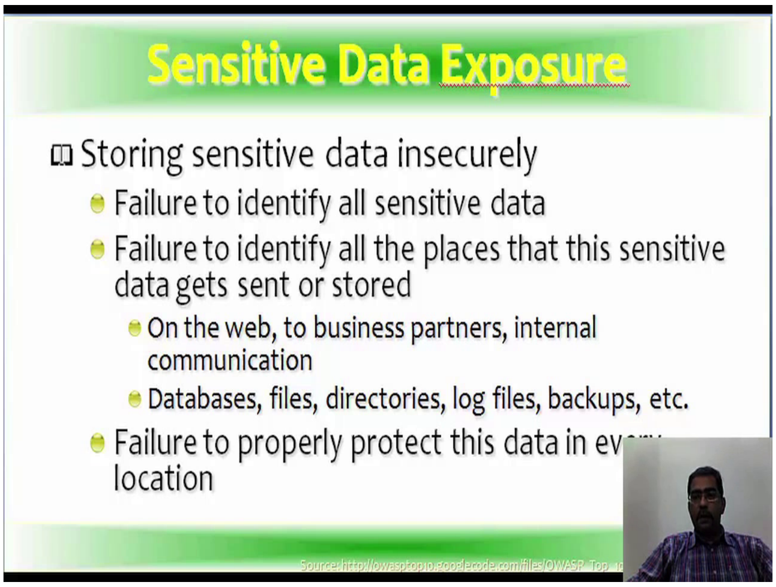
Introduction to Information Security

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

Lecture - 64

A 6 of OWASP is this sensitive data exposure.

(Refer Slide Time: 00:15)



Sensitive Data Exposure

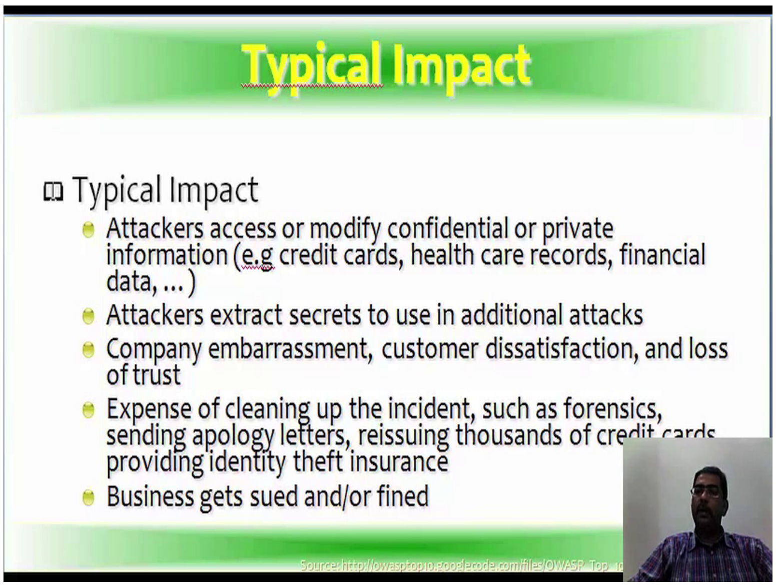
- Storing sensitive data insecurely
  - Failure to identify all sensitive data
  - Failure to identify all the places that this sensitive data gets sent or stored
    - On the web, to business partners, internal communication
    - Databases, files, directories, log files, backups, etc.
  - Failure to properly protect this data in ev location

Source: http://owasptop10.googlecode.com/files/OWASP_Top

So, what is sensitive data exposure? Storing sensitive data insecurely, failure to identify all sensitive data, failure to identify all the places that the sensitive data gets sent or stored, it can be on the web to business partners, internal communications, data files, directories, log files, backups. So, all of those things are covered in this, failure to properly protect this data in every location.

So, the sensitive data is the organization should determine what this sensitive data is user name and password is sensitive, credit card numbers are sensitive, bank account details are sensitive. So, invoices, confidential some of the invoices to the business partners or sensitive, the other business partners should not know if two are providing the same services, the data base should not store data insecurely, the lock file should not display sensitive information.

(Refer Slide Time: 01:25)



Typical Impact

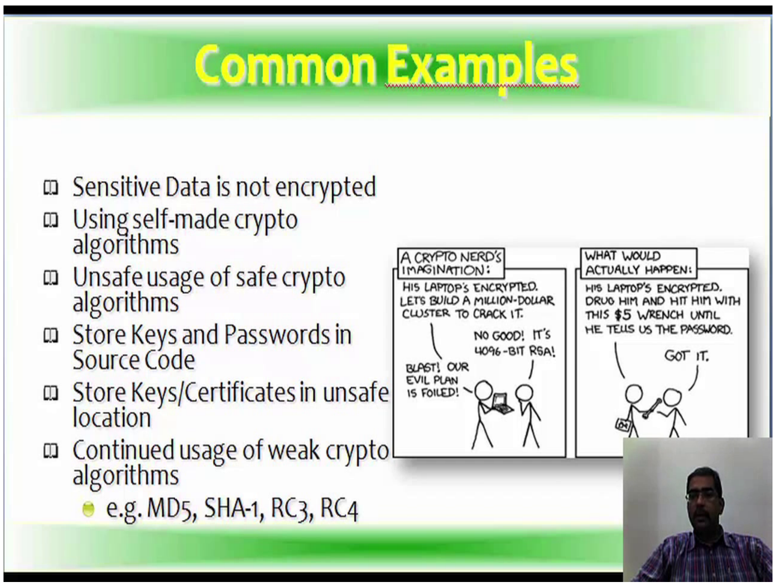
- Typical Impact
  - Attackers access or modify confidential or private information (e.g credit cards, health care records, financial data, …)
  - Attackers extract secrets to use in additional attacks
  - Company embarrassment, customer dissatisfaction, and loss of trust
  - Expense of cleaning up the incident, such as forensics, sending apology letters, reissuing thousands of cre
providing identity theft insurance
  - Business gets sued and/or fined

Source: http://owasptop10.googlecode.com/files/OWASP_Top

What is the impact of sensitive data exposure? Attackers access or modify the confidential or private information, like I said credit cards, health care records, financial data, attackers extract secrets to use in additional attacks. So, based on the information that they get the launch other attacks. Company embarrassment that is loss of reputation, embarrassment, customer dissatisfaction, loss of trust all these are consequences of sensitive data exposure.

Then the cost of cleaning up the incidence, such as forensics, sending apology letters to customers, reissuing thousands of credit cards providing identity theft insurance. The business can get sued or fined by regulators for not following or for exposing sensitive data.

