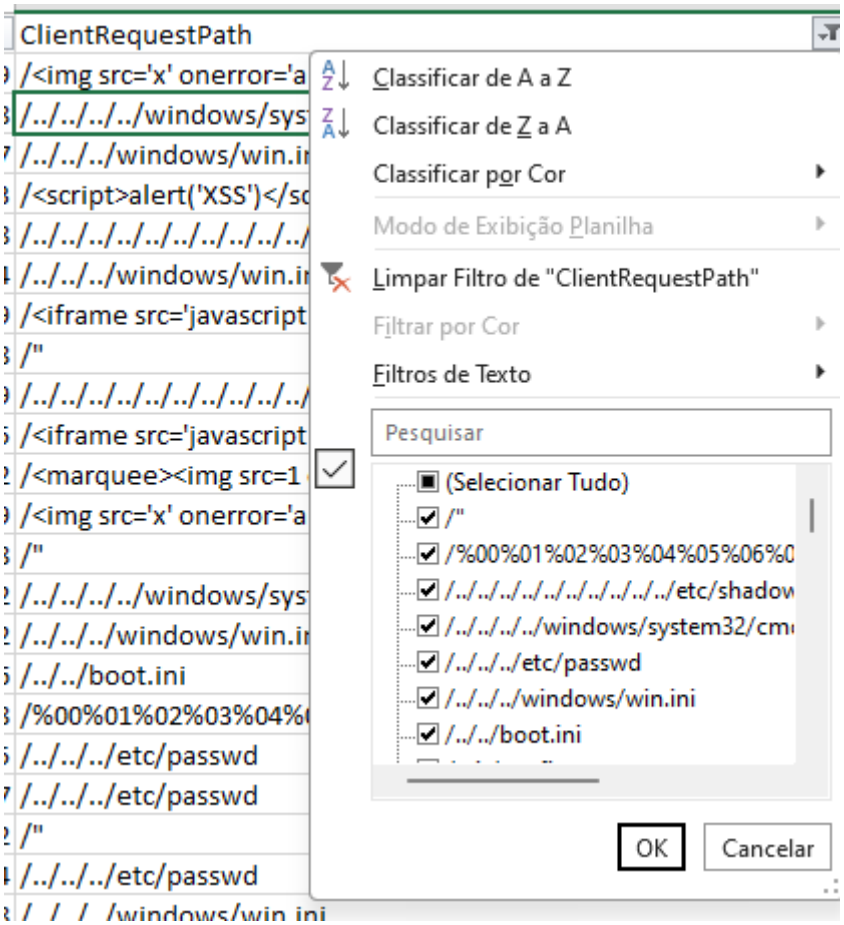
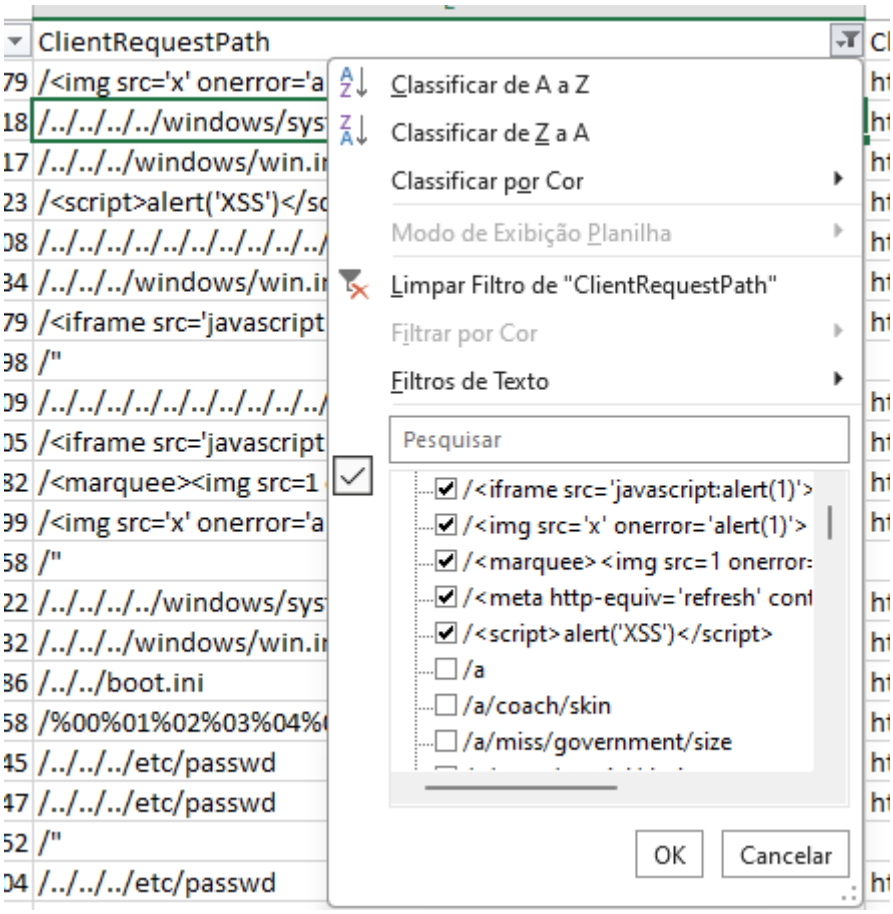
Desafio Análise de dados

