SJSU SAN JOSÉ STATE UNIVERSITY

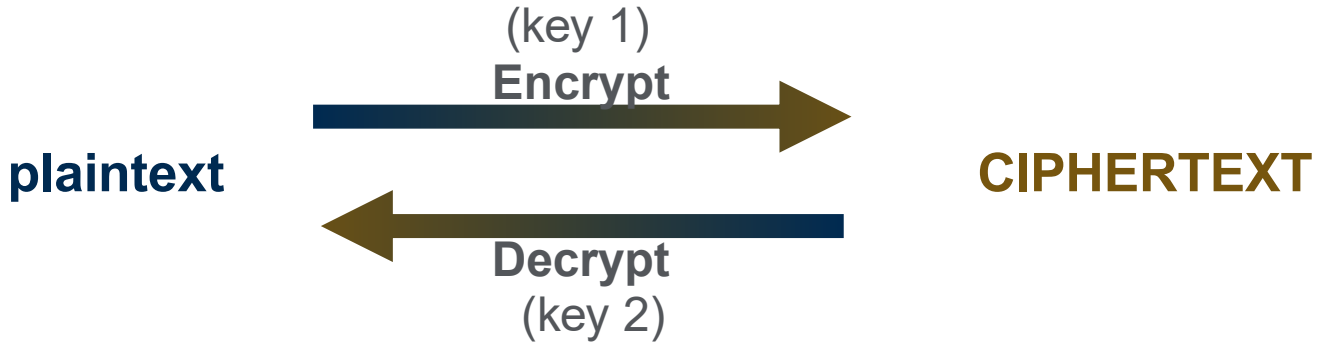
# Lesson 13 – Midterm 1 Review

**Yan Chen**

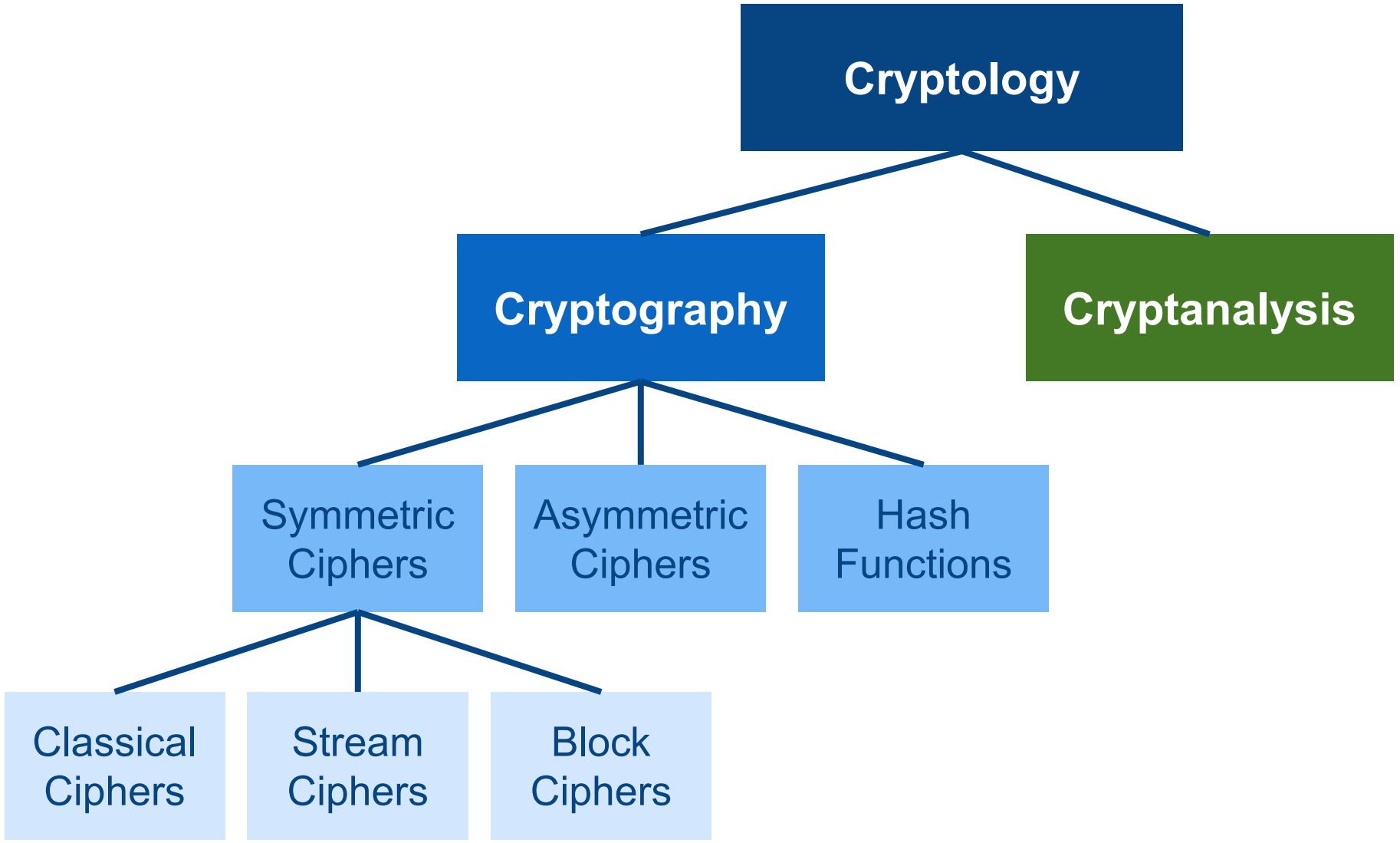
CS166 Fall 2024

- CIA triad: primary focus (L0 P16)
  - **Confidentiality, integrity**, availability
- **How to speak crypto** (L1 P8)
  - Cryptography, Cryptanalysis, Cryptology, Crypto
- **Kerckhoffs' principle**: the strength of a cryptosystem depends ONLY on the key (L1 P11)
  - Trudy only doesn't know the key (and of course, the plaintext)
- Key properties of a secure system by Claude Shannon
  - Confusion: hide relationship between plaintext & ciphertext
  - Diffusion: hide the statistics

- Cipher system (cryptosystem L1 P9, 10, 16)




  - Key used for encryption and decryption can be different
  - Keyspace: the set of all possible values of the key
  - Exhaustive key search: check the whole keyspace
- **Definition for "secure"** (L1 P21)
  - A cipher system is secure if best know attack is to try all keys
  - A cipher system is insecure if any shortcut attack is known
  - Under this def., an insecure cipher may be harder to break!



- Classical ciphers are dead, but worth analyze (L1 P13)
  - Modern ciphers developed from them!
- Classical ciphers covered (L1 P14 – 27)
  - Caesar cipher, **parameterized Caesar**, simple substitution
  - Double-transposition, **one-time pad**, codebook
- For each of above ciphers, you should know... (L1)
  - How it works, that is, how to encrypt & decrypt, and how to get key if given plaintext & ciphertext
  - **Keyspace and work factor** (except for codebook)
