Repte AI

Introducció

De cara a tenir un bon control sobre la seguretat de serveis exposats a Internet, és molt important supervisar constantment els accessos que es generen per poder detectar possibles intrusions.

Aquesta pràctica pren especial rellevància en el cas de solucions basades en web, donat que tenen una exposició elevada i existeixen multitud de tècniques per intentar atacar la plataforma (veure referències OWASP per a més informació).

Actualment existeixen patrons definits, basats en expressions regulars que ens permeten identificar amb facilitat la majoria d'atacs. Aquesta detecció es fa de manera heurística.

Malgrat els esforços per detectar i contenir els atacs, existeixen peticions que poden passar com a lícites i que no encaixen en cap d'aquests patrons i que no es pot definir cap patró que ens permeti detectar-les.

Exemple: Si algú obté unes credencials per accedir a un servei i accedeix de forma convencional, com si es tractés d'un usuari autoritzat, no encaixa amb cap patró d'atac.

Objectiu

Basant-nos en la possibilitat de detectar si una petició és maliciosa o no, i en altres paràmetres de la petició, es vol identificar totes aquelles peticions **susceptibles ser revisades** per un analista de seguretat. Es tracta d'identificar totes aquelles peticions que compleixin amb les següents condicions:

- Que no estiguin identificades com a una petició maliciosa.
- Que estiguin desviades del comportament habitual de navegació dels usuaris.

El resultat ha de ser una puntuació (**score**) sobre cada registre que ens defineixi si s'adapta més o menys amb els paràmetres de normalitat.

Això ens ha de permetre definir un llindar a partir del qual aquests registres s'incorporin a un **SIEM** per poder ser revisats per un analista. (Veure referències per saber què és un SIEM).

El procés de puntuar cada línia de registre hauria de poder funcionar en temps real, de manera que, a mida que es van generant aquestes línies de log en els fitxers, hi hagi un software capaç d'interrogar el motor d'AI que pugui donar la puntuació.

Cal que el sistema pugui anar aprenent a mida que es va generant nova informació i van canviant les tendències de navegació sobre la web.

Consideracions que poden influir en canvis de les tendències:

- Les webs son vives, per tant, és possible que es generin noves URLs (veure referències) a mida que qui gestiona a web va generant nous continguts i/o publicacions.
- Poden haver-hi tendències en funció de llançament de nous productes o serveis.

- Poden haver períodes on els administradors de la web entrin a fer modificacions als continguts a través dels panells de control o backends que també generen línies en els logs.

Material base

Per poder fer una proposta proporcionem un conjunt de registres (logs) i un conjunt de regles basades en expressions regulars que ens permeten identificar les peticions malicioses.

Estructura d'un arxiu de log

Cada solució de servei web genera un format de log diferent, però al cap i a la fi, tots els fabricants acaben generant més o menys la mateixa informació, que pot estar estructurada de manera diferent.

Nosaltres ens basarem en logs generats per motors web basats en Apache HTTP Server (veure referència).

Aquest servidor ens permet modificar a través de directives l'estructura de les dades que es generen a l'arxiu de log.

Nosaltres ens basarem en el següent format:

```
LogFormat "%V %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\""
```

Per entendre el significat, podem consultar:
https://httpd.apache.org/docs/2.4/mod/mod_log_config.html

A grans trets el que registra és:

- %V: The server name according to the UseCanonicalName setting.
- %h: Remote hostname. Will log the IP address if HostnameLookups is set to Off, which is the default. If it logs the hostname for only a few hosts, you probably have access control directives mentioning them by name. See the Require host documentation.
- %l: Remote logname (from identd, if supplied). This will return a dash unless mod_ident is present and IdentityCheck is set On.
- %u: Remote user if the request was authenticated. May be bogus if return status (%s) is 401 (unauthorized).
- %t: Time the request was received, in the format [18/Sep/2011:19:18:28 -0400]. The last number indicates the timezone offset from GMT
- %r: First line of request.
- %>s: Status. For requests that have been internally redirected, this is the status of the original request. Use %>s for the final status.
- %O: Bytes sent, including headers. May be zero in rare cases such as when a request is aborted before a response is sent. You need to enable mod_logio to use this.
- %{VARNAME}i: The contents of VARNAME: header line(s) in the request sent to the server. Changes made by other modules (e.g. mod_headers) affect this. If you're interested in what the request header was prior to when most modules would have modified it, use mod_setenvif to copy the header into an internal

environment variable and log that value with the %{VARNAME}e described above.

