# EE515
# Security of Emerging Systems

Yongdae Kim

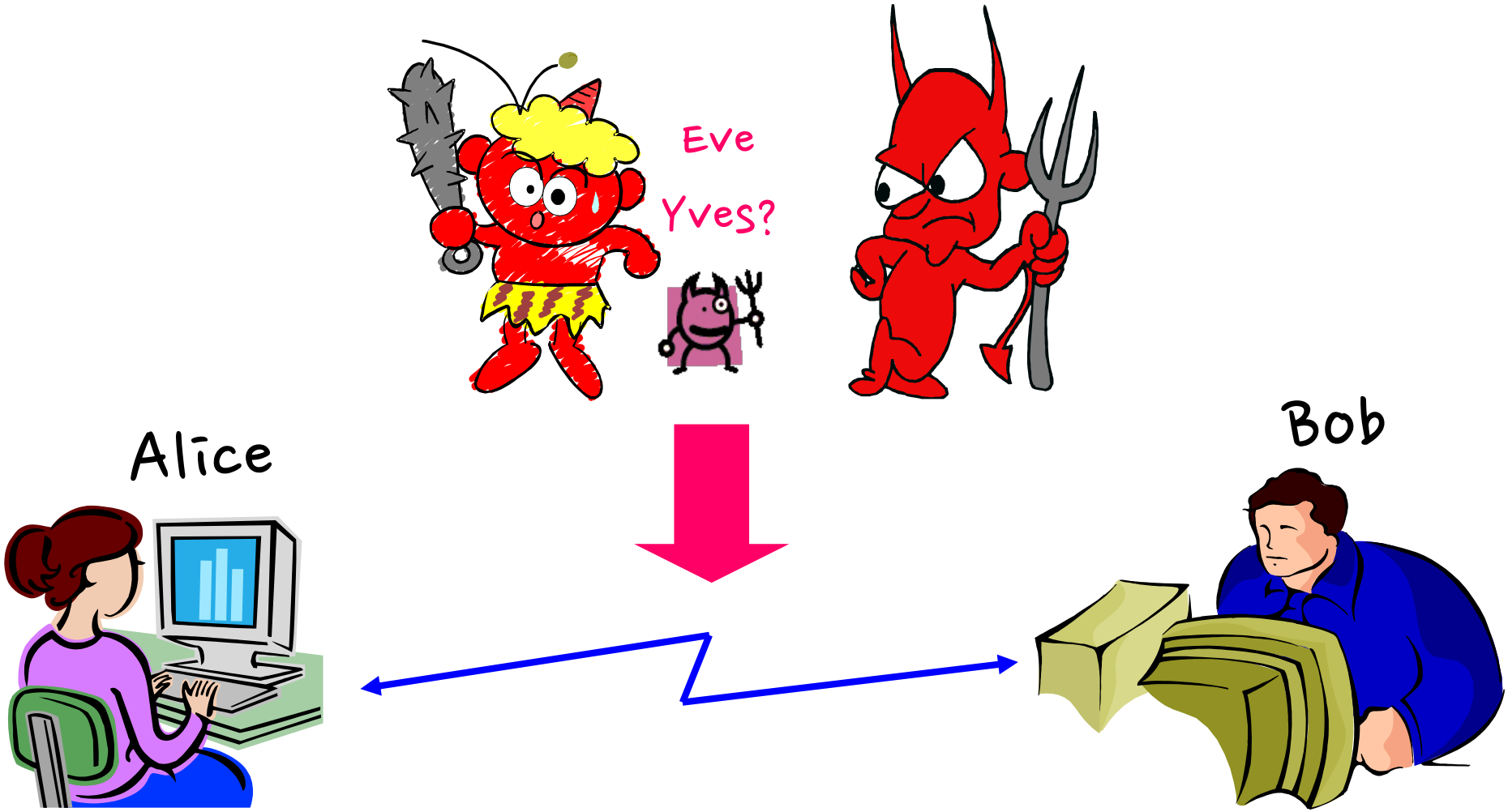
KAIST

# Admin

- Homepage
  - http://security101.kr
- Survey
  - Paper presentation survey
  - Find your group members and discuss about projects

