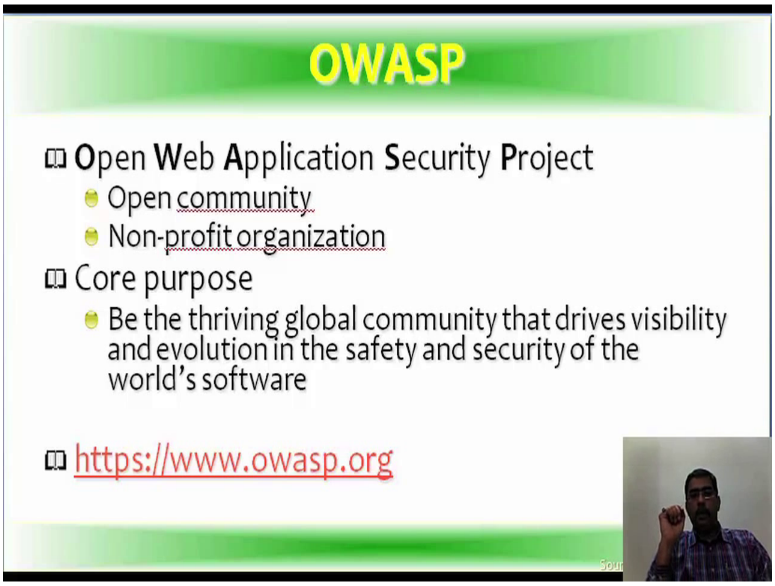
Introduction to Information Security
Prof. Dilip H. Ayyar
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 61

Now, we will take a look at Open Web Application Security Project.

(Refer Slide Time: 00:17)

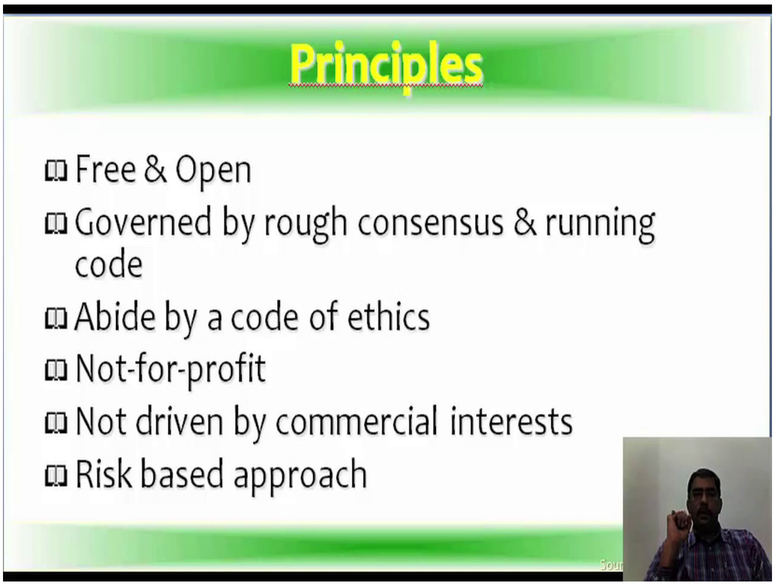


OWASP

- Open Web Application Security Project
  - Open community
  - Non-profit organization
- Core purpose
  - Be the thriving global community that drives visibility and evolution in the safety and security of the world's software
- https://www.owasp.org

OWASP is a open community, it is a non for profit organization, the primary purpose or core purpose is to be the thriving global community that drives visibility and evolution in the safety and security of the world's software. So, if you need to know the back ground of OWASP and other details you can go to www.OWASP.org.

(Refer Slide Time: 00:47)

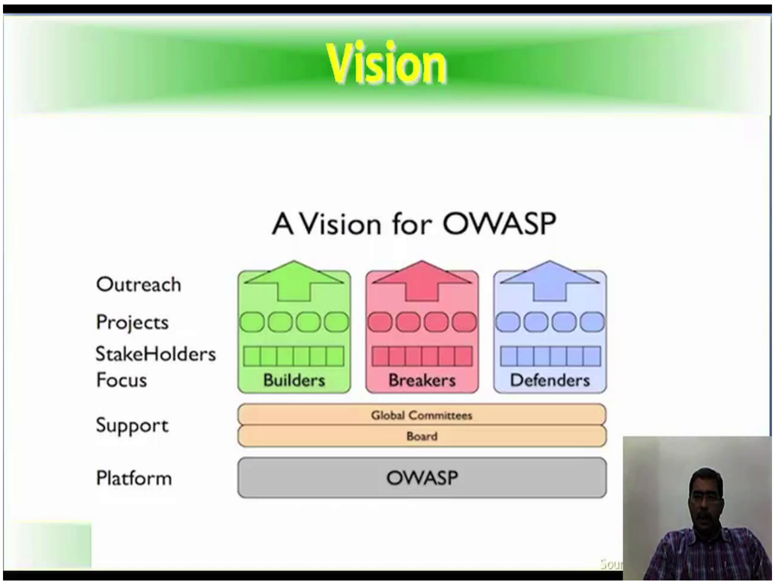


Principles

- Free & Open
- Governed by rough consensus & running code
- Abide by a code of ethics
- Not-for-profit
- Not driven by commercial interests
- Risk based approach

What are the principles of OWASP? One it is free and open, it is governed by rough consensus and running code, abide by a code of ethics it is not for profit, it is not driven by commercial interest and it follows of risk based approach.