Exemple de línies de log:

```
www.sitgesanytime.com 92.184.116.229 - - [23/Jan/2024:00:00:01 +0100]
"GET /media/site1/cache/images/mgl3751-playa-balmins-redim-w800-
h600.jpg HTTP/2.0" 200 89277 "https://www.google.com/" "Mozilla/5.0
(iPhone; CPU iPhone OS 15_3 like Mac OS X) AppleWebKit/605.1.15
(KHTML, like Gecko) Version/15.3 Mobile/15E148 Safari/604.1"

www.sitgesanytime.com 23.22.35.162 - - [23/Jan/2024:00:00:04 +0100]
"GET /fr/pl415/blog/llistat-blog/id109/sitgestiu-2023-la-cultura-i-l-
autenticitat-t-esperen-a-sitges.htm HTTP/1.1" 200 17495 "-"
"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_1) AppleWebKit/600.2.5
(KHTML, like Gecko) Version/8.0.2 Safari/600.2.5 (Amazonbot/0.1;
+https://developer.amazon.com/support/amazonbot)"

www.sitgesanytime.com 66.249.70.99 - - [23/Jan/2024:00:00:04 +0100]
"GET /media/site1/cache/images/77e0c046-6a0d-4600-87d6-
8eb39c475840.jpeg HTTP/1.1" 304 6170 "-" "Googlebot-Image/1.0"

www.sitgestur.cat 66.249.70.164 - - [23/Jan/2024:00:00:12 +0100] "GET
/robots.txt HTTP/1.1" 301 538 "-" "Mozilla/5.0 (compatible;
Googlebot/2.1; +http://www.google.com/bot.html)"

www.sitgestur.cat 66.249.70.163 - - [23/Jan/2024:00:00:12 +0100] "GET
/ HTTP/1.1" 301 518 "-" "Mozilla/5.0 (Linux; Android 6.0.1; Nexus 5X
Build/MMB29P) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/120.0.6099.224 Mobile Safari/537.36 (compatible; Googlebot/2.1;
+http://www.google.com/bot.html)"
```

Interpretació:

- %V: El servidor respon a diversos noms (www.sitgesanytime.com i www.sitgestur.cat) però respon amb els mateixos continguts.
- %h: això correspon a l'adreça IP de qui està fent la petició al servidor web.
- %l: aquí es registraria una etiqueta si el registre provingués d'un servidor remot. Com que en aquest cas el propi servidor és qui ofereix també el servei de registre, s'emmagatzema un guió "-" que indica que el procés de registre és local. És una informació que no té rellevància per classificar la informació.
- %u: aquí es registraria si l'accés a la pàgina es fa a través d'un usuari que està autenticat contra el servidor. No és habitual que sigui així, perquè l'autenticació acostuma a gestionar-se des de la pròpia aplicació que es serveix, per tant, l'autenticació no ve donada pel sistema operatiu. En cas de que no hi hagi usuari autenticat, es registra un guió "-".
- %t: aquí es registre la data, la hora i la zona horària de la petició.
- %r: Això registra les peticions al servidor, on podem identificar el mètode (GET, POST, HEAD, OPTIONS, etc.) i la URL relativa al lloc web amb els seus paràmetres si la petició es fa mitjançant el mètode GET. Si la petició ve pel mètode POST, no es registra el payload de la petició.
- %>s: Status. Aquí es registra el codi de resposta del servidor, que indica si s'ha pogut servir la petició de forma correcta o no. És important entendre els codis de resposta. Els més habituals són:

    - 200: resposta correcta
    - 30X: redirecció a una altra URL
    - 403: accés denegat
    - 404: pàgina no trobada
    - 50X: error a la banda de servidor
- %O: número de bytes enviats.
- %{VARNAME}i: tenim dues variables que s'obtenen a partir de la petició que fa el client:
    - REFERER: ens indica des de quina pàgina ha vingut la petició, és a dir, si el client abans estava visualitzant una pàgina i ha fet clic sobre un enllaç, aquí veurem la pàgina original des de la que ens han entrat. En cas que hagin entrat de manera directa, es registra "-".
    - USER-AGENT: ens dona informació tècnica sobre el navegador que s'ha utilitzat per accedir a la pàgina.

S'ha marcat de color verd aquells aspectes que ens poden proporcionar informació interessant, en groc aquells que són secundaris i en negre aquells que són irrellevants.

User-agent

Aquest paràmetre ens permet identificar detalls tècnics sobre el navegador que s'utilitza.

No només ens permet identificar el navegador sinó que també ens permet identificar si la petició prové d'un robot que està indexant la web.

Exemple:

```
Mozilla/5.0 (Linux; Android 6.0.1; Nexus 5X Build/MMB29P)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.224 Mobile
Safari/537.36 (compatible; Googlebot/2.1;
+http://www.google.com/bot.html)
```

En aquest cas veiem que la petició prové presumptament del cercador de Google.

També podem identificar el tipus de dispositiu, que seria un dispositiu basat en Android model Nexus 5X Build/MMB29P.

Cal entendre que aquesta informació la passa qui sol·licita la informació i, per tant, podria ser falsa.

Podem trobar moltes pàgines a Internet on ens explica diferents user-agents vàlids utilitzats pels navegadors i per les plataformes.

Exemple: https://www.useragents.me/

Adreces IP d'origen

L'adreça IP des d'on ens ve la petició ens pot proporcionar informació molt interessant, en combinació amb altres camps.

La primera cosa que cal tenir en compte és que l'adreça IP ens permet geolocalitzar la procedència de qui fa la petició. Mitjançant aquesta informació, es pot enriquir les dades obtingudes i ens poden donar informació molt rellevant.

Exemple: *si és una web que acostuma a tenir visites de determinats països i rebem una petició provinent d'un país no habitual, això podria ser un indicador de que és una petició que no encaixa amb els patrons de navegació estàndard*.

Existeixen diverses bases de dades a Internet que són de pagament, però també hi ha versions més limitades que són open source i que les podem descarregar per poder ser consultades offline, amb la finalitat d'enriquir les dades dels logs.

Proporcionem alguns exemples de serveis de geolocalització:

- https://www.ip2location.com/
- https://dev.maxmind.com/geoip/geolite2-free-geolocation-data

Aquestes bases de dades poden arribar a ser molt precises, però no sempre ho són, perquè la localització pot estar associada a la seu social de l'ISP que dona el servei d'accés a Internet. En aquest cas, la informació més interessant és la del país.

Cal comentar que aquest paràmetre podria tenir una relació directa amb l'hora i el user-agent, de manera que el més normal seria que en un interval curt de temps, totes les peticions provinents de la mateix adreça IP vinguessin marcades amb el mateix user-agent, perquè es tracta de la mateixa persona, que utilitza el mateix navegador.

Si hi ha diferències en el user-agent, pot ser principalment per 2 motius:

1. Que darrere d'aquella adreça IP hi hagi una empresa on hi ha diversos usuaris, cadascun d'ells amb un navegador diferents, que accedeixen de manera simultània a la web.
2. Que siguin peticions d'un atacant que emmascara cada petició amb user-agents aleatoris perquè no se'l pugui identificar.

La combinació IP + user-agent ens permet més o menys identificar el visitant (qui).