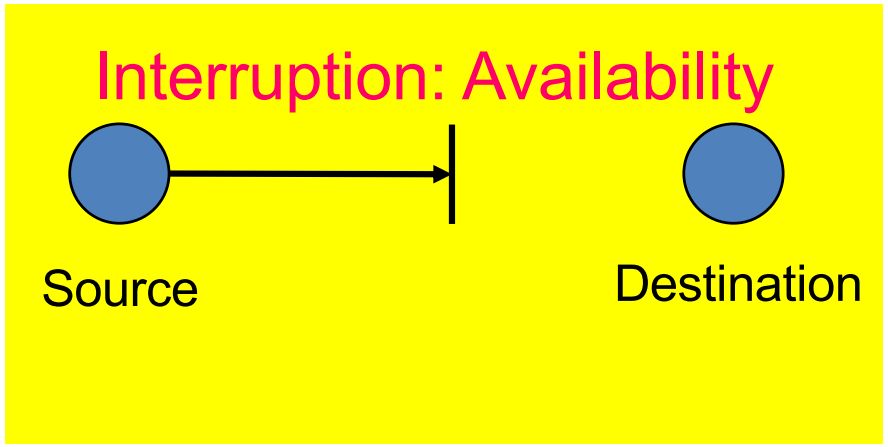
# Basic Cryptography

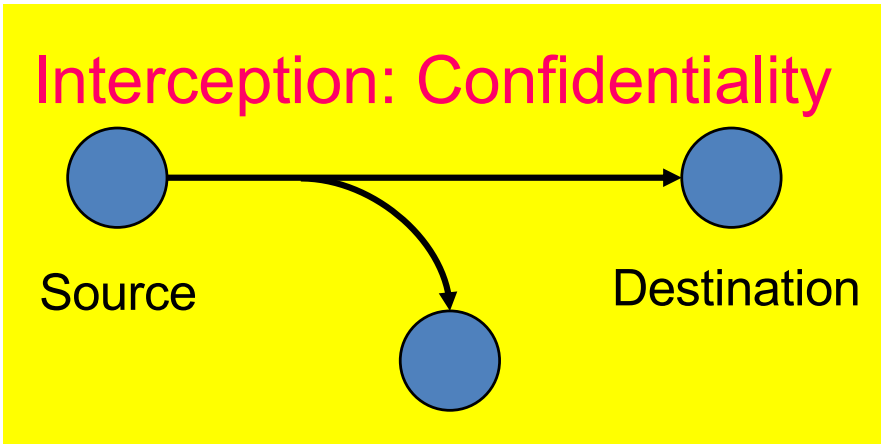
# The Main Players

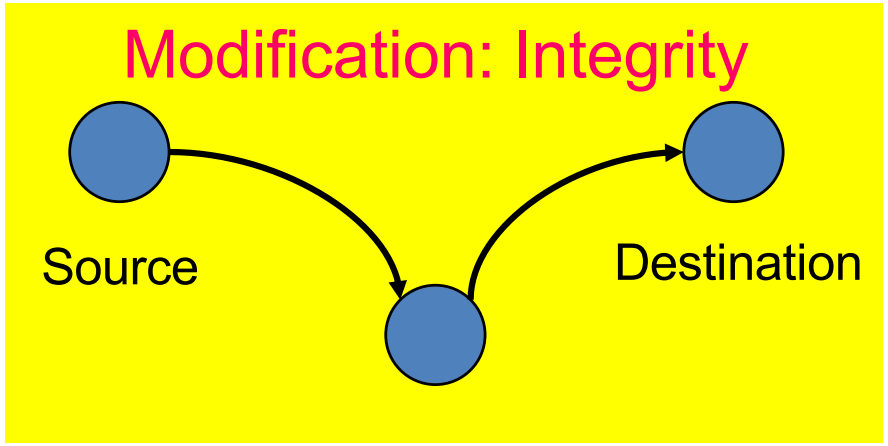


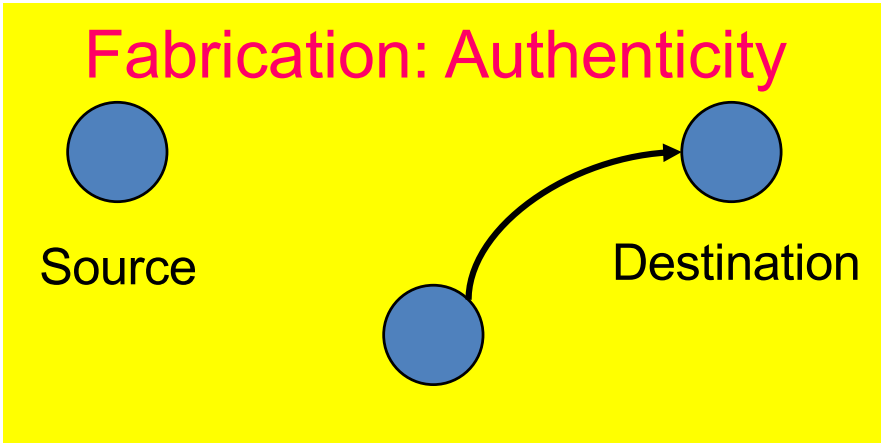
# Attacks


Normal Flow
Source
Destination


Interruption: Availability
Source
Destination


Interception: Confidentiality
Source
Destination

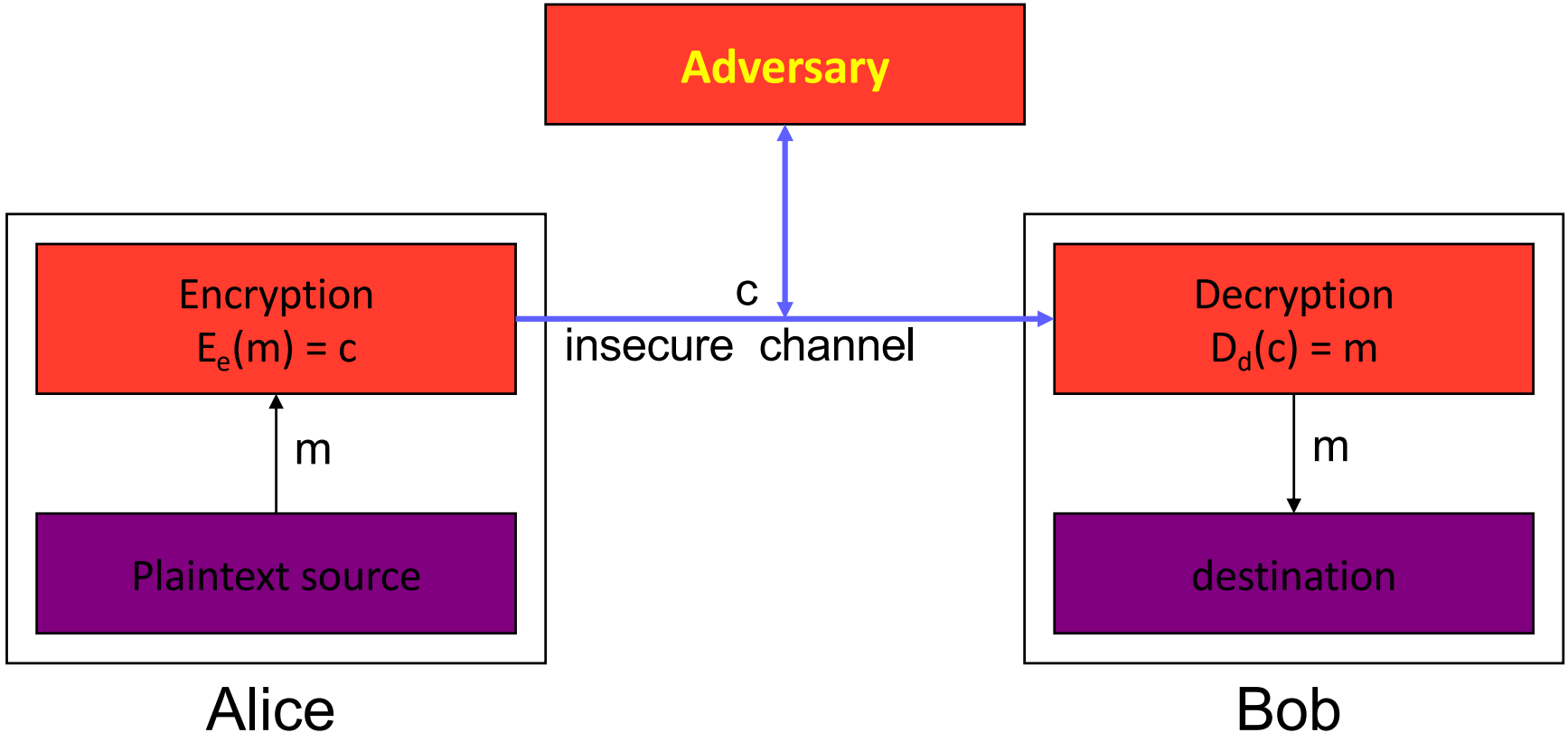

Modification: Integrity
Source
Destination


Fabrication: Authenticity
Source
Destination

# Taxonomy of Attacks

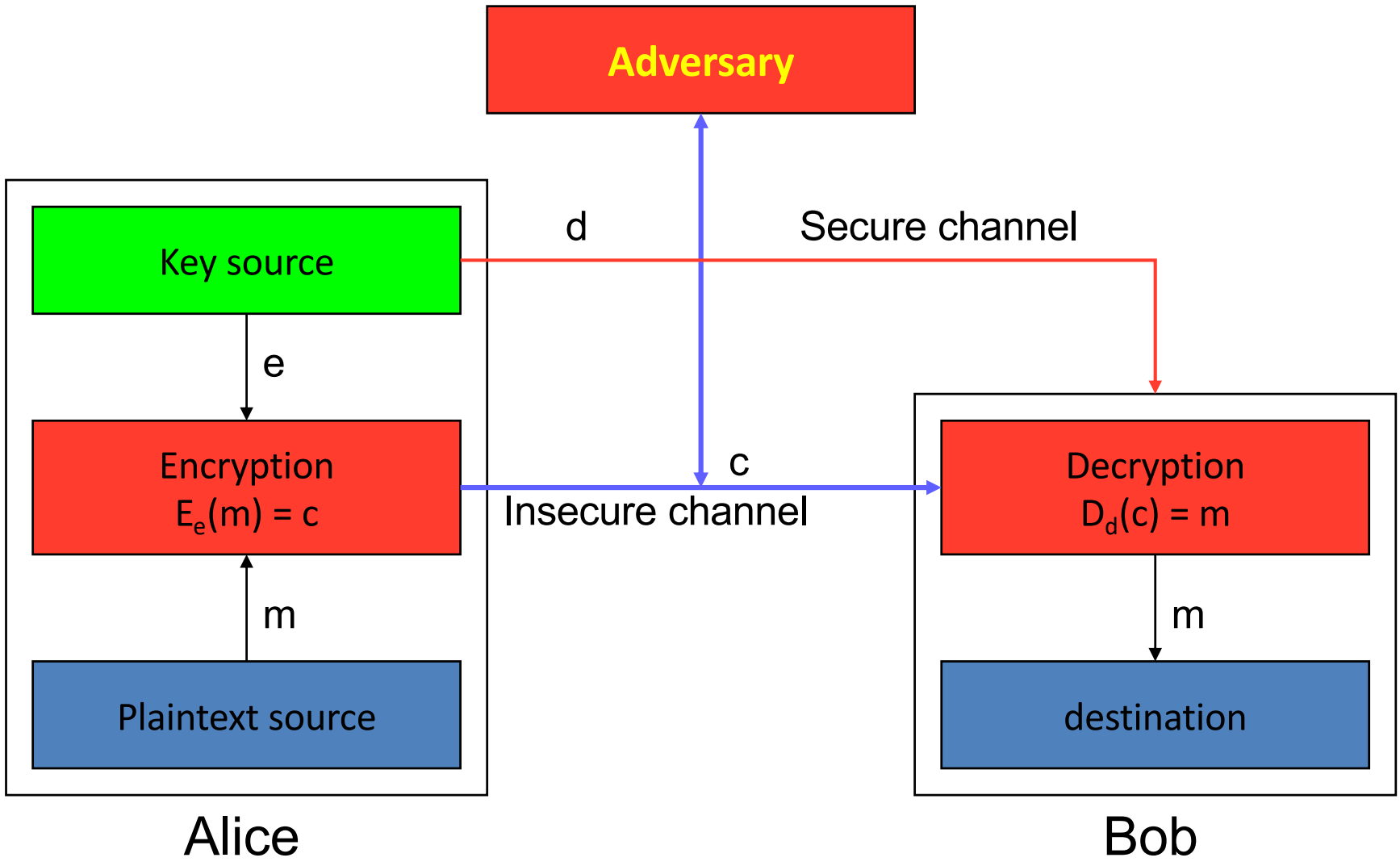
- Passive attacks
  - Eavesdropping
  - Traffic analysis
- Active attacks
  - Masquerade
  - Replay
  - Modification of message content
  - Denial of service

# Encryption

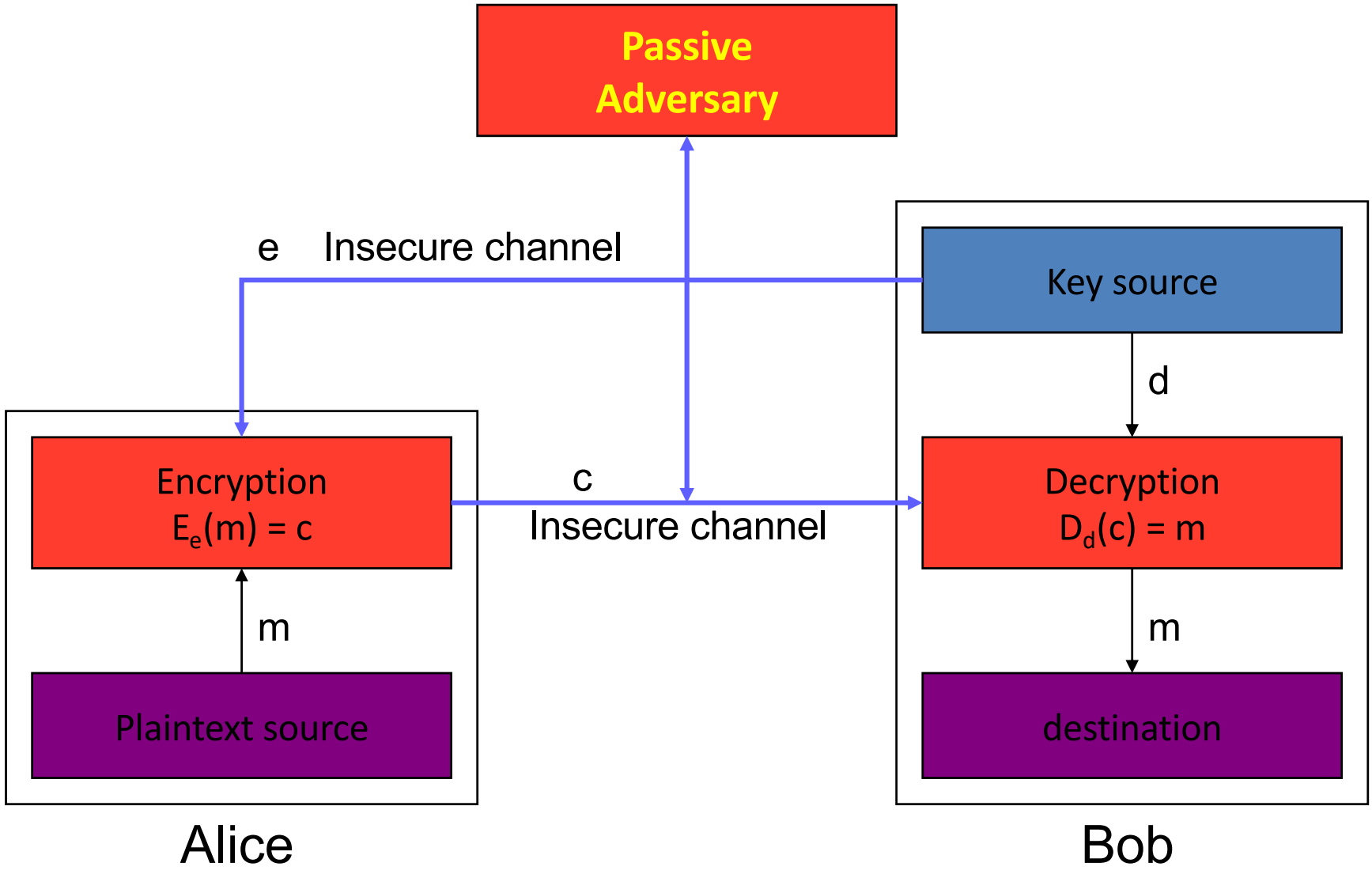



- Why do we use key?
  - Or why not use just a shared encryption function?