(Refer Slide Time: 01:06)



Vision

A Vision for OWASP

Outreach

Projects

StakeHolders Focus

Support

Platform

Builders

Breakers

Defenders

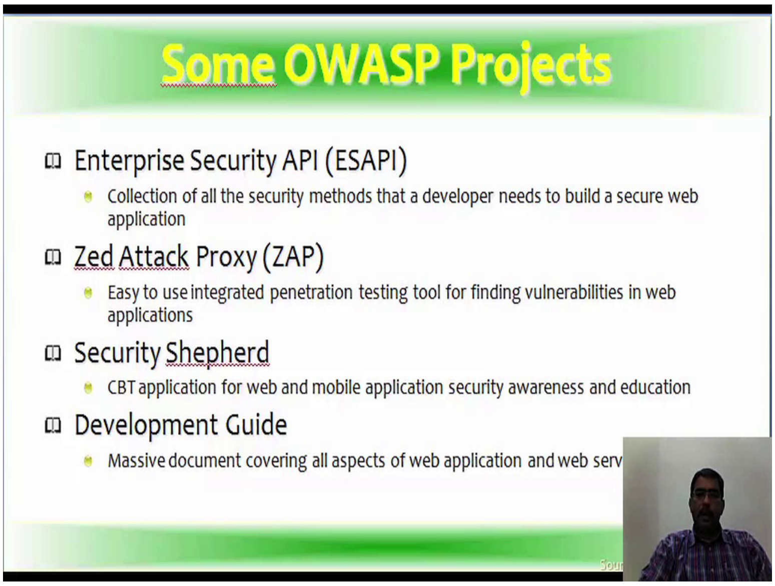
Global Committees

Board

OWASP

The vision for OWASP is for the builders, the breakers or hackers and the defenders. Now, what all areas covered; the platform is OWASP meaning that is the foundation then there is a global committee and board for OWASP, the stake holder focus can be either the builder, the breaker or the defender; projects again the same; outreach the same. Now, when we save breakers you can save in the positively were it is someone who tries to test the system for security class one and need not necessarily be occurs, but it could be information security professional within the organization.

(Refer Slide Time: 02:00)



Some OWASP Projects

- Enterprise Security API (ESAPI)
  - Collection of all the security methods that a developer needs to build a secure web application
- Zed Attack Proxy (ZAP)
  - Easy to use integrated penetration testing tool for finding vulnerabilities in web applications
- Security Shepherd
  - CBT application for web and mobile application security awareness and education
- Development Guide
  - Massive document covering all aspects of web application and web serv

OWASP has a lot of resources and some of the projects that OWASP has done is enterprise security

API, ESAPI which is the collection of all security methods that a developer needs to build a secure web application. Zed attack proxy which is zap, which is easy to use integrated penetration testing tool for finding vulnerabilities in web application. Security shepherd, which is the CBT application for web and mobile security awareness and education.

Then, there is massive document covering all aspects of web application and web service security on the development guide. We willlook at the OWASP the top 10 they have been different versions released over a period of years it is basically a guide line to conduct web application security audits or to assess the web application security risk.

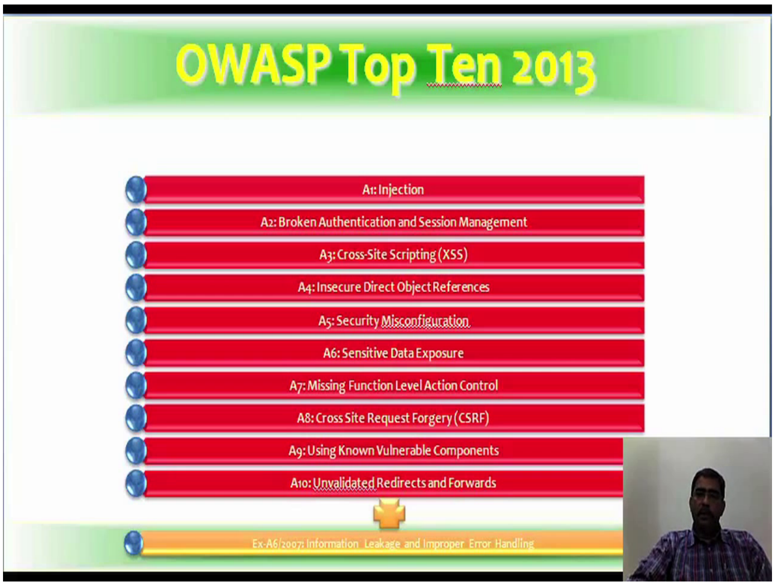
OWASP by following OWSAP you have a guided or organized approach to testing information security in web applications. But, it does not mean that is the end of testing of web applications, because you there are like I said in the beginning, there are several other areas which you will have tested manually it may fall in to the one of the categories of OWASP, but then step by step details are not given for example, existence of captcha is not mentioned in your but, it should come under it to which is broken access and an authentication.

So, similarly audit trails should come in one of the categories. So, you have to do a lot of manual testing as well in order to ensure that the application that you are auditing or you are accessing it is secure. Theremay be minute things which you may feel that it is not of consequences, but then it does carry a lot of risk. So, OWASP will give you a basic guideline on how to conduct web application audits.

What are the categorization of vulnerabilities or threats? And how your report structure can be the simple ways I have a 1 to a 10? So, classify all the risk in each of the categories of watched. So, I get a very nice frame work to work on, so was we will give you that guideline, but you need to work on different architectures to understand the intricacies of OWASP Top 10. For example, when you do a security assessment in a PHP environment it may be totally different than what you would do in a ASP IaaS based environment.

