Computer Security and Cryptography

CS381

来学嘉
计算机科学与工程系 电院3-423室
34205440 1356 4100825 laix@sjtu.edu.cn

2016-05

Password protection

Attacks and anti-attacks

- Guessing
  - Restriction, weakwords, Vault
- Stealing
  - Keyloger, graphic,
- Social engineering
- Password cracking
  - Dictionary search
  - PassHashing
- Honey words
- Biological: Finger, face, pupil

Why password?

- Oldest and most widely used authenticator
- Low cost; do not require additional hardware.
- Immediacy: instantaneous account setup
- Convenience: Revocation; Reset; Deploy.
- Resilient: Can be very simple and complicated. (user's needs are diverse.)
- May be a weak link of a security system.
- "the password is dead."—Bill Gates,2004
- Countless designs tend to replace it.
- still appear as the dominant form of authentication

Attacks and anti-attacks

- Guessing
  - Restriction, weakwords, Vault
- Stealing
  - Keyloger, graphic,
  - Social engineering
- Dictionary search
  - PassHashing
- Honey words

Password Guessing

- one of the most common attacks
- attacker knows login-name (from email/web page etc)
- then attempts to guess password for it
  - defaults, short passwords, common word searches
  - user info (variations on names, birthday, phone, common words/interests)
  - exhaustively searching all possible passwords
- check by login or against stolen password file
- success depends on password chosen by user
- surveys show many users choose poorly

常用口令

- 中国网民最常用**10**大密码
- abc123 123456 xiaoming 12345678 iloveyou
- admin qq123456 taobao root wang1234

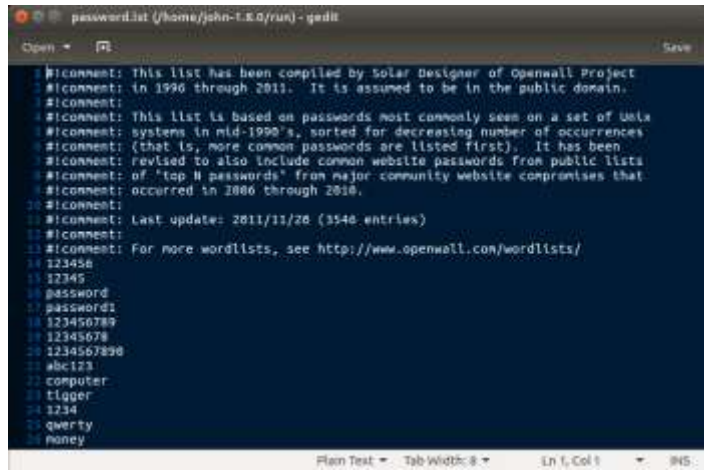
- 国外网民常用的**25** 个**:**
- *password*、123456、12345678、qwerty、abc123、 monkey、1234567、letmein、trustno1、dragon、baseball、111111、iloveyou、master、Sunshine、Ashley、Bailey、passw0rd、 Shadow、123123、654321 Superman、Qazwsx、Michael、football

Weak Passwords

- Google's 10 Worst Password Ideas
  - 1.Pet names
  - 2.A notable date, such as a wedding anniversary
  - 3.A family member's birthday
  - 4.Your child's name
  - 5.Another family member's name
  - 6.Your birthplace
  - 7.A favorite holiday
  - 8.Something related to your favorite sports team
  - 9.The name of a significant other
  - 10.The word "Password"

An example

- [John the Ripper]'s wordlist:



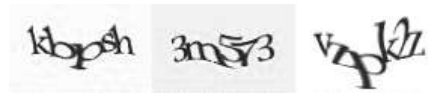
Prevent guessing

- Restrict the login trials
  - locking the account after 3 fails
  - Using CAPTCHA (验证码）to prevent automated password guessing
  - Graphic CAPTCHA

9

CAPTCHA

- CAPTCHA (Completely Automated Public Turing Test to Tell Computers and Humans Apart)
  - a type of challenge-response test used in computing to ensure that the response is not generated by a computer.
  - A common type of CAPTCHA requires that the user type the letters or digits of a distorted image that appears on the screen.



