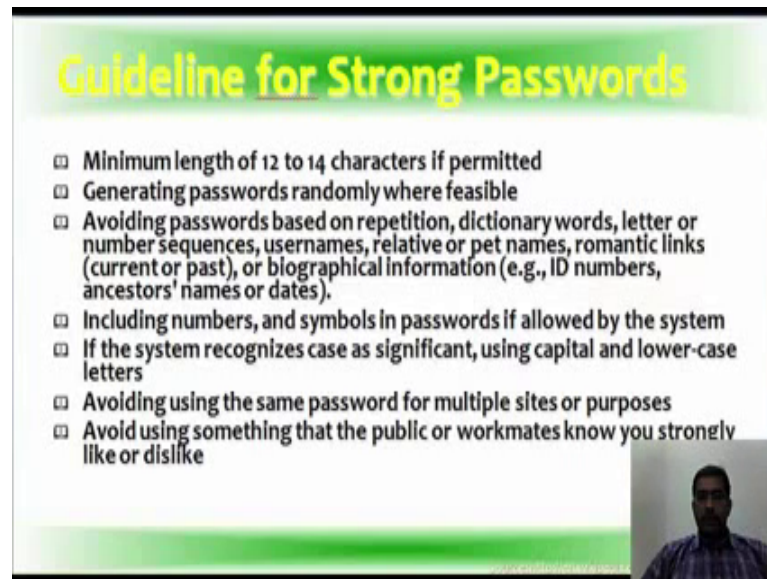


Introduction to Information Security
Prof. Dilip H. Ayyar
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 63

We have seen the guidelines for strong password in domain 1 and 2.

(Refer Slide Time: 00:11)



But, then for web application specific let us discuss at once, the minimum length of 12 to 14 characters if permitted, generating passwords randomly wherever feasible, avoid password based on repetition, dictionary words, letters or number sequences, user names relatives names, pet names, romantic links, current or past biographical information example, ID numbers, ancestors name or dates, including numbers and symbols and passwords if allowed by the system.

If the system recognizes the case as significant using capital and lower case letters, avoiding using the same password for multiple sites or purposes, avoid using something that the public or workmates know you strongly like or dislike. Now, when you look at that it is very good to say on paper or tell in presentation that use minimum 12 to 14 characters is permitted, but in reality many of the people don't use 12 characters.

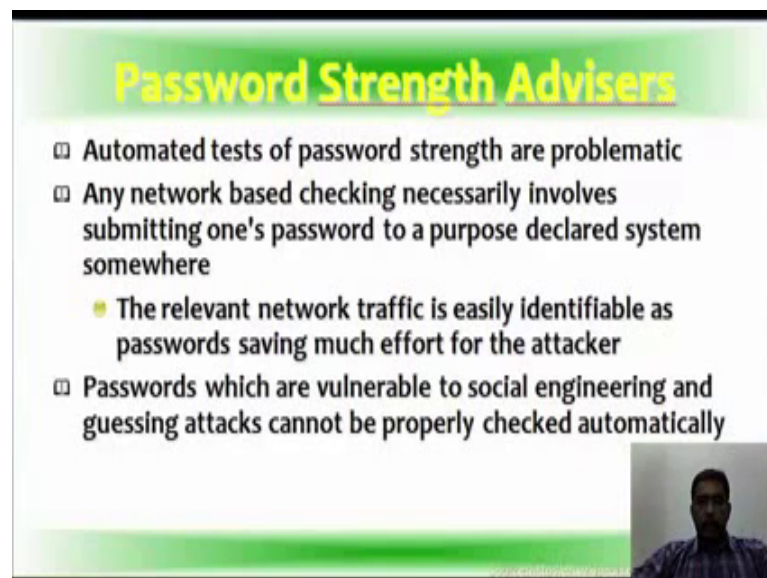
Now, there is a method also for making passwords, easy to remember password even if it is long. Now, marry had a little lamp, so MHALL is marry had a little lamp. So, you give different combinations of that instead of one value you can put 7 then add some special

characters at the end or add some numbers also will the end. So, you can make something that is easy to remember at the same time it is long, when random password is in your banking transactions before you do a financial transactions, let say you are buying something from online site random password is may be a OTP is generated and put, then repetition you have to avoid passwords on based on repetition like a, a, a, 2, 3, 4, 5, like that dictionary words letter or number sequences a, b, c, d now 1, 2, 3, 4, user names related.

