Computer Security Notes

Found at: benjaminshaw.uk

1 Basics

1.1 A Definition of Security

Confidentiality

Ensure that assets are accessed only by authorised parties.

Integrity

Assets can only be modified by authorised parties, or by authorised means.

Availability

Assets are only accessible to authorised parties at the appropriate times.

Accountability

Actions are traceable to those responsible

Authentication

User/data origin accurately identifiable

1.2 Security Countermeasures

Prevention

Stop security breaches via system design and defences

Detection

If a breach does occur, detect it.

Response

A plan utilised when a breach is detected.

1.3 Denial of Availability

A user will expect that services be available to them. A common attack is denying users this privilege. Denial of Service (DOS) attacks or malware are two common ways of attacking availability.

2 Cyber Security Essentials

2.1 Secure Configuration

Principles:

Devices on a network should be configured such that they minimise the number of inherent vulnerabilities.

Default settings can *often* be insecure, which includes default passwords.

Actions:

- Remove unnecessary user accounts, such as the *Admin* account found on Windows XP installs.
- Changing the default password
- Removal of unnecessary software
- Firewall software should regulate the incoming/outgoing connections on a device

2.2 Boundary Firewalls & Internet Gateways

Principles:

Devices should be protected against unauthorised access and disclosure.

Firewalls are the first line of defence and can stop attacks before they even reach the network.

Actions:

- Change default passwords
- Rules should be scrutinised before they are applied
- Unapproved services should be blocked by a rule
- Obsolete rules should be purged
- Firewall administration tools should not be accessible from outwith the network

2.3 Access Control and Privilege Management

Principles:

User accounts should have the minimum amount of privileges, with extended privileges awarded upon authorisation.

A compromised account with high levels of access can lead to a lot of damage.

Actions:

- Account creation should be subjected to an approval process
- Administration accounts should only be used for legitimate administration purposes and not activities that can be achieved with a standard account
- Elevated privilege accounts should require password changes periodically
- Users should be authenticated before being granted access to devices and applications
- Elevated accounts should be used when no longer required

2.4 Patch Management

Principles:

Remove unnecessary vulnerabilities by keeping software up-to-date.

Actions:

- Software should be kept up-to-date and fully licenced
- Out-of-date software removed
- Updates when made available should be installed in a timely manner

2.5 Malware Protection

Principles:

Internet-facing devices should make use of malware protection software that continuously monitor for known-malware instances.

Actions:

- Install anti-malware software
- Keep said software up-to-date
- Regularly scan all files
- Prevent connections to known malicious website

3 Network Security Threats

Types of Threat:

- **Interception**

  Unauthorised viewing of information
- **Modification**

  Unauthorised changing of information
- **Fabrication**

  Unauthorised creation of information
- **Interruption**

  Prevention of authorised access

3.1 Man-in-the-Middle

When communications between two devices is intercepted and changed by a third party intruder.

Man-in-the-middle is not necessarily an attack, as VPNs are man-in-the-middle by nature.

3.2 Denial of Service

When valid users are prevented from accessing a service.

3.2.1 SYN Flooding

When a large amount of *SYN requests* are sent to the victim with the intention of overloading them. SYN packets are the first packet sent in a TCP handshake.

The attacker would send many of these packets without acknowledging any of the replies to the point that the receiver can handle no more.

SYN flooding uses a lot of the attacker's bandwidth, and is also traceable if sent from their own IP. An ideal type of attack against a small target but tends not to be affective against larger targets.

SYN flooding in *conjunction with TCP source spoofing* makes it harder to trace the original attacker and ACKs will be directed to some other address.

3.2.2 Distributed Denial of Service

The same concept as a denial of service attack, but multiple systems flood the intended victim.

3.2.3 Smurfing

An example of *distributed* denial of service attack.

Exploits the *Internet Control Message Protocol* (ICMP) specification of *pings*, which allow for hosts to indicate that they are alive.

A forged ping packet is constructed with the source IP address as the victim's IP, and is sent to a *smurf amplifier*, subsequently that will swamp the target with replies.

A smurf amplifier is a poorly configured host that will not anticipate this attack, and will reply to the forged host.

3.3 Domain Name Service Attacks

Utilising the DNS system, where a DNS record will be maliciously altered to set up a man-in-the-middle attack, or prevent access to some URL.

4 Network Defences

4.1 Firewalls

Firewalls *divide* the untrusted exterior of the network and the trusted interior of the network; they are said to operate on the *Network* layer, or the 3rd layer, of the IP stack.

Firewalls either run on dedicated hardware or as a software package such as `iptables`. Hardware firewalls operate faster and act as a physical divider, whereas software is easier to deploy.

