Web Tracking Forensics: Detecting and Analyzing How Cookies and Scripts Track Users Across the Web

Biškup Dorian
Babić Lovro
Blažek Ilan
Puklek Dino

Cilj projekta

Nadzire mrežni promet pregledavanja kako bi se otkrilo praćenje temeljeno na kolačićima i praćenje bez kolačića.

Pohranjuje prikupljene dokaze u bazu podataka radi naknadne analize

Bilježi i klasificira metode praćenja, uključujući kolačiće trećih strana, beacone i pokušaje fingerprintinga.

Vizualizira veze između web stranica i trackera, odnosno prikazuje koje treće strane imaju uvid u korisnikovo pregledavanje.

Podjela rada:

Bavio se snimanjem mrežnog prometa web stranica pomoću alata za presretanje HTTP/HTTPS zahtjeva te identifikacijom domena koje primaju podatke o korisnicima.

Ilan Blažek

Zadužen za detekciju i klasifikaciju trackera, razlikovanje first-party i third-party praćenja te procjenu intenziteta praćenja po web stranici

Biškup Dorian

Radio je na dizajnu baze podataka i backend skriptama za pohranu i obradu zapisa o praćenju, uključujući kolačiće i mrežne zahtjeve.

Dino Puklek

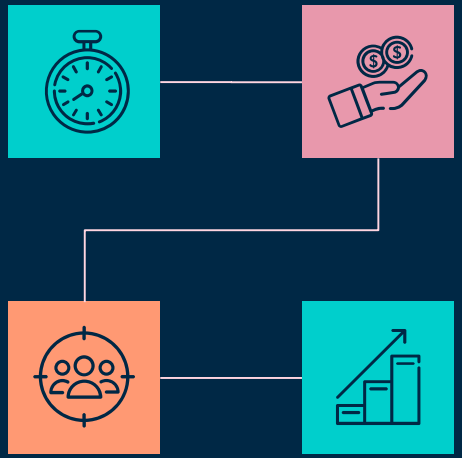
Izradio je grafičku vizualizaciju odnosa između web stranica i trackera te analizirao rezultate za završno izvještavanje i prezentaciju.

Babić Lovro

Programi korišteni u sklopu projekta:

MITMPROXY

Snimanje HTTP/HTTPS prometa
Presretanje zahtjeva, kolačića i trackera



PYTHON

Obrada i analiza podataka
Klasifikacija trackera i izvoz podataka

SQLITE

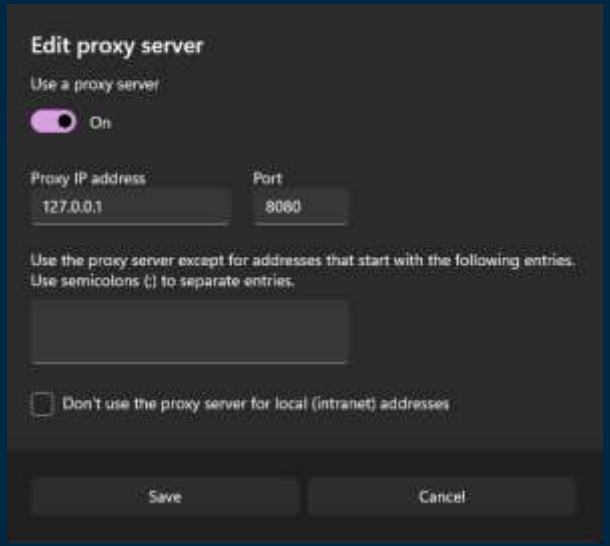
Pohrana podataka o praćenju
Baza zahtjeva, domena i kategorija

GEPHI

Vizualizacija mreže trackera
Graf odnosa web stranica i third-party domena

Postavljanje radnog okruženja

```
pip install mitmproxy
```



Skripta za snimanje HTTP/HTTPS prometa

```python
from mitmproxy import http
import json

def request(flow: http.HTTPFlow):
    referer = flow.request.headers.get("referer", "")
    site = referer.split("/")[2] if "://" in referer else ""

    entry = {
        "visited_site": site,
        "request_domain": flow.request.host,
        "url": flow.request.pretty_url,
        "method": flow.request.method,
        "cookies": dict(flow.request.cookies),
        "user_agent": flow.request.headers.get("user-agent", "")
    }

    with open("traffic_log.json", "a", encoding="utf-8") as f:
        f.write(json.dumps(entry) + "\n")
```

