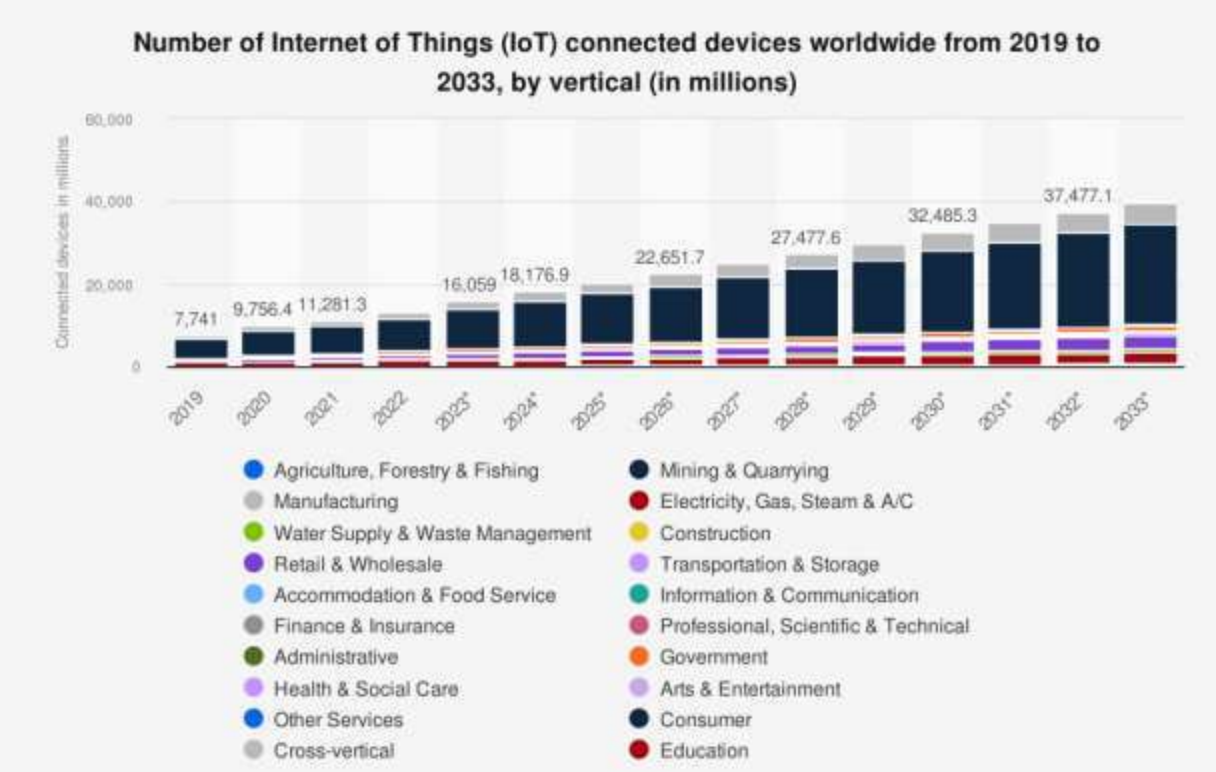
TwinGuard: An Adaptive Digital Twin for Real-Time HTTP(S) Intrusion Detection and Threat Intelligence

Yuanyuan Zhou, Anna Maria Mandalari

Ryu Kuki, Takayuki Sasaki, Katsunari Yoshioka

Motivation

- Modern IoT Challenges Demand New Defences



IoT devices are **widely deployed** across critical infrastructure domains

↓

Traditional IDS struggle with **evolving, obfuscated threats**

↓

Resource constraints on IoT and edge devices limit the feasibility of heavy-weight security solutions

↓

Limited labelled data in real world settings makes **supervised detection** difficult

↓

Real-time, adaptive, and explainable intrusion detection is urgently needed

Figure source: Transforma Insights. “Number of Internet of Things (IoT) Connected Devices Worldwide from 2019 to 2033, by Vertical (in Millions) .” *Statista*, Statista Inc., 10 May 2024, https://www.statista.com/statistics/1194682/iot-connected-devices-vertically/

Introduction

Digital Twin Framework

- mirrors real attacker behaviour: captured by honeypots
- using a virtual model that learns and adapts over time

Core Mechanisms

structured sequence modelling + ML classification + semantic profiling

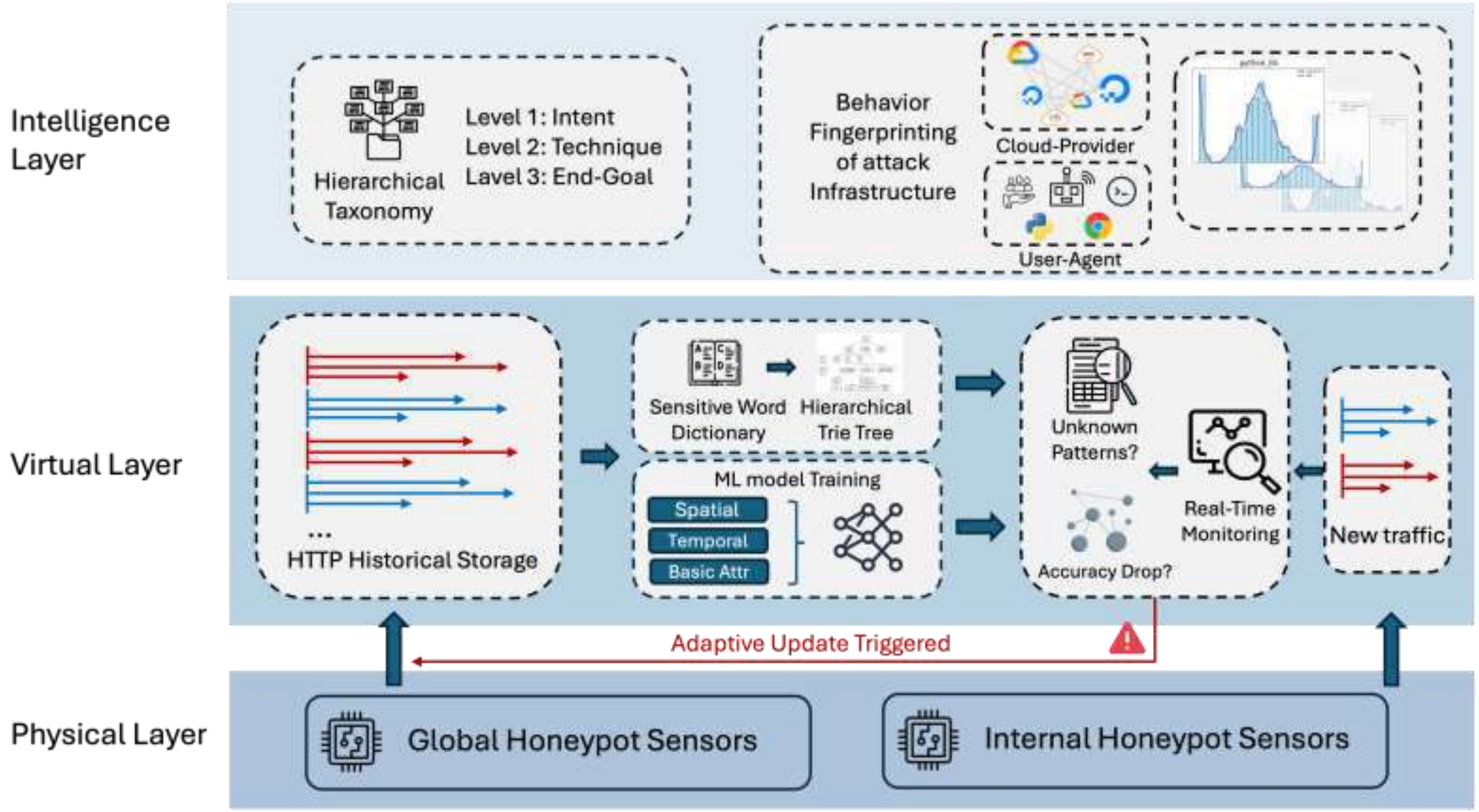
TwinGuard Properties

Modular

Lightweight

Extensible

TwinGuard Design



- Hierarchical Labelling
- Attacker Fingerprinting
- Reveals what, where, and how threats evolve