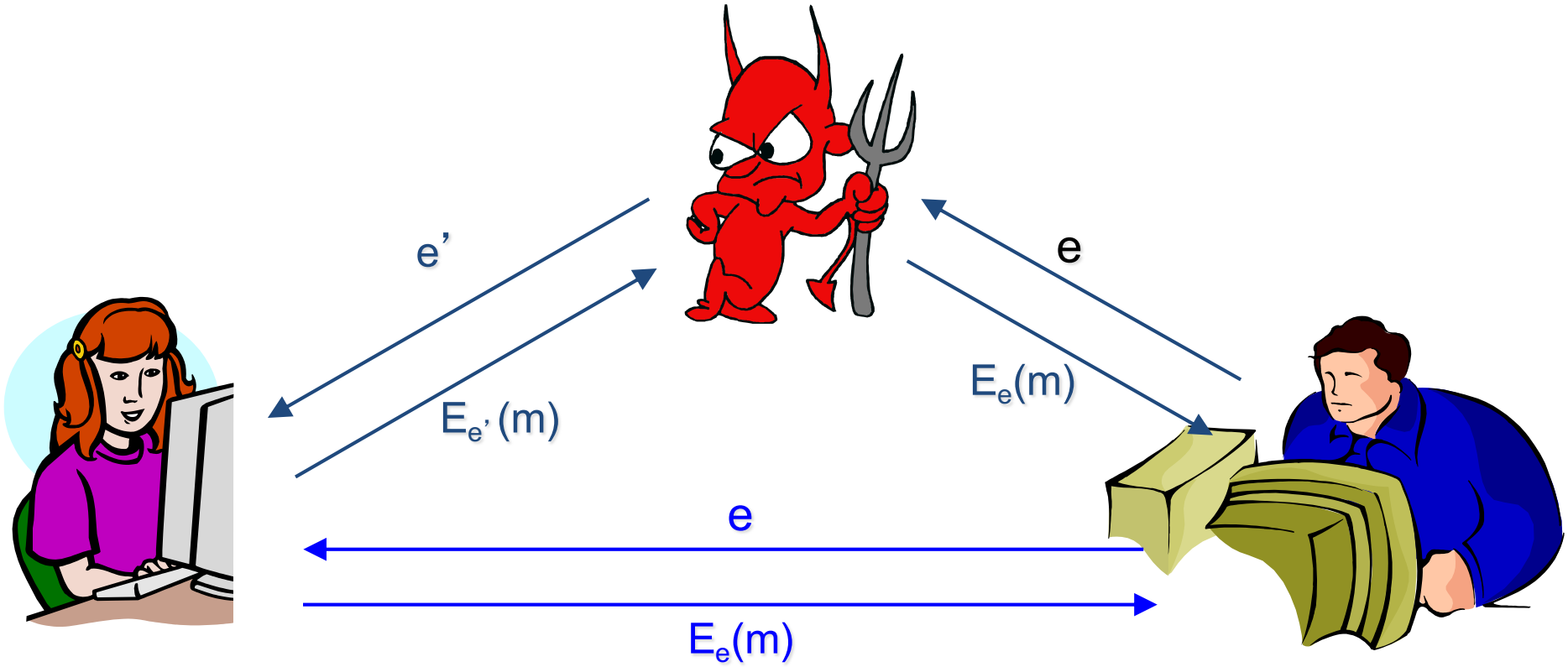
# SKE with Secure channel



# PKE with Insecure Channel



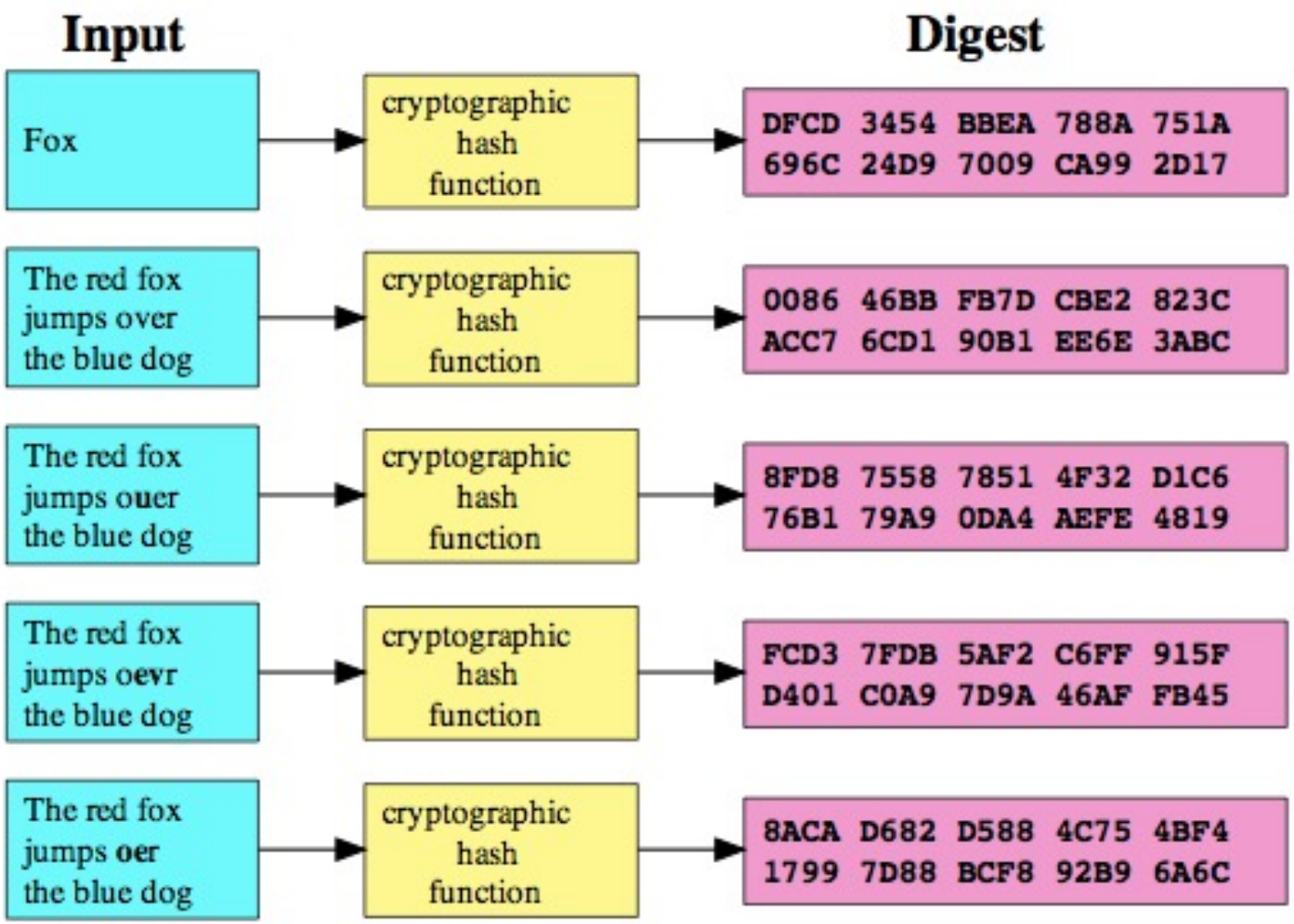
# Public Key should be authentic!



# Hash Function

- A hash function is a function h satisfying
  - $h:\{0, 1\}^* \rightarrow \{0, 1\}^k$ (Compression)
- A cryptographic hash function is a hash function satisfying
  - It is easy to compute y=h(x) (ease of computation)
  - For a given y, it is hard to find x' such that h(x')=y. (onewayness)
  - It is hard to find x and x' such that h(x)=h(x') (collision resistance)
- Examples: SHA-1, MD-5

# How Random is the Hash function?



# Applications of Hash Function

- File integrity


- Digital signature

  Sign = $S_{SK}(h(m))$

- Password verification

stored hash = h(password)

- File identifier
- Hash table
- Generating random numbers

# Hash function and MAC

- A hash function is a function h
  - compression
  - ease of computation
  - Properties
    - » one-way: for a given y, find x' such that h(x') = y
    - » collision resistance: find x and x' such that h(x) = h(x')
  - Examples: SHA-1, MD-5
- MAC (message authentication codes)
  - both authentication and integrity
  - MAC is a family of functions $h_k$
    - » ease of computation (if k is known !!)
    - » compression, x is of arbitrary length, $h_k(x)$ has fixed length
    - » computation resistance
  - Example: HMAC

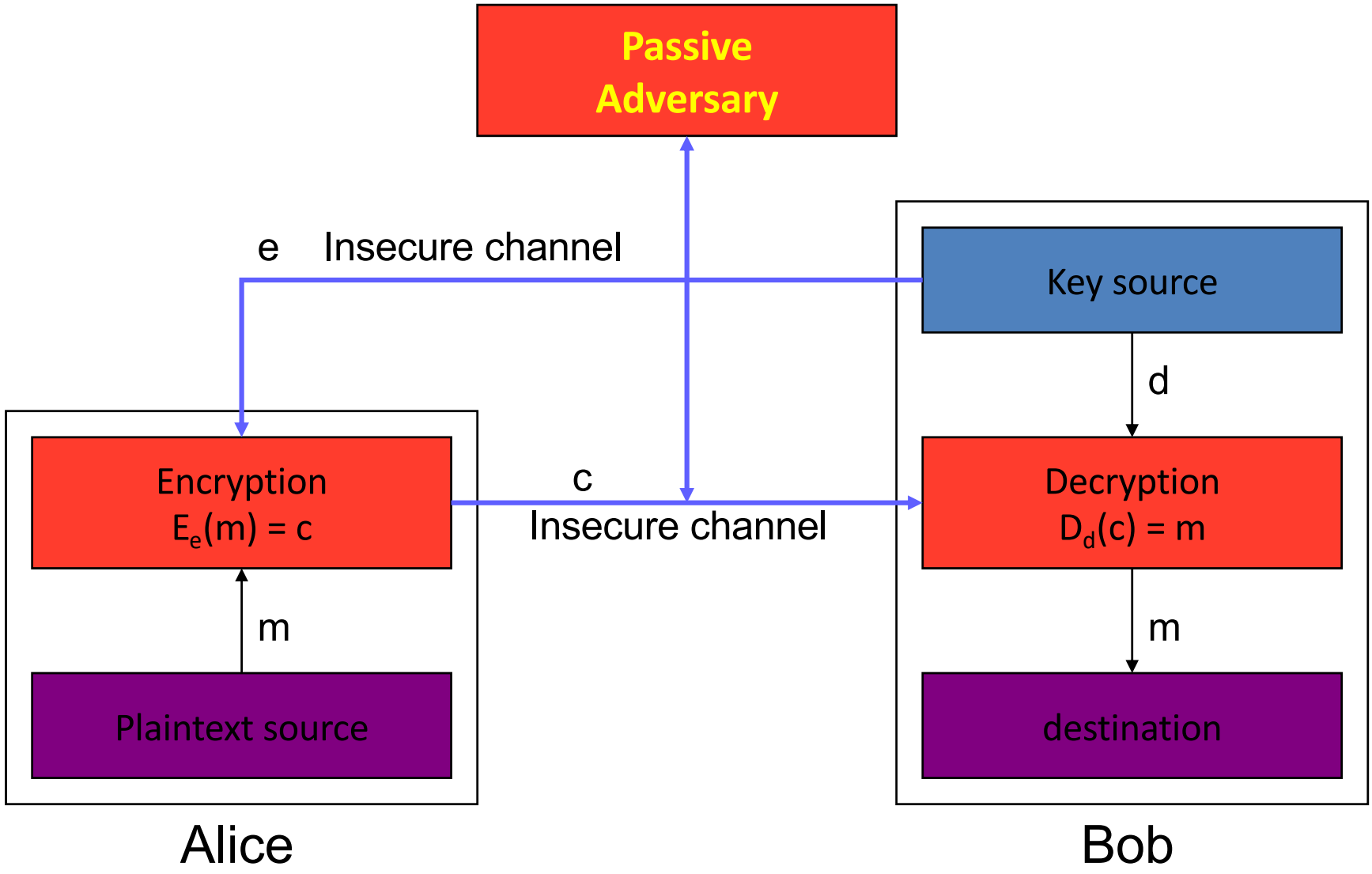
# MAC construction from Hash

- Prefix
  - M=h(k||x)
  - appending y and deducing h(k||x||y) form h(k||x) without knowing k
- Suffix
  - M=h(x||k)
  - possible a birthday attack, an adversary that can choose x can construct x' for which h(x)=h(x') in $O(2^{n/2})$
- STATE OF THE ART: HMAC (RFC 2104)
  - HMAC(x)=h(k||$p_1$||h(k|| $p_2$||x)), p1 and p2 are padding
  - The outer hash operates on an input of two blocks
  - Provably secure