So, there are several criteria for setting up strong password, you should make sure there it fulfills the criteria. So, that it your account is safe, brute force will be veritable to get if you have a long password and follow the rules of having the Alphanumeric special character allegation and do not use the same password one for Google, the same thing is repeated for yahoo, Amazon your flip kart sites, snap deal sites, so keep different password is possible.

Suppose let us say that I like chocolate, ice cream. So, I make combinations out of chocolate and make it as a password. So, avoid using something that a public or a work mates know that you strongly like out this slightly. So, that it is exactly the example for them.

(Refer Slide Time: 03:49)



Password Strength Advisers

- ❑ Automated tests of password strength are problematic
- ❑ Any network based checking necessarily involves submitting one's password to a purpose declared system somewhere
 - The relevant network traffic is easily identifiable as passwords saving much effort for the attacker
- ❑ Passwords which are vulnerable to social engineering and guessing attacks cannot be properly checked automatically

www.cyberintegrity.com

www.cyberintegrity.com

The automatic test of password strength are problematic, any network based checking necessarily involves submitting one's password to a purpose declared system somewhere. The relevant network traffic is easily identifiable as the password saving requires much

efforts for the attacker, passwords which are vulnerable to social engineering and guessing attacks cannot be properly checked automatically.

(Refer Slide Time: 04:21)

Let's test some passwords

- ❑ 12345 Instantly Crackable
- ❑ abcdefg Instantly Crackable
- ❑ abcdefg12345 PC would take ~37 years to crack
- ❑ abcd1234! PC would take ~1351 years to crack
- ❑ laura21052005 PC would take 16 billion years to crack

+ some creative trial & error
+ some research
+ some social engineering

ALL those p
cracked rath

Let us test some passwords 1, 2, 3, 4, 5 instantly track able using the tools a, b, c, d, e, f, g then a, b, c, d, e, f, g, 1, 2, 3, 4, 5 PC would take around 37 years to crack at 8 samples per second. So, that you need some creative trial in error some research to be done, some social engineering techniques, in the end you will find that all the characters or all the passwords that are mentioned above are easily crackable. So, based on the length of the password the system computes that it takes 1351 years to crack, but then a, b, c, d, 1, 2, 3, 4 is very simple to crack if you have the right rules, right methods, right methodology.

(Refer Slide Time: 05:13)

Two-Factor-Authentication

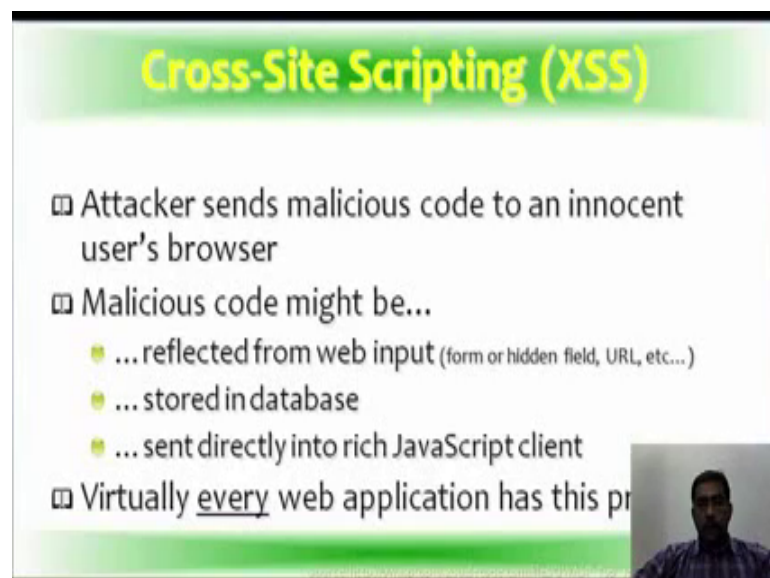
- Old school Paper TAN List
- Facial Recognition via Webcam for Unlocking computer
- Computer won't boot unless Security USB-Device is plugged in
- Protect your account via RSA-Token or Smartphone App
- TANs and dedicated confirmation of suspicious payments via text message