(Refer Slide Time: 02:18)



Common Examples

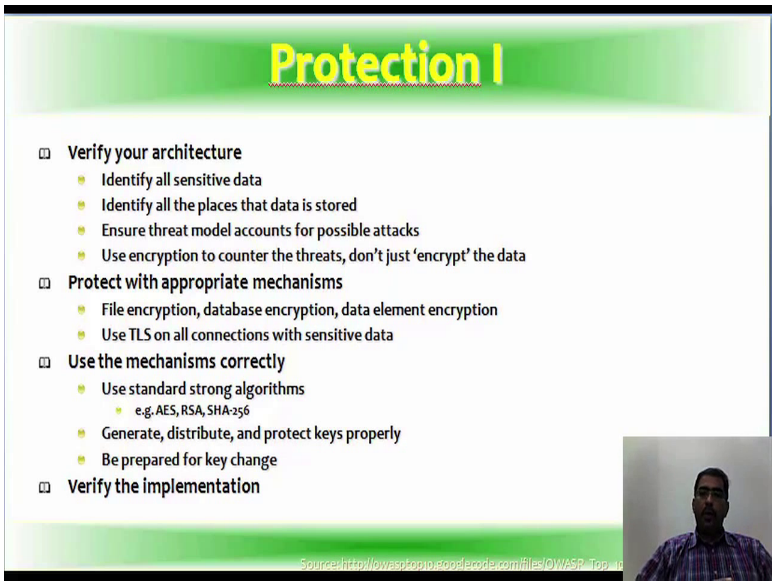
- Sensitive Data is not encrypted
- Using self-made crypto algorithms
- Unsafe usage of safe crypto algorithms
- Store Keys and Passwords in Source Code
- Store Keys/Certificates in unsafe location
- Continued usage of weak crypto algorithms
  - e.g. MD5, SHA-1, RC3, RC4

A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LETS BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

Some of the examples are common example is sensitive data is not encrypted like I say the user name password has a simple example, if it goes unencrypted, then it is a issue. Using self made crypto algorithms is not a good idea, unsafe usage of safe crypto algorithms, then store keys and passwords in source code sometimes in some sides and few year back we have seen the user name and password were hard coded in the source code.

Then, store keys or certificate in unsafe location where it can be retried easily, then if you continue use weak crypto algorithms, like MD 5, SHA 1, RC 3, RC 4, etcetera. Now, on the right hand side there is a nice cartoon, a crypto nerds imagination his laptop is encrypted, let us build a million dollar cluster to crack it. So, the other guy said no it is not a good idea with 4096 bit RSA. So, this guy says oh blast our evil plan is foiled, but what would actually happen? his laptop is encrypted drug him, hit him with this five dollar wrench until he tells us the password. So, that is the easier method to take over data.

(Refer Slide Time: 03:50)



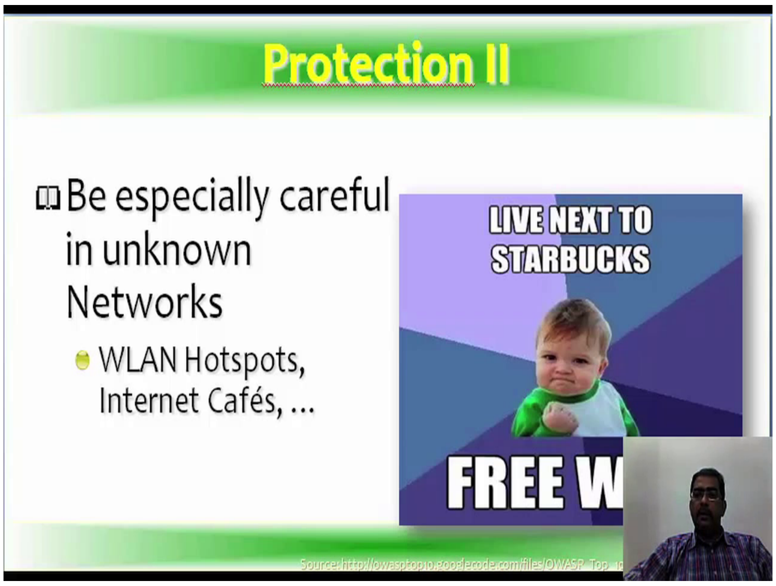
Protection I

- Verify your architecture
  - Identify all sensitive data
  - Identify all the places that data is stored
  - Ensure threat model accounts for possible attacks
  - Use encryption to counter the threats, don't just 'encrypt' the data
- Protect with appropriate mechanisms
  - File encryption, database encryption, data element encryption
  - Use TLS on all connections with sensitive data
- Use the mechanisms correctly
  - Use standard strong algorithms
    - e.g. AES, RSA, SHA-256
  - Generate, distribute, and protect keys properly
  - Be prepared for key change
- Verify the implementation

Source: http://owasptop10.googlecode.com/files/OWASP Top

What is the protection for sensitive data exposure; verify your architecture, identify all sensitive data, identify all the places that the data is stored, ensure threat model accounts for possible attacks, use encryption to counter the threat, do not just encrypt the data. Protect with appropriate mechanisms, file encryption is a mechanism, data base encryption, data element encryption, use TLS on all connections with sensitive data, use the mechanisms correctly means use standard strong algorithms like AES, RSA, SHA, SHA 512, generate distribute and protect the keys properly, be prepared for the key advantage or key change, verify the implementation.

(Refer Slide Time: 04:53)



Protection II

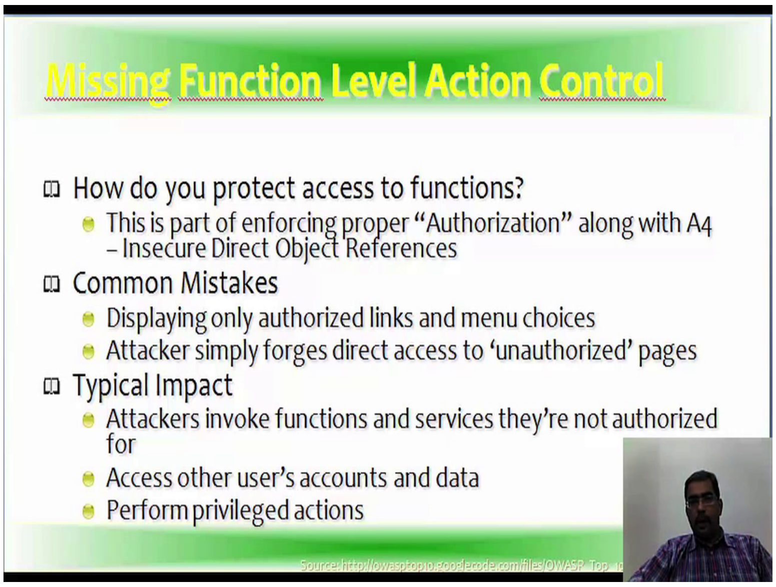
- Be especially careful in unknown Networks
  - WLAN Hotspots, Internet Cafés, ...

LIVE NEXT TO STARBUCKS
FREE W

Source: http://owasptop10.googlecode.com/files/OWASP Top

Be especially careful in unknown networks like WLAN hot spots, internet cafes. So, do not expose sensitive information when you are in a free WI-FI zone. A7 deals with missing function level action control, the attack vector is easy, the prevalance is common, detectivity is average and impact is moderate.