Firewalls operate upon a *set of rules*. These rules dictate if it allows or denies any given traffic.

4.1.1 Packet Filtering

The simplest task of a firewall, compares packet headers to the pre-defined rules.

However, this does not detect forged packets and requires a large and well-defined set of rules to function well.

4.1.2 Stateful Inspection

Remembers state between packets received, unlike a packet filter.

This allows it to filter attacks that utilise the scanning of many ports on a host, as it would detect the high number of packets from one source. With this piece of information, it could add to the firewall rules to block this specific host.

4.1.3 Application Proxy

A type of man-in-the-middle attack, as it screens messages received at the *Application* layer of the IP stack.

This allows for the firewall to block application requests, such as emails containing certain words or confidential information from a database leaving the network.

4.2 Intrusion Detection Systems

Designed to detect a potential intrusion *in progress*.

IDSs will monitor the network for behaviour similar to that of known attacks and raise the alert when it does so.

There is a fine line to walk as IDS needs to be sensitive to detect attacks, as they are not prone to being as blatant as possible; however, too many false alarms can soon prove to a be a liability.

4.3 Signature-based

Utilises *pattern matching*, where a pattern is provided in advance.

This method does not anticipate new types of attack but has a high level of accuracy which is effective for well-known and common types of attack.

4.4 Heuristic-based

The method builds a model of what *normal* behaviour looks like. This allows for possible anticipation of new types of attack.

However, it can take a long time to build this model and is susceptible to false positives.

5 Security Amongst Users

A large number of attacks occur due to user error, so it is in the best interests of organisations to make sure that users are able to identify attempts to fool them.

5.1 Phishing Attacks

A type of attack that *baits* the user into interacting with a malicious object. This can be done by impersonating a person or organisation that the user may trust beforehand.

This type of attack is usually delivered by email to try and get the user to go to an illegitimate link that will provide the user with malware.

Spear phishing is used against a small subset of users, but allows for the attacker to craft the attack in a way that is more likely to trick the user.

Whaling is an attempt to use a phishing attack against a financially wealthy victim.

Earmarks of a phishing attack are hyper links to an address that is designed to look like something else, e.g. *www.microsoftrewards.legitwebsite.com*, or emails that come with attachments with titles designed to lure the user to open it.

5.2 SSL/Malware Warnings

Secure Socket Layer is the technology that `HTTPS` is built upon, and allows for secure communications between two parties.

SSL allows for validation that the two parties are who they say they are and prevents a man-in-the-middle from reading packet contents, as they are encrypted.

Most modern web browsers will alert users when there are *certificate mismatches*, which can be an indication that the website is trying to mimic another's identity.

Many modern browsers will also look up the URL that the user is trying to visit, and cross-reference this with a blacklist of *known malicious websites*. If this check comes back as true, then a warning shall be displayed to users indicating that the site they are trying to visit has been knwon to distribute malicious content.

Whilst this can be effective against users visiting *new* websites, studies have shown that users are likely to ignore such warnings if they are trying to visit a website that they have been to before. This may be because users think that the blacklist check is at error, or that they can trust the source of the link that they are following.

Roughly *one and a half percent* of these warnings are false-positives.

5.3 Meaningful Warnings

It is obviously in the best interests of network administrators and developers to have warning messages that are understandable and effective when displayed to a user.

5.3.1 NEAT

- **Necessary**

  Is there a need for this user to see or decide upon the warning?

- **Explained**

  Is all information provided for the user to make an informed decision?

- **Actionable**

  Is there a set of steps that the user can take to make the correct decision?

- **Tested**

  Have you tested upon someone who would be an expected user, both in benign and malicious situations?

5.3.2 SPRUCE

- **Source**

  Ensure the source of who is asking for the decision is made clear

- **Process**

  Provide the user with actionable steps to make a good decision

- **Risk**

  Explain the risks of making the wrong decision to the user

- **Unique User Knowledge**

  Tell the user the information that *they* bring to the decision

- **Choices**

  List available options, *clearly recommending one*

- **Evidence**

  Highligh information the user should include or exclude whilst making the decision

6 Overflow Attacks

6.1 Vulnerabilities in C

Many of the original C functions are susceptible to attacks, as they look for the *null byte*, `'\0'`. The attacker can craft an input that doesn't contain the null byte, resulting in the vulnerable code to keep on reading input in. Such fundamentally vulnerable functions are `strcpy` and `scanf`.

6.2 Buffer Overflows

When executable code is injected into areas of memory earmarked for code execution, by overflowing areas that are, *most likely poorly*, designed to take input.

