertext combinations (m, c) that never occur, which implies that $Pr[m|c] = 0$.
6. For $n = 256 = 2^8$, the scheme is $(2^{41}, 2^{-16})$-secure. This is not very secure in practice, since 2^{41} steps are easily possible on standard computers, and the success probability of 2^{-16} is not small enough. The scheme is not computational secure in the asymptotic definition, since for a polynomial number of computing steps ($2n^5$) the scheme can be broken with a non-negligible probability (n^{-2}).
7. In the EAV experiment, the adversary chooses two plaintexts of the same length and obtains a ciphertext from the oracle. The adversary has to find out which plaintext was encrypted. The CPA experiment is similar, but gives the adversary more power: he can freely choose any plaintext (even the two provided plaintexts) and gets the associated ciphertext from the oracle. Therefore, a deterministic scheme might be EAV-secure but not CPA-secure. On the other hand, CPA security implies EAV security.
8. Suppose the adversary A chooses m_0 and m_1 in the EAV experiment. If a scheme is perfectly secure the ciphertext $c = \mathcal{E}_k(m_b)$ does not provide any information about the plaintext. One has $Pr[m_0|c] = Pr[m_1|c]$ and hence $\mathrm{Adv}^{\mathrm{eav}}(A) = 0$ for any adversary. Perfect security is much stronger than EAV security: all plaintexts have the same probability if a ciphertext is given, which is not required for EAV security. A perfectly secure scheme cannot be broken even with unlimited resources. On the other hand, EAV security 'only' requires that an adversary

cannot find the correct plaintext (from two given candidates) in polynomial time with a probability $> \frac{1}{2} + \text{negl}(n)$.

9. Consider a Vigenère cipher of length n. Note that the plaintext of a Vigenère cipher can be longer than the key. The adversary chooses two plaintexts m_0, m_1 of length $2n$ such that $m_0 = x\|y$, $m_1 = x\|y'$ and $y \neq y'$. The challenger randomly chooses $b \in \{0,1\}$ and the adversary is given the ciphertext $c = \mathcal{E}_k(m_b) = m_b \oplus (k \,\|\, k)$. Then the adversary only has to check whether $c - m_0$ or $c - m_1$ (modulo 26) is a string of type $k \,\|\, k$. This reveals the correct plaintext. The adversary wins the experiment, and so the scheme is not EAV-secure.
10. Then $c_1 \oplus c_2 = m_1 \oplus G(k) \oplus m_2 \oplus G(k) = m_1 \oplus m_2$. Hence the ciphertext $c_1\|c_2$ reveals information about the plaintext $m_1\|m_2$. The encryption scheme is not EAV-secure.
11. (a) No, this does not contradict CPA security since e^{-n} is negligible.
    (b) Yes, since $\frac{1}{n^3}$ is the inverse of a polynomial of degree 3 and not negligible.
12. In a CCA2 attack against a malleable scheme, an adversary can ask for the decryption of a ciphertext c' that is related to the challenge ciphertext c. The adversary obtains the corresponding plaintext m' which is related to the correct plaintext. From this the adversary can infer the correct plaintext m_b, at least if the chosen plaintexts m_0 and m_1 are not related.
13. A family of bit permutations is not pseudorandom. The number of zeros and ones is preserved by a bit permutation, whereas a random permutation does not have this property. Therefore, an adversary can distinguish the output of a bit permutation from a random permutation. An adversary can choose m to be an all-zero string; if the oracle outputs a zero string c then almost certainly a bit permutation was used. If c is non-zero, then the output was not computed by a bit permutation.
14. Suppose the block length is l. The adversary chooses two binary strings x, y of length l with $x \neq y$ and sets $m_0 = x\|x$, $m_1 = x\|y$. If the oracle outputs a ciphertext of type $c\|c$, then the plaintext m_0 was encrypted. Otherwise, the oracle encrypted m_1. An adversary can thus easily win the EAV experiment. A block cipher in ECB mode is not EAV-secure.
15. No, since an adversary knows the initial IV (and all following IVs) in advance and can therefore predict the first ciphertext block under a chosen plaintext attack. Although different encryptions of the same plaintext give different ciphertexts, the scheme is deterministic because the IV is not randomized.
16. Suppose the plaintexts m_0 and m_1 are encrypted in CTR mode using the same counter. If the ciphertexts are c_0 and c_1, then $c_0 \oplus c_1 = m_0 \oplus m_1$. An adversary who eavesdrops two ciphertexts learns the XOR combination of the associated plaintexts. The scheme can be broken with a simple ciphertext-only attack. The scheme is no longer randomized if the counter is re-used.
17. It cannot be perfectly secure, since the key space is smaller than the message space (strings of arbitrary length). There are many plaintext-ciphertext pairs that never occur if the key is shorter than the message.
18. a) $c = 0111\ 0010\ 1000$
    b) $c = 1010\ 0010\ 0110\ 0100$

c) $c = 1010\ 1100\ 1000\ 1111$
In part b) and c), the IV or the CTR value is the first ciphertext block c_0.

19. a) IV is missing: m_1 cannot be computed, but all following blocks can be decrypted correctly.
b) Transmission error in block c_k: m_k and m_{k+1} are faulty, but the following blocks are correct.
c) Bit error in block c_k: m_k is faulty and m_{k+1} has a bit error. The following blocks are correct.
d) Ciphering error in block c_k: the following ciphertext blocks are also faulty, m_k and all following plaintext blocks are incorrect.
Note the different consequences of errors during transmission and ciphering. In the latter case, the faulty ciphertext block c_k corrupts the computation of all subsequent ciphertext blocks.

20. The plaintext blocks are m_1, m_2 and m_3, where the last block contains only 44 bits. The ciphertext blocks are $c_0 = ctr$, c_1, c_2, c_3.
a) First counter bit is flipped: all decrypted plaintext blocks are faulty since all counter values are incorrect.
b) Bit error in the first bit of c_1: only the first plaintext bit is faulty and bits $2 - 300$ are correct.
c) Error in block c_1: the block m_1 (bits $1 - 128$) is faulty and the plaintext bits $129 - 300$ are correct.
d) Bit error in the last bit of c_3: the plaintext bits $1 - 299$ are correct and bit 300 is faulty.

21.

	ECB	CBC	CTR
Message expansion	no	extra IV	extra counter
Error propagation	no	yes	no
Pre-computation	no	no	yes
Parallelization	yes	only decryption	yes

22. We use the fact that CBC and CTR modes are malleable. In the CCA2 experiment, an adversary chooses any two unrelated plaintexts consisting each of one block. Suppose the oracle selects m and the challenge ciphertext is $c = c_0 \| c_1$, where c_0 is the initialization vector (CBC mode) or the counter (CTR mode).

First, consider a cipher in CBC mode. Suppose $e_1 = 10 \ldots 0$ is one block that is all-zero except for the first bit. The adversary chooses the ciphertext

$$c' = (c_0 \oplus e_1) \parallel c_1.$$

Since $c \neq c'$, the oracle accepts c' in a chosen-ciphertext attack and gives the associated plaintext

$$m' = E_k^{-1}(c_1) \oplus c_0 \oplus e_1$$

to the adversary. But m' is equal to $m \oplus e_1$, so that the adversary can easily select the correct plaintext and win the CCA2 game.

Now consider the CTR mode. The adversary chooses $c' = c_0 \parallel (c_1 \oplus e_1)$, where $e_1 = 10 \ldots 0$ is as above. The oracle computes

$$m' = E_k(c_0 + 1) \oplus c_1 \oplus e_1.$$

Again, we have $c \neq c'$. The adversary obtains the corresponding plaintext $m' = m \oplus e_1$ and easily wins the CCA2 game.

Symmetric Ciphers

1. Block ciphers transform a block of plaintext data of fixed length. Stream Ciphers generate a pseudorandom key stream, and encryption is done by XORing the plaintext data and the keystream.

