TOWERS WATSON

Security Training

Jacob Meline

July 8, 2015

Contents

1 Introduction

Some useful terms to know:

OWASP The Open Web Application Security Project. Is an online community dedicated to web application security. Participants include corporations, educational organizations, and individuals.

PHI Protected Health Information

1. The individual's past, present, or future physical or mental health or condition.
2. The Provision of health care to the individual, or
3. the past, present, or future payment for the provision of health care to the individual,
4. the individual's identity or for which there is a reasonable basis to believe it can be used to identify the individual

PII Personal Identifiable Information

1. Social Security Number
2. Driver's license or California Identification Card Number
3. Account number, credit or debit card number, in combination with any required security code, access code, or password that would permit access to an individual's financial account

2 SQL Injection A1

2.1 What is SQL Injection

A simple example could look like this: ‘http://mysite.com/color?colorid=4‘ It doesn’t matter what site it is, this site has a path called ”color” with a query string called ”colorid” with a value of 4. The query probably looks like something like this. ”select * from color where colorid=4”

”4” is data. It is untrusted. it comes in via the url. Changing ”4” to ”5” will give you a different color

Internal errors can give hackers an idea of what is allowed in your database or not.

2.2 Why SQL Injection Matters

Discoverability EASY, Prevalence COMMON, Exploitability EASY, Impact SEVERE

Further readings:

- acunetix’s article

2.3 Why is SQL Injection a problem?

Example of attack

- A simple example of SQLi
- A more realistic example
- Another realistic example

In the first hackthissite example, we first have to look everywhere in the site.

Investigate the Site Source Code You have to dig through the code. A lot of modern sites will have tests to stop sqlinjection. But you still need to watch out for vulnerabilities. By looking through the source code, you can sometimes find some pretty serious issues. The input is clearly visible. The first step is to attempt to get out of the string. Try registering an email address contatining apostrophes, both single and double. By trying different combinations, we get a different type of message. ”Error inserting into table ”email”! Email not Valid. Please contact an administrator of Fischer’s

Check the other pages Looking at the other products, both are using products.php with the category ID as an argument. You can change the argument to be ?category=2 or category=1 to get a combination of both pages on the same page.

Lets say that a form is asking for user credentials and enter ”Tom,” well, that is fine and all. If you enter Tom” you’re going to have some vulunerability concerns. You can do Tom”; DROP ALL DATABASES; to do some serious damage. To mitigate this attack, you can simply put a slash before any quotation mark. This forces SQL to read it as a quotation mark and not a continuation of a command.

2.4 How to mitigate SQLi attacks?

Links to useful sites

- SQLi Prevention Cheat Sheet

3 Cross Site Request Forgery

3.1 What is Cross-Site Request Forgery

A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim's broswer to generate requests the vulnerable application thinks are legitimate requests from the victim.

4 Cross Site Scripting

Cross-Site Scripting is a vulnerability that doesn't sanitize user input properly. It allows an attacker to inject HTML or client side script such as Javascript into a website. It is commonly used to steal cookies. Cookies are used for authenticating, tracking, and maintaining specific information about users. There are three different types of Cross Site Scripting:

Persistent
the malicious input originates from the website's database.

Non-Persistent
the malicious input originates from the victim's request.

DOM-Based
the vulnerability is in the client-side code rather than the server-side code.

4.1 How to mitigate Cross-Site Scripting

- The most important way to prevent XSS attacks is to perform secure input handling.
  - Most of the time, encoding should be performed whenever user input happens
  - In some cases, encoding has to be replaced by or complemented with validation
  - Secure input handling has to take into account which context of a page the user input is inserted into
  - To prevent all types of XSS attacks, secure input handling has to be performed in both client-side and server-side code
- Content Security Policy provides an additional layer of defense for when secure input handling fails

5 Clickjacking

5.1 What is Clickjacking?

OWASP gives a good example of what click jacking is:

Imagine an attacker who builds a web site that has a button on it that says ”click here for a free iPod”. However, on top of that web page, the attacker has loaded an iframe with your mail account, and lined up exactly the ”delete all messages” button directly on top of the ”free iPod” button. The victim tries to click on the ”free iPod” button but instead actually clicked on the invisible ”delete all messages” button. In essence, the attacker has ”hijacked” the user’s click, hence the name ”Clickjacking”.