So, you each audit like I said is a new experience. So, as you go step by step into auditing applications or assessing the security of the application you will come to know that OWASP is indeed or very useful guideline to follow.

(Refer Slide Time: 05:28)



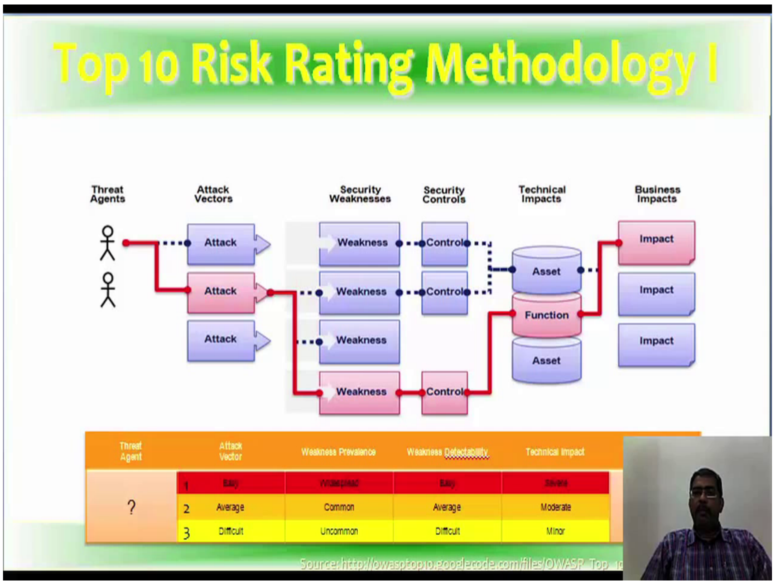
OWASP Top Ten 2013

- A1: Injection
- A2: Broken Authentication and Session Management
- A3: Cross-Site Scripting (XSS)
- A4: Insecure Direct Object References
- A5: Security Misconfiguration
- A6: Sensitive Data Exposure
- A7: Missing Function Level Action Control
- A8: Cross Site Request Forgery (CSRF)
- A9: Using Known Vulnerable Components
- A10: Unvalidated Redirects and Forwards
- Ex-A6/2007: Information Leakage and Improper Error Handling

The top 10 2013 has got, again top 10 OWASP web as always being top 10. Now, under 2013 version A 1 is injection, A 2 is broken authentication and session management, A 3 XSS, A 4 is in secure direct object reference, A 5 security misconfiguration, A 6 is sensitive data exposure, A 7 is missing function level access control, A 8 is cross site request forgery, A 9 is using known vulnerable components and A 10 is unvalidated redirects and forwards.

Now, we are added one more thing of OWASP which was Ex-A6 which isis information leakage and improper error handling to this. So, that you will understand how the classification has been done.

(Refer Slide Time: 06:29)



Top 10 Risk Rating Methodology I

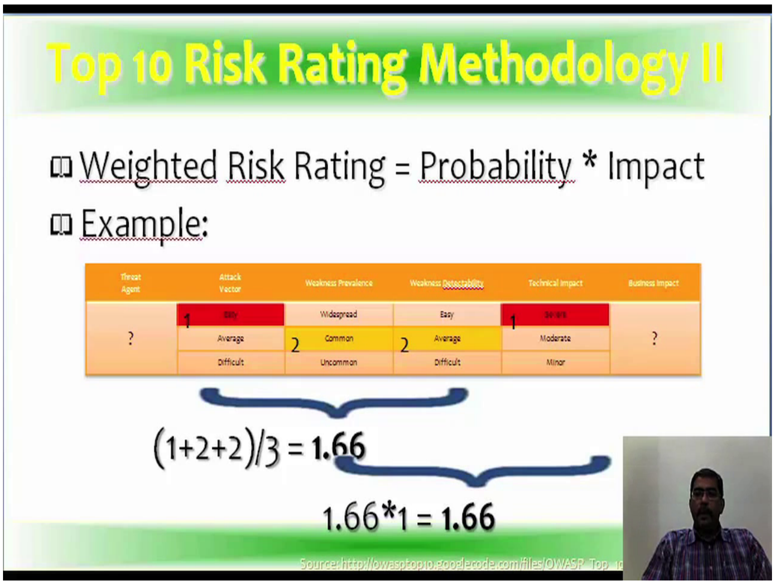
Threat Agent		Attack Vector	Weakness Prevalence	Weakness Detectability	Technical Impact
?	1	Easy	Widespread	Easy	Severe
	2	Average	Common	Average	Moderate
	3	Difficult	Uncommon	Difficult	Minor

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

The top 10 risk rating methodology is this we just taken from OWASP again, there are threat agents and then there are attack vectors, which exploits the security weaknesses, which in turn gives you what control you have to put and what are the technical impacts of that what function has to be change, ultimately what is the business impact. Now, if you see the table below question mark we have put. So, if there is a threat agent or whatever threat agent, the attack vector is classified as easy, average and difficult.

That means, particular exploit is easy to attack it is of average difficulty and it is very difficult to attack. The weakness prevalence is widespread that means it is affects a large number of systems and again weakness prevalence is common means it is a common occurrence and the third one is uncommon means it has not happen very frequently it is rarely it happens and detectability of the weakness is again average easy, average and difficult; there are some vulnerability that you can easily detect some with an average time frame and third it is very difficult to defect, but it still exist technical impact is severe moderate or minor. So, that is self explanatory and the business impact based on the vulnerability how we it affects the function, how we affect the business that will be put.