- Attack:
  - deep learning tech, success rate>70% [BH16]

10

Password protection

- Management
- front-line defense against intruders
- users supply both:
  - login – determines privileges of that user
  - password – to identify them
- passwords often stored encrypted
  - Unix uses multiple DES (variant with salt)
  - more recent systems use crypto hash function
- should protect password file on system

Password protections - Education

- use policies and good user education
- educate on importance of good passwords
- give guidelines for good passwords
  - minimum length (>6)
  - require a mix of upper & lower case letters, numbers, punctuation
  - not dictionary words
  - PAO: Person+Action+Object (Alice-catch-bus)
  - Change password periodically
- but likely to be ignored by many users

Passwords - Computer Generated

- let computer create passwords
- if random likely not memorisable, so will be written down (sticky label syndrome)
- even pronounceable not remembered
- have history of poor user acceptance
- FIPS PUB 181 one of best generators
  - has both description & sample code
  - generates words from concatenating random pronounceable syllables

Managing Passwords - Reactive Checking

- reactively run password guessing tools
  - note that good dictionaries exist for almost any language/interest group
- cracked passwords are disabled
- but is resource intensive
- bad passwords are vulnerable till found

Managing Passwords - Proactive Checking

- most promising approach to improving password security
- allow users to select own password
- but have system verify it is acceptable
  - simple rule enforcement (see earlier slide)
  - compare against dictionary of bad passwords
  - use algorithmic (markov model or bloom filter) to detect poor choices

Problem and solutions

- Problem: Low Entropy[CRYPTO89]
  - tradeoff between security and usability
- Solutions: Related standards:
  - FIPS PUB 112, 1985
  - FIPS PUB 181, 1993
  - ISO/IEC 17799(27002, 2005)
  - PKCS#5/RFC2898(2000, 2006)
  - Electronic Authentication Guideline[NIST Special Publication 800-63-2,2013]
  - ISO 11770-4
  - IEEE P1363.2

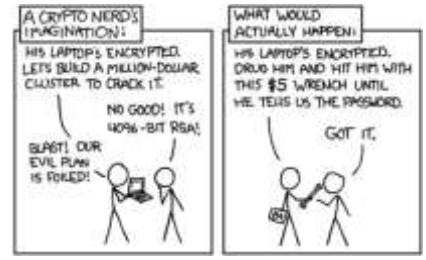
Attacks and anti-attacks

- Guessing
  - Restriction, weakwords, Vault
- stealing
  - Keyloger, graphic,
- Social engineering

- Dictionary search
  - PassHashing
- Honey words

17

Password stealing

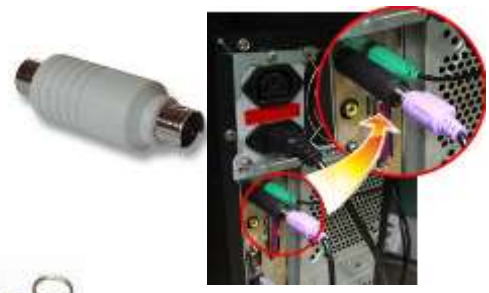


- Eavesdropping
  - Sniffer
- Keylogging
  - malware
- Physical observation attack
  - shoulder-surfing
- Phishing

- share[reuse], account recovery.

Keylogging

- **Keystroke logging**
- Software -- malware, recording (logging) the keys struck of keyboard



- Hardware
- acoustic analysis

19

shoulder surfing

- ATM：
  - Install card reader to get card number
  - Install a camera to Get password

20

phishing

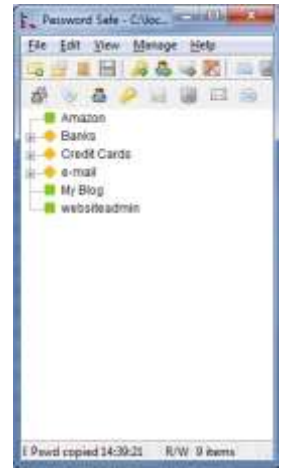
- Similar URL
- https is not on
- 预留信息

Password Vault

- Problem:
  - too many, and too weak passwords.
- Simple Solution: Password manager tools
  - generate and keep strong passwords automatically, and users only need to remember a master password.
- How:
  - k=KDF(master-password)
  - Enc_k(passwords)
- Local or web-based.