2. No, the counter ctr is not secret. The key blocks $E_k(ctr)$ and $E_k(ctr+1)$ are independent of each other.

3. Linear and affine maps can produce diffusion. However, the associated matrix and the translation vector can be recovered from input and output data, and therefore the round keys are not sufficiently protected. Hence diffusion cannot be achieved solely by linear maps and adding round keys.

4. The rounds are important in order to achieve sufficient diffusion and confusion and to protect against known plaintext attacks. An SPN with a single round without final key mixing can be easily broken with a known plaintext attack: apply P^{-1} and then S^{-1} to the ciphertext. This gives the XOR-sum of the plaintext and the first round key, from which the key can be computed if the plaintext is known.

5. The diffusion operations are MixColumns and ShiftRows. The AES S-Box and AddRoundKey provide confusion.

6. The hexadecimal numbers `01`, `02` and `03` correspond to the polynomials 1, x and $1+x$ mod $x^8+x^4+x^3+x+1$, respectively. Obviously, the inverse of 1 is 1. Furthermore, `8D` corresponds to $x^7+x^3+x^2+1$ and `F6` corresponds to $x^7+x^6+x^5+x^4+x^2+x$. Multiplying these polynomials with x and $1+x$, respectively, and reducing mod $x^8+x^4+x^3+x+1$ shows that the inverse of `02` is `8D` and the inverse of `03` is `F6`. Then $S_{RD}(\texttt{01}) = A \cdot \texttt{01} + b = \texttt{7C}$, $S_{RD}(\texttt{02}) = A \cdot \texttt{8D} + b = \texttt{77}$ and $S_{RD}(\texttt{03}) = A \cdot \texttt{F6} + b = \texttt{7B}$.

7. Since $(x+y)^2 = x^2 + 2xy + 2y \equiv x^2 + y^2 \bmod 2$, the functions $f(x) = x^2$, $f(x) = x^4$, x^8, ... are $GF(2)$-linear. However, if n is not a power of 2 then $f(x) = x^n$ is not $GF(2)$-linear. Consider $n = 254$. Since $GF(2^8)^*$ is a group of order 255, we have $x^{255} = 1$ and thus $x^{254} = x^{-1}$ (for $x \neq 0$). The function $f(x) = x^{254}$ is not linear: $\texttt{01} \oplus \texttt{02} = \texttt{03}$, but $\texttt{01}^{-1} \oplus \texttt{02}^{-1} = \texttt{01} \oplus \texttt{8D} = \texttt{8C}$ is not equal to $\texttt{03}^{-1} = \texttt{F6}$.

8. One has $S_{RD}(a) = y = Aa^{-1} + b$ for $a \in GF(2^8)^*$. This gives $y - b = Aa^{-1}$ and

$$a^{-1} = A^{-1}(y-b) = A^{-1}(y+b).$$

   Note that $+1 = -1$ over $GF(2)$ and $GF(2^n)$. One computes:

$$A^{-1} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}, \quad A^{-1}b = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$

Therefore, $S_{RD}^{-1}(y) = a = (A^{-1}(y+b))^{-1}$ for $y \neq b$. So the inverse SubByte operation is given by the affine map $A^{-1}x + A^{-1}b$, followed by the multiplicative inversion in $GF(2^8)^*$. For $y = b$, one sets $S_{RD}^{-1}(b) = \texttt{00}$.

9. We give a high-level description of the AES decryption function f_k^{-1} in pseudocode:

```
Inverse-Rijndael(State, CipherKey)
{
   KeyExpansion(CipherKey, ExpandedKey)
   // Inverse of Final Encryption Round
   AddRoundKey(State,ExpandedKey[Nr])
   Inverse ShiftRows(State)
   Inverse  SubBytes(State)
   for(i = Nr - 1; i >= 1 ; i--)  {
      // Inverse of Encryption Round i
      AddRoundKey(State,ExpandedKey[i])
      Inverse MixColumns(State)
      Inverse ShiftRows(State)
      Inverse SubBytes(State)
    }
    AddRoundKey(State,ExpandedKey[0])
}
```

10. The *diffusion* operations that spread small changes throughout the complete state of 128 bits would be missing. AES would be reduced to addition of the round key and the S-Box transformation. Changing *one plaintext byte* would change only *one ciphertext byte*. Encryption of each of the 16 plaintext bytes in a block would be independent of each other. The output of 256 chosen input blocks $\texttt{00}\ldots\texttt{00}$, $\texttt{01}\ldots\texttt{01}$, $\ldots$, $\texttt{FF}\ldots\texttt{FF}$ were sufficient to determine the output of any input block. The security of AES would be significantly reduced. The modified block cipher is not a pseudorandom permutation, since an adversary can easily distinguish it from a random permutation: an adversary could choose two input blocks which differ only in one byte. If the output blocks also differ only in one byte, then the modified block cipher was used since a random permutation would not have this property. The adversary therefore wins the *prp* experiment.

    If only the ShiftRows operations were missing, the transformation of the four state columns would be independent of each other. Similarly, if the MixColumns operations were missing, the transformation of the four state rows would be independent of each other. On the same lines as above, an adversary can distinguish the modified block ciphers from a random permutation.

11. All other operations are affine and their composition is also affine. Thus the encryption map would be affine altogether and *confusion* would be missing. A known plaintext attack with 129 plaintext-ciphertext pairs can be conducted. The attack requires only linear algebra and is therefore very efficient. The modified cipher is not a pseudorandom permutation: an adversary can easily find out whether a function is affine or not if they can choose input blocks and get the corresponding output blocks.

12. The nonlinearity of encryption and decryption is essential, since otherwise the cipher would be completely insecure. The nonlinearity of the key schedule is desirable.

Hash Functions

1. Assume H is collision-resistant. Then an efficient adversary cannot compute any collision $H(x) = H(x')$, $x \neq x'$, a fortiori if x and its hash value $H(x)$ is fixed. This gives second-preimage resistance.
2. Suppose a function $f : D \to R$ is given, where $f(x) = l(x) + b$ is affine, $l(x)$ is linear and $|D| > |R|$. Then the kernel of the $GF(2)$-linear map l is nontrivial and a vector $v \in D$, $v \neq 0$ with $l(v) = 0$ can be efficiently computed using Gaussian elimination. We have $f(v) = f(0) = b$, and so f is not collision-resistant.
3. Suppose the hash value would not depend on one of the input bits. Then a collision can be produced by choosing any message and flipping this particular bit. Both messages would have the same hash value.
4. a) No, since the hash value would not depend on the low bit.
   b) and c) Yes, since a collision of the modified hash function would yield a collision of the original hash function. However, we assumed that the original hash function is collision-resistant.
   d) No, since XORing the blocks produces collisions: let H' be the modified hash function k the block length. Choose any non-zero word a of length k with $a \neq 0^k$. Then $H'(a\|a) = H(a \oplus a) = H(0^k) = H'(0^k\|0^k)$. This shows that H' is not collision-resistant.
5. $1 - \frac{99}{100} \cdot \frac{98}{100} \cdot \cdots \cdot \frac{91}{100} \approx 0.37$
6. a) The inequality $(1 - \frac{i}{n}) \leq e^{-i/n}$ and

$$\prod_{i=0}^{k-1} e^{-i/n} = e^{\sum_{i=0}^{k-1}(-i/n)} = e^{-k(k-1)/(2n)} \ .$$

   imply that the probability that no collision occurs is less than $e^{-k(k-1)/(2n)}$. We conclude that the probability of a collision is at least

$$1 - e^{-k(k-1)/(2n)} \ .$$

   b) For a probability of $\frac{1}{2}$, we get

$$\frac{1}{2} = e^{-k(k-1)/(2n)} \iff \ln(2) = \frac{1}{2n} k(k-1) \ .$$

   We approximate $k(k-1)$ by k^2 and obtain $2n\ln(2) = k^2$, so that

$$k = \sqrt{2n\ln(2)} \approx 1.2\sqrt{n} \ .$$

   This is the approximate number of random values such that a collision occurs with a probability of at least $\frac{1}{2}$.
7. A collision is likely to occur after around 2^{40} hashes.
8. $H(m)$ is defined as the output of the last Merke-Damgård iteration *after padding* the message m. The padded message is $m\|10\ldots0\|L$, where L is the encoded length of m. The hash $H(m)$ is given by the last Merke-Damgård compression of the padded message. Let m' be any chosen message. Now the hash of the extended message

$$m\|10\ldots0\|L\|\ m'$$

can be computed from $H(m)$, L and m' without knowing m, simply by continuing with Merke-Damgård iterations. In fact, the hash of the extended message is the output of the last Merke-Damgård compression of the padded message $m\|10\ldots0\|L\|m'\|10\ldots0\|L'$, where L' is the length of the extended message.