  - Secure or not, confusion or diffusion, why it's dead

- **Stream ciphers uses the idea of "one-time pad"** (L2)
  - "Stretch" a small key to a long keystream (any size)
  - The keystream is used to encrypt/decrypt like a one-time pad
  - Keystream is pseudo-random (not truly random) and may repeat (important to know the upper bound)
  - Efficient in hardware –was popular
- We focused on A5/1 and RC4, you should know...(L2)
  - Basic information (summarized on L3 P3)
  - For **A5/1**, understand **majority function** (Assignment 1 Q6)
  - **RC4 operates on bytes, so it's also efficient in software**

- **Block ciphers uses the idea of "codebook"** (L3)
  - Each block of plaintext has a corresponding block of ciphertext
  - Key is used to generate codebooks
  - Change key = switch the codebook
  - "Electronic" version of codebook ("book" not fixed)
  - Notation: C = E(P, K), P = D(C, K)
- Feistel cipher: general block cipher design principle (L3)
  - A type of block cipher, not a specific block cipher
  - "Half-half", swap & XOR involved in each round
  - Round function F does not need to be invertible

- Focused on DES, AES, and TEA, you should know...(L3)
  - Basic information (summarized on L4 P4)
  - **Feistel or not**, and why
  - For **DES, how S-box works**, and why 3DES is needed
  - For **AES, functions confusion or diffusion**
- Block cipher modes: encrypt multiple blocks (L4 P6 – 11)
  - ECB, CBC, CTR (summarized on L5 P2)
  - How each works
  - ECB's weakness, CBC auto-recover, CTR like a stream cipher
- **MAC: used for integrity** (L4 P12 – 15)