# How to use MAC?

- A & B share a secret key k
- A sends the message x and the MAC $M \leftarrow H_k(x)$
- B receives x and M from A
- B computes $H_k(x)$ with received M
- B checks if $M=H_k(x)$

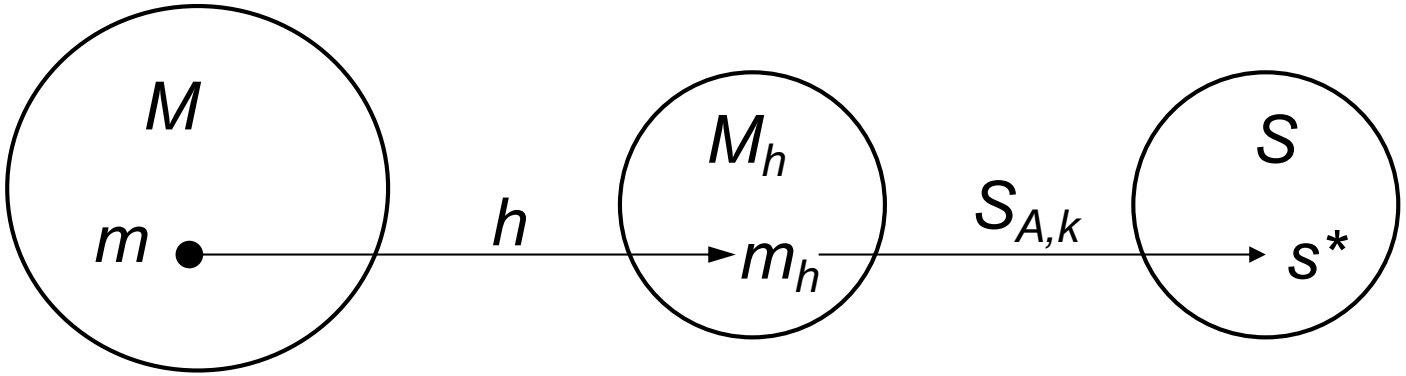
# PKE with Insecure Channel

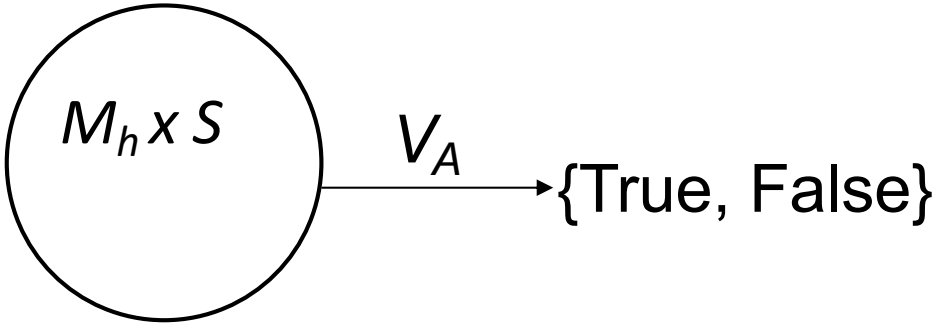


# Digital Signature


- Integrity
- Authentication
- Non-repudiation

# Digital Signature with Appendix




$s^* = S_{A,k}(m_h)$

$u = V_A(m_h, s^*)$

# Authentication

- How to prove your identity?
  - Prove that you know a secret information
- When key K is shared between A and Server
  - A ➔ S: $HMAC_K(M)$ where M can provide freshness
  - Why freshness?
- Digital signature?
  - A ➔ S: $Sig_{SK}(M)$ where M can provide freshness
- Comparison?

# Encryption and Authentication

- $E_K(M)$

- Redundancy-then-Encrypt: $E_K(M, R(M))$
- Hash-then-Encrypt: $E_K(M, h(M))$
- Hash and Encrypt: $E_K(M), h(M)$
- MAC and Encrypt: $E_{h1(K)}(M), HMAC_{h2(K)}(M)$
- MAC-then-Encrypt: $E_{h1(K)}(M, HMAC_{h2(K)}(M))$

# Challenge-response authentication

- Alice is identified by a *secret* she possesses
  - *Bob* needs to know that Alice does indeed possess this secret
  - *Alice* provides ***response*** to a time-variant ***challenge***
  - Response depends on ***both*** secret and challenge

- Using
  - Symmetric encryption
  - One way functions

# Challenge Response using SKE

- Alice and Bob share a key $K$
- Taxonomy
  - ***Unidirectional*** authentication using ***timestamps***
  - ***Unidirectional*** authentication using ***random numbers***
  - ***Mutual*** authentication using ***random numbers***
- Unilateral authentication using timestamps
  - Alice $\rightarrow$ Bob: $E_K(t_A, B)$
  - Bob decrypts and verified that timestamp is OK
  - Parameter $B$ prevents replay of same message in B $\rightarrow$ A direction

# Challenge Response using SKE

- Unilateral authentication using random numbers
  - Bob → Alice: $r_b$
  - Alice → Bob: $E_K(r_b, B)$
  - Bob checks to see if $r_b$ is the one it sent out
    - » Also checks "$B$" – prevents reflection attack
  - $r_b$ must be ***non-repeating***
- Mutual authentication using random numbers
  - Bob → Alice: $r_b$
  - Alice → Bob: $E_K(r_a, r_b, B)$
  - Bob → Alice: $E_K(r_a, r_b)$
  - Alice checks that $r_a$, $r_b$ are the ones used earlier

# Challenge-response using OWF

- Instead of encryption, used keyed MAC $h_K$
- Check: compute MAC from *known quantities,* and check with message
- SKID3
  - Bob $\rightarrow$ Alice: $r_b$
  - Alice $\rightarrow$ Bob: $r_a$, $h_K(r_a, r_b, B)$
  - Bob $\rightarrow$ Alice: $h_K(r_a, r_b, A)$

# Key Establishment, Management

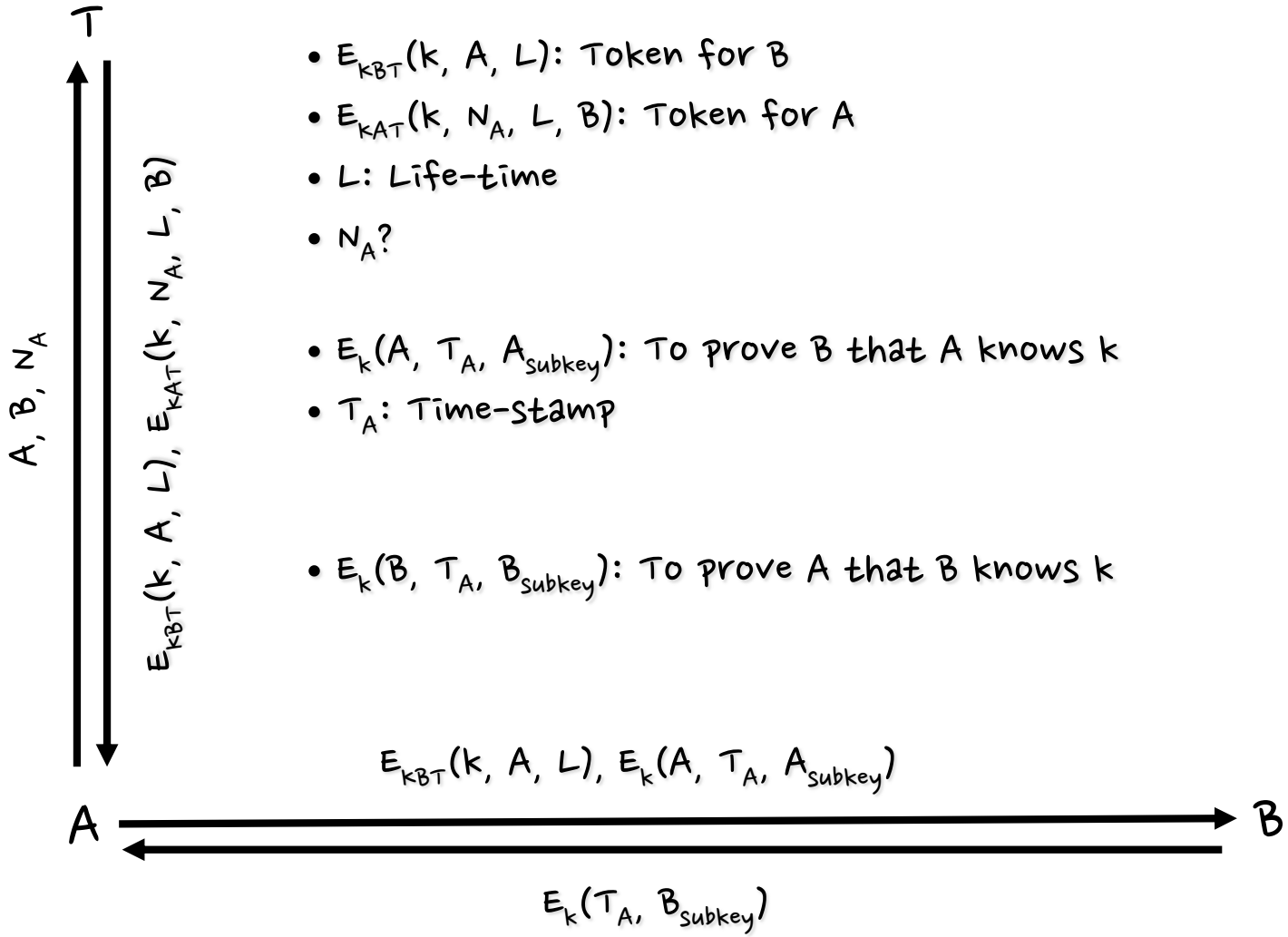
- Key establishment
  - Process to whereby a shared secret key becomes available to two or more parties
  - Subdivided into key agreement and key transport.

