Regis University

Combining URL and User-Agent Features for Malicious Web Request Identification

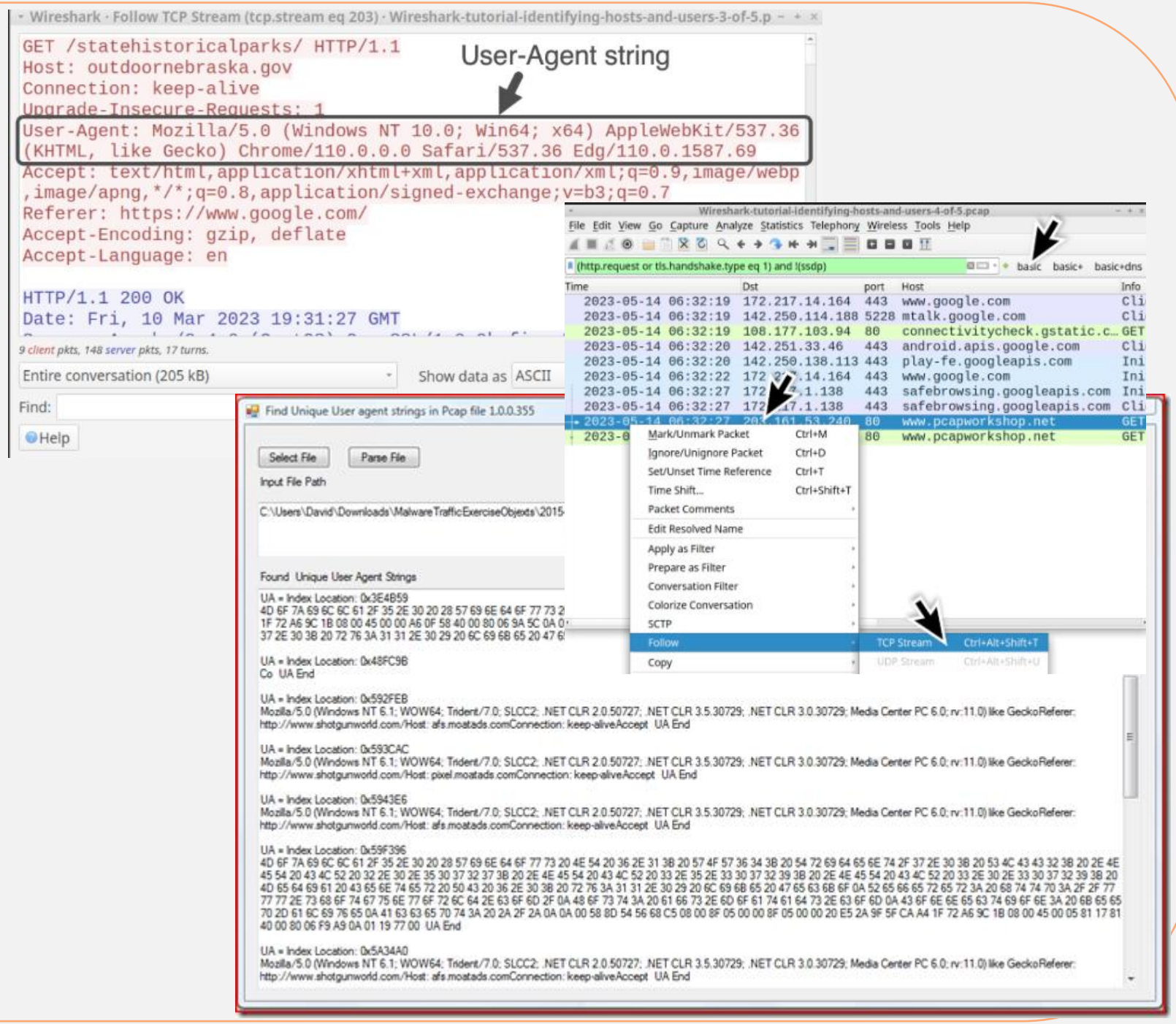
March 5, 2026

Michael Evans

Practicum I

Background

- Introduction
- The Problem
- Data
- Analysis
- Modeling
- Conclusion

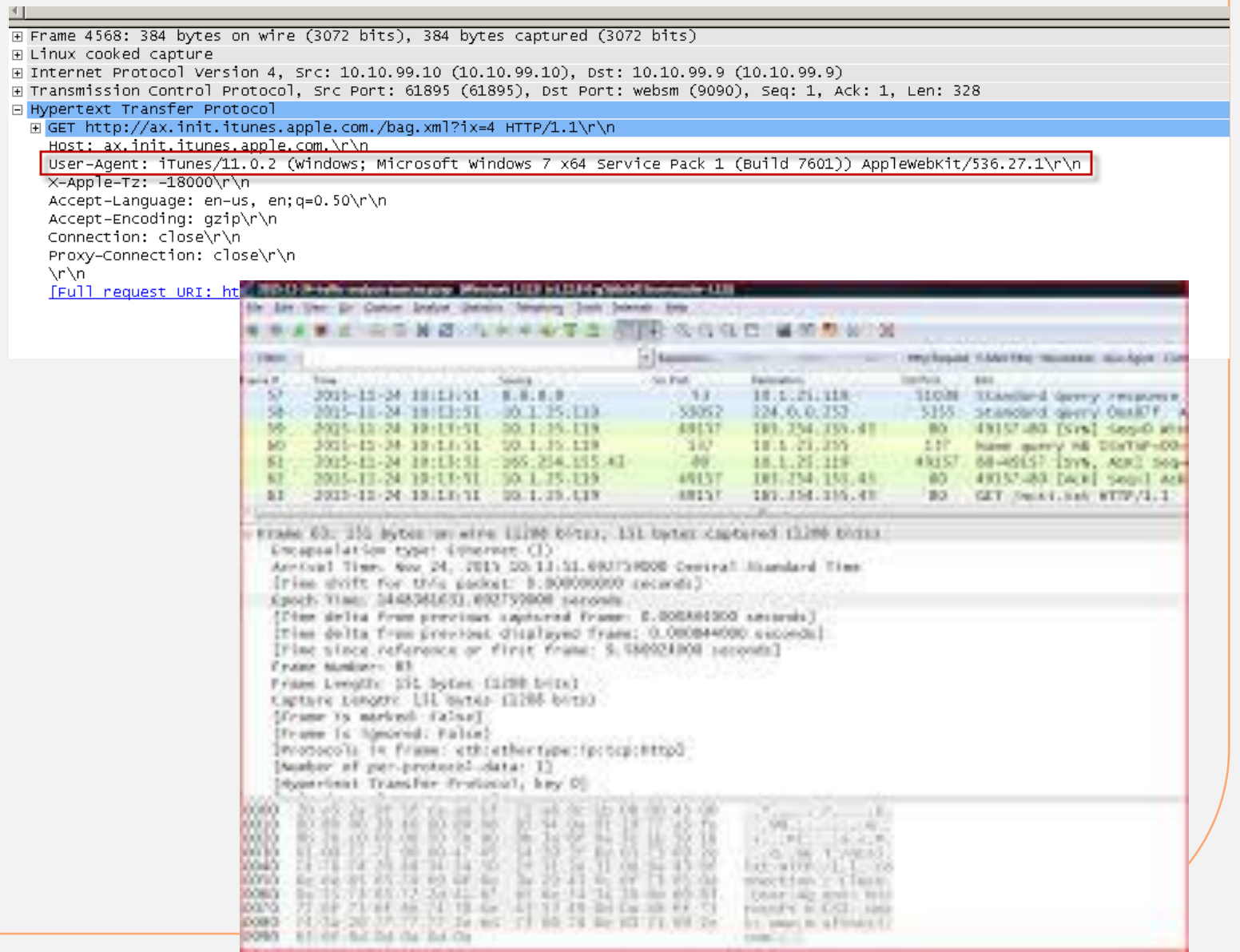


Introduction

Cybersecurity threats targeting web applications have grown exponentially, with attackers exploiting HTTP requests through phishing URLs and spoofed User-Agent strings.

Traditional detection systems often rely on single-feature analysis, limiting their ability to identify sophisticated attacks.

Using machine learning techniques, this project aims to improve detection accuracy and provide actionable insights for web security.



The Problem

Cyberattacks targeting web applications often exploit HTTP requests through phishing URLs and spoofed User-Agent strings.

Traditional detection systems rely on single-feature analysis, limiting their ability to identify sophisticated attacks.

This practicum proposes a hybrid approach that combines lexical features from URLs with behavioral indicators from User-Agent strings to improve malicious request detection using machine learning.