- Public key crypto: different keys to encrypt and decrypt
  - No key exchange needed! (L5 P5)
  - Based on “trapdoor one-way functions” (L5 P7)
  - More mathematical than symmetric key ciphers
- Can be used for encryption or signature (L5 P6)
  - $\{M\}_{receiver}$: encrypt M by receiver’s public key so only receiver can decrypt using his/her private key
  - $[M]_{signee}$: “encrypt” M by signee’s private key so only he/she can sign, and others can verify by decrypting using public key (Others must also know M!)

- RSA (L5 P17 – 20 )
  - How to generate the keys: public (N, e), private d (and p & q)
  - How to encrypt and decrypt (L5 P18)
  - How to choose secure keys (L5 P19 – 20)
- Diffie-Hellman: key exchange algorithm
  - Basic information (L7 P4) and how it works (L7 P5)
  - MIM attack (L7 P6)
- ECC: a different math approach (L7 P7)
  - Pros & cons (L7 P8)

- Signature using public key crypto provides both **integrity and non-repudiation** (L7 P4)
  - Only signee knows his/her private key
  - If signed using public key crypto, signee cannot deny after!
- Corrections to common misinterpretations (L7 P5 – 6)
  - **Signature cannot identify the sender!**

    A message can be signed and sent by different people!
  - **Everyone can encrypt a message using public key!**

    Always remember public key is public!

    The signee, the encrypter, the sender all can be different!

- Public Key Infrastructure (PKI): the stuff needed to securely use public key crypto (L7 P7)
  - Certificate authority (CA): a trusted 3rd party (TTP) to create and sign digital certificate for users
  - Digital certificate: contains user's name and public key
- Verify CA's signature to verify integrity & identity of owner of corresponding private key (L7 P8)
  - **Does NOT verify the identity of the sender of certificate**
- PKI can use different "trust models" (L7 P9)
  - Monopoly, oligarchy, anarchy

- Hash function: "map" big M to smaller "fingerprint" of M
  - Notation: h(M), also called hash, or digest
  - **Collisions exist** since input space is larger than output space
  - If find a collision, hash is broken
  - If h(M) has n bits, then $2^n$ possible hashes
  - If hash x messages, $x^2$ comparisons are done

    **To find m collisions, need x = sqrt(m * $2^n$) tries** (L8 P5 – 7)
- Properties of a secure crypto hash (L8 P8 – 9)
  - Deterministic, compressive, efficient, one-way, avalanche effect, weak collision/strong collision resistance

- Algorithms similar to block ciphers (L7 P13 – 16)
  - Be familiar with the names: MD5, SHA-1, Tiger hash
- Understand usages of hash functions (L8 P17 – 20)
  - HMAC: hashed MAC, used for integrity
  - **Online bids**: bidders submit h(bid) instead of bid

    Hashes don't reveal bids (one way)

    Can't change bid after hash sent (weak collision resistance)
  - **Reduce spam email**: request sender's "proof-of-work"

    Make spam more costly to send emails to limit the amount

    NOT to block/eliminate spam emails!

- Covered 3 unintentionally software flaws (L9)
  - Buffer overflow, incomplete mediation, race condition
- For each flaw, need to know...
  - Why & how it will cause problems
  - How Trudy can exploit this flaw
- For buffer overflow, also need to know how to defense
  - Ways covered: non-executable stack, canary, ASLR, use safer language/methods (L9 P17 – 18)
  - Need to be able to name at least 3 of them
  - And can explain one of them in more detail

- Malware examples on timeline(L10 P6 –10)
  - Need to be able to analyze the trend (summarized on L11 P2)
- Malware detection(L10 P11 –14)
  - Signature, change, anomaly
  - For each, need to know what is it trying to detect, what it can detect, and pros & cons (summarized on L11 P3)
- Evade detection(L10 P15 –18)
  - Avoid common signatures: encryption, polymorphic, metamorphic –know the difference (summarized on L11 P4)
  - Or infect fast: flash worm