(Refer Slide Time: 08:18)



Top 10 Risk Rating Methodology II

- Weighted Risk Rating = Probability * Impact
- Example:

Threat Agent	Attack Vector	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact
?	1 Easy	Widespread	Easy	1 Severe	?
	Average	2 Common	2 Average	Moderate	
	Difficult	Uncommon	Difficult	Minor	

$(1+2+2)/3 = \mathbf{1.66}$

$1.66*1 = \mathbf{1.66}$

Source: http://owasptop10.googlecode.com/files/OWASP Top 1

If you see this the

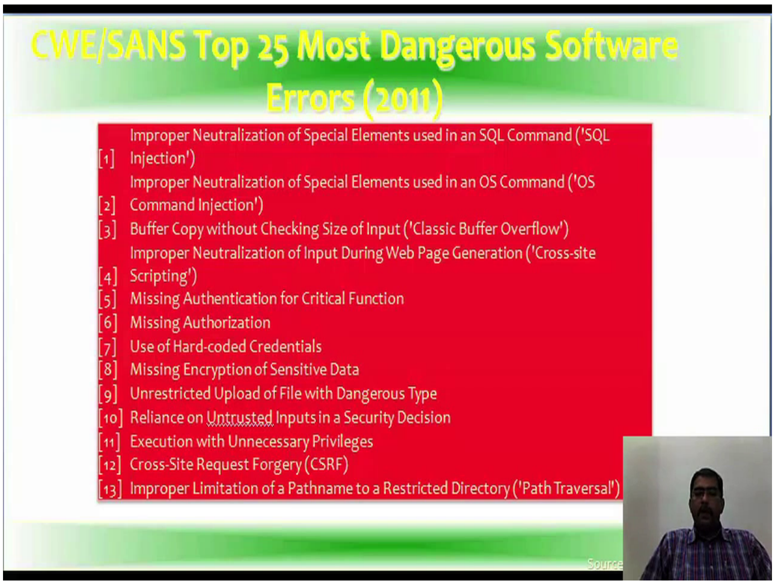
Weighted risk rating = Probability * impact.

Now, if you see an example there is a particular threat agent which has got an attack vector of one the weakness problems of 2, the weakness detectability as 2 and the technical impact as 1. So, 1 plus 2 plus 2 divided by 3, so you are taking average 1.66. So, if you see the attack vector the weakness prevalence and detectability, it is 1.66.

Now, weakness prevalence, the weakness detectability and technical impact is also 1.66, because 1.66 in to 1 is 1.66 again how the 1.66 need 1.66 in to 1 because the technical impact is 1, this is just to show you how the risk rating methodology is. So, for different threat agencies, the risk weightage will be different, the calculation even though is a similar the output will be different say let us say that there is a weighted risk rating of 7.5 then you have to classify as sever, if it is 9 or 10 then it is catastrophic.

So, again depending on the kind of risks that are coming out, the kind of vulnerability there are dropping out and depending upon the attack vector,weakness prevalence weakness detectability, the averages from and then ultimately you can see what ever impact for the organization as a whole. Are there any alternatives to OWASP top 10 there are like your SANS top 25 let us take a...

(Refer Slide Time: 10:10)



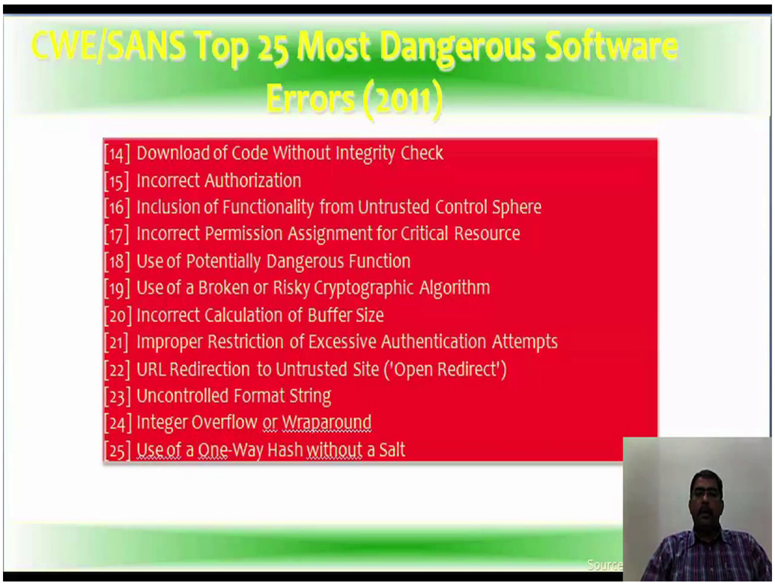
CWE/SANS Top 25 Most Dangerous Software Errors (2011)

[1] Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
[2] Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
[3] Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')
[4] Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
[5] Missing Authentication for Critical Function
[6] Missing Authorization
[7] Use of Hard-coded Credentials
[8] Missing Encryption of Sensitive Data
[9] Unrestricted Upload of File with Dangerous Type
[10] Reliance on Untrusted Inputs in a Security Decision
[11] Execution with Unnecessary Privileges
[12] Cross-Site Request Forgery (CSRF)
[13] Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')

Source

So, here in SANS, CWE SANS top 25 most dangerous software errors, this is 2011 version, there are 25 categories here; you can match the 10 categories of OWASP to certain categories of SANS top 25 also. Now, you just see many of these are development related also in the CSRF is that as point 12 and in a OWASP top 10 also it is there, buffer over flow it is categorized under some other domain in OWASP here it is there in three, then improper limitation of path name path traversal or directory traversal which is there in both.