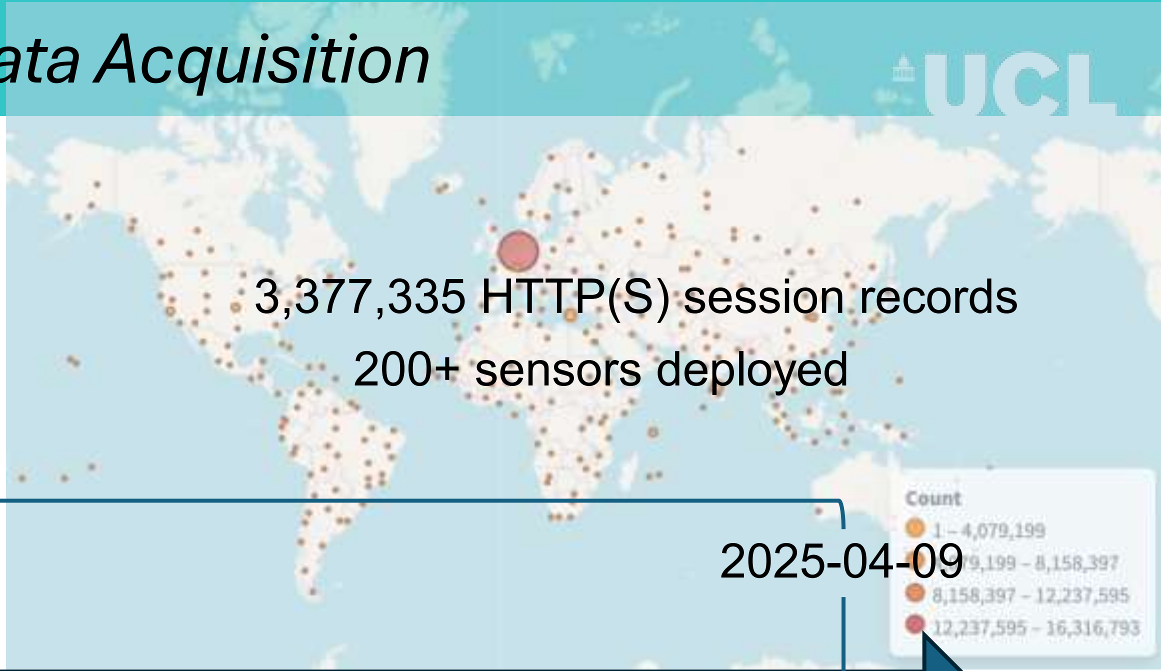
- Trie-Based Path Model matching
- Keyword dictionary for Granularity Reduction
- ML Classifiers for IDS
- Sliding-Window retraining Mechanism

- Capture Real-world HTTP(S) attacks

Physical Layer – *Honeypot Networks and Data Acquisition*

GLOBAL CYBER ALLIANCE

Primary Honeypot Network
ProxyPot



3,377,335 HTTP(S) session records
200+ sensors deployed

2025-03-15

2025-04-09

2025-03-26

2025-03-31

To test generalization under heterogeneous input ➡

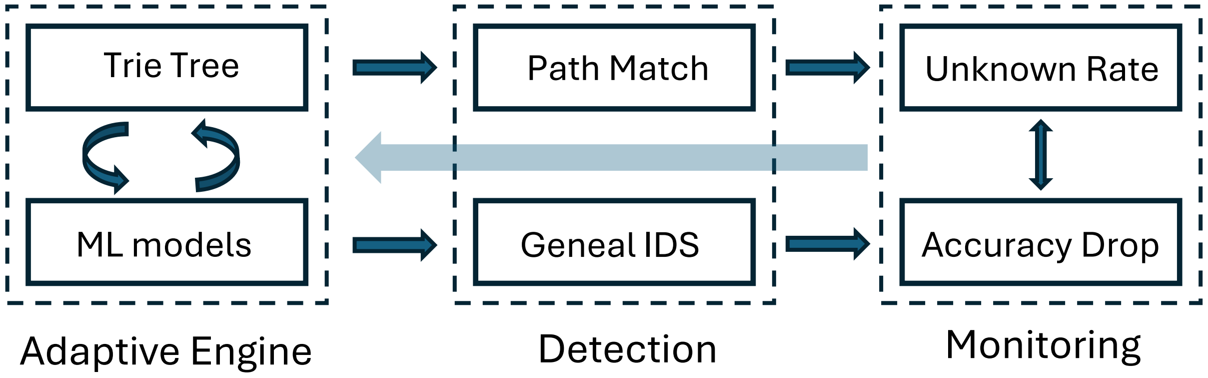
YOKOHAMA National University YNU

Internal Honeypot Network
X-POT

847,869 HTTP requests
19 sensors deployed

70% of fields align with our primary schema

Virtual Layer – *Real-Time Monitoring and Adaptive Detection*



Classification:

Scan | Attempt | Intrusion-Control

Stable Periods:

- both classifiers drops by less than ***6.0%***
- the unknown pattern rate under ***3.0%***

Trie Monitoring

interpretable view of structured request paths by aggregating common behaviour patterns