Some Products

- PCMag 2016
  - Dashlane
  - LastPass
  - Sticky Password
  - RoboForm
  - Keeper
  - LogMeOnce
  - Password Boss
  - …
- Trust,Robust?



Attacks on Password Managers

- Remote extraction[USENIX14-Silver]
- XSS,CSRF, JavaScript[USENIX14-Li]

- Besides, good support for roaming is needed. Single point of failure may be a problem.
- Still, “[A Password Manager] is one of the best ways to keep track of each unique password or passphrase that you have created for your various online accounts without writing them down on a piece of paper and risking that others will see them.”—US-CERT, 2012

Secure use of password

- A: Password π, verifier B knows $k=H(\pi)$
- A sends e_k(data) to B, B check e_k(data) .
  - Brute-force attack: guess π', check $e_{k'}$(data)
  - Could be easier than breaking the cipher.
- Solution
  - B generates a public key p_B, send to A.
  - A send $e_{pB}(\pi$, nonce) to B
  - Brute-force attack becomes difficult (need to break the public-key cipher)

- ISO 11770-4, IEEE P1363.2

Attacks and anti-attacks

- Guessing
  - Restriction, weakwords, Vault
- stealing
  - Keyloger, screen capture
  - Phishing
- Dictionary search （cracking）
- PassHashing
- Honey words

Password Cracking

- 1.Brute force guess (Cracker)
- 2.Dictionary attack(Smart guess)
  - Example: Morris worm.

- Tools:
  - John the Ripper, Ophcrack, RainbowCrack, L0phtCrack, Cain and Abel

- Methods:
  - Rainbow table(Time/Space trade-off)[no salt]
  - Markov modeling techniques
  - Probabilistic context-free grammar

Password file

- [login-name; password]
  - unsafe but still in use

- [login-name; hash(password)]
  - better but crackable with pre-computation

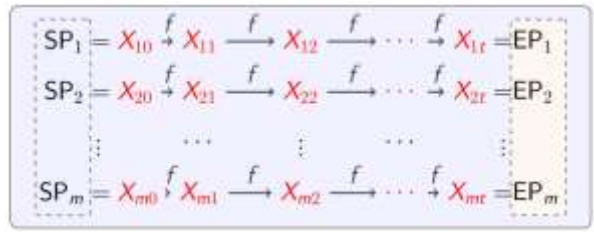
- [login-name; salt; hash(password,salt)]
  - now standard, still has problem

- Non-readable store -- recent device

28

Rainbow Table

- Originate from a time-memory tradeoff method.[Hellman 1980]



- Only store two list SP_i and EP_i, calculate X_{ij} from SP_i
  - SP: start point EP: end point
- f is a mapping: $f_k=R(E_k(P))$, R is a reduction function.
- Essentially a brute force attack .

Markov modeling techniques

- The distribution of letters in memorable passwords is similar to that of native language.
- Markov modeling techniques can be used to reduce the size of the password space.[CCS05]
- Zero order:

model: $P(\alpha) = \prod_{x\in\alpha} v(x)$
dictionary: $\mathcal{D}_{v,\theta} = \{\alpha : \prod_{x\in\alpha} v(x) \geq \theta\}$

- First order:

model: $P(x_1x_2\ldots x_n) = v(x_1)\prod_{i=1}^{n-1} v(x_{i+1}|x_i)$
dictionary:
$\mathcal{D}_{v,\theta} = \{x_1x_2\ldots x_n : v(x_1)\prod_{i=1}^{n-1} v(x_{i+1}|x_i) \geq \theta\}$

Markov modeling techniques

- Regular expressions: descript the patterns of passwords. And can be converted to deterministic finite automata.
- Hybrid Markovian/DFA dictionary:

$$\mathcal{D}_{v,\theta,\langle M_i\rangle} = \left\{\alpha : \prod_{x\in\alpha} v(x) \geq \theta, \text{and } \exists i : M_i \text{ accepts } \alpha\right\}$$

