Authentication

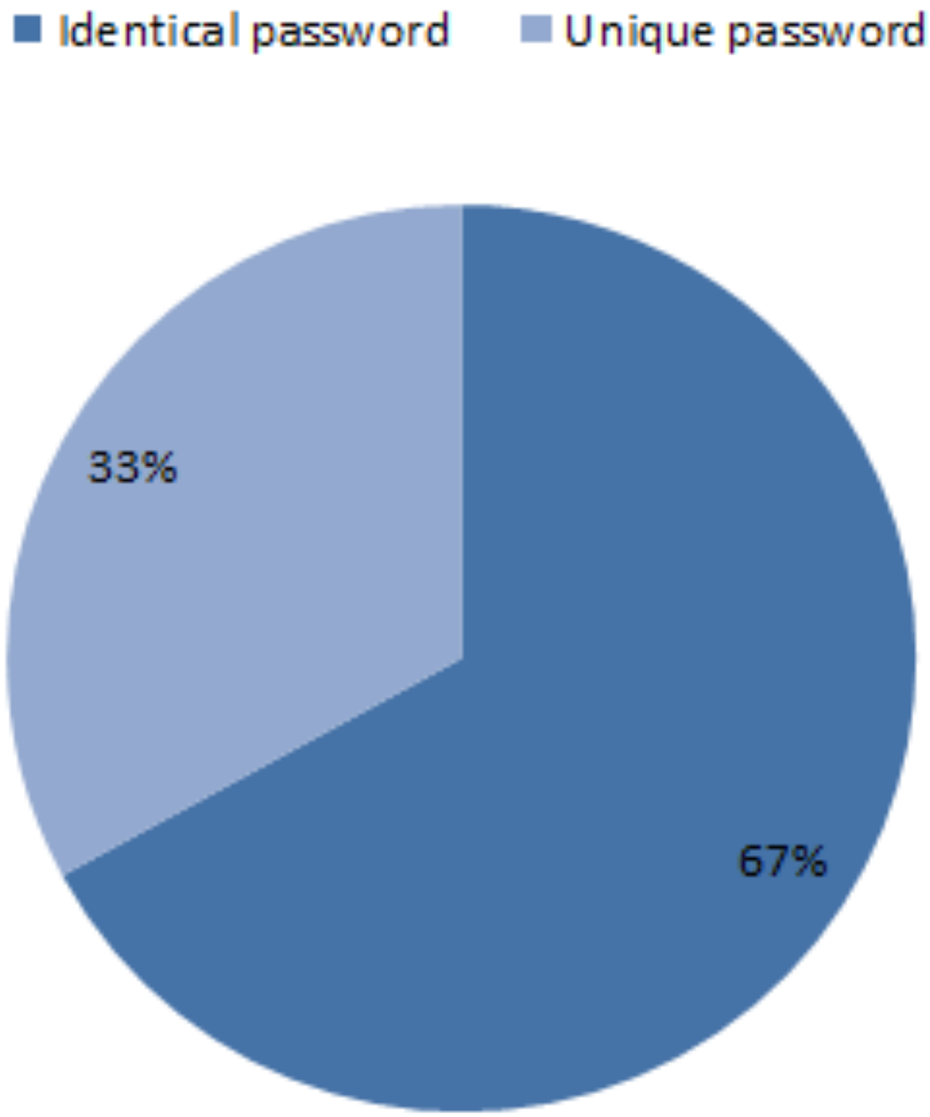
Objectives

- Explain Identification and Authentication.
- Explain why passwords are terrible.
- Explain what makes a good hashing algorithm.
- Store a password *securely* using BCrypt.