Machine learning classifiers

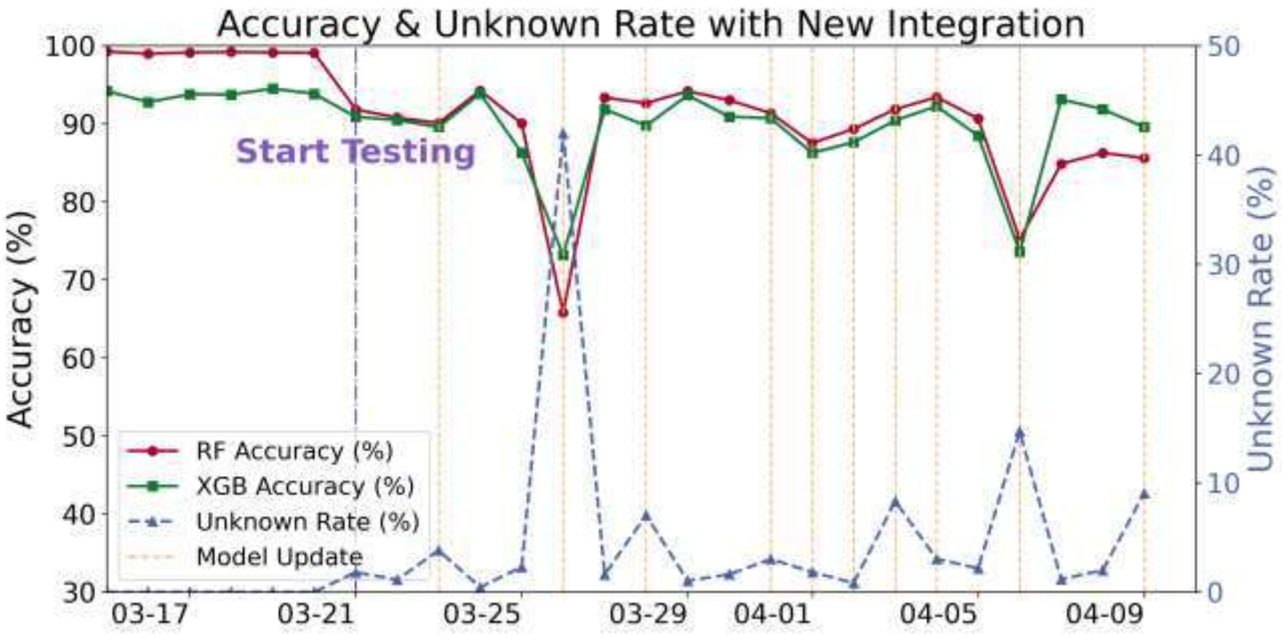
general-purpose intrusion detection component

Sliding Window Mechanism

continuously monitors performance degradation and structural novelty within the HTTP(S) traffic stream

Virtual Layer – *Real-Time Monitoring and Adaptive Detection*

Adaptive ability with the integration of X-POT



Adaptation to a new honeypot (X-Pot) source under window size w = 6.

A surge in unknown sequences and an accuracy drop is observed upon integration, followed by recovery after retraining.

Intelligence Layer: *Intrusion Labelling and Attacker Attribution*

Hierarchical Pattern-Based Intrusion Labelling

Intrusion Category	Technique	End Goal
Exploit Attempts	File Inclusion (LFI/RFI)	Code Execution
	Misconfiguration Exploit	Priv. Esc. / Info Leak
	REST/JSON Abuse	Data Leak / Enumeration
	SQL Injection (SQLi)	DB Access / Bypass
	Command Injection	Code Execution
	Denial of Service (DoS)	Resource Exhaustion
Web Shell Upload	Simple Shell Upload	Persistent Access
	Obfuscated Shell Upload	Stealth Backdoor
	Two-Stage Payload	Loader & Dropper
Post-Exploitation Activity	Botnet C2 Callback	Remote Control
	Cronjob Deployment	Persistence
	Spam Mailer Setup	Email Abuse
	Proxy/Relay Deployment	Lateral Movement
Delivery / Downloader	Direct Script Drop	Code Execution
	Drive-by Download / JS	User Exploitation
Obfuscated / Anomalous Behavior	Junk Payload Flood	Resource Exhaustion
	Unknown Pattern	Undiscovered Variant

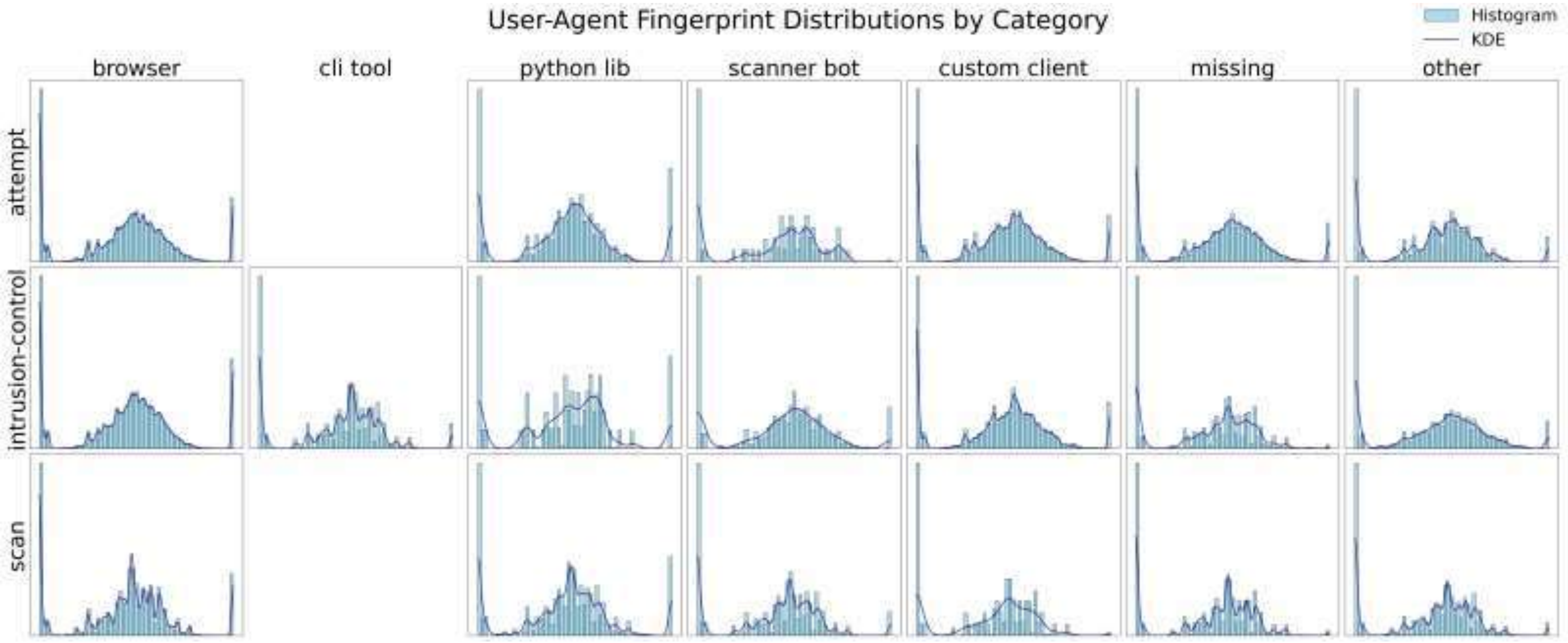
Hierarchical taxonomy structure:

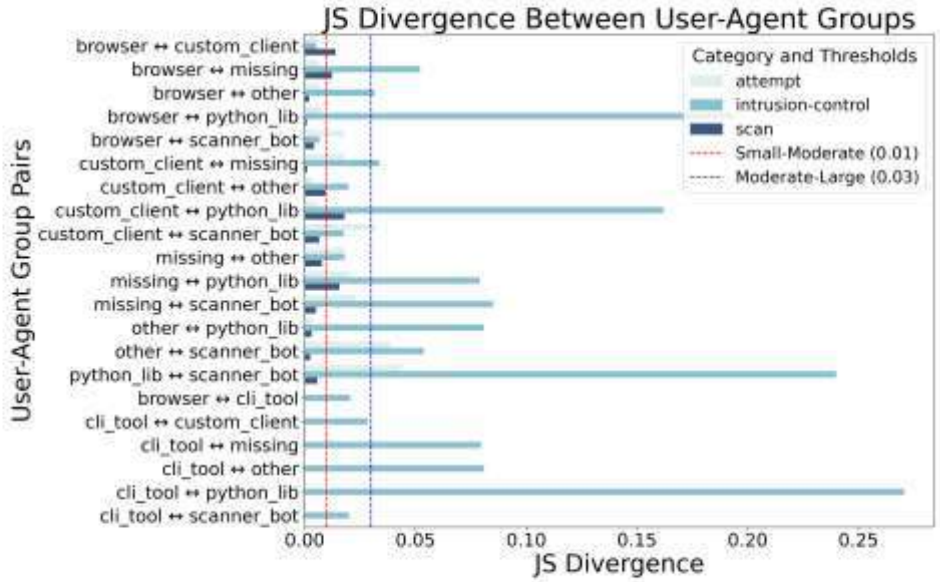
- Level 1: Parent Category (e.g., Exploit, Downloader) *~high-level intent*
- Level 2: Subtypes (e.g., SQLi, Command Injection). *~how it’s done*
- Level 3: End Goals (Execution, Leak, etc.). *~why the attacker is doing it*

