

CSCI262

System Security

(S0)

Subject Admin Introduction

Contact details

Dr. Dung Duong

hduong@uow.edu.au

If you email me it makes it easier if you include the subject and topic in the subject line: For example: SIM-CSCI262: A1.

- This way I can tell if an email is about almost due assessment or similar important matters.
- While I generally reply to emails within a couple of working days there will be times when other activities will take priority.
- **If possible, use your university account for email.**
 - UOW preferably, SIM if not.

- The delivery method for lectures will be power point slides.
- You will be able to find material for the subject on the subjects eLearning site reachable through the UOW website

<http://www.uow.edu.au/student/index.html>

- The slides will be provided in pdf.

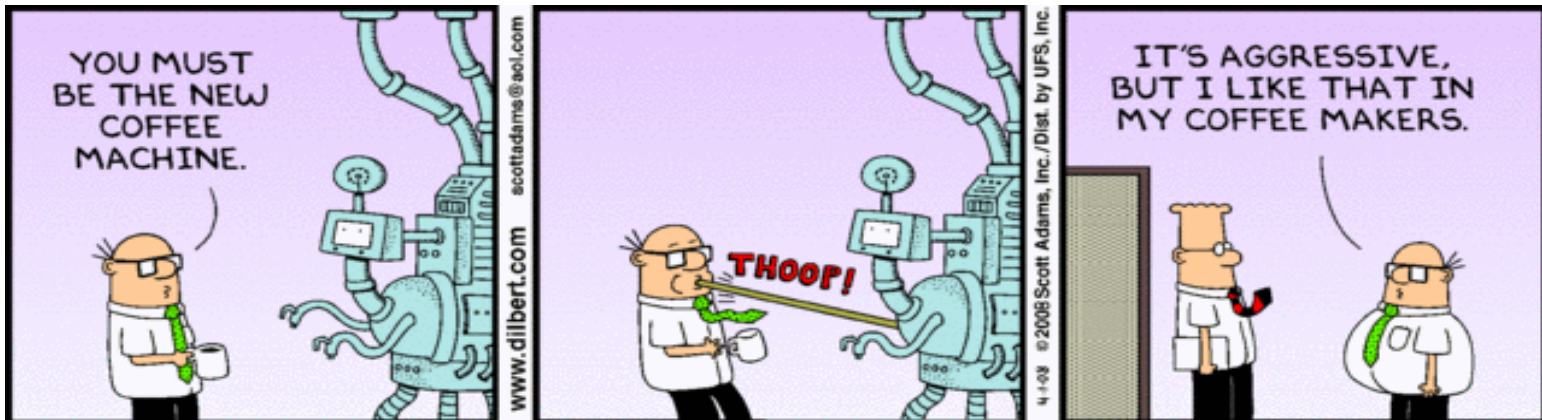
- Attending lectures is a good idea for success in this subject.
 - The textbook doesn't cover all the content, nor will the lecture notes themselves necessarily make sense without listening to explanations.
- Tutorial/Lab material supplements the lecture material.
 - Attending tutorials/labs is necessary to completely understand that material. Some solutions may only be available during the tutorial/lab.
- The final exam will be based on contents from the lectures **and** tutorials/labs.

Assessment: Passing this subject.

- 3 assignments: 14%, 12%, 14%.
 - The assignments involve some programming and some other activities:
 - Generally Java, C++ or C will be acceptable.
- The exam is worth 60%.
 - You **must** get at least **45%** for the exam (this means at least **27/60**) to pass.
 - Not meeting this requirement, and obtaining enough marks to otherwise pass, will result in a TF grade.
 - That's a technical fail.

The exam ...

- There are exams from previous years available through the UOW library website.
 - Go to <https://ereadingsprd.uow.edu.au/> and enter CSCI262.
 - Exam scope is slightly larger at Wollongong
 - It's unlikely anyone will start studying for the exam this week but don't leave it until the last couple of weeks.



Textbook and references

■ The textbook:

- William Stallings and Lawrie Brown, *Computer Security : Principles and Practice*, 4th edition, Pearson.

■ References:

- Matt Bishop, *Computer Security: Art and Science*, Addison-Wesley, 2002.
- William Stallings, *Network Security Essentials: Applications and Standards*, 3rd Edition, Pearson Education pressed, 2007.
- Dieter Gollmann, *Computer Security*, Wiley, 2006.
- Ross Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*, 2nd edition, 2008.
 - <http://www.cl.cam.ac.uk/~rja14/book.html>

The eLearning site

- Check the eLearning site for this subject regularly!
 - Any change to the subject will be announced on the eLearning site.
 - Any information posted to the eLearning site is deemed to have been notified to all students.
- Check SOLS mail too, since urgent updates are likely to be sent there.

Extensions etc.

- If you require additional time to complete an assignment you must submit claims for extensions electronically via SOLS, ***before the DUE date.***
- You may be granted an extension if your circumstances warrant it.
- Of course, if you are in hospital for the last week or similar, and cannot get in contact I will understand.

<http://www.uow.edu.au/students/sols>

A word on Plagiarism and similar concerns ...

- The university handbook states:
 "Plagiarism means using the ideas of someone else without giving them proper credit."
- I strongly suggest you read that website about what is considered plagiarism.
- There are two primary concerns for us:
 - Students copying directly from sources, or copying without appropriate referencing.
 - Students copying each other.
 - You can discuss ideas but need to use your own words!
 - With code and mathematical solutions you need to work fairly independently.

CSCI262

System Security

(S1)

Subject Content Introduction

So ... what is this subject about?

- A subject in computer system security:
 - Basic issues and solutions in computer system security.
 - User awareness.
 - Access control and authentication.
 - Malware.
 - etc.
 - Theory and practice.
 - A range of systems are looked at:
 - Operating systems.
 - Code security.
 - Web security.
 - Database Security.

What is this subject not about?

- Cryptography:
 - There are some elements of cryptography that will be discussed, but cryptography is mostly in CSCI361.
- Mathematics:
 - There is some mathematics and statistics, but they certainly aren't the primary focus.
- Code syntax:
 - There is a fair bit on programming but this subject is not about teaching you syntax of specific languages.

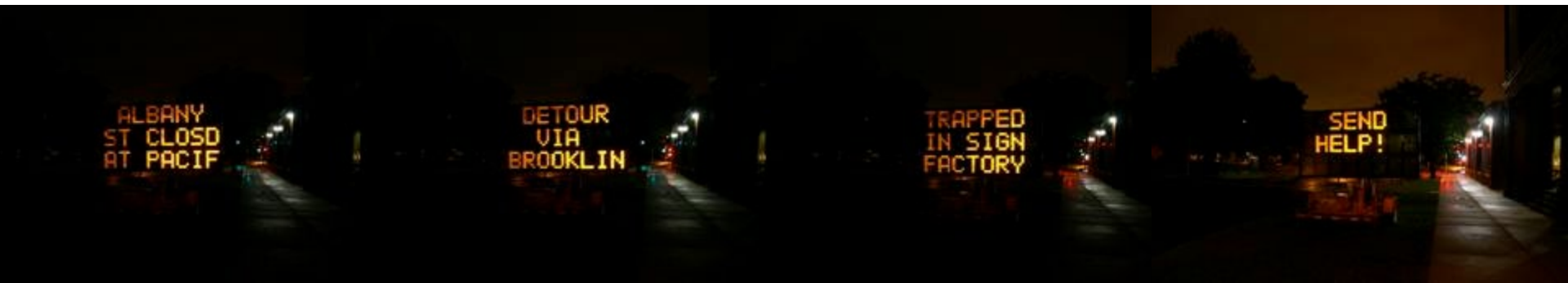
Motivation: Why does security matter?

- Why does security matter generally?
- What about information security?
 - Electronic commerce.
 - Internet banking.
 - Protection of intellectual property.
 - “Privacy”.

- It is difficult to accurately calculate the cost of attacks against information security, and more broadly the cost of protecting against such attacks.
 - In direct monetary terms it is billions of dollars, each year.
 - There are also less direct results though;
 - Such as harder to predict loss of future revenue and damaged reputations.
- Yearly CSI/FBI reports have information on the costs, countermeasures, and the current significant problems.

Some examples...

- You will have heard of people breaking into computers and stealing information.
 - And you have almost certainly had first hand experience with malware, phishing and spam.
 - Ransomware attacks
- But (information) security faces less obvious concerns too ...
 - <https://www.wired.com/2009/02/austin-road-sig/>
 - Another very similar one ...
 - http://hacks.mit.edu/Hacks/by_year/2008/sign_factory/



The cost of information ...

Hackers' discount – stolen card details for 8 cents

Conrad Walters

THE theft of personal information by hackers is so prevalent – and efficient – that stolen credit card details now sell for as little as 8 cents a card, a report by one of the world's biggest computer security companies says.

The global report, to be released today, has been compiled by monitoring nearly 250,000 online sensors and deploying more than 2.5 million decoy email accounts.

While the wider economy has suffered its worst period in generations, the online underground economy has "consolidated and matured", the *Global Internet Security Threat Report* by the anti-virus vendor Symantec says.

Fears about online theft have been high since the news that

financial fraud in Australia has leapt to a record \$161.7 million a year and concerns that the Conficker virus was poised to wreak financial havoc on computers around the world.

The price of 8 cents a stolen credit card has been made possible because hackers have been able to lower their margins through greater efficiency and speed in deploying their traps for consumers, a Symantec spokesman, David Dzienciol, said.

The low price is available to criminals buying common cards in bulk – up to 5000 at a time.

When Symantec released a similar report a year ago the lowest price was 55 cents a card.

