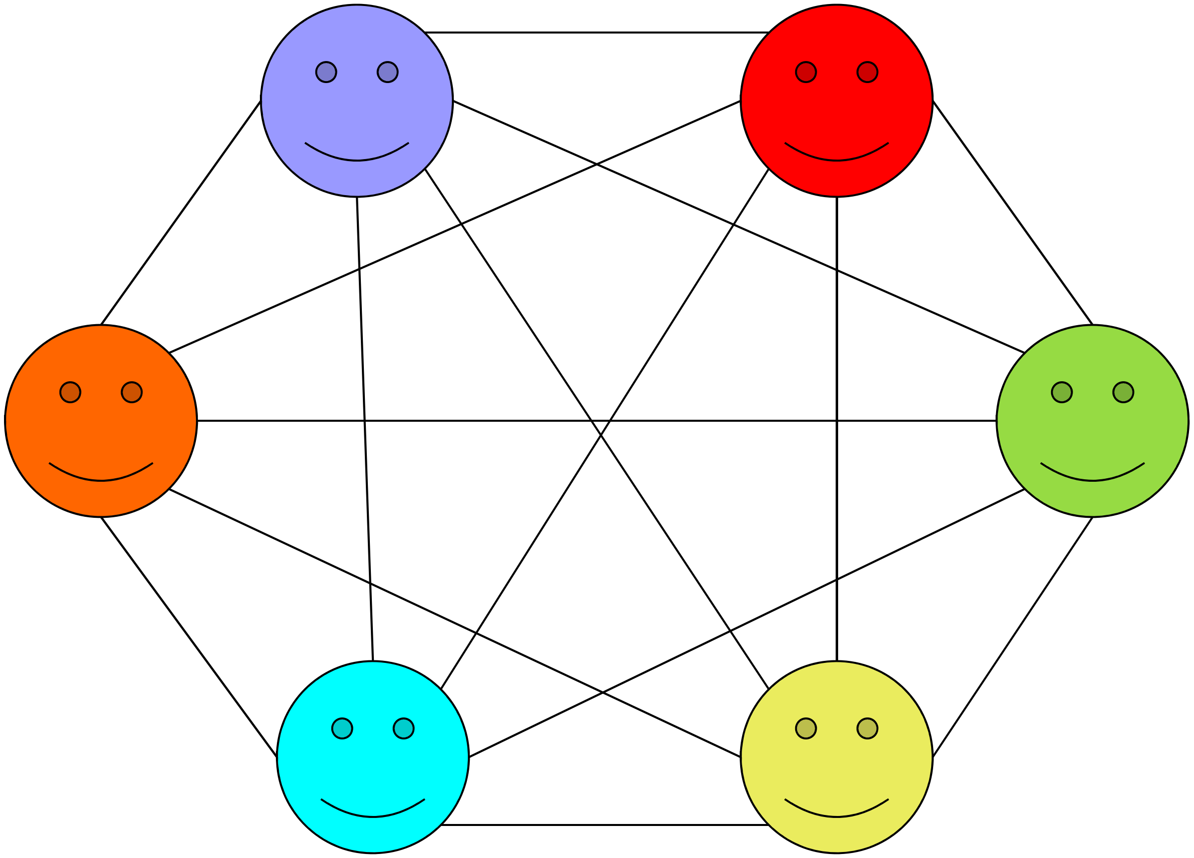
Chapter 2

Cryptography

Exercise 2.2 Decipher text

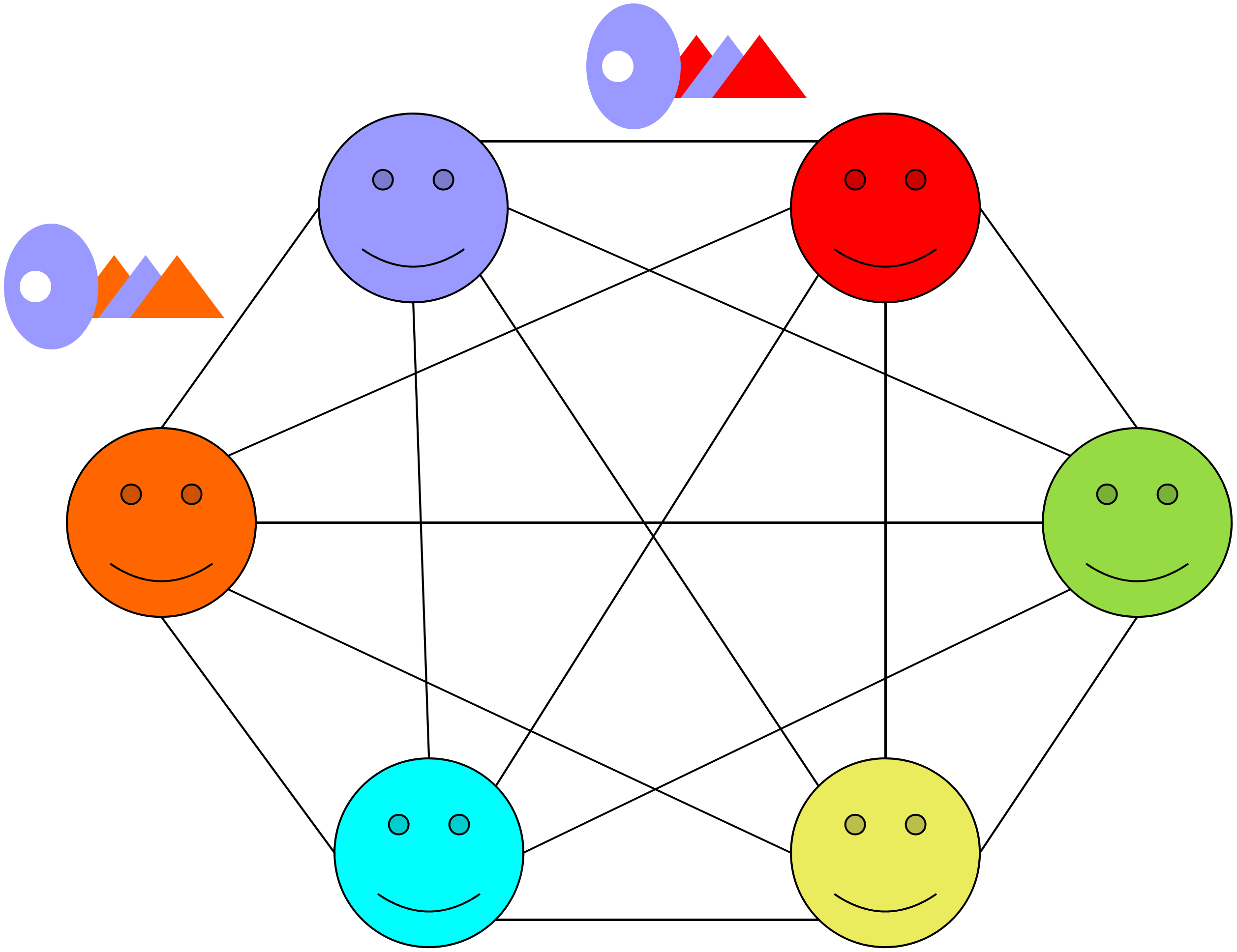
- Guess Key length
  - Lets try 1 first
  - Can brute force; but to make more efficient:
- Check frequencies
  - Recall common letters;
    - E: 12%, T: 9 %, A,I,N,O,R: 8%
  - Check frequency
    - q: 8x , d: 4x, u: 3x
    - Lets try q -> e; quick check: d -> r, u -> i, both common
    - Decrypting text: is meaningful.
  - If that would have failed: try other main letter
    - (e.g. q -> t)
  - If that fails; try other key lengths
    - Length 2 is like breaking twice