(Refer Slide Time: 11:07)



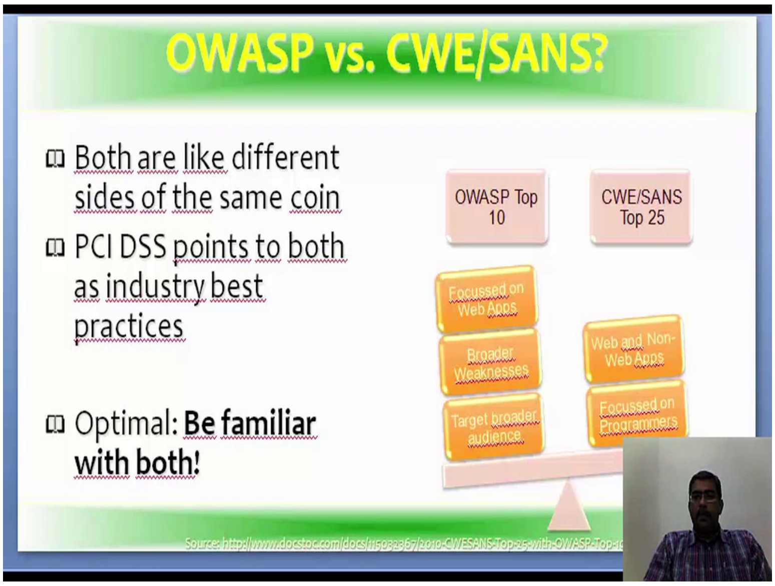
CWE/SANS Top 25 Most Dangerous Software Errors (2011)

[14] Download of Code Without Integrity Check
[15] Incorrect Authorization
[16] Inclusion of Functionality from Untrusted Control Sphere
[17] Incorrect Permission Assignment for Critical Resource
[18] Use of Potentially Dangerous Function
[19] Use of a Broken or Risky Cryptographic Algorithm
[20] Incorrect Calculation of Buffer Size
[21] Improper Restriction of Excessive Authentication Attempts
[22] URL Redirection to Untrusted Site ('Open Redirect')
[23] Uncontrolled Format String
[24] Integer Overflow or Wraparound
[25] Use of a One-Way Hash without a Salt

Source

This is the remaining, download of code without integrity check, incorrect authorization, inclusion of functionality from untrusted control sphere, incorrect permission assignment, use of potentially dangerous function, use of a broken or risky cryptographic algorithm which is there in OWASP under different name, incorrect calculation of buffer size, improper restriction of excessive authentication attempts, URL redirection open redirection also classified under one of the domains of OWASP, uncontrolled format string, integer overflow, use of one way hash without a salt.

(Refer Slide Time: 11:56)



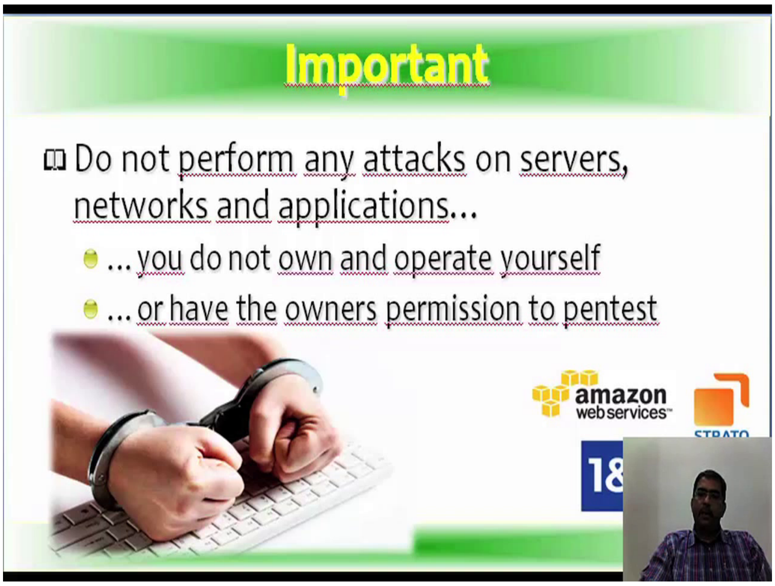
OWASP vs. CWE/SANS?

- Both are like different sides of the same coin
- PCI DSS points to both as industry best practices
- Optimal: **Be familiar with both!**

Source: http://www.docstoc.com/docs/115032367/2010-CWESANS-Top-25-with-OWASP-Top-1

So, OWASP was a CWE/SANS when a look at it both are like different sites of the same coin PCI DSS points to both as industry best practices. So, if you need a PCI DSS certifications you can either follow OWASP top 10 or CWE/SANS top 25. Now, the optimal knowledge level is you be similar with both, because except for a few variations the standard the run in parallel or the guidance run in parallel.

