Entity Authentication

CSIE 7190 Cryptography and Network Security, Spring 2019

https://ceiba.ntu.edu.tw/1072csie_cns

cns@csie.ntu.edu.tw

Hsu-Chun Hsiao

Housekeeping

4/02: No class next week!

4/07 23:59: HW1 deadline

- Ask early if you have questions

4/09 2pm: Reading critique #6 deadline

4/09: Final project topic & members

4/16: 1st midterm exam

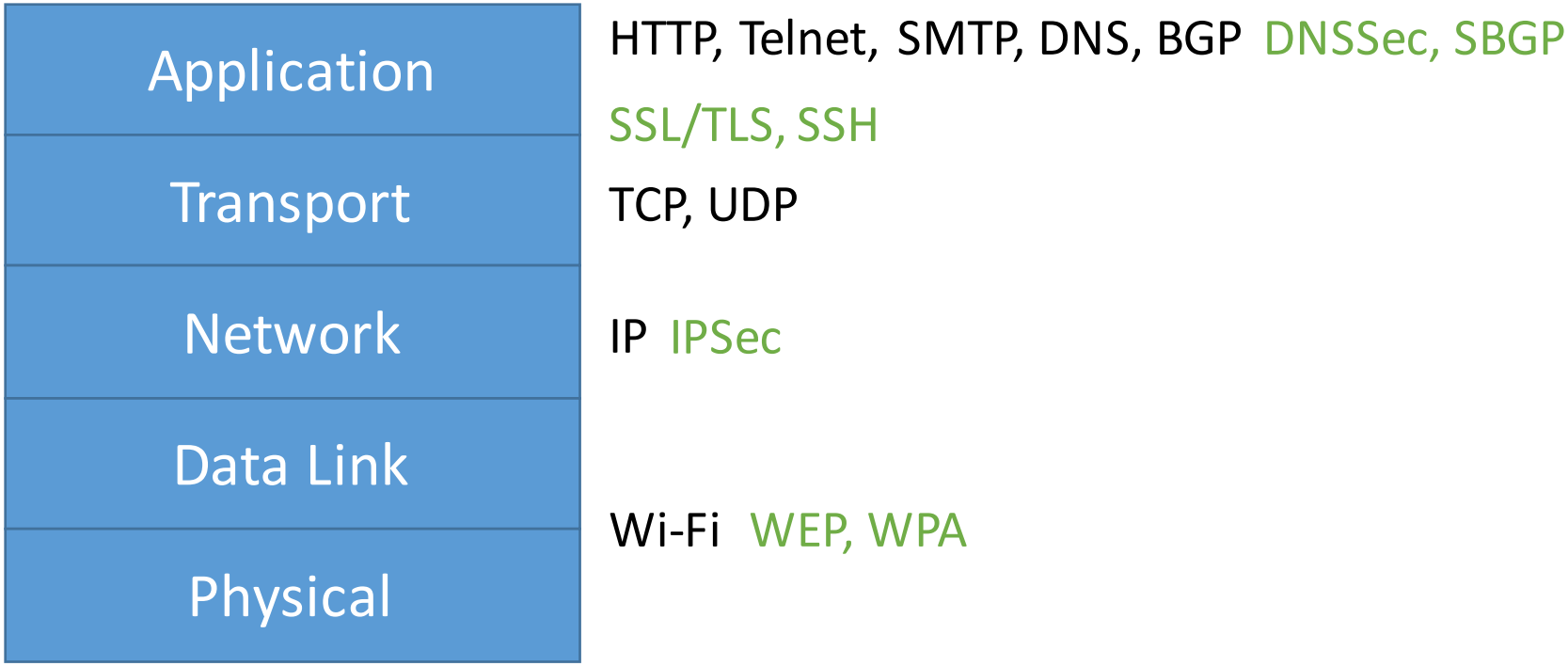
Reading critique #6

Write a critique on one of the following:

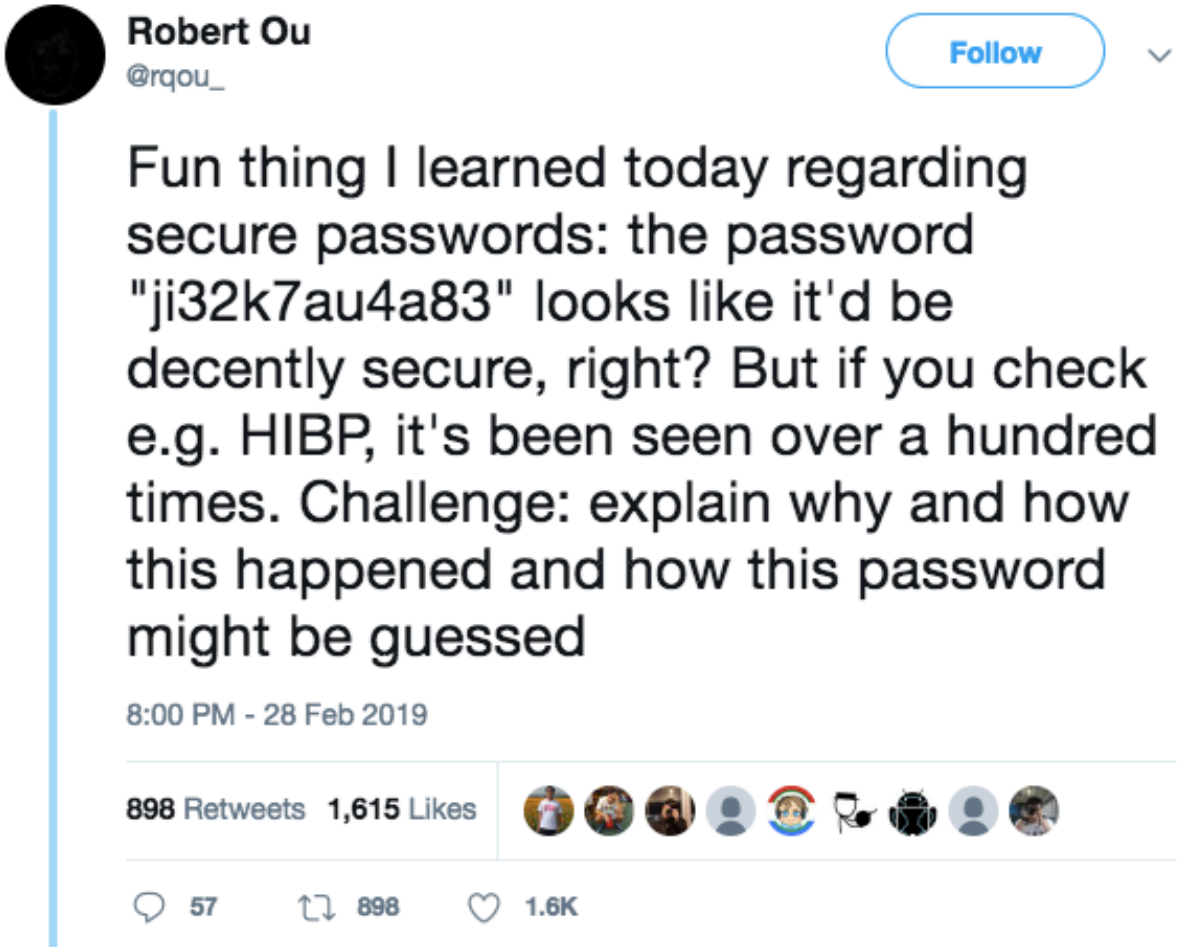
- R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation Onion Router," in USENIX Security, 2004.
- D. Fifield, C. Lan, R. Hynes, P. Wegmann, and V. Paxson, "Blocking-resistant communication through domain fronting," in Privacy Enhancing Technologies, 2015.

Text only, one page

What now? Applying Cryptography to Secure Network Protocols



We also need: Entity authentication, anonymous communication, DDoS defense



https://haveibeenpwned.com/Passwords

List of 2018's "Worst Password Offenders"

4. **Nutella**: Nutella came under fire for giving some of the *nuttiest* password advice of the year as the beloved hazelnut-and-chocolate spread company encouraged its Twitter followers to use "Nutella" as their password. As if the advice wasn't bad enough, the company sent out the ill-advised tweet to celebrate World Password Day.

https://blog.dashlane.com/password-offenders-2018/

Discussion: Reading critique #5

A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang. “The Tangled Web of Password Reuse,” in NDSS, 2014.

K. C. Wang and M. K. Reiter, “How to End Password Reuse on the Web,” in NDSS, 2019.

M. Abadi and R. Needham, "Prudent engineering practice for cryptographic protocols," in IEEE Transactions on Software Engineering, vol. 22, no. 1, pp. 6-15, Jan. 1996.

Agenda

Entity authentication

- Overview
- How to authenticate a human user
- Entity authentication protocols

Common flaws in protocol design

Reference: Ch. 10 in *Handbook of applied cryptography*

Entity authentication

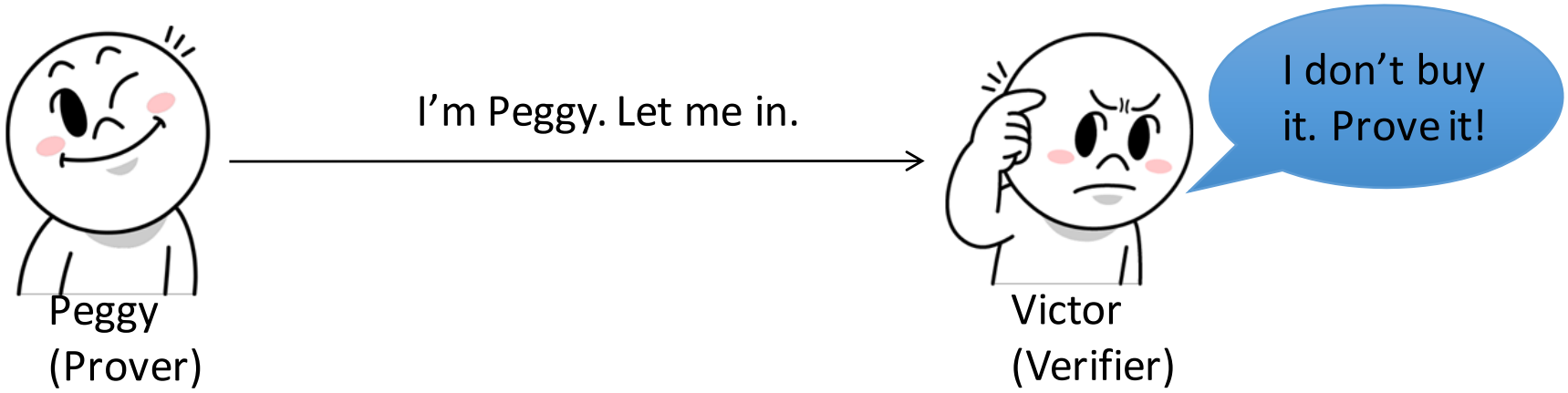
Verifier checks the claimed identity of *prover*

Prevents impersonation

Required for access control, user accountability, etc.

Entities can be people, machines, programs, etc.

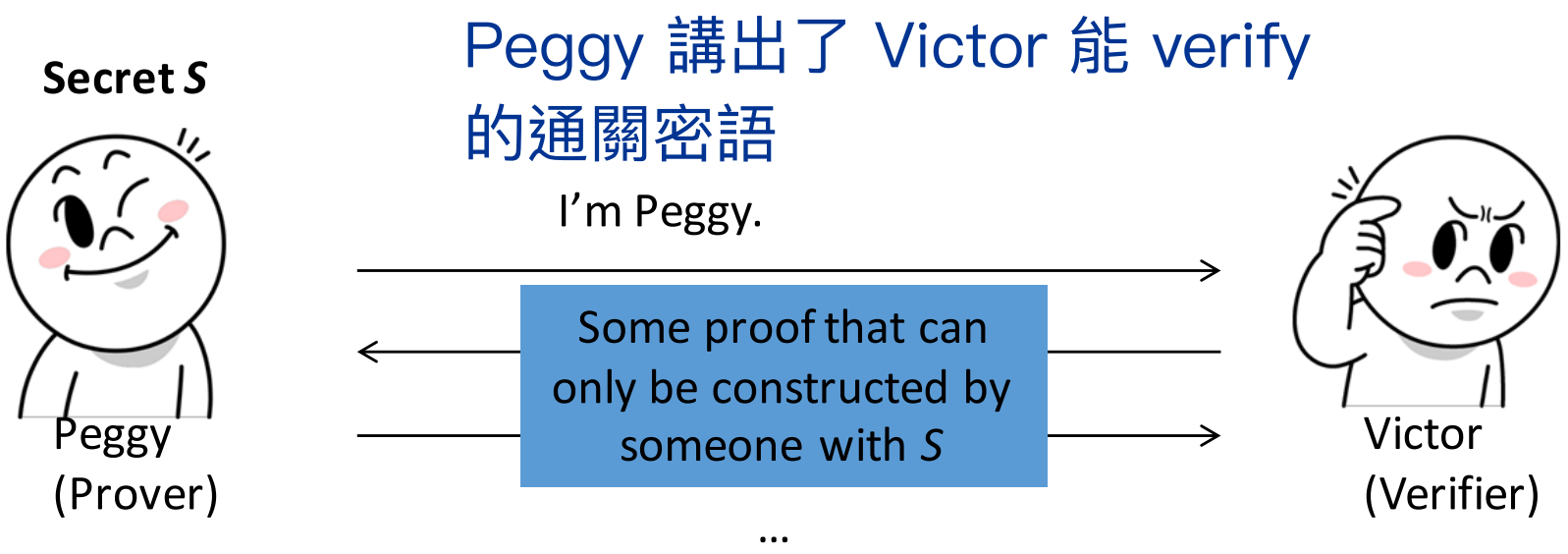
- V->P: Owner -> car; account holder -> ATM; user -> website login
- Authenticating people is harder. (why?)