Exercise 2.3 Setup: 6 persons



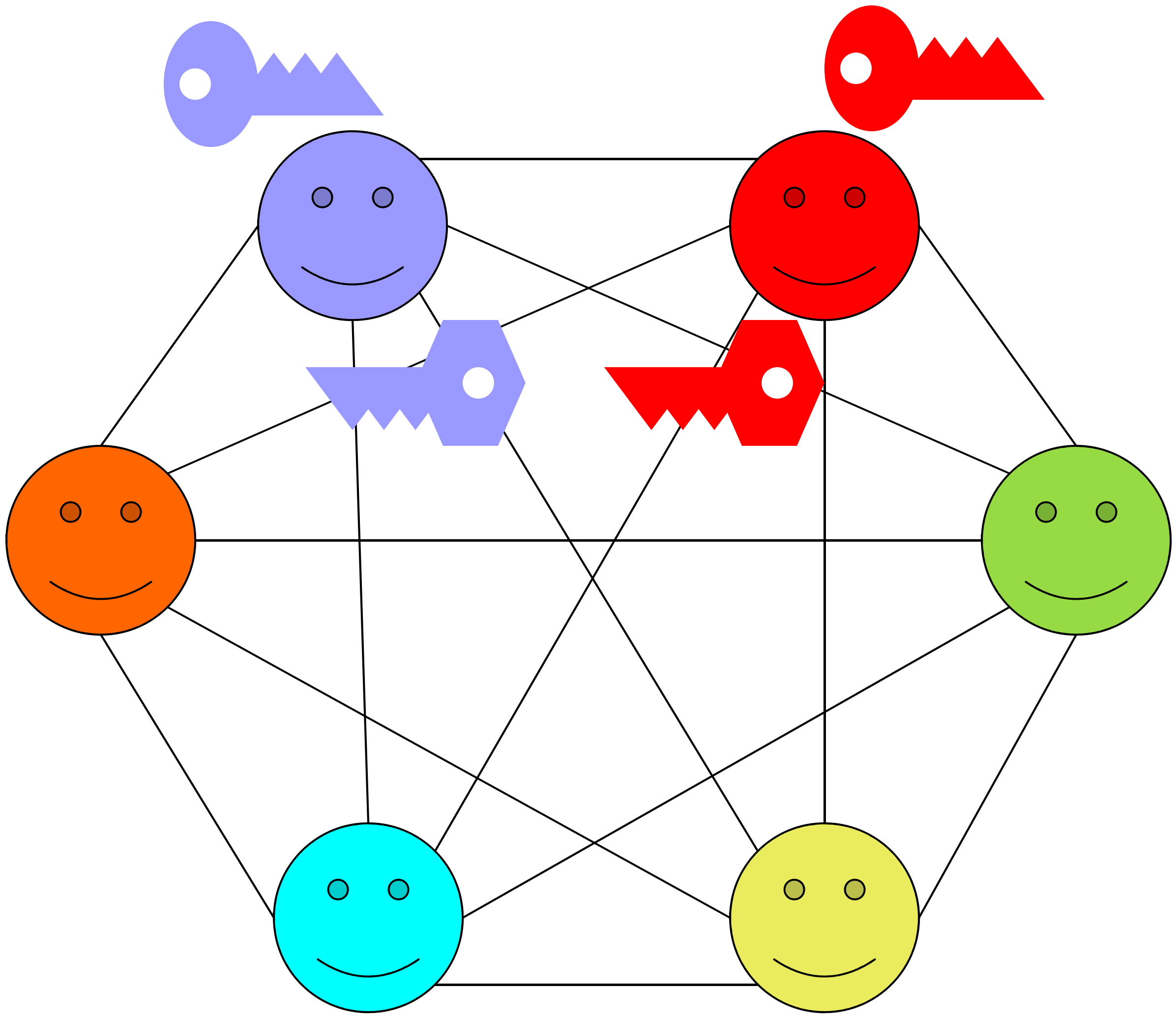
Each pair wants to be able to communicate
Others should not be able to eavesdrop.