Intelligence Layer: *Intrusion Labelling and Attacker Attribution*

Attacker Behavioural Fingerprinting Feature distributions are visualized using histograms and kernel density estimates (KDE)

User-Agent

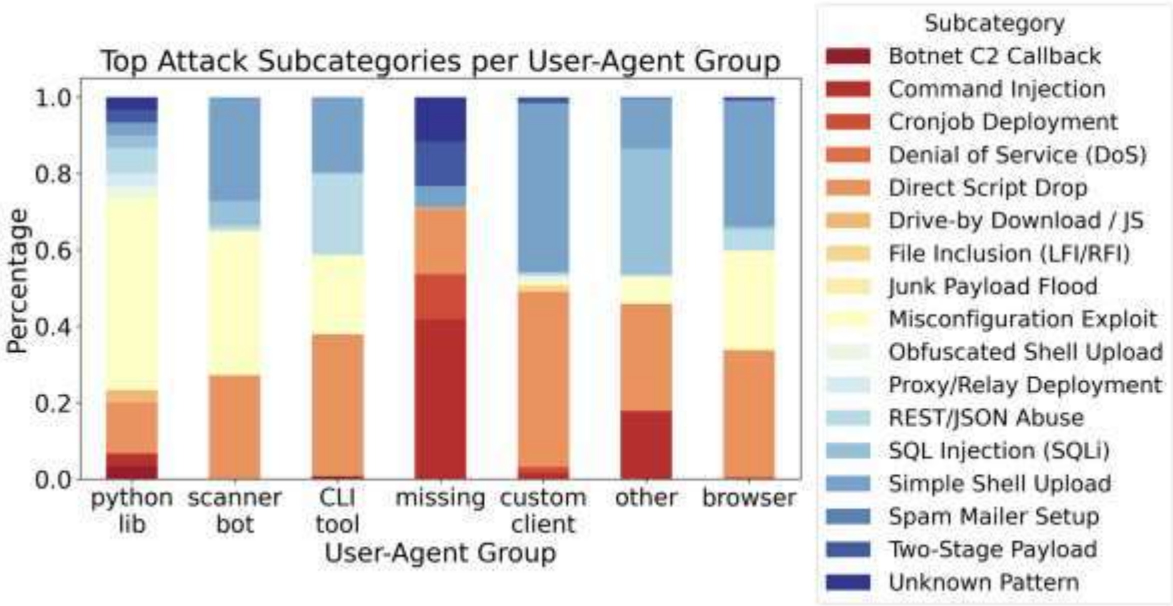


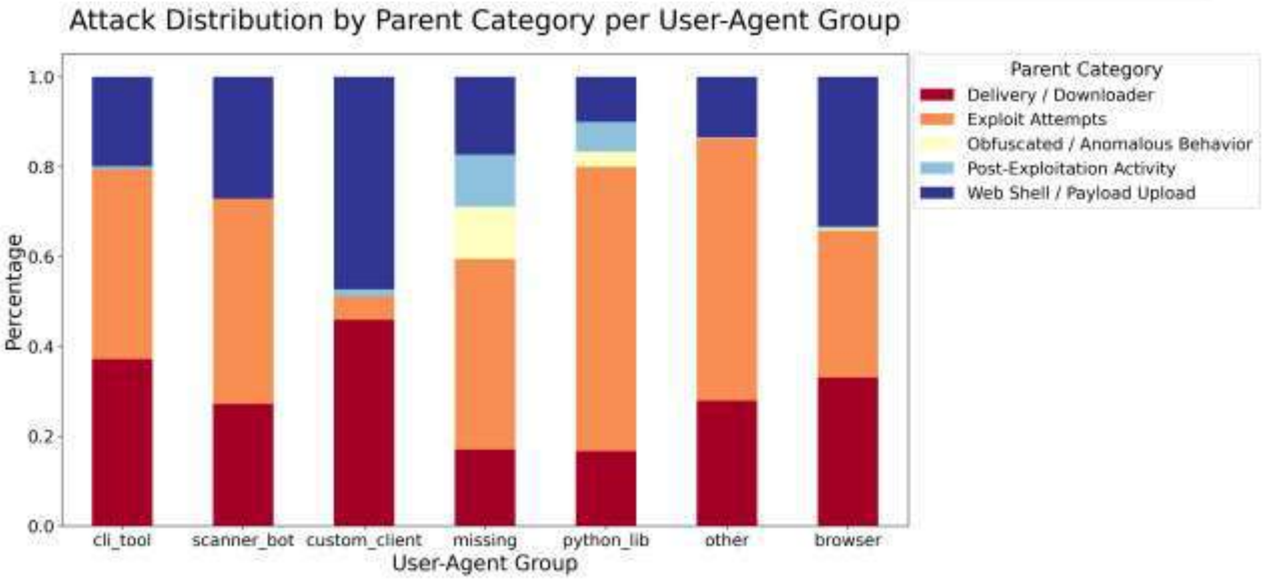


The *x-axis* represents different HTTP session features, and the *y-axis* indicates their normalized values across sessions.

- **Diverse behaviour across UA groups**, especially in intrusion-control.
- **High divergence** observed between *scanner bot*, *python library ,* indicates distinct attack behaviours.

User-Agent





Browser and CLI tool
- traditional probing behaviour.

python libraries **and** ***scanner bots***
- greater technique diversity

The missing and other categories
- spoofed or unstable automation strategies.

- Same analysis apply to “Cloud Providers”, but shows Minor Exploit Variations
- Confirms cloud-based attacks are likely **templated and automated**, regardless of provider.