Entity authentication

Usually the problem becomes that Peggy needs to prove that she possesses a *secret*, *S* (e.g., a password)

Proof can be interactive



Entity authentication vs. message authentication

Entity authentication

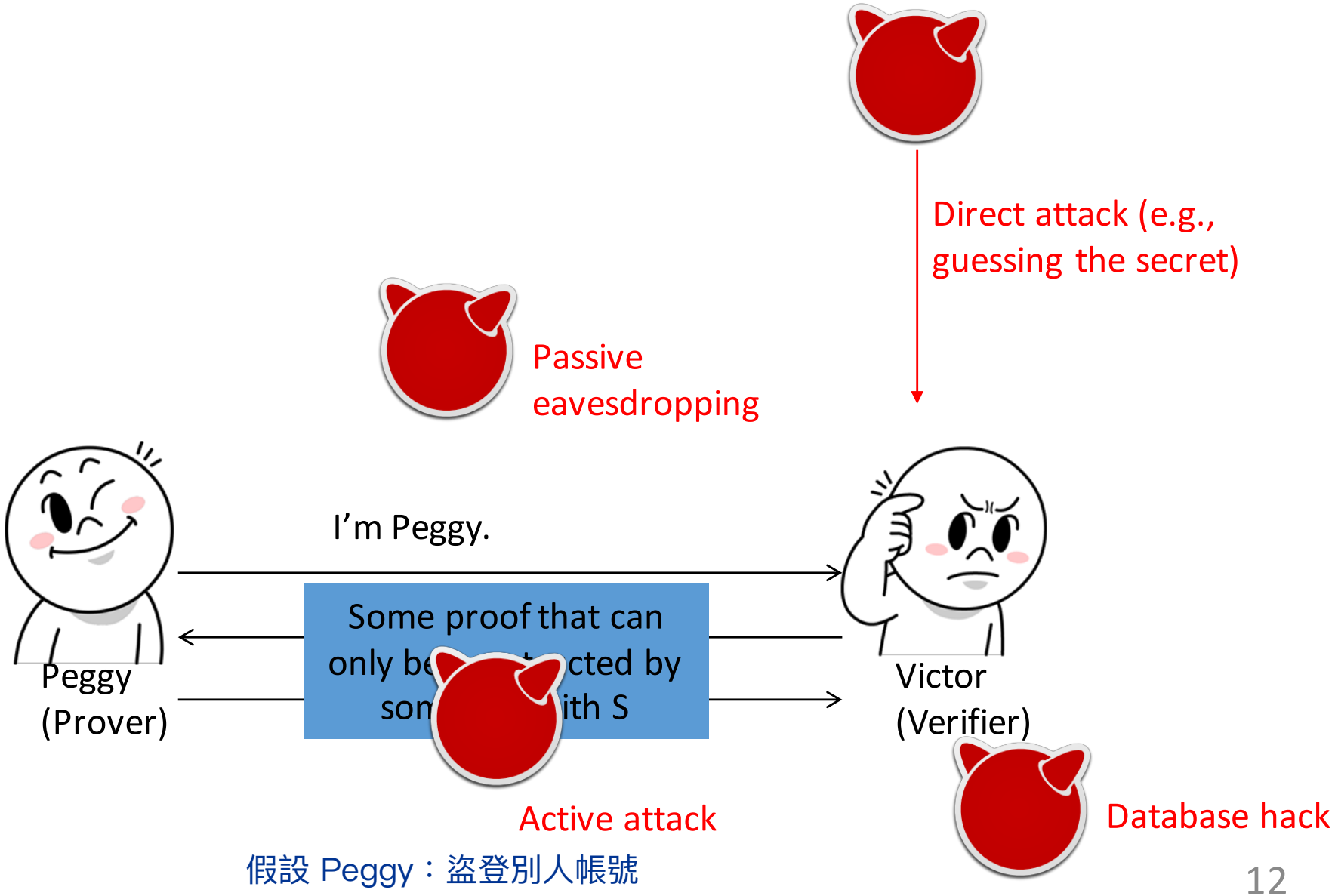
- Authenticates an entity in a communication session
- Ensures that the prover is online and participating in the communication (freshness)

freshness：在線上

Message authentication

- Authenticates a message *m*
- Ensures *m* is originated from the intended sender
- Implies *m* is not modified
- The sender can be offline

Threat model



假設 Peggy：盜登別人帳號

假設 Victor：把別人丟過來的帳號盜走

We will discuss the following questions

What should be the prover's secret, S, especially when the prover is a human user?

- How to deal with direct attack
- How to deal with database leak

What should be the protocol (interaction between the prover and the verifier)?

- How to defend against eavesdropping
- How to defend against active attack

Review: Password hashing

Why hashing passwords?

Threat model: leaked password database

If passwords are stored in their hash forms (rather than in plaintext), then it's harder for the the attacker to recover the passwords in case of data leak.

Hack of Cupid Media dating website exposes 42 million *plaintext* passwords

Sony hacked yet again, plaintext passwords, e-mails, DOB posted

More passwords: https://github.com/danielmiessler/SecLists/tree/master/Passwords

A. Juels and R. L. Rivest, "Honeywords: Making Password-Cracking Detectable," in *Proceedings of ACM CCS*, 2013.

Review: Salting and stretching

Store $\{\text{salt}, H^t(salt, P)\}$ instead of $H(P)$

- H^t means the hash function is applied t times
- t is at the order of 1000
- Why does these help?

Salting: adding random values such that two accounts with the same password generate different hashes