We all heard of two factor authentication. So, two factor authentication is a method, where in addition to your password you use to some other form of authentication. So, it cannot be two passwords, then it becomes a multilevel authentication. So, if one should be a remember info and the other should be a possessed info. Now, there are different methods, the are of-late its been on your laptops you would have seen facial recognition via web cam for unlocking the computer.

Then, on the mobile phones protect your account via RSA token or there are several apps available for the android and IOS . Then your dedicated conformation of subspecies payments via the text messages. So, there are multiple levels of doing that. In the banking industries specifically the banks for the corporate customers give a token something similarly to a RSA or Lasco. So, some kind of token is given it was I remember first it came in early 2000 in HSBC.

So, a lot of multi layered authentication methodologies or different methods are being used by the banks itself, if you need to transfer money to a third party, you need your account login details, you need your transactions ID, you need the transaction password, then use your option to enter the third password or a token information. So, there are multi levels of authentication which are performed on critical services. Let us look at Cross Site Scripting XSS, the attack vector is average, the prevalence is very widespread, the detectability is easy and the impact is moderate.

(Refer Slide Time: 07:22)



Cross-Site Scripting (XSS)

- ❑ Attacker sends malicious code to an innocent user's browser
- ❑ Malicious code might be...
 - ... reflected from web input (form or hidden field, URL, etc...)
 - ... stored in database
 - ... sent directly into rich JavaScript client
- ❑ Virtually every web application has this pr

Video inset showing a man speaking.

What is XSS? Attacker sends malicious code to an innocent users browser, the malicious code might be reflected from the web input, the form or hidden field or URL, it could have been stored in database, sent directly to the rich java script client. So, virtually every web application has this problem, normally java script commands are sent from the server to the browser and execute it. Now, XSS happens when an attacker tricks a logged in victim into entering the java script into his browser and submitting it to the server the server then dutifully echoes them to the browser, which the browser thinks that they are commands and executes them.

The attacker tricks the by browser into buying things, transferring money, delivering cookies, etcetera, etcetera. Now, the fix for this is to filter and encode all potentially dangerous characters. So, that the application does not return data that the browser will interpret as executable. any unescaped or uncoded data that is returned to the browser, this a very serious security risk. Now, there are characters like greater than, less than, colon, exclamation, open bracket, close brackets, square bracket, square close bracket, you know semi colon, colon, forward slash. So, there are several characters that need to be filter.

(Refer Slide Time: 09:19)



The slide features a green header with the title "Typical Impact" in yellow. Below the header, there is a list of four items, each preceded by a square icon with a checkmark. The first three items are "Steal user's session", "Steal sensitive data", and "Rewrite web page". The fourth item is "Redirect user to phishing or malware site". Below these, there is a section titled "Most Severe:" followed by a bullet point (a small yellow circle) that reads "Install XSS proxy which allows attacker to observe and direct all user's behavior on vulnerable sites and force user to other sites". In the bottom right corner of the slide, there is a small inset video frame showing a man's face.

- ☐ Steal user's session
- ☐ Steal sensitive data
- ☐ Rewrite web page
- ☐ Redirect user to phishing or malware site