Realizado processo de baixar a base de dados para início da análise, base se encontra organizada em colunas e feito o processo de filtro por meio da string “ClientRequestPath” .Com esse filtro foram identificadas cerca de 1281 requisições atendendo esse padrão de ataque vindo de 914 ips diferentes.

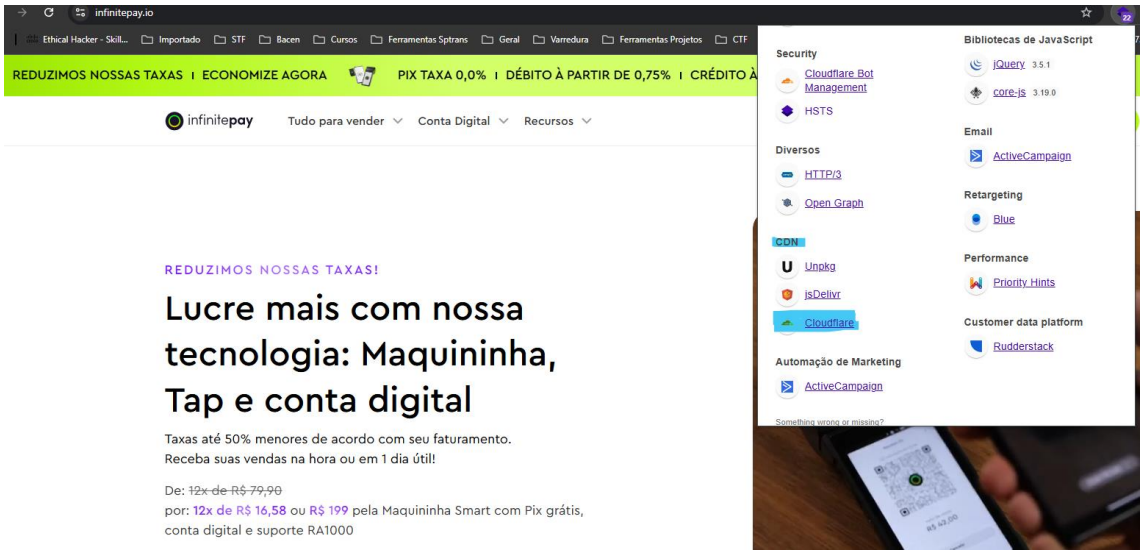




Objetivo de facilitar a analise por conta disso foram selecionados os caracteres acima pois são utilizados em ataques web.

Por meio da analise previa, e possível identificar o destino do ataque https://www.infinitepay.io/

O site identificado possui a CDN Cloudflare em uso, muitos dos ataques podem ser bloqueados ao utilizar regras personalizadas, regras quais o analista apontou no documento atual.



Com essa analise foi criado regras WAF cloudflare para bloqueio e analise de requisições

Cross-Site Scripting (XSS):

O ataque ocorre quando um atacante consegue injetar scripts maliciosos (geralmente em JavaScript) em uma aplicação web. O objetivo é fazer com que esses scripts sejam executados no navegador de outros usuários.