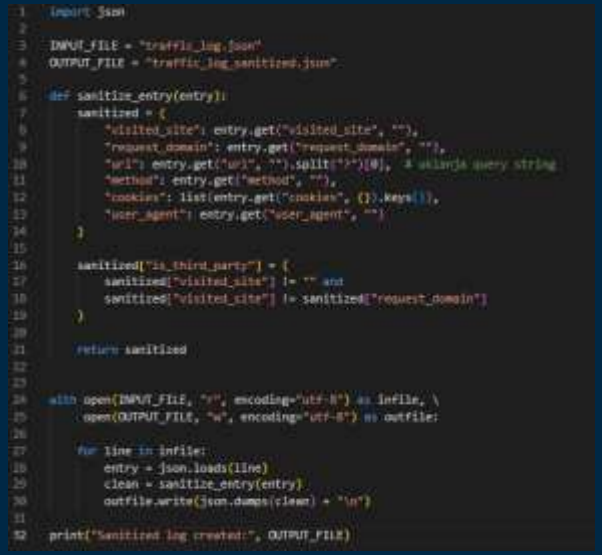
Tracker_logger.py

Sprema sirove podatke o prometu (URL, domena, cookies, headers...)

Presreće HTTP/HTTPS zahtjeve

Radi s **mitmproxyjem**

Skripta za čišćenje i pripremu logova



```python
import json

INPUT_FILE = "traffic_log.json"
OUTPUT_FILE = "traffic_log_sanitized.json"

def sanitize_entry(entry):
    sanitized = {
        "visited_site": entry.get("visited_site", ""),
        "request_domain": entry.get("request_domain", ""),
        "url": entry.get("url", "").split("?")[0],  # uklanja query string
        "method": entry.get("method", ""),
        "cookies": list(entry.get("cookies", {}).keys()),
        "user_agent": entry.get("user_agent", "")
    }

    sanitized["is_third_party"] = (
        sanitized["visited_site"] != "" and
        sanitized["visited_site"] != sanitized["request_domain"]
    )

    return sanitized


with open(INPUT_FILE, "r", encoding="utf-8") as infile, \
     open(OUTPUT_FILE, "w", encoding="utf-8") as outfile:

    for line in infile:
        entry = json.loads(line)
        clean = sanitize_entry(entry)
        outfile.write(json.dumps(clean) + "\n")

print("Sanitized log created:", OUTPUT_FILE)
```