- Experiment result: compare with rainbow table

Category	Count	Rainbow	Hybrid
Length ≤5	63	29	63
Length 6	21	10	17
Length 7	18	0	10
Length 8,A*	9	0	6
Others	31	0	0
Total	142	39(27.5%)	96(67.6%)
only length≥6	79	79(12.7%)	33(41.8%)

Probabilistic Context-Free Grammars

- Facts:
  1.not all guesses have the same probability.
  – Example: Pr(“password12”) > Pr(“P@$$W0rd!23”)
  2.memorable passwords are patterned.
  – Example: $password123: {1Special}{8Letters}{3Digits}

- Attack:[SnP09]
  – 1.prepare training sets;
  – 2.derive word-mangling rules and probabilities;
  – 3.given input dictionaries, generate password guesses;
  – 4.dictionary attack.

Attacks and anti-attacks

- Guessing
  - Restriction, weakwords, Vault
- stealing
  - Keyloger, graphic,
  - Social engineering

- Dictionary search
- PassHashing
- Honey words

35

Password hashing

- Aim：make the password verification SLOW
- History
  - Plaintext (immediate)
  - Encryption (fast)
  - Iterated Encryption with salt (slower)
  - Dedicated hash^{1000} (slower)
- Dictionary Crack can be improved with:
  - CPU, GPU, ASIC, FPGA
- Schemes:
  - RFC2433 ,crypt, md5crypt, bcrypt, PBKDF2,..., Argon2

PHC

- Password Hashing Competition (PHC)
  - Secure: Crypto-hash
  - Slow: efficiency improvement should be minimal
- Timeline
  - 2012 fall, Initiation
  - 2013 Q1, call for submissions
  - 2014 March 31, deadline
  - 2014 December, 9 finalists
  - 2015 July, announced the winner
- Now cryptanalysis is coming.

37

Submissions, Finalists and Winner

Name	Designer(s)
AntCrypt	Markus Duermuth, Ralf Zimmerman
Argon and Argon2	Alex Biryukov, Dmitry Khovratovich
battcrypt	Steve Thomas
Catena	Christian Forler, Stefan Lucks, Jakob Wenzel
Catfish	Bo Zhu, Xinxin Fan, Qi Chai, Guang Gong
Centrifuge	Rafael Alvarez
EARWORM	Daniel Franke
Gambit	Krisztian Pinter
Lanarea	Haneef Mubarak
Lyra2	Marcos A.Simplicio Jr, Leonardo C.Almeida, Ewerton R.Andrade, Paulo C.F.dos Santos, Paulo S.L.M.Barreto
M3lcrypt	Isaiah Paul Makwakwa
Makwa	Thomas Pornin
MCS_PHS	Mikhail Maslennikov
Omega Crypt	Brandon Enright
Parallel	Steve Thomas
PolyPassHash	Justin Cappos, Santiago Torres Arias
POMELO	Hongjun Wu
Pufferfish	Jeremi Gosney
RIG	Donghoon Chang, Arpan Jati, Sweta Mishra, Somitra Kumar Sanadhya
Schvrch	Rade Vuckovac
Tortuga	Teath Sch
TwoCats	Bill Cox
Yarn	Evgeny Kapun
Yescrypt	Alexander Peslyak