(Refer Slide Time: 05:24)



Missing Function Level Action Control

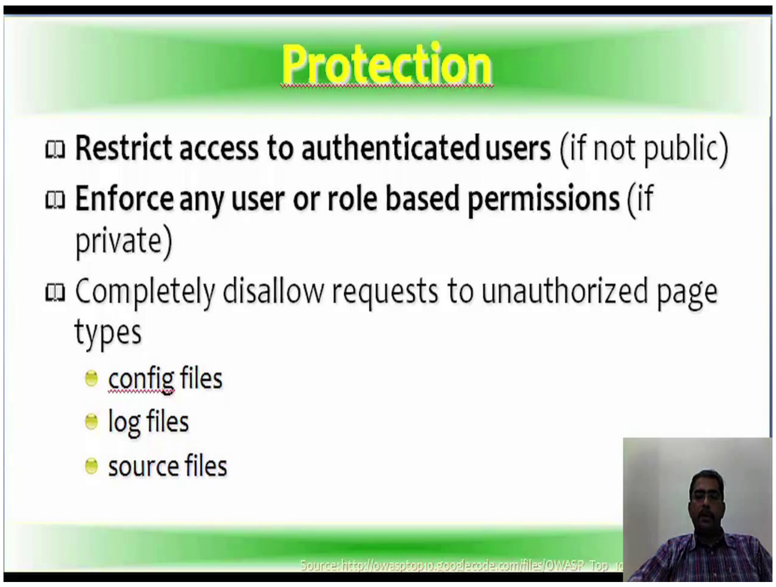
- How do you protect access to functions?
  - This is part of enforcing proper "Authorization" along with A4 – Insecure Direct Object References
- Common Mistakes
  - Displaying only authorized links and menu choices
  - Attacker simply forges direct access to 'unauthorized' pages
- Typical Impact
  - Attackers invoke functions and services they're not authorized for
  - Access other user's accounts and data
  - Perform privileged actions

Source: http://owasptop10.googlecode.com/files/OWASP_Top_10

How do you protect access to functions that is the key question. This is the part of enforcing proper authorization along with A 4, A 4 is insecure direct object references. The common mistakes that are done here are; displaying only authorizing links and menu choices, attacker simply forges direct access to unauthorized pages, typical impact is attacker invokes functions and services that they are not authorized for, access other users accounts and data, perform privileged actions.

So, if you take the example displaying only authorized links and denote choices, if insecure direct object reference vulnerability is there. So, it does not matter whether they are displayed only authorize links menus, you can actually because of the A 4 vulnerability you can actually access function, which you are not supposed to access of the rest.

(Refer Slide Time: 06:31)



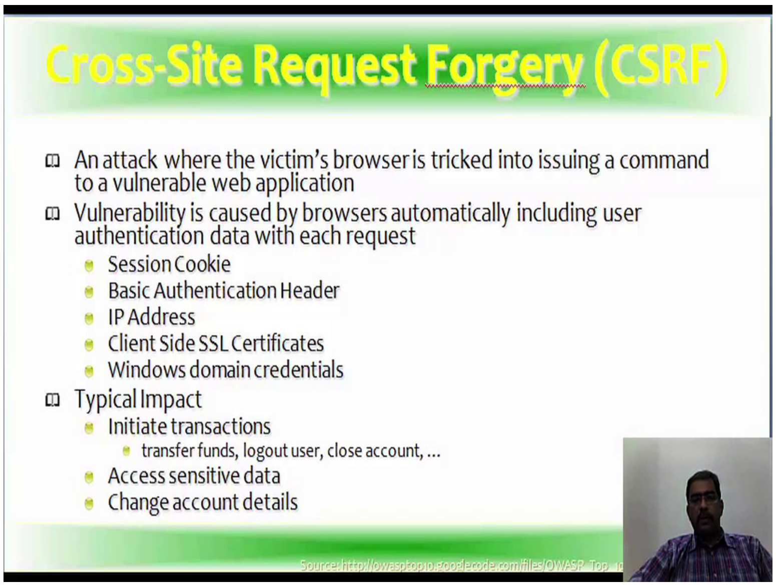
Protection

- **Restrict access to authenticated users** (if not public)
- **Enforce any user or role based permissions** (if private)
- Completely disallow requests to unauthorized page types
  - config files
  - log files
  - source files

Source: http://owasptop10.googlecode.com/files/OWASP Top

The protection for this is restrict access to authenticated users, if not public. Enforce any user or role based permissions, if it is private, completely disallow request to unauthorized page or page types, like config files, the log files, the source files. Then, we come to A 8 which is CSRF of Cross Site Request Forgery, the attacker vector is average, prevalance is common, from detectability is easy and impact is moderate.

(Refer Slide Time: 07:07)



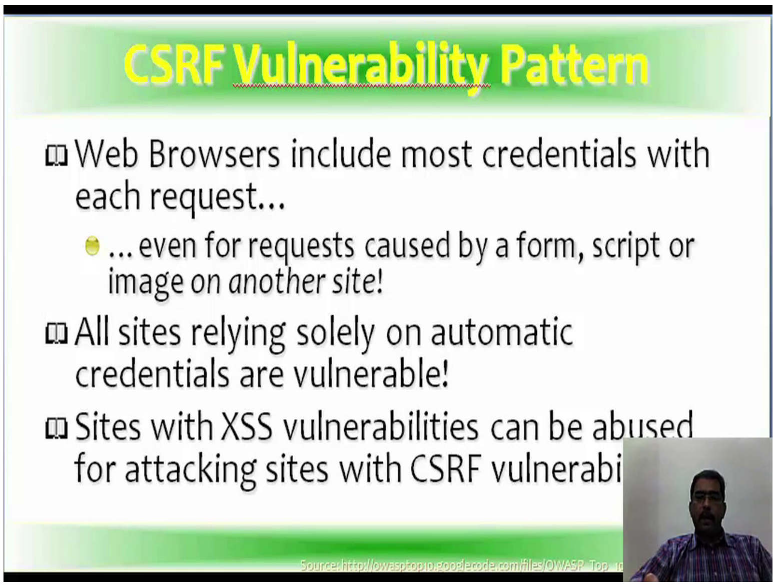
Cross-Site Request Forgery (CSRF)

- An attack where the victim's browser is tricked into issuing a command to a vulnerable web application
- Vulnerability is caused by browsers automatically including user authentication data with each request
  - Session Cookie
  - Basic Authentication Header
  - IP Address
  - Client Side SSL Certificates
  - Windows domain credentials
- Typical Impact
  - Initiate transactions
    - transfer funds, logout user, close account, ...
  - Access sensitive data
  - Change account details

Source: http://owasptop10.googlecode.com/files/OWASP Top

CSRF is when an attack an attack where the victims browser is tricked into issuing a command to a vulnerable web application. Vulnerability is caused by browsers automatically including user authentication data with each request, which can be a session cookie, basic authentication header, the IP address, client side SSL certificates, Windows domain credentials and what are the typical impacts, you can initiate transactions, transfer funds, log out the users, close accounts, access sensitive data, change the account details with CSRF you can do all these things.