Request Headers

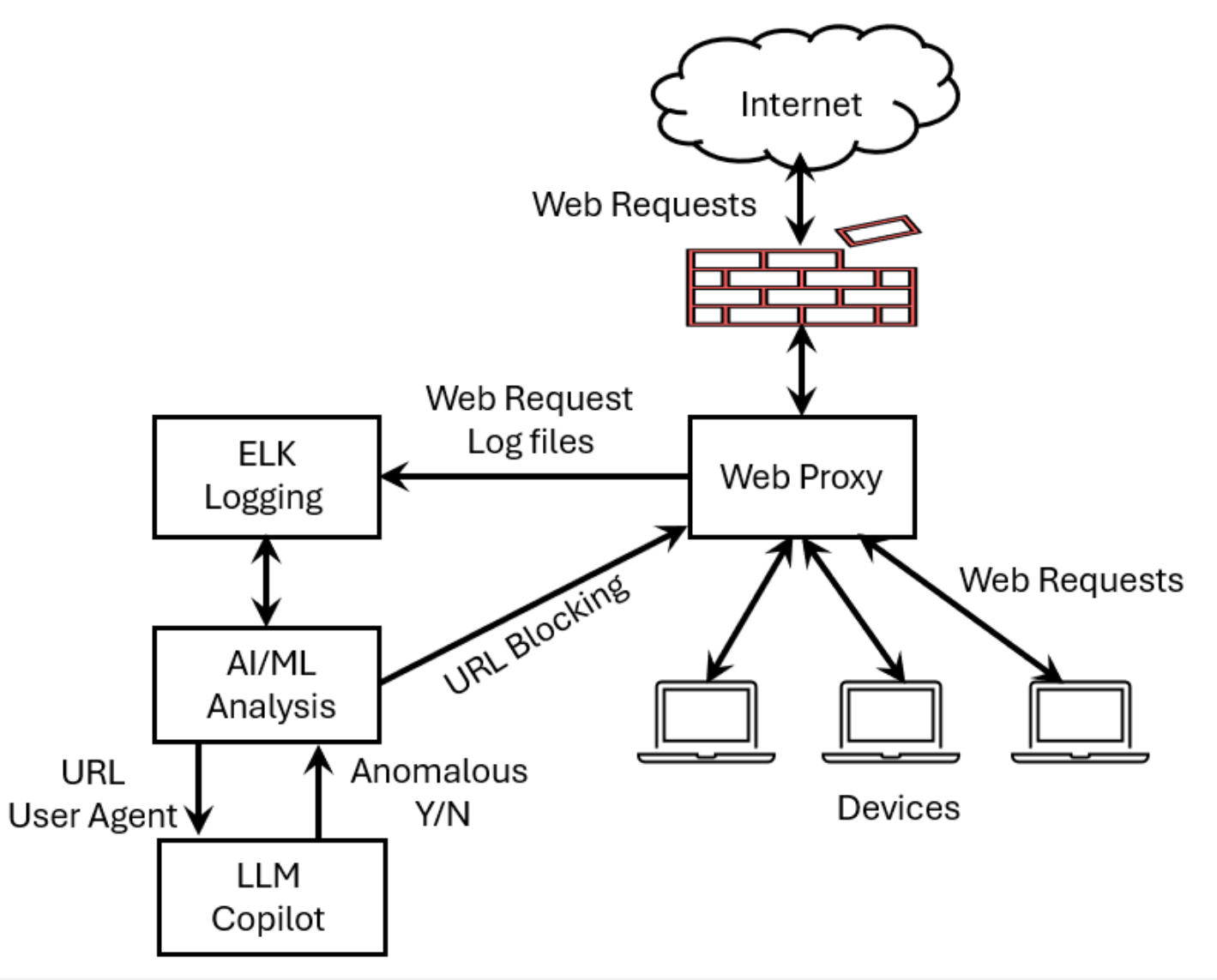
- **Accept:** text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
- **Accept-Encoding:** gzip, deflate, sdch
- **Accept-Language:** en-US,en;q=0.8
- **Cache-Control:** max-age=0
- **Connection:** keep-alive
- **Host:** www.domain.com
- **Upgrade-Insecure-Requests:** 1
- **User-Agent:** Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.80 Safari/537.36

User Agent String

The Data and how we get it

Key Results

- High Correlation (expected)
  - Header_coherence
  - Anomaly_score
- Medium Correlation (expected)
  - Status_code (403,404)
  - Header_coherence
- Minor Correlation (expected)
  - Status_code (403, 404)
  - Anomaly_score