9. The length padding ensures that a collision-resistant compression function yields a collision-resistant Merkle-Damgård hash function. In other words, a collision in H would give a collision in the compression function: if two inputs have different lengths, then their last blocks are different after length padding. In this case, a collision in H implies a collision in the last Merkle-Damgård iteration. On the other hand, if two different inputs of the same length have identical hash values, then there must be a collision in one of the iterations of the compression function.
10. A length-extension attack would require the full state. However, SHA-3 outputs only a part of the state (the rate). The other part (the capacity) has at least 256 bits and is not output.

Message Authentication Codes

1. Possible reasons are:
   a) The message was modified (error or changed by an adversary).
   b) The MAC tag is faulty (error or changed by an adversary).
   c) A computational error occurred during MAC generation or verification.
   d) The wrong key was used; the keys for MAC generation and verification do not match.
2. Both can be used to protect the integrity of messages against random events and errors. However, an adversary might change a message *and* the hash. In the presence of active adversaries, the authentic hash value of the original message is required to verify the integrity. In contrast, authenticity of a MAC is not a precondition, since an adversary should not be able to forge a valid MAC without the secret key.
3. An adversary chooses any message m and asks the oracle for the corresponding tag t. Then the adversary defines m' which differs from m only in the first bit. The tag t is also valid for m'. This message was not queried before and the adversary wins the MAC forgery experiment.
4. Suppose the tag is defined by appending the encoded message length and computing the CBC MAC. Let l be the block length and let L be the encoding of l. Then the tags of 0^l and 1^l are $t_0 = E_k(L \oplus E_k(0^l))$ and $t_1 = E_k(L \oplus E_k(1^l))$, respectively. Define messages $m_0 = 0^l \,\|\, L \|\, t_0$ and $m_1 = 1^l \,\|\, L \|\, t_1$ of length $3l$. One checks that the tag of both m_0 and m_1 is $E_k(L' \oplus E_k(0^l))$, where L' is the encoding of $3l$. An adversary asks for the tags of 1^l and m_0 and can thus provide a valid tag of m_1.
5. This tag would always be valid: let $c_1, \ldots, c_{N-1}, c_N$ be the ciphertext blocks and assume the last block c_N is also used as CBC MAC. Decryption in CBC mode gives the plaintext blocks $m_1, \ldots, m_N$. The last block is $m_N = E_k^{-1}(c_N) \oplus c_{N-1}$. In order to verify the tag, one would encrypt the plaintext and always obtain c_N, even if the ciphertext has been tampered with:
$$E_k(m_N \oplus c_{N-1}) = E_k(E_k^{-1}(c_N) \oplus c_{N-1} \oplus c_{N-1}) = c_N$$
6. The construction is susceptible to a length-extension attack: an adversary can compute $H(k\|m\|m')$ without knowing k. HMAC is not affected by this attack, since an adversary only controls the output of the inner hash function but not the outer hash function. The outer hash function takes a derived message of fixed length as input.
7. If the hash function behaves as a random oracle then HMAC gives a pseudorandom function.
8. Choose a uniform random IV of length 96 and set $ctr = IV\|0^{31}\|1$. Then $m = m_1$, $c_1 = E_k(ctr+1) \oplus m_1$, $c = IV\|c_1$, $H = E_k(0^{128})$, $A = 0^{128}$, $X_1 = 0^{128}$, $X_2 = c_1 \cdot H$, $X_3 = (X_2 \oplus (0^{64}\|0^{56}10000000)) \cdot H$. The tag is $t = X_3 \oplus E_k(ctr)$ and the authenticated ciphertext is (c, t).
9. An adversary can easily attack a block cipher in counter mode in a chosen ciphertext attack: let c be a challenge ciphertext in the CCA experiment. The adversary changes one ciphertext bit and sends the modified ciphertext c' to

the oracle, who returns the corresponding plaintext m'. The adversary flips the same bit as before and obtains the plaintext m.

This attack is not possible when the ciphertext is combined with a secure MAC (encrypt-then-authenticate approach), since the adversary cannot produce a valid tag of c'. Although the ciphertexts c and c' (and their corresponding plaintexts) differ only in one bit, their tags are not related. An adversary who knows the tag of c (as part of the challenge ciphertext) is unable to produce a valid tag of c'. Any chosen ciphertext is most likely to be invalid, so that the oracle only returns the error symbol $\perp$ and not the corresponding plaintext. In other words, appending a secure MAC prevents an adversary from leveraging the decryption oracle.

Algebraic Structures

1. L is the set of all integer linear combinations of the linearly independent vectors $(2,0)$ and $(1,1)$. This is an additive commutative group. The lattice contains all points of $\mathbb{Z}^2$ such that the coordinates are either both even or both odd.
2. The subgroups of $(\mathbb{Z}_{10},+)$ are $\{\overline{0}\}$, $<\overline{2}>$, $<\overline{5}>$ and $<\overline{1}>=\mathbb{Z}_{10}$. The only subgroups of $(\mathbb{Z}_{11},+)$ are $\{\overline{0}\}$ and $\mathbb{Z}_{11}$. One has an isomorphism $\mathbb{Z}_{10}\cong\mathbb{Z}_{11}^*$. Since $\overline{2}$ is a generator of $\mathbb{Z}_{11}^*$, the map $f(k \bmod 10)=2^k \bmod 11$ is an isomorphism. The subgroups of $\mathbb{Z}_{11}^*$ are $\{\overline{1}\}$, $<\overline{4}>$, $<\overline{10}>$ and $<\overline{2}>=\mathbb{Z}_{11}^*$.
3. The possible orders are $1,2,3,6,9,18,27,54$.
4. Homomorphism: $f(x_1+x_2)=5(x_1+x_2)=5x_1+5x_2=f(x_1)+f(x_2)$. f is an isomorphism since f has an inverse map $f^{-1}(x)=4x \bmod 19$.
5. Suppose $m\in\mathbb{Z}_n^*$. Since ord $(\mathbb{Z}_n^*)=(p-1)(q-1)$, the congruence follows from Euler's Theorem for the group $\mathbb{Z}_n^*$. By reducing modulo p or modulo q, the statement follows for any $m\in\mathbb{Z}_n$.
6. ord $(\mathbb{Z}_{23}^*)=22$. We have $2^{11} \bmod 23\equiv 1$ and hence ord $(\overline{2})=11$. Since $5^{11} \bmod 23\equiv 22$ and $5^2 \bmod 23\equiv 2$, we obtain ord $(\overline{5})=22$ (maximal) so that $\overline{5}$ is a generator of $\mathbb{Z}_{23}^*$.
7. Since $2^9\equiv 18 \bmod 19$, we have ord $(\overline{2})=18$ (a generator of $\mathbb{Z}_{19}^*$). On the other hand, $5^9\equiv 1 \bmod 19$ and hence ord $(\overline{5})=9$ (not a generator).
8. The order of any element in $\mathbb{Z}_n\times\mathbb{Z}_n$ is less than or equal to n. Since ord $(\mathbb{Z}_n\times\mathbb{Z}_n)=n^2$, this group cannot be cyclic.
9. $\mathbb{Z}_2^3=\mathbb{Z}_2\times\mathbb{Z}_2\times\mathbb{Z}_2$, $\mathbb{Z}_2\times\mathbb{Z}_4$ and $\mathbb{Z}_8$. Only $\mathbb{Z}_8$ is a cyclic group.
10. ord $(\mathbb{Z}_{12}^*)=4$. Note that $\mathbb{Z}_{12}\cong\mathbb{Z}_4\times\mathbb{Z}_3$ and $\mathbb{Z}_{12}^*\cong\mathbb{Z}_4^*\times\mathbb{Z}_3^*$ (two cyclic groups of order 2).
    ord $(\mathbb{Z}_{23}^*)=22$ since 23 is prime. $\mathbb{Z}_{23}^*$ is cyclic of order 22. By the Chinese Remainder Theorem, the group is isomorphic to $\mathbb{Z}_{11}\times\mathbb{Z}_2$.
11. $a \bmod n$ is a generator of the additive group $\mathbb{Z}_n$ if and only if $\gcd(a,n)=1$.
12. $p=13$ and $q=19$. The Extended Euclidean Algorithm gives $1=3p-2q$ and hence $k=7\cdot(-2)q+2\cdot 3p=59$.
13. $(1,0)$ and $(0,1)$ are non-zero and not invertible in $R_1\times R_2$.
14. a) 16 elements. It is not a field since $1+x^2+x^4=(1+x+x^2)^2$ is reducible in $GF(2)[x]$.
    b) 9 elements. It is a field since $1+x^2$ is irreducible in $GF(3)[x]$; the polynomial does not have zeros over $GF(3)$.
    c) p^3 elements. It is not a field since $x^3-1=(x-1)(x^2+x+1)$ is reducible.
15. R is not a field since $x^2-4=(x+2)(x-2)$ is reducible. R is represented by polynomials a_1x+a_0, where $a_0,a_1\in GF(17)$. Hence R contains $17^2=289$ elements. R maps to $GF(17)^2$ by sending a polynomial f to $(f(2),f(-2))$. This is a ring homomorphism. For the inverse map, let $(y_0,y_1)\in GF(17)^2$. We need to find a polynomial $f(x)$ of degree 1 or 0 such that $f(-2)=y_0$ and $f(2)=y_1$. We may use Lagrange interpolation and obtain:

$$f(x)=y_0\cdot\frac{x-2}{-4}+y_1\cdot\frac{x+2}{4}$$

This gives the inverse ring homomorphism.

16. $x^3 \equiv x+1$, $x^4 \equiv x^2+x$, $x^5 \equiv x^2+x+1$, $x^6 \equiv x^2+1$, $x^7 \equiv 1$ modulo $1+x+x^3$.

17. For example, $h(x) = 1+x+x^6$ or $h(x) = 1+x^3+x^6$. Both polynomials have no zeros over $GF(2)$. One can manually check that they are not divisible by any polynomial of degree 2 and 3. They are therefore irreducible. We may also use Sage.

```
sage: R.<x> = PolynomialRing(GF(2), 'x')
sage: h=1+x+x^6; h.is_irreducible()
True
sage: h=1+x^3+x^6; h.is_irreducible()
True
```

18. We factorize $x^{256}-x$ over $GF(2)$.

```
sage: R.<x> = PolynomialRing(GF(2), 'x')
sage: R(x^256-x).factor()
x * (x + 1) * (x^2 + x + 1) * (x^4 + x + 1) * (x^4 + x^3 + 1) *
(x^4 + x^3 + x^2 + x + 1) * (x^8 + x^4 + x^3 + x + 1) *
(x^8 + x^4 + x^3 + x^2 + 1) * (x^8 + x^5 + x^3 + x + 1) *
(x^8 + x^5 + x^3 + x^2 + 1) *
(x^8 + x^5 + x^4 + x^3 + 1) *
(x^8 + x^5 + x^4 + x^3 + x^2 + x + 1) *
(x^8 + x^6 + x^3 + x^2 + 1) *
(x^8 + x^6 + x^4 + x^3 + x^2 + x + 1) *
(x^8 + x^6 + x^5 + x + 1) *
(x^8 + x^6 + x^5 + x^2 + 1) * (x^8 + x^6 + x^5 + x^3 + 1) *
(x^8 + x^6 + x^5 + x^4 + 1) *
(x^8 + x^6 + x^5 + x^4 + x^2 + x + 1) *
(x^8 + x^6 + x^5 + x^4 + x^3 + x + 1) *
(x^8 + x^7 + x^2 + x + 1) *
(x^8 + x^7 + x^3 + x + 1) * (x^8 + x^7 + x^3 + x^2 + 1) *
(x^8 + x^7 + x^4 + x^3 + x^2 + x + 1) *
(x^8 + x^7 + x^5 + x + 1) * (x^8 + x^7+ x^5 + x^3 + 1) *
(x^8 + x^7 + x^5 + x^4 + 1) *
(x^8 + x^7 + x^5 + x^4 + x^3 + x^2 + 1) *
(x^8 + x^7 + x^6 + x + 1) *
(x^8 + x^7 + x^6 + x^3 + x^2 + x + 1) *
(x^8 + x^7 + x^6 + x^4 + x^2 + x + 1) *
(x^8 + x^7 + x^6 + x^4 + x^3 + x^2 + 1) *
(x^8 + x^7 + x^6 + x^5 + x^2 + x + 1)*
(x^8 + x^7 + x^6 + x^5 + x^4 + x + 1) *
(x^8 + x^7 + x^6 + x^5 + x^4 + x^2 + 1) *
(x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1)
```

The seventh factor is $x^8+x^4+x^3+x+1$.

19. $f(x) = x$, $h(x) = x^7+x^3+x^2+1$. Then $f(x) \cdot h(x) \equiv x^8+x^4+x^3+x \equiv 1 \bmod (x^8+x^4+x^3+x+1)$. $h(x)$ corresponds to `8D`.

Public-Key Encryption and the RSA Cryptosystem

1. An adversary can generate a list of plaintexts and associated ciphertexts, if the scheme is deterministic and the plaintext space is small. This requires only the public key. Then eavesdropped ciphertexts can be decrypted using that list.
2. Generate a large random prime p and a number e with $1 < e < p$. Compute d with $ed \equiv 1 \bmod p-1$. The encryption key is $k = (e, p)$. The plaintext and ciphertext space is $\mathbb{Z}_p^*$, the encryption function is $\mathcal{E}_k(m) = m^e \bmod p$, and the decryption function is $\mathcal{D}_k(c) = c^d \bmod p$. The scheme provides correct decryption. However, the decryption key $d \equiv (e \bmod (p-1))^{-1}$ can be easily derived from e and $p-1$ using the Extended Euclidean Algorithm. Therefore, it is insecure as a public-key scheme. It is rather a secret-key scheme.
3. a) $c = 100^5 \bmod 437 \equiv 85$.
   b) $N = 19 \cdot 23$, $\varphi(N) = 18 \cdot 22 = 396$, $d \equiv \frac{1}{5} \bmod 396 = \frac{1+4\cdot 396}{5} = 317$. Alternatively, run the Extended Euclidean Algorithm on input 396 and 5 and obtain $1 = 396 - 79 \cdot 5$. Hence $d = -79 \equiv 317 \bmod 396$.
   c) $m = c^d = 85^{317} \bmod 437 \equiv 100$.
4. Suppose $x \in \mathbb{Z}_N$ is chosen uniformly at random and $x \notin \mathbb{Z}_N^*$. Then an integer representative of x must be zero or a multiple of p or q. Since $\text{size}(p) = \text{size}(q) = n$ and $\text{size}(N) = 2n$, the probability of this being the case is negligible in n.
5. a) $c = m^e \bmod N \equiv 66^{35} \equiv 66^{32} \cdot 66^2 \cdot 66 \equiv 35 \cdot 157 \cdot 66 \equiv 264 \bmod 323$. We use fast exponentiation: the sequence of modular squares of 66 is 157, 101, 188, 137, 35.
   b) $c_1 c_2 \bmod N \equiv 26 \cdot 213 \equiv 47 \bmod 323$ is the ciphertext of $m_1 m_2$, since $(m_1 m_2)^e = m_1^e m_2^e = c_1 c_2 \bmod 323$.
   Similarly, $c_1 c_2^{-1} \bmod N$ is the ciphertext of $m_1 m_2^{-1}$. We compute $c_2^{-1} \bmod N \equiv (213 \bmod 323)^{-1}$ by running the Extended Euclidean Algorithm on input $N = 323$ and $c_2 = 213$.