John Doe
3245 0557 5106 5406 5465 7065 76799


User name
Password
X
Cancel
Next
MacBook Pro

Password reuse across Sony and Gawker



THE TOP (WORST) PASSWORDS

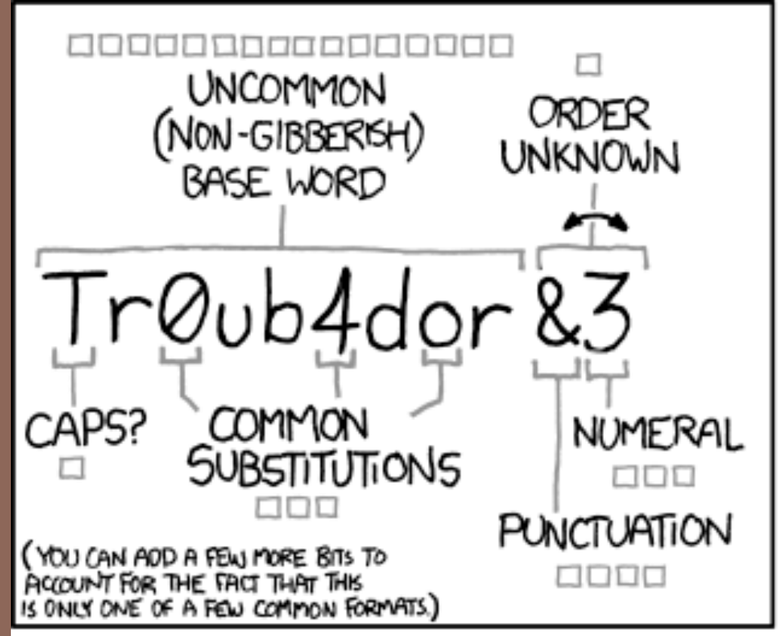
RANK	PASSWORD
1	123456
2	PASSWORD
3	12345
4	12345678
5	QWERTY
6	123456789
7	1234
8	BASEBALL
9	DRAGON
10	FOOTBALL
11	1234567
12	MONKEY
13	LETMEIN
14	ABC123
15	111111
16	MUSTANG
17	ACCESS
18	SHADOW
19	MASTER
20	MICHAEL
21	SUPERMAN
22	696969
23	123123
24	BATMAN
25	TRUSTNO1

