This comes from the make SSRF great again blackhat talk

Parsing URLS is hard! -> there are exploitable errors in almost all URL parsers!! Since URL parsing is not yet standardized almost all popular program languages implement their own libraries for such tasks. Many if not all have holes which we can exploit as shown.

For this talk we will only be focusing on http and https for the scheme. (Often people like to use gopher, file, or other schemes which can surly be useful but there is no guarantee a webpage is using those protocols.)

--NOTE: red boxes = space--

Basic structure of URL:

Diagram, timeline

Description automatically generated

Example 1: This is looking at how pythons different URL parsing libraries handle a malformed URL

Diagram

Description automatically generatedEach library python has handles this specific URL in a different way. This shows how important it is to test different payloads for SSRF as so many factors go into it even if we can pinpoint OS, backend language etc.

A picture containing calendar

Description automatically generatedThese are the complete results here, again be sure to test each different type of injection on any potential SSRFs as there are other libs besides these, and yea

Text

Description automatically generatedThis checks the URL to ensure the port number is 80 and the host is google.com ….

Well what if we pass in this:



Diagram

Description automatically generatedso the parse\_url actually recognized the port as 80, but the readfile method actually fetched localhost:11211. So this php and perl code is vulnerable to port injection. These are the main libraries used in PHP to do this so if you encounter php on a page be sure to try this out.

Consider this:

Diagram

Description automatically generatedThis is the same type of issue except by submitting a malformed host name.

The parse\_url method recognizes 1 thing and readfile recognizes the other. Again make sure to try on anything running php! Even in http headers.

Timeline

Description automatically generated with low confidencecURL: a powerful tool with many language bindings

Most URL parsers recognize google.com as the URL here except for cURL which can be used to exploit it. cURL is often used to parse URLs especially in PHP which doesn’t have good libraries for parsing.

THIS HAS BEEN PATCHED IN 2017 BUT WE CAN STILL BYPASS WITH:

 note the space between the words. So this will work if the backend programmer doesn’t remove whitespace and of course uses the cURL library to parse the URL.

On to NodeJS and its associated URL parsing library:

Consider the following code used to prevent directory traversal by removing ..

Text

Description automatically generated

We can use ^^^ to bypass this (2 capital N’s)

Heres how It works.. the NN gets encoded in Unicode

 then node JS strips the both instances of xFF leaving only x2E x2E which decodes into .. leaving us with: 

So .. in this library NN/ is the new ../ make sure to try it if you identify node JS or really either way check it.

Same thing but for protocol smuggling:

Text

Description automatically generatedThis would not work! As the new lines are getting encoded, but……

We can bypass this :

Graphical user interface, text

Description automatically generatedBy submitting FF0D and FF0A around the URL as shown we can bypass

In Glibc the c libaray for URL parsing we can obfuscate our string with / to get through then

Text

Description automatically generatedthe gethostbyname() method will convert these to decimal giving us an IP

Text

Description automatically generatedusing “127.0.0.1 foo” as a URL will remove everything after the IP and leave us with the local host. This is only if the getaddrinto() method is used but try it out! Many things use this method!

Graphical user interface, text, application

Description automatically generatedMore good stuff to try! With double encodings

Graphical user interface, text, application, chat or text message

Description automatically generated

Also can smuggle URLs into http headers   
this could also work: Graphical user interface, text

Description automatically generated

Python URLlib and URLlib2

Text

Description automatically generatedNotice the spaces after the newline characters may have been patched since but still make sure to try

Graphical user interface, text

Description automatically generatedAlso works with request smuggling

Table

Description automatically generated

Text

Description automatically generated

ßß -> copy paste this to potentially bypass. Gethostname() verifies the url but cURL doesn’t

Text

Description automatically generatedPretty self explanatory on this one

p.s if you ever test github check out github enterprise

0 also stands for localhost in linux so try this too!!