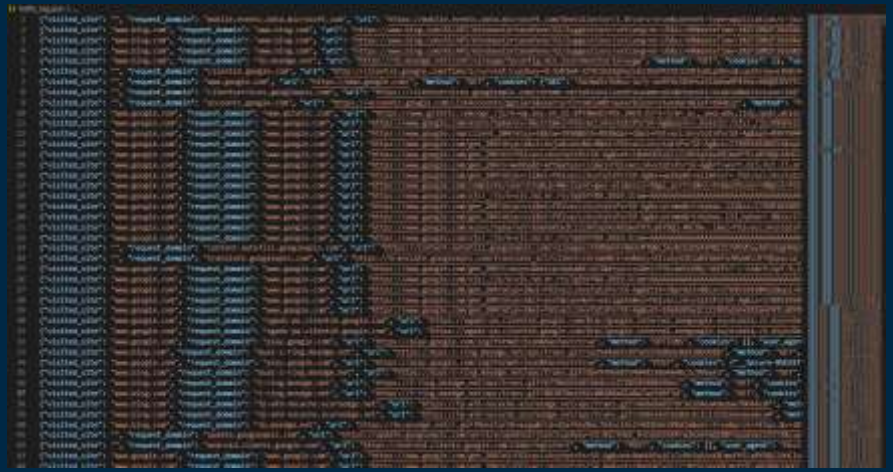
sanitize_log.py

Čisti sirove logove

Uklanja nepotrebne ili osjetljive dijelove

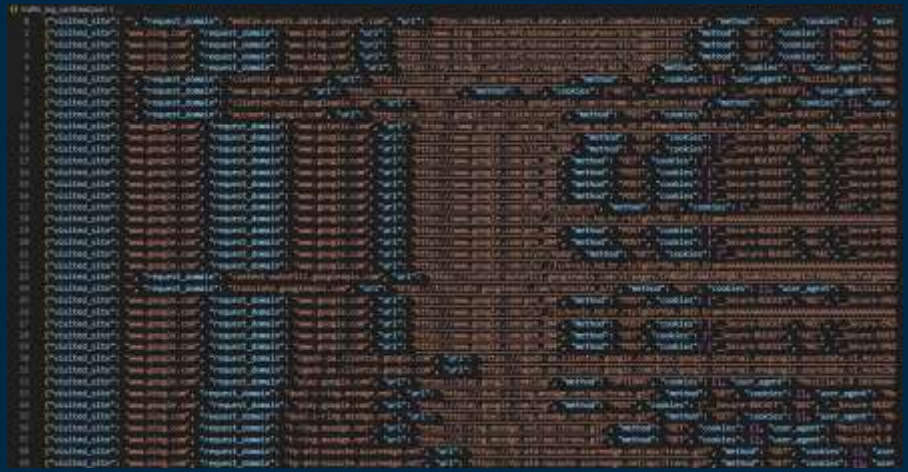
Priprema podatke za daljnju obradu

Dobiveni rezultati



traffic_log.json

Sadrži sirove podatke snimljenog mrežnog prometa – sve HTTP/HTTPS zahtjeve, dugačke URL-ove, kolačiće i tehničke detalje koji nisu odmah pogodni za analizu.



traffic_log_sanitized.json

Sadrži očišćene i strukturirane podatke – uklonjeni su nepotrebni i osjetljivi dijelovi, a zadržane su samo informacije važne za analizu web praćenja.

Kreiranje baze podataka

log_to_db.py

Skripta učitava očišćene podatke iz JSON datoteke i sprema ih u SQLite bazu podataka.

Kreira tablicu za mrežne zahtjeve i bilježi osnovne informacije potrebne za analizu web praćenja.

Provjera baze podataka

```python
import sqlite3

conn = sqlite3.connect("tracking.db")
cur = conn.cursor()

print("=== OSNOVNE STATISTIKE ===")
cur.execute("SELECT COUNT(*) FROM requests")
print("Ukupan broj zapisa:", cur.fetchone()[0])

print("\n=== PRIMJER ZAPISA ===")
cur.execute("""
SELECT visited_site, request_domain, is_third_party, category
FROM requests
LIMIT 10
""")
for row in cur.fetchall():
    print(row)

print("\n=== TRACKERI PO KATEGORIJI ===")
cur.execute("""
SELECT category, COUNT(*)
FROM requests
WHERE is_third_party = 1
GROUP BY category
""")
for row in cur.fetchall():
    print(row)

conn.close()
```

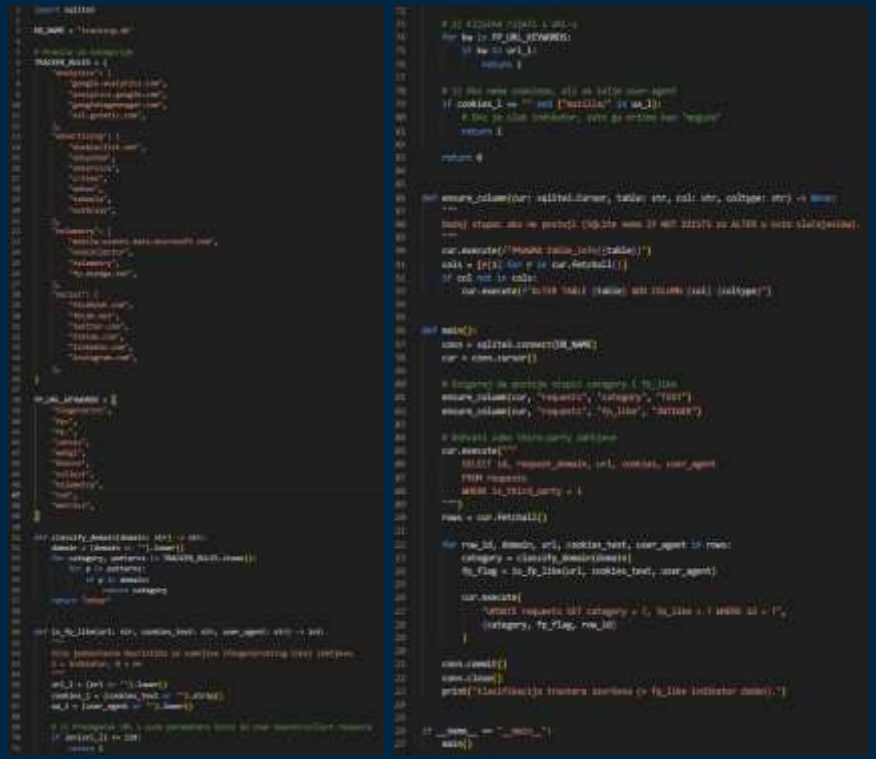
check_db.py

```
PS C:\Users\snaxy\Desktop\sis> python check_db.py
>>
=== OSNOVNE STATISTIKE ===
Ukupan broj zapisa: 7556

=== PRIMJER ZAPISA ===
('', 'mobile.events.data.microsoft.com', 1, 'telemetry')
('www.bing.com', 'www.bing.com', 0, None)
('www.bing.com', 'www.bing.com', 0, None)
('www.bing.com', 'www.bing.com', 0, None)
('www.bing.com', 'fp.msedge.net', 1, 'other')
('', 'clients2.google.com', 0, None)
('', 'www.google.com', 0, None)
('', 'clientservices.googleapis.com', 0, None)
('', 'accounts.google.com', 0, None)
('www.google.com', 'www.gstatic.com', 1, 'other')

=== TRACKERI PO KATEGORIJI ===
('advertising', 222)
('analytics', 88)
('other', 5110)
('social', 109)
('telemetry', 2)
```