Overview

	Algorithm	Based On	Memory Usage RAM	Memory Usage ROM	Parallel	Primitive BC/SC/PERM	Primitive HF	Mode
Finalists	Argon	AES	1 kB - 1 GB	-	-	AES (5R)	BLAKE2b	-
	Argon2d		250 MB - 4 GB	-	✓	BLAKE2b (CF, 2R)	BLAKE2b	-
	Argon2i		1 GB - 6 GB	-	✓	BLAKE2b (CF, 2R)	BLAKE2b	-
	battcrypt	Blowfish/bcrypt	128 kB - 128 MB	-	part.	Blowfish-CBC	SHA-512	-
	Catena-DBG	DBG	4 MB	-	part.	-	BLAKE2b/BLAKE2b-1	-
	Catena-BRG	BRG	128 MB	-	part.	-	BLAKE2b/BLAKE2b-1	-
	Lyra2	Sponge	400 MB - 1 GB	-	✓	BLAKE2b (CF)/(BlaMka)	-	-
	Makwa	Signatings	negl.	-	✓	-	SHA-256	HMAC_DRBG
	Parallel		negl.	-	✓	-	SHA-512	-
	POMELO		(8 KB, 256 GB)	-	✓	-	-	-
	Putterfish	mod. Blowfish/bcrypt	4 kB - 16 kB	-	-	mod. Blowfish	SHA-512	HMAC
	yescrypt	scrypt	3 MB	3 GB	part.	Salsa20/8	SHA-256	PBKDF2 HMAC
Non-Finalists	AntCrypt		32 kB	-	part.	-	SHA-512	-
	CENTRIFUGE		2 MB	-	-	AES-256	SHA-512	-
	EARWORM		-	2 GB	✓	AES (1R)	SHA-256	PBKDF2 HMAC
	Gambit	Sponge	50 MB	-	-	Keccak	-	-
	Lanarea DF		256 B	-	-	-	BLAKE2b	-
	MCS_PBS		negl.	-	-	-	MCSSHA-8	-
	ocrypt	scrypt	1 MB - 1 GB	-	-	ChaCha	CuboHash	-
	PolyPassHash	Shamir Sec. Sharing	negl.	-	-	AES	SHA-256	-
	Rig	BRG	15 MB	-	part.	-	BLAKE2b	-
	scrypt		1 GB	-	-	Salsa20/8	-	PBKDF2 HMAC
	scherch		8 MB	-	part.	-	-	-
	Tortuga	Sponge & recursive Feistel	o	-		Turtle	-	-
	SkinnyCat	BRG	o	-	-	-	SHA-*/BLAKE2*	-
	TwoCats	BRG	o	-	✓	-	SHA-*/BLAKE2*	-
	Yarn		o	-	part.	BLAKE2b (CF), AES	-	-

39

Argon2

- Argon2i and Argon2d are parametrized by
  - A time cost, which defines the execution time
  - A memory cost, which defines the memory usage
  - A parallelism degree, which defines the number of threads
- Specification and demo code:
  - https://github.com/P-H-C/phc-winner-argon2

40

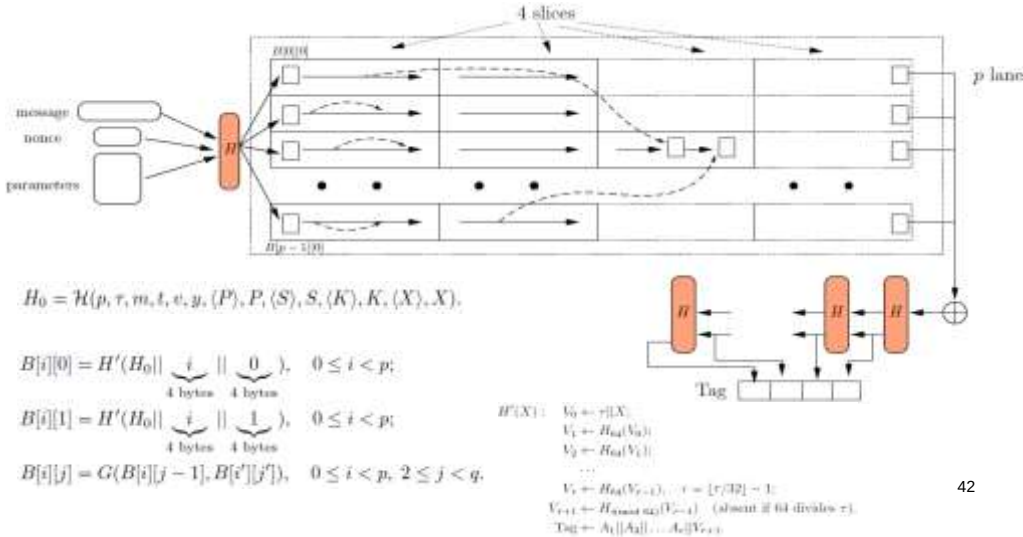
Inputs

- Primary inputs:
  - Message P
  - Nonce S
- Secondary inputs:
  - Degree of parallelism p
  - Tag length τ
  - Memory m
  - Number of iterations t
  - Version number v
  - Secret value K
  - Associated data X
  - Type y