Data Input and Output

Key Results

- High Correlation (expected)
  - Header_coherence
  - Anomaly_score
- Medium Correlation (expected)
  - Status_code (403,404)
  - Header_coherence
- Minor Correlation (expected)
  - Status_code (403, 404)
  - Anomaly_score

Data Input

A. Event identity & routing

- `txid` *(string)*: transaction id
- `timestamp` *(RFC3339 or `%d/%b/%Y:%H:%M:%S %z`)*
- `host` *(string)*
- `client_ip` *(string, optional if sensitive)*
- `server_ip` *(string, optional)*
- `status_code` *(int)*

B. Request line & protocol

- `method` *(string)*
- `uri` *(string, redacted if needed)*
- `http_version` *(string, e.g., `1.1` / `2`)*

C. Headers (summarized, not raw wall-of-text)

- `headers_summary` *(object)* with the minimum you need:
  - `host`
  - `user_agent`
  - `accept`, `accept_language`, `accept_encoding` (presence/values optional)
  - `connection`
  - `content_length` *(float or null)*
  - `transfer_encoding` *(string)*
  - `te_header` *(string, if present)*
- Optionally, include a **short list** of "**present header names**" so LLM can reason about **expected vs present** without full values:
  - `headers_present` *(string[])*

LLM

Data Output

Behavior/category signals (from the pipeline)

- `categories` *(string[], split from your semicolon list)* — e.g., `["protocol_violation","restricted_file"]`
- `anomaly_score` *(float/int)*
- `flags` *(string[])* — e.g., `["cl_and_te","te_non_trailers"]`
- `ua_family` *(string)* — e.g., `chrome` / `safari` / `clientlib` / `bot` / `other`
- `protocol_mismatch` *(bool)*
- `exp_header_presence` *(float, 0..1)*
- `contradictions_cnt` *(int)*
- `header_coherence` *(float)*
- `get_with_body` *(bool)*
- `head_with_ctype` *(bool)*
- `suspicious_path` *(bool)*

== UA/Behavior Quantification ==	
Total Events	5,84
ua_browser_like_events:	3,86
ua_browser_like_pct:	66.18
ua_anomalous_events:	1,35
ua_anomalous_pct:	23.18
behaviour_anomalous_events:	4,17
behaviour_anomalous_pct:	71.44

Non-Anomalous User-Agent Strings

Examples of *non-anomalous* User-Agent strings in flagged traffic

From the dataset (and typical CRS-flagged traffic), these UAs are **syntactically and semantically normal**:

Legitimate-looking desktop browsers

- `Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36`
- `Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/15.1 Safari/605.1.15`
- `Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0`

Legitimate-looking mobile browsers

- `Mozilla/5.0 (iPhone; CPU iPhone OS 13_2_3 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/13.0.3 Mobile/15E148 Safari/604.1`
- `Mozilla/5.0 (Linux; Android 10; SM-G973F) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Mobile Safari/537.36`

Anomalous User-Agent Strings

Examples of *anomalous* User-Agent strings

From the dataset (and typical CRS-flagged traffic), these UAs are **NOT syntactically and semantically normal**:

NOT Legitimate-looking desktop browsers

Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36 **– late 2021, Chrome EOL 2021**

Mozilla/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/60.0.3112.107 Moblie Safari/537.36 **– Chrome/60 - 2017**

Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.89 Vivaldi/1.0.94.2 Safari/537.36 – **policy – Vivaldi 2015 version – Chrome 2015 version – Windows NT 6.1 –also old**