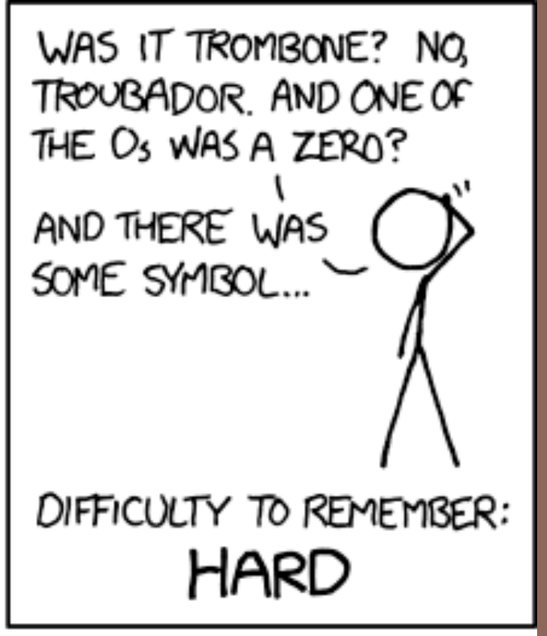
Credit: SplashData

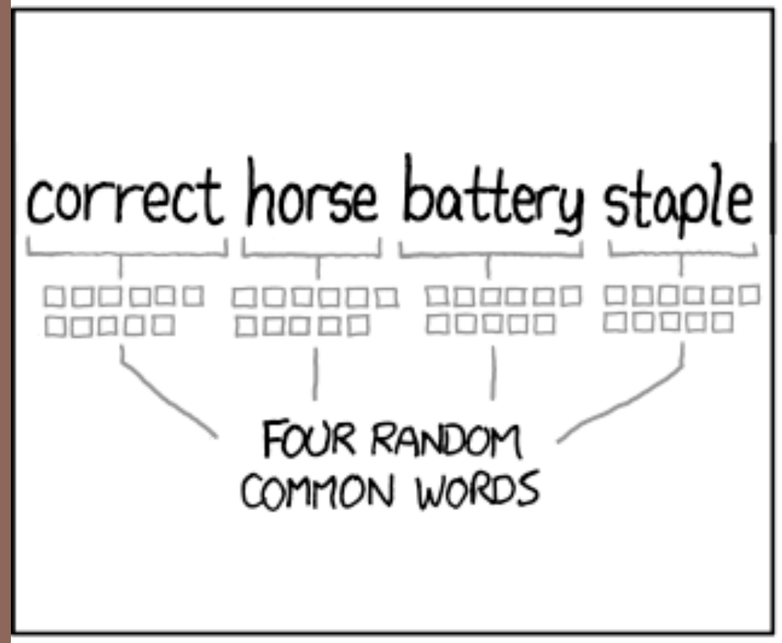
Help: List of Password Rules

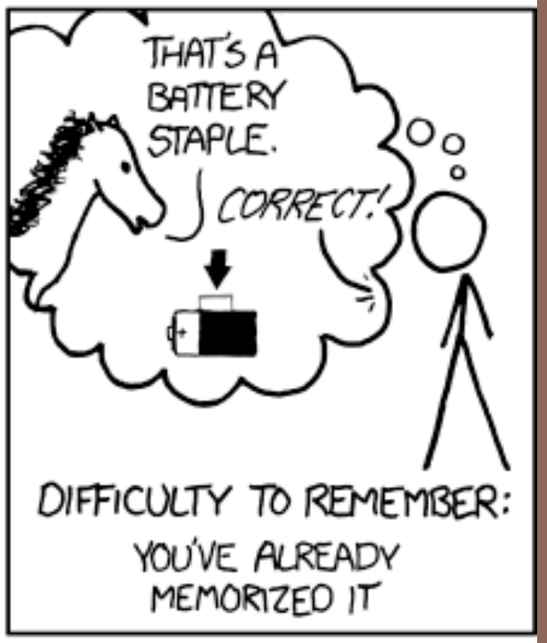
1. The password must be **exactly** 8 characters long.
2. It must contain **at least** one letter, one number, and one special character.
3. The **only** special characters allowed are: @ # $
4. A special character must **not** be located in the first or last position.
5. Two of the same characters sitting next to each other are considered to be a "set." No "sets" are allowed.
6. Avoid using names, such as your name, user ID, or the name of your company or employer.
7. Other words that cannot be used are Texas, child, and the months of the year.
8. A new password cannot be too similar to the previous password.
    a. Example: previous password - abc#1234, acceptable new password - acb$1243
    b. Characters in the first, second, and third positions cannot be identical. (abc*****)
    c. Characters in the second, third, and fourth positions cannot be identical. (*bc#****)
    d. Characters in the sixth, seventh, and eighth positions cannot be identical. (*****234)
9. A password can be changed voluntarily (no Help Desk assistance needed) once in a 15-day period. If needed, the Help Desk can reset the password at any time.
10. The previous 8 passwords cannot be reused.

Top of page









THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Storing Passwords


Rambler.ru Hacked
100 Million Plaintext Passwords Leaked
3:55
Папки
Входящие 4
Найти письма
Настя Иванова
Фотографии из отпуска
Igor Silkin
Олег Поляков
Merdan Agayev
Жду ответа на
Роман Седых
Новая таблица
Рамблер.Касса
Кино в зимние каникулы
password, 12345, passcode, 000000, 654321, 123123, password123, password, 12345, passcode, 000000