41

Operation

- Single-pass Argon2 with p lanes and 4 slices.



$$H_0 = \mathcal{H}(p, \tau, m, t, v, y, |P|, P, |S|, S, |K|, K, |X|, X).$$

$$B[i][0] = H'(H_0 || \underbrace{i}_{4\text{ bytes}} || \underbrace{0}_{4\text{ bytes}}), \quad 0 \le i < p;$$

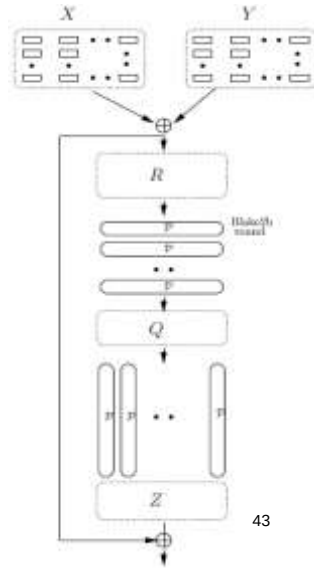
$$B[i][1] = H'(H_0 || \underbrace{i}_{4\text{ bytes}} || \underbrace{1}_{4\text{ bytes}}), \quad 0 \le i < p;$$

$$B[i][j] = G(B[i][j-1], B[i'][j']), \quad 0 \le i < p,\ 2 \le j < q.$$

$H'(X)$:
$V_0 \leftarrow \tau || X$
$V_1 \leftarrow H_{64}(V_0)$
$V_2 \leftarrow H_{64}(V_1)$
...
$V_r \leftarrow H_{64}(V_{r-1}), \quad r = \lceil \tau/32 \rceil - 1$
$V_{r+1} \leftarrow H_{\tau \bmod 32}(V_{r-1}) \quad$ (absent if 64 divides τ)
$\text{Tag} \leftarrow A_1[0]||A_2[0]|| \ldots A_r[0]||V_{r+1}$

42

Compression function G



$$G: \quad (X,Y) \rightarrow R = X \oplus Y \xrightarrow{\mathcal{P}} Q \xrightarrow{\mathcal{P}} Z \rightarrow Z \oplus R.$$

$$(Q_0, Q_1, \ldots, Q_7) \leftarrow \mathcal{P}(R_0, R_1, \ldots, R_7);$$
$$(Q_8, Q_9, \ldots, Q_{15}) \leftarrow \mathcal{P}(R_8, R_9, \ldots, R_{15});$$
$$\ldots$$
$$(Q_{56}, Q_{57}, \ldots, Q_{63}) \leftarrow \mathcal{P}(R_{56}, R_{57}, \ldots, R_{63});$$

$$(Z_0, Z_8, Z_{16}, \ldots, Z_{56}) \leftarrow \mathcal{P}(Q_0, Q_8, Q_{16}, \ldots, Q_{56});$$
$$(Z_1, Z_9, Z_{17}, \ldots, Z_{57}) \leftarrow \mathcal{P}(Q_1, Q_9, Q_{17}, \ldots, Q_{57});$$
$$\ldots$$
$$(Z_7, Z_{15}, Z_{23}, \ldots, Z_{63}) \leftarrow \mathcal{P}(Q_7, Q_{15}, Q_{23}, \ldots, Q_{63}).$$