$323 : 213 = 1$ rem. 110	$323 = 213 + 110$	$110 = 323 - 213$
$213 : 110 = 1$ rem. 103	$213 = 110 + 103$	$103 = 213 - 110$
$110 : 103 = 1$ rem. 7	$110 = 103 + 7$	$7 = 110 - 103$
$103 : 7 = 14$ rem. 5	$103 = 14 \cdot 7 + 5$	$5 = 103 - 14 \cdot 7$
$7 : 5 = 1$ rem. 2	$7 = 5 + 2$	$2 = 7 - 5$
$5 : 2 = 2$ rem. 1	$5 = 2 \cdot 2 + 1$	$1 = 5 - 2 \cdot 2$

This yields:

$$\begin{aligned}\mathbf{1} &= 5 - 2 \cdot 2 = 5 - 2 \cdot (7 - 5) = 3 \cdot 5 - 2 \cdot 7 = 3 \cdot (103 - 14 \cdot 7) - 2 \cdot 7 \\ &= 3 \cdot 103 - 44 \cdot 7 = 3 \cdot 103 - 44 \cdot (110 - 103) = -44 \cdot 110 + 47 \cdot 103 \\ &= -44 \cdot 110 + 47 \cdot (213 - 110) = 47 \cdot 213 - 91 \cdot 110 = 47 \cdot 213 - 91 \cdot (323 - 213) \\ &= -91 \cdot 323 + 138 \cdot 213\end{aligned}$$

Hence $(213 \bmod 437)^{-1} \equiv 138$. The ciphertext is $26 \cdot 138 \bmod 323 \equiv 35$.

c) Mallory computes $y = s^e \bmod N = 5^{35} \bmod 323 \equiv 23$. Then

$$c' = y \cdot c \bmod N = 23 \cdot 104 \bmod 323 \equiv 131.$$

In a chosen ciphertext attack, he asks Bob to decrypt the ciphertext $c' = 131$. Bob returns $m' = (yc)^d = y^d c^d = sm \equiv 142$. Mallory computes $(5 \bmod 323)^{-1} \equiv 194$ and finds the plaintext

$$m = s^{-1}m' \equiv 5^{-1} \cdot 142 \equiv 194 \cdot 142 \bmod 323 \equiv 93.$$

d) Fermat factorization of $N = 437$ gives $p = 19$ and $q = 17$. Then $\varphi(N) = 288$. We have $e = 35$ and compute $(35 \bmod 288)^{-1}$ using the Extended Euclidean Algorithm (alternatively, use the approach $\frac{1}{35} \bmod 288 = \frac{1+k\cdot 288}{35}$).

$288 : 35 = 8$ rem. 8	$288 = 8 \cdot 35 + 8$	$8 = 288 - 8 \cdot 35$
$35 : 8 = 4$ rem. 3	$35 = 4 \cdot 8 + 3$	$3 = 35 - 4 \cdot 8$
$8 : 3 = 2$ rem. 2	$8 = 3 \cdot 2 + 2$	$2 = 8 - 3 \cdot 2$
$3 : 2 = 1$ rem. 1	$3 = 2 + 1$	$1 = 3 - 2$

This gives:

$$\begin{aligned}\mathbf{1} &= 3 - 2 = 3 - (8 - 3 \cdot 2) = -8 + 3 \cdot 3 = -8 + 3 \cdot (35 - 4 \cdot 8) \\ &= -13 \cdot 8 + 3 \cdot 35 = -13(288 - 8 \cdot 35) + 3 \cdot 35 = -13 \cdot 288 + 107 \cdot 35\end{aligned}$$

Hence $d = 107$.

6. a) The given sequence of modular squarings and multiplications yields the binary expansion of the private key. The first bit (MSB) must be 1. For the next bits, SQ (without MULT) corresponds to 0 and the combination SQ, MULT corresponds to 1. Hence $d = 1000\ 0110\ 0011 = 2147$.
   b) $ed - 1 = 23616$ is a multiple of $\varphi(N)$, say $ed - 1 = k\varphi(N)$ with $k \in \mathbb{N}$. This gives the integer equation $\varphi(N) = \frac{ed-1}{k}$. One computes $\frac{ed-1}{k}$ for small integers k. The result should be somewhat smaller than N. For $k = 3$ one obtains $\frac{ed-1}{3} = 7872 = \varphi(N) = (p-1)(q-1) = pq - p - q + 1$. This implies $p+q = N+1-\varphi(N) = 180$, but p and q are still unknown. Since $pq = N = 8051$, the factors p and q are the zeros of the quadratic equation $x^2 - 180x + 8051$. This implies $x = 90 \pm \sqrt{90^2 - 8051}$. The solutions are $x_1 = p = 97$ and $x_2 = q = 83$.
7. The Fermat test is correct, since $a^{p-1} \equiv 1 \bmod p$ for prime numbers p and $a \not\equiv 0 \bmod p$. Now consider $n = 561 = 3 \cdot 11 \cdot 17$. Then $\mathbb{Z}_{561}^* \cong \mathbb{Z}_3^* \times \mathbb{Z}_{11}^* \times \mathbb{Z}_{17}^*$. The group orders are 2, 10 and 16, respectively. We have $n - 1 = 560$ and $560 \equiv 0 \bmod 2$, $560 \equiv 0 \bmod 10$ and $560 \equiv 0 \bmod 16$. Therefore, $a^{560} \equiv 1 \bmod n$ for all $a \in \mathbb{Z}_n^*$, and so 561 is a Carmichael number.
8. $n - 1 = 262 = 2 \cdot 131$. Hence $d = 131$ and $s = 1$.
   Let $a = 3$. Then $gcd(3, 263) = 1$ and we compute $3^{131} \equiv 1 \bmod 263$. The Miller-Rabin algorithm outputs that n could be a prime.
   Now let $a = 5$. Then $gcd(5, 263) = 1$ and we have $5^{131} \equiv 262 \equiv -1 \bmod 263$. Again, the test outputs that n could be a prime. In fact, $n = 263$ is a prime.
9. $c = m^e \bmod n = 2314^5 \bmod 10573 \equiv 6637$.
   $\sqrt{10573} = 102.8$, $103^2 = 10609$, $10609 - 10573 = 36 = 6^2$. Hence

$$10573 = (103 - 6)(103 + 6) = 97 \cdot 109.$$

   We have $(p-1)(q-1) = 10368$. Then $gcd(5, 10368) = 1$, i.e., $e = 5$ is admissible, whereas $gcd(3, 10368) = 3$, i.e., $e = 3$ is not admissible.
   We have $d = (5 \bmod 10368)^{-1} \equiv 6221 \bmod 10368$. Then
   $d_p = d \bmod (p-1) \equiv 77$, $d_q = d \bmod (q-1) \equiv 65$, $c_p = c \bmod p \equiv 41$,

$c_q = c \bmod q \equiv 97$, $m_p = c_p^{d_p} \bmod p \equiv 83$, $m_q = c_q^{d_q} \bmod q \equiv 25$.
Running the Extended Euclidean Algorithm on input $p = 97$ and $q = 109$ gives

$$1 = 9p - 8q = 9 \cdot 97 - 8 \cdot 109.$$

Then the Chinese Remainder Theorem (CRT) gives

$$m = m_q(9p) + m_p(-8q) = -50551 \equiv 2314 \bmod 10573.$$

Without the CRT, we would have to compute

$$m = c^d \bmod N = 6637^{6221} \equiv 2314 \bmod 10573.$$

10. In this case, a divisor of N_1 and N_2 can be easily computed: $\gcd(N_1, N_2) = 10007$. Both moduli are insecure and we obtain the factorizations $N_1 = 10007 \cdot 10133$ and $N_2 = 10007 \cdot 11003$.