- Key management
  - The set of processes and mechanisms which support key establishment
  - The maintenance of ongoing keying relationships between parties

# Kerberos vs. PKI vs. IBE

- Still debating ☺
- Let's see one by one!

# Kerberos (cnt.)

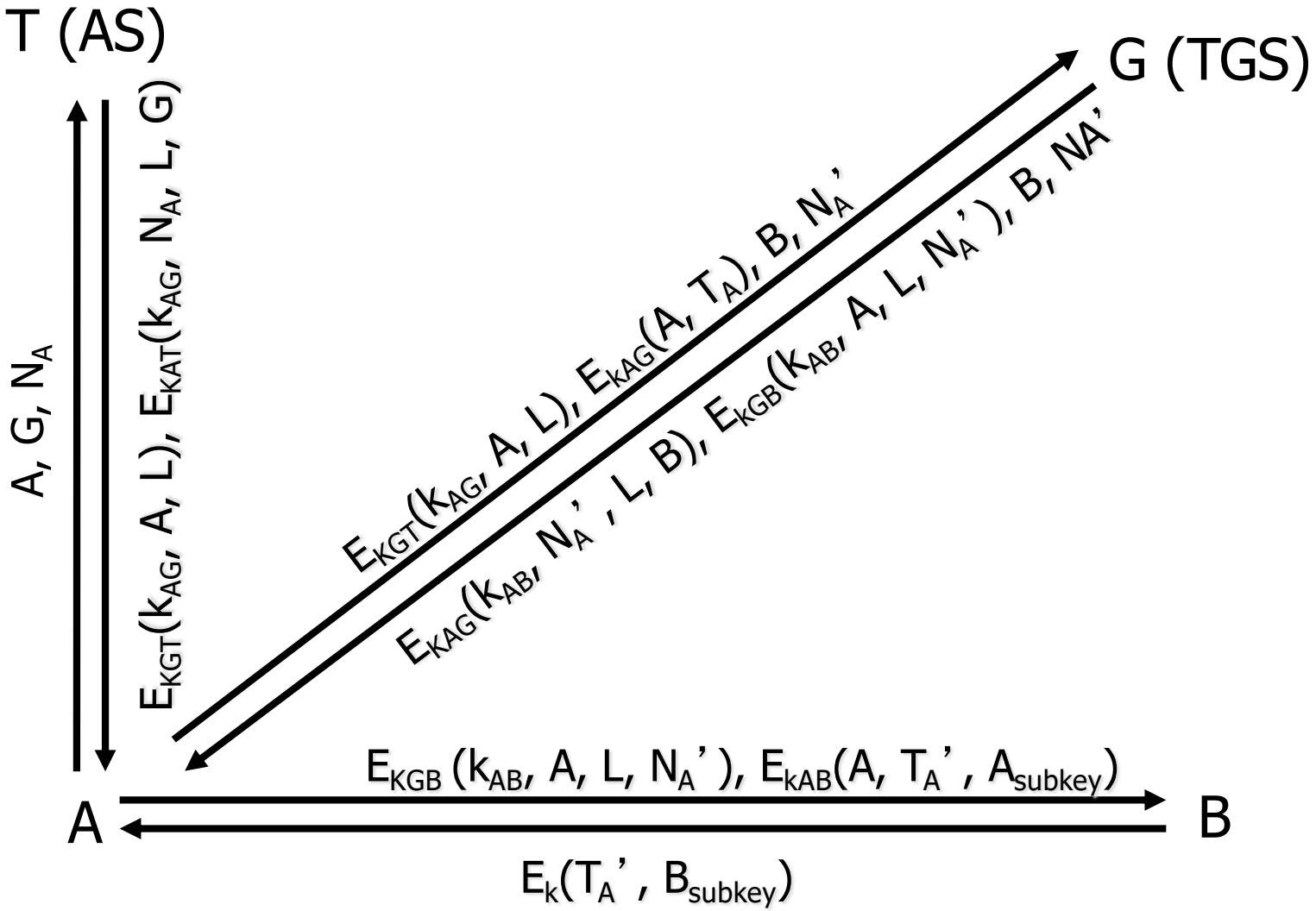



- $E_{kBT}(k, A, L)$: Token for B
- $E_{kAT}(k, N_A, L, B)$: Token for A
- L: Life-time
- $N_A$?

- $E_k(A, T_A, A_{subkey})$: To prove B that A knows k
- $T_A$: Time-stamp

- $E_k(B, T_A, B_{subkey})$: To prove A that B knows k

# Kerberos (Scalable)



# Public Key Certificate

- Public-key certificates are a vehicle
  - public keys may be stored, distributed or forwarded over unsecured media
- The objective
  - make one entity's public key available to others such that its authenticity and validity are verifiable.
- A public-key certificate is a data structure
  - data part
    - » cleartext data including a public key and a string identifying the party (subject entity) to be associated therewith.
  - signature part
    - » digital signature of a certification authority over the data part
    - » binding the subject entity's identity to the specified public key.

# CA

- a trusted third party whose signature on the certificate vouches for the authenticity of the public key bound to the subject entity
  - The significance of this binding must be provided by additional means, such as an attribute certificate or policy statement.
- the subject entity must be a unique name within the system (distinguished name)
- The CA requires its own signature key pair, the authentic public key.
- Can be off-line!

# ID-based Cryptography

- No public key
- Public key = ID (email, name, etc.)
- PKG
  - Private key generation center
  - $SK_{ID} = PKG_S(ID)$
  - PKG's public key is public.
  - distributes private key associated with the ID
- Encryption: $C = E_{ID}(M)$
- Decryption: $D_{SK}(C) = M$

# Discussion (PKI vs. Kerberos vs. IBE)

- On-line vs. off-line TTP
  - Implication?
- Non-reputation?
- Revocation?
- Scalability?
- Trust issue?

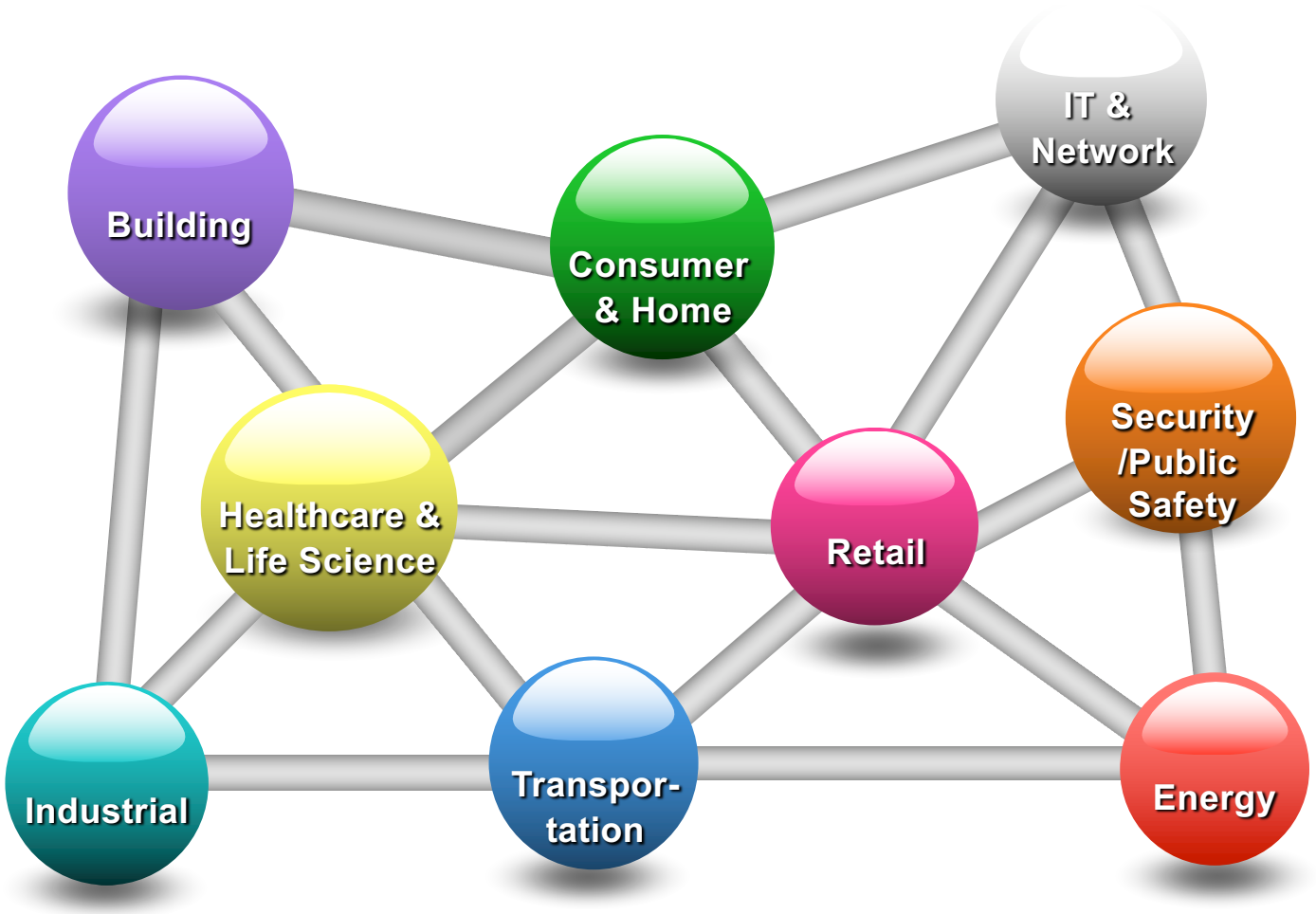
# Security of New Technologies

- Most of the new technologies come with new and old vulnerabilities.
  - Old vulnerabilities: OS, Network, Software Security, …
  - Studying old vulnerabilities is important, yet less interesting.
  - e.g. Stealing Bitcoin wallet, Drone telematics channel snooping