- P is Blake2b

$\mathcal{P}(A_0, A_1, \ldots, A_7) \quad (B_0, B_1, \ldots, B_7)$

43

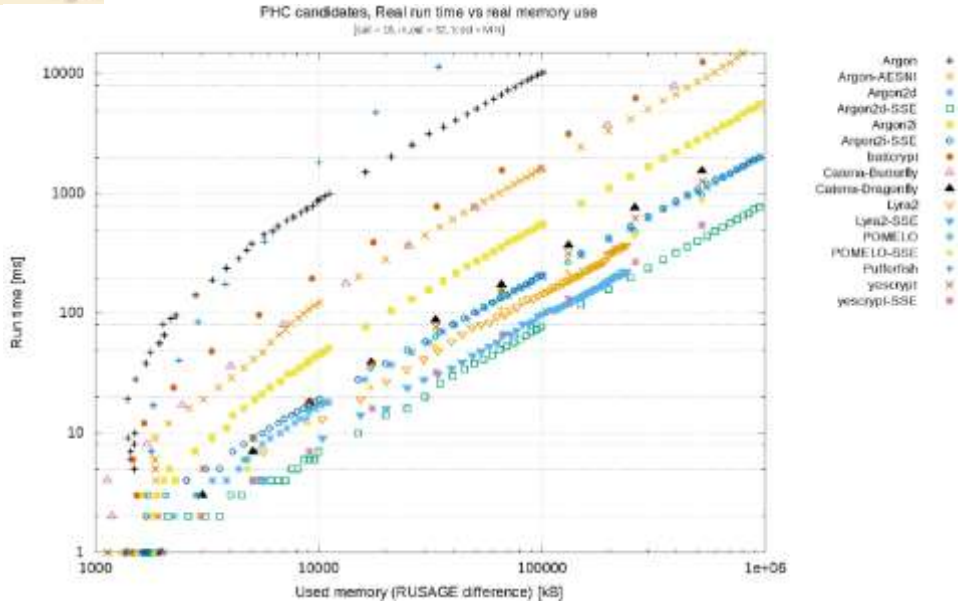
Performance

- Core i7-4500U Intel Haswell 1.8 GHz, 4 cores
- 1 GB memory filled

Processor	Threads	Argon2d (1 pass)		Argon2i (3 passes)	
		Cycles/Byte	Bandwidth (GB/s)	Cycles/Byte	Bandwidth (GB/s)
i7-4500U	1	1.3	2.5	4.7	2.6
i7-4500U	2	0.9	3.8	2.8	4.5
i7-4500U	4	0.6	5.4	2	5.4
i7-4500U	8	0.6	5.4	1.9	5.8

46

PHC Benchmarks



Security

- Memory-Hard: $D(\alpha) > 1/\alpha \text{ as } \alpha \to 0$
- Adv's goal: $\mathcal{E}_{max} - \max_{\alpha} \frac{1}{\alpha D(\alpha)}$
- Side channel attack[timing,cache,AC14]
- Ranking tradeoff attack[AC15]

α	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{6}$	$\frac{1}{7}$
$C(\alpha)$	1.5	4	20.2	344	4660	2^{18}
$D(\alpha)$	1.5	2.8	5.5	10.3	17	27

Time and computation penalties for the ranking tradeoff attack for the Argon2(one pass) indexing function

- AT product reduction:
  - Argon2i: 5 [ePrint16]
  - Argon2d: ?

48

撞库

- 通过收集互联网已泄露的[用户;口令】，生成对应的字典表，尝试批量登陆其他网站。
- 很多用户在不同网站使用相同的帐号密码，
- 因此可以通过获取用户在A网站的账户从而尝试登录B网址，这就可以理解为撞库攻击。[1]
- 2014年12月25日，12306网站用户信息在互联网上疯传。对此，12306官方网站称，网上泄露的用户信息系经其他网站或渠道流出。据悉，此次泄露的用户数据不少于131,653条。该批数据基本确认为黑客通过“撞库攻击”所获得

49

Alternatives

- Alternatives to passwords for authentication

- graphic
- One-time password (token)
- Finger print
- Face recognition
- Iris recognition

50

New Designs

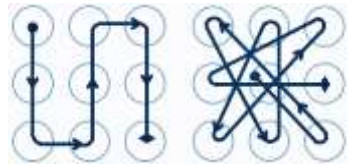
- Naturally Rehearsing Passwords[AC13]
  - Person-Action-Object stories

  - Security vs Usability
    - Usability: PAO — Easy to remember.
    - Security: try to increase the entropy.—Hard to break. ?

Note the difference between graphic password and graphic CAPTCHA

Android lock patterns

- Easy to use:



- May be predictable too:[DEFCON15]

Length	#Combinations
4	1624
5	7152
6	26,016
7	72,912
8	140,704
9	140,704
Total	389,112

One-time password

One-time password, change every 60 sec.