(Refer Slide Time: 12:30)

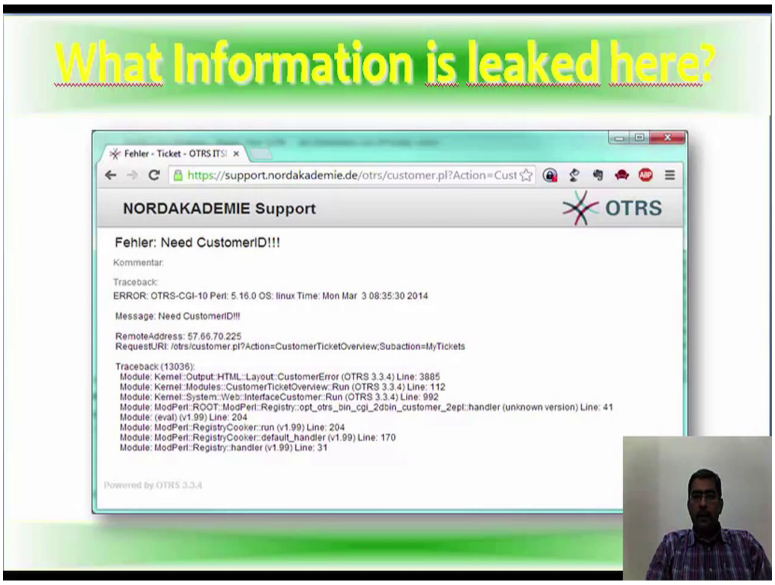


Important

- Do not perform any attacks on servers, networks and applications…
  - … you do not own and operate yourself
  - … or have the owners permission to pentest

We have to keep read-writing a particular thing is do not perform any attacks on servers, we have all ready discussed in the beginning of this module. So, do not perform any attacks on servers, networks and applications, you do not own and operate yourself; that means, do not touch anybody else's systems, servers, network, if you have a specific permission from the owner of that infrastructure you can do it analysis it is a no no. Now, we will continue with the top 10 before starting the top 10 we will just look at Ex-A6 of 2007 version which is improper leakage and improper error handling.

(Refer Slide Time: 13:20)



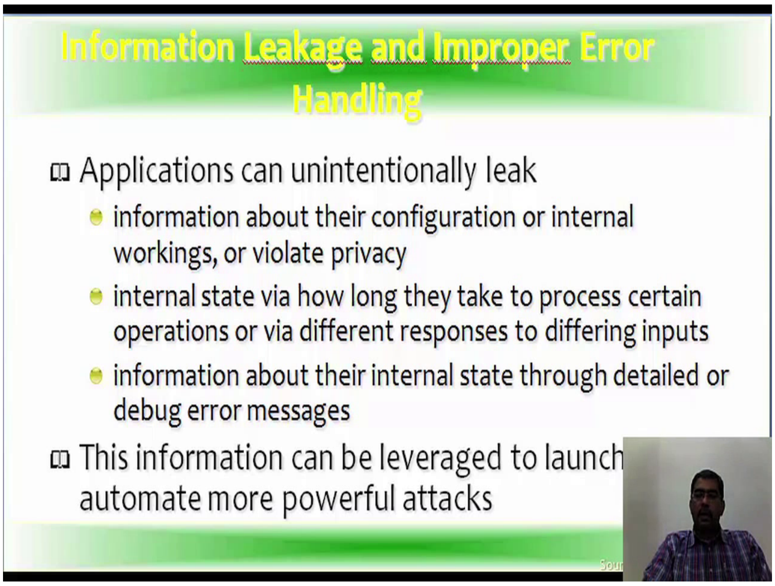
What Information is leaked here?

Fehler - Ticket - OTRS ITS
https://support.nordakademie.de/otrs/customer.pl?Action=Cust
NORDAKADEMIE Support
OTRS

```
Fehler: Need CustomerID!!!
Kommentar:
Traceback:
ERROR: OTRS-CGI-10 Perl: 5.16.0 OS: linux Time: Mon Mar  3 08:35:30 2014

 Message: Need CustomerID!!!

 RemoteAddress: 57.66.70.225
 RequestURI: /otrs/customer.pl?Action=CustomerTicketOverview;Subaction=MyTickets

 Traceback (13036):
   Module: Kernel::Output::HTML::Layout::CustomerError (OTRS 3.3.4) Line: 3885
   Module: Kernel::Modules::CustomerTicketOverview::Run (OTRS 3.3.4) Line: 112
   Module: Kernel::System::Web::InterfaceCustomer::Run (OTRS 3.3.4) Line: 992
   Module: ModPerl::ROOT::ModPerl::Registry::opt_otrs_bin_cgi_2dbin_customer_2epl::handler (unknown version) Line: 41
   Module: (eval) (v1.99) Line: 204
   Module: ModPerl::RegistryCooker::run (v1.99) Line: 204
   Module: ModPerl::RegistryCooker::default_handler (v1.99) Line: 170
   Module: ModPerl::Registry::handler (v1.99) Line: 31

Powered by OTRS 3.3.4
```

If you see here the prevalence is the widespread and the impact is minor, take a look at this particular site and can you find out what information is leaked here, probably we will hold like this for a couple of vacant. So, that you can see what kind of information is leaked here.

(Refer Slide Time: 13:46)



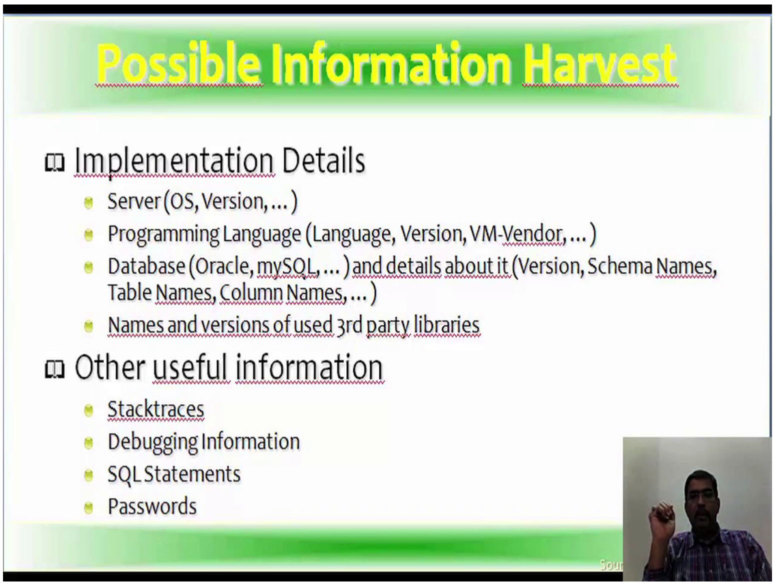
Information Leakage and Improper Error Handling