Conversely, increased proficiency by criminals has allowed them to charge a premium for cards that are harder to secure, carry a higher

GOING CHEAP

RANK ITEM

- 1 Credit card information
- 2 Bank account credentials
- 3 Email login details
- 4 Email accounts
- 5 Anonymous computers for hire
- 6 Full identities
- 7 Spam software
- 8 Money laundering

- 9 Malicious software writing
- 10 Other

SOURCE: SYMANTEC

GOODS AND SERVICES FOR SALE ONLINE

PRICE RANGE

- | |
|--|
| \$0.08 - \$42 |
| \$13.80 - \$1380 |
| \$0.14 - \$140 |
| \$0.46 - \$140 per megabyte |
| \$0.22 - \$28 |
| \$0.97 - \$83 |
| \$2.76 - \$55 |
| 8% - 50% or flat rate of \$280 - \$2800 per item |
| \$2.80 - \$28 |
| \$4 - \$56 a week for hosting, \$3 - \$28 design |

credit limit or come with personal details about the legitimate owner.

"Credit card information ad-

vertised on the underground economy consists of the credit card number, address, phone number, CVV2 number (the se-

curity number on the back of the card) and PIN," the report says.

For those cards, criminals can pay as much as \$41.40, a substantial rise on the maximum the previous year of \$27.60.

The goods and services available through the underground economy include bank account details. The report says these have become increasingly popular because they give thieves access to all a consumer's money, not only the maximum allowed on a credit card.

Log-in details for email accounts are also popular on the underground market. These can be used to send spam or to impersonate friends and family. Once an account has been compromised, criminals gain access to other services by having password reminders sent to the hacked account.

CUSTOMERS ARE
COMPLAINING THAT
WE SOLD THEIR
PERSONAL DATA.



AND APPARENTLY
ALL OF THE BUYERS
WERE IDENTITY
THIEVES.



THAT'S IMPOSSIBLE.
WE CHECKED EVERY
BUYER'S IDENT...



OH.

Dilbert.com DilbertCartoonist@gmail.com

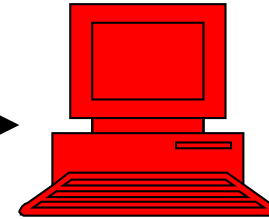
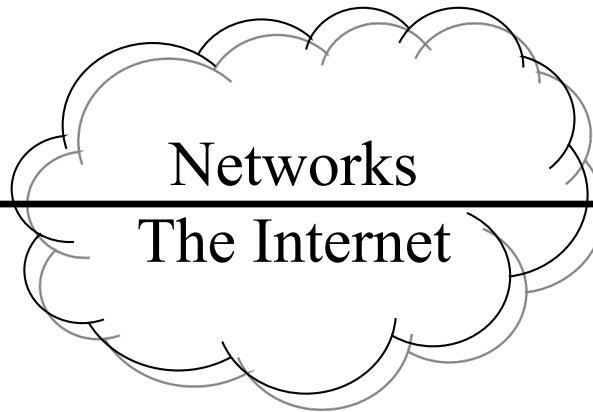
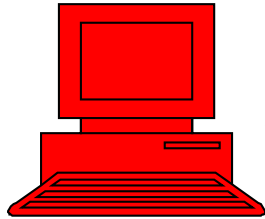
10-11-10 ©2010 Scott Adams, Inc./Dist. by UFS, Inc.

Sydney Morning Herald
April 15th 2009.

Dilbert: 14-Oct-2010

Network exploitation

- One of the most expensive computer crimes is **denial of service**.
 - This attack usually involves distributed systems and remote access.
- **Viruses, worms and trojans** are the most common form of electronic attacks.
 - They are prevalent because of networking.
- Problems such as **Intrusion** and **XSS** also largely rely on networking.
- We will look at most of these in this subject.



Networks
The Internet

Databases:

Access control
Inference
Concurrency
SQL injection

Environments:

Networks
Computers
OS: Windows
Web Servers
Databases

Web Servers:

Apache
JavaScript
CGI
PHP

Attacks:

Buffer overrun
DoS & DDoS
Malicious Code
Phishing

Fundamentals:

User Authentication
and
Access control

Defence:

Auditing
Client puzzles
Good coding practices
IDS
Firewalls

SYSTEM SECURITY

Tentative lecture schedule

Introduction.

User authentication.

Access control: General, Windows, Unix/Linux.

Denial of Service attacks, protection methods.

Buffer overflows.

Mobile code security.

Malware.

Intrusion detection.

Firewalls.

Database Security.

Revision.

Describe security properties ...

- Some properties are critical.
 - **C***onfidentiality*: Assets should be inaccessible to unauthorised parties.
 - **I***ntegrity*: Assets should be unmodifiable or unforgeable, without detection, by unauthorised parties.
 - **A***vailability*: Assets should be available to authorised people.
- In this subject (CSCI262) we look at availability, and how we distinguish between unauthorised and authorised entities.

Authorised versus unauthorised

- Authorised persons are allowed to do something, unauthorised are not.
- How do we know someone (or something) is authorised?
- Two aspects:
 - Authentication is needed so we (the system) know who the “someone” is.
 - Access control is needed to determine what that identified someone is allowed to do.
- In most contexts we have authentication and access control.

We can summarise security as the control of access to resources.

CSCI262

System Security

(S2a)

User Authentication I

Outline

- Authentication: What is it?
- Bases for authentication.
- False positives and false negatives.
- Password based authentication.
 - Problems with passwords.
 - Attacking password systems.
- Hashing.
- Salting.


Authentication: What is it?

- Authentication is the binding of an identity to a subject.
- A user or entity is often required to authenticate itself to a computer system.
 - When (if) you use Internet banking you need to be authenticated by the bank site.
 - When using email you need to provide a password linked to your username or account name, that is to your identity, before you access your email.

Bases for authentication

- A subject, (a user or an entity), must provide information to enable the computer system to confirm its identity.
- This “information” could be one or a combination of the following:
 - **What the subject knows...**
 - A password or a secret.
 - **What the subject has ...**
 - A card or a badge.
 - **What the subject is ...**
 - Biometrics such as fingerprints or retinal characteristics.
 - **Where the entity is ...**
 - In front of a particular terminal.

We focus on
passwords!



- The first three are sometimes summarised as:

"Something you have, something you know, or something you are."

- Garfinkel says it a little differently 😊

"Something you had once, something you've forgotten, or something you once were."

Simson Garfinkel is mainly known as an author of computing and security texts.

Credentials ...

■ See rfc4949: Internet Glossary v2.

\$ credential

1. (I) /authentication/ "identifier credential": A data object that is a portable representation of the association between an identifier and a unit of authentication information, and that can be presented for use in verifying an identity claimed by an entity that attempts to access a system. Example: X.509 public-key certificate. (See: anonymous credential.)

2. (I) /access control/ "authorization credential": A data object that is a portable representation of the association between an identifier and one or more access authorizations, and that can be presented for use in verifying those authorizations for an entity that attempts such access. Example: X.509 attribute certificate. (See: capability token, ticket.)

3. (D) /OSIRM/ "Data that is transferred to establish the claimed identity of an entity." [I7498-2]

Deprecated Definition: IDOCs SHOULD NOT use the term with definition 3. As explained in the tutorial below, an authentication process can involve the transfer of multiple data objects, and not all of those are credentials.

4. (D) /U.S. Government/ "An object that is verified when presented to the verifier in an authentication transaction."
[M0404]

Deprecated Definition: IDOCs SHOULD NOT use the term with definition 4; it mixes concepts in a potentially misleading way. For example, in an authentication process, it is the identity that is "verified", not the credential; the credential is "validated". (See: validate vs. verify.)

Tutorial: In general English, "credentials" are evidence or testimonials that (a) support a claim of identity or authorization and (b) usually are intended to be used more than once (i.e., a credential's life is long compared to the time needed for one use). Some examples are a policeman's badge, an automobile driver's license, and a national passport. An authentication or access control process that uses a badge, license, or passport is outwardly simple: the holder just shows the thing.

The problem with adopting this term in Internet security is that an automated process for authentication or access control usually requires multiple steps using multiple data objects, and it might not be immediately obvious which of those objects should get the name "credential".

False positives and negatives...

- One concern in any situation where we attempt to distinguish between authorised and unauthorised entities, is the likelihood of getting a result which is wrong.
- This is relevant in the context of intrusion detection systems, malware detection, spam, and also in the context of authentication.

- A false positive is when we make a match but “shouldn’t have”.
 - Sometimes when installing a game it will be flagged as a key logger, since some of the behaviour is consistent with a key logger.
- We have to be a little careful thinking about this for authentication.
 - There is the false acceptance rate (FAR).
 - This will be the proportion of authentication attempts resulting in false acceptances.
- Notice that the false positive effects are quite different in the context of “raising an alarm” from the context of “making a match”.

- A false negative is when we don't make a match but “should have”.
 - We install a game we have downloaded, scan it and find nothing.
 - But actually it has a virus in it.
- Again, we have this reverse thinking for authentication.
 - There is the false rejection rate (FRR).
 - This will be the proportion of authentication attempts resulting in false rejections.

Passwords ...

- We are going to focus on passwords, and I assume you know what they are.
- So, are the FAR and FRR appropriate?
- We'll see later that FAR might make sense once we get to storing transformed passwords rather than plain visible passwords.

Passwords

- One simple and common method of user authentication is the password-based method.
 - It is also possibly the worst but its use is widespread! ☹
 - Sasse states that it is hard to think of a worse authentication mechanism, given what we know about human memory.
- Passwords involve authentication on the basis of what an entity knows.

"Something you have, something you know, or something you are."

How do password systems work?

- Account registration:
 - Could be by an administrator.
 - Passwords should be changed.
 - Double passwords.
 - Confirmation through an email...
- Authentication.
- Reset...

■ Password Authentication:

- The user supplies an identity.
- The user supplies a password.
 - This isn't necessarily what is sent to the “server”, that may be some function of the password.
- The server checks the supplied information.
- If the password information is associated with the user, the user's identity is authenticated; otherwise, the password is rejected.

Threats against password systems

- Here go some common threats against password systems:
 - Password guessing.
 - Password exposure.
 - Trojan login programs.
 - Poor passwords.
 - Common attacks:
 - Dictionary attacks.
 - Brute force attacks.
 - Hybrid attacks.
 - On-line or off-line:
 - Compromise of the password file.
- We are going to examine these now.