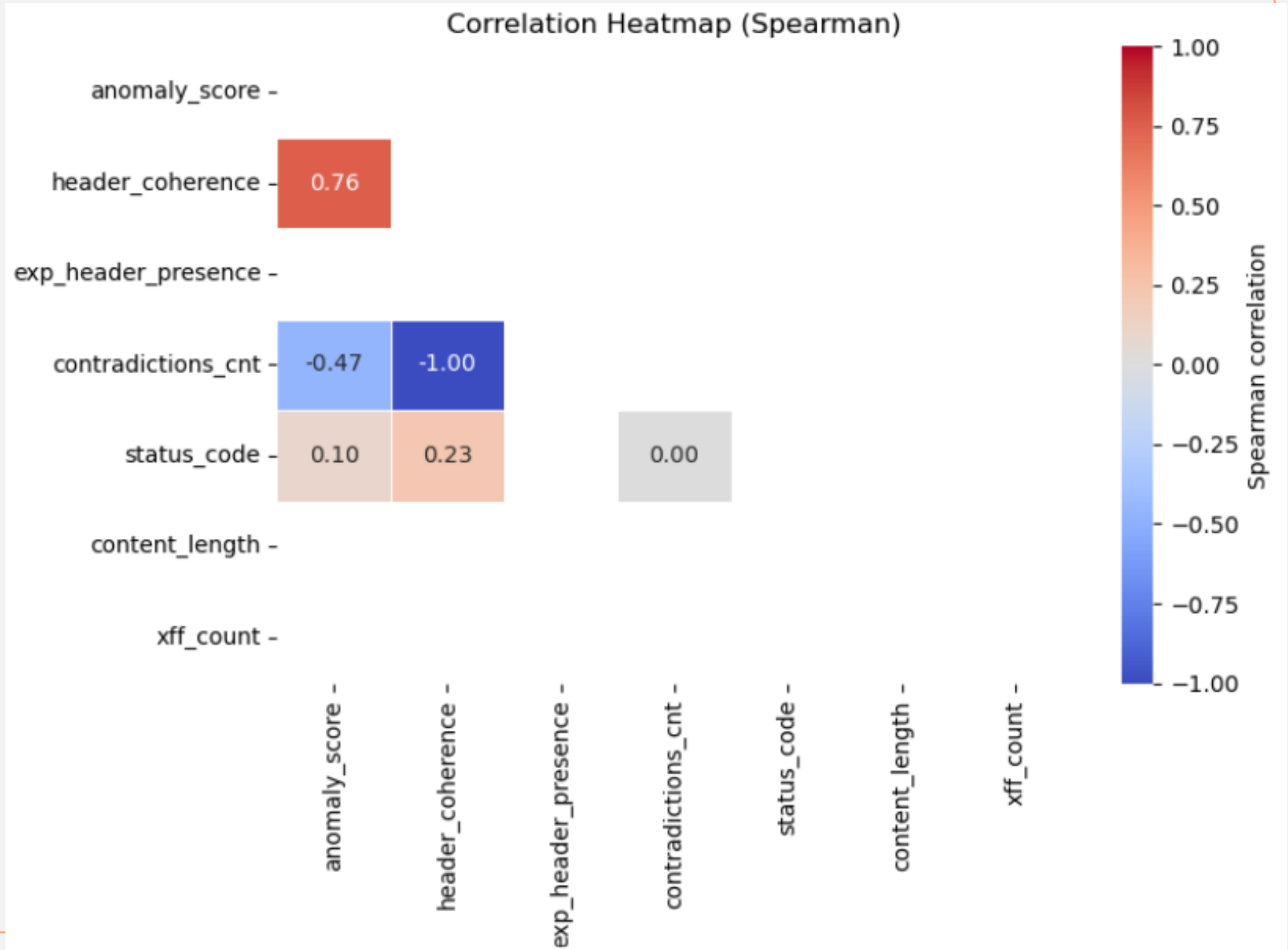
No User-Agent String going to this URL mail.d0466e7.hu/d_3b1ec613/d_bd205cb5/gzdecodes.php **-** top probed/exploited paths for PHP

python-httpx/0.28.1 – used in web scraping

Correlation HeatMap (Spearman)

Key Results

- High Correlation (expected)
  - Header_coherence
  - Anomaly_score
- Medium Correlation (expected)
  - Status_code (403,404)
  - Header_coherence
- Minor Correlation (expected)
  - Status_code (403, 404)
  - Anomaly_score



Three mutually exclusive segments

Segment	Description	p	Lift_vs_overall
UA-only	UA anomaly matches (UA contains one of: bot, crawler, python, curl, wget, java, okhttp, node, axios) and there is no suspicious header flag.	0.198864	0.757832
Behavior-only	there is a suspicious header flag and no UA anomaly.	0.408696	1.557462
Both	both UA anomaly and suspicious header flag are present.	1.0	3.810811

p is P(high_risk | segment) = the fraction of rows in that segment that are high-risk.

lift_vs_overall = p / baseline

==========

UA-only: p=0.199 and lift = 0.76 (Lower Risk)

UA anomalies alone (without header/behavioral flags) are less risky than average in your dataset.
Interpretation: **UA strings are easily spoofed** / broad; signal is weak without corroborating behavior.

Behavior-only: p ≈ 0.409 and lift ~1.56 (High Risk)

Behavioral flags alone (no UA anomaly) are substantially risk-enriching vs baseline.
Interpretation: on-wire behavior (header contradictions, protocol misuse, etc.) discriminates risk better than identity claims.