First Rule

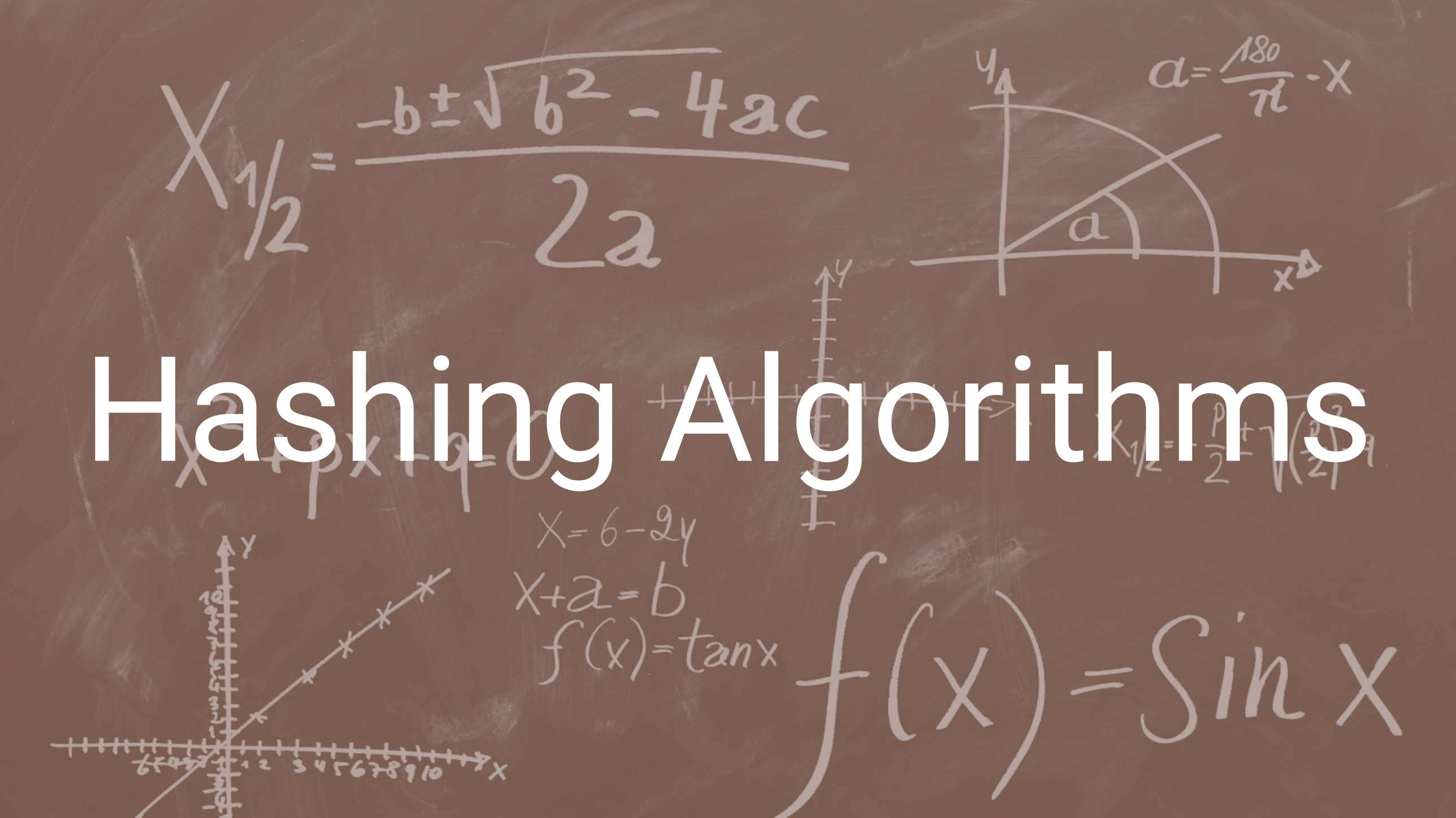
Never store passwords in plain text.