Symmetric keys



Nr of Keys = Nr of Links = (N * N-1) / 2 = 6 * 5 / 2 = 15

Asymmetric keys

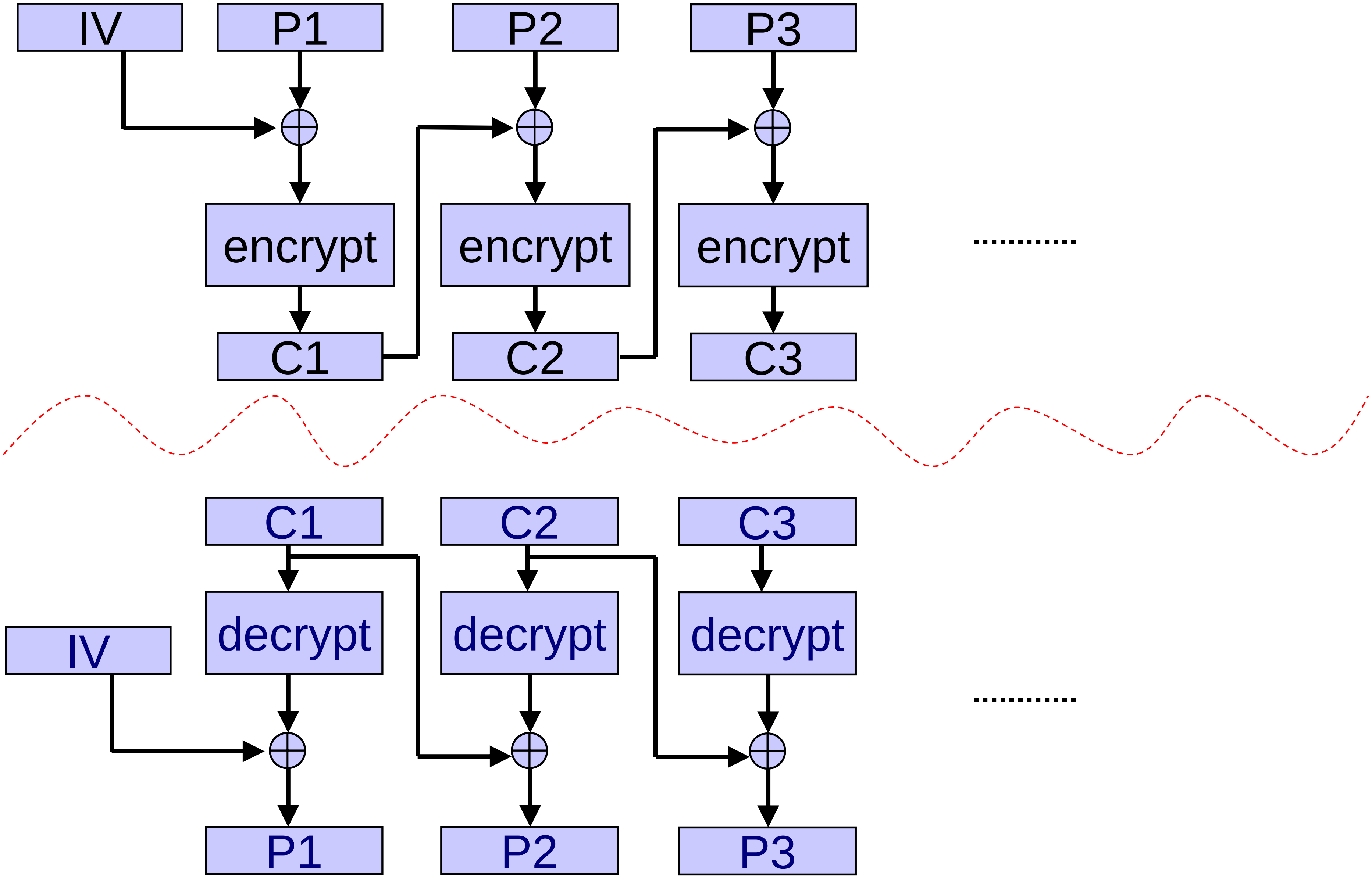


Nr of Keys = N + N = 12

Block modes Encryption -> Decryption

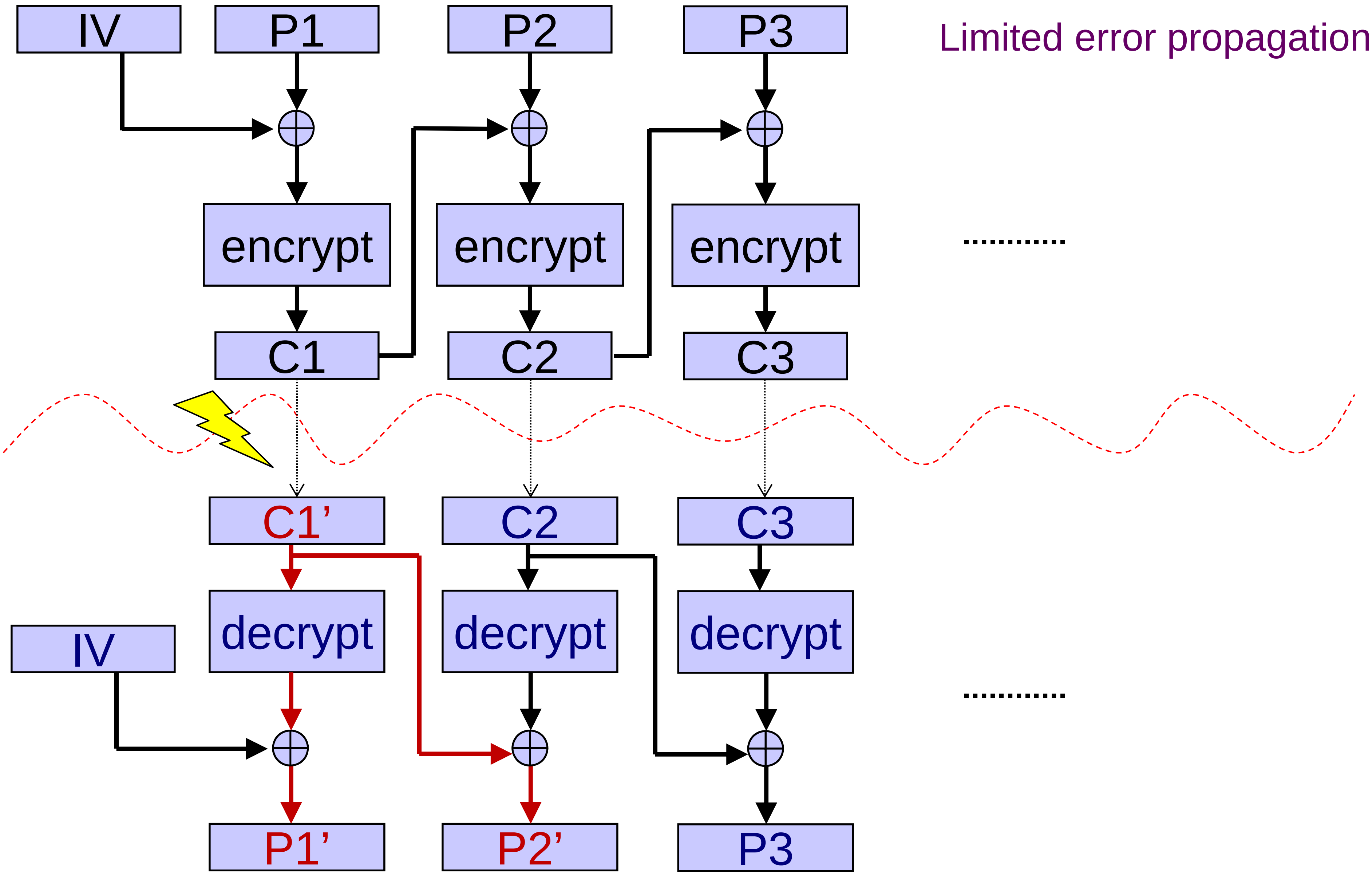
- Follow arrow from ciphertext backwards
- Undo each operation:
  - Encryption undo by Decryption
  - XOR undo by XOR with same value