Conclusion

High Accuracy & Responsiveness

- Maintains **>90% accuracy** during stable periods
- **Dual classifiers** + **sequence monitoring (Trie)** ensure robustness

Adaptive Retraining Triggered by Novelty

- **Strong negative correlation** between unknown rate and accuracy
- **42% spike** in unknowns + **30% accuracy drop** mitigated in **1 update cycle**

Real-World Deployment with Diverse Traffic

- Processes traffic from **heterogeneous honeypot sources**
- Demonstrates **adaptability across environments**

Behavioral Intelligence

- Reveals **diverse attacker behaviour** across user-agent types
- **Cloud-based traffic** shows consistent patterns → shared tooling

Follow us:
https://safenetiot.github.io/
https://www.youtube.com/watch?v=0fg0acuRbUA

Contact:
yuanyuan.zhou.23@ucl.ac.uk

others

Future Work

Quantitative Measurements of Lightweight and Criteria for threshold selection

How the attacks are evolving
Quantify the **behavior drift**

Sequence input simulating live stream, instead of the daily window

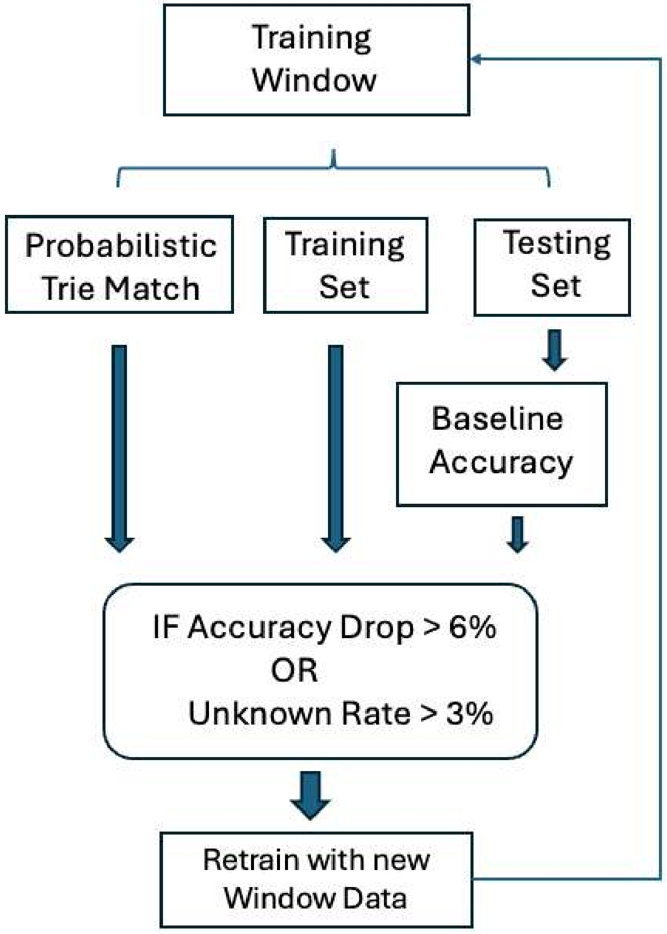
Temporal Bias (26 days) -> 3 months

Real-World Deployment & Evaluation
Transition from honeypot-only testing to real production environments

Virtual Layer – *Real-Time Monitoring and Adaptive Detection*

Sliding Window Mechanism

continuously monitors performance degradation and structural novelty within the HTTP(S) traffic stream



Monitoring module: Adaptive Loop Structure

Classification:

Scan | Attempt | Intrusion-Control

Stable Periods:

- both classifiers drops by less than ***6.0%***
- the unknown pattern rate under ***3.0%***

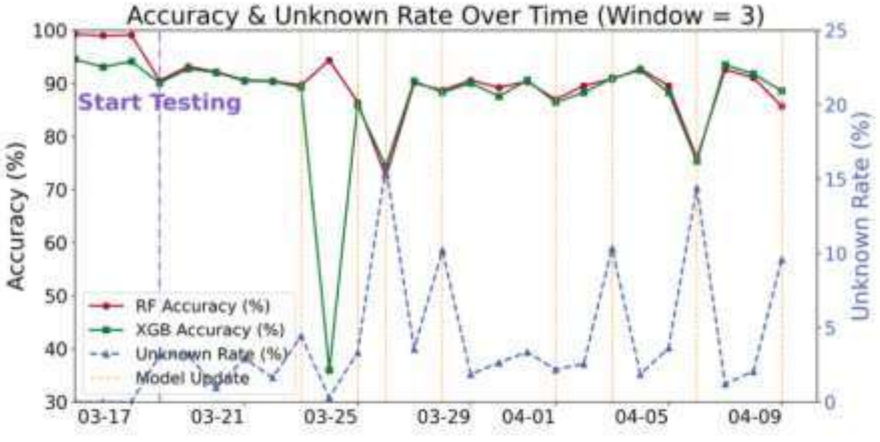
Labeling Criteria:

- Intrusions are labelled using **rule-based matching** of structured request paths, **payload content**, and **endpoint semantics**.

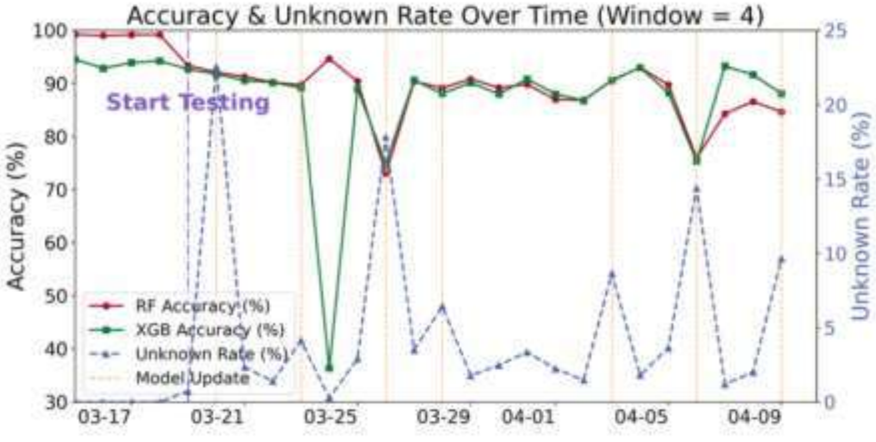
- If a spike in unknown patterns occurs without existing labels, we check if **new labelling is needed** to maintain detection accurate.