Password guessing

- It is pretty much always possible to *attempt to guess* a password on-line.
- You can try and guess the password of a user on Banshee simply by trying to login as that user.
 - (You shouldn't try to do this) 😊
 - If you get logged in you have something acceptable as a password. Depending on the authentication mechanism it may not actually be the password.

Password exposure

- An “eavesdropper” may see the password when it is typed.
 - Typing very slowly isn’t a good move.
- Some users write their passwords down, even next to their computer.
 - This is often not a good idea!
- Some users pass ***their*** password to others.
 - Even if you appropriately protect your password, the other person may not.
 - Trust assumptions are critical in security.

Login Trojan Horses

- These are programs that produce an apparently genuine login screen.
- The user logs in, but the program captures the password and stores it along with the username for the malicious owner of the Login Trojan Horse.
- The program can subsequently pass the information to the genuine login program so the user doesn't realize something is wrong.
 - The protection against this lies in not installing it in the first place 😊

Poor passwords

- Passwords must be remembered by its owner, so users usually choose simple passwords.
 - Often as simple as possible.
 - Many systems have significant restrictions on the passwords allowed.



Dilbert: 05-Dec-2004

Poor passwords → Dictionary attacks

- The requirements could be only a length restriction, so even if passwords meet the requirements they may be a dictionary word.
 - A dictionary attack exploits this.
- Dictionaries of common words can be used as sets of passwords to try.
- The attacker steps through the words in a dictionary and tries them as passwords.
- This dictionary attack may not succeed but is quite fast.

Tailored dictionary attacks

- It is possible to be more specific than a complete English dictionary, or targeted from another source.
 - A dictionary doesn't have to just be real English words.
- For example, users may like cars or motorbikes, and a suitable dictionary could be a list of car or motorbike brands.
 - Or sports teams or players names or ...
- Users may use even more personal information for passwords:
 - Birthdates, family names, pet names.

Brute force

- All password systems are vulnerable to somebody guessing the correct password.
- A brute force attack involves trying every possible password, and more generally every possible solution.
- Unlike a dictionary attack, brute force always works...
 - Eventually!
 - Sort of anyway. In some context we may not recognise the answer ☹
 - The important factor is that this guessing is unlikely within the lifetime of the password.
 - Changing passwords making a moving target which is “harder to hit”.
- With a brute force attack, you start with the letter a, then try aa, ab, ac, and so on until zz; then you try aaa, aab, aac, and so on.

Choosing secure passwords

- Time in seconds to definitely test the correct password is N/R , where ...

N: Size of the set of possible passwords.

R: Number of passwords that can be tested in a second.

- The expected time (or expectation value of the associated random variable) is $\frac{1}{2}$ of N/R .

- Consider a randomly generated password of length 8, from a character set of the lower case letters a-z.
 - The size of the password space is $26^8 = 208827064576 = \sim 2 \cdot 10^{11}$.
 - If we can test 1000 passwords a second, testing the whole space will take ~ 6.6 years.
- A password chosen as above is secure against that kind of attack, but the password is fairly hard to remember. ☹️
- And going to an arbitrary password over the 10 digits and upper and lower case \rightarrow factor of 1000 increase.

Password entropy

- Entropy is to do with the information content, and randomness, and uncertainty.
 - In this context the uncertainty of someone else regarding your password is a helpful way to think!
- Entropy is often measured in bits.
- If there are two equally likely options we have 1 bit of entropy, four equally likely options we have 2 bits of entropy and so on...
 - How many decision (yes/no) questions about your password would someone need to have you answer?
- The entropy for N equally likely options is $\log_2 N$.

Protective mechanisms

- Keep track of incorrect password attempts:
 - Limit the number of account/passwords guesses per connect attempt.
 - Or lock the account when a threshold is exceeded.
 - Although the attacker can use this to perform a DOS attack. ☹
 - Or raise an alarm and try to trace the intruder.
- Slowly process passwords, it doesn't make much difference to a legitimate user, but it makes a lot to the processing speed of an attacker.

Trying to improve passwords

- Using pronounceable passwords makes remembering passwords easier. But ...
 - ... this reduces the number of possible passwords, since the number of vowels is likely to be fairly high. Every third character could be a vowel.
- Using a "pass-phrase" is another alternative.
- Dictionaries can be used for either of these scenarios so ...
 - ... we could use pass-phrases with intentional misspellings, odd capitalizations and symbol replacements. Or insert some numbers.

Hybrid attacks

- But these are vulnerable to ...
- ... hybrid attacks.
- Basically we use a dictionary as a basis but take variants on each of the words tested.
 - We might replace each lower case “L” with 1, or “O” with 0, and so on.
- The hybrid attack falls between the dictionary and the brute force attack, in the time consumed, the number of passwords tried, and, therefore, in the (naïve) chance of success.

Rules for password systems

- These are examples of rules, the application and details should be context dependent.
 - Change passwords every 45 days.
 - Minimum length of eight (or higher) characters.
 - Must contain at least one alpha, one number, and one special character.
 - Alphabetical, numerical, and special characters must be mixed up.
 - For example, fg#g3s^hs5gw is good, abdheus#7 is not.
 - Cannot contain dictionary words.
 - Cannot reuse any of the previous five passwords.
 - After five failed logon attempts, password is locked for several hours.

How can you remember?

- You can choose a phrase and take the first letter from each word as your password.
- Choosing a well-known phrase is not such a good idea → phrase dictionary attack.
- A rolling stone gathers no moss.

Arsgnm

- My cat Boris has a long tail and 16 teeth.

McBhalta16t

“Online” versus “Offline” guessing

- Online (“live”) guessing will usually face restrictions on the number of attempts.
- Offline attacks do not usually face this problem, and may take place without the awareness of the password owner or system administrator.
- Offline attacks can occur if an intruder accesses the password file on a computer.
- Or if the transmission of a password is intercepted, say in logging into a website on the Internet.
 - The interception may capture the password directly, which means the communication wasn’t adequately protected, or it may be some function of the password, possibly an encrypted or a hashed version.
 - Communication security and cryptography is looked at more in CSCI361 and CSCI368.

- The distinction between “online” and “offline” isn’t really that important anymore.
- What is important?
 - The issue that really matters is whether the number of “guesses” is restricted or not.
 - The distinction completely changes the way in which attackers are likely to operate.
- If you can guess without restriction it is probably worthwhile trying, at least a dictionary attack.
 - If you cannot guess without restriction then another approach, probably some form of social engineering, is probably more useful.

Brute force and the GPU ...

- Graphics processing units have been used for improving significantly the speed of cryptographic brute force attacks.
- These units are designed for massively parallel operations and have been applied to many tasks.

Protecting passwords

- Password lists in a system must be well protected.
- In the UNIX operating system, users passwords are not stored.
 - Hashes of the passwords are stored.
 - Such information is vulnerable to off-line password guessing, so this file must be protected.
 - With appropriately hashed information it is computationally infeasible to find the associated password.

What is hashing?

- Hashing is a procedure often used in providing integrity for messages.
 - Integrity is about providing checks that a message hasn't changed (in transmission or storage).
 - Hashing is also used in for indexing, in compilers and elsewhere, and for other purposes, but we are going to focus on cryptographic hash functions.
- One common integrity mechanism is the digital signature (see CSCI361), a function of the message signed and the signer.
 - Hashing is used to reduce the computational overhead of digital signatures.
- Informally the hash of a message is a “fixed length fingerprint” of the block of data.

Hash Functions

- A hash function transforms data from an arbitrary length into a fixed short length.
- Let $H()$ be a hash function and X be a message.
 - The hash value of X can be efficiently computed as $h = H(X)$.
 - The efficiency matters if hash functions are going to be of practical use.
 - The output is referred to as the hash value or message digest.

- Two examples:

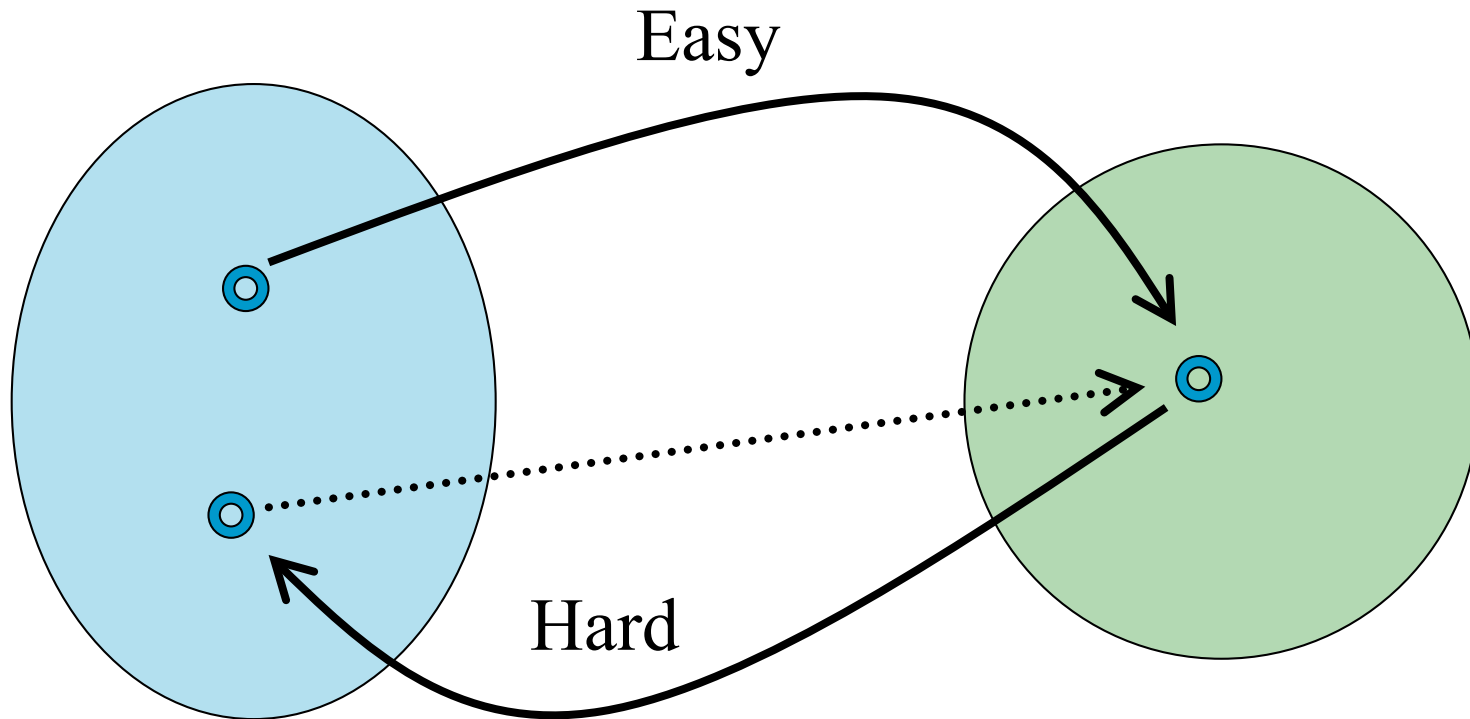
1. H is a modulo function, so calculates $x \bmod 7$ for example.
2. H takes the last 128 bits of a binary representation of X , unless the message is less than 128 bits, where it takes the whole message followed by enough bits to make the message digest 128 bits in length.

