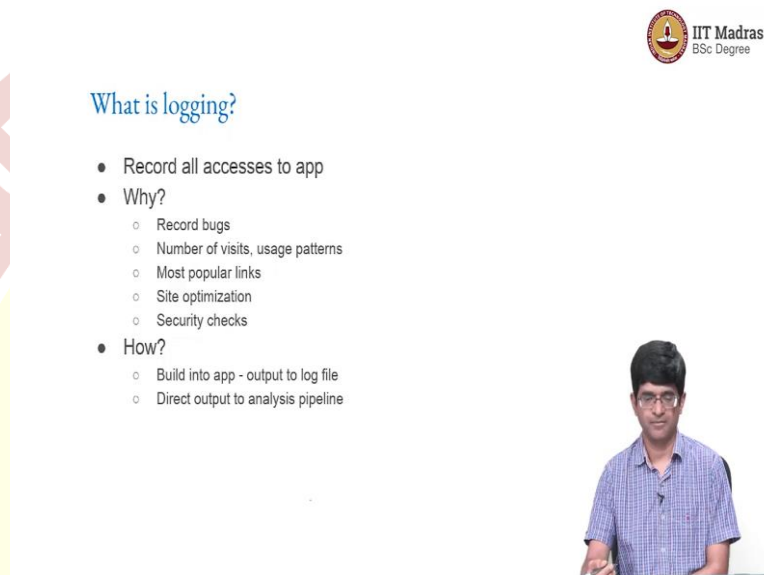


IIT Madras
ONLINE DEGREE

Modern Application Development – I
Professor Nitin Chandrachoodan
Department of Electrical Engineering
Indian Institute of Technology Madras
Logging

Hello, everyone, and welcome to this course on modern application development.

(Refer Slide Time: 00:16)



The slide is titled "What is logging?" in blue text. It contains a bulleted list with the following items:

- Record all accesses to app
- Why?
 - Record bugs
 - Number of visits, usage patterns
 - Most popular links
 - Site optimization
 - Security checks
- How?
 - Build into app - output to log file
 - Direct output to analysis pipeline

In the bottom right corner of the slide, there is a small video inset showing Professor Nitin Chandrachoodan, a man with glasses wearing a blue checkered shirt, speaking.

The slide is part of a presentation from IIT Madras BSc Degree, as indicated by the logo in the top right corner.

Now in the last part on this section on security, we are going to look at the topic of logging. Now what exactly is logging? We need to record all accesses to a particular application. Why do we need to do this? For multiple reasons; one is that we want to find out if there are any bugs or if there are any problems or server related errors. But we might also want just statistics, how many visits were there to a given site? What were the usage patterns on which are the most popular links on which days was the number of accesses the most?

And you can also do certain kinds of site optimization based on this. Now, how do you go about doing this logging? That can also be done at different levels. As an application builder, you should try and build it into the app itself and log the information that is useful to you. And typically, what is done over here is that you would then log it directly into a log file, which you can then process later.

Or there might be ways by which you can directly output to an analysis pipeline. What I mean by an analysis pipeline is some kind of software that specializes in being able to read and understand what different log patterns are indicating.

(Refer Slide Time: 1:37)



Server logging

- Built in to Apache, Nginx, ...
- Just accesses and URL accessed
- Can indicate possible security attacks:
 - Large number of requests in short duration
 - Requests with "malformed" URLs
 - Repeated requests to unusual endpoints



The logging can be done at the server level. It is usually built into servers, web servers, such as Apache, Nginx, and so on. But over here, it can only log information about which URL was being accessed. So, what exactly was the user trying to do?

What is there inside the post information which requests were being made, that information is not even available to the server, it does not sort of store any of that in a log file. It is useful in order to sort of indicate certain kinds of security issues, especially security attacks. If you find for example, that there are large number of requests in a short duration.

Or if there are malformed URLs, things which are obviously not correct requests that are coming to your server, there might be attempts to try and break your server in some way. It may be an SQL injection, or some other kind of thing, which is trying to break your server.

(Refer Slide Time: 02:30)



Application level logging

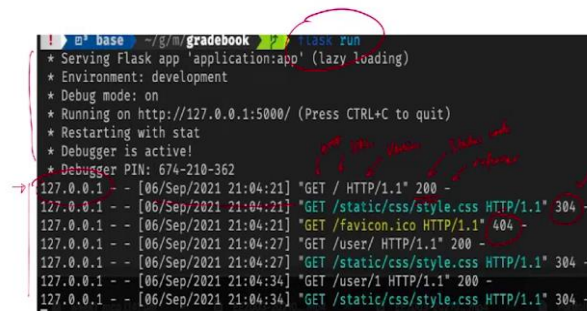
- Python logging framework
 - Output to file, other "stream" handlers
- Details of application access
 - Which controllers
 - What data models
 - Possible security issues
- All server errors



On the other hand, you as an application developer, can do your own logging. And the Python logging framework is something that can be used very efficiently for this. It can provide multiple different outputs, it can either write into a file, or it can create so called stream outputs, which can be used in order to maybe even send information out over a network connection to a remote server, if needed.