CBC mode

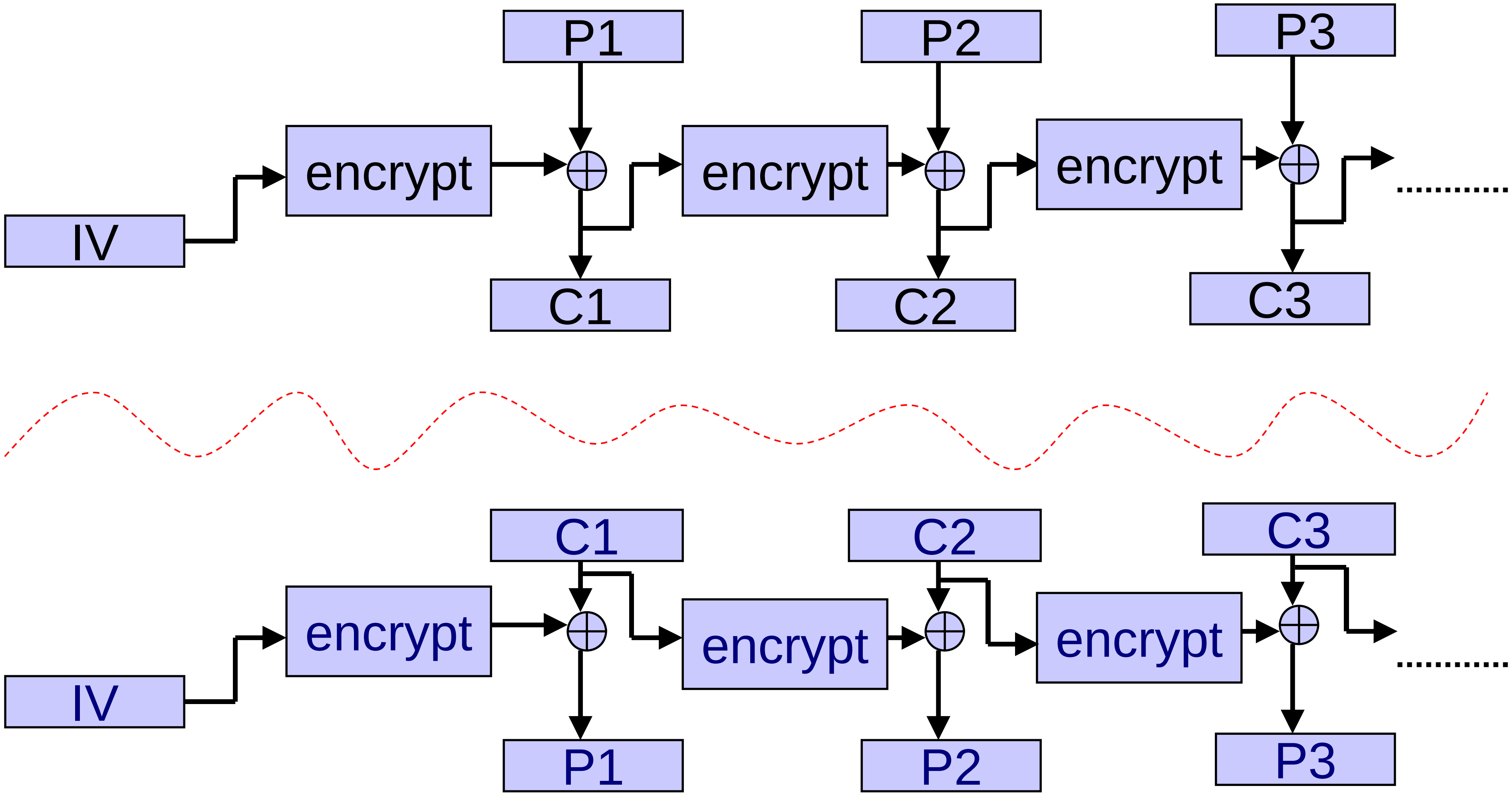

IV
P1
P2
P3
encrypt
encrypt
encrypt
…………
C1
C2
C3
C1
C2
C3
IV
decrypt
decrypt
decrypt
…………
P1
P2
P3

CBC mode

Limited error propagation



CFB mode



Note: No decryption

Block modes comparison

- Secrecy
  - Recognized patterns
  - Note: stream ciphers and (IV) reuse...
    - Suppose I know the encryption of message X=B1B2B3...
- Integrity
  - Can we detect tampering with ciphertext?
- Performance
  - parallelization and pre-computation

    (# encryptions needed same for all, xor cheap)

Block modes comparison

	ECB	CBC	OFB	CFB
Secrecy	patterns remain	encrypted text basically random; no patterns	Like one time pad with pseudo random key ... (reuse?)	Similar to CBC
Integrity	Can exchange replace blocks	Each block linked to next No exchange Replace effects next block	Encoding differs each block. No No exchange. Replace possible	Similar to CBC
Perform.	Full par. No pre comp.	No parallel No precomp	No parallel Full pre comp	No parallel Pre: Only first block

2.5 Entropy

- Need source of entropy
  - Unknown/random event
- Fair dice harder to predict than unfair dice
  - Assumption: Distribution is known
- If all options equally likely: more options = more entropy
  - Need 1 bit for coin, several for dice
- A known text has 0 entropy; no unknown/randomness
- Entropy pincode depends on how it is chosen
  - e.g. assigned vs. chosen

- Recall (attacker) knowledge influences entropy
  - distribution different; model with conditional probabilities
  - P(MyPswd | I'm an opel fan)
  - Example; roll of the dice, attacker is told whether result is even.
  - To find remaining entropy use: P(roll | (roll % 2))

Fair coin:

$- \frac{1}{2} \log2 \frac{1}{2} - \frac{1}{2} \log2 \frac{1}{2} = \frac{1}{2} + \frac{1}{2} = 1$

Example unfair coin (¾ heads, ¼ tails):

$- \frac{1}{4} \log2 \frac{1}{4} - \frac{3}{4} \log2 \frac{3}{4} = \frac{1}{2} + \frac{3}{4} * .41 = 0.8$

Fair dice:

$6 * - 1/6 \log2 1/6 = \log2 6 = 2.6$

State of the union:

$- 1 * \log2 1 = -1 * 0 = 0$

Pin code (assuming 4 random decimals):

$- 10{,}000 * 1/10{,}000 \ \log 2\ 1 / 10{,}000 = 13.29$

2.6 One time pad

a) How does it differ from the Vigenere cipher ?
  - It is like a vigenere but with a block size equal to the message size.
  - No key letter is reused so frequencies will not be maintained as with Vigenere