- These are little use as hash functions for cryptography, we need some additional properties.

Cryptographic Hash Functions

- We already have the first properties:
 - Our hash function can be applied to input of any size and produces a fixed size output.
- There are other properties, we will just look at two here...
- One-way or pre-image resistant:
 - It is computationally infeasible that for a given message digest Y , we can find an X such that $H(X) = Y$.
- Collision-resistant:
 - A hash function H is called collision-resistant if it is computationally infeasible to find messages X and X' , with $X \neq X'$, such that $H(X) = H(X')$.
- The examples on the previous page don't satisfy these.

Cryptographic Hashing



One-way or pre-image resistant:
Collision-resistant:

MD5 and SHA-1

- MD5 is the most well-known and one of the most popular hash algorithms ...
 - It produces a 128-bit message digest.
- Secure Hash Algorithm (SHA-1): A hash algorithm proposed and adopted by NIST, and to be used with DSA standard. It produces a 160 bit message digest and uses the design approach used in MD5.
- Both are broken, at least with respect to collisions.
- They are still both used.
- You don't always need collision resistance and it is (a lot) harder to be collision resistant than pre-image resistant.
 - More on this in CSCI361.

Back to the password file

- You might recall that we mentioned earlier that passwords are not directly stored in UNIX, instead the hash of them is stored.
- The password file is encrypted using a password known by the system.
- The password is inputted when the system is booted.
- Actually it is a little more complicated, we wouldn't typically directly store the hash either. 😊
 - At least not anymore.
 - We use something called salting.

Password Salting

- The “salt” is a value that is randomly generated.
- The hash of the combination of the salt and the password, is stored.
- The salt is stored somewhere too.

user ID	salt value	password hash
Alice	3487	hash(3487 password_Alice)
Bob	8254	hash(8254 password_Bob)
Oscar	1098	hash(1098 password_Oscar)

- Using salt slows down some attackers ...
- If the whole password file is disclosed, the intruder can compute the hash of a password and compare it against all hashes.
- This attack isn't possible against salted systems, we can only check against one-user at a time.
 - Even if the salts are known it is non-trivial to link the passwords of users, either in the same system or between systems.
 - Without salt this linking is possible because two users with the same password would have the same stored password hash.
 - An attacker may only be interested in one account though.

- With these hashed values we (someone anyway) might enter the wrong password but have it accepted 😊
- This is a false positive and will contribute to the false acceptance rate.
 - It's very unlikely though...

How Does UNIX "Encrypt" Passwords?

- UNIX uses an hashing algorithm called crypt to protect its passwords.
 - This isn't actually the same as simply running crypt from the command line in UNIX, that is an encryption algorithm.
- The protection is through the one-way transformation of the password by the one-way hash function.
- There is no way to “decrypt” a password that has been “encrypted” with crypt, but there doesn't need to be.

- Each “encrypted” password is a 13-character stored string constructed by “crypting” the password concatenated with the salt.
- As mentioned earlier the salt has the advantage of (probably) giving different encrypted passwords to users with the same actual password.
 - The salts must be random, and from a large enough space that the chance of two users having the same salt is low.
- For example, if create two users with a password of yellow, the stored passwords could be:
 XcRqBAu2wfRQo
 5pj oJnbeVEUbw
- They are different!

- To illustrate how UNIX encryption works, consider the following small Perl program that takes a plain text word and generates the encrypted/hashed password:

```
#!/usr/bin/perl
srand( time() ^ ($$ + ($$ << 15)) );
@salts=('46' .. '57', '65' .. '90', '97' .. '122');
print "Enter a password: ";
chop ($password=<STDIN>);
print "The results are: ", crypt ($password, ( chr
($salts[int(rand $#salts+1)] ) .
chr ($salts[int(rand $#salts+1) ] ))), "\n";
```

Running 2-crypt.pl

```
$ perl 2-crypt.pl
```

```
Enter a password: uowcsci262
```

```
The results are: W9b2.PWtBMEbs
```

```
$ perl 2-crypt.pl
```

```
Enter a password: uowcsci262
```

```
The results are: WVZmQRjsGZGg6
```

Newer UNIX & Linux

- With newer versions of UNIX, you can employ other encryption/confidentiality options rather than crypt.
 - Such as MD5, which provides stronger hashing.

From https://en.wikipedia.org/wiki/Cryptographic_hash_function
See also <https://www.cryptolux.org/images/0/0d/Argon2.pdf>

- In 2013 a [Password Hashing Competition](#) was announced to choose a new, standard algorithm for password hashing.^[7] The winner, selected in July 2015, was a new key stretching algorithm, [argon2](#). In June 2017, NIST issued a new revision of their digital authentication guidelines, NIST SP 800-63B-3,^{[8]:5.1.1.1} stating that:
"Verifiers SHALL store memorized secrets [i.e. passwords] in a form that is resistant to offline attacks. Memorized secrets SHALL be salted and hashed using a suitable one-way [key derivation function](#)."
- Key-stretching is what bcrypt does.
 - I guess Argon2 does it better 😊

Password storage in UNIX: Where?

- Early versions of UNIX contained a file `/etc/passwd`, which stored all of the user IDs and protected/transformed passwords in the same file. This file was a text file and contained the user ID, encrypted password, home directory, and default shell. The following is a sample `passwd` file:
- ```
root:6T1E6qZ2Q3QQ2:0:1:Super-User:/:/sbin/sh
john:.D532YrN12G8c:1002:10::/usr/john:/bin/sh
mike:WD.ADWz99Cjjc:1003:10::/usr/mike:/bin/sh
. . .
cathy:BYQpdSZZv3gOo:1010:10::/usr/cathy:/bin/sh
frank:bY5CQKumRmv2g:1011:10::/usr/frank:/bin/sh
tom:zYrxJGVGJzQL.:1012:10::/usr/tom:/bin/sh
karen:OZFGkH258h8yg:1013:10::/usr/karen:/bin/sh
```

# The general format for the passwd file

- **Username:passwd:UID:GID:full\_name:home directory:shell**
- **Username:** Stores the username of whom the account belongs to.
- **Passwd:** Stores the user's transformed password.
  - If shadow files are used, an x appears in this location.
- **UID:** The user ID or the user identification number, generally chosen by the system.
- **GID:** The group ID or group identification number, which reflects the native group (base group of membership).
- **Full name:** This field usually contains the user's full names but is not mandatory.
- **Home Directory:** Stores the location of the user's home directory.
- **Shell:** Stores the user's default shell, which is what runs when the user first logs onto the system.

# Shadow Files

- A solution to the readability problem.
- UNIX does, and has for a long time, split the passwd file information into two files.
- The passwd file still exists and contains everything except the protected passwords.
- A second file, shadow, was created.
  - This contains the transformed password and is only accessible to the root user.
- This information is stored centrally.
  - /etc/passwd
  - /etc/shadow

- Using "shadow passwords" is the preferred way of storing password hashes.
- You shouldn't have any system that still stores password hashes in /etc/passwd. Update if you are!
- Consider the following pair of /etc/passwd and /etc/shadow files:

```
root:x:0:1:Super-User:/:/sbin/sh
eric:x:1001:10::/usr/eric:/bin/sh
John:x:1002:10::/usr/john:/bin/sh
mike:x:1003:10::/usr/john:/bin/sh
...
tim:x:1009:10::/usr/tim:/bin/sh
cathy:x:1010:10::/usr/cathy:/bin/sh
frank:x:1011:10::/usr/frank:/bin/sh
tom:x:1012:10::/usr/tom:/bin/sh
karen:x:1013:10::/usr/karen:/bin/sh
```

```
root:6T1E6qZ2Q3QQ2:6445::::::
eric:T9ZsVMlmal6eA:::::::
John:.D532YrN12G8c:::::::
mike:WD.ADWz99Cjjc:::::::
...
tim:sXu5NbSPLNEAI:::::::
cathy:BYQpdSZZv3gOo:::::::
frank:bY5CQKumRmv2g:::::::
tom:zYrxJGVGJzQL:::::::
karen:OZFGkH258h8yg:::::::
```

# Shadow files: The fields

## **username:passwd:last:min:max:warning:expire:disable**

- **username:** The user's name of the account. There should be a corresponding line in the passwd file with the same username.
- **passwd:** Contains the transformed password.
- Only the first two fields are mandatory.
- **last:** Contains the date of the last password change.
- **min:** The minimum number of days until the password can be changed.
- **max:** The maximum number of days until the password must be changed.
- **warning:** The number of days that the user is warned that the password must change.
- **expire:** The number of days in which the password expires and the account is disabled.
- **disable:** The number of days since the account has been disabled.