Prikaz baze podataka

DB Browser (SQLite)

Detekcija i klasifikacija trackera



classify_trackers.py

sustav automatski prepoznaje i klasificira third-party zahtjeve prikupljene tijekom pregledavanja web-stranica.

Analiza se temelji na poznatim domenama trackera (npr. Google, Facebook, TikTok) i pravilima koja ih svrstavaju u kategorije poput *analytics*, *advertising*, *social* i *telemetry*.

Rezultati klasifikacije spremaju se u bazu podataka, čime se podaci pripremaju za daljnju analizu, bodovanje web-stranica i vizualizaciju u grafovima.

Detekcija i klasifikacija trackera

```python
import sqlite3

conn = sqlite3.connect("tracking.db")
cur = conn.cursor()

cur.execute("""
UPDATE requests
SET is_third_party = 1
WHERE visited_site = ''
AND (
    request_domain LIKE '%microsoft.com%'
    OR request_domain LIKE '%facebook.com%'
    OR request_domain LIKE '%tiktok.com%'
)
""")

conn.commit()
conn.close()

print("Third-party oznake ispravljene za telemetry trackere.")
```

fix_third_party.py

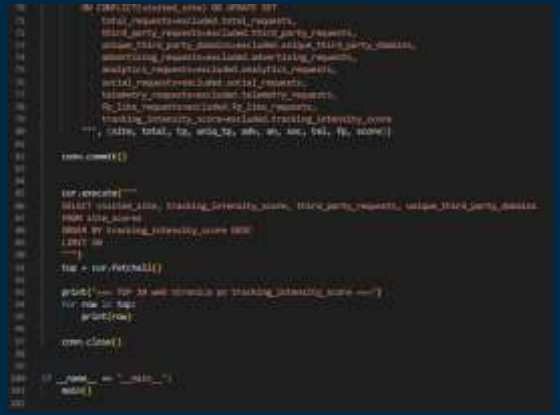
Označava zahtjeve prema **third-party domenama**.

Ispravlja klasifikaciju za poznate **telemetry i tracking servise**.

Priprema podatke za **daljnju analizu i vizualizaciju**.

