The slow Loris is a type of Denial of Services (DoS) Attack, purpose of a DoS os that you try and defeat soe web server nad or some computer by giving it as much bandwidth as you can so that it breaks. They can only use a certain amount of bandwidth, if you give them more then they can handle or try to request too many webpages, their server will go down.

Distributed DoS is the next level where you have multiple computers all attempting to do the same thing, nad more modern Dos will use amplification and things like this to try and improve this even more. The main point is to produce as much bandwidth as you can all at the same time, if you get enough, you can take them down. Can have massive impacts on smaller companies that cannot manage a bigger traffic plus cost money.

Slow-Loris comes at this from a completely different way. It is a protocol attack, so a layer-7 application attack, which does not need a lot of bandwidth so you can do one of these attacks and then continue to browse the web or play computer games.

How the web works. Send of a “get” request to a website and I say “get me index.html”, web-server sends this back and that’s the end of the conversation. Now when read, new conversation, now need “header.jpeg”, another “get” request. In the code shows up as text, shows a little bit about the user e.g. using firefox. Always ends with two caret return line feed, so two new lines. Slow loris was invented by someone saying “what if I don’t send back those return line feeds (at the end) I just keep the website waiting for me”. Can I go so slowly that I can break the website? Yes, yes you can.

When someone disconnect from the server e.g. signal drop, this will mean the server will stop serving this person and be open to serve another. The problem with a slow loris is that, information is still being sent juts extremely slowly thus, occupies that space. Slow-loris is barely any code and all it does is say “get me index.html” and sends them a space or a 0 or random number, then it waits form 20-30 seconds, just when the website assumes it is gone it sends another single byte and keeps the connection as long as it can and then does it with 200 other connections. Sending out 200 byte packets every minute or so and is very difficult for a firewall to notice because they are valid HTTP requests that are just slow.

Doesn’t affect every web server, mainly affects Apache which makes up about 49.9%. When it was made, it is made to start up a new thread to server every concurrent connection. So when a HTTP request comes in, a new thread is made, this handles that request and then the thread goes away when it is done. Now when the connections start to stay open longer than anticipated, then our connection limit get reached. When the limit is reached it starts to grind to a halt. So a slow Loris begins to open up these connections and when it becomes filled up from someone else using the website. They will keep opening connections till they have all the connections available.

Code is not very long 67 lines. Has some code to start up a socket, using a TCP connection, a GET request text, and for all existing sockets, can we send a little bit more data every 15 seconds. If the socket dies, it recreates it and keeps it going to 200 concurrent sockets which is more than this examples Apache can handle. The webserver thinks that it has 200 people looking at the page, In fact it’s one person looking at it really slowly. When refreshed in the example, the cache is full and we are left waiting. No access to the server due to the sockets being full, might see a timeout on client side saying it can’t get into the server. The attacker’s bandwidth is fine and can continue to do things online as they are using barely any of it. Another example of these slow and low attacks is RUded attacks, hard to detect, doing normal HTTP but doing extremely slow, the assumption that they will nt do these weird things