Regra Waf Cloudflare

Exemplos:

```
o   <img src='x' onerror='alert(1)'>
o   <script>alert('XSS')</script>
o   <iframe src='javascript:alert(1)'></iframe>
o   <marquee><img src=1 onerror=alert(1)></marquee>
```

Indo além dos dados encontrados na análise podemos criar a seguinte regra na cloudflare usando port swigger academy como base. A aplicação permite, vai bloquear completamente o uso de chaves (< ou >)

Base Port Swigger - [Cross-Site Scripting (XSS) Cheat Sheet - 2024 Edition | Web Security Academy]

Regra Waf Cloudflare

```
(http.request.uri.query matches "(<script.*?>.*?</script>)|(<.*?on[a-zA-
Z]+=['\"].*?['\"])|(<.*?javascript:.*?>)|(<img.*?on[a-zA-
Z]+=['\"].*?['\"]>)|(<iframe.*?javascript:.*?>)|(<marquee.*?on[a-zA-
Z]+=['\"].*?['\"]>)" or
 http.request.uri.query contains "<script>" or
 http.request.uri.query matches
"on(?:click|mouseover|mouseout|mousedown|mouseup|mousemove|dblclick|keydo
wn|keypress|keyup|submit|reset|focus|blur|change|input|load|error|abort|c
anplay|canplaythrough|durationchange|emptied|ended|loadeddata|loadedmetad
ata|loadstart|pause|play|playing|progress|ratechange|seeked|seeking|stall
ed|suspend|timeupdate|volumechange|waiting|afterprint|beforeprint|beforeu
nload|hashchange|message|offline|online|pagehide|pageshow|popstate|resize
|storage|unload|touchstart|touchmove|touchend|touchcancel|pointerdown|poi
ntermove|pointerup|pointercancel|pointerover|pointerout|pointerenter|poin
terleave|animationstart|animationend|animationiteration|transitionend|bef
oretoggle|toggle|webkitmouseforceup)=['\"]?.*['\"]?" or
```

```
 http.request.uri.query matches
"(javascript:alert\\(.*?\\))|(alert\\(.*?\\))"
)
```

Path Traversal (Traversal de Diretórios):

Esse ataque ocorre quando o invasor manipula caminhos de arquivos para acessar áreas que não deveriam ser acessíveis. A ideia é explorar diretórios "acima" do permitido por meio de strings como ../ (directory traversal).

Regra Waf Cloudflare

```
• (http.request.uri.path contains "../" or
• http.request.uri.path contains "%2E%2E%2F" or
• http.request.uri.path contains "/etc/passwd" or
• http.request.uri.path contains "/etc/shadow" or
• http.request.uri.path contains "/boot.ini" or
• http.request.uri.path contains "/windows/win.ini" or
• http.request.uri.path contains "/cmd.exe")
```

Assim como fizemos com a base de dados da port Swiger podemos criar uma regra usando como base repositórios do Git que contém wordlist do ataque.

Repositório -
https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Directory%20Traversal/Intruder/directory_traversal.txt

PayloadsAllTheThings/Directory Traversal/Intruder/deep_traversal.txt at master · swisskyrepo/PayloadsAllTheThings · GitHub

Regra Waf Cloudflare

```
    {
      "description": "Proteção contra Path Traversal, XSS, e File
Inclusion",
      "expression": "(http.request.uri.path contains \"../\" or " +
                    "http.request.uri.path contains \"..%2f\" or " +
                    "http.request.uri.path contains \"..%5c\" or " +
                    "http.request.uri.path contains \"%00\" or " +
                    "http.request.uri.path contains \"%2e%2e\" or " +
                    "http.request.uri.path matches
\"(?i)\\b(boot\\.ini|win\\.ini|global\\.asa|cmd\\.exe|passwd|shadow|hosts
|htaccess)\\b\" or " +
```

```
                    "http.request.uri.path matches
\"(?i)(\\/%2e%2e|\\/%5c|\\.\\.|\\/\\..|\\%c0%af|\\%2e%2e|\\%5c\\.\\.|\\.\
\%5c)\" or " +
                    "http.request.uri.query contains \"cmd=dir\" or " +
                    "http.request.uri.query matches
\"(?i)\\b(cat|id|dir|type)\\b\" or " +
                    "http.request.uri.path matches
\"(?i)<script.*?>.*?<\\/script>\" or " +
                    "http.request.uri.path matches
\"(?i)<iframe.*?>.*?<\\/iframe>\" or " +
                    "http.request.uri.path matches
\"(?i)<img.*?onerror=.*?>\" or " +
                    "http.request.uri.path matches
\"(?i)<marquee.*?>.*?<\\/marquee>\" or " +
                    "http.request.uri.path matches \"(?i)<meta.*?http-
equiv.*?>\" or " +
                    "http.request.uri.path matches
\"(?i)(\\.git\\/config|%00|%01|%02|%03|%04|%05|%06|%07)\" or " +
                    "http.request.uri.path matches
\"(?i)(a\\/|\\/\")\")",
      "action": "block"
}
```