Procjena intenziteta praćenja web-stranica

score_sites.py



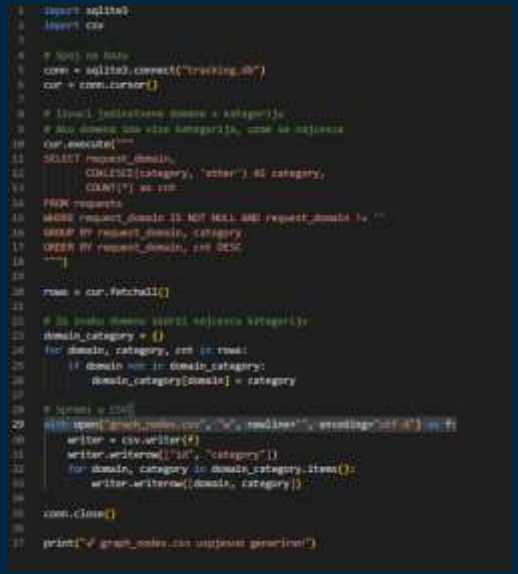
```
PS C:\Users\snaxy\Desktop\sis> python score_sites.py
=== TOP 10 web stranica po tracking_intensity_score ===
('edition.cnn.com', 1989.0, 423, 153)
('www.theguardian.com', 1883.0, 445, 110)
('www.tiktok.com', 1604.0, 400, 23)
('edition.cnn.com', 1989.0, 423, 153)
('www.theguardian.com', 1883.0, 445, 110)
('www.tiktok.com', 1604.0, 400, 23)
('www.jutarnji.hr', 1276.0, 324, 60)
('www.theguardian.com', 1883.0, 445, 110)
('www.tiktok.com', 1604.0, 400, 23)
('www.jutarnji.hr', 1276.0, 324, 60)
('www.jutarnji.hr', 1276.0, 324, 60)
('www.bbc.com', 1271.0, 279, 46)
('www.amazon.de', 1206.0, 307, 8)
('www.bbc.com', 1271.0, 279, 46)
('www.amazon.de', 1206.0, 307, 8)
('search.brave.com', 1128.0, 255, 24)
('search.brave.com', 1128.0, 255, 24)
('www.gloria.hr', 1107.0, 226, 59)
('www.aliexpress.com', 1088.0, 324, 19)
('www.khanacademy.org', 1022.0,_244, 9)
```

Računa tracking intensity score za svaku web stranicu na temelju broja i vrste trackera.

U obzir uzima third-party zahtjeve, broj jedinstvenih tracker domena i kategorije (advertising, analytics, social, telemetry).

Omogućuje rangiranje web stranica prema razini praćenja (najviše → najmanje).

Prikaz podataka



export_nodes.py

```python
import sqlite3
import csv

conn = sqlite3.connect("tracking.db")
cur = conn.cursor()

cur.execute("""
SELECT visited_site, request_domain
FROM requests
WHERE is_third_party = 1
AND visited_site != ''
""")

rows = cur.fetchall()

with open("graph_edges.csv", "w", newline="", encoding="utf-8") as f:
    writer = csv.writer(f)
    writer.writerow(["source", "target"])
    for site, domain in rows:
        writer.writerow([site, domain])

conn.close()

print("graph_edges.csv created")
```

export_graph.py

Prikaz podataka

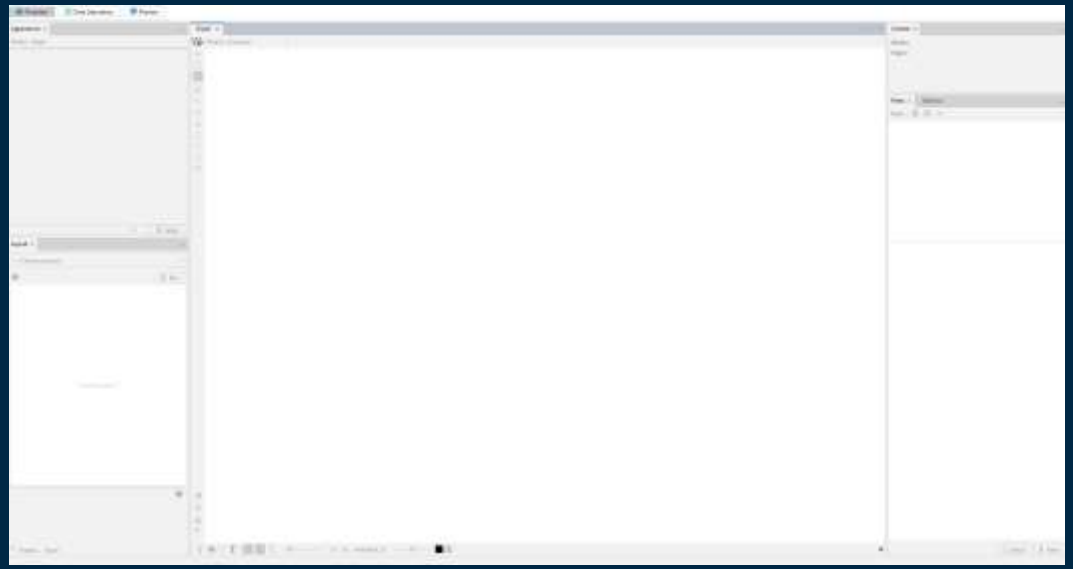
Graph_nodes.csv

Sadrži čvorove grafa – web stranice i tracker domene, zajedno s njihovom kategorijom (analytics, advertising, social, other…).
→ Definira *tko* se pojavljuje u mreži i *što* predstavlja.