Basic ingredients to prepare for a clickjacking attack are:

Iframe This is a frame in the HTML that frames a webpage in it

Z-Index Decides the iframe index in the stack

Opacity Makes the iframe transparent

Position:Absolute Lines up the iframe with the dummy page

5.2 How to mitigate against it?

- Sending the proper X-Frame-Options HTTP response headers that instruct the browser to not allow framing from other domains
- Employing defensive code in the UI to ensure that the current frame is the most top level window
  - Frame-breaker By inserting this script into the header of your site, it is an easy way to break the iframe.

```
<script>
 if (top != self){ top.location = self.location; }
</script>
```

6 Authentication

All approaches for human authentication rely on at least one of the following

Multi Authenication

Something you know (eg. a password) This is the most common kind of authentication used for humans. We use passwords every day to access our systems. Unfortunately, something that you know can become something you just forgot. And if you write it down, then other people might find it.

Something you have (eg. a smart card) This form of human authentication removes the problem of forgetting something you know, but some object now must be with you any time you want to be authenticated. And such an object might be stolen and then becomes something the attacker has.

Something you are (eg. a fingerprint) Base authentication on something intrinsic to the principal being authenticated. It's much harder to lose a fingerprint than a wallet. Unfortunately, biometric sensors are fairly expensive and (at present) not very accurate.

6.1 Account Lockout

In an account lockout attack, an attacker attempts to lock out user accounts by purposely failing the authentication process as many times as needed to trigger the account lockout functionality. This in turn prevents even the valid user from obtaining access to their account. For example, if an account lockout policy states that users are locked out of their accounts after three failed login attempts, an attacker can lock out accounts by deliberately sending an invalid password three times. On a large scale, this attack can be used as one method in launching a denial of service attack on many accounts. The impact of such an attack is compounded when there is a significant amount of work required to unlock the accounts to allow users to attempt to authenticate again.

6.1.1 Ebay Account Lockout Attack

At one time, eBay displayed the user-id of the highest bidder for a given auction. In the final minutes of the auction, an attacker who was wanting to outbid the current highest bidder could attempt to authenticate three times using the targeted account. After three deliberately incorrect authentication attempts, eBay password throttling would lock out the highest bidder's account for a certain amount of time. An attacker could then make their own bid and the legitimate user would not have a chance to place a counter-bid because they would be locked out of their account.