b) Taking key = (c XOR p) gives that c decrypts to p
  - so any p is possible for any c

c) What happens if the key of a one-time pad is reused?
  - Relation (XOR) of messages is revealed
    - Structure in data revealed
  - If not all messages meaningful; eliminates possibilities further
    - Same key has to make both meaningful

d) A shorter/reusable key cannot be used (length = n bits)
  - Given cipher text, unknown key: all n-bit plain text equally likely
  - Thus entropy has to be n, can only come from key
  - Max entropy = length in bits, so key needs to have at least n-bits

Exercise 2.7 El-Gamal

The El-Gamal cryptosystem is a variant of the Diffie-Hellman cryptosystem. Given a random large prime p and a generator g, Alice selects here private key x at random such that $1 \leq x \leq p-2$. Alice's public key is then (p,g,g^x).

To encrypt a message m (with $0 \leq m \leq p-1$) to Alice, Bob should select a random r such that $1 \leq r \leq p-2$. Bob then sends the message $(g^r, m\, h^r)$ to Alice, where $h=g^x$ comes from Alice's public key.

- How can Alice decrypt the message (c,d) she receives?
- Why can only Alice decrypt this?
- Why is it needed for Bob to generate a random number r
- Does Alice need to know that the number Bob chooses is really random?

Decryption & Security

- Alice has x, $c = g^r$, and $d = m * g^{xr}$
    - $m = d / g^{xr}$
    - $g^{xr} = c^x$
    - Division possible
- Finding m equivalent finding g^{xr}.
    - To build from g^r need x, from g^x need r
    - Cannot get x from g^x / r from g^r; discrete log hard
    - Formally; solution allows to distinguishes between

      $$(g^x, g^r, g^{xr}) \text{ and } (g^x, g^r, g^z)$$

      which is a `hard problem’

Salting

- Can decrypt without it !
  - g^x is public so m * g^x is not safe.
  - also does randomization of the encryption (see 2.2)
- Picking good r in interest Bob
  - Guarantee only Alice can read m.
  - Receiving (c,d) gives no guarantees to Alice;
    - No authentication of Bob
    - Charlie could have sent (c,d)
    - Does not know whether m secret
    - Within a larger protocol; need to analyse
      - If Alice relies on bob to only send securely then important
      - Keep in mind when looking at protocols later in course

Bits (entropy)

- Message is in group so can be any of p-2 values; thus `blocksize’ is log2 p-2
- To send larger message: Use a block mode.