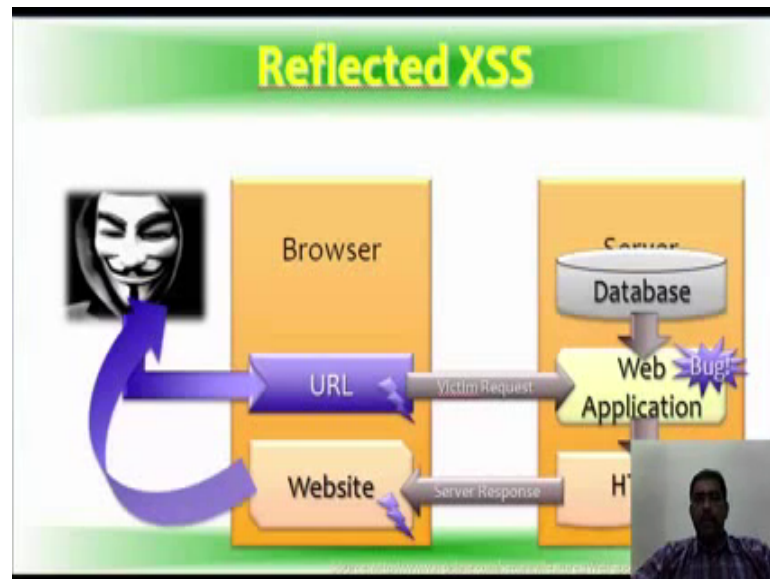
☐ Most Severe:

- Install XSS proxy which allows attacker to observe and direct all user's behavior on vulnerable sites and force user to other sites

What is the typical impact? It is steal a user's session, steal sensitive data, a rewrite the web page, redirect user to phishing or malware sites and the most severed is install XSS proxy, which allows attackers to observe and direct all users behavior on vulnerable sites and force user to other sites. Now, if you look at example let us take the example of bob analysis, which is very widely used in information security teachings.

Now, let say that bob is hosting a web site with a forum allowing users to post comments similar to a NPTEL web site. Now, AliceAlice force have comment with java script embedded in it, all users can now see this comment. AliceAlice visits or Mallory visits a site, bob's site beautifully prints Alice's comments which contain there java script. Mallory's browser, thinking that the java script should be executed, executes it... So, that java script is malicious and it causes havoc in the system.

(Refer Slide Time: 10:38)



There is a variation an XSS called reflected XSS. So, reflected XSS are those, where the injected script is reflected off the web server, such as in a error message, in a search result or any other response that may include some are all of the input sent to the server as a part of the request, reflected attacks a delivered to a victims via another route, such as from a email message or from some other sites.

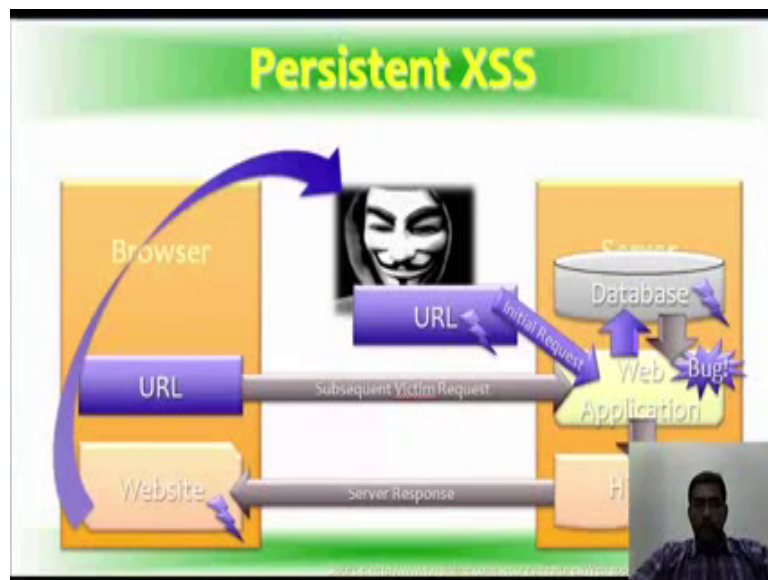
So, when a user is tricked into clicking up malicious link, submitting a typically crafted form or even just browsing to a malicious site, the injected code travels or traverses to the vulnerable web site which reflects the attack back to the users browser. The browser then executes the code, because it came from a trusted server, reflected XSS is also sometimes refer to as non persistent XSS or type 2 XSS. How the attackers do a reflected attack?