Both: p = 1.0 and lift ~3.81 (Higher Risk)
Interpretation: All events are high-risk in this sample.

Score distributions

Key Results

Positive means “above average (more concerning).”

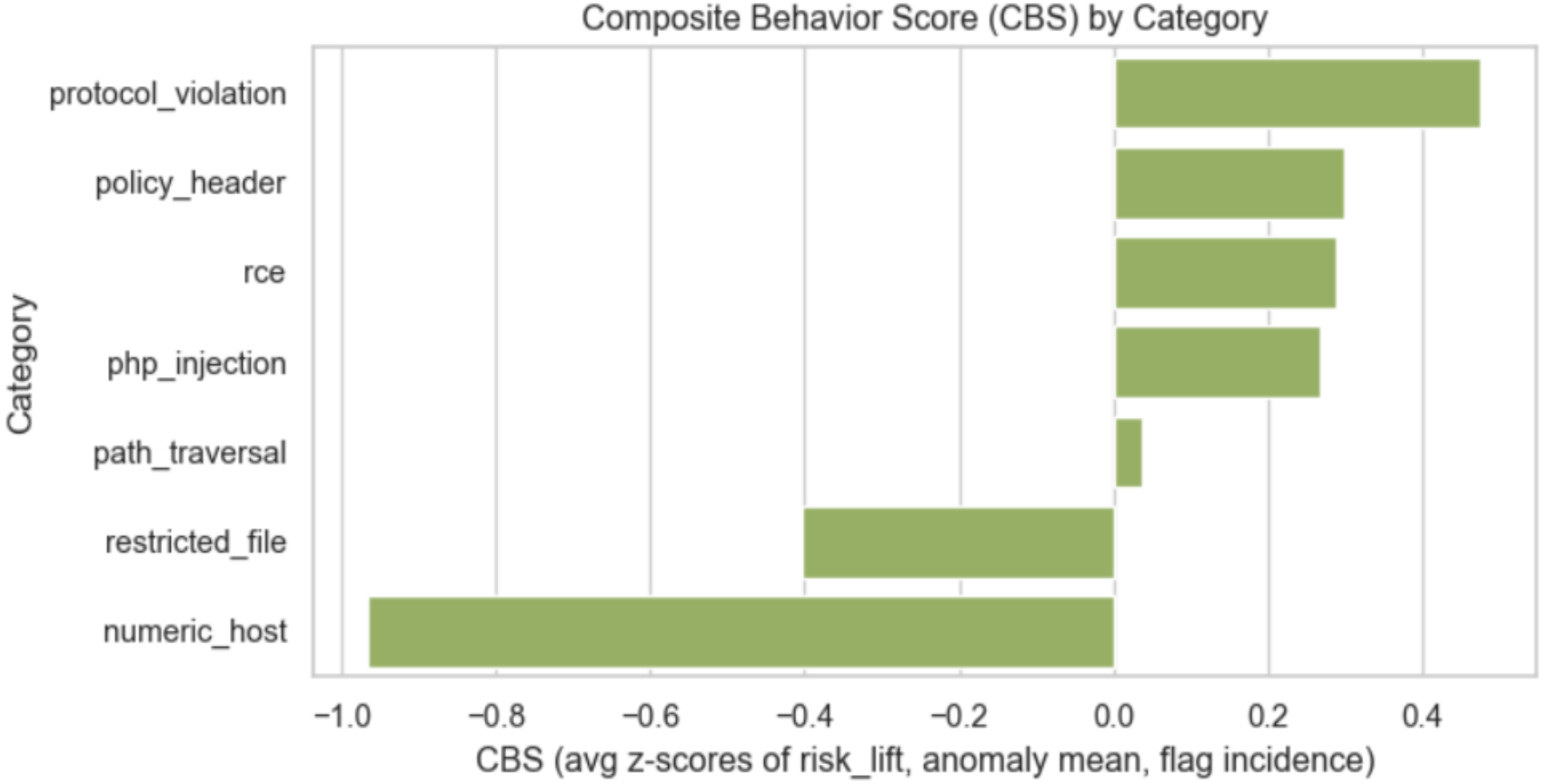
- Protocol violation
- Policy_header
- Rce
- php-injection

0.0 means “right at the overall average across categories in this run.”

- Protocol violation

Negative means “below average (less concerning).