Virtual Layer – *Real-Time Monitoring and Adaptive Detection*

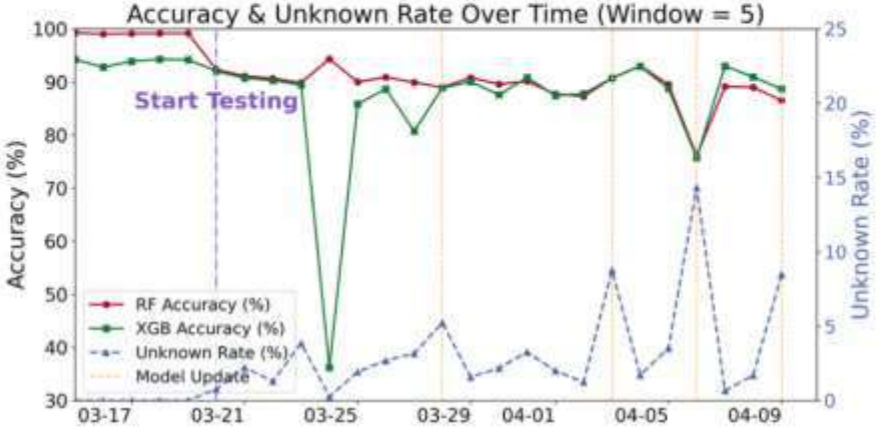
Accuracy and Unknown Rate Dynamics



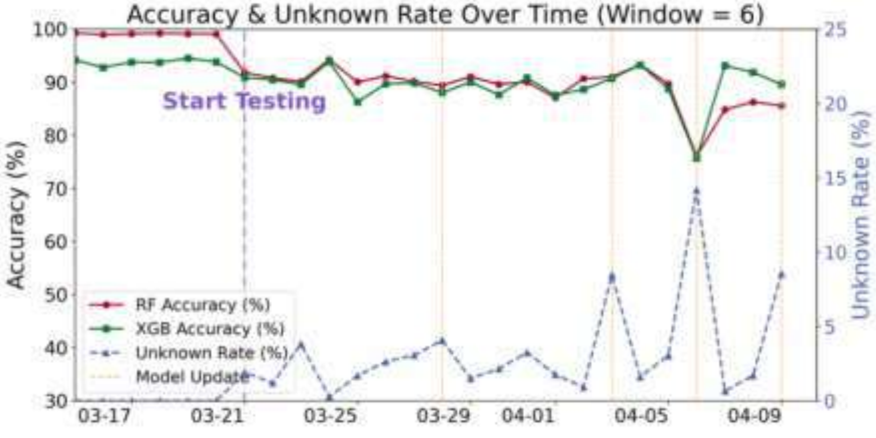
(a) $w = 3$



(b) $w = 4$



(c) $w = 5$



(d) $w = 6$

Smaller Windows

- Fast Reaction
- Frequent Updates
- Higher Volatility

Larger Windows

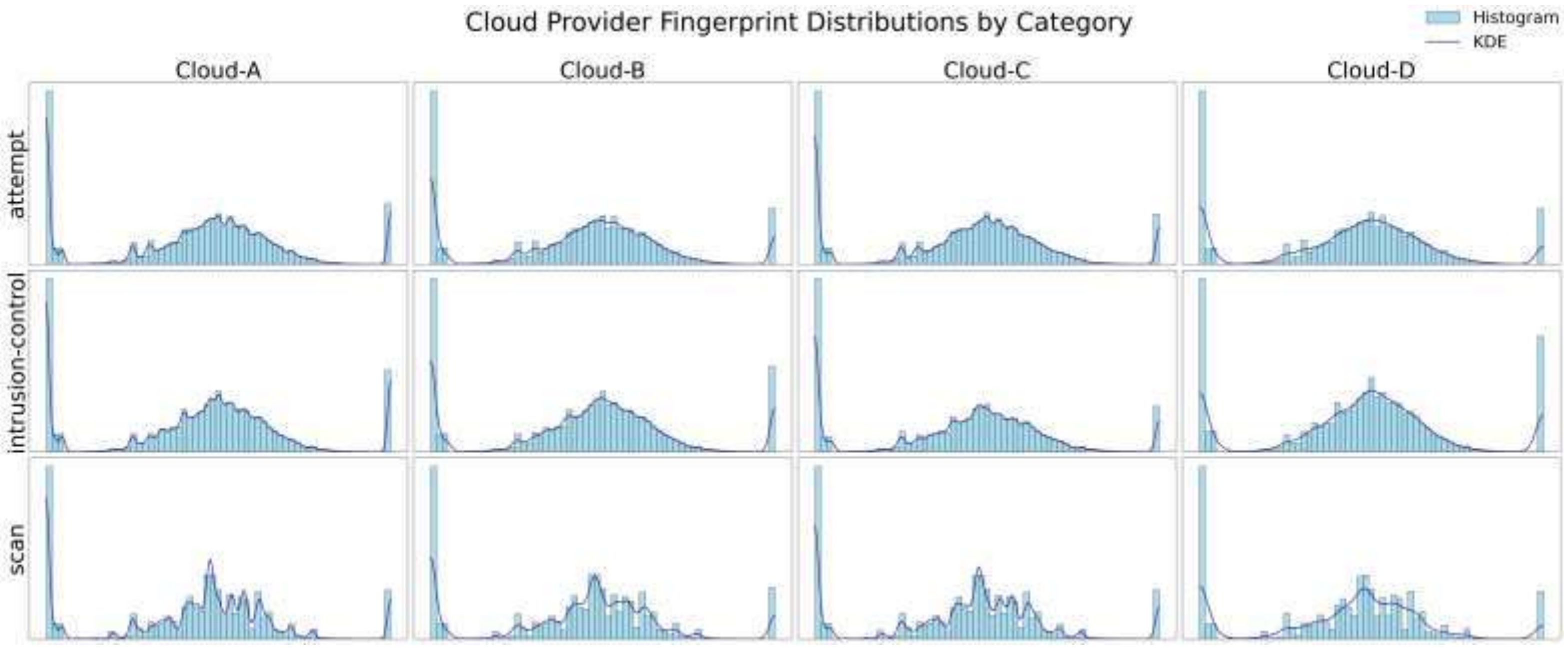
- Stable Accuracy
- Fewer Updates
- Lower Unknown Rate

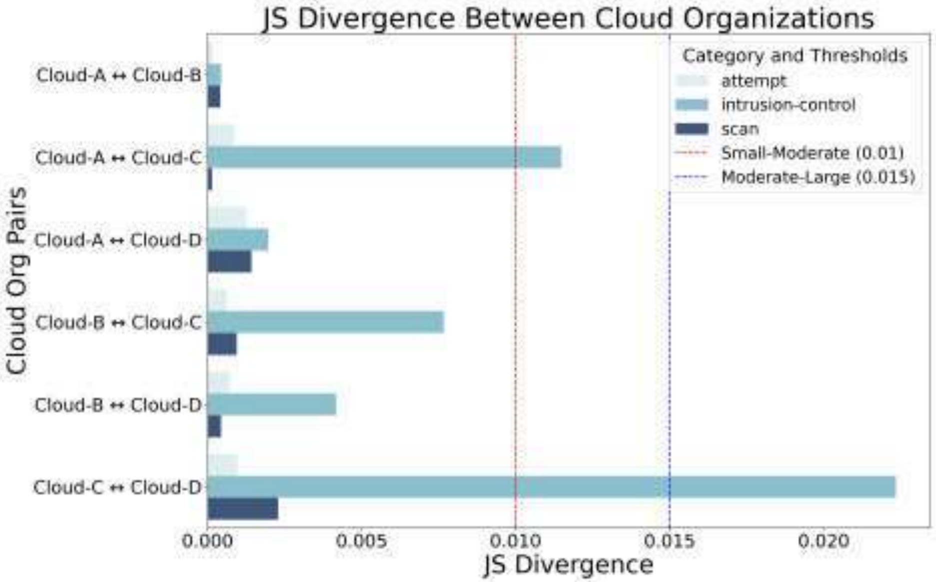
w = 6 strikes a balance between the model utility and stable performance