Exercise 2.7 ad Secure sending

Large message authenticated and secret to multiple parties:

- Sign a hash of the message
- Generate random (symmetric) `session’ key
- encrypt message, signature with session key
- encrypt session key with public key of each receiver
- send encrypted message & keys to all

Important aspects (Q2.7)

- Understand security notion (IND-CPA)
- Understand basic working of crypto schemas
- See why security guaranteed
  - Explain reason
  - Understand why change breaks algorithm
- See what properties are achieved and which are not.

Chapter 3

Network and Web security

- See 8.1 and lab sessions.

Chapter 4

Certificates and Trust

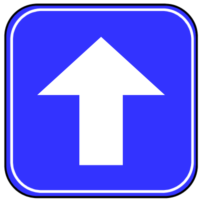
Exercise 4.1 Hash on FTP site

- Download changed: hash does not match
- Protects against errors in downloading
    - Very unlikely both hash
- Does not defend against malicious tampering
    - The hash function is public
    - An attacker that can alter files could compute hash of new file and also alter hash to match.

Exercise 4.2 Digital Signatures

Properties of Hash functions:

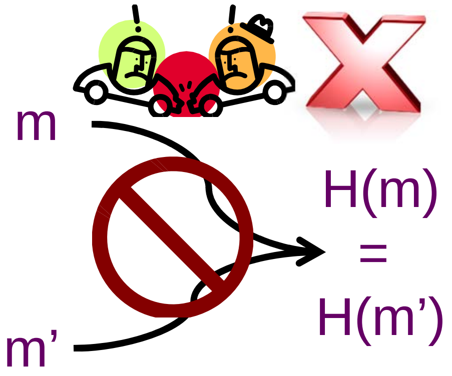
Practical



Efficiently computable

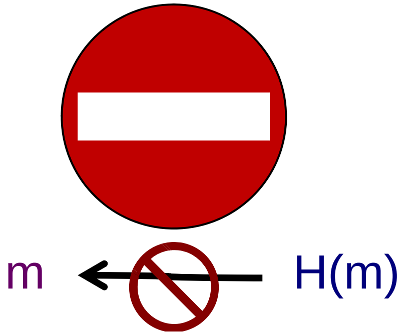
m → H(m)

Collision resistant



m, m’ with H(m) = H(m’)

Pre-image resistant



Hard to find:

m with H(m) = h

Importance for use with Digital Signatures

Essential:
Otherwise cannot make signatures

Essential:
Signature should not match other messages.

May not be needed:
If signature leaks information about message this may not be an issue (depends on use).

Exercise 4.3 RSA signing.

- RSA Signing
  - Hash message, decrypt the hash
- Correctness:
  - Decryption only possible with private key
- Completeness:
  - Of checks:
  - Of generation:

Exercise 4.5 Security proof for hash

- Fixed hash functions, such as MD5, SHA-1 etc. give a fixed size output.
  - Where is the randomness – cannot talk about probabilities.
  - Fixed function/output size implies constant amount of work.
- Use families of hash functions
  - Given security parameter n randomly select hash function h_i from family n.
  - Now can talk about complexity and probabilities.