Second Rule

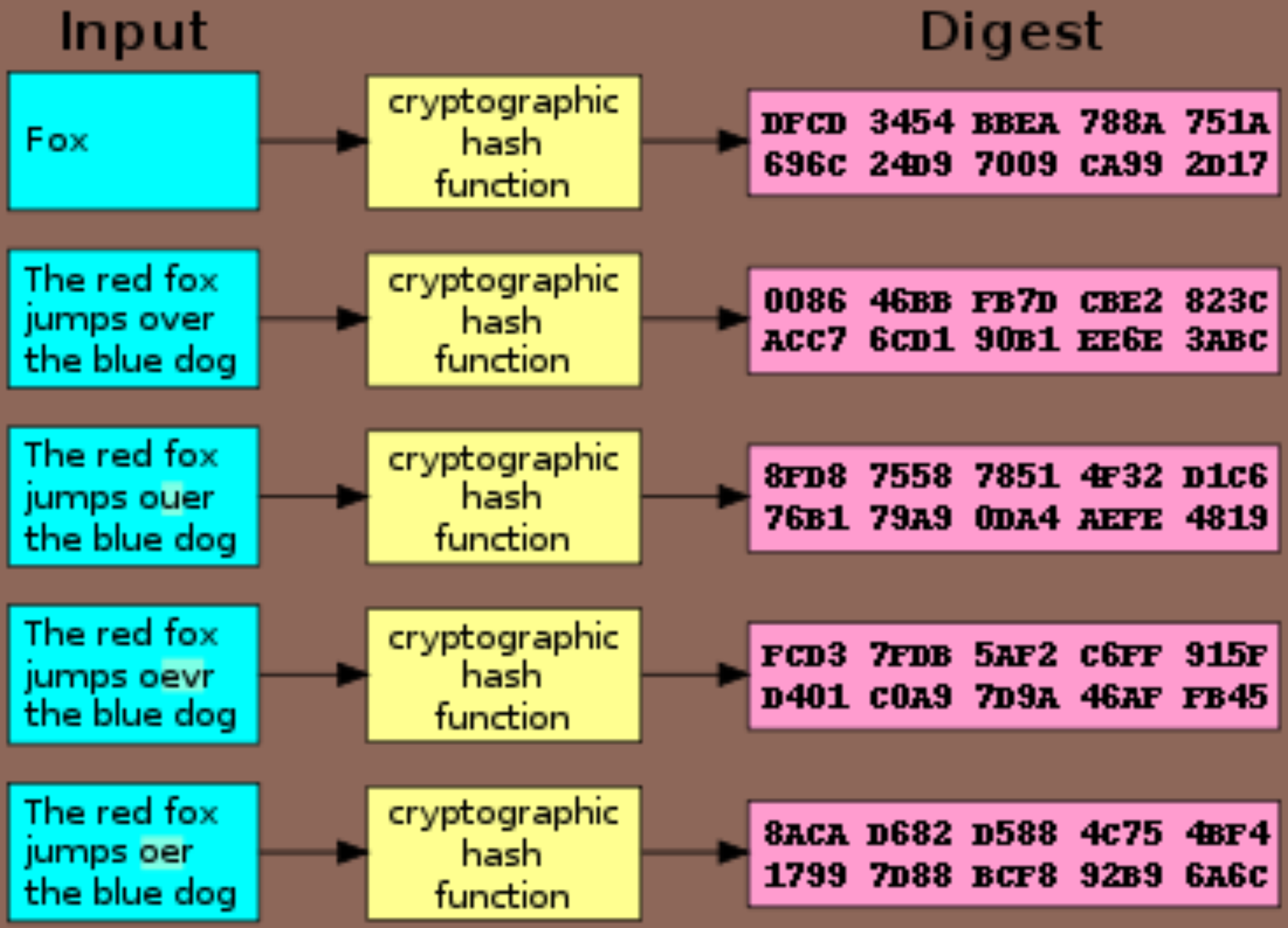
Never store passwords in plain text!

Always ask:

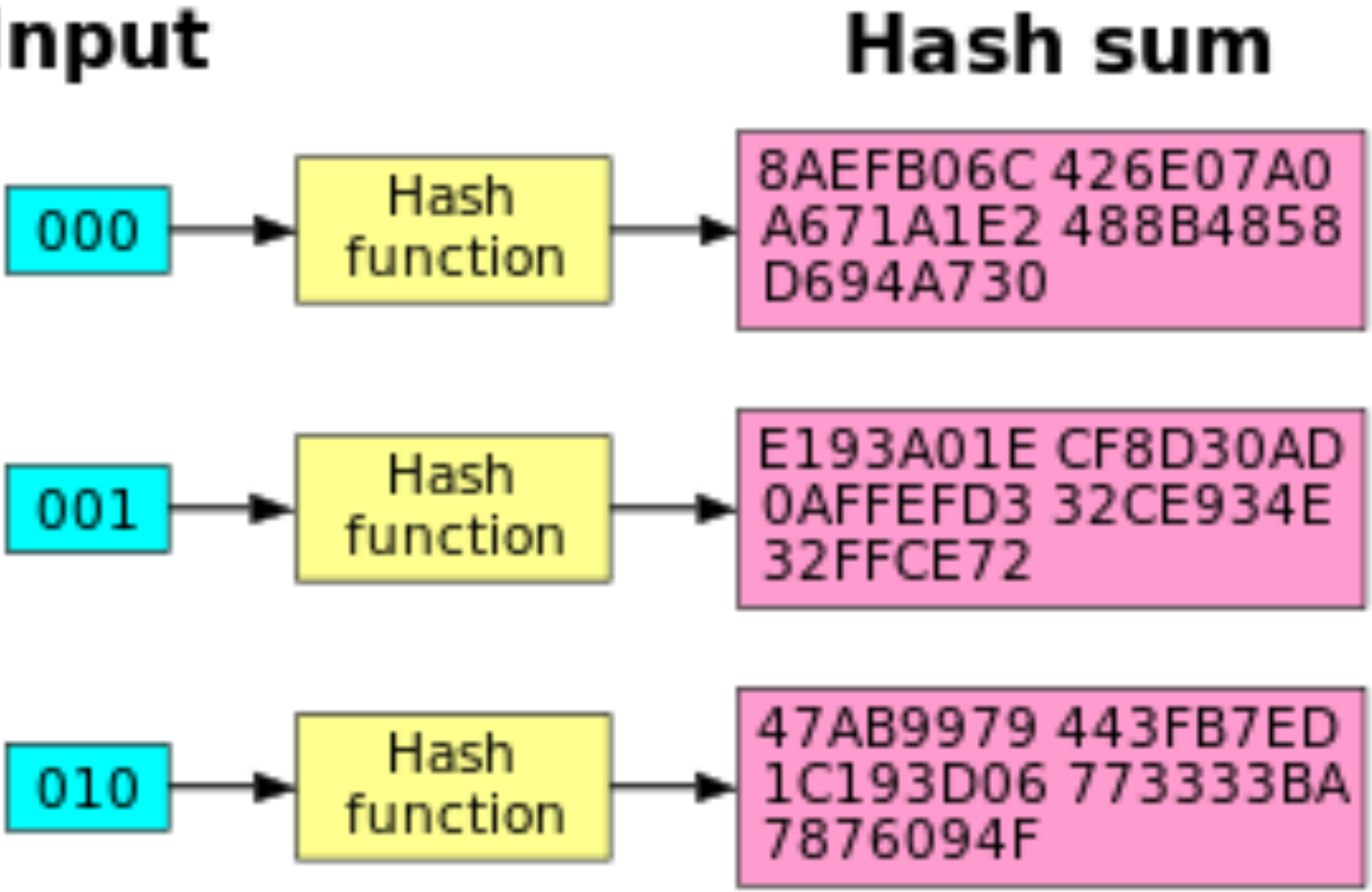
Do I actually need to store this?


Hashing Algorithms


One way


Input
Digest
Fox
cryptographic hash function
DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17
The red fox jumps over the blue dog
cryptographic hash function
0086 46BB FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC
The red fox jumps ouer the blue dog
cryptographic hash function
8FD8 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819
The red fox jumps oevr the blue dog
cryptographic hash function
FCD3 7FDB 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45
The red fox jumps oer the blue dog
cryptographic hash function
8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C


EASY


Input
Hash sum
000
Hash function
8AEFB06C 426E07A0 A671A1E2 488B4858 D694A730
001
Hash function
E193A01E CF8D30AD 0AFFEFD3 32CE934E 32FFCE72
010
Hash function
47AB9979 443FB7ED 1C193D06 773333BA 7876094F

Are We Safe?