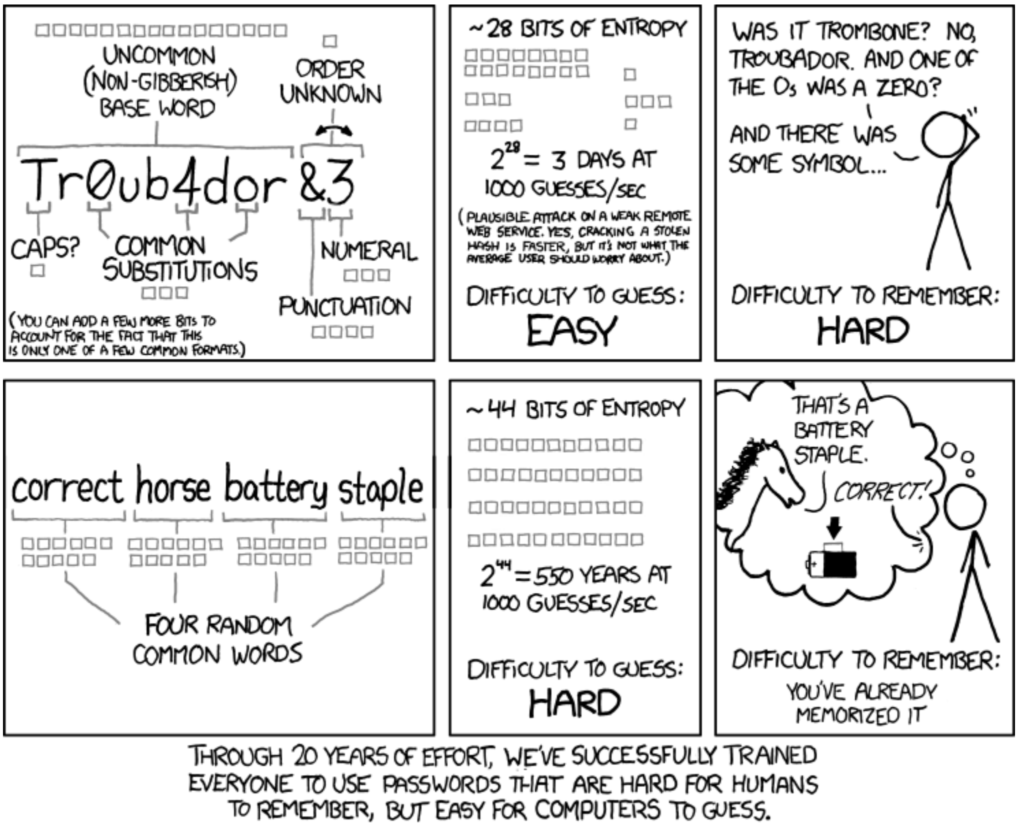
6.1.2 Mitigation Strategies

Brute force attacks are surprisingly difficult to completely mitigate. Careful design and implementation will assist in helping to minimize these attacks.

- Since passwords are not always the best option. Having a public-private key setup (Something you have) can help in some circumstances

- Time-increasing failed login. For example, after 3 unsuccessful logins, you could block the login for this account for 30 seconds. After the 30 seconds, if the 4th login is still unsuccessful, you stop it for 1 minute, then 5m, and you increase more and more

6.2 Passwords and Hashes


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Sources:

- John the ripper
- Your password is too short
- cracking passwords like a pro

6.3 Password History

It is typical in a company to require its employees to have a new password every 6 - 12 months. On top of that, there can be time restrictions on the previous passwords so that even if old passwords were compromised, attackers wouldn't be able to brute force their way through a secure network.

6.4 What is the correct way of storing passwords?

Never store passwords as plaintext Store the hashes, never the actual passwords

Add a long, unique random salt to each password you store The point of a **Salt** is to make each password unique and long enough that brute force attacks are a waste of time. This allows the password to be immune to rainbow table attack.

Don't invent your own "clever" password storage scheme Use something that has been thoroughly tested.

Use a cryptographically secure hash SHA-1 and MD5 are ones to avoid. These don't add any salt to the passwords

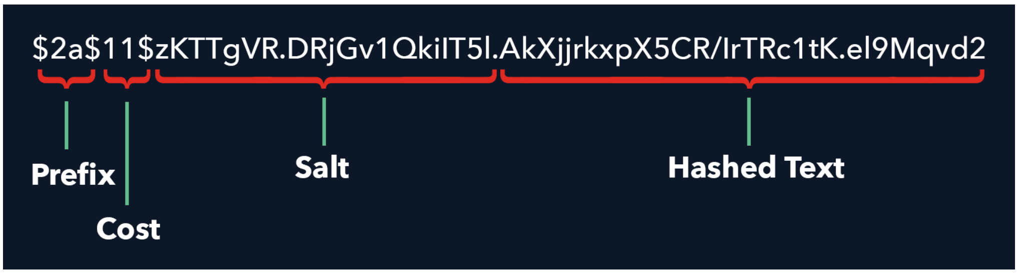
6.5 Hashes

Hashes are a one-way function which take an input string and turns it into a value called a Digest. They are considered practically impossible to invert.

- It is easy to compute the hash value for any given message
- It is infeasible to generate a message from its hash
- It is infeasible to modify a message without changing the hash
- It is infeasible to find two different messages with the same hash

6.5.1 Hash Anatomy

a Hash is made up of multiple parts. This allows you to determine the settings used to create the hash, thereby allowing you to validate against it with needing any additional information.


$2a$11$zKTTgVR.DRjGv1QkiIT5l.AkXjjrkxpX5CR/IrTRc1tK.el9Mqvd2
Prefix
Cost
Salt
Hashed Text

6.5.2 Cracking Hashes

Depending on the operating system that you are on, you can find a tool to crack passwords.

Windows Cain and Abel: Strictly for Windows, it can crack numerous hash types including NTLM, NTLMv2, MD5, wireless, Oracle, MySQL, SQL Server, SHA1, SHA2, Cisco, VoIP, ando others

Linux John the ripper: Powerful, lack of gui. Has an unique password cracking strategy. First, it attempts a dictionary attack, if that fails, it then attempts to use combined dictionary words, then tries a hybrid attack of dictionary words with special characters and numbers, and only if all those fail will it resort to a brute force method. Has support for computing over a GPU with CUDA and OpenMPI

OSX Various tools are available.

EXAMPLE!