- Prevent the use of pre-computed tables

Stretching: deliberately slow down password cracking

Password hashing functions: PBKDF2, scrypt, argon2

How to Authenticate People

How to prove your identity

一覺醒來，穿越到20年後了…
要如何證明身份跟親朋好友相認？

胎記/痣的相認: 我就是這個人!
信物相認: 只有我有的東西
暗號相認: 只有我知道的東西

Common factors of entity authentication

Something you have: smart card, mobile phone, USB token, car controller

Something you know: password, PIN, unlock pattern

Something you are (biometrics): fingerprint, iris, vein, DNA, voice, gait

Something you have

Advantages? 無法 brute force, dict atk

Disadvantages? 無法驗證使用者是誰 –> 別人撿走就 g

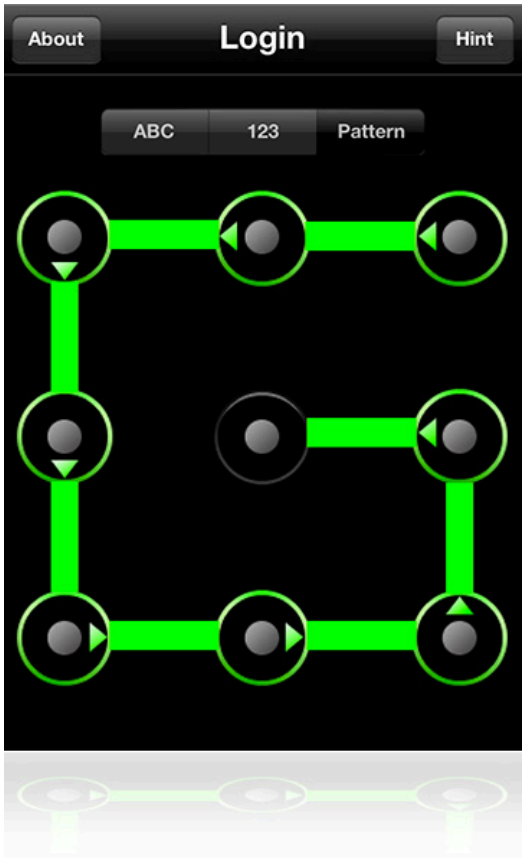
Something you know

Advantages? 認證不用帶著走，方便

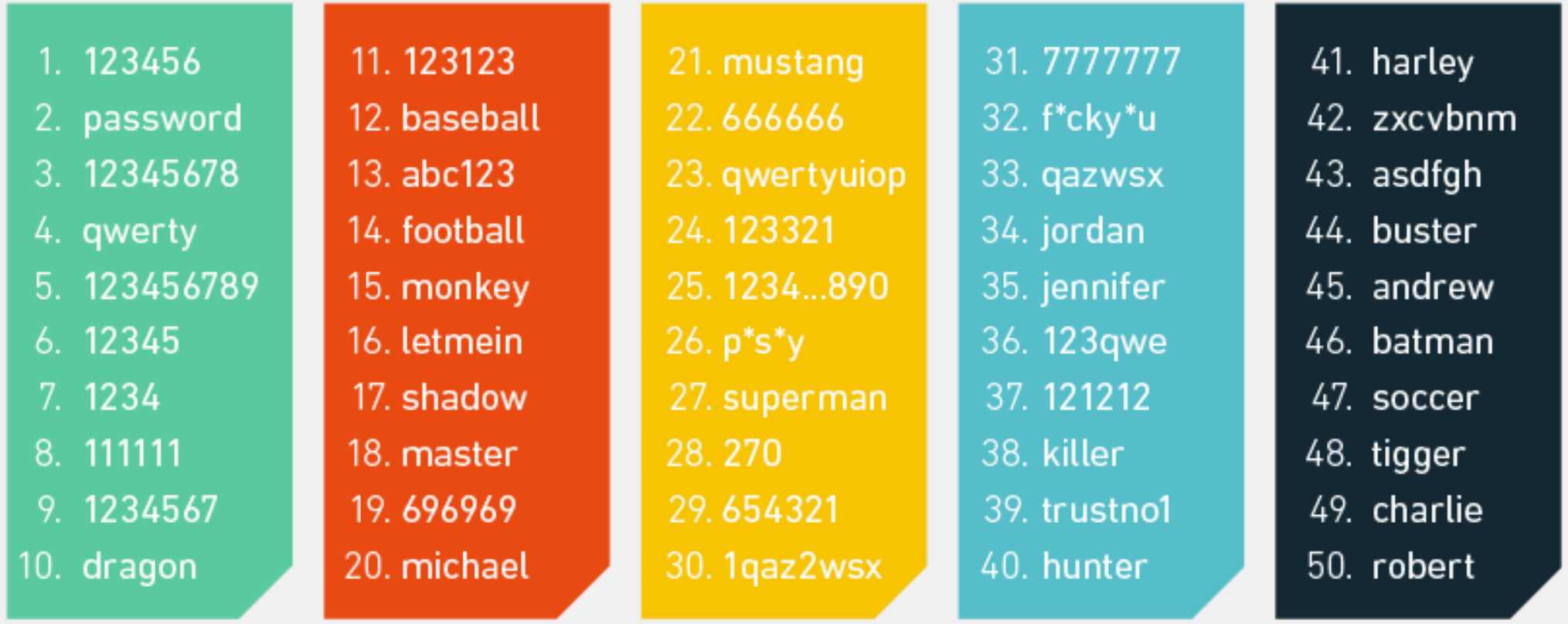
可能被旁邊的人偷看，像 key 偷看就沒用

Disadvantages? 手上的油，熱感壓

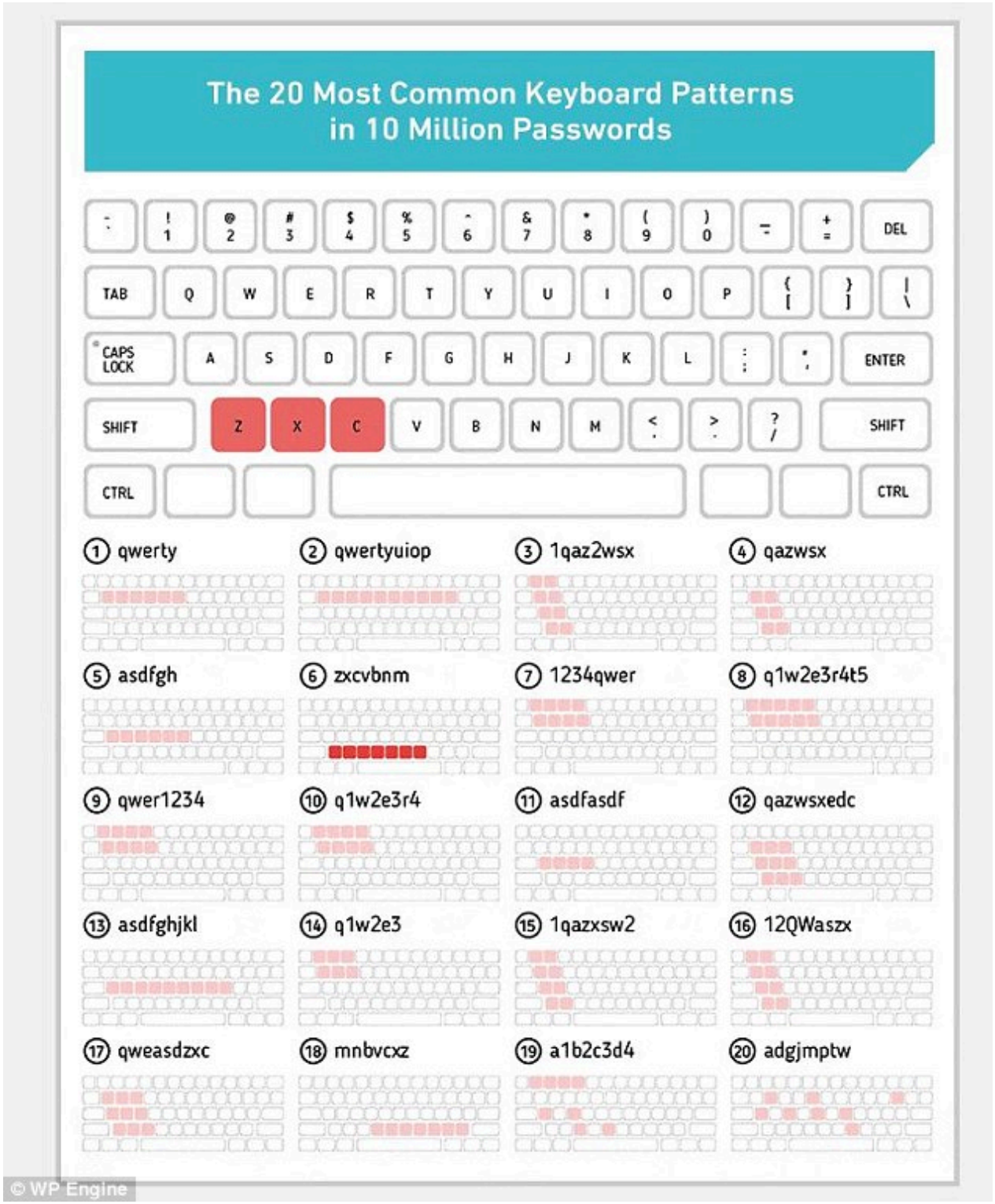
就是能被破掉



The 50 Most Used Passwords



https://wpengine.com/unmasked/



http://wpengine.com/unmasked/

課堂小調查

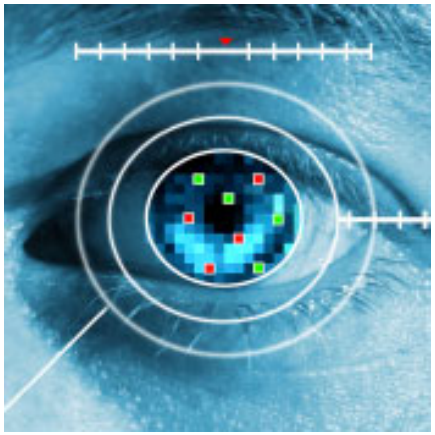
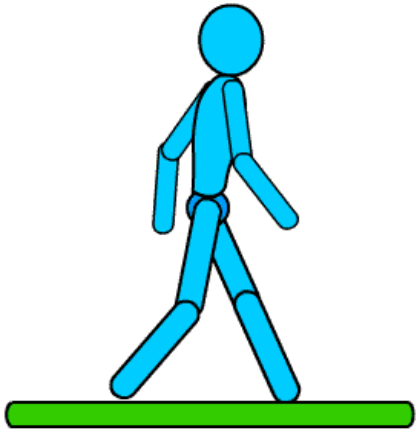
密碼在上一頁投影片上面的
寫在便條紙上貼在書桌前面
重複使用同一個密碼的
密碼根據網站不同加上些微小變化
使用多個密碼，分別給不同安全等級的網站
重要的網站都使用完全不同的密碼
每個網站都使用完全不同的密碼

其他特別的做法？
拒絕回答以上任何問題的

Something you are (biometrics)

Advantages? 方便，生物特徵本來就在身上

Disadvantages? 跟 key 這種物品很像的是，你可能被錄音，指紋可能被拿取


這邊有一批好便宜的指紋

http://www.theregister.co.uk/2014/12/29/german_minister_fingered_as_hackers_steal_her_thumbprint_from_a_photo/

User Authentication on the Web

Password manager

Single-Sign-On (SSO)

Chip Authentication Program (CAP)

Two-factor authentication

說了這麼多，我們還是一直在用不安全又難用的passwords，why!?

Bonneau, Joseph, et al. "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes." IEEE S&P, 2012.

- A framework helps evaluation, not absolute scoring, as metrics are not all of equal weight
- Metrics: usability-deployability-security
- Password-based authentication is easy to deploy
- No clear winner, whether a scheme is better than another depends on the application and attacker model

usability
developability
security

Why usability?

"Many real attacks exploit psychology at least as much as technology" – Ross Anderson

Usability problem: users will circumvent security mechanisms if they get in the way

"Crypto is bypassed, not penetrated" – Adi Shamir

人是資安最脆弱的環節

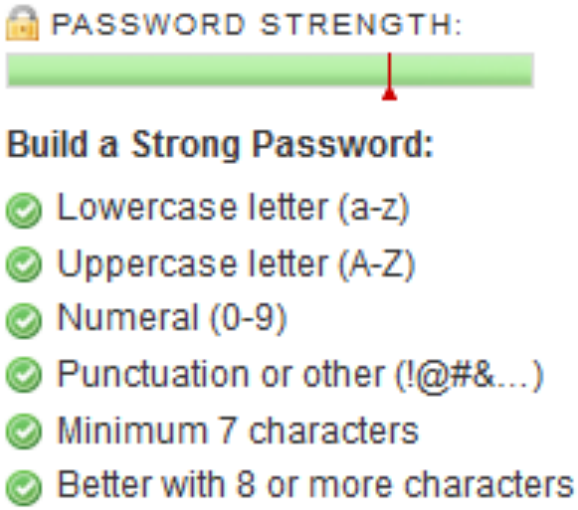
Password's usability problem

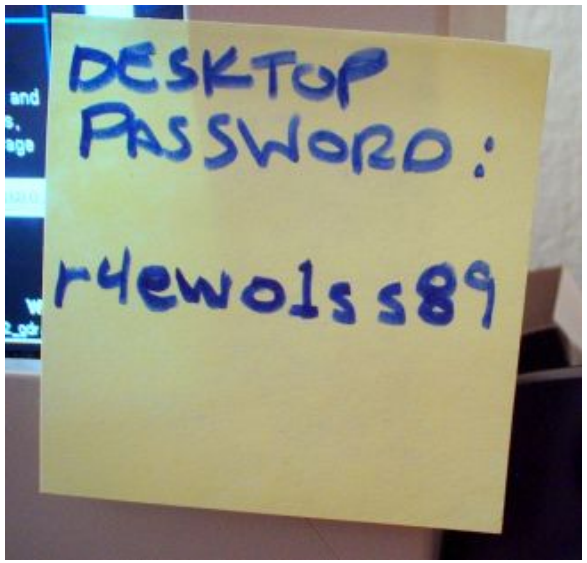
Easy to guess by attackers with the help of computers and/or knowledge of the victims

Hard to remember by human users

Against human's cognitive model!

"average employee using LastPass is managing 191 passwords"[1]





[1] https://blog.lastpass.com/2017/11/lastpass-reveals-8-truths-about-passwords-in-the-new-password-expose.html/

Usability

Memorywise-Effortless

Scalable-for-Users

Nothing-to-Carry

Physically-Effortless

Easy-to-Learn

Efficient-to-Use

Infrequent-Errors

Easy-Recovery-from-Loss

Deployability

Accessible

Negligible-Cost-per-User

Server-Compatible

Browser-Compatible

Mature

Non-Proprietary

Security

Resilient-to-Physical-Observation

Resilient-to-Targeted-Impersonation

Resilient-to-Throttled-Guessing

Resilient-to-Unthrottled-Guessing

Resilient-to-Internal-Observation

Resilient-to-Leaks-from-Other-Verifiers

Resilient-to-Phishing

Resilient-to-Theft

No-Trusted-Third-Party

Requiring-Explicit-Consent

Unlinkable

Baseline: evaluating legacy passwords

Usability	
Memorywise-Effortless	
Scalable-for-Users	
Nothing-to-Carry	★
Physically-Effortless	
Easy-to-Learn	★
Efficient-to-Use	★
Infrequent-Errors	O
Easy-Recovery-from-Loss	★

Deployability	
Accessible	★
Negligible-Cost-per-User	★
Server-Compatible	★
Browser-Compatible	★
Mature	★
Non-Proprietary	★

Security	
Resilient-to-Physical-Observation	
Resilient-to-Targeted-Impersonation	O
Resilient-to-Throttled-Guessing	
Resilient-to-Unthrottled-Guessing	
Resilient-to-Internal-Observation	
Resilient-to-Leaks-from-Other-Verifiers	
Resilient-to-Phishing	
Resilient-to-Theft	★
No-Trusted-Third-Party	★
Requiring-Explicit-Consent	★
Unlinkable	★

Baseline: evaluating legacy passwords

How about other schemes?

No scheme dominates passwords

Some are apparent worse than passwords

Graphical and cognitive schemes are only slightly better than passwords in security

Password managers and **Single Sign-On** are promising solutions that substantial improvements in both usability and security

CAP reader is the best in terms of security

Password managers

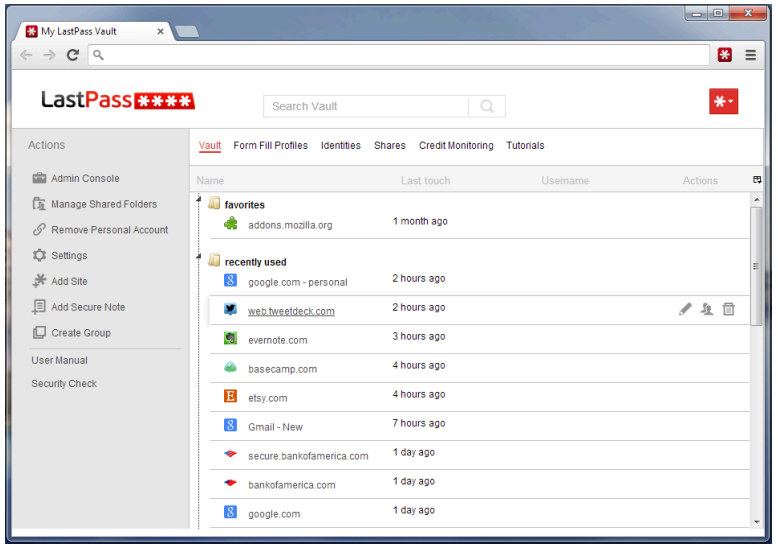
Help store, generate, fill passwords

Supported by modern browsers

3rd party tools: 1Password, LastPass

May be local or sync to a cloud

What are the assumptions and the threat model?



D. Silver, S. Jana, E. Chen, C. Jackson, and D. Boneh, "Password managers: Attacks and defenses," in Proceedings of USENIX Security, 2014.

Single-Sign-On (SSO)

Log in different services with a single ID and password

What are the assumptions and the threat model?

Y. Zhou and D. Evans, "SSOScan: Automated Testing of Web Applications for Single Sign-On Vulnerabilities," in Proceedings of USENIX Security, 2014.