Ataques a Arquivos Específicos do Sistema:

O invasor explora falhas em permissões ou validação de caminhos de arquivos para acessar arquivos sensíveis no servidor. Por exemplo, explorando um upload de arquivo mal configurado para enviar um script PHP malicioso, ou explorando endpoints vulneráveis que permitem a leitura de arquivo

Regra Waf Cloudflare

```
/boot.ini (Windows)
/etc/passwd (Linux)
/etc/shadow (Linux)
```

Forceful Browsing (Navegação Forçada)

Exemplo identificado
URL: /admin.php?user=admin&password=admin

Forceful Browsinge é um tipo de ataque onde o invasor realiza a tentativa de acessar um painel administrativo diretamente, explorando endpoints conhecidos.

tablet	49082	6405 /wp-admin/setup-config.php

Regra Waf Cloudflare

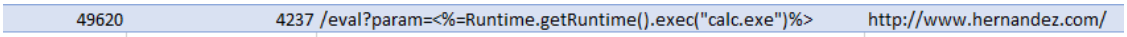
```
(http.request.uri.path matches
"(?i)^/(admin|config|private|dashboard|settings|login|wp-admin)(/.*)?$"
and not cf.access.did_validate)
```

Command Injection

URL: /eval?param=<%=Runtime.getRuntime().exec("calc.exe")%>

Command Injection é um tipo de ataque onde o invasor manipula entradas de um sistema para executar comandos diretamente no servidor.

Caminhos como esse aparecem na base, sugere a tentativa da exploração de vulnerabilidades de execução remota



49620	4237 /eval?param=<%=Runtime.getRuntime().exec("calc.exe")%>	http://www.hernandez.com/

Regra Waf Cloudflare

```
(http.request.uri.query matches
"(\\bRuntime\\.getRuntime\\(\\)|\\bexec\\(|\\bcmd\\.exe\\b|\\bcalc\\.exe\
\b|\\bid\\b|\\bwhoami\\b|\\bwget\\b|\\bcurl\\b|\\bpython\\b|\\bperl\\b|\\
brm -
rf\\b|\\bls\\b|\\bnc\\b|\\btelnet\\b|\\bping\\b|\\btraceroute\\b|\\bcat\\
b|\\becho\\b)" or
http.request.uri.query contains "../" or
http.request.uri.query contains "%00" or
http.request.uri.query matches ".*[;|&`\\$><].*")
```

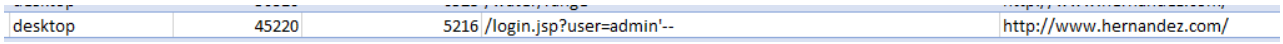
SQL Injection (SQLi)

Exemplo:

URL: /login.jsp?user=admin'--

Ataque é uma tentativa de injeção SQL para manipular consultas ao banco de dados e ignorar autenticação.

URLs como /login.jsp são altamente suscetíveis a testes de SQLi.



desktop	45220	5216	/login.jsp?user=admin'--	http://www.hernandez.com/

Regra Waf Cloudflare

```
http.request.uri.query contains "' or 1=1 --" or
http.request.uri.query matches "(?i)(\\bor 1=1|or '1'='1|and 1=1|and
'1'='1)" or
http.request.uri.query matches "(\\b(select .*? from|union select .*?
from|insert into .*? values|update .*? set|delete from|drop table|alter
table|exec\\b))")
```

HTML Injection com Refresh Automático:

Esse ataque é uma variação de injeção HTML em que o invasor utiliza tags ou scripts HTML que desencadeiam atualizações automáticas ou redirecionamentos de páginas.