(Refer Slide Time: 07:49)



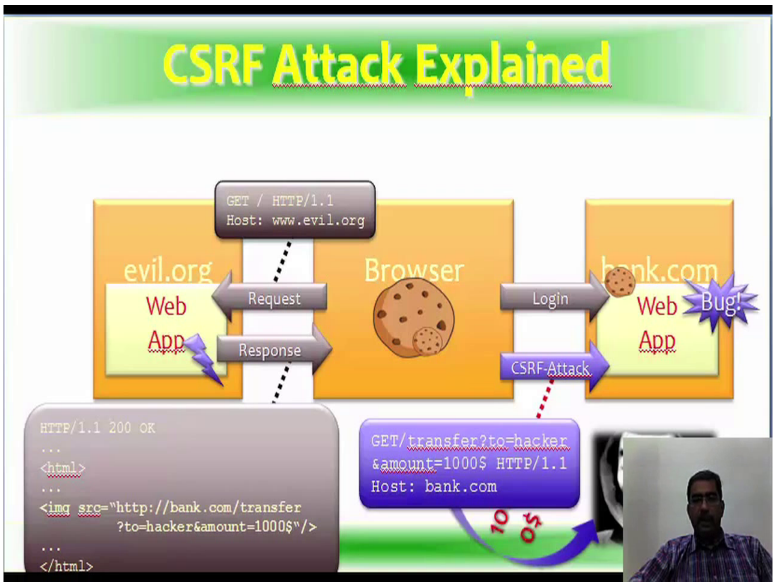
CSRF Vulnerability Pattern

- Web Browsers include most credentials with each request...
  - ... even for requests caused by a form, script or image *on another site!*
- All sites relying solely on automatic credentials are vulnerable!
- Sites with XSS vulnerabilities can be abused for attacking sites with CSRF vulnerabi

Source: http://owasptop10.googlecode.com/files/OWASP_Top_1

The pattern is web browsers include most credentials with, each request even for request caused by a form or script or image on another site, all sites relaying solely on automatic credentials are vulnerable, sites with XSS vulnerabilities can be abused for attacking sites with CSRF vulnerabilities.

(Refer Slide Time: 08:14)



CSRF Attack Explained

```
GET / HTTP/1.1
Host: www.evil.org
```

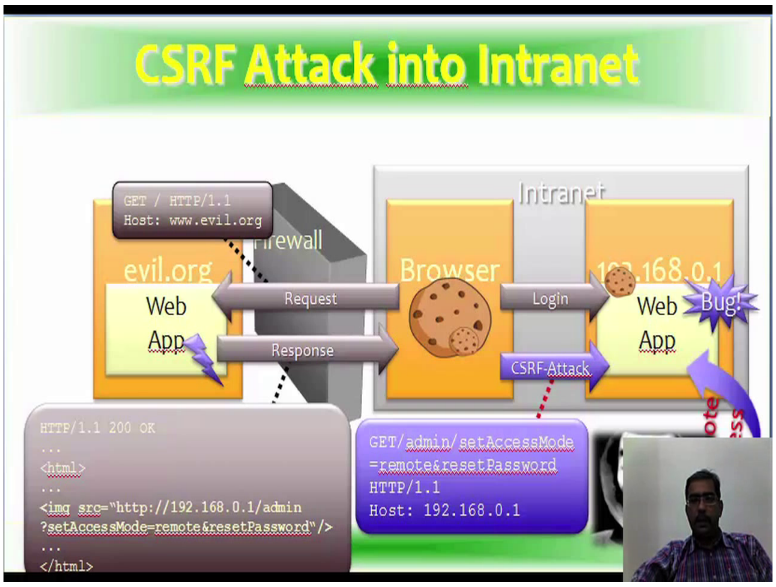
```
HTTP/1.1 200 OK
...
<html>
...
<img src="http://bank.com/transfer
        ?to=hacker&amount=1000$"/>
...
</html>
```

```
GET/transfer?to=hacker
&amount=1000$ HTTP/1.1
Host: bank.com
```

Now, if you look at this slide you will see how a CSRF attack happens with the browser, there is bank dot com on the right hand side and evil dot org on the left hand side. Now, from the browser you log into the web app in bank dot com there is bug there. So, what basically happens is a CSRF attack is done on that what is that attack get transferred to hacker and amount equal to 1000 dollars http one dot one and host bank .

Now, that is one series stored there then from the attackers side he actually tampers that particular website, so common thing here is the browser on the victim site, now he uses a image source command and then exploits that particular bug by making a transfer or requesting a transfer from the bank site to his particular account.

(Refer Slide Time: 09:24)



CSRF Attack into Intranet

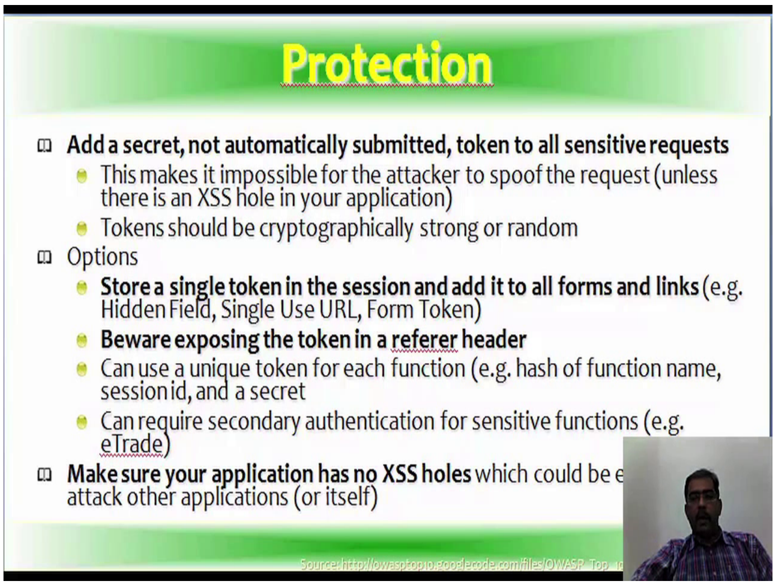
```
GET / HTTP/1.1
Host: www.evil.org
```

```
HTTP/1.1 200 OK
...
<html>
...
<img src="http://192.168.0.1/admin
?setAccessMode=remote&resetPassword"/>
...
</html>
```

```
GET/admin/setAccessMode
=remote&resetPassword
HTTP/1.1
Host: 192.168.0.1
```

This is another example CSRF attack into the intranet. So, the browser request to 192 168 0.1 web app the logs in; the attacker through remote access, exploited the client machine and sends a request response, what is it remote and reset password. So, using the same image source command he does a password reset on the intranet sitting outside.

(Refer Slide Time: 09:56)



Protection

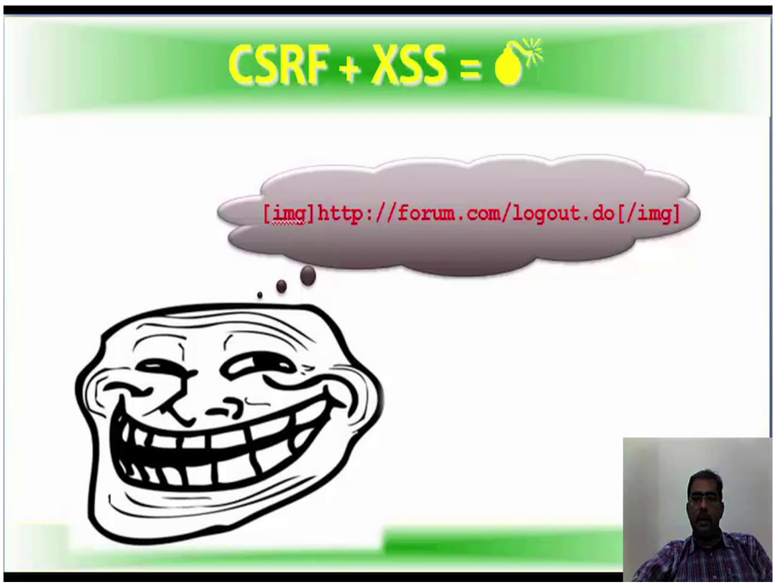
- Add a secret, not automatically submitted, token to all sensitive requests
  - This makes it impossible for the attacker to spoof the request (unless there is an XSS hole in your application)
  - Tokens should be cryptographically strong or random
- Options
  - **Store a single token in the session and add it to all forms and links** (e.g. Hidden Field, Single Use URL, Form Token)
  - **Beware exposing the token in a referer header**
  - Can use a unique token for each function (e.g. hash of function name, session id, and a secret
  - Can require secondary authentication for sensitive functions (e.g. eTrade)
- **Make sure your application has no XSS holes** which could be e attack other applications (or itself)