# How safe are shadowed systems?

- Using shadow files is safer than before but ...
- To break into a system using a password, an attacker needs a valid user ID and a password.
  - Valid user ID's can be obtained from `/etc/passwd`.
  - A password guessing attack could then be launched.
  - It is more difficult to detect single attempts against multiple users than multiple attempts against a single user.
- Although shadow files require root access, there were attacks that can be used to acquire a copy of the shadow file without obtaining root access directly.
  - For example `imapd` (a mail related server) and `telnet` were, at one time, both guilty of dumping core on occasion complete with the shadow file in the core where it was user-readable.
  - It is possible to recover information from the core.

exploits , vulnerabilities , articles , Solaris FTP Core Dump Shadow Password Recovery Vulnerability

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Title          | Solaris FTP Core Dump Shadow Password Recovery Vulnerability                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Published      | 2001-04-17-12:00AM                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Updated        | 2001-04-17-03:10PM                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Class          | Configuration Error                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| CVE            | <a href="#">CAN-2001-0421</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Remote         | No                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Local          | Yes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Credit         | This vulnerability was announced to Bugtraq by Warning3 <warning3@mail.com> on April 17, 2001.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Vulnerable     | Sun Solaris 2.6                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Not Vulnerable | Sun Solaris 8.0<br>Sun Solaris 7.0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Code           | <pre> [root@ /usr/sbin]&gt; telnet localhost 21 Trying 127.0.0.1... Connected to localhost. Escape character is '^]'. 220 sun26 FTP server (SunOS 5.6) ready. user warning3 331 Password required for warning3. &lt;-- a valid username pass blahblah &lt;--- a wrong password 530 Login incorrect. CWD ~ 530 Please login with USER and PASS. Connection closed by foreign host. [root@ /usr/sbin]&gt; ls -l /core -rw-r--r-- 1 root root 284304 Apr 16 10:20 /core [root@ /usr/sbin]&gt; strings /core more [...snip...] lp:NP:6445:~~~~~ P:64 eH:~~~ uucp:NP:6445:~ </pre> |
| TXT            | <a href="#">TXT</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |



strings 😊



# Strings is handy ☺

- The utility strings finds printable strings within a binary or object file.
- It doesn't search in a particularly clever way and the output from strings could be processed more.
- Look at applying it to an executable.
- Explicit quoted statements will always appear.
  - It means, for example, that you can get a list of all possible fixed outputs.

```
#include<iostream>
using namespace std;
```

```
int main()
{
 string
 password="as#d3fFkbfh";
 int x=23782;

 cout << "This is a test run"
 << endl;

 system("sleep 2");
}
```

2a\_strings-test.cpp

■ If I compile and run strings on the executable I get:

```
$ strings a.out
as#d3fFkbfh
This is a test run
sleep 2
```

We can use strings -a or strings -n to get strings from everywhere, or of different lengths.

- You should get the impression that hard-coding passwords is something you should never ever do!!!!
- It is possible to do something like hard-code it in encrypted or hashed form, and decrypt it or hash the password before use.
  - But if the attacker knows the encryption/decryption algorithm you are using, and we generally have to assume they will, then they immediately have something to launch a dictionary or brute force attack against.

- Or, even more fun from the point of view of an attacker, the executable can be edited to replace the hashed string with something else.
- Assume you know some program with a login uses MD5 to protect the password, and strings reveals a 32 hex character string X.
  - Choose a password p.
  - Apply MD5 to p and get the output  $Y=H(p)$ .
  - Replace X with Y.
  - Login using p. 😊

# Rainbow tables ...

- Sometimes it is worthwhile, for an attacker, to do some pre-computation to speed up an attack.
  - Rainbow tables are an example.
- The basic idea is to have a lookup table of passwords, maybe salted, and the corresponding hash value...
- Having obtained a particular hash value we look it up in the rainbow table.
- Last year students were asked to implement a rainbow table, but these were extended versions that carry out more pre-computation to give a smaller lookup table.
  - This is done using hashing and reduction functions.

- Reduction functions (R) map from the hash output space back into whatever password space is considered appropriate.
- **Table construction:** Take an original password and hash it, then use a reduction function to take you to another password, which you then hash, then reduce, then hash and so on.
  - These sequences of  $H R H R \dots$  are sometimes referred to as hash chains.
  - The saving comes in only storing the first and last elements in a particular sequence.

- **Table lookup:** If you have a hash value you can check it in the table.
  - If it isn't in the table you reduce and hash it until it does appear in the table.
  - Having found the value in the table you know the password to start from, effectively a row of the table.
  - You can hash, then reduce and hash, until your original hash value appears.
  - The password immediately preceding that is the one you want.

# One-time passwords

- With a one-time password system the user and the system have “a list” of valid passwords such that each one is valid only once.
- Provided the passwords are not obviously correlated, this system is immune to eavesdropping.
- An observed password leaks no information about the other passwords.
- **Problems:** The number of passwords to be shared (established) and stored.



# Why are these problems?

- The passwords need to be established requiring significant initial costs.
  - More in CSCI368 on key establishment and cryptographic authentication.
- From the point of view of the server they need to store more information.
- From the point of view of the user they are more likely to write down passwords and be less careful in choosing them.
  - Users cannot rely on repeated usage to reinforce their memory of a single password.

# Lamport's One-time Password

- Alice, the user, remembers a password.
- Bob, the server (computer), has a database in which it stores, for each user, ...
  - The username  $U_i$ .
  - A counter  $n$  that decrements each time Bob authenticates the user.
  - The hash value  $x_n = h^n(\text{password})$ , for some specified hash function  $h$ .
    - $h^i(X) = h(h^{i-1}(X))$  and  $h^0(X) = X$ .
- The setup phase is establishing this information.

# How does it work?

- **Alice** has a workstation, and **Bob** is the server.
- To authenticate we use the following protocol:  
Workstation  $\rightarrow$  **Bob** : **Alice**  
**Bob**  $\rightarrow$  Workstation :  $n$   
Workstation  $\rightarrow$  **Bob** :  $h^{n-1}(\text{password})$
- **Bob** checks if
$$h(h^{n-1}(\text{password})) = h^n(\text{password})$$
- If it does then **Bob** accepts the communicating party as **Alice**. If it doesn't **Bob** rejects the communication.

- Once Alice has been authenticated, the server needs to update its information.
- We replace  $x_n = h^n(\text{password})$  with the one-time password sent by Alice's workstation ...  $x_{n-1} = h^{n-1}(\text{password})$ .
- The value  $n$  is replaced by  $n-1$ .
- When  $n$  reaches 0 we will have run out of passwords in the hash chain and will have to run a new setup process, with a new base password.

# Alice and Bob?

- Alice and Bob are the names typically used to represent the participants in two-party cryptographic protocols.
- There are various other cryptographic identities, so of whom you might meet in next year or so.
- Eve the eavesdropper, Mel the malicious entity, Oscar the opponent, Peggy the prover, Victor the verifier, and so on ...

CSCI262

System Security

(S2b)

User Authentication II:  
Beyond the basic password

# Outline

- Introduction.
- Master passwords and password masters.
- Single-Sign-on.
- Two-factor authentication.
- Biometrics in brief.
- Two-channel authentication.
- CAPTCHA.

# Introduction

- We have seen that passwords have problems.
- In particular the need to securely remember and renew passwords makes good choices of passwords beyond some and difficult for many.
  - When we consider that there are many websites which require passwords the problem is multiplied.
  - Using the same passwords in multiple places is generally a bad idea. Why?
  - Passwords are vulnerable to social engineering, communication eavesdropping.
- There are other mechanisms for authentication.
- We are going to mention some.
  - We will first look at using passwords but in such a way as to alleviate the memory problem.