11. Since $(m^2)^e = (m^e)^2 = c^2 \bmod N$, the adversary only needs to square the RSA ciphertext $\bmod N$. However, if an adversary modifies an RSA-OAEP ciphertext, then the decrypted data block DB which contains the plaintext m is almost certainly invalid. Tampering with the ciphertext results in a decryption error.

12. The ciphertext is $c = 2090^5 \bmod 10057 \equiv 1981$. Factorization with Fermat's method gives $N = 89 \cdot 113$, hence $\varphi(N) = 88 \cdot 112 = 9856$. Then $d = (5 \bmod 9856)^{-1} \equiv 7855$.

13. Since $c = m^e = m^{(2^{16})} \cdot m \bmod N$, the encryption requires 17 modular multiplications. For decryption, the exponent d has almost the same size as N, i.e., 1024 bits. Thus the computation of $c^d \bmod N$ requires (at most) 1023 quadratures and 1023 multiplications modulo N, if fast exponentiation or the square-and-multiply algorithm is used. Hence at most 2046 modular multiplications are needed.

14. We have $\lceil \sqrt{N} \rceil = 46$. We are looking for smooth numbers $x^2 - N$ with respect to a factor base consisting of 2, 3, 5 and 7.

```
sage: N=2041
sage: for x in range(46,60):
            s=factor(x^2-N)
            print ("x={:2}, x^2-N ={:4} = {}".format(x,x^2-N, s))
x=46, x^2-N =  75 = 3 * 5^2
x=47, x^2-N = 168 = 2^3 * 3 * 7
x=48, x^2-N = 263 = 263
x=49, x^2-N = 360 = 2^3 * 3^2 * 5
x=50, x^2-N = 459 = 3^3 * 17
x=51, x^2-N = 560 = 2^4 * 5 * 7
x=52, x^2-N = 663 = 3 * 13 * 17
x=53, x^2-N = 768 = 2^8 * 3
x=54, x^2-N = 875 = 5^3 * 7
x=55, x^2-N = 984 = 2^3 * 3 * 41
x=56, x^2-N =1095 = 3 * 5 * 73
x=57, x^2-N =1208 = 2^3 * 151
x=58, x^2-N =1323 = 3^3 * 7^2
x=59, x^2-N =1440 = 2^5 * 3^2 * 5
```

Set $x = 46 \cdot 47 \cdot 49 \cdot 51$ and $y = 2^5 \cdot 3^2 \cdot 5^2 \cdot 7$. Then $x^2 \equiv y^2 \bmod N$. We have $x \equiv 311$, $y \equiv 1416$ and $\gcd(1416 - 311,\ 2041) = 13$. Hence $N = 2041 = 13 \cdot 157$.

15. Let $N = 10573$ and $k = 2^3 3^3 = 216$; then $a^k - 1 \bmod N \equiv 1744$ and $\gcd(1744, N) = 109$. The method is successful since $N = 109 \cdot 97$ and $108 = 2^2 3^3 \mid k$, whereas $96 = 2^5 3 \nmid k$. However, choosing $k = 2^5 3^3 = 864$ would not work since k is divisible by 108 *and* 96.

Key Establishment

1. The discrete-logarithm problem in the additive group $\mathbb{Z}_p$ is to find a solution to the modular equation $A = g \cdot a \bmod p$, where a is unknown. But this is easy: $a = Ag^{-1} \bmod p$. Note that the computing the multiplicative inverse is very efficient with the Extended Euclidean Algorithm.

2. Since $q \mid p-1$ we have $p-1 = rq$ and thus $p = rq+1$, where r is even. First, generate a random prime q of length n_q. Choose an even uniform integer r of length $n_p - n_q$ and set $p = rq+1$. Then p has the required bit length n_p. Check the primality of p and choose a new integer r if p is not a prime number. To find a generator g with $\text{ord}(g) = q$, choose a random number h in the range $1 < h < p-1$ and compute $g = h^r \bmod p$. If $g \equiv 1 \bmod p$, choose another h. If $g \not\equiv 1 \bmod p$, then we have $\text{ord}(g) = q$. Note that the definition of g yields $\text{ord}(g) \mid \frac{p-1}{r} = q$. Since q is a prime, $\text{ord}(g)$ must be either q or 1.

3. $G = \langle 2 \rangle$ is a subgroup of $\mathbb{Z}_{89}^*$ and $\text{ord}(G) = \text{ord}(2) \mid 88$. Since $2^{11} \equiv 1 \bmod 89$ and 11 is a prime, one has $\text{ord}(2) = 11$. The Diffie-Hellman shared secret key k is an element of G. Therefore, 11 different keys can occur in this case.

4. a) $p = 43$ and $g = 3$ are given. Then $p-1 = 42 = 2 \cdot 3 \cdot 7$. Compute $g^{\frac{p-1}{q}} \bmod p$ for all prime divisors q of $p-1$. We get $3^{21} \equiv 42$, $3^{14} \equiv 36$ and $3^6 \equiv 41 \bmod 43$. None of these values is congruent to 1, and so the order of g must be maximal, i.e., $\text{ord}(g) = 42$.
   b) We send Alice the public key $B = g^b = 3^{26} \equiv 15 \bmod 43$. The shared secret key is $k = A^b = 14^{26} \equiv 23 \bmod 43$.

5. a) $\text{ord}(\mathbb{Z}_{107}^*) = 106 = 2 \cdot 53$. Let $g \equiv 3$. Since $g^{53} \bmod 107 \equiv 1$, we have $\text{ord}(g) = 53$, which is a prime number.
   b) Since $\lfloor \sqrt{53} \rfloor = 7$, at most 7 babysteps and 7 giantsteps, i.e., ≤ 14 exponentiations are necessary.
   c) We have $m = 7$ and $A = 12$. Firstly, compute the babysteps $Ag^{-r} \bmod p$, where $r = 0, 1, \ldots, 6$:
   $$12, 4, 37, 48, 16, 41, 85$$
   Secondly, compute $T = g^7 \bmod 107 \equiv 47$ and the giantsteps $T^s \bmod p$, where $s = 0, 1, \ldots, 7$:
   $$1, 47, 69, 33, 53, 30, 19, 37.$$
   We have a match for $r = 2$ (babystep) and $s = 7$ (giantstep). Hence $a = ms + r = 51$ and $\log_3(12) = 51$.

6. Use SageMath. The hexadecimal strings defining p, g and q can be copied from RFC 5114, Section 2.3. First, check the primality of p and q.

```
sage: ps='87A8E61D B4B6663C FFBBD19C 65195999 8CEEF608 660DD0F2\
       5D2CEED4 435E3B00 E00DF8F1 D61957D4 FAF7DF45 61B2AA30\
       16C3D911 34096FAA 3BF4296D 830E9A7C 209E0C64 97517ABD\
       5A8A9D30 6BCF67ED 91F9E672 5B4758C0 22E0B1EF 4275BF7B\
       6C5BFC11 D45F9088 B941F54E B1E59BB8 BC39A0BF 12307F5C\
       4FDB70C5 81B23F76 B63ACAE1 CAA6B790 2D525267 35488A0E\
       F13C6D9A 51BFA4AB 3AD83477 96524D8E F6A167B5 A41825D9\
       67E144E5 14056425 1CCACB83 E6B486F6 B3CA3F79 71506026\
       C0B857F6 89962856 DED4010A BD0BE621 C3A3960A 54E710C3\
```

```
       75F26375 D7014103 A4B54330 C198AF12 6116D227 6E11715F\
       693877FA D7EF09CA DB094AE9 1E1A1597'
sage: p=ZZ(ps,16)
sage: p.is_pseudoprime()
True
sage: qs = '8CF83642 A709A097 B4479976 40129DA2\
       99B1A47D 1EB3750B A308B0FE 64F5FBD3'
sage: q=ZZ(qs,16)
sage: q.is_prime()
True
```