Chapter 5

Access Control

Exercise 5.1 Lattices

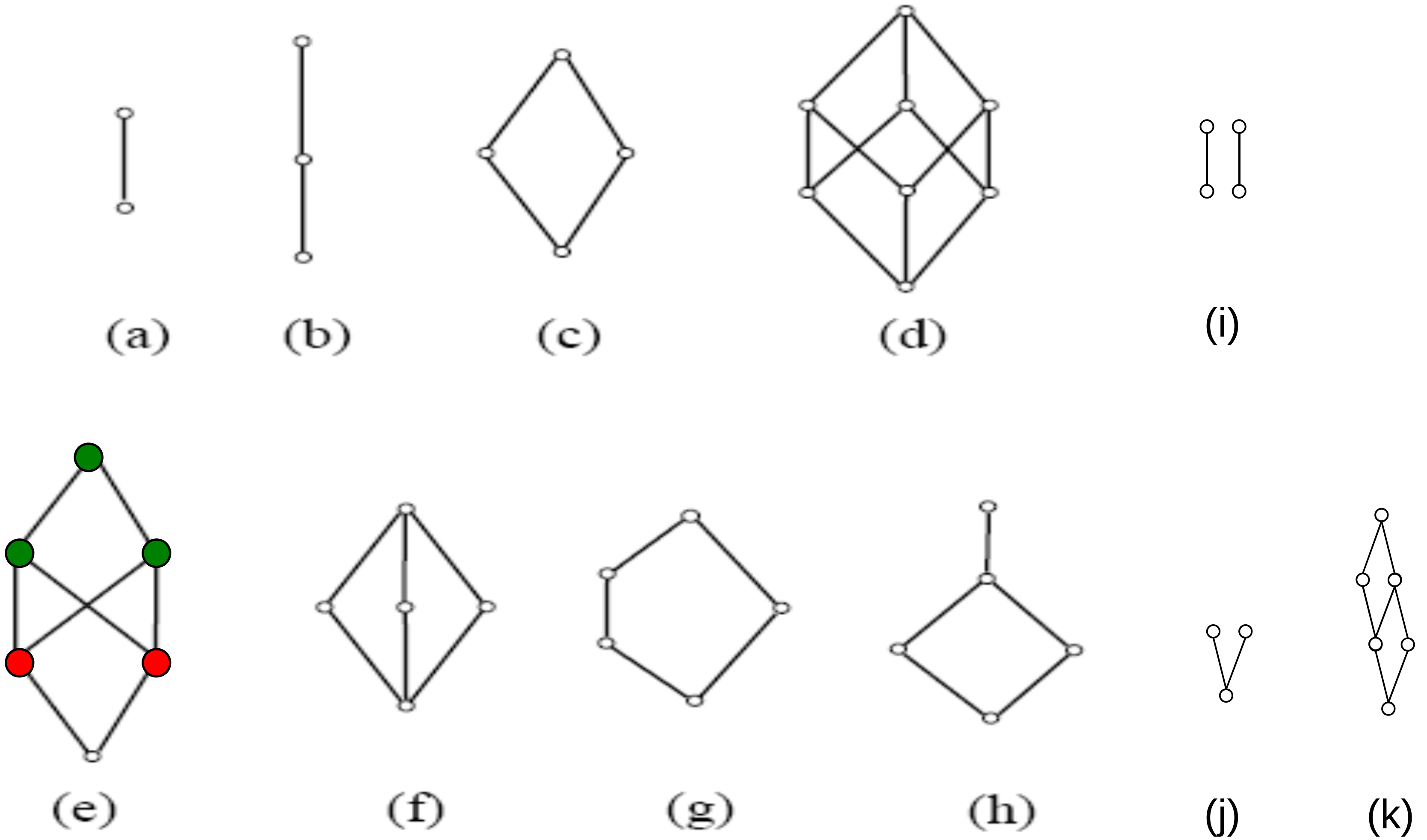


Figure 1 Examples of Hasse diagrams

(e),(i) and (j) are not lattices.

(b) Often lattice in MLS

Ordered levels for handling information flows.

LUB/GLB restriction of lattice for combinations of resources / users:

- New resource is combination several resources: level is LUB of levels these resources
- Users working/viewing together: clearance is GLB of individual clearance

(c) Monotone

Safety; Message/information loss may happen - this should never increase permissions.

Exercise 5.4 AC policy

Data: EHR records with medication history of a patient

Actions: read, add prescription.

Users: Doctors Daisy and Edward,
Nurses Nancy and Mark and
Patients Alice, Bob and Charlie

Policy:

- Doctors are allowed to read the health records of patients
- Doctor treating patient may add new prescriptions and may let a nurse read the EHR of the patient

Facts:

- Daisy is treating Alice and Bob, Edward is treating Charlie
- Nurse Nancy is assisting Daisy with the treatment of Alice

Give ACM, role based model and logical model.

ACM

	EHR Alice	EHR Bob	EHR Charlie
Alice			
Bob			
Charlie			
Daisy	read, write	read, write	read
Edward	read	read	read, write
Nancy	read		

Assumes: Nurse assisting implies treating Dr has given the read permission.

Role based

Roles: Dr, Nurse, Patient

	EHR Alice	**EHR Bob**	**EHR Charlie**
Dr	read	read	read
Nurse			
Patient			

Role	**Members**
Dr	Daisy Edward
Nurse	Nancy Mark
Patient	Alice Bob Charlie

What about `treating’, `assisting’ ?

Don’t fit well need e.g. Treating & assisting roles for each patient

Logical system

Predicates & system rules

- dr, treating, patient, nurse, RehrOf, WehrOf, maySay, says
- standard logical rules and x maySay p /\ x says p => p

Translation policy rules:

- Doctors are allowed to read the health records of patients
  dr(x) /\ patient(y) => RehrOf(x, y)
- Doctor treating patient may add prescriptions, let nurse read ehr
  (dr(x) /\ patient(y) /\ treating(x, y)) =>
  (mayWriteEhrOf(x, y) /\ (nurse(z) => x maySay REhrOf(z, y)))

Translation Facts:

- dr(Daisy), dr(Edward), nurse(Nancy), patient(Alice) , etc.
- Daisy is treating Alice and Bob, Edward is treating Charlie
  treating(Daisy, Bob), treating(Daisy, Bob), treating(Edward, Charlie)
- Nancy assists Daisy with treatment Alice
  Daisy says mayReadEhrOf(Nancy, Alice)

Scenario

Patient Alice is treated by Daisy who wants Nancy to prepare some follow up actions.

Step 1) Daisy allows Nancy to read Alice's Record

- ACM: Add entry for Nancy in Column EHR Alice (who does this and how to check that this is ok is not addressed by this system.)
- Role Based: Add Nancy to the `assisting-treatment-Alice' role.
- Logical: add fact Daisy says mayReadEhrOf(Nancy, Alice)

Step 2) Nancy read's Alice's Record

- ACM: The entry is present in the ACM so Alice gets access
- Role based: Nancy has a role that has read permission: she gets access.
- Logical: From facts: dr(Daisy) /\ patient(Alice) /\ treating(Daisy, Alice)

+ 2^e policy rule: nurse(Nancy) => Daisy maySay mayReadEhrOf(Nancy, Alice)

With facts: nurse(Nancy) and Daisy says mayReadEhrOf(Nancy, Alice)

this gives: mayReadEhrOf(Nancy, Alice) (maysay – says rule)

Thus Nancy gets access

Extension

- What if we want to add the rule
  - Patients can read their own health record
- Logical: Easy

  patient(y) => mayReadEhrOf(y, y)
- Role based: Does not fit (easily)
  - Cannot assign right (eg read EHR Alice) to role patient
    - Would imply Bob, Charlie can also read.
  - Would need role for each Patient
    - Or extensions/special interpretation right/resource

Extension: Resulting ACM

	EHR Alice	EHR Bob	EHR Charlie
Alice	read		
Bob		read	
Charlie			read
Daisy	read, write	read, write	read
Edward	read	read	read, write
Nancy	read		

Exercise 5.5 XACML Policy

- Permit-overrides
  - 1 permit is enough (logical **or** of the rules)
- Target: Any action by anyone on SampleServer
  - Policy applies to these requests
- Rule Login: Permit login during working hours
- Rule Finalrule: Everything is not allowed
  - becomes everything *else* due to `permit overrides’

Thus policy is:

The only action allowed on SampleServer
is logins during the day.