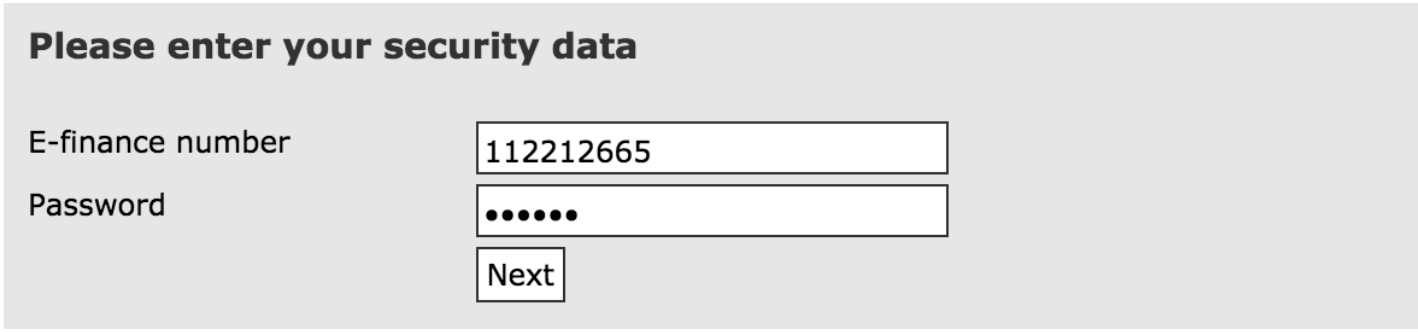
Chip Authentication Program (CAP) reader

Commonly used for banks

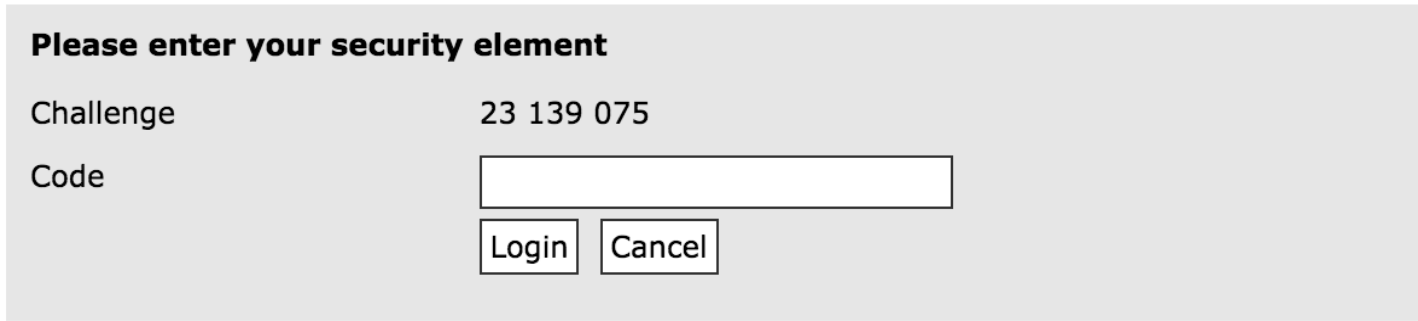
加密小精靈

Example: PostFinance e-login (Password + CAP reader)

1. Login the banking site as usual



2. Insert card, type “Challenge”, type PIN, and then get “Code” from the CAP reader
3. Type “Code” on the banking site



To login, you need to know/have…

Secret:

Password

Card

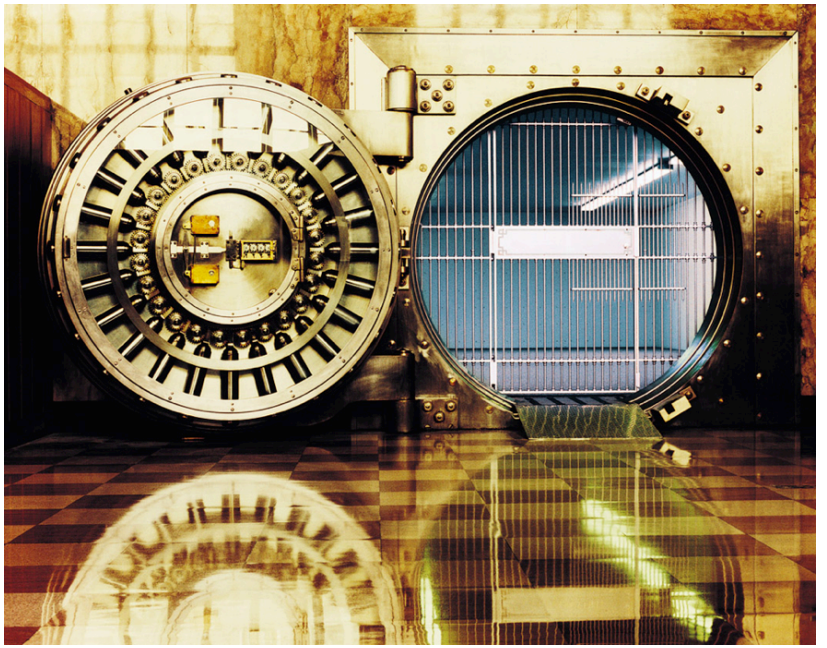
PIN

到底需要多少
東西才能**login**

Non-secret:

Card reader

Account number



Usability

Memorywise-Effortless

Scalable-for-Users

Nothing-to-Carry

Physically-Effortless

Easy-to-Learn ★

Efficient-to-Use ○

Infrequent-Errors ○

Easy-Recovery-from-Loss

Deployability

Accessible

Negligible-Cost-per-User

Server-Compatible

Browser-Compatible ★

Mature ★

Non-Proprietary

Security ★

Resilient-to-Physical-Observation ★

Resilient-to-Targeted-Impersonation ★

Resilient-to-Throttled-Guessing ★

Resilient-to-Unthrottled-Guessing ★

Resilient-to-Internal-Observation ★

Resilient-to-Leaks-from-Other-Verifiers ★

Resilient-to-Phishing ★

Resilient-to-Theft ★

No-Trusted-Third-Party ★

Requiring-Explicit-Consent ★

Unlinkable ★

Exercise: CAP reader (Do you agree?)

Security issues with CAP readers

The analytical result looks promising on paper, but…

CAP protocol is proprietary, lacking public scrutiny

- Remember Kerckhoffs's principle? Security through obscurity is bad.

Numerous design and implementation errors found in the UK version of CAP

- Drimer, Saar, Steven J. Murdoch, and Ross Anderson. "Optimised to fail: Card readers for online banking." Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2009.

None of these is perfect.
Let’s combine them for better security.

Multi-factor authentication

Combing two or more authentication factors

- E.g., card + PIN, password + code received from phone

Hope to boost up security

Example: Physical Token

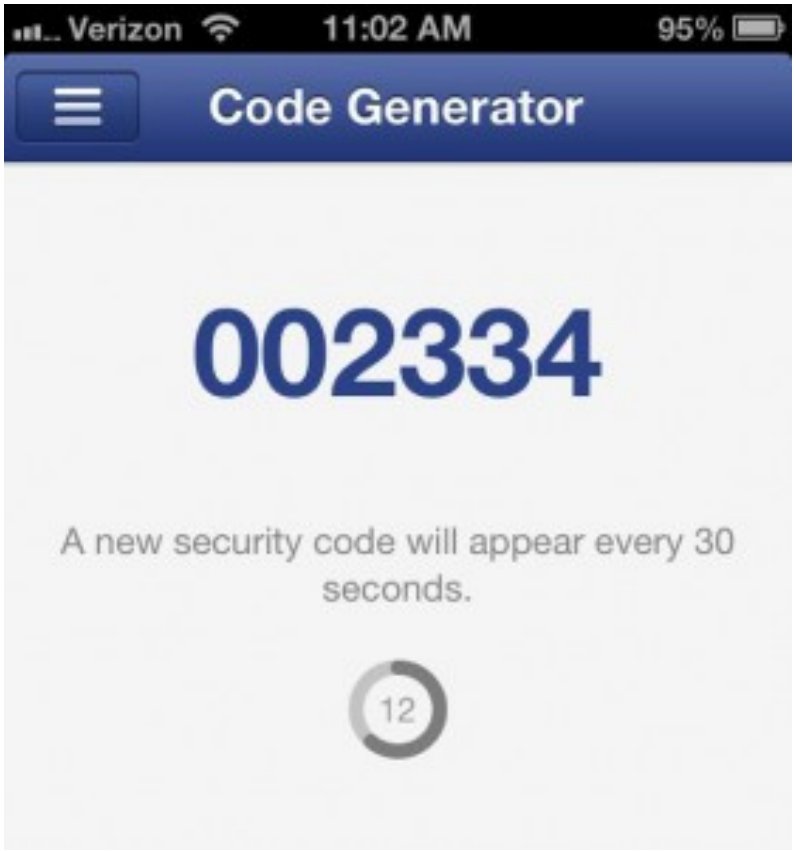
Physical token supporting 2-factor authentication

Yubikey

Google Titan Key

Example: Facebook Code Generator

2FA when logging into your Facebook account from a new computer or mobile phone



UK bank to trial fingerprint technology for card payments

PUBLISHED MON, MAR 11 2019 • 8:55 AM EDT | UPDATED MON, MAR 11 2019 • 10:17 AM EDT

https://www.cnbc.com/2019/03/11/uk-bank-to-trial-fingerprint-technology-for-card-payments.html

Multi-factor authentication

Multi factor 不見的比較好，因為如果一個被破了，其它也會跟著 g 掉

Ideally increase security because attacker now has to break multiple schemes at the same time

However, can it be 1+1<2?

- Researchers show that users pick much weaker passwords when two-factor authentication enabled
- Correlation between authentication factors
  - E.g., 密碼存在手機裡，壞人拿到手機可以收簡訊也拿到密碼
  - E.g., 手機跟電腦同步，手機中毒電腦也中毒

Two-Factor Authentication Using SMS is no longer recommended

In the Digital Authentication Guideline of U.S. National Institute for Standards and Technology (NIST):

> "Out-of-band authentication using the PSTN (SMS or voice) is discouraged and is being considered for removal in future editions of this guideline."

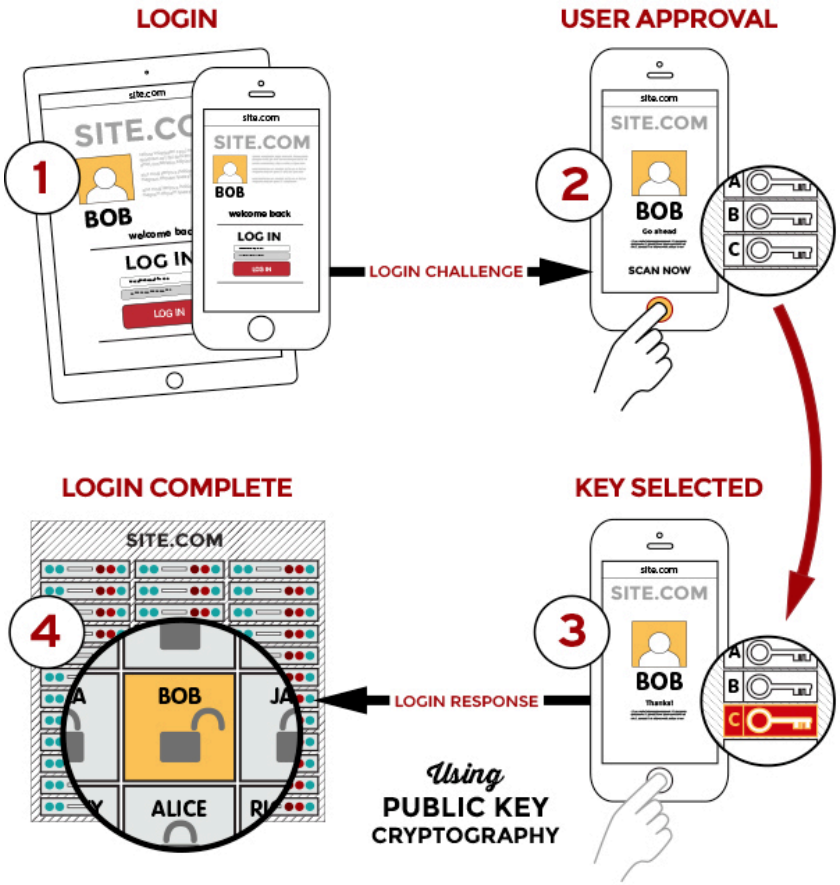
http://thehackernews.com/2016/07/two-factor-authentication.html

FIDO (Fast Identity Online)

UAF: universal authentication framework, supports passwordless auth

U2F: universal second factor

FIDO2: UAF, U2F, Client-to-Authenticator Protocols



https://fidoalliance.org/specifications/

Entity Authentication Protocols

We will discuss the following questions

What should be the prover's **secret**, S, especially when the prover is a human user?