Let us again take the case of AliceAlice, Bob and Mallory, AliceAlice visits bobs web site which stores sensitive data. Now, after Alice logs in Mallory observe that bob's web sites is vulnerable to XSS attack Mallory crafts a URL to exploit the vulnerability and

sends an e mail to Alice enticing her to click a link for the URL under some false pretense, the URL will point to bobs web site, but will contain Mallory's malicious code. Alice clicks on Mallory's link while logged into bobs web site itself.

Now, the malicious script embedded in the URL executes in Alice's browser as if it came directly from bobs browser. So, that is why it is this is actually an XSS vulnerability and it is call the type two or a reflected XSS.

(Refer Slide Time: 13:09)



Then, there is something called a persistent XSS, the store attacks or those, where the injected script is permanently stored on the target servers. Such as, a data base in a message forum, in the visitors log, in the comment field and or the any other field. The victim then retrieves the malicious script from the server, when it request the store information. So, stored XSS is also sometimes called as persistent or type one XSS, this is much more insidious.

Since, in user does not see the attack on the page, so it effects XSS can effect hijacks also and we also have to filter untrusted input, to CSS also to HTML attributes, to java script, URL parameters or quarry sings, a this is all in addition to the HTML elements. So, you have to filter of the unnecessary characters, so where all it effects. So, what you have to notice any scripting language that the browser can process, is vulnerable to this kind of attacks.

(Refer Slide Time: 14:34)



Local XSS it is a gum base XSS or sometimes it is called a type 0 XSS also. So, this is an XSS attack, where the attack pay load is executed as a result of modifying the dome environment in the victims browser used by the original client side script. So, that the client side code runs in an unexpected manner that is the page itself, page itself is the Http response that is there does not change, but the client side code contained in the page executes differently due to malicious modifications, this is in contrast to the XSS attacks we have seen so far which is stored or reflected where the attack pay load is placed in the response page due to the server site flop.

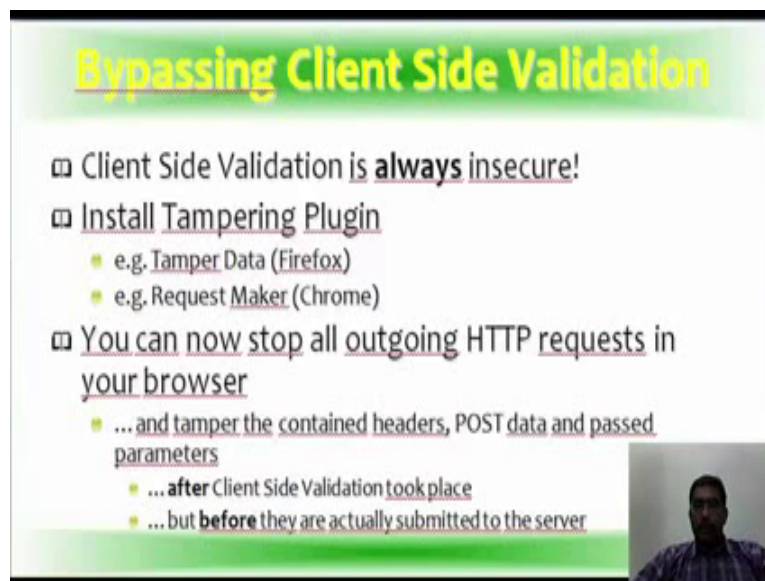
(Refer Slide Time: 15:37)

XSS Attack Patterns I

- ❑ Simple Patterns
 - `<SCRIPT>javascript:alert('XSS');</SCRIPT>`
 - ``
 - `<IFRAME SRC="javascript:alert('XSS');"></IFRAME>`
- ❑ Masked / Evasive Patterns
 - ``
 - `"!-"<XSS>=&{() }`
 - `<SCRIPT>alert("XSS");</SCRIPT>">`
 - ``
 - ``

The attack patterns of XSS is beyond again like I mentioned earlier the source is given on the bottom right, you can go to that URL and there is a XSS cheat sheet available then where all these scripts are there; the hackers dot org also has a page where they have classified from the I 6 or 7 on wards to the latest version and which particular attack works and which particular browser. So, it is a really good informative site, if you want to try out XSS on your own V M go to the site, download the cheat sheet and write down. So, a few more samples are given the cheat sheet, I have given the URL.