-::TYPE	-::HASH	-::PASS	-::STATUS	-::TIME	-::SUBMITED
md5	7e89bcc6151b24992a255cd665d4aa16		waiting	0:0:46	2006-11-11 10:45:31
md5	0696eeaff05bf2105b0bcf6d93ac73a0		waiting	0:0:47	2006-11-11 10:45:30
md5	db549b9d18aabe8ad07aa3d9338d441c		waiting	0:1:38	2006-11-11 10:44:39
md5	70c9ecbd2512460fa861de25fb3d7c6e		waiting	0:24:8	2006-11-11 10:22:09
md5	c32cf089d464d3ed1a3af347ae208188		processing3	0:25:6	2006-11-11 10:21:11
md5	c6fe5851aff10a64e8a52e82b323304f		processing3	0:46:29	2006-11-11 09:59:48
md5	a79c879d28c5c8a4707d52bbaa57607f	12050	cracked	0:45:41	2006-11-11 09:51:43
md5	a79e1c64d27737e3f959a6a56b41c650		processing3	0:57:18	2006-11-11 09:48:59
md5	2ef5b8b0eee93568a1126bb923664057		processing3	0:57:36	2006-11-11 09:48:41
md5	e53cc072934b25e45dc273c6c342556d		processing3	0:58:7	2006-11-11 09:48:10
md5	d38ad0e58c9525343f492161b87400a1	htmldb	cracked	0:58:23	2006-11-11 09:44:01
md5	d926dbaeb7fac97612ec219f7f172610		processing3	1:4:30	2006-11-11 09:41:47
md5	fcf2483ced1768308584987713 4fd50c		processing3	1:6:32	2006-11-11 09:39:45
md5	377a8f80271a6f920df0e4aa84d1029a	bombi	cracked	0:43:12	2006-11-11 09:38:26
md5	85d95e2ad51bfcd5d6d352486fbe2769	pupsi	cracked	1:8:2	2006-11-11 09:28:25
md5	96bc2c727049b5dce27bd8b9e8b264bf		processing3	1:19:6	2006-11-11 09:27:11
md5	8aa12bbde69504ba86b942726b4d7623		notfound	1:18:15	2006-11-11 09:02:54
md5	5ce1d8097499634487676 22e0ca8169f	28264451	cracked	0:48:15	2006-11-11 09:02:35

Third Rule

Never just store hashes!

As you may or may not recall, given how much time has passed, hackers broke into LinkedIn's network back in 2012, stole some 6.5 million encrypted passwords, and posted them onto a Russian hacker forum. Because the passwords were stored as unsalted SHA-1 hashes, hundreds of thousands were quickly cracked.

This time, the breach is said to come from MySpace, and the number of passwords claimed is an eye-popping 427 million.

Apparently, there are only 360 million users on the list, but some accounts have more than one password listed, for reasons that aren't explained.

Once again, the passwords allegedly exposed in this breach were simple, unsalted SHA-1 hashes, vulnerable to just the same sort of high-speed *try 'em all* attack as in the LinkedIn breach of 2012.

BCrypt

BCrypt gives us

1. Good cryptographic hashing algorithm.
2. Automatically generated salts.
3. Control over the computational complexity.

From @garthk, on a 2GHz core you can roughly expect:

```
rounds=8 : ~40 hashes/sec
rounds=9 : ~20 hashes/sec
rounds=10: ~10 hashes/sec
rounds=11: ~5  hashes/sec
rounds=12: 2-3 hashes/sec
rounds=13: ~1 sec/hash
rounds=14: ~1.5 sec/hash
rounds=15: ~3 sec/hash
rounds=25: ~1 hour/hash
rounds=31: 2-3 days/hash
```

BCrypt - How It Works

```
bcrypt.hashSync(req.body.password,8)
```

returns

```
$2a$08$.jEA1FSMXOSGQhLXrKkN9.k9gyad8xN6r76YqOIzoA318fhoqUp7a
```

To Compare

```
bcrypt.compareSync(myPlaintextPassword, hash);
```

Objectives

- Explain Identification and Authentication.
- Explain why passwords are terrible.
- Explain what makes a good hashing algorithm.
- Store a password *securely* using BCrypt.

Questions?