Per altra banda, també es pot enriquir la informació relativa a les IP si les identifiquem en llistes negres.

Per exemple:

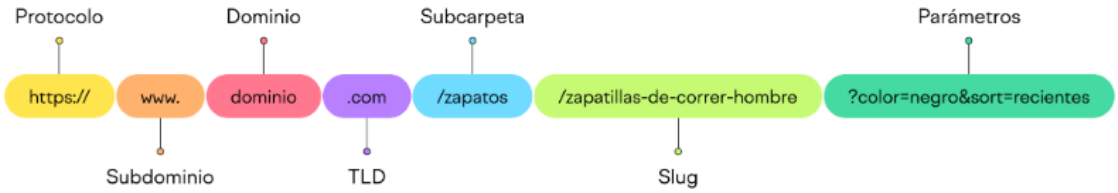
- Recopilació d'adreces IP que estan en llistes negres: https://myip.ms/files/blacklist/csf/latest_blacklist.txt
- Nodes de sortida de TOR (veure referències) https://check.torproject.org/torbulkexitlist

Mètodes HTTP

Existeixen diversos mètodes per fer peticions al servidor web, però els més usats són els següents:

- **GET**: s'utilitza a la majoria de peticions. Els paràmetres es poden veure a la URL.
- **POST**: s'utilitza habitualment quan es volen enviar dades a través d'un formulari. Els paràmetres s'envien en el cos de la petició, per tant, no es veuen a la URL i per tant, no queden registrats per defecte en els logs. Cal entendre que un paràmetre podria ser un fitxer que estem pujant a una web. Si el fitxer és molt gran i el registrem en els logs, no tindria massa sentit.

Una URL té el següent aspecte:



El protocol no el registrem en el log, però pot ser HTTP o HTTPS. En aquest cas és irrellevant. La part de subdomini+domini+TLD ve donat per la part del log identificada amb el paràmetre %V, i la part de subcarpeta+slug+paràmetres ve identificada per %r, juntament amb el mètode i el protocol.

Exemple:

```
GET /plantilles/turisme/css/estils-capcalera.css?v=3 HTTP/2.0
```

Aquí podem veure les següents coses:

- El mètode de la petició que és GET.
- La URL amb els seus paràmetres que és /plantilles/turisme/css/estils-capcalera.css?v=3
- La versió del protocol utilitzat que és HTTP/2.0

El caràcter ? serveix per separar la part dels paràmetres a la URL, que van concatenats amb nom del paràmetre, el símbol igual "=" i un valor. Els diversos paràmetres que es passen, van separats pel caràcter "&".

Exemple:

```
/plantilles/turisme/eltemps/wstemps.php?site=1&lang=en
```

Aquí tenim ds paràmetres:

- **site**, que té valor "1"
- **lang**, que té valor "en"

Cal dir que és molt habitual trobar valors estranys en els paràmetres, que indiquen que poden indicar que un atacant ens està intentant fer algun tipus d'atac, en el cas de que els valors introduïts no es tractin correctament a la banda del servidor. Els atacs més habituals poden ser els d'injecció, XSS, path tranversal, etc.

Codis de resposta

Hi ha diversos codis de resposta que en poden indicar coses:

- **200**: indica que la petició s'ha processat correctament. Si això passa però veiem que la URL correspon a una petició sospitosa, això pot indicar que l'atacant ha aconseguit el seu propòsit. Del contrari, no hagués obtingut una resposta de tipus 200.
- **301** o **302**: corresponen a redireccions. Això significa que quan el visitant accedeix a la pàgina, el servidor li indica que cal redirigir-se a una URL diferent, de manera que el navegador ja porta a l'usuari directament a aquella pàgina.
- **403**: indica que l'usuari intenta accedir a un recurs al qual no té accés. Si veiem moltes peticions d'aquest tipus, podria indicar que algun atacant està intentant accedir per força bruta a un recurs protegit.