(Refer Slide Time: 16:32)



Bypassing Client Side Validation

- ❑ Client Side Validation is **always** insecure!
- ❑ Install Tampering Plugin
 - e.g. Tamper Data (Firefox)
 - e.g. Request Maker (Chrome)
- ❑ You can now stop all outgoing HTTP requests in your browser
 - ... and tamper the contained headers, POST data and passed parameters
 - ... after Client Side Validation took place
 - ... but before they are actually submitted to the server

Bypassing client site validation, client site validation if it is done is always insecure. Now, there are lot of plugging add-ons available for Mozilla, for Chrome, for Internet Explorer, tamper data is a very popular plug in for add on for fire fox, you can stop all out going HTTP request in your browser and tamper the contained headers, post data and passed parameters after the client site validation took place, but before they are actually submitted to the server.

So, once you enter the user name and password before after the client site validation happens, if it is a client site validation you install the plugging like or add on like tamper data trap that particular request you will know multiple things from there one what is the actually user name entered, you can alterate what is the password entered, if password is not encoded or encrypted you will come to know the password enclintense. If it is encoded you can further go and see what kind of encryption is used.

So, there are many things that can be done, the most popular one is tamper data request maker the same thing you can do it in work suit also, you need to install it as proxy on port 8008 and then use you bug you can still do the same thing.

(Refer Slide Time: 17:58)

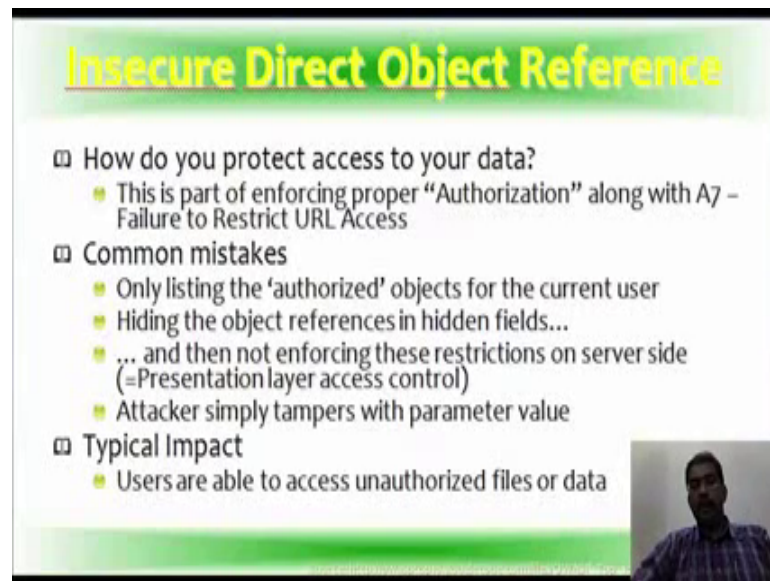
Protection

- ❑ **Eliminate XSS**
 - Don't include user supplied input in your output!
- ❑ **Defend against XSS**
 - Output Encode all user supplied input
 - OWASP Enterprise Security API
 - For GWT: com.google.gwt.safehtml.shared.SafeHtml
 - Perform White List Input Validation on user input
 - Use an HTML Sanitizer for larger user supplied HTML chunks
 - OWASP Java HTML Sanitizer

What is the protection mechanism for XSS or how to eliminate XSS, first thing is do not include user supplied input in your output; how the defend against XSS, output encode all user supplied in input, there is OWASP enterprise security API use it for GWT for Google use that perform white list input validation on user input, we have seen that use and HTML sanitizer for larger user supplied HTML chunks, so that is OWASP java HTML sanitizer.

Now, for your information ASP dot net comes with XSS protection enabled by default, you can turn it off also in the web dot config file. We will now go to A 4 which is insecure direct object reference, the attack vector is easy, the prevalence is common, the detectability it is easy and the impact is moderate.