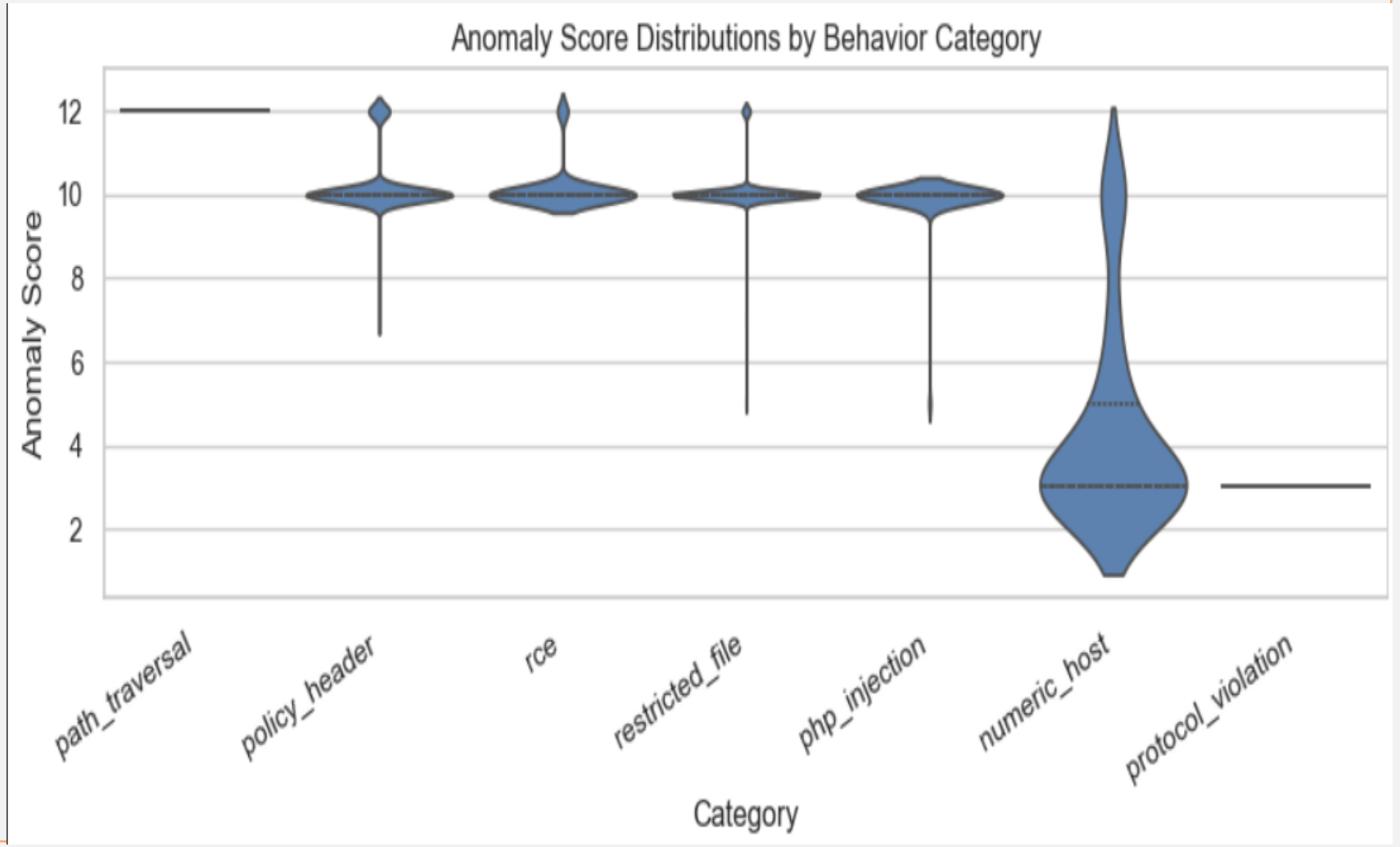
- Restricted_file (config files)
- Numeric_host (IP address)



Score distributions

Key Results

- Path_transversal – always anomalous
- Policy_header, rce, restricted file, php-injection - high
- Protocol Violation, policy_header, rce, php-injection
  - higher z-score – more anomalous