Intelligence Layer: *Intrusion Labelling and Attacker Attribution*

Attacker Behavioural Fingerprinting

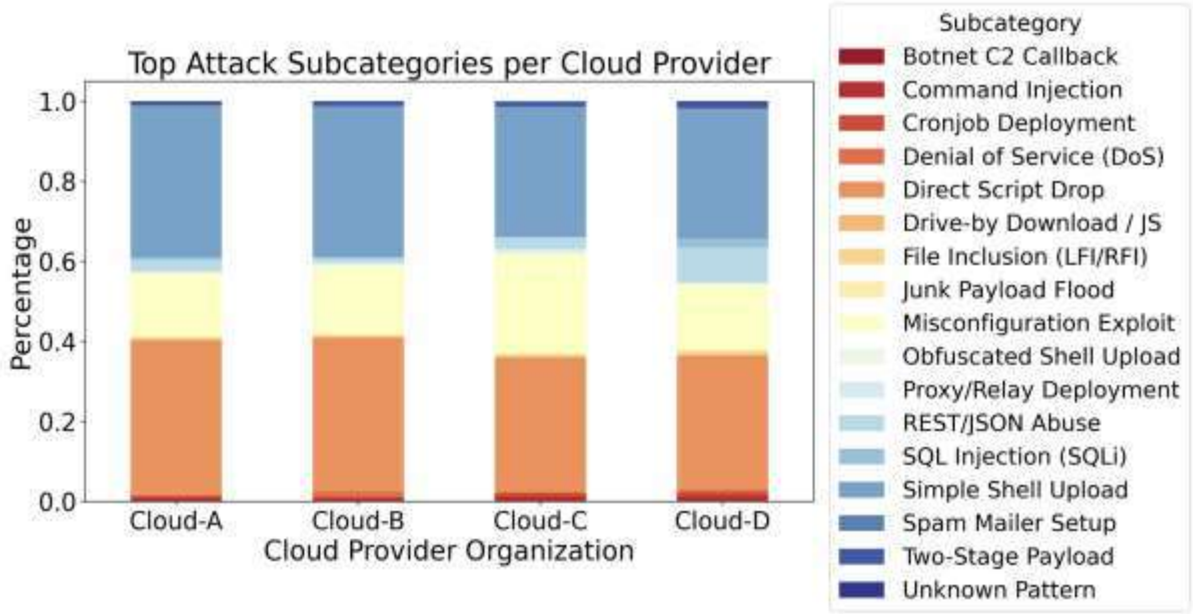
Cloud Provider

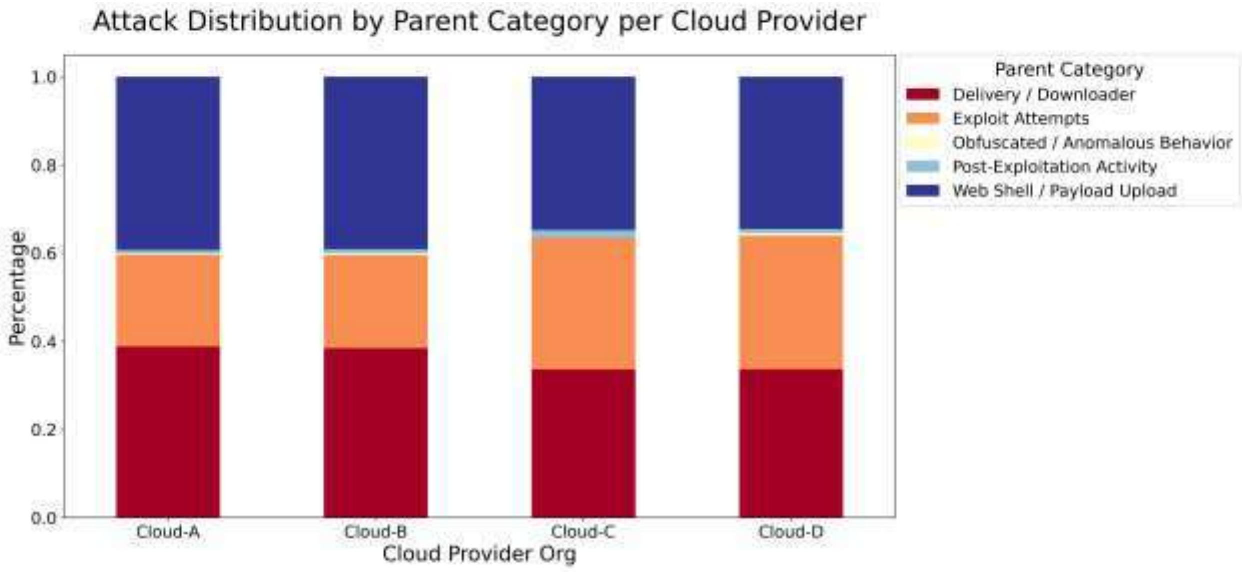




- **Overall low divergence** → attack behaviour is largely consistent across cloud platforms.
- **Cloud C shows slight divergence** in intrusion-control attacks.
- **Impact is minimal** → cloud provider has **limited influence** on attack diversity.