- Introduced SRE with a simple example (L11 P9 –12)
  - Be able to use a disassembler (whatever one you want, an online one should be enough for the quiz) to get the serial number of an .exe (you don't need to run the .exe!)
- Also covered 4 ways to mitigate SRE (L11 P13 –16)
  - Anti-disassembly, anti-debugging, tamper-resistance, code obfuscation
  - Need to be able to name at least 3 of them
  - And can explain one of them in more detail

- Covered 3 other attacks (L12)
  - Salami attack, **linearization attack**, time bomb
- For each (especially first 2), need to know...
  - **Under which situation the attack is possible**
  - Be able to outline such attack
- For linearization attack, also need to know...
  - How to analyze the **work factor** (L12 P9)
  - **Be able to come up a way of checking the serial number that is "immune" to such an attack**

- Timed quiz on Canvas (no access code)
  - Ingredients on the next slide
- Closed all materials
  - Except tools (include your own program, etc.) to calculate
  - One grade off if cheat (copy from each others, internet, etc.)!
- Cover L0 – L12 (assignment 1 – 3)
- Checkpoint & practice for the final
  - speed, format, etc.
- 3 pts if submitted on time with a score over 50%
  - All-or-nothing

- In terms of question types

| Type | # of Questions | Points | Time (mins) |
|---|---|---|---|
| **Matching** | 2 (4 matches) | 4 (4 * 1) | 4 (4 * 1) |
| **MC** | 9 | 9 (9 * 1) | 9 (9 * 1) |
| **Fill-in-blanks** | 6 (8 blanks) | 9 (7 * 1 + 1 * 2) | 13 (7 * 1.5 + 1 * 2.5) |
| **Short Answers** | 11 | 28 (8 * 2 + 3 * 4) | 32 (8 * 2.5 + 3 * 4) |

- In terms of topic covered

| | | | | | |
|---|---|---|---|---|---|
| Crypto Misc | 8 pts | Public Key | 9 pts | Software Insecurity | 13 pts |
| Symmetric | 16 pts | Hash | 4 pts | | |

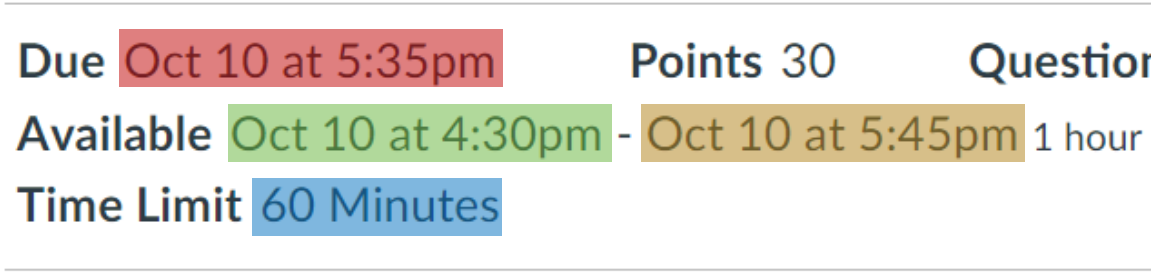
- In terms of difficulty level: roughly 7-2-1 scheme
  - 70% “easy”: conceptual, similar to assignment question
  - 20% “medium”: need some understanding
  - 10% “hard”: didn’t explicitly cover in class

- Oct. 10, Thursday, 16:30 – 17:45
  - 16:30 – 17:30 (60 mins): Take the midterm
  - 17:30 – **17:35** (5 mins): Finalize your submission

    Make sure you submitted the midterm before deadline

    No submission after deadline will be accepted!
  - 17:45 – 17:45 (15 mins): Check your submission

    Contact me for any unsuccessful submissions
- Or pick an earlier time (submit before Oct. 7, 23:59)
- Check on Canvas (on Oct. 8):



- Go over assignments, focus on those you did wrong
  - Should be enough to give you 50%...
- Go over the appendix of each lecture
  - Recap each concept (recap the key points, give examples)
  - Try the exercises
- Open a discussion if you have any question
  - Answer questions may help you to understand better
- Open any programs you wrote before starting quiz
- Set an alarm 2~3 minutes before deadline
  - Be sure to submit it before the deadline