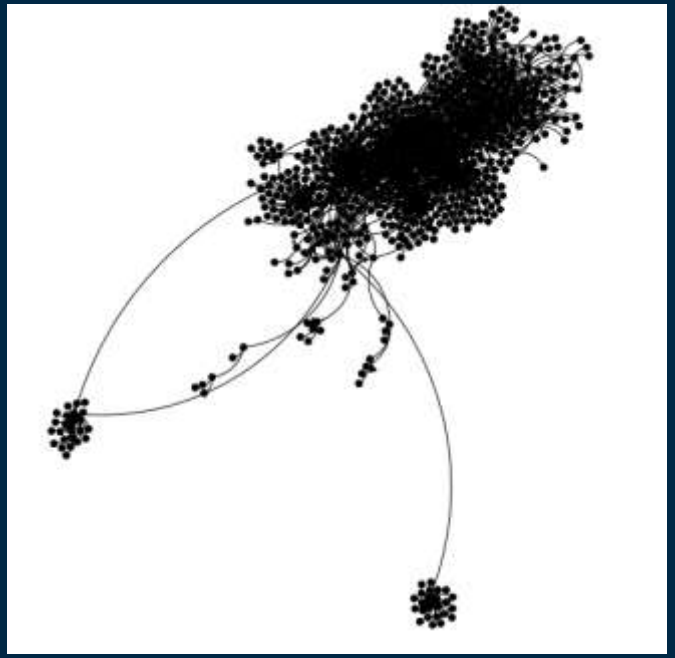
Graph_edges.csv

Sadrži veze između čvorova – pokazuje koja web stranica komunicira s kojim trackerom.
→ Definira *odnose* i tokove praćenja.

Gephi



Vizualizacija mreže trackera (Graph_edges.csv)



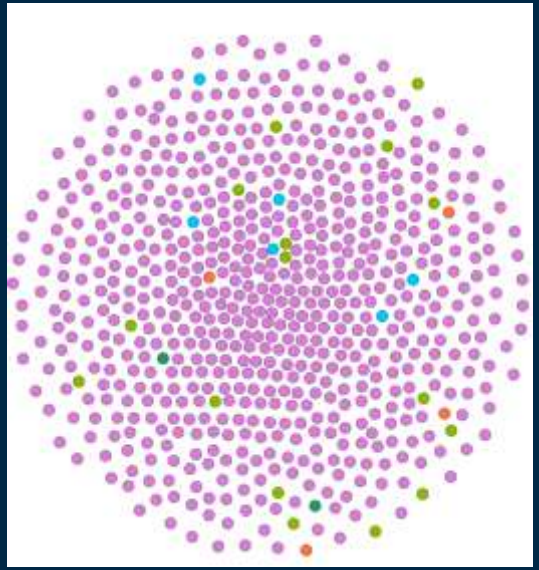
Graph_edges.csv

Graf prikazuje odnose između posjećenih web stranica i third-party tracker domena.

Svaki čvor predstavlja web stranicu ili tracker, a veze (bridovi) označavaju da je tijekom posjeta ostvaren mrežni zahtjev.

Gusti centralni dio pokazuje trackere koji se pojavljuju na velikom broju stranica, dok su izolirani klasteri specifični za pojedine web stranice.

Vizualizacija čvorova mreže (Graph_nodes.csv)



Graph_nodes.csv

Svaki čvor predstavlja web stranicu ili tracker domenu detektiranu tijekom analize.

Boje označavaju kategorije trackera (analytics, advertising, social, telemetry, other).

Gušći centar pokazuje najčešće korištene i najpovezanije trackere koji se pojavljuju na velikom broju stranica.

Vizualizacija omogućuje brzu identifikaciju dominantnih trackera i njihove uloge u ekosustavu praćenja.

Hvala na Pažnji!