Next, we verify that $g^q \equiv 1 \bmod p$ so that $\mathrm{ord}\,(g) \mid q$. Since $g \not\equiv 1$ and q is prime, we conclude that $\mathrm{ord}\,(g) = q$.

```
sage: gs='3FB32C9B 73134D0B 2E775066 60EDBD48 4CA7B18F 21EF2054\
       07F4793A 1A0BA125 10DBC150 77BE463F FF4FED4A AC0BB555\
       BE3A6C1B 0C6B47B1 BC3773BF 7E8C6F62 901228F8 C28CBB18\
       A55AE313 41000A65 0196F931 C77A57F2 DDF463E5 E9EC144B\
       777DE62A AAB8A862 8AC376D2 82D6ED38 64E67982 428EBC83\
       1D14348F 6F2F9193 B5045AF2 767164E1 DFC967C1 FB3F2E55\
       A4BD1BFF E83B9C80 D052B985 D182EA0A DB2A3B73 13D3FE14\
       C8484B1E 052588B9 B7D2BBD2 DF016199 ECD06E15 57CD0915\
       B3353BBB 64E0EC37 7FD02837 0DF92B52 C7891428 CDC67EB6\
       184B523D 1DB246C3 2F630784 90F00EF8 D647D148 D4795451\
       5E2327CF EF98C582 664B4C0F 6CC41659'
sage: g=ZZ(gs,16)
sage: power_mod(g,q,p)
1
```

7. Without the hashing we would have $k = s$, i.e., the key is the RSA plaintext. In a CPA experiment for RSA key encapsulation an adversary could simply test a challenge k' by computing $(k')^e \bmod N$ and compare the result with c. If they are equal, then $k = k'$ and the adversary outputs $b' = 1$. Otherwise, k' is random and the adversary outputs $b' = 0$. The adversary wins the experiment, and so the modified scheme is insecure. With the hashing operation in place, the adversary obtains $k = H(s)$ or a random key. But H is one-way and therefore, he cannot obtain s and compute the ciphertext.
8. If a and b are both re-used, then the shared Diffie-Hellman key $k = g^{ab}$ remains constant. If a is re-used and b is fresh, then a new shared secret key k is generated. However, if the long-term private key a is compromised, the consequences are severe: the shared secret keys of all Diffie-Hellman exchanges which used the private key a can be computed. In order to achieve *perfect forward secrecy* (PFS), both Diffie-Hellman keys a and b must be ephemeral.
9. A Man-in-the-Middle (Mallory) runs separate Diffie-Hellman protocols with Alice and Bob. In the key exchange with Alice he masquerades as Bob and in the key exchange with Bob he masquerades as Alice. Mallory generates private keys a' and b'. He intercepts the message A from Alice and sends his own public key $A' = g^{a'}$ to Bob. Accordingly, he intercepts B from Bob and sends $B' = g^{b'}$ to Alice. Then Mallory's shared secret key with Alice is $k_A = A^{b'}$ and the shared

secret key with Bob is $k_B = B^{a'}$. Alice and Bob cannot detect this attack, unless they are able to verify the authenticity of the public keys A and B.

10. a) Let $(c_1, c_2) = (g^b, A^b m)$ be an ElGamal ciphertext. Decryption is correct:

$$c_1^{-a} c_2 = g^{-ab} A^b m = g^{-ab} g^{ab} m = m.$$

b) $m = 17^{-20} \cdot 16 \equiv 25 \cdot 16 \equiv 46 \bmod 59$.

c) If an adversary knows b and eavesdrops a ciphertext (c_1, c_2), then they can use the equation $c_2 = A^b \cdot m$ to compute the plaintext $m = c_2 A^{-b}$ without knowing the secret key a.

If b is re-used, then the encryption is no longer randomized. Furthermore, $A^b = c_2 m^{-1}$ can be derived from any known plaintext/ciphertext pair. Suppose a new plaintext m' is encrypted using the same secret parameter b. Then decryption is easy since $m' = c_2 (A^b)^{-1}$.

Digital Signatures

1. Possible reasons are:
   a) The message was modified (error or changed by an adversary).
   b) The signature value is faulty (error or changed by an adversary).
   c) A computational error occurred during signature generation or verification.
   d) The wrong private or public key was used.

2. The new scheme is insecure since different messages have the same signature value. An adversary who knows the signature of (m_1, m_2) automatically gets a valid signature of $(m_1 \oplus m_0,\ m_2 \oplus m_0)$ for any message m_0 of the same length as m_1 and m_2.
   The scheme should be changed: sign $m_1 \| m_2$ instead of $m_1 \oplus m_2$.

3. Only signature b) is valid.
   An existential forgery for a chosen signature value s is $m = s^e \bmod N$. For $s = 99$, we get $m = 17$.

4. The prime factors of N are $p = 127$ and $q = 227$. Then $\varphi(N) = (p-1)(q-1) = 28476$. Then compute $d = (5 \bmod 28476)^{-1} \equiv 22781 \bmod 28476$.
   Signature: $s = 11111^{22781} \equiv 7003 \bmod 28829$.
   Verification: $s^e = 7003^5 \equiv 11111 \bmod 28829$.

5. It is not sufficient to check a given hash value. The verifier might otherwise check the integrity of another document or a fabricated hash value (existential forgery). The verifier needs the received message and has to compute the hash value himself.

6. It is very hard to generate a valid RSA-PSS signature without the private key. An adversary can try to choose a signature s, but it is very unlikely that the encoded message $EM = s^e \bmod N$ is valid, i.e., that $H(m')$ corresponds to any message m that he provides.

7. Then the signature generation is deterministic, but the security of RSA-PSS is not substantially reduced. During verification, DB and the salt value is computed from $maskedDB$ and $H(m')$. The result should be compared to the expected salt value.

8. In this experiment, the adversary is given a message m, which is generated uniformly at random, and the public RSA parameters e and N. They need to find a signature value s such that $s^e = m \bmod N$. This is equivalent to plain RSA decryption of m. The RSA assumption says that there is a key generation algorithm such that for *uniform* messages the probability of success is negligible in terms of the key size. Hence the plain RSA signature is secure under these conditions. Note the difference to the original signature forgery experiment: the adversary wins if he creates a valid signature of any new message.

9. k must remain secret, since otherwise one could easily compute the private key x from a challenge r and a response s.

$$x = r^{-1}(s - k) \bmod q$$

   An adversary should be unable to guess k in polynomial time. Hence k should be uniform random.

10. All computations are straightforward: Alice's public key is $y = 53$. Her initial message is $I = 41$ and her response is $s = 6$. Verification is successful.

11. a) Suppose that (r, s) is an ElGamal signature of m, i.e.,

$$r \equiv g^k \bmod p \text{ and } s \equiv k^{-1}(H(m) - ar) \bmod q.$$

Then:

$$A^r r^s = g^{ar} g^{ks} = g^{ar} g^{H(m)-ar} = g^{H(m)} \bmod p.$$

This shows that the verification of the ElGamal signature is correct.
b) $A = 4^{20} \equiv 17 \bmod 59$.
c) $r = 4^5 \bmod 59 = 21$, $k^{-1} \equiv 6 \bmod 59$ and $s = 6 \cdot (8 - 20 \cdot 21) \equiv 22 \bmod 29$. The ElGamal signature is $(r, s) = (21, 22)$.
d) $A^r r^s = 17^{21} \cdot 21^{22} \bmod 59 \equiv 46$ is equal to $g^{H(m)} = 4^8 \bmod 59 \equiv 46$.
e) If a signature (r, s), hash $H(m)$ and k is known, then an adversary can easily compute the secret key a:

$$sk = H(m) - ar \bmod q \Rightarrow a = r^{-1}(H(m) - sk) \bmod q$$

If the same secret parameter k is used to generate the signatures s_1 and s_2 of m_1 and m_2, then we have:

$$k = s_1^{-1}(H(m_1) - ar) = s_2^{-1}(H(m_2) - ar) \bmod q.$$

Rearranging the last equation gives a.