- How to deal with direct attack
- How to deal with database leak

What should be the **protocol** (interaction between the prover and the verifier)?

- How to defend against eavesdropping
- How to defend against active attack

Common assumptions when analyzing protocols: the adversary cannot break cryptographic primitives and the implementation is correct.

Entity authentication protocols

Fixed passwords

One-time passwords

Challenge-response

Zero-knowledge proofs

Password Authenticated Key Exchange (PAKE)

Menezes, Alfred J., Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of applied cryptography*. CRC press, 1996. http://cacr.uwaterloo.ca/hac/ See Chapter 10.

Threat model: Eavesdropping

When a *fixed* password is used…

- Attacker can eavesdrop & subsequently replay the password
- E.g, keylogger, insecure communication, shoulder surfing…

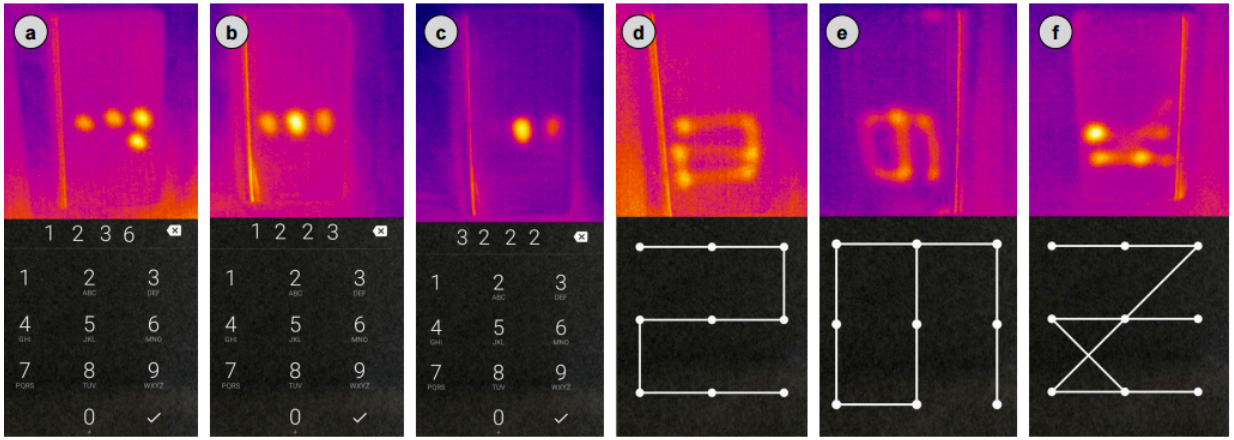


Figure 1: In this work we investigate thermal attacks against PINs and patterns on mobile devices. After entering PINs (a–c) or patterns (d–f) on a touch screen, a heat trace remains on the screen and can be made visible via thermal imaging.

shoulder surfing defense (?)

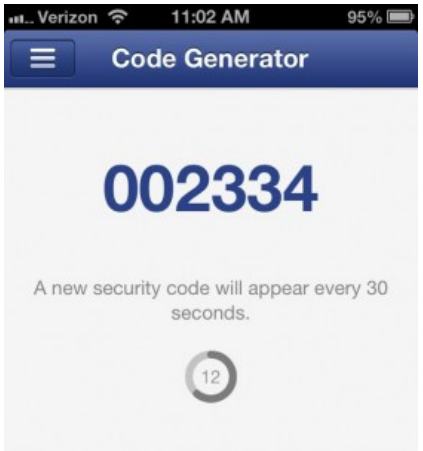
Abdelrahman, Yomna, et al. "Stay cool! understanding thermal attacks on mobile-based user authentication," CHI 2017.

One-time password can prevent eavesdropping

One-time password: password valid for only one session

- != one-time pad

Eavesdropping doesn't help anymore since the password can only be used once.



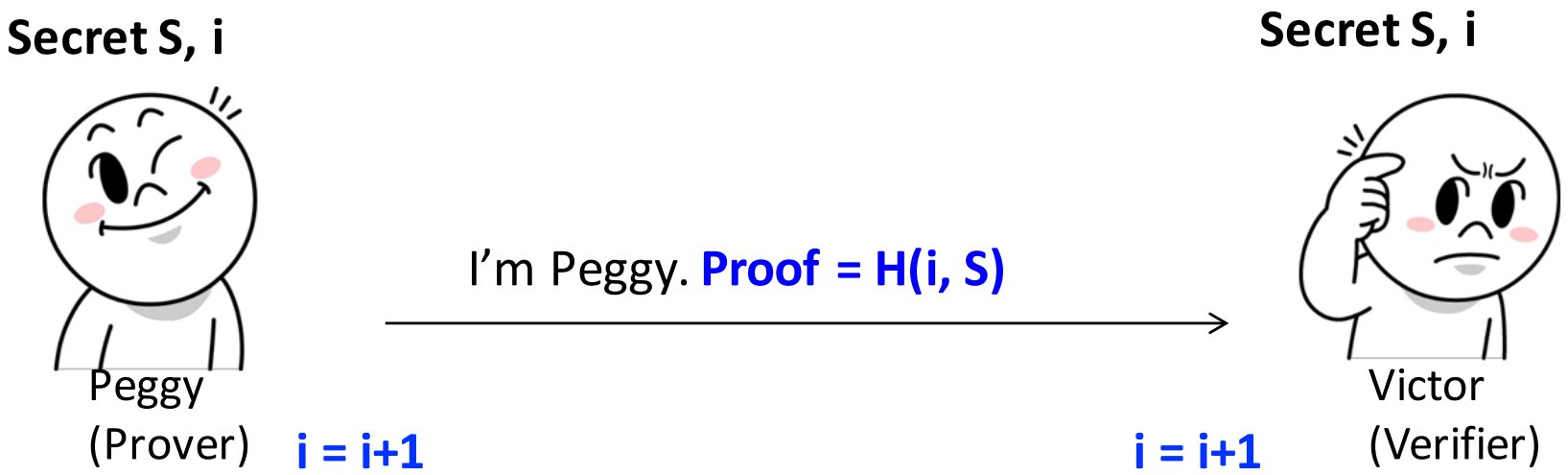
One-time password: Approach 1

每一次密碼都不一樣

Assume a shared secret and a synchronized state i

Can use *time* instead of *counter* i to avoid keeping state

Resilient-to-Physical-Observation



One-time password: Approach 2

Lamport's one-time password scheme

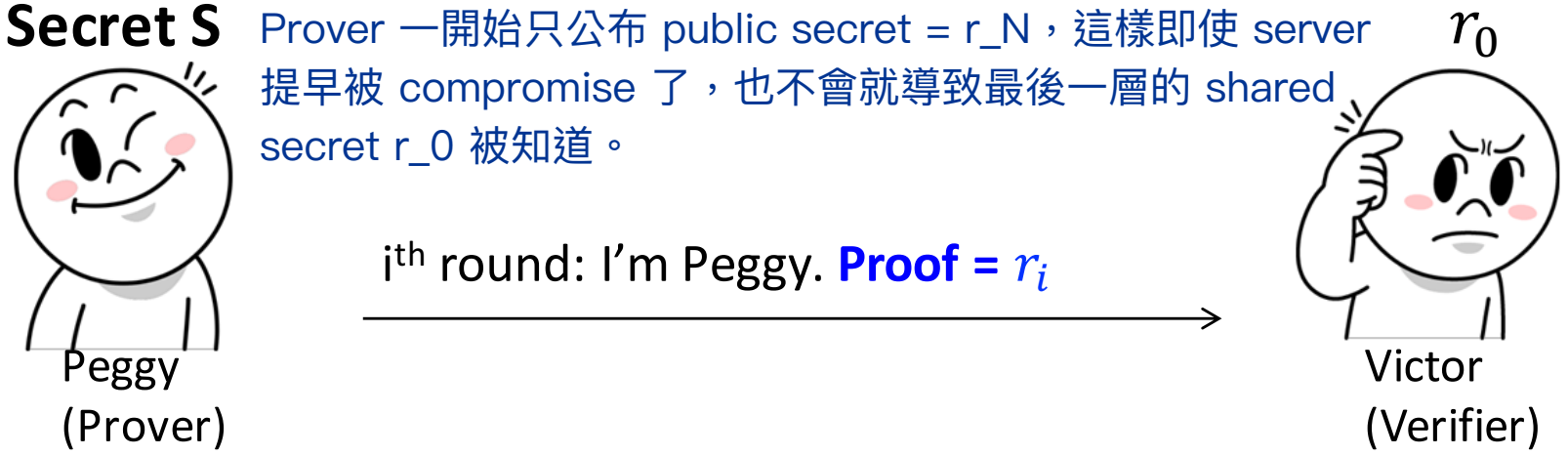
Verifier doesn't know S (secure against server compromise)

- Resilient-to-Physical-Observation & Resilient-to-Internal-Observation

Prover constructs a **one-way hash chain**

- $r_N = S$
- $r_i = F(r_{i+1})$
- Securely share r_0 with Verifier

How can Victor verify the proof, r_i?



Review: One-Way Hash Chains

Versatile cryptographic primitive

Construction

- Pick random r_N and public one-way function F
- $r_i = F(r_{i+1})$
- Secret value: r_N, public value r_0

Properties

- Use in reverse order of construction: $r_1, r_2, r_3, \ldots, r_N$
- Infeasible to derive r_i from r_j $(j < i)$
- Efficiently authenticate r_i using r_j $(j < i)$: $r_j = F^{i-j}(r_i)$
- Robust to missing values

One-time password: Approach 2

Robust to missing values: Victor can always check r_i using r_j $(j < i)$, because $r_j = F^{i-j}(r_i)$

Need to regenerate the chain after *N* uses

Victor 同時可在每 i 回合，去檢驗 Prover 給的 r_i 是否正確，藉由檢查下式是否成立

Victor verifies r_i by checking if $r_{i-1} = F(r_i)$

r_{i – 1} = F(r_i)

Secret S

Peggy
(Prover)

r_0

Victor
(Verifier)

i[th] round: I'm Peggy. **Proof = r_i**

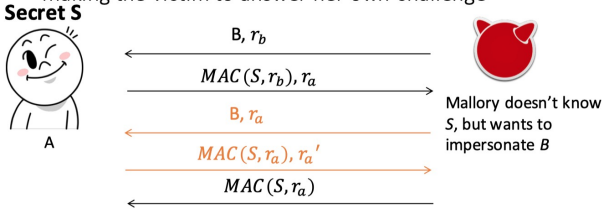
Threat model: Active attack

What if attacker can actively participate in the protocol?

The attacker can interact with the user, collect one-time passwords, and later try to impersonate the user.