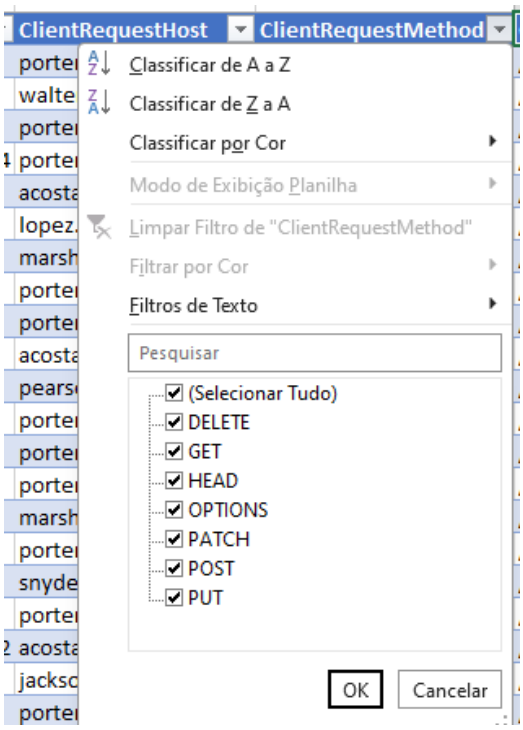
Exemplo:

```
<meta http-equiv='refresh' content='0'>
```

Regra Waf Cloudflare

```
(http.request.uri.query contains "<meta http-equiv")
```

Outra regra que é interessante habilitar seria de bloqueio de métodos HTTP, falo isso pois foi observado na planilha que havia solicitação referente aos métodos.



Bloqueio por métodos HTTP e algo que deve ser conversado com os desenvolveres, pois, algumas aplicações como api usam para realizar validação e requisições porem métodos podem ser usados por atacantes para excluir dados, modificar recursos ou explorar vulnerabilidades no servidor se mal configurados, permitem ações destrutivas, como upload de arquivos maliciosos (PUT) ou captura de informações sensíveis (TRACE). Como não tenho essa informação irei considerar bloquear, usando a regra abaixo.

Métodos a bloquear (por padrão):

- **PUT**: A menos que seja estritamente necessário (ex.: APIs).
- **DELETE**: Altamente sensível; bloquear se não for usado. Validar com desenvolvedores da aplicação
- **PATCH**: Bloquear a menos que seja necessário.
- **TRACE**: Não listado, mas deve ser sempre bloqueado, pois pode expor informações sensíveis.

Nome: "Block Unused HTTP Methods".

Regra Waf Cloudflare

```
(http.request.method in {"PUT" "DELETE" "PATCH" "TRACE"})
```

1. Regras para Símbolos e URL Encoding

URL encoding é uma forma de transformar caracteres especiais em uma URL para que sejam transmitidos de forma segura pela internet usado para evitar interpretações incorretas de caracteres especiais pelo servidor.

Regra 1.1: Bloqueio de caracteres perigosos

```
(http.request.uri.path contains "<" or
 http.request.uri.path contains ">" or
 http.request.uri.path contains "'" or
 http.request.uri.path contains "\"" or
 http.request.uri.path contains ";" or
 http.request.uri.path contains "&" or
 http.request.uri.path contains "%")
```

Regra 1.2: Normalização e inspeção de URL encoding

```
(http.request.uri.path contains "%3C" or
 http.request.uri.path contains "%3E" or
 http.request.uri.path contains "%22" or
 http.request.uri.path contains "%27" or
 http.request.uri.path contains "%3B" or
 http.request.uri.path contains "%26" or
 http.request.uri.path contains "%25")
```

---

2. Adicionar Cabeçalhos de Segurança

Cabeçalhos de segurança são configurados no servidor para ajudar a proteger aplicações web contra ataques comuns, previne **XSS (Cross-Site Scripting)**, **Clickjacking**, e outros ataques explorando vulnerabilidades.

Controle de recursos:

Restringe de onde scripts, imagens ou arquivos podem ser carregados, reduzindo riscos de conteúdo malicioso.

Melhoria da segurança do navegador:

Ajuda a bloquear comportamentos inseguros ou não intencionais que podem ser explorados por atacantes.

Cabeçalhos recomendados:

```
Strict-Transport-Security: "max-age=31536000; includeSubDomains"
X-Content-Type-Options: "nosniff"
X-Frame-Options: "DENY"
Referrer-Policy: "strict-origin-when-cross-origin"
Permissions-Policy: "geolocation=(), microphone=(), camera=()"
```

Caso o site não possui esses cabeçalhos, adicione-os com regras como:

Regra 2.1: Adicionar cabeçalhos de resposta

```
Headers to Add:
  - Header Name: Strict-Transport-Security
    Value: max-age=31536000; includeSubDomains
  - Header Name: X-Content-Type-Options
    Value: nosniff
  - Header Name: X-Frame-Options
    Value: DENY
  - Header Name: Referrer-Policy
    Value: strict-origin-when-cross-origin
  - Header Name: Permissions-Policy
    Value: geolocation=(), microphone=(), camera=()
```

3. Monitoramento de User-Agents

Atacantes menos experientes podem esquecer de trocar o user agent, e por causa disso, o defensor consegue se proteger mais fácil bloqueando user agents conhecidos como maliciosos.