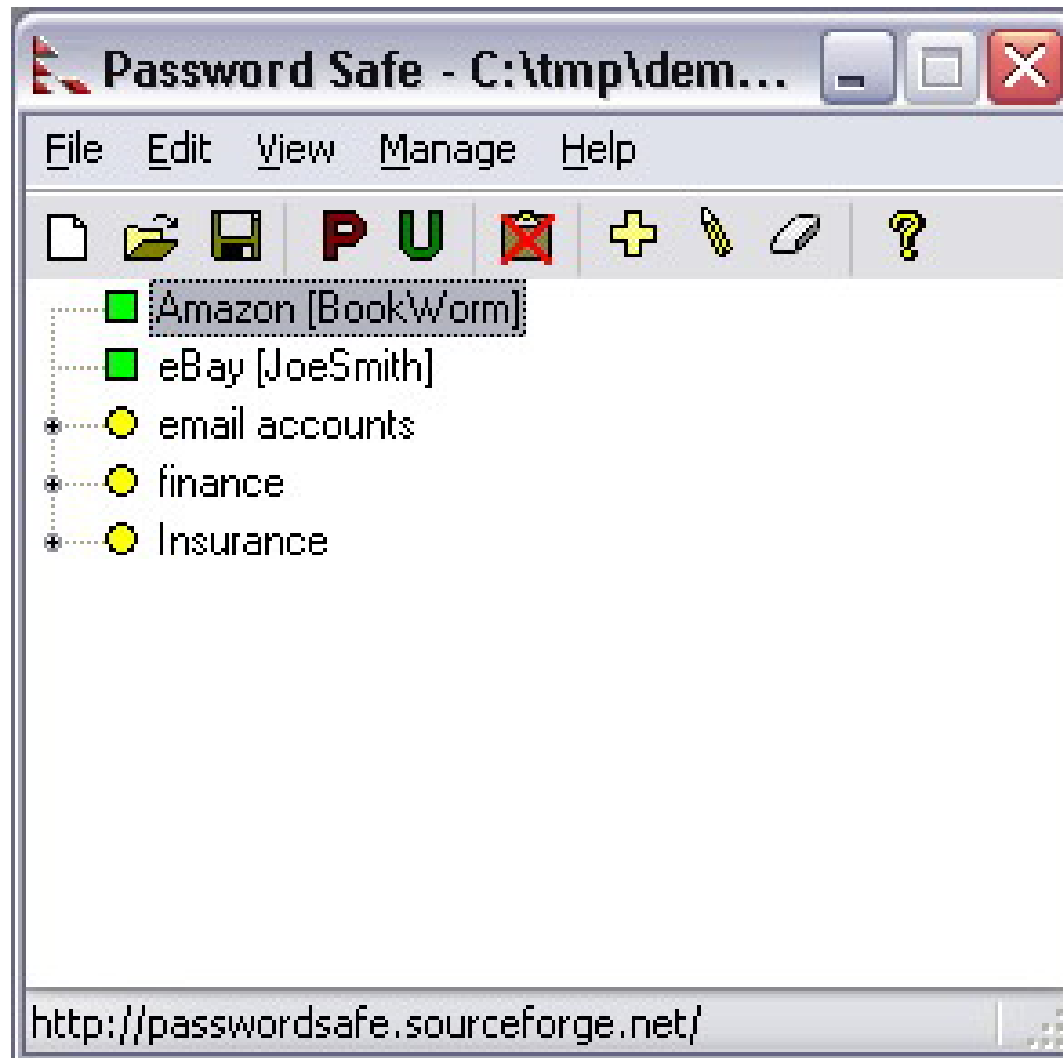


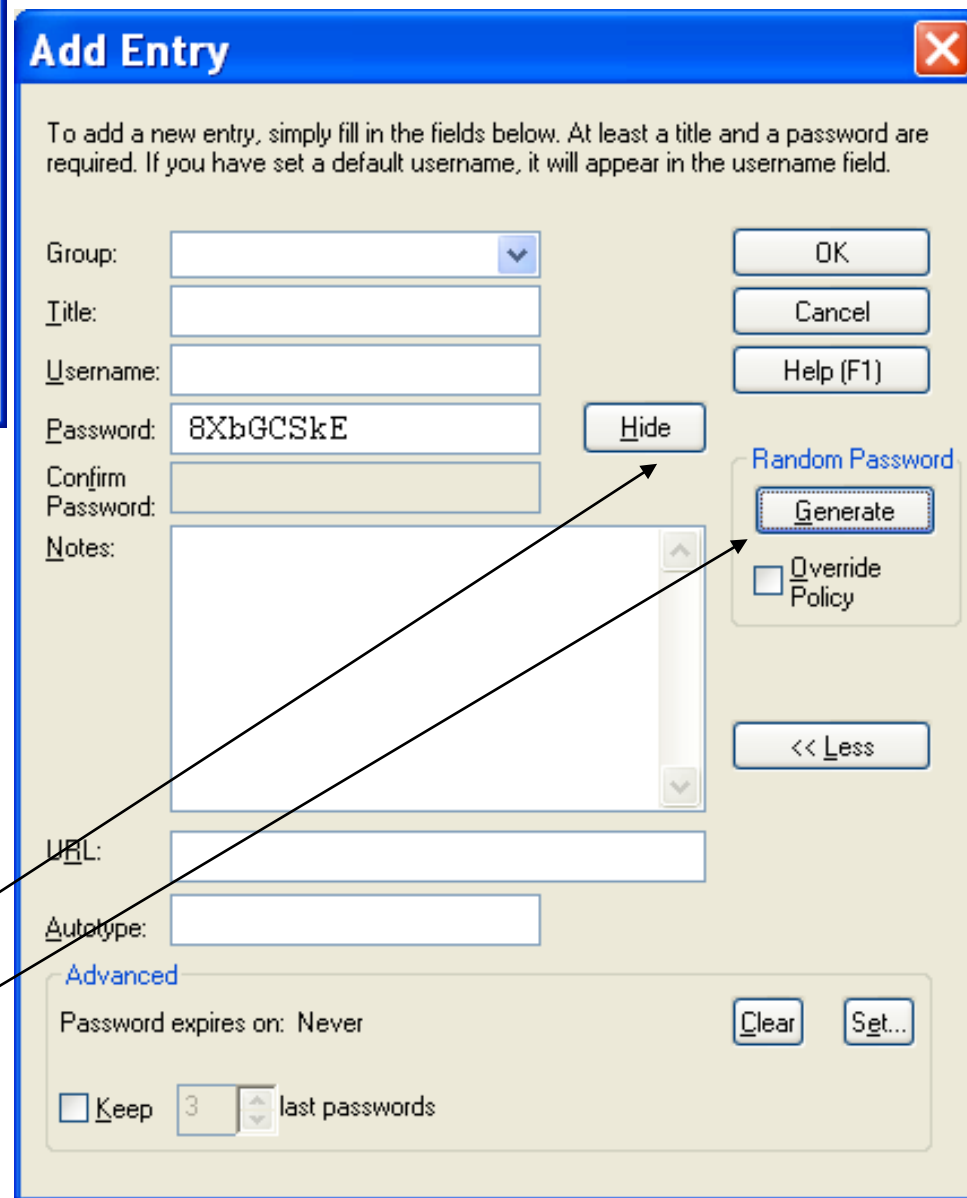
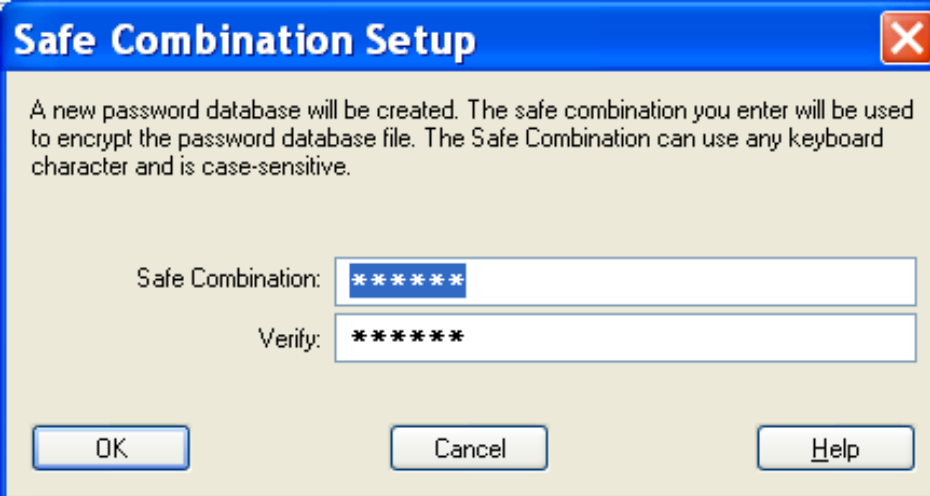
One password to rule them all!...

## Master passwords and password masters

- Rather than remember dozens of passwords, and thus choose very simple weak passwords, or duplicate passwords, we can use a master password.
- The idea is you make the master password a good one, apply the appropriate rules like changing every so often, and never remember any of the others at all.
- Don't forget this one though!

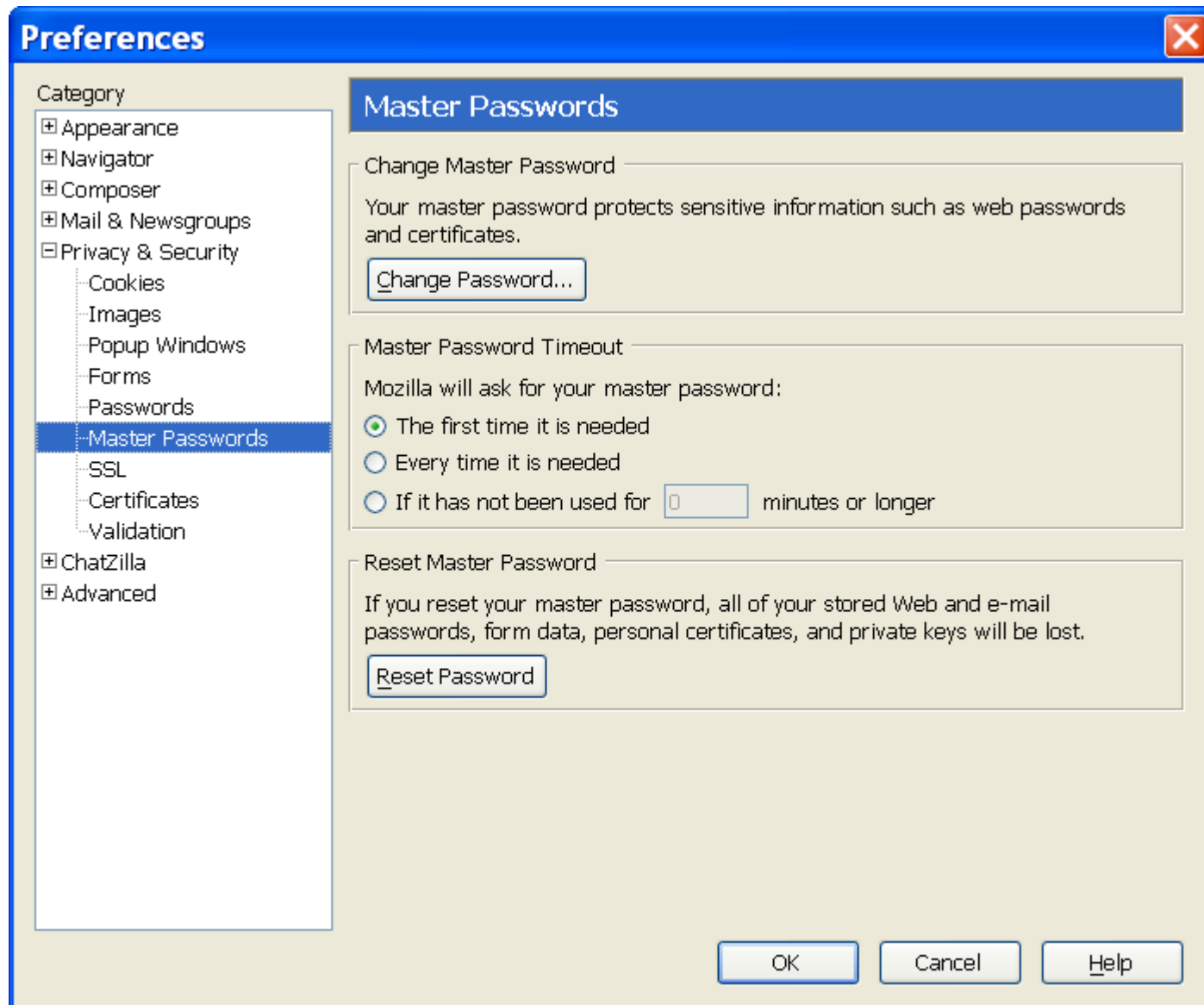
# Software based ...