We also need a way to prove freshness, preventing replay attacks

希望對方真的在線上



padding nonce 可以確保不會被 p.61 reflection atk，因為 reflection atk 用的機制是，Eve 想要仿冒自己是 Bob，但他沒有 shared secret S，於是在收到 Alice 傳回來的 MAC(S, r_b)、r_a 後，他沒辦法回傳 MAC(S, r_a)，但他可以再馬上將這個 r_a 送給 Alice，讓 Alice 再誤簽一次 MAC(S, r_a) 傳回來，這樣 Eve 就拿到了 MAC(S, r_a)，即使他不知道 S。

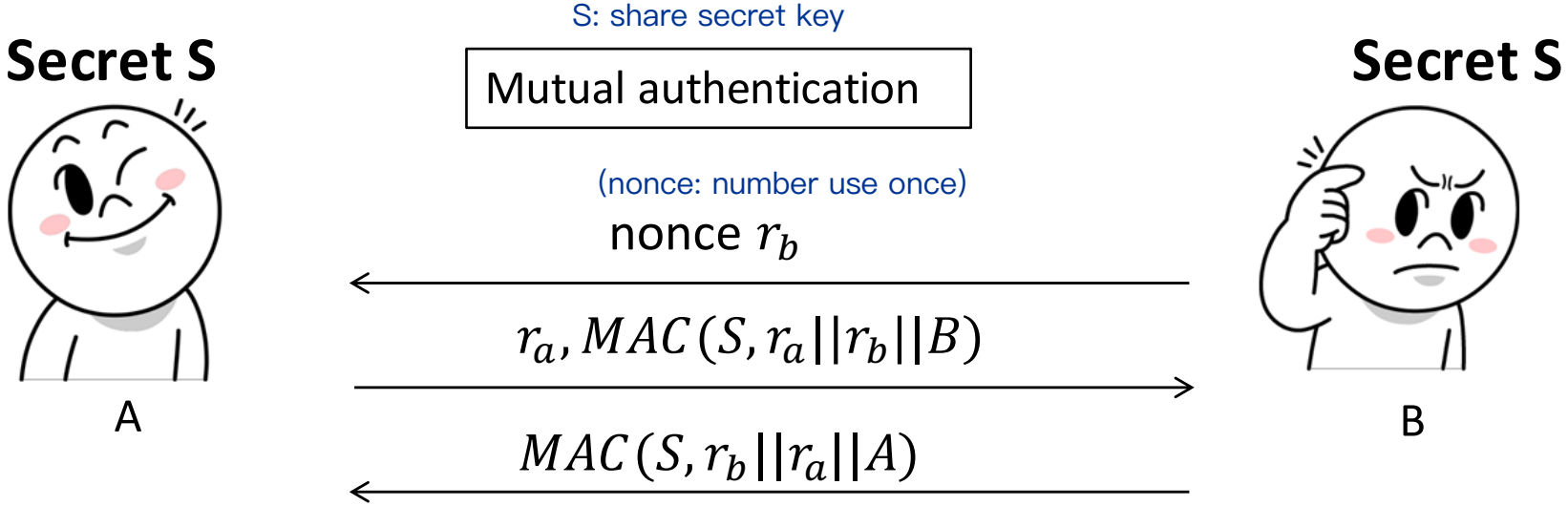
Challenge-response

Without sending the secret

Prevent active attacks

Show freshness of the proof using nonce

如果運用到了 padding nonce，那 Alice 送過來的 MAC(...)，裡面總是包含 B，並且 Bob 傳給 Alice 的 MAC(...)，裡面總是會 padding A，這樣即使 Eve 想要透過再傳一次剛才從 Alice 拿到的 r_a，讓 Alice 誤簽一次也沒用，因為此時 Alice 會回傳 MAC(S, r_a', r_a ‖ B)，但在此 protocol 下，Eve 若想騙過 Alice，他必需回傳的是 MAC(S, r_a, r_a' ‖ A)，因為無法拿 Alice 誤傳的 MAC 騙過去，因為那包 MAC 沒有 padding A！



1. B 送了一個 nonce r_b 給 A，有點像在說：「Alice，如果你是我朋友，就用剛才我送你的 nonce r_b，簽名後再傳回來給我吧！」–> 因為若 Alice 透過 shared secret S 簽該 nonce 的話，Bob 就能驗證 Alice 到底是不是「有 shared secret S」的那個 Alice。
2. A 回傳了，r_a、MAC(S, r_a ‖ r_b ‖ B) 給了 B，有點像在說：「Bob，我是你朋友，於是我用你剛才給我的 nonce r_b 簽了一份回傳給你，並且為了確保 freshness，我在後方 append 了一個 B，你如果也是我朋友，再拿剛才我送你的 nonce r_a，簽名後再傳回來給我吧！」
3. B 回傳 MAC(S, r_b ‖ r_a ‖ A) 給了 A，有點像在說：「Alice，我是你朋友，於是我用你剛才給我的 nonce r_a 簽了一份回傳給你。」

Proofs of freshness

Nonces

- NONCE = Number used only ONCE

Timestamps

- Accept message if timestamp is recent enough
- May need synchronized clocks

Alice and Bob check whether the nonce is never used before or the timestamp is recent

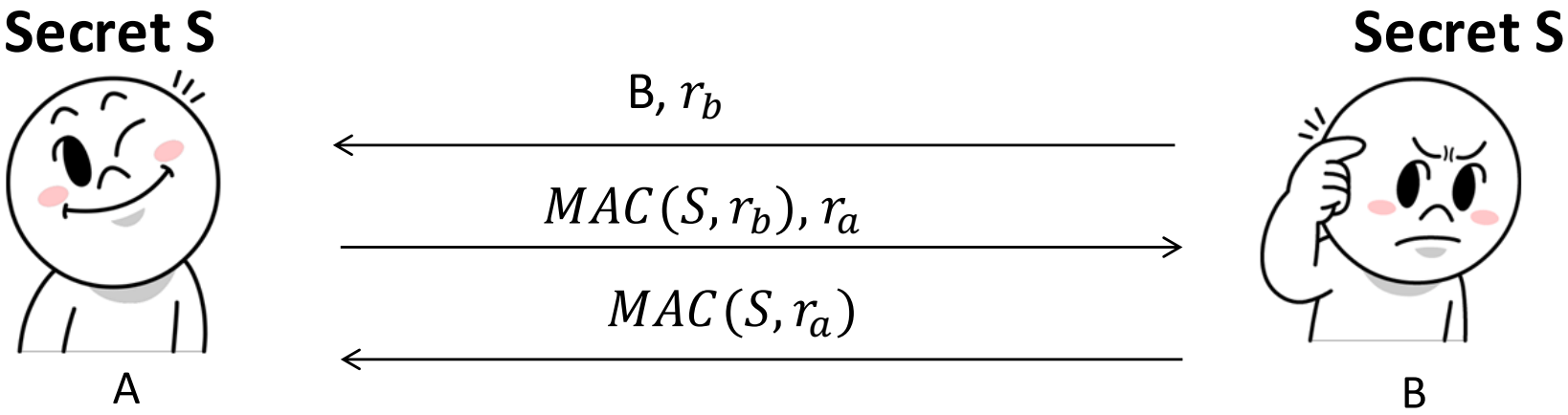
Pros and cons?

Pros: base on 時間，不用記憶
Cons 網路 delay

Exercise

Is this mutual authentication protocol secure against an active attacker?

- Can the attacker pretend to be Bob and fool Alice without knowing the secret S?

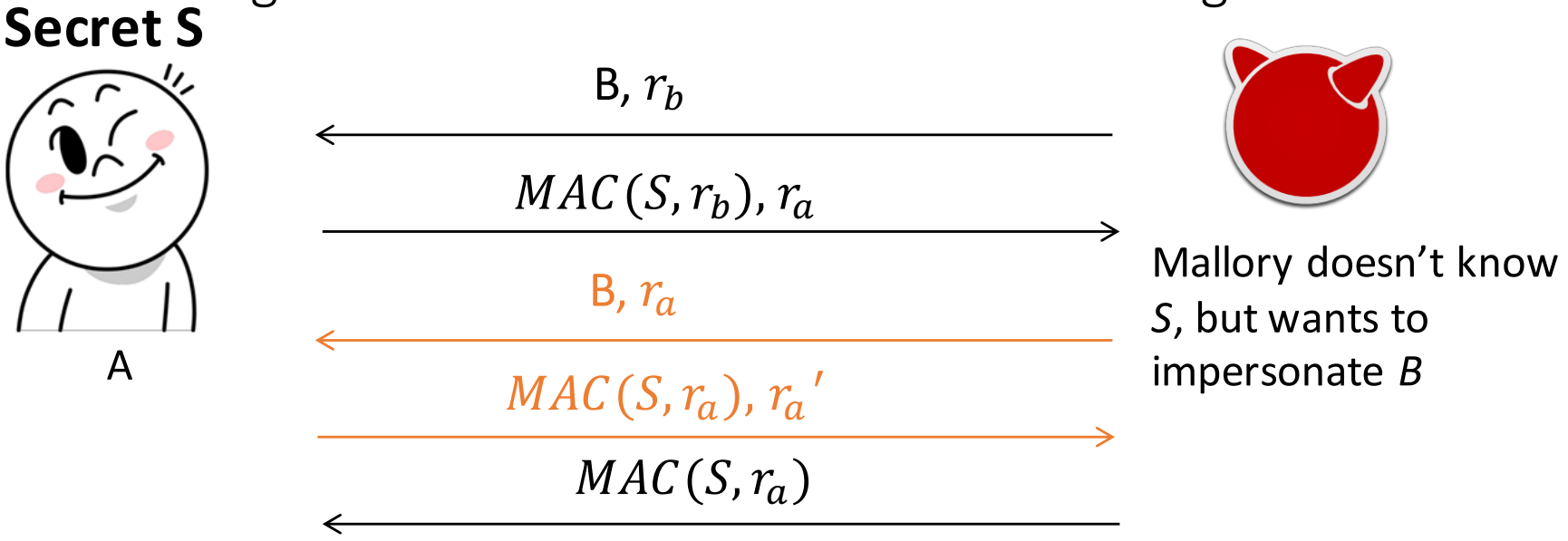


Exercise

Is this mutual authentication protocol secure against an active attacker?

Insecure, reflection attack is possible.

- Return the message to its originator
- Commonly used in breaking authentication protocols, making the victim to answer her own challenge



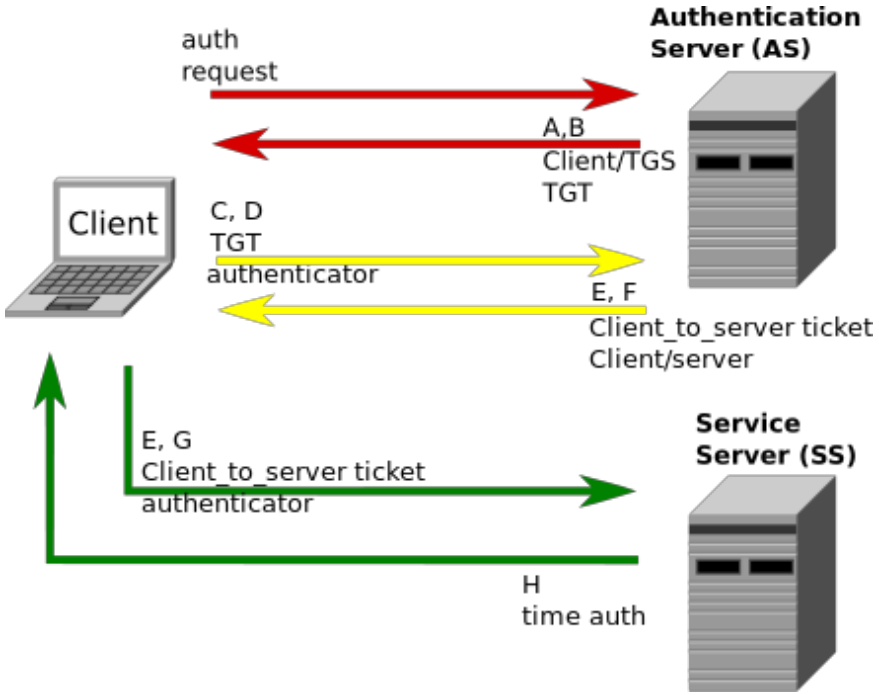
Kerberos Protocol

Maintaining pairwise keys is impractical; use a centralized **Key Distributed Center** in practice.

Widely used for entity authentication and key establishment

Uses symmetric cryptography only and a trusted third party

"ticket" includes timestamp to limit the lifetime



Exercise: Needham-Schroeder protocol

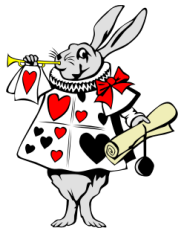
Basis of Kerberos

A historically famous protocol but no longer recommended

settings，旭君沒先講…囧

N_A and N_B are nonces; K_A and K_B are Alice's and Bob's shared keys with KDC, respectively

symmetric key K_{AB}



(1) A, B, N_A

(2) KDC picks a random K_{AB} for Alice and Bob

(3) $E(K_A, N_A||B||K_{AB}||E(K_B, K_{AB}||A))$

(4) Alice gets K_{AB}

(5) $E(K_B, K_{AB}||A)$

(6) Bob gets K_{AB}

(7) $E(K_{AB}, N_B)$

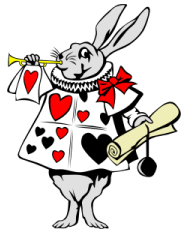
(8) $E(K_{AB}, N_B - 1)$

Exercise: Needham-Schroeder protocol

Security issues?

- Can the attacker pretend to be Alice and fool Bob if the session key K_{AB} is leaked at some point?