Regra 3.1: Bloqueio de User-Agents fixos

```
(http.user_agent eq "malicious-bot" or
 http.user_agent eq "curl/7.64.1" or
 http.user_agent eq "wget/1.20.3")
```

Regra 3.2: Detectar repetição de User-Agents com IPs rotativos

```
(count(http.user_agent) > 10 and count(http.request.ip) > 10 and
http.user_agent in {"user-agent1", "user-agent2"})
```

Durante a análise foi identificado que o país com maior quantidade de requisições maliciosas foi US e IN.

/research/partner/she	2024-11-06 18:07:24+00:00	infinitepay.io	396982	us	tablet	38918	6534 /<script>alert('XSS')</script>
/research/partner/she	2024-11-06 11:13:26+00:00	infinitepay.io	396982	us	tablet	53394	6552 /<marquee><img src=1 onerror=alert(1)></marquee>
/research/partner/she	2024-11-11 11:56:08+00:00	infinitepay.io	396982	us	desktop	55194	6476 /%00%01%02%03%04%05%06%07
/research/partner/she	2024-11-11 19:40:51+00:00	infinitepay.io	396982	us	desktop	47908	6477 /<iframe src='javascript:alert(1)'></iframe>
/research/partner/she	2024-11-11 20:50:18+00:00	infinitepay.io	396982	us	tablet	42962	6462 /<marquee><img src=1 onerror=alert(1)></marquee>
/research/partner/she	2024-11-05 23:14:14+00:00	infinitepay.io	396982	us	tablet	48206	6508 /../../../windows/win.ini
/research/partner/she	2024-11-07 16:35:23+00:00	infinitepay.io	396982	us	desktop	57250	6468 /<marquee><img src=1 onerror=alert(1)></marquee>
/research/partner/she	2024-11-07 23:27:50+00:00	infinitepay.io	396982	us	tablet	60736	6363 /../../boot.ini
/research/partner/she	2024-11-07 13:51:26+00:00	infinitepay.io	396982	us	desktop	57392	6393 /../../../../windows/system32/cmd.exe
/research/partner/she	2024-11-10 16:01:39+00:00	infinitepay.io	396982	us	tablet	47948	6538 /<script>alert('XSS')</script>
/research/partner/she	2024-11-10 14:43:55+00:00	infinitepay.io	396982	us	tablet	44672	6434 /<iframe src='javascript:alert(1)'></iframe>
/research/partner/she	2024-11-10 18:07:27+00:00	infinitepay.io	396982	us	tablet	46868	6500 /%00%01%02%03%04%05%06%07
/research/partner/she	2024-11-08 11:23:14+00:00	infinitepay.io	396982	us	tablet	33100	6546 /../../boot.ini
/research/partner/she	2024-11-14 14:22:16+00:00	infinitepay.io	396982	us	desktop	37068	6577 /../../../windows/win.ini
/research/partner/she	2024-11-09 16:30:36+00:00	infinitepay.io	396982	us	desktop	55236	6446 /../../../../windows/system32/cmd.exe
/research/partner/she	2024-11-09 22:50:00+00:00	infinitepay.io	396982	us	tablet	52158	6538 /../../../windows/win.ini
/research/partner/she	2024-11-09 20:41:24+00:00	infinitepay.io	396982	us	tablet	45840	6569 /<iframe src='javascript:alert(1)'></iframe>
/research/partner/she	2024-11-09 03:20:49+00:00	infinitepay.io	396982	us	desktop	48582	6388 /../../../windows/win.ini
/research/partner/she	2024-11-13 00:45:44+00:00	infinitepay.io	396982	us	desktop	33950	6427 /<meta http-equiv='refresh' content='0
/research/partner/she	2024-11-13 15:51:32+00:00	infinitepay.io	396982	us	desktop	39276	6495 /../../boot.ini
/war	2024-11-06 21:51:27+00:00	infinitepay.io	396982	in	desktop	4473	6960 /<iframe src='javascript:alert(1)'></iframe>
/war	2024-11-06 00:05:30+00:00	infinitepay.io	396982	in	tablet	2780	9106 /<meta http-equiv='refresh' content='0

Onde ASN 396982 identificado como a que está sendo usada pela maioria das requisições maliciosa, cerca de 498 requisições, com isso a possibilidade de bloqueio de forma manual ou por script.

