Identifying Threats Using Flexible Server Logs

Context Slide

* Working on an 8.01 PSet, need to Google a constant
* You take out your laptop
* <CLICK> and open your browser expecting to see
* <CLICK>
* But instead you get this message
* <CLICK>
* After thorough troubleshooting follows
* Not your internet connection, other websites load
* Client side
* Must be Google’s problem
* Server side
* <CLICK>

Background Slide

* Suppose several students are working on the problem set (on left)
* They also come across different problems where they need Google’s help (on right)
* Not all at same time
* <CLICK>
* Maybe this student needs a Khan Academy video
* <CLICK>
* Then later this one needs to study a chart of moments of inertia
* <CLICK>
* And here you are, looking for your constant
* All other computers remain idle
* <CLICK>
* Now suppose these computers don’t belong to confused 8.01 students
* They are constantly making illegitimate requests, not waiting for response
* <CLICK>
* Then you try to Google your constant
* <CLICK>
* But you can’t get through, and get this error message
* <CLICK>
* These malicious computers, are executing an organized attack on Google’s server
* <CLICK>

Title Slide

* How can Google, or any webserver, tell legitimate users from illegitimate?
* <CLICK>
* I developed a program for my own webserver to identify threats using intelligent, flexible logging
* <CLICK>

Contribution Slide

* When an 8.01 student forms a search on Google for their constant, a lot of information is exchanged
* <CLICK>
* Here is some information that will be transferred
* Now, a lot of this is useless for identifying the threat depicted earlier
* My program allows a web administrator to select what information is important
* <CLICK>
* Once the admin chooses the fields, my program sorts out all the unimportant info
* Then it fills in the selected information
* <CLICK>
* And outputs into a log file in real time
* According to the administrator’s specified format
* Now, it is easy to recognize the threat posed earlier
* Thousands of lines of the same IP address making the same request within a second

Conclusion

* Detecting a threat like this is the most important step
* Then the information can be streamed into a database or another program
* Countermeasures can then be applied
* Threats change
* Administrators can select different fields to log
* Nothing is hardcoded
* This program can be easily applied to guarantee that a confused 8.01 student will always their answers