若 K_{AB} 外洩了，Eve 就能假冒 Alice 跟 Bob 溝通



(1) A, B, N_A

(2) KDC picks a random K_{AB} for Alice and Bob

KDC 只有跟 Alice 溝通，沒有跟 Bob，Bob 無法確定 K_{AB} 是不是真的從 Alice 收到的，還是其實是 Eve 假冒的

(3) $E\left(K_A, N_A||B||K_{AB}||E(K_B, K_{AB}||A)\right)$

(4) Alice gets K_{AB}

(5) $E(K_B, K_{AB}||A)$

(6) Bob gets K_{AB}

(7) $E(K_{AB}, N_B)$

(8) $E(K_{AB}, N_B - 1)$

Zero-knowledge proof

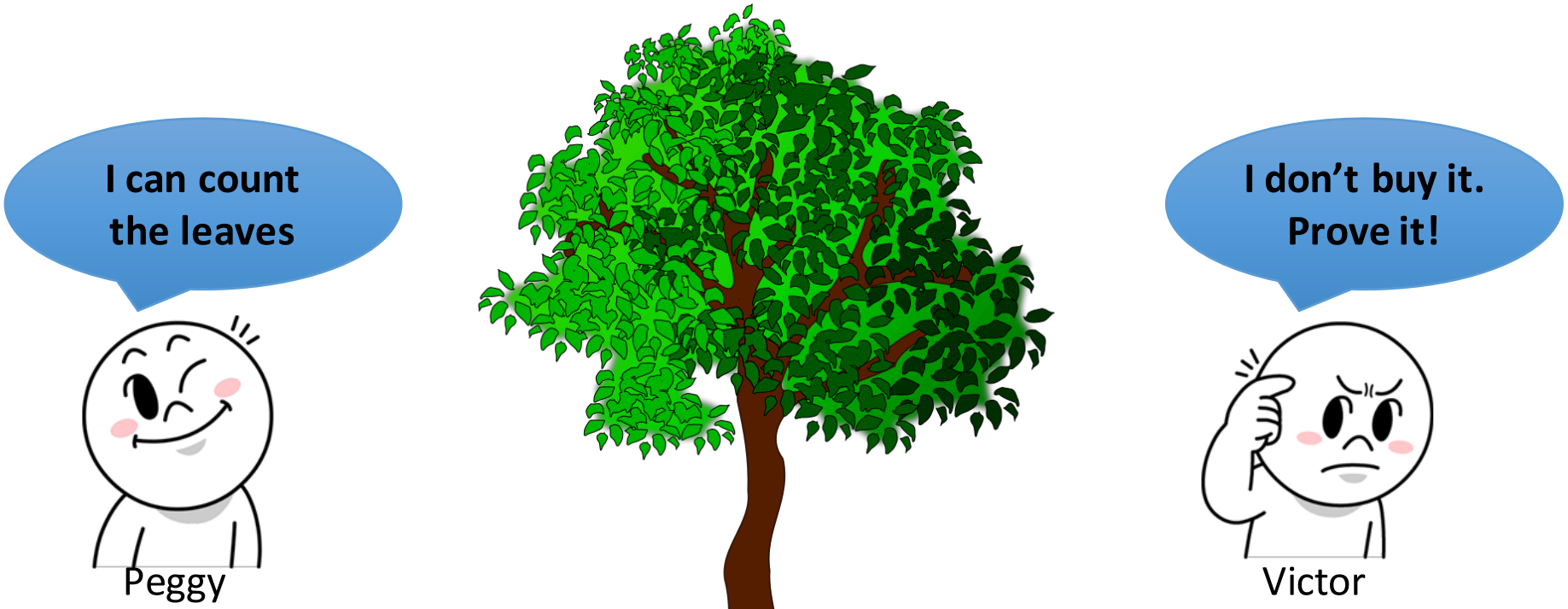
All the above protocols leak some info about the secret to the prover

Can the prover demonstrate her **knowledge of a secret** w/o revealing any information (except 1 bit)?

e.g. 使用者要登入某網站，使用者為 prover，而網站為 verifier，prover 可以不透過直接輸入密碼，就讓 verifier 認為擁有密碼？

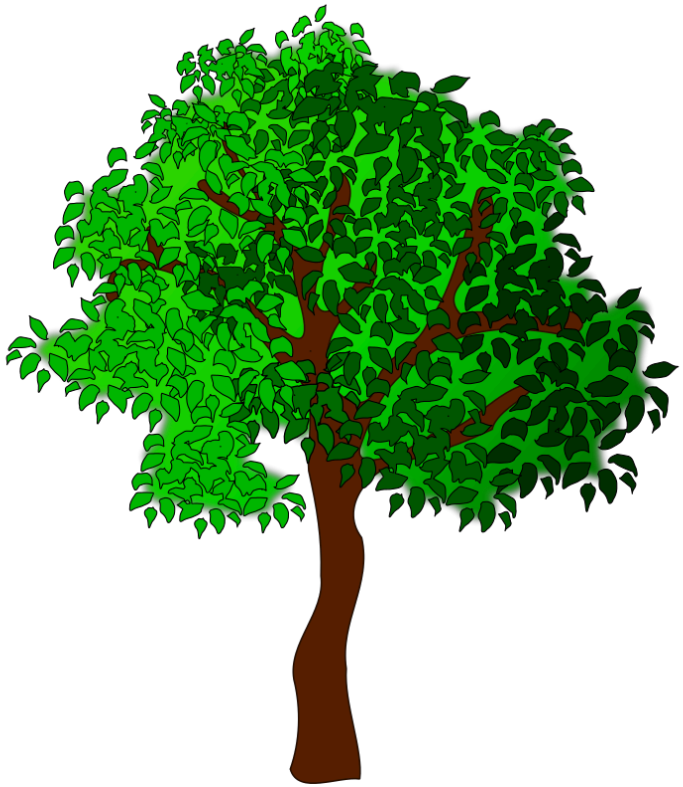
Zero-knowledge proof

How Can Peggy prove her capability without revealing how she did it?



Zero-knowledge proof

1. There are *n* leaves



2. ok, turn around. I will either remove one leaf or do nothing.

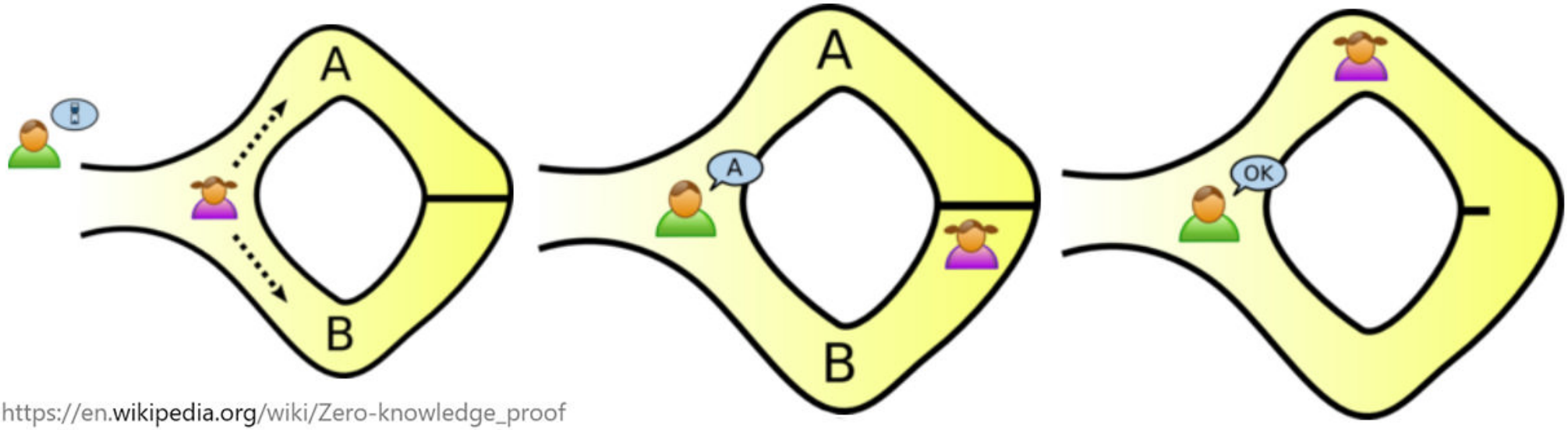
3. Challenge: how many leaves are there now?

4. Response:

Peggy

Victor

Another example of ZKP



https://en.wikipedia.org/wiki/Zero-knowledge_proof

Password-Authenticated Key Exchange (PAKE)

A class of cryptographic key exchange protocols in which the client uses her password (i.e., a short secret) for authentication

- Prevent MitM
- Can be applied for entity authentication without leaking the password

https://blog.cryptographyengineering.com/2018/10/19/lets-talk-about-pake/

Password-Authenticated Key Exchange (PAKE)

Setting

- Alice and Bob know a password P
- Password: a **short** shared secret

Goal: establish shared key based on P

PACK: Attempt 1

Attempt 1

- $K = H(P)$
- Use K to encrypt / authenticate communication
- A → B: $E_K(m_A)$
- B → A: $E_K(m_B)$
- What's wrong with this approach?

Attack: An eavesdropper can perform a dictionary attack to guess password

Dictionary vs. brute force

Dictionary Attack

Trying apple : failed
Trying blueberry : failed
Trying justinbeiber : failed
…
Trying letmein : failed
Trying s3cr3t : success!

Brute Force Attack

Trying aaaa : failed
Trying aaab : failed
Trying aaac : failed
…
Trying acdb : failed
Trying acdc : success!

https://crackstation.net/hashing-security.htm

PACK: Attempt 2

Same setting as before

Attempt 2

- $K = H(P)$
- Alice picks K' at random
- A → B: $E_K(K')$
- B → A: $E_{K'}(\text{“}Message\ Header\text{: …”})$

Dictionary attack possible?

- Yes! Pick candidate password P
- Compute K, decrypt K’, and verify that message matches “Message Header:”

PACK: EKE DH Protocol

DH: 還記得Diffie-Hellman key agreement嗎？

EKE: Encrypted Key Exchange

Setting

- Alice randomly picks a secret $\mathbf{a}$ and a nonce N_A
- Bob randomly picks a secret $\mathbf{b}$ and a nonce N_b

PACK: EKE DH Protocol

1. $K = H(P)$ 就算猜對了，也不知道自己猜對了
2. A → B: $E_K(g^a)$
3. $K' = H(g^{ab})$
   B → A: $E_K(g^b), E_{K'}(N_B)$
4. A → B: $E_{K'}(N_A, N_B)$
5. B → A: $E_{K'}(N_A)$

Dictionary attacks? Man-in-the-middle attacks? Replay attacks?

Common Flaws in Protocol Design

Man-in-the-middle attack

Replay attack

Reflection attack

Attack 1: Man in the middle (MitM)

MitM attack intercepts communication between two parties who believe they are directly talking to each other

Example: MitM attack against DH key agreement

- Eve impersonates Alice to Bob and Bob to Alice

$g^{am} \bmod p \quad \neq \quad g^{bm} \bmod p$

Mallory can decrypt the comm. between Alice and Bob

Attack 1: Man in the middle (MitM)

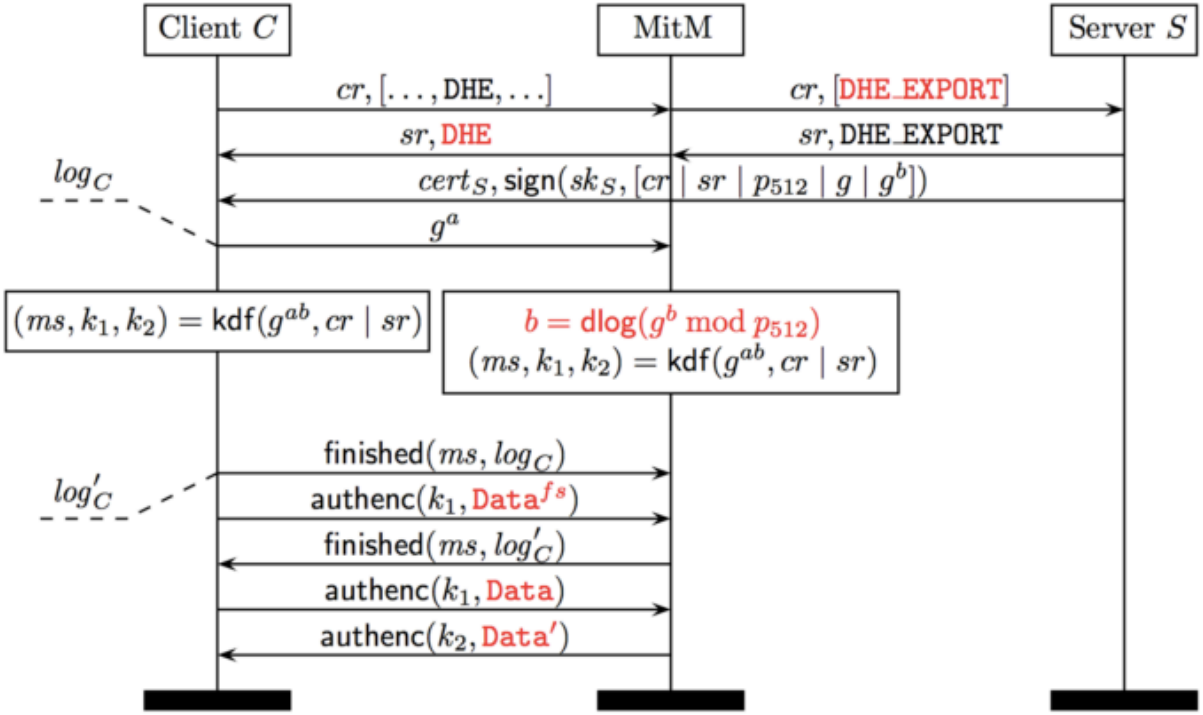


Figure 2: **The Logjam attack.** A man-in-the-middle can force TLS clients to use export-strength DH with any server that allows DHE_EXPORT. Then, by finding the 512-bit discrete log, the attacker can learn the session key and arbitrarily read or modify the contents. $\texttt{Data}^{fs}$ refers to False Start [30] application data that some TLS clients send before receiving the server's Finished message.