Hidden random passwords

# ... browser (Firefox) based ...



... and hardware based?

(Владелец Пароля)



Certainly you can install apps on mobile devices.

# Single-sign-on

- Single-sign on is similarly designed to reduce the volume of authentication information, in other words the number of passwords, that needs to be remembered.
- What is the idea?
  - Sign in once!
  - Access lots of resources.

- How does it work? (Very roughly)
  - Users are registered with multiple entities which share information.
  - Centralised authentication generates behind the scene tokens for passing authentication at other locations without explicit subsequent input by the user.
- A couple of major issues ...
  - The single-sign on has to be very well protected.
  - Scalability to work across multiple domains, multiple platforms and with multiple types of application authentication is tricky.
- There is single sign-on for some of the UOW systems now ... but not all.

# Microsoft account (Windows Live ID)

- This is Microsoft's single-sign on system.  
**<http://login.live.com/>**
- It is associated with, but not limited to, the **Windows Live** branding for the Microsoft services and software products.
  - Those are mostly web applications.
- The predecessors to Windows Live ID were Microsoft Passport, Microsoft Wallet ...
  - The plan was to have it being ***the single*** single-sign-on service for all web commerce.
  - This wasn't well received.
- Kerberos, which you might see in CSCI368, can be used as a single-sign on system.



# Multiple factor authentication

- Rather than use one factor, such as a password, we can use multiple factors.
- Generally the different types of authentication have different advantages and disadvantages that can be advantageously combined, or combined to the detriment of security.
- We will firstly look a special case, “two-factor authentication” which is a name reserved for a special type of pairing, not simply any two factors.

# Two-factor authentication

- Rather than rely on a single password, two-factor authentication systems use a password or PIN and a device (card or calculator) which is able to provide one-time type passwords.
- For example, consider you want to connect to your bank:
  - You might enter the PIN into your device and it gives you a one-time login code.
  - Or, you enter the PIN and the displayed value on the card.



From Wikipedia

# Schneier on two-factor authentication

- Schneier has some interesting things to say about two-factor authentication:
  - It solves the problem of eavesdropping and offline password attacks.
  - It doesn't address the current real problems:
    - Trojan horses.
    - Phishing.
    - We will look at both of these later in the subject.

[http://www.schneier.com/blog/archives/2005/03/the\\_failure\\_of.html](http://www.schneier.com/blog/archives/2005/03/the_failure_of.html)

# Forgotten passwords: Wish-it-was-two-factor...

- Read 2b\_WishItWasTwofactor.pdf.
  - And there is more on the website as others commented...
- What was the name of your first pet?
- What is your favourite colour?
- Where were your born?
- All sorts of problems with this kind of thing.
  - Usually a weaker authentication than the password directly.
- And not two factor at all!

# Chip and PIN (The UK name)

- This 2-factor authentication system is based on smartcard technology and is successor to the old sign the imprint of the card type mechanism or swipe the magnetic stripe.
  - It is in accordance with an EMV (Europay, Mastercard, Visa) Standard.
  - There are similar systems in many countries, including Australia.
  - In Australia it is now compulsory to use the PIN.
    - As of August 2014, you cannot sign anymore.

- The card contains an embedded microchip which is read by entry into a Chip machine, or a modified swipe-card reader. This checks the card itself.
  - Subsequently the customer enters their PIN and this is compared against that on the card, or they sign, depending on the reader.
- Generally the supermarket/shop PIN entering is more exposed than PIN entering at an ATM.
  - This is a significant downside.
- Some attacks:
  - 2b\_ChipPinRelayAttacks.pdf.
  - <http://www.cl.cam.ac.uk/research/security/banking/nopin/>
  - <http://www.youtube.com/watch?v=JPAX32lgkrw>
    - Fools the card into thinking it is signature approved ☹

# The Chip Authentication Program (CAP)



- This uses the same EMV smartcards used in the Chip and PIN project.
  - This is for remote authentication though.
- It is a specific instance of the two-factor systems described a few slides back.
- Your card can be inserted into the calculator like device which, when given the correct PIN, generates an appropriate one-time password or similar for some online authentication protocol.
  - It could be used to calculate a message authentication code value to provide integrity for the transaction. This requires entering information twice though. ☹



# Biometrics

- Biometrics for authentication should only ever be used as a component of a multi-factor authentication system.
- Why?
  - They are not private!
- Biometrics are generally used to make attacks by outsiders more difficult, and probably more expensive.
- Biometrics tend to most reliable where the use is supervised, by a guard perhaps
- Beware the Jelly Baby trick. 😊
  - 2b\_Jelly-Babies.pdf

# Types ...

- Handwriting.
- Fingerprints.
  - The most widely used requiring significant technology.
- Face recognition.
- Iris codes.
  - Probably the best hope for a robust biometric system, although it still has some problems.
- Voice recognition.
- DNA.
- More on this in CSCI358: Security Engineering.



Dilbert: 17-Nov-2007

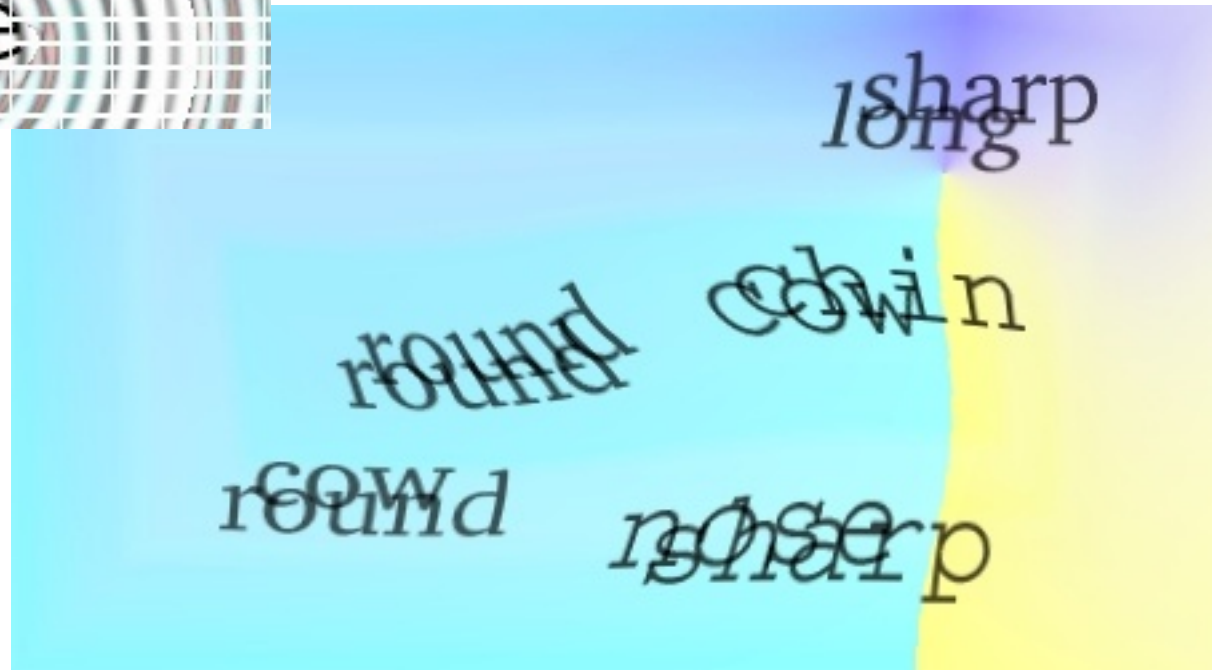
# Two-channel authentication

- A channel is a path which information can be sent through.
- Traditional online authentication is a single channel process.
- Two-channel authentication, of a client to a server, involves the server using a second channel, perhaps a landline or mobile phone, to send information to the client to give targeted authentication.
- On registration with the server the client would need to specify an appropriate alternative channel.
- The security effectively relies on the assumption that the channels are independent.
  - If you use voice over IP and IP then both channels could be compromised together.