- Applications can unintentionally leak
  - information about their configuration or internal workings, or violate privacy
  - internal state via how long they take to process certain operations or via different responses to differing inputs
  - information about their internal state through detailed or debug error messages
- This information can be leveraged to launch automate more powerful attacks

So, the fact is applications can unintentionally leak, information about their configuration or internal workings or violation privacy. Internal state via how long they take to process certain operations or via different responses to different or different inputs, information about their internal state through detailed or debug error messages, this information can be leveraged to launch or even automate more powerful attacks .

Now, if you go back to that slide you know what is leaked here errors OTRS, CGI, top 10, OX, Linux, time is given, message, remote, address is also available 57, 66, 70 something, then we trace back is quite a bit of information which is coming here. Now, using this information and attacker can potentially get a lot or we can customize the attack to see this particular environment from information leakage. So, this is what was there in A6 of 2007 and this could give a lot of information to the attack of to proceed further on how to attack the system.

(Refer Slide Time: 15:13)



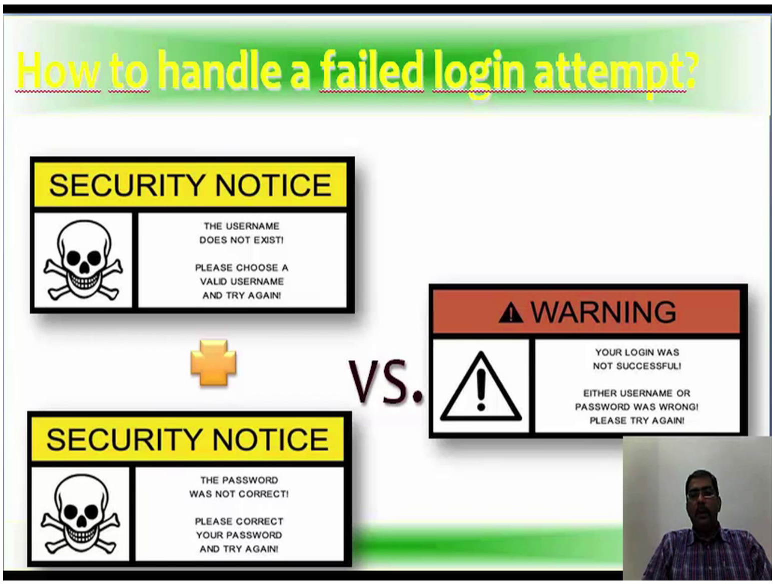
Possible Information Harvest

- Implementation Details
  - Server (OS, Version, …)
  - Programming Language (Language, Version, VM-Vendor, …)
  - Database (Oracle, mySQL, …) and details about it (Version, Schema Names, Table Names, Column Names, …)
  - Names and versions of used 3rd party libraries
- Other useful information
  - Stacktraces
  - Debugging Information
  - SQL Statements
  - Passwords

So, there is a possibility of a information harvesting that is implementation details we saw that server OS and version was there, programming language, language version, VM vendor, I mean there are possibility that this could be leaked. Database Oracle, mySQL and detail about the version, the schema names, tables names, column names, names and versions of used third party libraries if any had use.

Other useful information that we will get a stack traces, debugging information, SQL statements and some cases even the passwords again depends on how the application it is developed and how it is configure.

(Refer Slide Time: 15:57)



How to handle a failed login attempt?

SECURITY NOTICE

THE USERNAME DOES NOT EXIST!

PLEASE CHOOSE A VALID USERNAME AND TRY AGAIN!

SECURITY NOTICE

THE PASSWORD WAS NOT CORRECT!

PLEASE CORRECT YOUR PASSWORD AND TRY AGAIN!

VS.

⚠ WARNING

YOUR LOGIN WAS NOT SUCCESSFUL!

EITHER USERNAME OR PASSWORD WAS WRONG! PLEASE TRY AGAIN!

Many of you will have a internet banking sites or you will be subscribing to certain sites for information for account transfer or so many other things. So, how to you login a failed attempt.

Now, if you see the first on the left it says security notice, the user name does not exist, please choose a valid username and try again that is one kind of pop up. The second one is the password was not correct, please correct your password and try again versus third one is warning your login was not successful either username or password was wrong please try again.