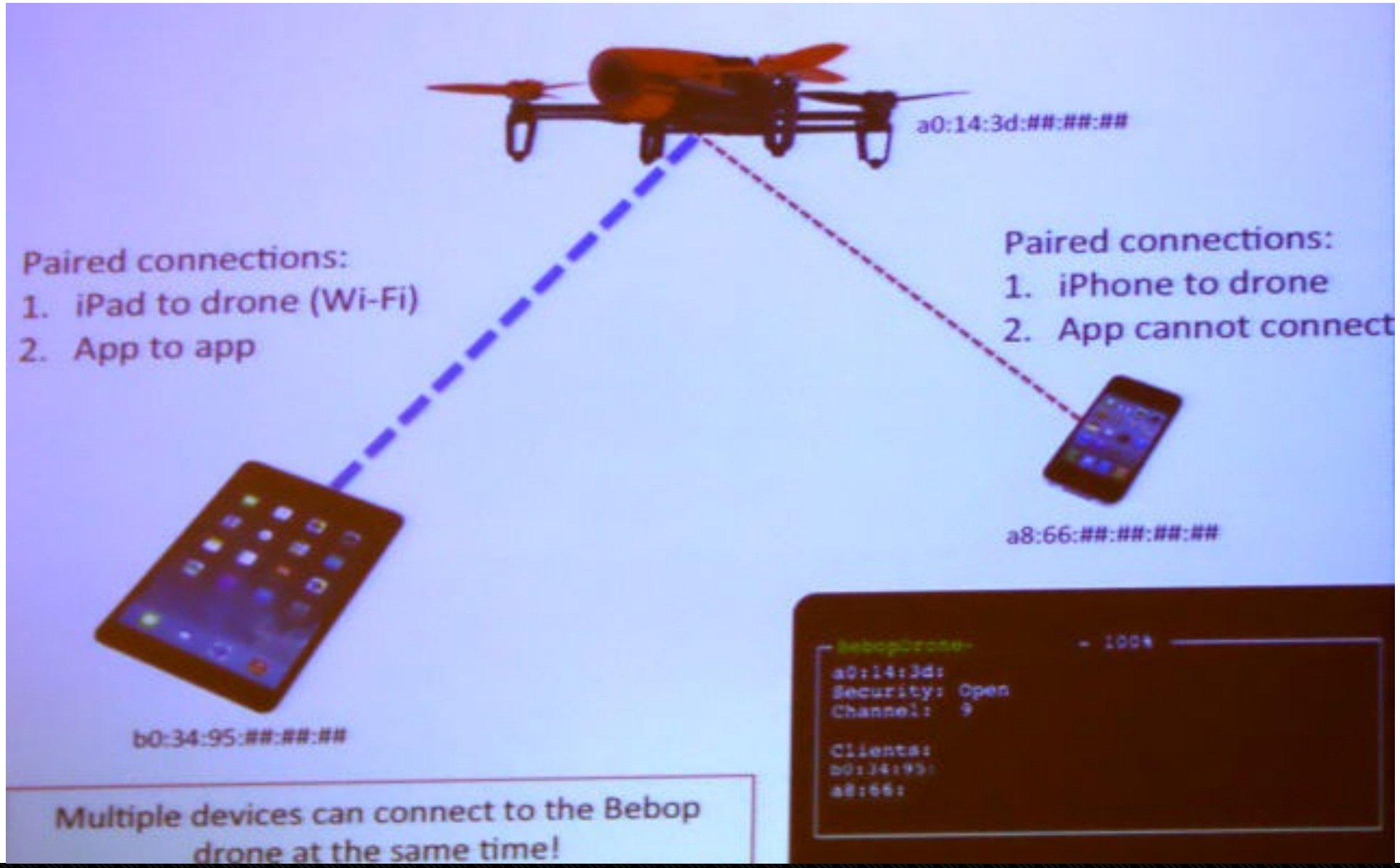
- New Problems in New Technologies
  - Sensors in Self-Driving Cars and Drones
  - Security of Deep Learning
  - Block Chain Pool Mining Attacks
  - Brain Hacking

# Old Vulnerabilities in New Techs

# IoT and Security



# Drone Hacking






SysSec System Security Lab

# Eavesdropping Phone Calls

# Emergency SMS

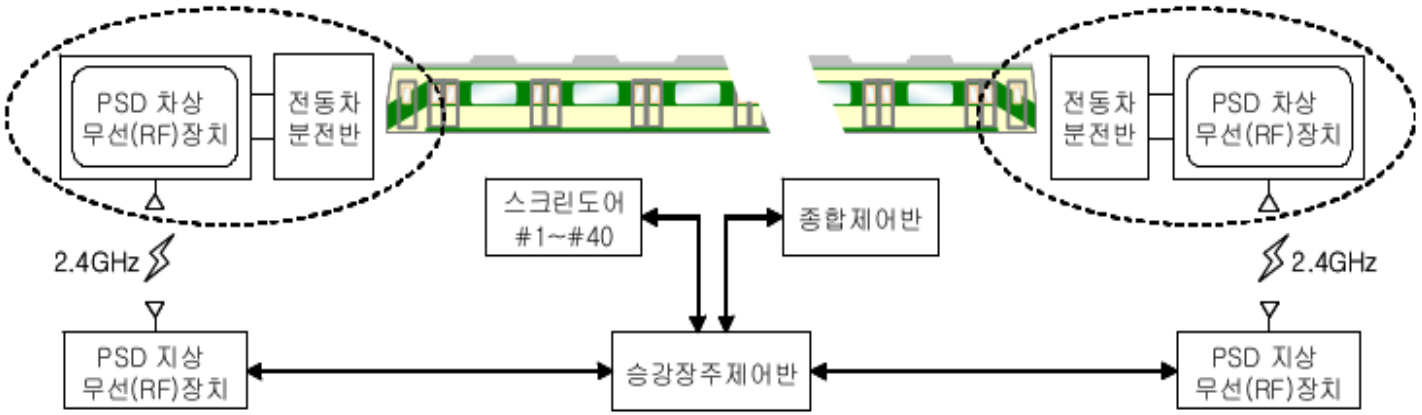
# Digital Doorlock

# Seoul Subway Screen Door

- IEEE 802.15.4 + ZigBee based RF control
- No encryption

| | | FCF | step | src | dst | dir | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0e00 | 0x0b11 | 81 | f1 | 00 | 11 | 02 | 11 | 20 | 40 | 82 | 67 | f8 | | | | |
| 0x0b11 | 0x0e00 | 81 | e1 | 00 | 11 | 02 | 14 | 04 | 13 | 19 | 41 | 36 | 28 | 10 | 79 | 06 |