Source: http://owasptop10.googlecode.com/files/OWASP Top

What is the protection for CSRF add a secret, not automatically submitted token to all sensitive request. So, this makes it impossible for the attacker to spoof the request, unless there is the XSS hole in the application. Tokens should be cryptographically strong or it should be random, not easily decipherable, options are store a single token in the session and add it all forms and links, example in the hidden fields, single use URL's, form token.

Beware exposing the token in the referer header can use unique token for each function example, hash of function name, session id and a secret. Can require secondary authentication for sensitive functions, example it doing a eTrade. Make sure your application has no XSS holes which could be exploiting the attack other applications or itself.

(Refer Slide Time: 10:57)


CSRF + XSS =

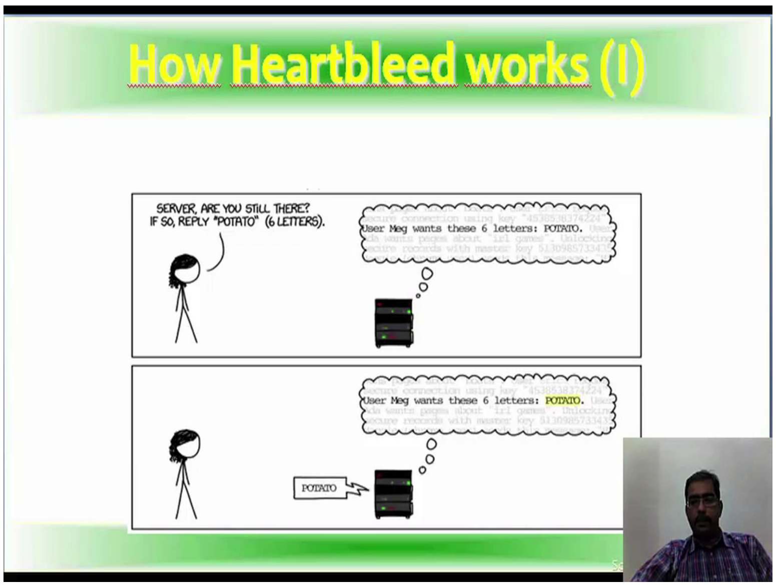
So, CSRF plus XSS is a atom bomb or dynamite, we come to A 9 which is using components or known vulnerable components attack vectors is average, prevalence is widespread, detectability is difficult and impact is moderate.

(Refer Slide Time: 11:20)

Exercise: Name the Vulnerability!
OpenSSL
1.0.1 – 1.0.1f
Heartb

We will see what A 9 is, now you all heard about heart bleed which is vulnerability in the open SSL version 1.0.1 to 1.0.1 f. So, this comes under using known vulnerable components.

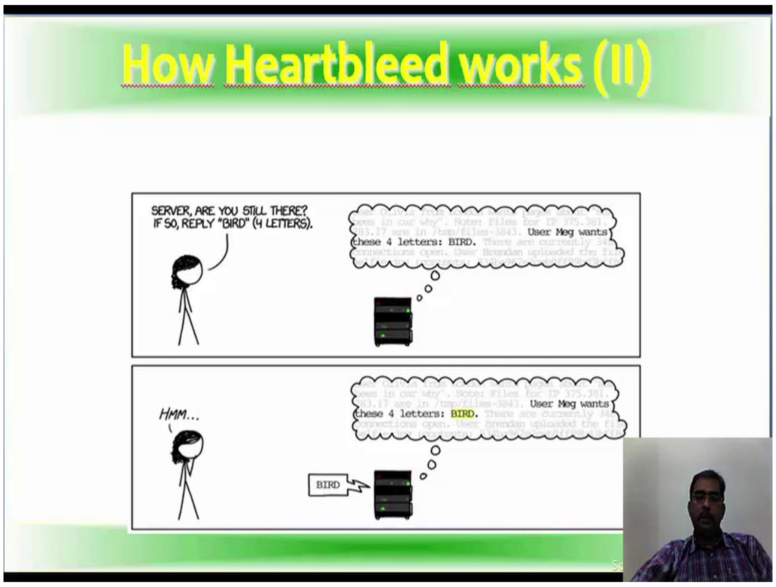
(Refer Slide Time: 11:38)


How Heartbleed works (I)

SERVER, ARE YOU STILL THERE? IF SO, REPLY "POTATO" (6 LETTERS).
User Meg wants these 6 letters: POTATO.
POTATO

How heart bleed works, now server are you still there if so reply potato; that is what the URL is asking, user Meg wants these six letters potato. So, it responds back the potato.

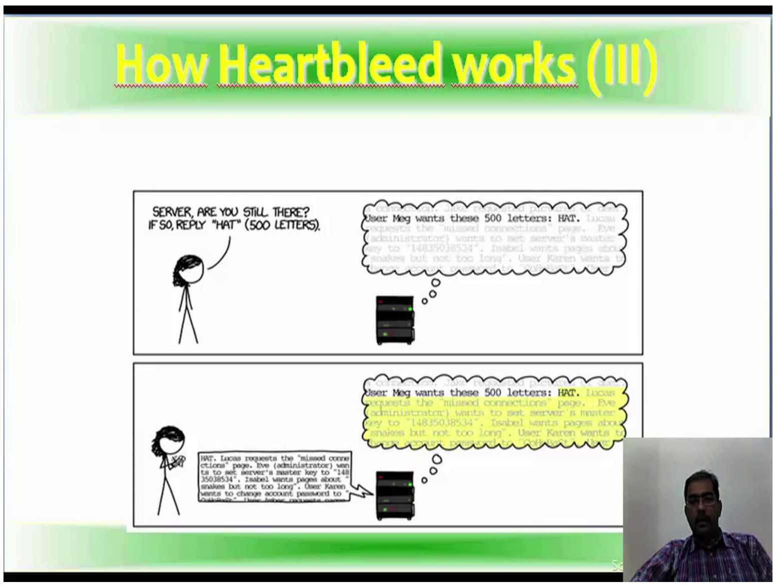
(Refer Slide Time: 11:55)


How Heartbleed works (II)

SERVER, ARE YOU STILL THERE? IF SO, REPLY "BIRD" (4 LETTERS).
User Meg wants these 4 letters: BIRD.
HMM...
BIRD

Then the user or server are you still there if so reply bird, 4 four letters. So, theses four letters are bird user Meg wants these four letters, it responds back or it responds back as bird.