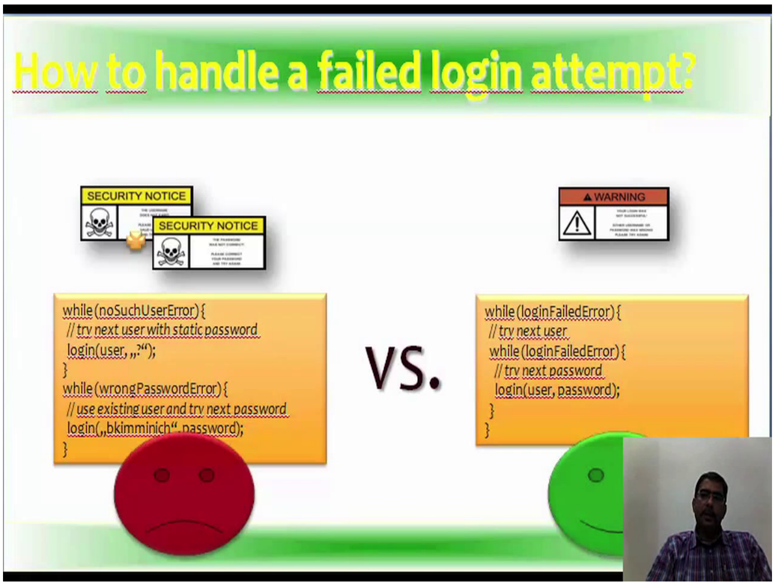
So, when you looked at this is very evident that one of the right hand side, the only one this is right one can a system tell you or tell a potential attacker that your user name does not exists. So, if we know sure that particular user name does not exist we will go on trying some other user names, if you password was not corrected or not correct again if the system tells you the password is not correct; obviously, it is going to tell you or other we to login when the password is correct. So, you eliminate a certain level of task or hard work when you get these kind of messages.

Now, the correct one is your login was not successful, either user name or password is wrong please try again, most of the sites use the one on the right, but I have seen instances, where the first one and the second one the one's in the yellow also where use, which is actually if you see it is stupid because that is not the kind of message that should be put on a website or other the application it gives the attacker a whole lot of information he knows also.

Suppose I try admin it says username does not exist, so I know admin is not there similarly administrator or root. So, there are several ways to gather information it is the way it is done is more important, possibility is suppose you use at tool like THC high drop for web application brute force academy, suppose the login and password screen is not validated for 3 unsuccessful attempts or 5 unsuccessful attempts the tool will keep trying different combinations of user name and password. So, a brute force or a dictionary attack and be done.

So, when such a situation happens based on the error messages or based on the information that the server is going to give you, the attacker can fine tune the tool to say this is what I need to check; so that becomes very dangerous for the application.

(Refer Slide Time: 19:06)



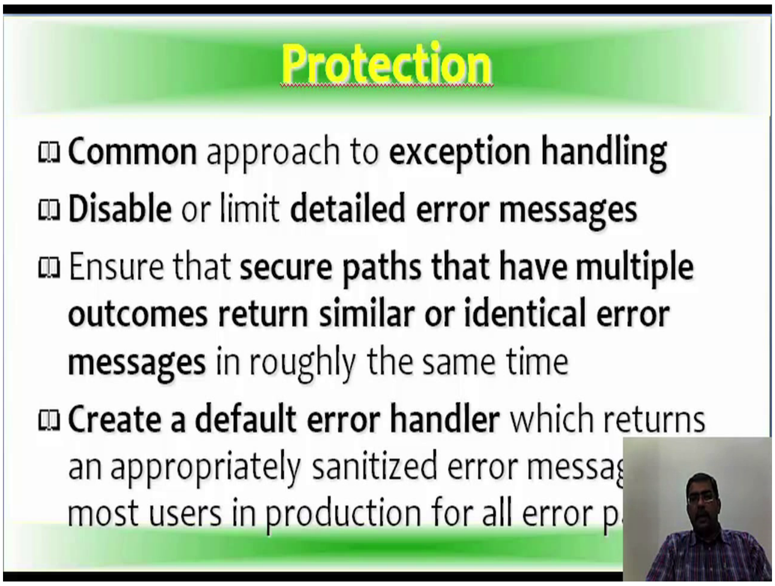
How to handle a failed login attempt?

```
while (noSuchUserError) {
 // try next user with static password
 login(user, „?“);
}
while (wrongPasswordError) {
 // use existing user and try next password
 login(„bkimminich“, password);
}
```

```
while (loginFailedError) {
 // try next user
 while (loginFailedError) {
  // try next password
  login(user, password);
 }
}
```

In fact, you are speeding up the process of attack rather than delaying it. As say sample see how to handle a failed login attempt. Now, the syntax may not be right it is just for a explanation that we are given yet. The two yellow once is a wrong one versus the warning message what you hear is the right one. So, if you read through the statement you with no the differences in the syntax, not in the syntax in the way that he application has been written and how the application will handle these instances.

(Refer Slide Time: 19:44)



Protection

- **Common** approach to **exception handling**
- **Disable** or limit **detailed error messages**
- Ensure that **secure paths that have multiple outcomes return similar or identical error messages** in roughly the same time
- **Create a default error handler** which returns an appropriately sanitized error messag most users in production for all error p

So, what is the protection for this; the common approach to exception handling, disable or limit detailed error messages only give what is required, in some places are seeing 403 forbidden. So, forbidden when you get a 403 error you know that there is a directory. Now, instead of 403 imagine a situation that I give a 404 error instead of 403 error, so it become is difficult to find out whether that directory really exist. So, I have the actually do a invasive scanning to find out whether the directory really exist.

Otherwise, the error messages itself will tell you that yes there is a directory like this and access not for method to that directory. Ensure that secure paths that have multiple outcomes return similar or identical error messages in roughly the same time and create a default error handler which returns an appropriately sanitized error message for most users in production for all error paths.

Now, instead of displaying the OS version, the data base version and other confidential information there should be a default error handler, where the errors can be customized and say for this kind of the error, this is the error page that we will come for this kind of error this should be displayed. So, you should have error handler which will return the appropriately sanitized, sanitized means after filtering of the critical information or if there is a chance of information disclosure that information should be remove and the appropriately message should be put, it could just be that no page not found or it could be there function not found. So, it could be any message like that, but details of the data base, the OS, the version, the frame work, the library.