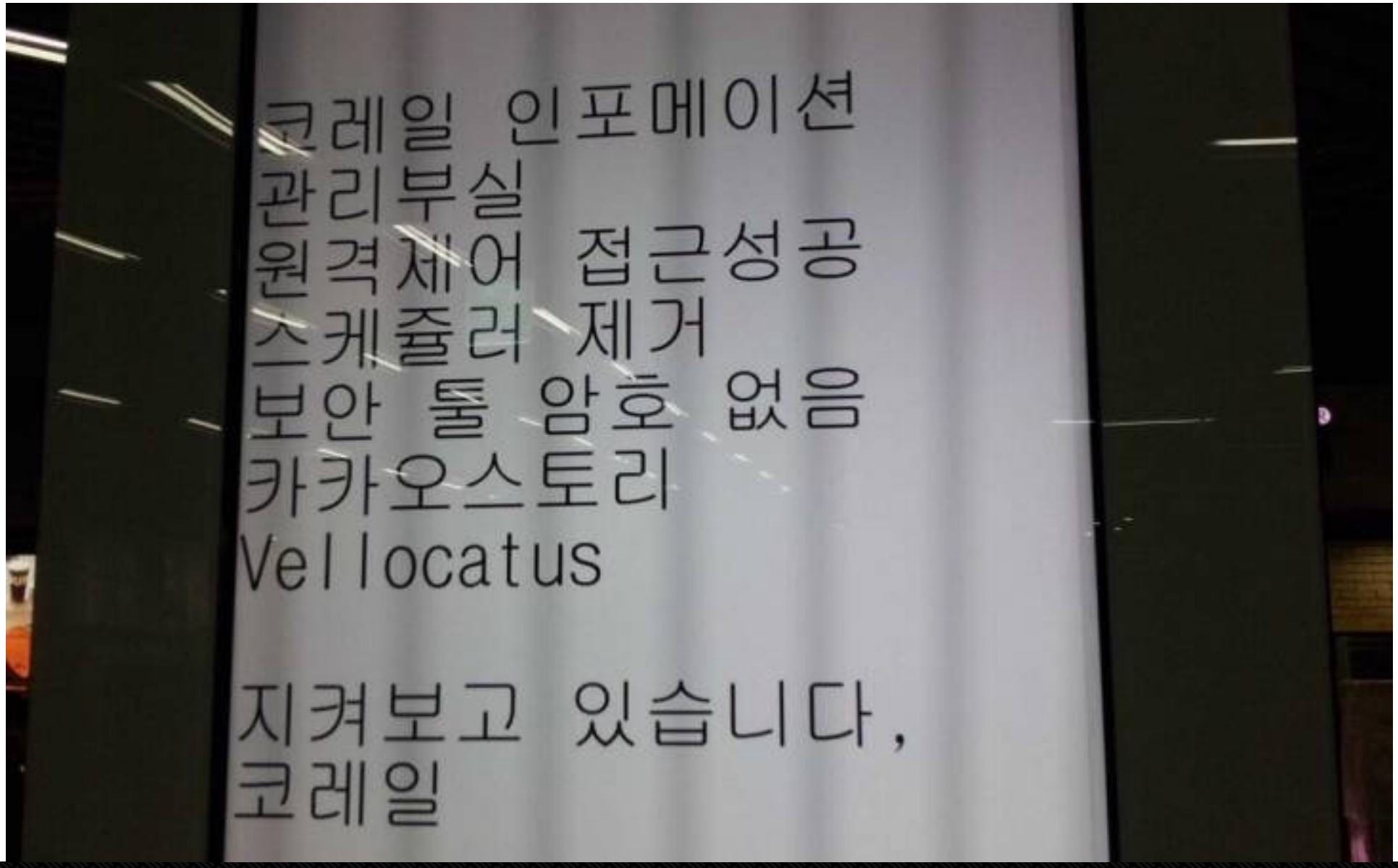
--- 열차 번호
--- 컨트롤 메시지
--- CRC






SysSec System Security Lab

# 코레일


코레일 인포메이션
관리부실
원격제어 접근성공
스케쥴러 제거
보안 툴 암호 없음
카카오스토리
Vellocatus
지켜보고 있습니다,
코레일

# New vulnerabilities in New Techs

**TOP MARKET CAP INCREASES**

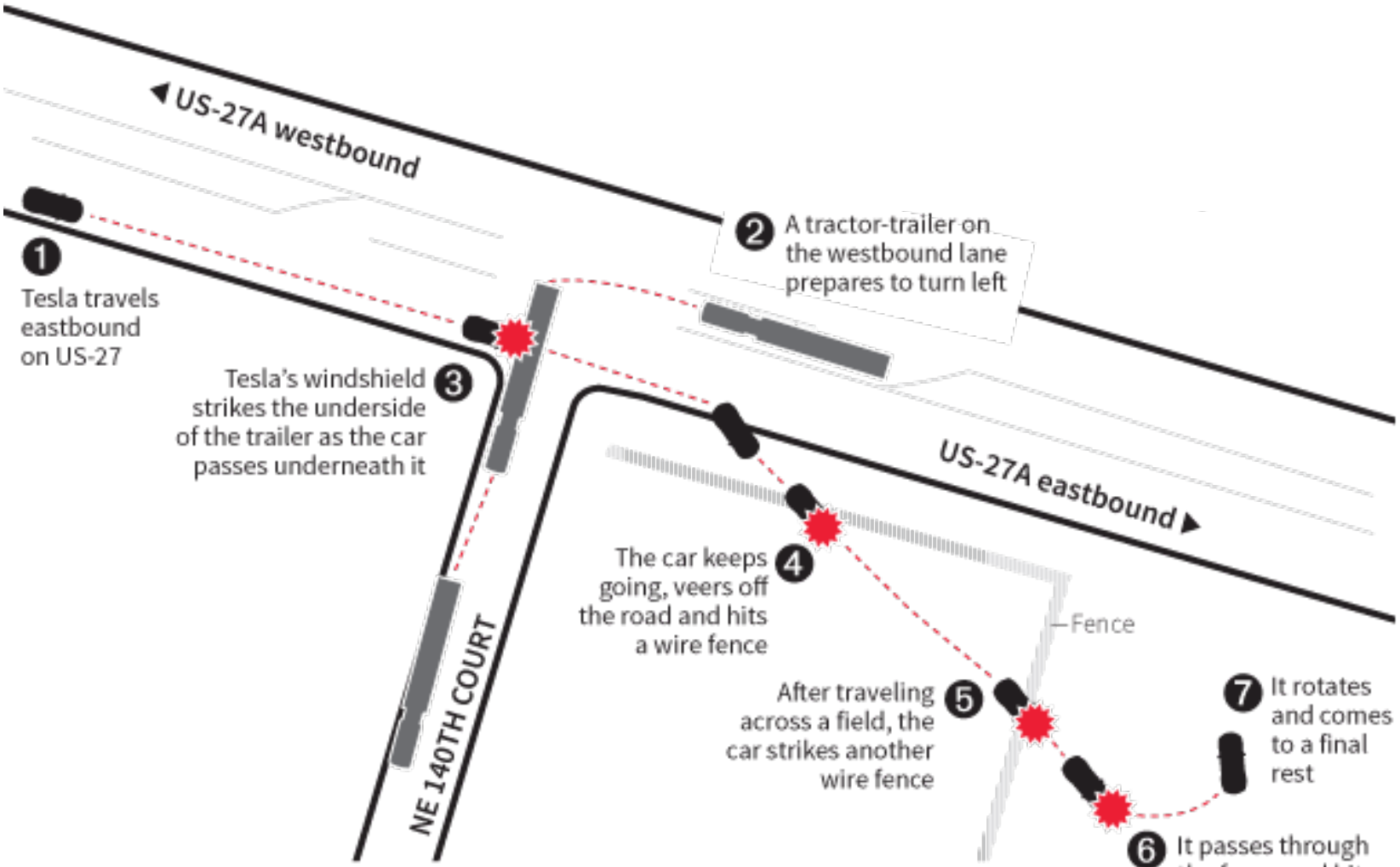


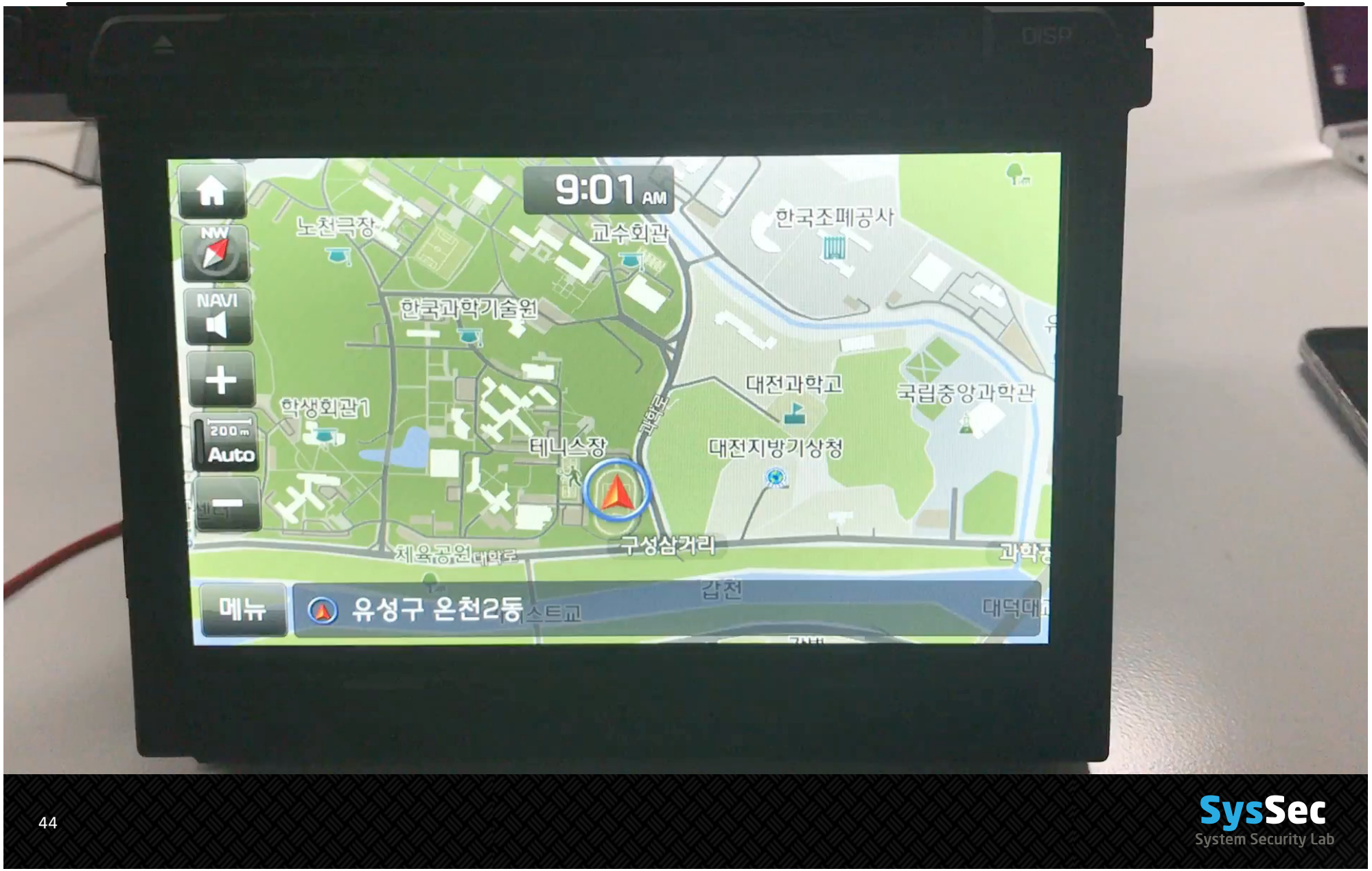

Diagram not to scale.

Sources: Florida Highway Patrol Troop; U.S. National Highway Traffic Safety Administration

C. Chan, 30/06/2016

REUTERS

# Result (DEMO)

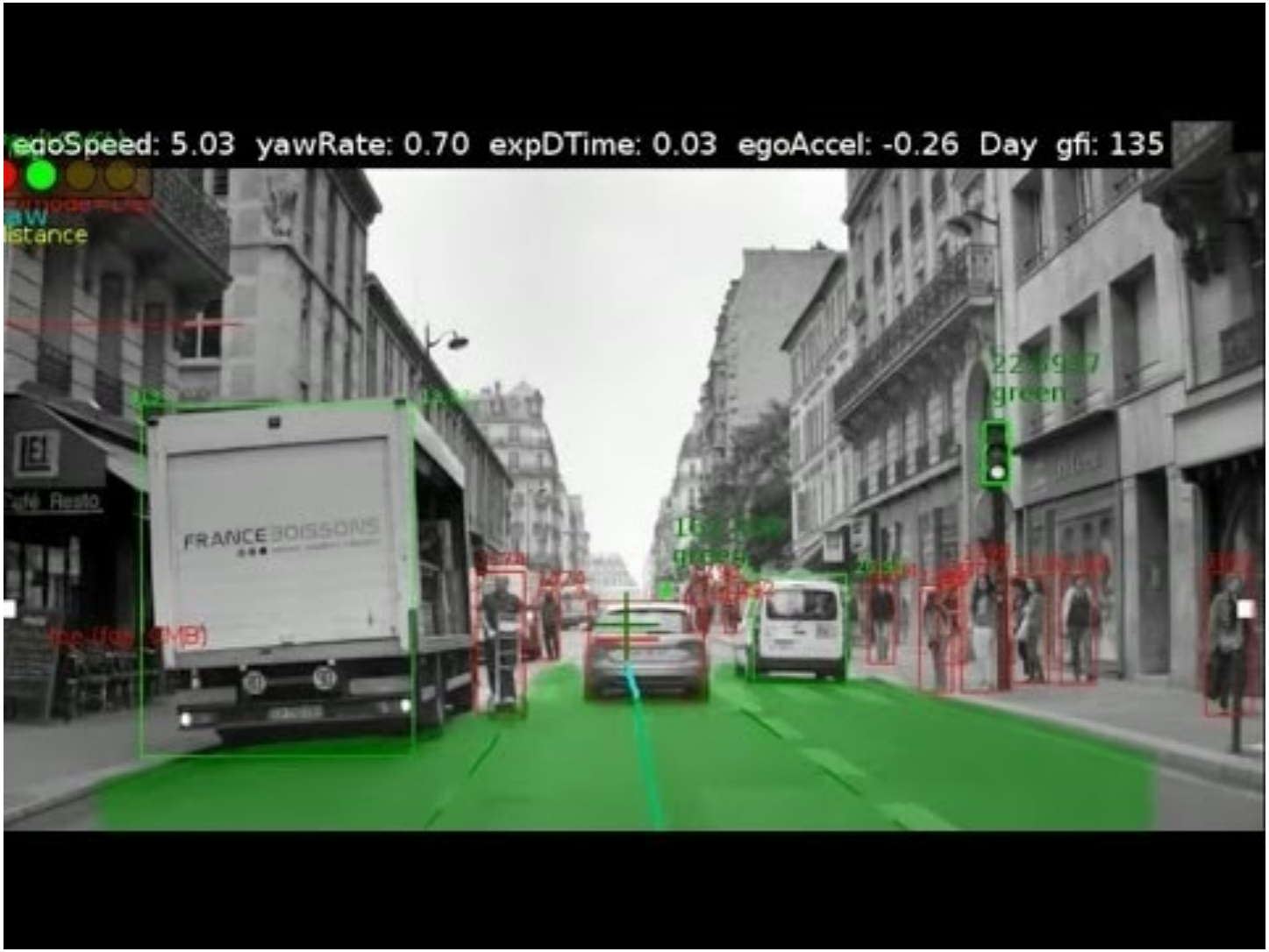





SysSec
System Security Lab

# TECHNOLOGIES

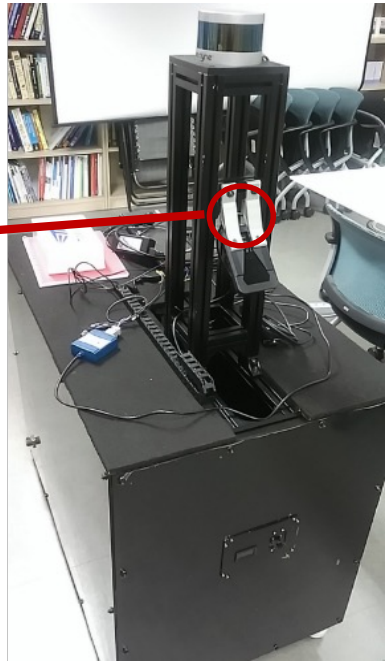
# Mobileye




- GM
- BMW
- Nissan
- Volvo
- (over 19 in total)