Chapter 6

Authentication

Exercise 6.4: Two factor authentication with pin (debit card) payments

(a) Factors: What you have (the card) and know (pin code).
  - Combine factors as payment sensitive application; needs a strong authentication. Factors are complementary;
  - Only card; risk of loss/theft too large
  - Only code; `random try’ attack too easy

(b) Signature could be used as third factor (what you are).

(c) Terminal asks to confirm amount
  - Protect against amount deducted not what agreed.
  - Effectiveness depends on attacker
    - Could be effective if attacker e.g. checkout employee
    - If attacker is store: could have fake terminal

Chapter 7

Security Protocols

Exercise 2.5 Security protocol analysis

```
1. A->B: A, K
2. B->A: { B, K, Nb }*pk(A)
3. A->B: { Nb }+K
```

a) No Authentication of B to A;
   Show how intruder can impersonate B

b) Provide a fix

c) Shared secrets?

d) Authentication of A to B?

e) "B" in 2. B->A: {"B",K,Nb}pk(A) needed?

```
1. A->B: A, K
2. B->A: { B, K, Nb }*pk(A)
3. A->B: { Nb }+K
```

What is Authentication of Alice to Bob:

An honest Bob knows that:

- A is active, running the same protocol
- *A thinks she is talking to Bob*
  - *Sufficient for honest Bob*
- (Note: Secrecy Nb is different property...)

Arguing Authentication

Check Authentication of Alice to Bob:

- Ensure a secret of Alice(*) is used
  - E.g. include challenge only Alice can answer
- Ensure secret is used in this session
  - Freshness of the challenge, no replay
- Ensure secret is used for Bob
  - Link challenge to authentication to Bob
  - No other way of answering challenge

(*) Could be shared secret with Bob also.

(a) No Authentication Bob

```
1. A->B: A, K
2. B->A: {B, K, Nb}*pk(A)
3. A->B: {Nb}+K
```

No authentication of Bob, attack:

1. A -> M(B): A, K
2. M(B) -> A: {B, K, N}*pk(A)
3. A->M(B): {N}+K

(b) A Fix

```
1. A->B: { A, K }*pk(B)
2. B->A: { B, K, Nb }*pk(A)
3. A->B: { Nb }+K
```

- Honest Alice knows after receiving message 2:
  - only Bob can decrypt message 1 so secret B used
  - K is fresh so Bob must have decrypted in this session
  - Bob decrypted for authentication to Alice
    - A included in message 1
    - Message cannot be misinterpreted as other message in protocol

(c) Shared secrets

```
1. A->B:  A, K
2. B->A: { B, K, Nb }*pk(A)
3. A->B: { Nb }+K
```

- No secrets shared
  - ☐ K revealed in message 1, Nb in message 3.

```
1. A->B: { A, K }*pk(B)
2. B->A: { B, K, Nb }*pk(A)
3. A->B: { Nb }+K
```

- Both K and Nb remain secret
  - ☐ Messages 1, 2 encrypted with public keys A,B
    - no information leaked to other parties
  - ☐ Message 3 reveals neither Nb nor K.

(d) Authentication Alice to Bob

```
1. A->B: A, K
2. B->A: { B, K, Nb }*pk(A)
3. A->B: { Nb }+K
```

Now honest Bob knows:

- After receiving message 3
  - only Alice can decrypt message 2 so secret Alice used
  - Nb is fresh: Alice must have decrypted in this session
  - Alice decrypted for Bob as B included in message 2

(e) "B" needed in message 2?

- Yes, otherwise can attack:

1.1 A -> I : A, K

2.1 I(A) -> B : A, K

2.2 B -> I(A) : { K, N }*pk(A)

1.2 I -> A : { K, N }*pk(A)

1.3 A -> I : { N }+K

2.3 I(A) -> B : { N }+K

Now honest Bob thinks Alice is talking with him but Alice is talking (and wants to talk) to Intruder not Bob

Chapter 8

Privacy and Anonymity

(Also: Exercises 7.1/6.1/5.2/etc.)

8.1 Scenario Security Analysis

- **Online music store**
- **members**
  - **music** with **ads** for free
  - music without ads for a **fee**.
  - recommend songs to other members
    - free ringtone if recommended song listened to
- security requirements:
  - Actors, interests, interdependencies
  - Attackers, goals, weaknesses attacks.
- Design with countermeasures

Online music store scenario

- 1st iteration Actors: stakeholders
  - shop, members, music provider, ad company, (bank, telephone operator,...)
- Goals, Attackers and Threats:
  - (fill in during discussion)
  - Both inside (stakeholders) and outside attackers

Important aspects

- Consider different viewpoints of the issue
  - Not only users, but also other stake holders; companies, governments, etc.
  - Identify stakeholders not directly mentioned (e.g. ad or content providers).
- Show insight in main security problems
  - Goals the security measures should reach
    - Not only `data protection’ but all `value’ protection
  - Determine what needs protection.
    - Not all interest equally at risk / equally likely
  - Determine trade-offs to be made in the design
    - related/conflicting goals; protecting one may harm other ...

8.2 DB privacy

a) Nationality cannot stay; only one Greek
   - Will be in group of size 1
   - All groups at least 2 without

b) All in (a) (obviously) + Age:
   - 60+ ers are all computation experts
   - Ok without

c) Cyber crime high in class `females in Eindhoven’

8.3 Database protection

- List of members
  - indexed by lastname
  - data field contains full name, address, etc.
- Defense:
  - index: hash of lastname.
  - Data field: encrypted with symmetric algorithm, key=lastname
- Finding information:
  - Simple; Hash lastname, find in DB, decrypt with lastname
- Protects against:
  - Spammer stealing complete DB; assumes attacker not willing to perform large amounts of computations/trial and error.
- No protection against:
  - Attacker looking for info on specific person (lastname known).