(Refer Slide Time: 12:10)

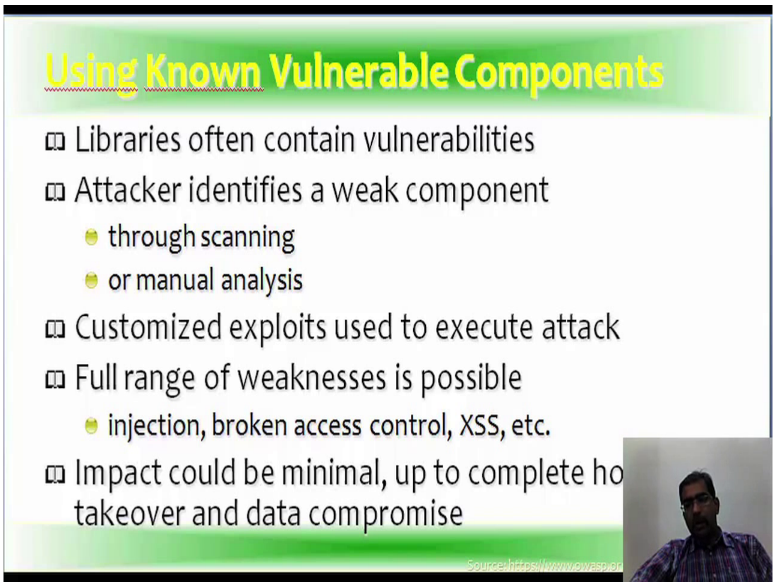


How Heartbleed works (III)

SERVER, ARE YOU STILL THERE? IF SO, REPLY "HAT" (500 LETTERS).
User Meg wants these 500 letters: HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about "snakes but not too long". User Karen wants to change account password to "CoHoBaSt". User
HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about "snakes but not too long". User Karen wants to change account password to "CoHoBaSt".

Then, this similarly the user requests for HAT. So, here if you can see HAT 500 liters user Meg want these 500 liters hat. So, it brings out all junks and gives it to her.

(Refer Slide Time: 12:34)



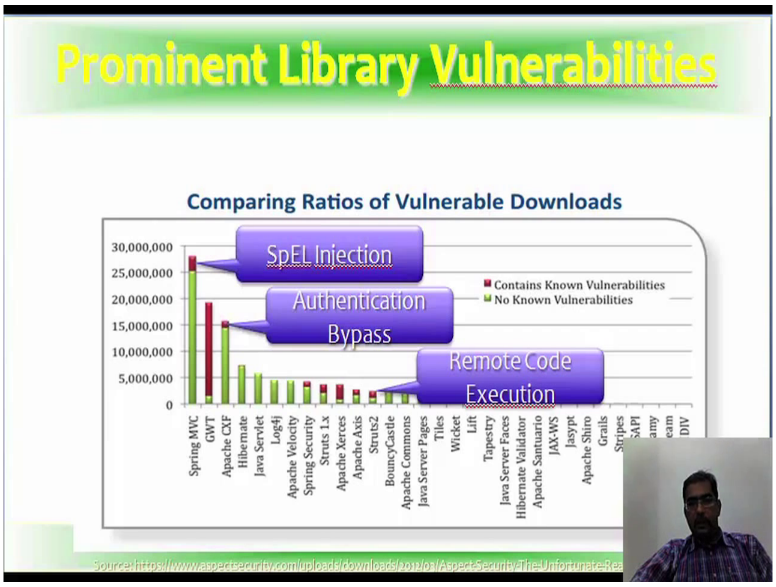
Using Known Vulnerable Components

- Libraries often contain vulnerabilities
- Attacker identifies a weak component
  - through scanning
  - or manual analysis
- Customized exploits used to execute attack
- Full range of weaknesses is possible
  - injection, broken access control, XSS, etc.
- Impact could be minimal, up to complete ho takeover and data compromise

Source: https://www.owasp.or

So, what basically happens is the user tries to trip the server into showing something that it is not supposed to issue. Now, libraries often contain certain vulnerabilities, attacker identifies the weak component, through scanning or through manual analysis, then customize exploits used to execute the attack, full range of weaknesses is possible in this, injection, broken access control, XSS all of this impact could be minimal up to complete host take over and data compromising, meaning it could be as little as the impact being minimal or you can go up to being very critical like host take over or compromise on data.

(Refer Slide Time: 13:22)



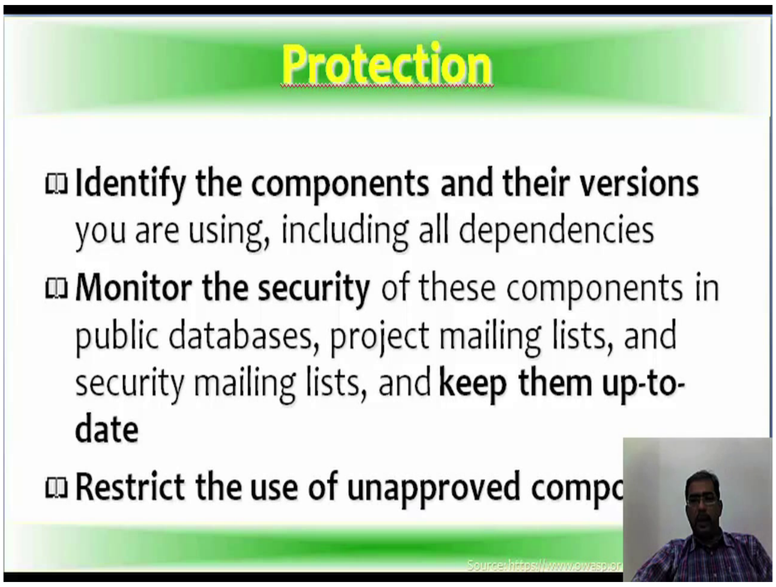
Prominent Library Vulnerabilities

Source: https://www.aspectsecurity.com/uploads/downloads/2012/03/Aspect-Security-The-Unfortunate-Rea

Now, this is a small statics on the prominent library vulnerability, if you see authentication bypass, remote code execution, SpEL injection, if you see the red contain no known vulnerabilities and green is no known vulnerabilities, contains known vulnerabilities and does not contain no known vulnerabilities. So, you can see the products which are there below are the frame works, which are effected most.

Then we go to the protection.