The malicious code will then be pushed into some other area and possibly change the flow of execution that the program follows.

6.2.1 Shell Injection

The aim of this type of attack is to *spawn a shell*, which would give the attacker general access to the victim's system.

6.3 Integer Overflows

When an attempt is made to store a value greater than the maximum value an integer can hold.

Poorly implemented code is susceptible to an attack using such a method as it may result in length checks failing to terminate.

6.4 Formatted String Exploits

If an attacker is able to ascertain what *format* a program is expecting to receive, they can craft some input that will directly load malicious code into memory.

6.5 Defences Against Overflow Attacks

6.6 Stack Canaries

Named after the canaries used in mines to detect carbon monoxide leaks in deep mines, they are used to detect when an overflow has caused some data within the stack to be overwritten.

The *stack canary* will be placed just before a return pointer. Before execution, the canary will be checked if it still present before the pointer, as to overwrite the pointer, the canary must be overwritten too.

If the canary is not present, the code will not be executed.

Canaries can hold random values calculated at the start of the process, or can hold termination values such as the `NULL` byte.

6.7 Make Stack and Heap Non-Executable

Prevention of overflowing code being executed, as the range it is within is not executed. However, the code can still be executed by using some of the *system calls* found within `libc`.

6.8 Address Space Layout Randomisation

Standard libraries are placed in randomly allocated, non-standard locations. This prevents attackers from directly pointing to where a system library call may usually be.

7 Authentication

To verify a fact about an entity before it can be allowed to execute some action. This can be achieved by requiring something they *know*, *are* or *have*.

7.1 Multifactor Authentication

An authentication process that requires more than one factor; examples being the *chip* and *pin* within your bank card.

7.2 Passwords

The most popular method of authentication, a *password* is a character string known to the user and system. They are easy to store, enter and require no special equipment whilst being easily scalable.

7.3 Password Entropy

The problem is different passwords have differing levels of *safety*.

A bad password would be one that has low *entropy*, or predictable in other words. It is commonly found that users use a common string, such as `'123456789'`, or an easily identifiable trait about their self, such as their name or date of birth.

A good password contains a wider range of characters, such as special characters and mixed cases.

A way to combat this is utilising *machine generated passwords*. A well made program can create passwords that are much harder to guess, but tend to be harder for a human to remember. Algorithms have been created to create passwords that are *pronounceable*, increasing the likelihood of a human being able to remember it.

7.4 Protecting Passwords

Passwords are rendered useless if an attacker is able to compromise the server and learn it that way. There are a number of mechanisms designed to keep passwords safe in the instance that the server has been compromised.

7.4.1 Hashing

A *good* hashing function translates some input to some unique output in a way that is hard to determine the underlying function itself.

This means that the server only stores *hashes*, instead of the plain-text passwords. Even if the hashes were obtained, they would be meaningless to the attacker.

A method called *salting* is used as additional input to further increase the security of hashing. This means that for one string as input but a set of different *salts*, the output would be different for each salt.

This can serve as a protection mechanism against *brute force attacks*. If a server is compromised, the attacker could continuously create a number of passwords with the hashing function to figure out the mapping between inputs and outputs. Salting makes this far more difficult to break the hashing function as it would need to be done for *each* salt alongside the hashing function.

7.4.2 Lockout

Another brute force attack is to utilise make a massive number of guesses. This takes advantage of the fact that many user-defined passwords are predictable. An attacker can guess many times against an unsecured authentication system and correctly guess the password in a minuscule amount of time if the password has a low enough entropy.

The idea of a *lockout* is to limit the amount guesses that the attacker has before they are prevented from another attempt, or requiring another factor that they may not have.

7.5 Keys

Accompanies the concept of using something you *have* as a means of authentication. Like having a key to unlock a physical doors, keys are also used within digital authentication.

7.5.1 Key Fob

A *key fob* has a securely generated number that must be provided alongside the password as a means of two factor authentication.

7.5.2 Public/Private Pair

A pair of keys, one of which is *private*, the other which is *public*. A message that one locks and the other unlocks, and vice versa.

7.6 Biometric & Behaviour

To accommodate the idea of something you *are* as a method of authentication.

7.6.1 Fingerprint Readers

Fingerprints are *nearly unique* to each person, but can sometimes be difficult to read or, more akin to Cold War antics, can be destroyed. They can also never be changed, which is severely detrimental in the event that they are somehow obtained.

7.6.2 User Behaviour

A means of *continuous authentication*, a user's behaviour is monitored. A user's typing patterns is something that is hard to mimic, but this can suffer from real world events such as the user being injured.

8 Access Control

8.1 Access and Information Flow