12. a) $y = 4^{20} \bmod 59 = 17$.
b) $r = (4^5 \bmod 59) \bmod 29 = 21$, $k^{-1} \bmod 29 = 6$, $s = 6 \cdot (8 + 20 \cdot 21) = 16 \bmod 29$.
c) $s^{-1} \bmod 29 = 20$ and $4^{8 \cdot 20} \cdot 17^{21 \cdot 20} \bmod 59 = 21$, which verifies the signature.
d) If a signature (r, s), a hash $H(m)$ and k is known, an adversary can easily compute the secret key x:

$$sk = H(m) + xr \bmod q \Rightarrow x = r^{-1}(sk - H(m)) \bmod q$$

Therefore, k must be uniform random and secret.

Elliptic Curve Cryptography

1. a) $K^2 = \{(0,0),(0,1),(0,2),(1,0),(1,1),(1,2),(2,0),(2,1),(2,2)\}$,
   $\mathbb{P}^2(K) = \{[0:0:1],[0:1:1],[0:2:1],[1:0:1],[1:1:1],[1:2:1],$
   $[2:0:1],[2:1:1],[2:2:1],[1:0:0],[0:1:0],[1:1:0],[1:2:0]\}$.
   b) $y^2z = x^3 + xz^2 + 2z^3$.
   c) The discriminant is $\Delta = 2$.
   d) $E(GF(3)) = \{[0:1:0],[1:1:1],[1:2:1],[2:0:1]\}$.
   e) The line through $P = (1,1)$ and $Q = (2,0)$ has the equation $y = -x + 2$. Inserting into the Weierstrass equation gives $(-x+2)^2 = x^3 + x + 2$ and hence $x^3 - x^2 + 2x - 2 = x^3 + 2x^2 + 2x + 1 = (x+1)(x+2)^2 = 0$. Therefore, the third point R on the line has the coordinates $x = 1$, $y = -x + 2 = 1$. Hence $R = P$ and we obtain $P + Q = -R = -P = (1,2)$. Since $Q = -Q$ we have $2P = Q$ and $2Q = O$.
   f) It follows from e) that $2P \neq O$ and $4P = 2Q = O$. Therefore, P has order 4. Since $E(GF(3))$ also has order 4, the point P is a generator and $E(GF(3)) \cong \mathbb{Z}_4$. The points on the curve are P, $2P = Q$, $3P = P + Q = -P$ and $4P = O$.
2. a) $\Delta \equiv 12 \bmod 19$. Since $9^2 = 4^3 + 3 \cdot 4 + 5$, the point $P = (4,9) = [4:9:1]$ lies on the curve. We use the formulas for point addition. The slope is $m = 6$ and $2 \cdot P = (9,18)$.
   b) $13P = 2 \cdot (2 \cdot (2 \cdot P))) + 2 \cdot (2 \cdot P) + P = (2,0)$.
   b) $O = [0:1:0]$ has order 1, $[2:0:1]$ has order 2, $[9:18:1]$ has order 13 and $[4:9:1]$ has order 26.
   c) $\Delta = -12528$, $P = (4,9) \in E(\mathbb{Q})$, $m = \frac{17}{6}$ and $2 \cdot P = (\frac{1}{36}, \frac{487}{216})$.
3. a) Hasse's Theorem implies $|23 + 1 - \text{ord}\,(E(GF(23)))| \leq 9.59$ and hence
   $$15 \leq \text{ord}\,(E(GF(23))) \leq 33.$$
   b) Since the order of $E(GF(23))$ is less than or equal to 33, five point doublings and five additions are sufficient when using the double-and-add algorithm, i.e., at most 10 point operations. In fact, 9 point operations are sufficient.
4. First, we verify that p and n are prime numbers:

```
sage: p=0xA9FB57DBA1EEA9BC3E660A909D838D726E3BF623D52620282013481D1F6E5377
sage: p.is_prime()
True
sage: n=0xA9FB57DBA1EEA9BC3E660A909D838D718C397AA3B561A6F7901E0E82974856A7
sage: n.is_prime()
True
```

   The parameters define a nonsingular curve E over $GF(p)$:

```
sage: a = 0x7D5A0975FC2C3057EEF67530417AFFE7FB8055C126DC5C6CE94A4B44F330B5D9
sage: b = 0x26DC5C6CE94A4B44F330B5D9BBD77CBF958416295CF7E1CE6BCCDC18FF8C07B6
sage: disc=-16*(4*a^3+27*b^2);mod(disc,p)
15036242490247342171513009477805930598983339216081386851174014206346325949410
sage: E=EllipticCurve(GF(p),[a,b])
```

   We verify that the order of $E(GF(p))$ is n. The computation of the order may take a few seconds.

```
sage: E.order()==n
True
```

   Then we check that $g = (x_g, y_g) \in E(GF(p))$:

```
sage: xg=0x8BD2AEB9CB7E57CB2C4B482FFC81B7AFB9DE27E1E3BD23C23A4453BD9ACE3262
sage: yg=0x547EF835C3DAC4FD97F8461A14611DC9C27745132DED8E545C1D54C72F046997
sage: g=E(xg,yg)
```

Since $E(GF(p))$ has prime order n, the order of the non-zero point g must be n. Finally, we check that ord $(p \bmod n)$ is large:

```
sage: mod(p,n).multiplicative_order()
38442478198522672110404873314500824546368765892207264769377759531531768179539
```

In fact, we have ord $(p \bmod n) = \frac{n-1}{2}$.

5. Use SageMath.
   $A =$
   (30786306364684019669845085647834227301026705121148702657850323422577469426661,
   62738119601096463087058618165599972860801258532835385944058084661017583328220).
   $B =$
   (63856341335644447573330799294730313060965602021945406582077408231386506305403,
   69225670661515104449943687281706110118505391815211949231460931578788174425194).
   $b \cdot A = a \cdot B = k =$
   (62277408572425350581153587818274169049667602786711788049878422423086378303275,
   33362437316335065570684137232427223851259119247580365682976721759161769329886).

6. Since y^2 equals $x^3 + ax + b$ modulo p, the x-coordinate determines $\pm y$. Hence the y-coordinate provides at most one bit of additional entropy.

7. a) Let r be the x-coordinate (modulo n) of the point $k \cdot g$ and let
   $$s = k^{-1}(H(m) + xr) \bmod n.$$
   The signature parameters r and s must be non-zero.
   Then $k = s^{-1}(H(m) + xr) \bmod n$ and
   $$H(m)s^{-1} \cdot g + (rs^{-1}) \cdot y = s^{-1}(H(m) + xr) \cdot g = k \cdot g.$$
   By definition, the x-coordinate of this point modulo n is r, which shows that the verification is correct.
   b) We compute $y = x \cdot g = (11, 18)$ and $k \cdot g = (9, 1)$. The x-coordinate of $k \cdot g$ modulo 13 is $r = 9$. Furthermore, $k^{-1} = (3 \bmod 13)^{-1} \equiv 9$. Then $s = 9 \cdot (11 + 2 \cdot 9) \equiv 1 \bmod 13$, and the signature is $(r, s) = (9, 1)$.
   To verify the signature, we check that r and s are between 1 and 12. Then we compute
   $$s^{-1}H(m) \cdot g + s^{-1}r \cdot y = 11 \cdot (18, 18) + 9 \cdot (11, 18) = (9, 1).$$
   The x-coordinate of this point modulo 13 is 9, which is equal to r. This verifies the signature.
   c) Suppose the signature (r, s), the hash $H(m)$, the order n and k is known. Then an adversary can easily compute the private key:
   $$x = r^{-1}(s\,k - H(m)) \bmod n$$