And in your, when you implement your own logging, you can have a lot more detailed information. You can, for example, say which controllers are being invoked, which data models? Are there any requests that are sort of potential security issues? And all server errors, anything that causes the server to crash, even the stack traces could be logged. And ideally, should be logged so that you have a better way of sort of handling such problems.

(Refer Slide Time: 03:20)



So, this is an example of what a log file might look like. I mean, this is just the log that comes up, for example, when I run using the flask run command on the gradebook example. And as soon as flask comes up, it puts out some information about the server itself. But the main log information is coming out here. It basically, in this case, of course, it is running on my local machine, and I am accessing it locally.

Therefore, this IP address is always going to be local, it gives the date and time when the access happened, it also gives the HTTP verb, the URL, the version and the status code. This last one is usually something called the referrer, which, if you have an internal link, which goes from one to another, then you would find that something is there. Otherwise, you will just find a dash. Now, the important things to look for would be as long as everything is 200, great.

Three, or fours are alright, in the sense that all they indicate is that either something has not changed since the last time it was accessed, or has been redirected, but it is not necessarily a problem. 404 on the other hand, is not good news. It means that, somebody was trying to access something which was not there. And the more such requests that you have, it is actually going to be a bit of a problem. You do not really want this to be something common.

(Refer Slide Time: 04:47)



Log rotation

- High volume logs - mostly written, less analysis
- Cannot store indefinitely
 - Delete old entries
- Rotation:
 - Keep last N files
 - Delete oldest file
 - Rename log.i to log.i+1
 - Fixed space used on server



And, as far as logs are concerned, one of the things that you need to look out for is you need to make sure that you do not have high volume logs, meaning that it does not fill up your disk. Logs can be generated in high volumes; you have a large amount of data. They are mostly being written somewhere. And in a lot of cases, you might not be doing too much analysis, you might only be looking for security.

What that means is you cannot store it indefinitely, you need to delete the older entries. And one of the things you need to do is rotation. One way of handling this would be to do log rotation. That is to say, you keep the last ten files, delete the oldest file, rename all the intermediate files and take the most recent one and rename it as a dot 1 file or something of that sort. So, that you can now create a new file. It basically uses a fixed amount of space on the server, and you do not run into space problems eventually.

(Refer Slide Time: 05:46)



Logs on custom app engines

- Google app engine
 - Custom logs
 - Custom reports
- Automatic security analysis



In case you are using a custom application engine, something like Google App Engine. they have their own logging mechanisms, and can generate custom logs and customized reports. And in a lot of cases can also do automated security analysis. And in case you do plan to run on some kind of a framework like that, you should definitely make use of those facilities.

(Refer Slide Time: 06:10)



Time series analysis

- Logs are usually associated with timestamps
- Time series analysis:
 - How many events per unit time
 - Time of specific incident(s)
 - Detect patterns (periodic spikes, sudden increase in load)
- Time-series databases
 - RRDTool, InfluxDB, Prometheus, ...
 - Analysis and visualization engines



One thing you can notice from all of this logging information is that what you are actually looking at is time series analysis. Because one of the most important kinds of patterns that you look for is, is there any specific time of day or some other thing, which indicates either a problem or, you know that there are a large number of requests coming in at a certain time, maybe an exam was opened up?

And how is the server performing under those conditions? So, you have time series data, the every piece of information has a timestamp associated with it, which means that certain kinds of databases like our RRDTOol or InfluxDB, are probably the best way to store and analyze such information.

(Refer Slide Time: 06:57)



Summary

Security is key to successful applications!

- Requires good understanding of principles
 - Crypto
 - SQL, OS vulnerabilities, ...
- Good frameworks to be preferred
- Analyze, Identify, Fix



So, to summarize, having good security is the key to having a successful application, you cannot afford to have security problems. Because it will definitely impact how you are going to work. Implementing good security does require a good understanding of principles of various things such as cryptography and also something about SQL and operating system level vulnerabilities.

You do not need to be an expert but you at least need to be aware of the concepts over there. In general, one relatively safe option is to always go with good known frameworks. And ultimately, there is no replacement for, constantly analyzing your logs and data, identifying problems and then going about fixing them.