# CAPTCHA

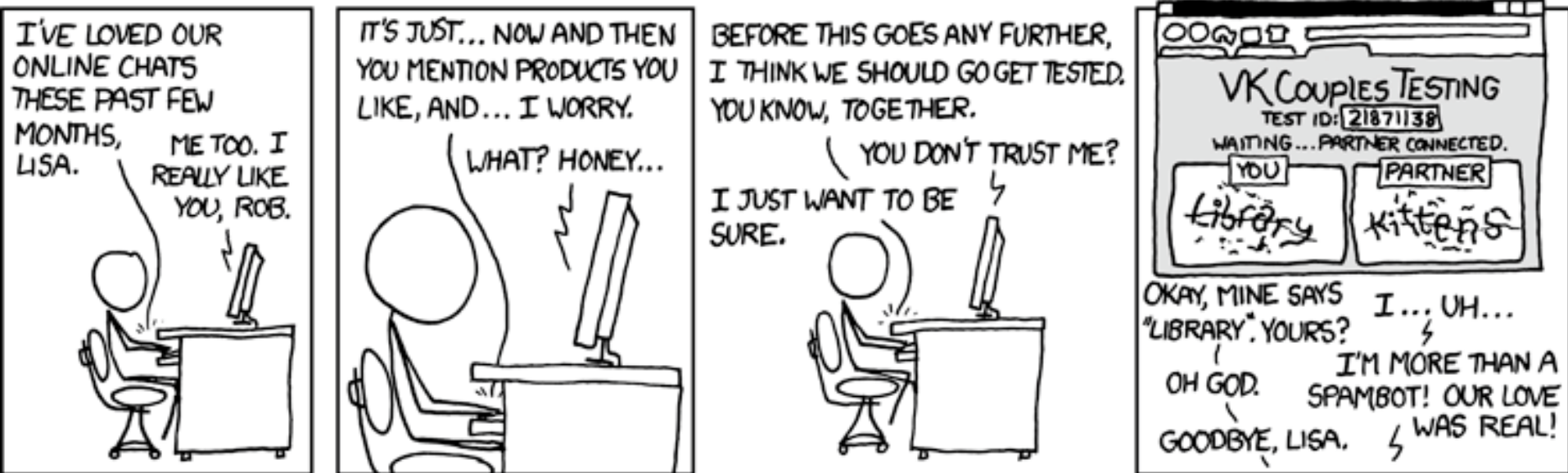


- **C**ompletely **A**utomated **P**ublic **T**uring **T**est to **T**ell **C**omputers and **H**umans **A**part.
- CAPTCHA exploits the human ability to correct distortions in images, and generally perform image recognition, far better than existing automated systems.
- It has uses in authentication and in denial of service.
  - Denial of service is somewhat related to authentication and will be discussed later.



Computer algorithms aren't as good at recognising such distorted words as we are.

- The idea is that only humans can easily read the content of the distorted message.
  - There are image and audio versions as well as text ones.
- So, we declare something to be human if it can read one, or often a series of images.
- For authentication we can embed a challenge to make the response a function of the entity being authenticated.
  - For example, the image could tell you to enter 1452522 into your key calculator and reply with the output.
  - The key calculator is “keyed” to you.



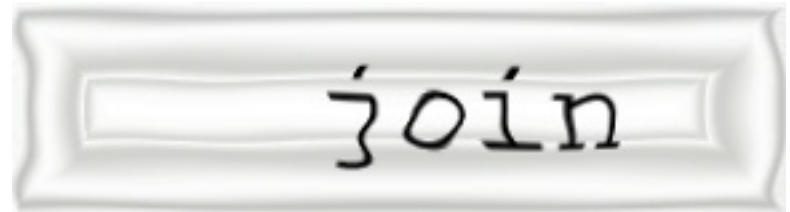
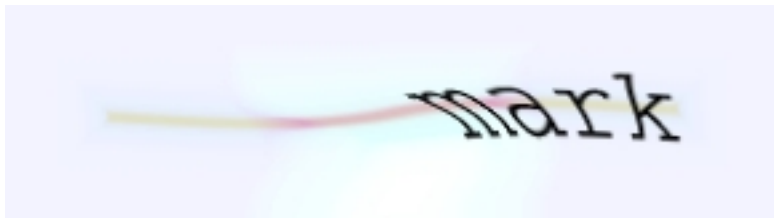
From <http://xkcd.com>

# CAPTCHA

- CAPTCHA can be used as a means to distinguish between live users who want to access a resource, and parties such as spammers using bots to try to overwhelm, or spread messages, without monitoring each connection.
- It can be used to stop posts to blogs, or general websites, effectively stopping resource consumption by malicious entities.



- For example, before uploading a file to somewhere, insist that you enter the words in each of the following:



- Pretty easy for a human, not so easy for a computer.

# Deployment

- Yahoo Mail.
  - Google Gmail.
  - Hotmail.
  - Windows Live Mail.
  - Paypal.
  - ...
- 
- Great to see a useful technology widely deployed.
  - But ...

# CAPTCHA is broken ☹

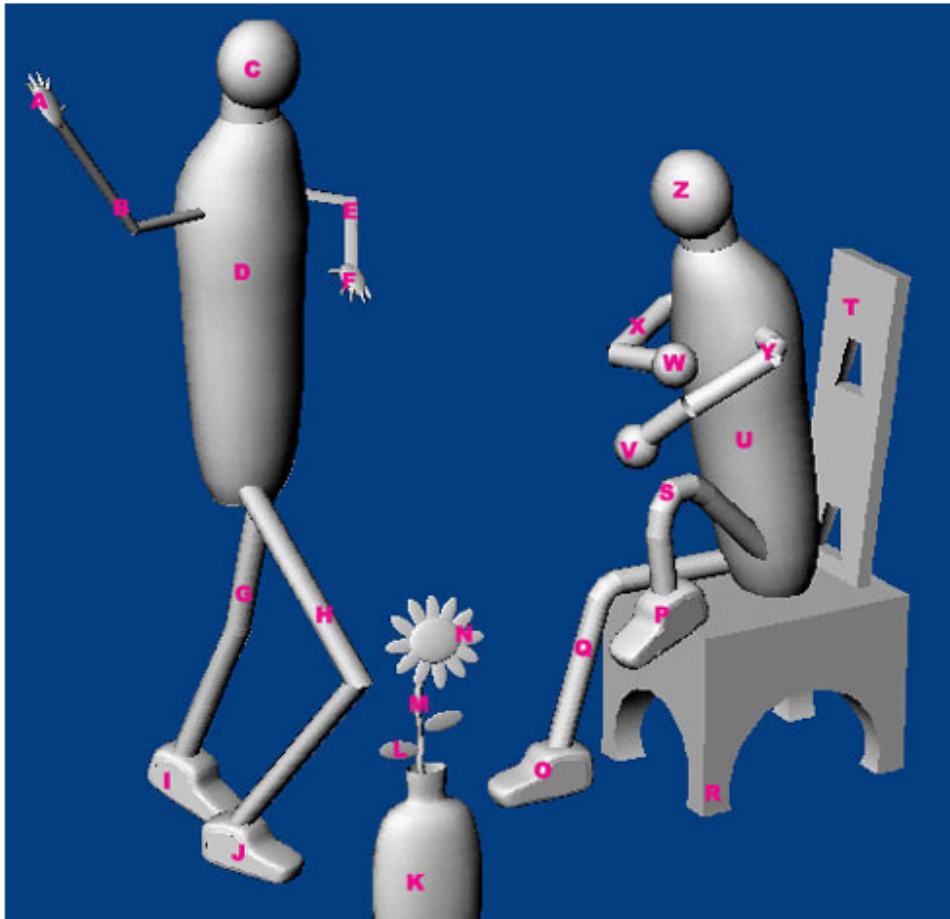
- See 2b\_CAPTCHA-broken.pdf.
- Jan 2008: Yahoo Mail. 35% success rate.
  - Yahoo changed their system.
- Feb 2008: 30-35% against Microsoft's Live Mail.
- Feb 2008: 20% against Google's Gmail.
- And it goes on (getting worse for the CAPTCHA systems).
- Considerable care needs to be taken in using such systems now since CAPTCHA crackers are now freely available.
- Weak CAPTCHA can be broken using Optical Character Recognition (OCR).
- But it isn't all gloom and doom ...

Please click on or enter each letter corresponding to the following list in the field below. You must enter them in the exact sequence listed.

- The Head of the Walking Man
- The Vase
- The Back of the Chair

Enter Letters Here: \_ \_ \_

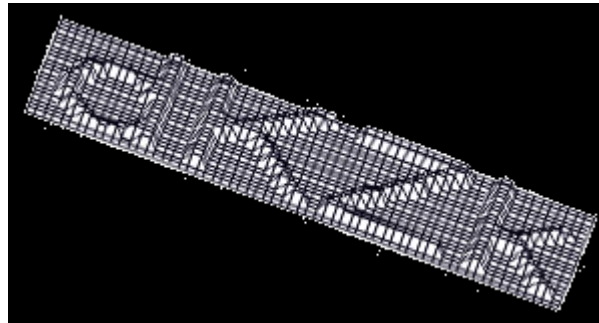
# CAPTCHA 3d ☺



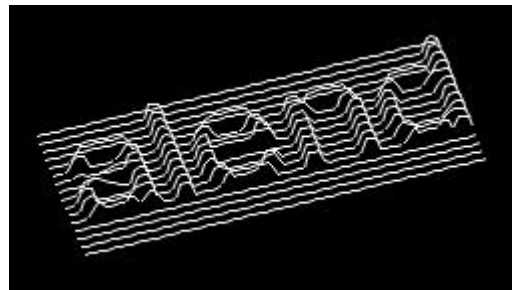
■ <http://spamfizzle.com/CAPTCHA.aspx>

# Others ...

- <http://graphcomp.com/index.cgi?v=0000s2p5>



- <http://code.google.com/p/3dcaptcha/>



# BotDetect CAPTCHA

- Have a look at <http://captcha.biz/>
- You can download free, but restricted PHP, ASP and ASP.NET versions.
- They have support for multiple “localizations”

# CAPTCHA Sniper

- <http://www.captchasniper.com/>
- They claim excellent success rates for breaking a lot of CAPTCHA implementations - up to 100% in some cases.

Perl Guestbook

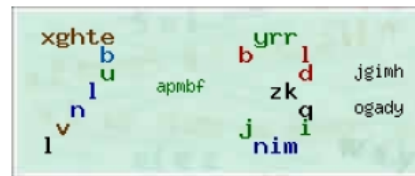


100%

Phoca Guestbook



77%



75%

PhpBook Guestbook



100%

Ricar Guestbook



100%

Silentrum Guestbook



77%

Generic Guestbook #1



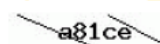
77%

Pixel Post Image Gallery



98%

Article PLR Script



89%