Name: "Block Malicious ASN"

```
(ip.geoip.asnum eq 396982) or
(ip.geoip.asnum eq 267309 and ip.geoip.country eq "IN") or
(ip.geoip.asnum eq 212238)
```

Podemos melhorar a regra, ASN são usadas por diversos range de ips e alguns podem ser de pessoas que estejam querendo realmente acessar ou utilizar o site, por conta o processo de block fosse feito de forma automática e pos o período de 24 horas realizasse processo de desbloqueio de forma automática.

Abaixo está o processo para criar e gerenciar uma regra no WAF da Cloudflare que bloqueia ASNs identificados como maliciosos, com remoção automática após 24 horas. Isso pode ser feito usando a API da Cloudflare.

Configuração Inicial

Antes de criar a regra, você precisa:

- **Token da API**: Gere no painel da Cloudflare com permissões para gerenciar firewall e regras WAF.
- **Zone ID**: ID do domínio para o qual as regras serão aplicadas.

---

2. Estrutura da Regra

A regra será criada com o seguinte critério:

- **Condição**: Bloqueia solicitações de um ASN específico (ip.geoip.asnum).
- **Ação**: **Block**.
- **Remoção Automática**: Um script irá monitorar e remover a regra após 24 horas.

---

3. Script Python para Criar e Gerenciar a Regra

Passos

1. **Criar a Regra de Bloqueio**
2. **Agendar Remoção Automática Após 24 Horas**

Script:

```
import requests
import time
from datetime import datetime, timedelta

# Configurações da API Cloudflare
API_TOKEN = "YOUR_API_TOKEN"
ZONE_ID = "YOUR_ZONE_ID"
HEADERS = {
    "Authorization": f"Bearer {API_TOKEN}",
    "Content-Type": "application/json"
}

# Função para criar uma regra de bloqueio por ASN
def create_block_rule(asn, description):
    url =
f"https://api.cloudflare.com/client/v4/zones/{ZONE_ID}/firewall/rules"
    data = {
        "action": "block",
```

```
        "filter": {
            "expression": f"(ip.geoip.asnum eq {asn})",
            "paused": False,
            "description": description
        },
        "description": f"Block ASN {asn} - {description}"
    }
    response = requests.post(url, headers=HEADERS, json=data)
    if response.status_code == 200:
        print(f"Rule created to block ASN {asn}")
        return response.json()["result"]["id"]
    else:
        print(f"Error creating rule: {response.text}")
        return None

# Função para remover uma regra de bloqueio
def remove_block_rule(rule_id):
    url =
f"https://api.cloudflare.com/client/v4/zones/{ZONE_ID}/firewall/rules/{ru
le_id}"
    response = requests.delete(url, headers=HEADERS)
    if response.status_code == 200:
        print(f"Rule {rule_id} removed successfully.")
    else:
        print(f"Error removing rule: {response.text}")

# Exemplo de uso
asn_to_block = 396982  # Trocar pelo ASN malicioso detectado
description = "Malicious activity detected"
rule_id = create_block_rule(asn_to_block, description)

# Aguarda 24 horas antes de remover a regra
if rule_id:
    print("Blocking for 24 hours...")
    time.sleep(24 * 60 * 60)  # 24 horas em segundos
    remove_block_rule(rule_id)
```

4. Como Configurar

1. Substitua “YOUR_API_TOKEN” pelo token da API Cloudflare.
2. Substitua “YOUR_ZONE_ID” pelo Zone ID do domínio.
3. Atualize o asn_to_block com o ASN identificado como malicioso.

---

5. Execução Automática

- **Linux/Mac**: Use cron para executar o script periodicamente.
- **Windows**: Use o Agendador de Tarefas para executar o script.