(Refer Slide Time: 13:56)



Protection

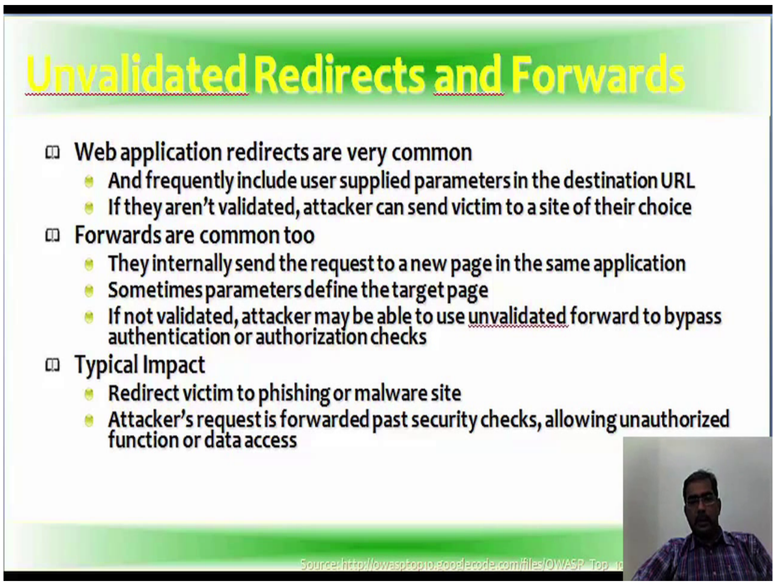
- **Identify the components and their versions** you are using, including all dependencies
- **Monitor the security** of these components in public databases, project mailing lists, and security mailing lists, and **keep them up-to-date**
- **Restrict the use of unapproved compo**

Source: https://www.owasp.or

Identify the components and their versions that you are using including all dependencies, monitor the security of these components in public data basis, project mailing list and security mailing list keep them up to date, restrict the use of unapproved components and example of this is, if a dependency is repaired for installing a software let us say that particular software requires a Visual C framework in Windows or say a PHP add on or dependency satisfying component, which is required in a Unix server.

Before you use that keep you them up to date install the latest version of the dependency and make sure that they are not vulnerable. So, in that way you can address this issues, the last one is unvalidated redirects and forwards, the attack vector is average, prevalence is uncommon, detectability is easy and impact is moderate.

(Refer Slide Time: 15:06)



Unvalidated Redirects and Forwards

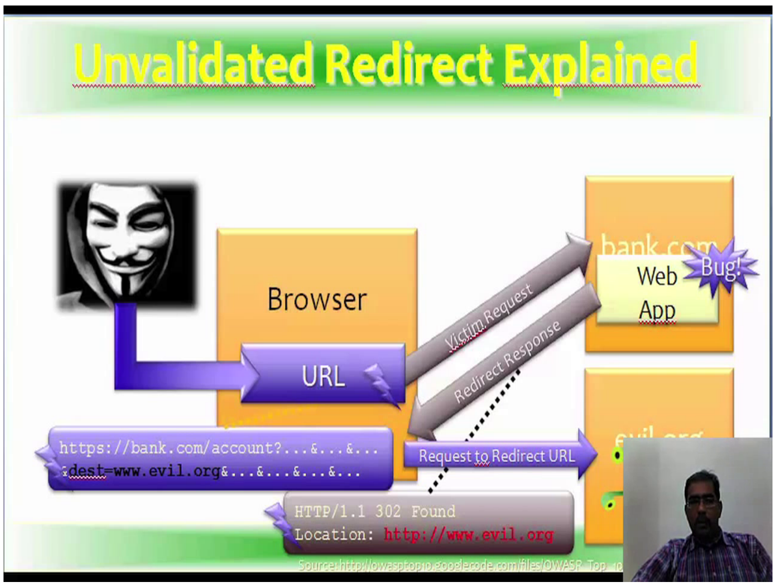
- Web application redirects are very common
  - And frequently include user supplied parameters in the destination URL
  - If they aren't validated, attacker can send victim to a site of their choice
- Forwards are common too
  - They internally send the request to a new page in the same application
  - Sometimes parameters define the target page
  - If not validated, attacker may be able to use unvalidated forward to bypass authentication or authorization checks
- Typical Impact
  - Redirect victim to phishing or malware site
  - Attacker's request is forwarded past security checks, allowing unauthorized function or data access

Source: http://owasptop10.googlecode.com/files/OWASP_Top_10

Now, what unvalidated redirects and forwards is all about, web application redirects are very common and frequently those include user supplied parameters in the destination URL, if they are not validated, attacker can send victim to a site of their choice. Forwards are also common, they internally send the request to a new page in the same application, sometimes parameters define the target page, if it is not validated the attacker may be able to use unvalidated forward to bypass the authentication or authorization checks.

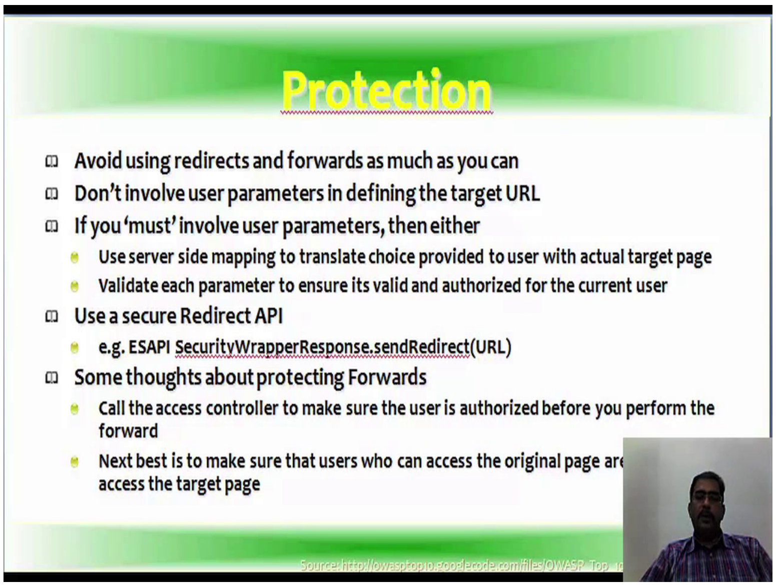
The impact of this is redirect the victim to the phishing or malware site, attackers request is forwarded past the security check allowing unauthorized function or data access.

(Refer Slide Time: 15:58)


Unvalidated Redirect Explained
Browser
URL
Victim Request
Redirect Response
bank.com
Web App
Bug!
evil.org
Request to Redirect URL
https://bank.com/account?...&...&...
&dest=www.evil.org&...&...&...&...
HTTP/1.1 302 Found
Location: http://www.evil.org

Now, if you see an example of this, the attacker hacks into your system from the victims browser, the victims to the bank dot com request is sent, there is a redirect response from the bank dot com, the request is redirected to evil dot org and then what ever malicious things the hacker wants to do is done.

(Refer Slide Time: 16:28)



Protection

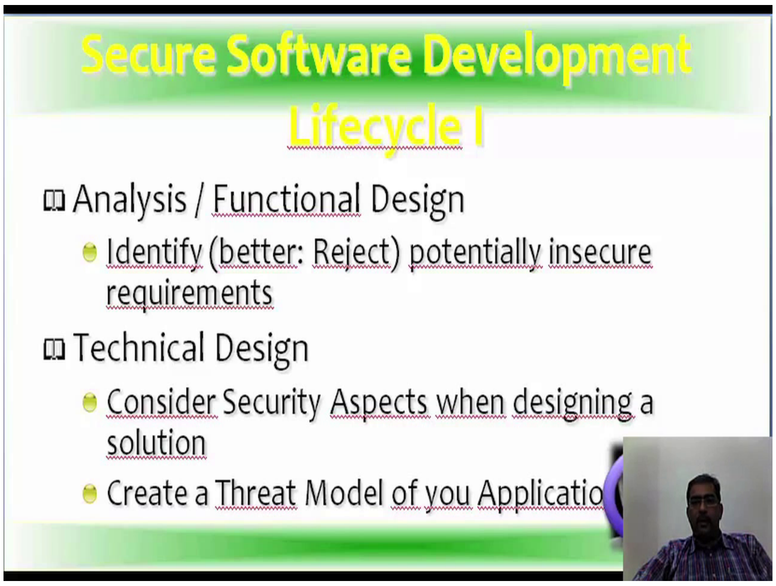
- Avoid using redirects and forwards as much as you can
- Don't involve user parameters in defining the target URL
- If you 'must' involve user parameters, then either
  - Use server side mapping to translate choice provided to user with actual target page
  - Validate each parameter to ensure its valid and authorized for the current user
- Use a secure Redirect API
  - e.g. ESAPI SecurityWrapperResponse.sendRedirect(URL)
- Some thoughts about protecting Forwards
  - Call the access controller to make sure the user is authorized before you perform the forward
  - Next best is to make sure that users who can access the original page are access the target page