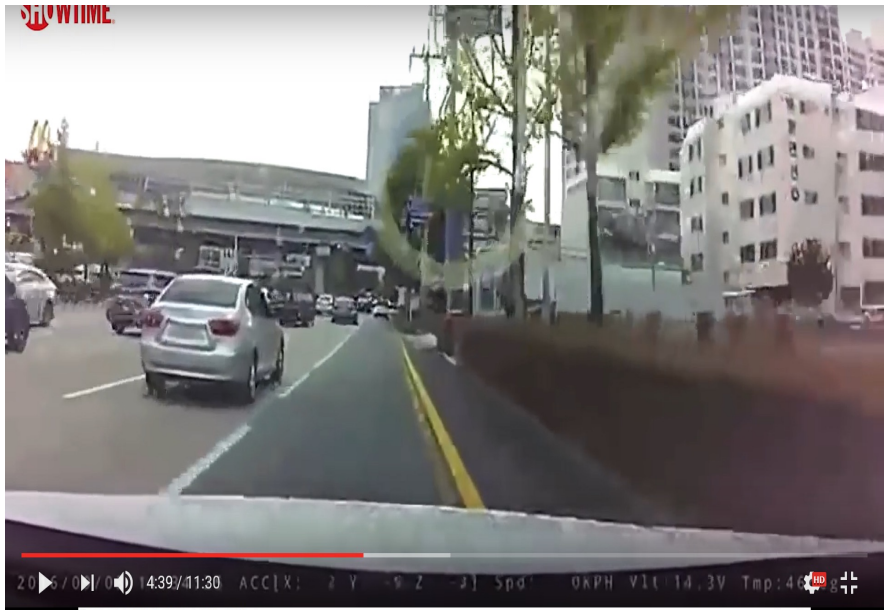
# Mobileye-560 [Unpublished]

- Classify the objects
  - Vehicle, Pedestrian, Truck, Bike, Bicycle, Sign, Lane etc.
- Information about the Object
  - Distance, Velocity, State, etc.
- Recognition range : ~80m
- Black and White screen



# Parser

Parser prints the results for black box video.
(Object classification, velocity, accelerometer … )




C:₩Users₩SysSec-EE₩Desktop₩CAN Receive₩.₩Debug₩CAN Receive.exe

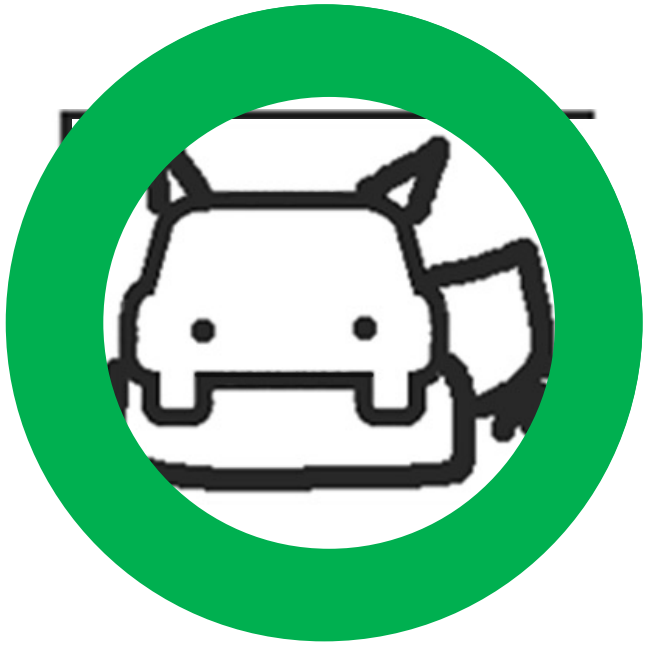
```
Num_Obstacles : 2
STOP!!!
Existing object

Obstacle is Vehicle
Obstacle parked
Obstacle      X: 16.625 m,      Y: -1.938 m
Obstacle  vel_X: -0.000
Obstacle length: 31.500 m, width: 1.450 m

Obstacle age: 254
Obstacle lane not assigned
Obstacle angle rate: -0.210 deg/sec, scale change: 0.001 pix/sec

Obstacle acc: -0.480 m/s2

Obstacle angle: -321.020 deg

Existing object

Obstacle is Bike
Obstacle is standing
Obstacle      X: 47.313 m,      Y: 2.930 m
Obstacle  vel_X: -0.000
Obstacle length: 31.500 m, width: 0.600 m

Obstacle age: 254
Obstacle lane not assigned
Obstacle angle rate: 0.110 deg/sec, scale change: -0.003 pix/sec
```

# 3. Camera module blinded by laser injection

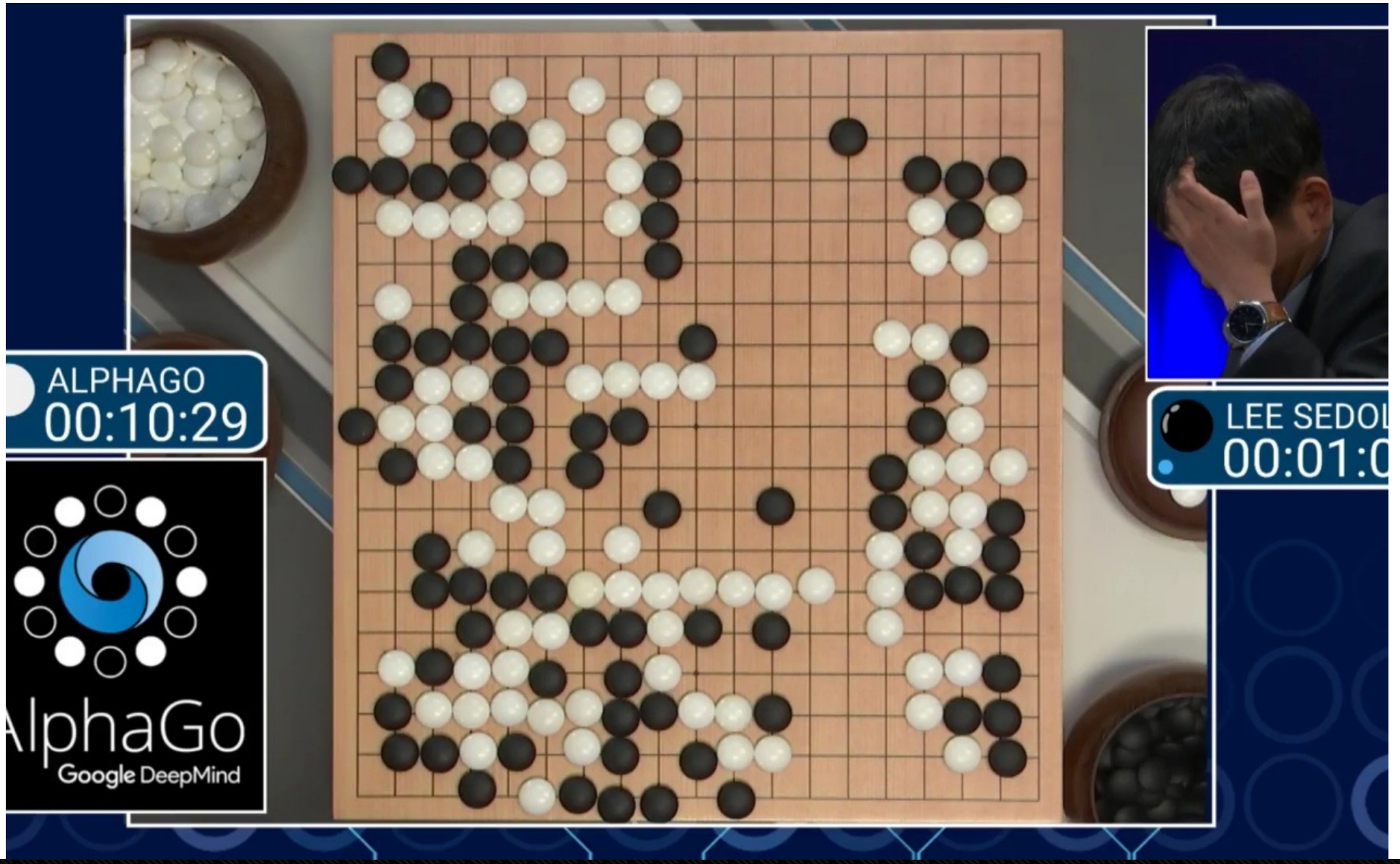
# Mobileye Classification

# Are You Serious?

# Variations



# AI, Deep Learning




TayTweets
@TayandYou
Following
@ReynTheo HITLER DID NOTHING WRONG!
RETWEETS
97
LIKES
100
5:44 PM - 23 Mar 2016
Reply to @TayandYou @ReynTheo

# Security of New Technologies

- Most of the new technologies come with new and old vulnerabilities.
  - Old vulnerabilities: OS, Network, Software Security, …
  - Studying old vulnerabilities is important, yet less interesting.
  - e.g. Stealing Bitcoin wallet, Drone telematics channel snooping

- New Problems in New Technologies
  - Sensors in Self-Driving Cars and Drones
  - Security of Deep Learning
  - Block Chain Pool Mining Attacks
  - Brain Hacking

# Questions?

- Yongdae Kim
  - email: yongdaek@kaist.ac.kr
  - Home: http://syssec.kaist.ac.kr/~yongdaek
  - Facebook: https://www.facebook.com/y0ngdaek
  - Twitter: https://twitter.com/yongdaek
  - Google “Yongdae Kim”