Exemplo com cron:

1. Edite o cron com:

```
2. bash
3. CopiarEditar
4. crontab -e
5.
```

6.Adicione a linha:

```
6. bash
7. CopiarEditar
8. 0 * * * * python3 /path/to/script.py
9.
```

E recomendado incluir um managed challenge (turnstile) da Cloudflare, onde tal controle implementa um mecanismo de recaptcha, bloqueando efetivamente bots e atores maliciosos, mantendo apenas o trafego legitimo sendo processado pelo servidor."

1. Configuração da Regra com Managed Challenge

Em vez de usar block, configure a ação como “managed_challenge” no WAF.

Estrutura da Regra

- **Condição**: ASN ou origem de tráfego identificado como suspeito.
- **Ação**: **Managed Challenge**.
- **Validação Automática**: Após 24 horas, a regra será removida automaticamente para minimizar impacto em usuários legítimos.

---

2. Script Python para Configurar Managed Challenge

Código:

```
import requests
import time
```

```python
# Configurações da API Cloudflare
API_TOKEN = "YOUR_API_TOKEN"
ZONE_ID = "YOUR_ZONE_ID"
HEADERS = {
    "Authorization": f"Bearer {API_TOKEN}",
    "Content-Type": "application/json"
}

# Função para criar uma regra de Managed Challenge para ASN
def create_challenge_rule(asn, description):
    url =
f"https://api.cloudflare.com/client/v4/zones/{ZONE_ID}/firewall/rules"
    data = {
        "action": "managed_challenge",
        "filter": {
            "expression": f"(ip.geoip.asnum eq {asn})",
            "paused": False,
            "description": description
        },
        "description": f"Managed Challenge for ASN {asn} - {description}"
    }
    response = requests.post(url, headers=HEADERS, json=data)
    if response.status_code == 200:
        print(f"Managed Challenge rule created for ASN {asn}")
        return response.json()["result"]["id"]
    else:
        print(f"Error creating rule: {response.text}")
        return None

# Função para remover uma regra
def remove_challenge_rule(rule_id):
    url =
f"https://api.cloudflare.com/client/v4/zones/{ZONE_ID}/firewall/rules/{ru
le_id}"
    response = requests.delete(url, headers=HEADERS)
    if response.status_code == 200:
        print(f"Rule {rule_id} removed successfully.")
    else:
        print(f"Error removing rule: {response.text}")

# Exemplo de uso
asn_to_challenge = 396982  # Trocar pelo ASN identificado
description = "Malicious activity detected"
rule_id = create_challenge_rule(asn_to_challenge, description)

# Aguarda 24 horas antes de remover a regra
if rule_id:
    print("Applying Managed Challenge for 24 hours...")
```

```
    time.sleep(24 * 60 * 60)  # 24 horas em segundos
    remove_challenge_rule(rule_id)
```

---

3. Funcionamento do Script

1. **Criação da Regra**:
    - O script adiciona uma regra de **Managed Challenge** para um ASN específico, utilizando a API da Cloudflare.
    - Os usuários suspeitos são desafiados a provar que são legítimos antes de acessar o site.
2. **Remoção Automática**:
    - Após 24 horas, o script remove automaticamente a regra para evitar impacto prolongado em usuários legítimos.
3. **Configurações a Personalizar**:
    - asn_to_challenge: Substitua pelo ASN identificado como malicioso.
    - description: Adicione uma descrição clara para identificar a regra.

---

4. Benefícios do Managed Challenge

- Reduz falsos positivos.
- Garante que usuários legítimos com IPs suspeitos possam acessar o site.
- Adiciona uma camada extra de proteção com verificação de comportamento automatizada.

Com base na análise registrada no documento e das soluções apresentadas, os seguintes ações foram realizadas

Bloqueio dos ataques

- **Cross-Site Scripting (XSS):**
- **Path Traversal (Traversal de Diretórios):**
- **Ataques a Arquivos Específicos do Sistema:**
- **HTML Injection com Refresh Automático:**
- **SQL Injection (SQLi)**
- **Forceful Browsing (Navegação Forçada)**
- **Command Injection**

Melhorias

- **Caracteres perigosos**
- **Normalização e inspeção de URL encoding**
- **Adicionar Cabeçalhos de Segurança**
- **Adicionar cabeçalhos de resposta**
- **Monitoramento de User-Agents**
- **Bloqueio de ASN**
- **E incluímos desafios de Managed Challenge**

Com isso concluo o desafio de analise de dados, onde o processo me permitiu testar meus conhecimentos como profissional de cyber segurança e melhorar minhas habilidades de analise e resposta a requisições maliciosas.