- **404**: indica que la pàgina no s'ha trobat. Si veiem moltes peticions d'aquest tipus procedents d'una mateixa adreça IP, podria ser que estiguessin fent un reconeixement, per exemple per trobar directoris ocults que poguessin oferir informació sensible com passwords, backups de la base de dades, etc.
- **50X**: aquests codis corresponen a errors en el servidor, en el moment que intenta processar dinàmicament una petició. Això pot ser degut a que hi ha un error en el codi de l'aplicació i que un accés normal l'hagi provocat, o que l'error s'hagi disparat per un atacant que està manipulant paràmetres per intentar provocar aquests errors.

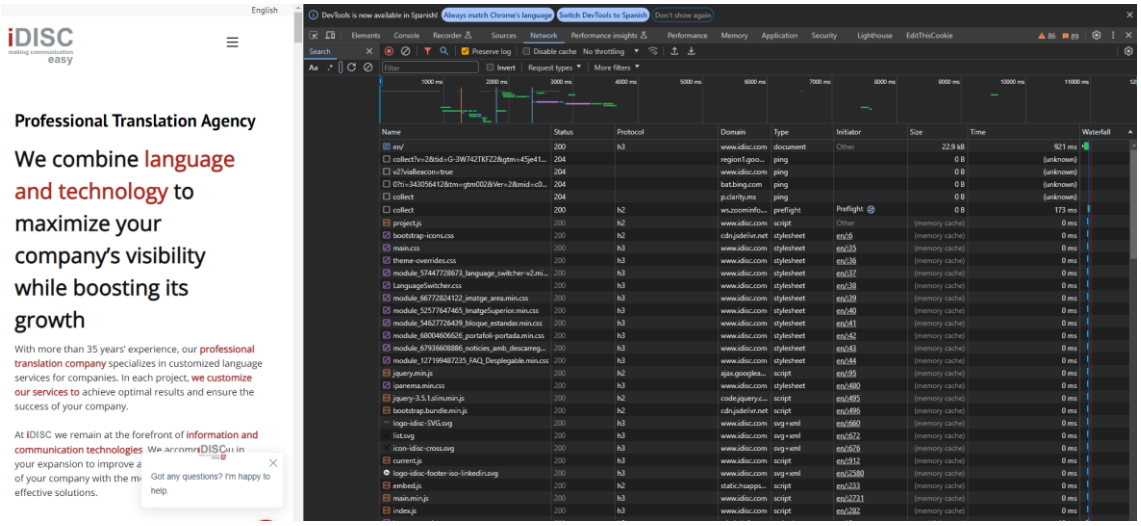
Referer

El referer ens indica la pàgina anterior des de la qual ha arribat la petició.

En condicions normals, quan un usuari navega per dins d'una web, hauríem d'observar que ens arriben peticions espaiades en el temps, és a dir, no seria normal que un usuari de manera manual tingués la capacitat per fer 20 peticions en un segon. A més, hauríem de veure que, si les ordenem cronològicament, el referer d'una petició seria la pàgina que l'usuari havia visitat anteriorment. Tampoc seria normal que un usuari fes moltes peticions a diferents URLs d'una mateixa web, sense que indiqués cap referer. Això ens indicaria una anomalia en la navegació que està fent.

Cal entendre també que una petició a una pàgina de la web, en realitat genera múltiples línies de log, donat que, per a cada element que sol·licitem es fa una petició. Els elements de la pàgina poden ser les imatges, fulles d'estil, llibreries de javascript, etc.

Exemple de peticions que genera la càrrega d'una sola pàgina:



Nota: Per poder visualitzar informació tècnica de les peticions que fa un navegador com Google Chrome, podem prémer F12 i anar a la pestanya network.

Regles de processament d'un SIEM

Introducció

En aquest material, també es proporcionen 3 arxius utilitzats en un SIEM, en aquest cas correspon a Wazuh, que és una solució open source, per processar de manera heurística aquests logs d'activitat dels servidors web.