D. Adrian *et al.*, "Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice," *ACM CCS*, 2015.

Attack 1: Man in the middle (MitM)

The Password Reset MitM Attack

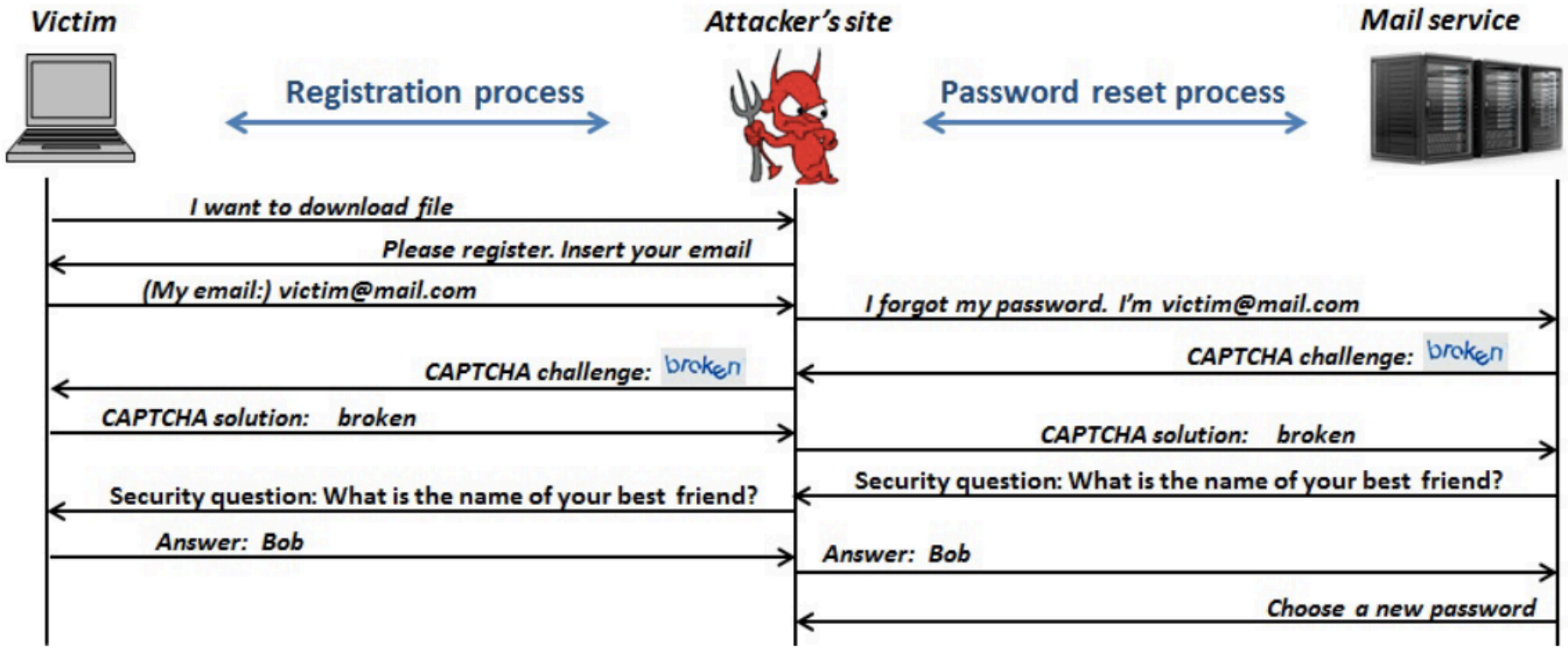


Fig. 1: Basic PRMitM attack illustration. In this example, the email service provider challenges the attacker with a CATPCHA and a security question.

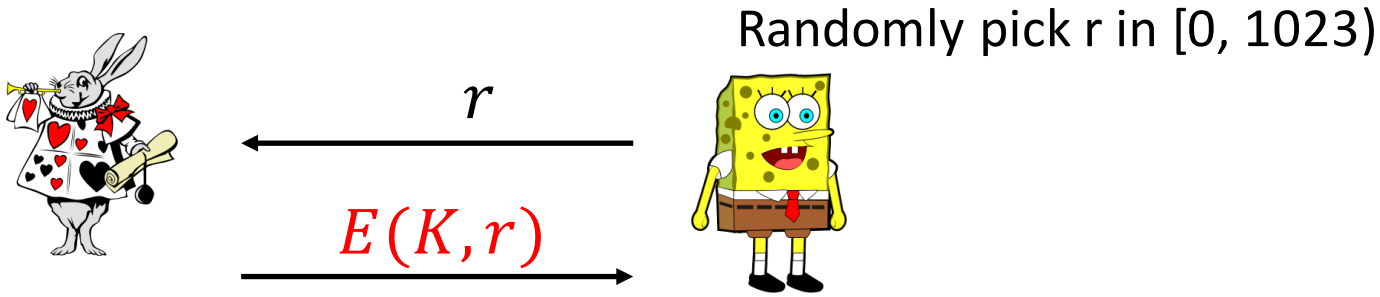
N. Gelernter, S. Kalma, B. Magnezi, and H. Porcilan, "The Password Reset MitM Attack," in *IEEE Symposium on Security and Privacy*, 2017.

Attack 2: Replay attack

Replay attack captures message and re-sends it at a later stage of the protocol

Example:

- Alice and Bob share a symmetric key *K*
- Bob wants to authenticate Alice as follows

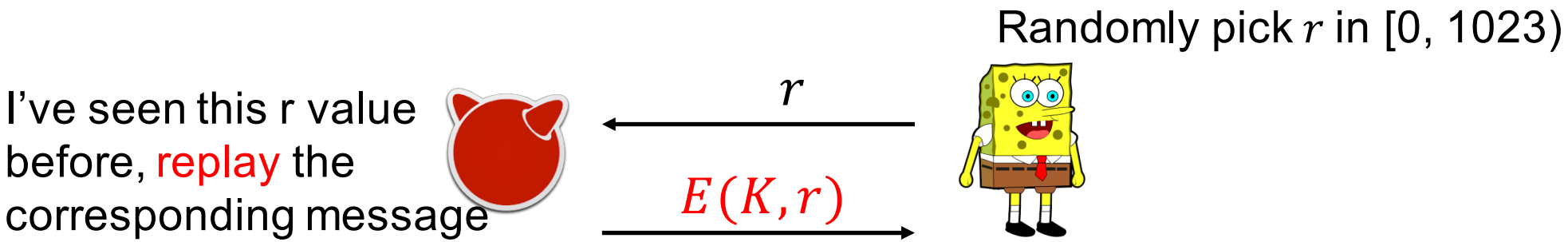


Attack 2: Replay attack

Replay attack captures message and re-sends it at a later stage of the protocol

Example:

- Alice and Bob share a symmetric key K
- Bob wants to authenticate Alice as follows
- Once capturing enough messages, Eve can impersonate Alice!



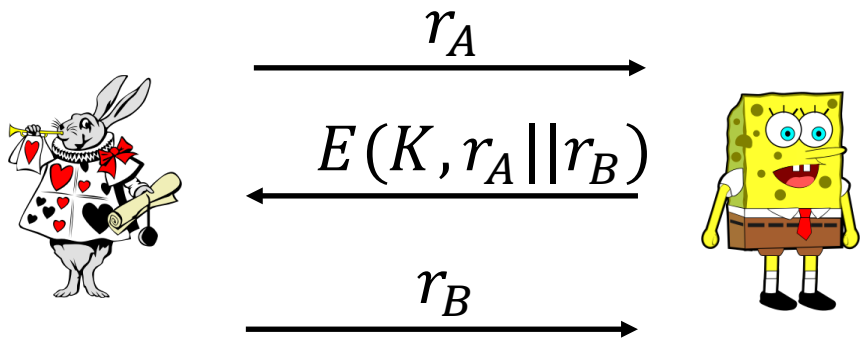
Attack 3: Reflection attack

Reflection attack returns the message to its originator

Commonly used in breaking authentication protocols, making the victim to answer her own challenge

Example:

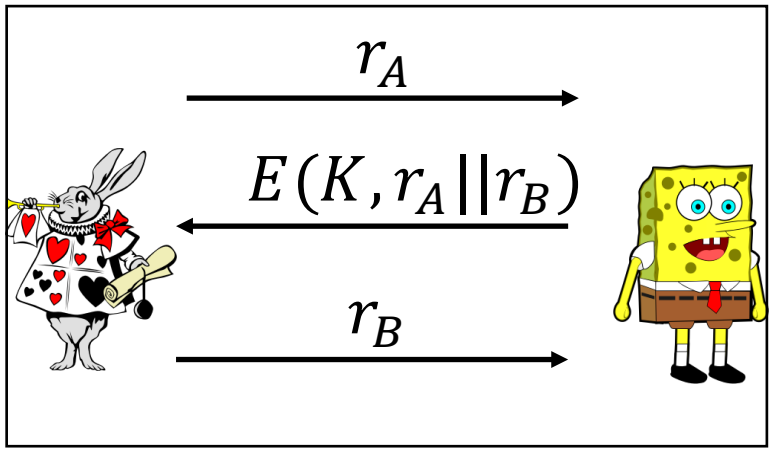
- Alice and Bob share a symmetric key K
- They want to authenticate each other as follows

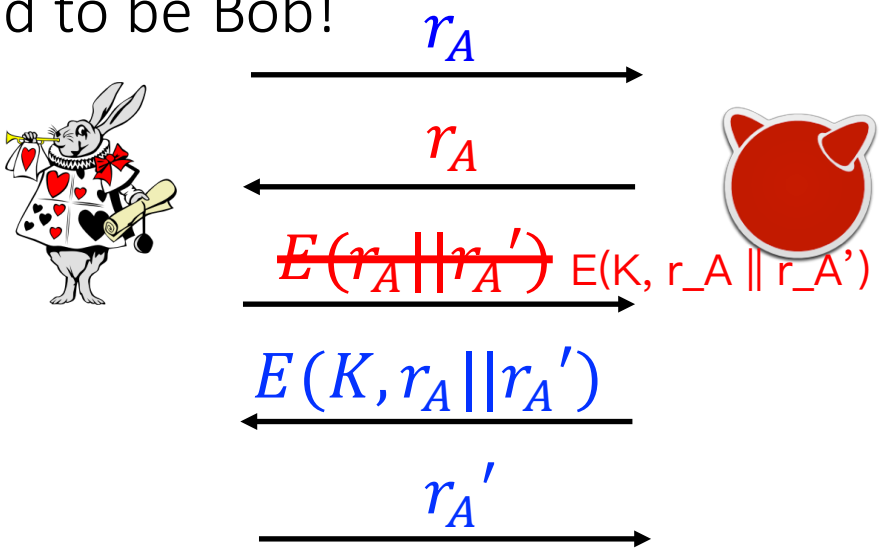


Attack 3: Reflection attack

Example:

- Alice and Bob share a symmetric key K
- They want to authenticate each other
- However, Eve can pretend to be Bob!





Take away

Proof of freshness: to prevent replay attacks, protocols often use nonce, counter, and/or timestamp

Explicit information should be added to indicate the role of the communication entity.

Keep your protocols simple (KISS)

M. Abadi and R. Needham, "Prudent engineering practice for cryptographic protocols," in IEEE Transactions on Software Engineering, vol. 22, no. 1, pp. 6-15, Jan. 1996.