(Refer Slide Time: 19:04)

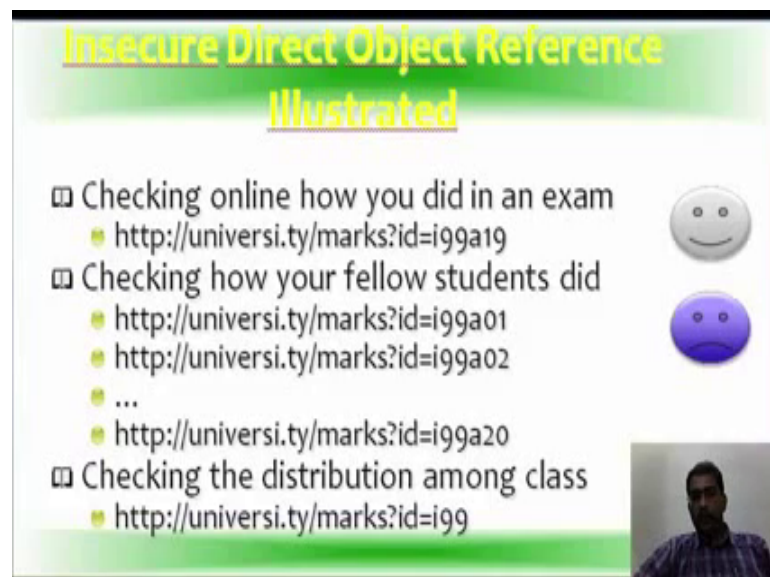


Insecure Direct Object Reference

- ❑ How do you protect access to your data?
 - This is part of enforcing proper “Authorization” along with A7 – Failure to Restrict URL Access
- ❑ Common mistakes
 - Only listing the ‘authorized’ objects for the current user
 - Hiding the object references in hidden fields...
 - ... and then not enforcing these restrictions on server side (=Presentation layer access control)
 - Attacker simply tampers with parameter value
- ❑ Typical Impact
 - Users are able to access unauthorized files or data

Now, how do you protect access to your data? This is a part of enforcing proper authorization along with A 7 that is A 7 of OWASP, which is failure to restrict URL access. The common mistakes that have done are only listing the authorized objects for current user, hiding the object references in the hidden fields and then not enforcing these restrictions on the server side that the presentation layer has access control, attacker simply tampers with the parameter value.

(Refer Slide Time: 19:49)



Insecure Direct Object Reference Illustrated

- ❑ Checking online how you did in an exam
 - <http://universi.ty/marks?id=i99a19>
- ❑ Checking how your fellow students did
 - <http://universi.ty/marks?id=i99a01>
 - <http://universi.ty/marks?id=i99a02>
 - ...
 - <http://universi.ty/marks?id=i99a20>
- ❑ Checking the distribution among class
 - <http://universi.ty/marks?id=i99>

What is typical impact users are able to access unauthorized files of data that is the impact. Now, if I have given an example now your checking online on how you did in an

exam. So, you have HTTP university slash marks id equal to something, checking how your fellow students did you are not supposed to see how your fellow student did.

So, you are giving the ID equal to your changing and saying, checking the distribution among class. So, there also the URL is given, now what actually happens is eliminate the, protection is, eliminate the direct object references if you see the previous slide, this is a wrong way of doing it. Because, if I know your ID I can go and in the URL I can put it and then I can see what marks you have scored or what the other person has scored. So, there is a direct objects reference there, vulnerability there.

(Refer Slide Time: 20:43)

Protection

- ❑ **Eliminate the Direct Object References**
 - Replace with temporary mapping value
 - e.g. with ESAPI AccessReferenceMap
- ❑ **Validate the Direct Object Reference**
 - Verify parameter value format
 - **Verify user authorization to access target object**
 - Query Constraints work great! (Data Access Restriction)
 - Verify the requested mode of access is allowed to target object (read, write, delete, ...)