Arxius:

- **0375-web-accesslog_decoders.xml**: serveix per identificar el format dels arxius dels servidors web de diferent tipus, fent una decodificació inicial.
- **0245-web_rules.xml**: serveix per identificar alertes o patrons d'atac relacionats amb logs de servidors web.
- **0250-apache_rules.xml**: serveix per identificar alertes o patrons d'atac relacionats amb logs de servidors web Apache.

Els decoders són per identificar el tipus de log i les "rules" són per aplicar regles sobre els logs que encaixen amb un decoder. En aquest cas, disposem de 2 arxius de regles i el més important és el 0245-web_rules.xml que conté patrons d'atac o alertes genèriques per a qualsevol servidor web.

A continuació s'explica una de les regles.

```
<rule id="31104" level="6">
  <if_sid>31100</if_sid>

  <!-- Attempt to do directory transversal, simple sql injections,
    -  or access to the etc or bin directory (unix). -->
  <url>%027|%00|%01|%7f|%2E%2E|%0A|%0D|../..|..\..|echo;|</url>
  <url>cmd.exe|root.exe|_mem_bin|msadc|/winnt/|/boot.ini|</url>
  <url>/x90/|default.ida|/sumthin|nsiislog.dll|chmod%|wget%|cd%20|</url>
  <url>exec%20|../..//|%5C../%5C|./././/|2e%2e%5c%2e|\x5C\x5C</url>
  <description>Common web attack.</description>
  <mitre>
    <id>T1055</id>
    <id>T1083</id>
    <id>T1190</id>
  </mitre>
  <group>attack,pci_dss_6.5,pci_dss_11.4,pci_dss_6.5.1,gdpr_IV_35.7.d,nist_800_53_SA.11,n
</rule>
```

La regla la veiem identificada amb un codi, en aquest cas és el 31104 i posa com a condició que prèviament el patró encaixi amb una altra regla amb codi 31100.

Aquesta regla posa com a condició que ha de correspondre a una línia de log d'un servidor web.

```
<group name="web,accesslog,">
  <rule id="31100" level="0">
    <category>web-log</category>
    <description>Access log messages grouped.</description>
  </rule>
```

Tornant a la regla original, veiem que treballa amb la URL (<url>) i utilitza diverses expressions regulars que identifiquen possibles intents de fer un atac de tipus "path tranversal", a més d'identificar intents d'injectar instruccions malicioses (per a més informació consultar OWASP top 10). Això ho classifica com a "Common web attack" i ho classifica segons uns grups i segons unes tècniques d'atac segons la matriu MITRE ATT&CK (no és rellevant, però he deixat informació en les referències).

Veiem a més a més que hi ha un nivell que en aquest cas és el 6. Això ens indica el nivell de perillositat de l'alarma, que va des de 1 fins a 15, on 15 és el nivell màxim.

Referències

- OWASP Web Security Testing Guide: guia per revisar la seguretat de les aplicacions web. https://owasp.org/www-project-web-security-testing-guide/
- OWASP Top Ten: informació sobre les 10 principals amenaces o vectores d'atac que s'utilitzen per comprometre les aplicacions web. https://owasp.org/www-project-top-ten/
- Què és un SIEM? https://www.ambit-bst.com/blog/qu%C3%A9-significa-siem-y-c%C3%B3mo-funciona
- Què és una URL? https://es.semrush.com/blog/que-es-una-url
- Apache HTTP Server. https://httpd.apache.org/
- Mètodes HTTP. https://www.w3schools.com/tags/ref_httpmethods.asp
- Codis de resposta HTTP: https://en.wikipedia.org/wiki/List_of_HTTP_status_codes
- Què és TOR: https://www.xataka.com/basics/red-tor-que-como-funciona-como-se-usa
- Matriu MITRE ATT&CK: https://attack.mitre.org/
- Sintaxi de regles definides a Wazuh: https://documentation.wazuh.com/current/user-manual/ruleset/ruleset-xml-syntax/rules.html
- Treballar amb expressions regulars: https://regexone.com/