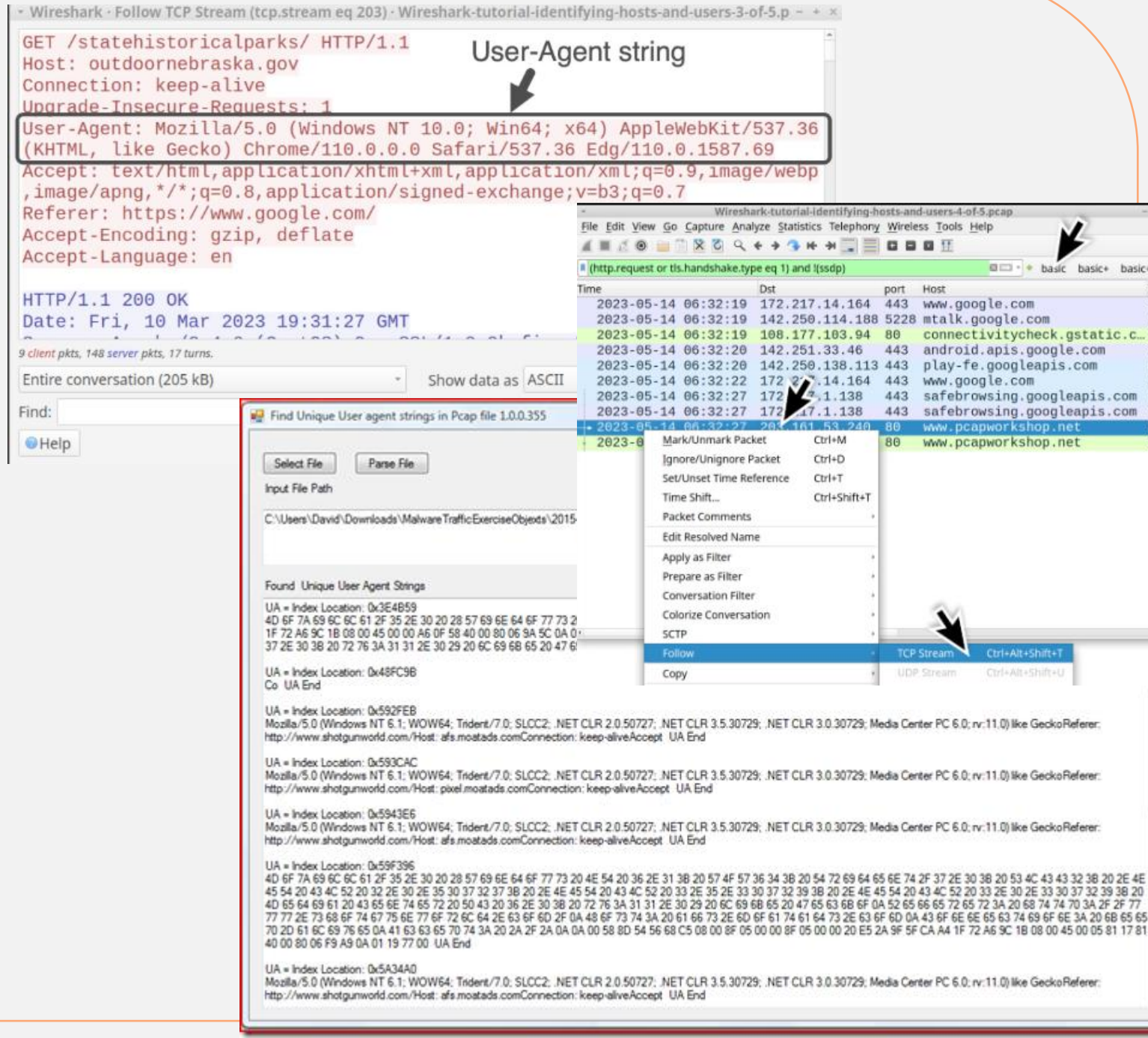


Conclusion

There is a strong anomalous correlation between URL and User-Agent Strings of near 100%. These behavior based Cyberattacks targeting web applications often exploit HTTP requests through phishing URLs and spoofed User-Agent strings.

Scoring just User-Agent strings for anomalous activity is near 40%. It creates more FPs, but if these events can be correlated with another field like URL the TP percentage almost doubles.

This hybrid approach that combines lexical features from URLs with behavioral indicators from User-Agent strings to improve malicious request detection using machine learning.



What's next –future changes

Near live updates

- Script batches of new User-Agents and URLs on a cadence (ie. 20 mins)
- Work at reducing cadence as experience database grows

Efficiencies

- Keep a list of User_Agents and URLs with their respective anomalous status

Expert Feedback

- Events that go to InfoSec experts for manual review – update status in Database