Who you are
What you know
What you have

User supply:
- Acc. number
- Password
- SecureID number

Bank check
- acc. Number
- Password
- the numbers stored

- $SID=h(userID,key,N)$ $N>N_0$

Hash, AES

53

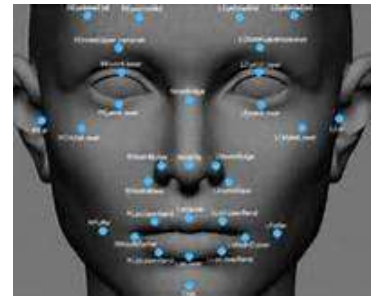
Finger print

- Human fingerprints are nearly unique, difficult to alter, and durable over the life of an individual, making them suitable as long-term markers of human identity
- Need special hardware
- Can be forged if not well-designed

54

Face/iris recognition

- Similar to finger print
- nearly unique for human
- Need hardware
- Can be forged if not well-designed



55

Attacks and anti-attacks

- Guessing
  - Restriction, weakwords, Vault
- stealing
  - Keyloger, graphic,
  - Social engineering

- Dictionary search
- PassHashing
- Honey words

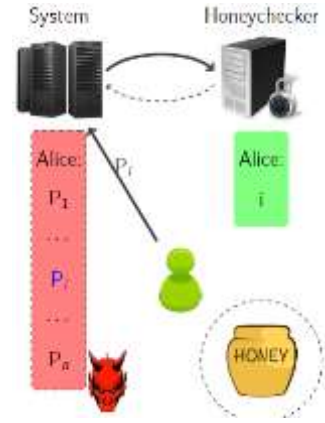
56

Honeywords

- Assuming passwords are always crackable.[CCS13]

- Attack scenarios:
  - Files of password hashes are stolen;
  - Get the passwords by brute-force/dic attack.

  Is everything lost?



Main system

- For each user u_i, a list W_i of distinct words is represented: $W_i = (w_{i,1},w_{i,2},\ldots,w_{i,k})$, main system's file is (u_i,H_i), where $H_i=(v_{i,1},v_{i,2},\ldots,v_{i,k})$ and $v_{i,j}=Hash(w_{i,j})$
  - K is a parameter, may be different in different circumstances.
- If a user(or adversary) enters correct password, then logins as usual; else if honeyword, then:
  - Setting off an alarm;
  - Letting login proceed as usual;
  - Letting the login proceed, but on a honeypot system;
  - Tracing the source of the login carefully;
  - Turning on additional logging of the user's activities;
  - Shutting down that user's account until the user establishes a new password.
  - Shutting down the computer system and requiring all users to establish new passwords.

Honeychecker

- Honeychecker is a separate hardened computer system.
- Communicate with the main system over a secure channel.
- Raising an alarm when an irregularity is detected.
- May or may not reply to the main system when a login is attempted.
- Maintains a single database value c(i) for each user u_i.
- Accepts two types of commands:
  - Sets c(i) to have value j;
  - Checks that c(i)==j

Design principles

- Provide a basic form of distributed security.
  - Compromise of the honeychecker database at worst only reduces security to the level it was at before the introduction of honeywords and the honeychecker.
- Few system changes and little overhead.
- A user does not need to know the values of the honeywords or even know about their existence.

Honeyword Generation

- Methods:
  (1)Tweak selected character positions of the password.
  (2)Randomly chosen a “tail” and append to the input password to form a new password.
- for(1):
  – No provable guarantee.
  – Typos may cause accidents
- for(2):
  – ”perfectly flat”
  – May burden the users

Attacks

- General password guessing;
- Targeted password guessing
  – personal information
- Attacking honeychecker
  – Get c(i)
- Likelihood attack
  – Bad honeyword generation algorithm
- Denial of service
  – Attack honeychecker
- Multiple systems attack
  – Shared password

Exercise 19

1. Describe the methods of
   – Attacking passwords
   – Protecting passwords

2. How to protect your passwords if there are undetected keystroke loggers in your system?