The protection for it is avoid using a redirects and forwards as much as you can do not involve user parameters in defining the target URL. If you must involve user parameter then either use server site mapping to translate choice provided to user with actual target page or validate each parameter to ensure it is valid and authorized for the current user. Use a secure redirect API for example, ESAPI security wrapper response send redirect.

Now, what is the wrapper I believe we discussed what a wrapper is in domain 5, but wrapper is the program used in TCP or Transmission Control Protocol to provide layer of security by intercepting calls to computer services and determine whether the services or authorized to execute, the wrapper provides a necessary protection against host name and host address spoofing, some thoughts of the protecting forwards; call the access controller to make sure that user is authorized before you perform the forward.

The next best is to make sure the users who can access the original page are authorized to access the target site, we will take a look at secure SDLC also Secure Software Develop Life Cycle.

(Refer Slide Time: 18:09)

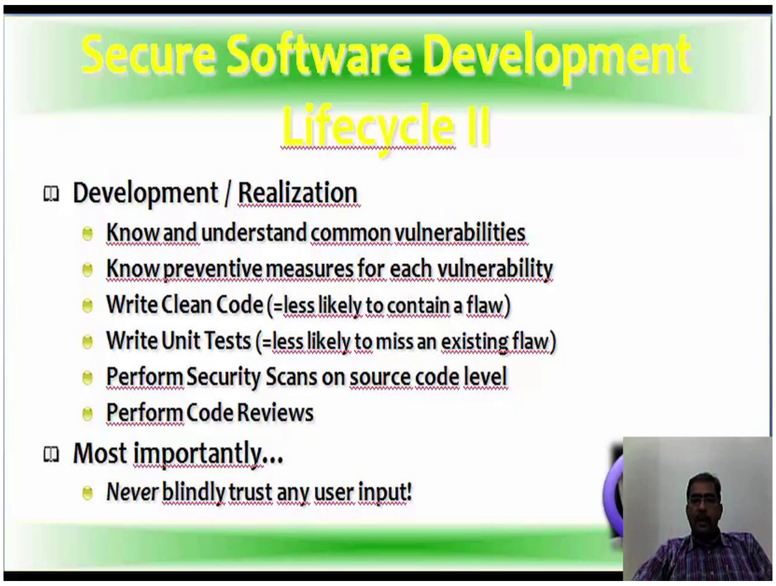


Secure Software Development Lifecycle I

- Analysis / Functional Design
  - Identify (better: Reject) potentially insecure requirements
- Technical Design
  - Consider Security Aspects when designing a solution
  - Create a Threat Model of you Applicatio

Now, there is analysis and functional design that is identify or better reject potentially insecure requirement. So, you do an analysis and in the functional design, you reject potentially insecure requirements, in the technical design consider security aspects when designing a solution, create a threat model for your application. So, you have to involve your internal security team during the technical design or an external team, so, that the security aspects are adequately addressed during the design face itself, it is very difficult to retrofit is security once a product is completed. So, it is always better to analyze the security requirements and design the security requirements in the technical design phase itself.

(Refer Slide Time: 19:13)

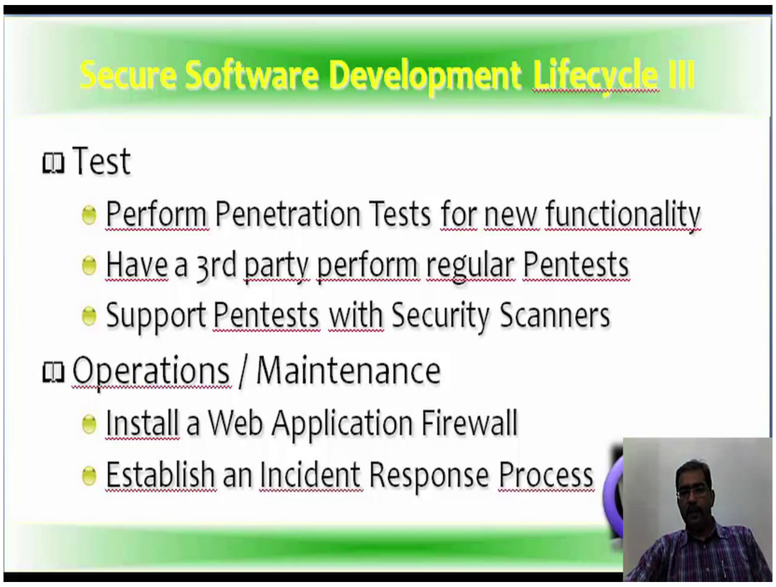


Secure Software Development Lifecycle II

- Development / Realization
  - Know and understand common vulnerabilities
  - Know preventive measures for each vulnerability
  - Write Clean Code (=less likely to contain a flaw)
  - Write Unit Tests (=less likely to miss an existing flaw)
  - Perform Security Scans on source code level
  - Perform Code Reviews
- Most importantly...
  - *Never blindly trust any user input!*

Development realization know and understand common vulnerabilities, know preventive measures for each vulnerability, write clean code that is less likely to contain a flaw, write unit tests less likely to miss an existing flaw, perform security test scans on source code level, perform code reviews. Most importantly never blindly trust any user input, this we have been retreating throughout this module that never trust any user input.

(Refer Slide Time: 19:50)



Secure Software Development Lifecycle III

- Test
  - Perform Penetration Tests for new functionality
  - Have a 3rd party perform regular Pentests
  - Support Pentests with Security Scanners
- Operations / Maintenance
  - Install a Web Application Firewall
  - Establish an Incident Response Process

In the test phase perform penetration test for new functionality have a third party perform regular pentests, support pentests with security scanners, then in operation and maintenance install a web application firewall. Now, establish the incident response process, now what is the web application firewall we have discussed it earlier, but there is the difference between your web application firewall and your network firewall, in your web application firewall any request that is coming in is analyzed or probed for the existence of malicious script or code, if it is found to be malicious that packet is dropped, if it is found to be then packet is passed through.

So, web application firewall becomes the important component in an internet environment today, especially for company is doing online business or financial companies where the bank's data is very critical, insurance company. So, almost any company having to do anything with internet should have a web application firewall installed, it not only protects your server or protects your application from being hacked, but also gives you a update or it gives you on what actually has gone wrong or what kind of request are coming in from a malicious person outside the network, which will enable the development into fine tune the application.