Cloud Provider

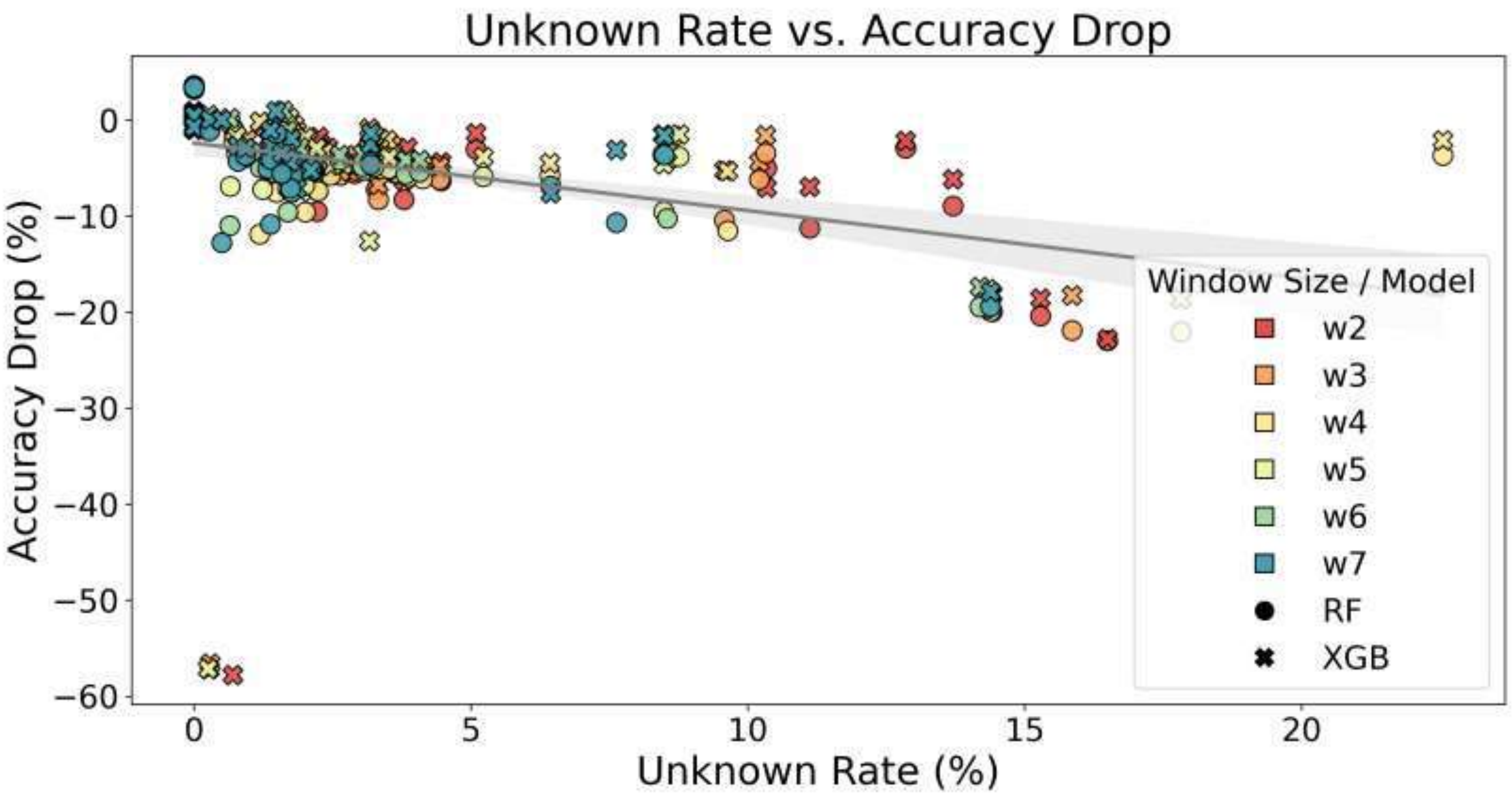


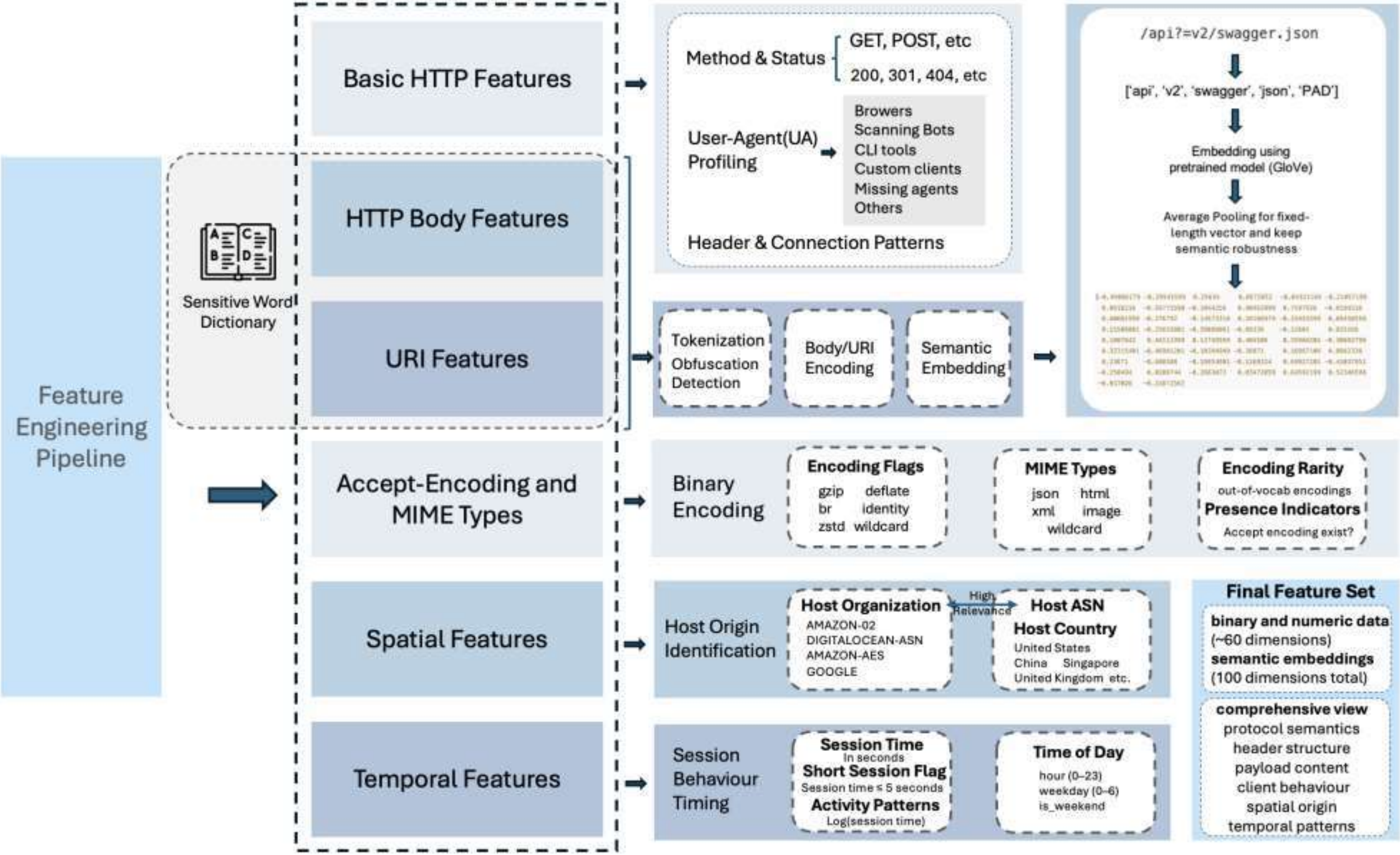


- **Shared Attack Focus**: All cloud providers show similar dominance in script drops & shell uploads, matching low JS divergence.

- **Minor Exploit Variations**: Slight shifts (e.g., more SQLi on Cloud-D, misconfiguration on Cloud-C) don't alter overall behaviour.

- Confirms cloud-based attacks are likely **templated and automated**, regardless of provider.


Unknown Rate vs. Accuracy Drop
Accuracy Drop (%)
0
−10
−20
−30
−40
−50
−60
Unknown Rate (%)
0
5
10
15
20
Window Size / Model
w2
w3
w4
w5
w6
w7
RF
XGB


Feature Engineering Pipeline
Sensitive Word Dictionary
Basic HTTP Features
HTTP Body Features
URI Features
Accept-Encoding and MIME Types
Spatial Features
Temporal Features
Method & Status
GET, POST, etc
200, 301, 404, etc
User-Agent(UA) Profiling
Browers
Scanning Bots
CLI tools
Custom clients
Missing agents
Others
Header & Connection Patterns
Tokenization Obfuscation Detection
Body/URI Encoding
Semantic Embedding
/api?=v2/swagger.json
['api', 'v2', 'swagger', 'json', 'PAD']
Embedding using pretrained model (GloVe)
Average Pooling for fixed-length vector and keep semantic robustness
Binary Encoding
Encoding Flags
gzip deflate
br identity
zstd wildcard
MIME Types
json html
xml image
wildcard
Encoding Rarity
out-of-vocab encodings
Presence Indicators
Accept encoding exist?
Host Origin Identification
Host Organization
AMAZON-02
DIGITALOCEAN-ASN
AMAZON-AES
GOOGLE
High Relevance
Host ASN
Host Country
United States
China Singapore
United Kingdom etc.
Session Behaviour Timing
Session Time
In seconds
Short Session Flag
Session time ≤ 5 seconds
Activity Patterns
Log(session time)
Time of Day
hour (0–23)
weekday (0–6)
is_weekend
Final Feature Set
binary and numeric data
(~60 dimensions)
semantic embeddings
(100 dimensions total)
comprehensive view
protocol semantics
header structure
payload content
client behaviour
spatial origin
temporal patterns

1. Drift Detection Across Sequences

Goal:

Quantify how much new behavior appears over time.

You can do this by:

•Comparing each new sequence to a baseline (e.g., first day/week)

•Measuring:

- **Unknown rate** (e.g., new URI tokens or unseen paths)
- **Distributional change** (e.g., cosine distance between feature means)
- **Jensen-Shannon divergence**, etc.

Outcome:

Plots like:

X-axis: Time (each sequence window) Y-axis: Drift score (distance or novelty)

"We observe that while new sensitive keywords and unique attack sequences continue to appear throughout the monitoring period, the rate of discovery slows over time, and most features become inactive soon after their initial appearance. This long-tailed, bursty dynamic reflects a continually evolving attack landscape, with only a small subset of features repeatedly targeted over multiple days."