Now, eliminate the direct object of reference replace with temporary mapping value example is OWSAP ESAPI access reference map validate the direct object reference, verify the parameter value format, verify user authorization to access the target object, then query constraints work great data access restriction, verify the requested mode of access is allowed to the target object, read, write, delete.

Now, this what you have to note here is this is not the method that is to be used id equal to, it should be in a form where a normal attacker or an attacker cannot decipher very simply and do a insecure direct object prevalence. Then we go to security misconfiguration, here the attack vector is easy, the prevalence common, detectability is easy and impact is moderate.

(Refer Slide Time: 21:47)



Security Misconfiguration

- ❑ Web applications rely on a secure foundation
 - Everywhere from the OS up through the App Server
 - Don't forget all the libraries you are using!
- ❑ Is your source code a secret?
 - Think of all the places your source code goes
 - Security should not require secret source code
- ❑ Do you change **all** credentials regularly in production environment?

© 2015 Cisco and/or its affiliates. All rights reserved. Cisco Confidential


Application misconfiguration attacks exploit the configuration weakness in the web applications. So, web applications rely on a secured foundation, everywhere right from the OS up through the app server. So, do not forget the libraries also that you are using, libraries also can be vulnerable. Is your source code a secret, think of all the places that your source code will go, security should not require secret source code.

Do you change all credentials regularly in a production environment, these are things to ponder upon. Misconfiguration is defined as a configuration mistake that results in unintended application behavior that includes misuse of default passwords, misuse of privileges and excessive debugging information distortion. The effect of misconfiguration can lead to service outages, loss of sensitive data or any other serious problem that cannot occur.

(Refer Slide Time: 22:57)

Typical Impact

- ❑ Install backdoor through missing OS or server patch
- ❑ XSS flaw exploits due to missing application framework patches
- ❑ Unauthorized access to default accounts, application functionality or data, or unused but accessible functionality due to poor server configuration




What is the typical impact, it is all about back door through missing OS or server patch. Now, XSS flaws exploits due to missing application frame work patches, unauthorized access to default accounts, application functionality or data or unused, but accessible functionality due to poor server configuration. Now, all these can be severe you can have a partial or full data lose that could be data modification, there could be a compromise of the full system and recovery is going to be very, very expensive.

(Refer Slide Time: 23:35)

Protection

- ❑ Verify your system's configuration management
 - Secure configuration "hardening" guideline
 - Automation is really useful here
 - Must cover entire platform and application
 - Keep up with patches for all components
 - This includes software libraries, not just OS and Servers!
 - Analyze security effects of changes
- ❑ Deactivate unnecessary stuff
 - Ports, Services, Accounts, Sites, ...
- ❑ Verify the implementation
 - Scanning finds generic configuration and missing patch proble



The protection for this verify systems configuration management, secure configuration hardening guideline. So, hardening we have discussed under OS domain 5, automation is really useful is here a large number of servers and you need to do hardening then

automation is really useful, you must cover the entire platform and application keep up with patches for all components. So, just by installing a patch for your Linux server or Windows server does not mean that your system is no longer vulnerable.

You may be using some insecure components or you may be requirement some framework which is older version and patches have been released and you are not updated, so update those batches. So, analyze security effect of changes, so before you actually put it in production, test it and see what is the effect on not only the security, but also the functionality. Deactivate or disable unnecessary stuff like the ports that you do not required block it, services that you do not required block it, the default accounts that come with it disable it and if multiple sites are there, test site is put and all disable it then verify the implementation.

So, you do a scanning to find if it is a generic configuration or if there is a missing patch problem. So, if you ask the question I am at risk? Then you will have all the following questions; is the software out of date or any unnecessary features enabled or default accounts and associated credentials unchanged does error handling reveal traces to users, stack traces to users are the security setting not set to secure values or are the securities setting set to insecure values.

Now, is a default installation being used, is there a consistency maintained between versions or configurations then is the default configuration option restricted are all the non required ports block or the roles and privileges restricted and is the configuration centralized, then has a scan or audit been done on the particular sight, whether strong encryption has been used. So, you can probably add more to this, but more or less this is what you need to do.