DNS Data Exfiltration Prevention: Kernel-Enforced Endpoint Security

Scalable Framework to Disrupt DNS C2 and Tunneling

Vedang Parasnis

University of Washington

Agenda

- Data Exfiltration / Data Breaches Phases
- DNS attack vectors to exfiltrate data
- Drawbacks of current approaches
- Security Framework Overview
- Kernel Enforced Endpoint Security Architecture (Kernel + Userspace)
- Results, Evaluation, Demo
- Protect Linux Kernel from malicious tampered endpoint security eBPF programs
  - Cloud CA + kernel keyring + BPF LSM + Kernel Datapath

Data Exfiltration / Data Breaches

Definition: Unauthorized extraction or transmission of sensitive data from a system

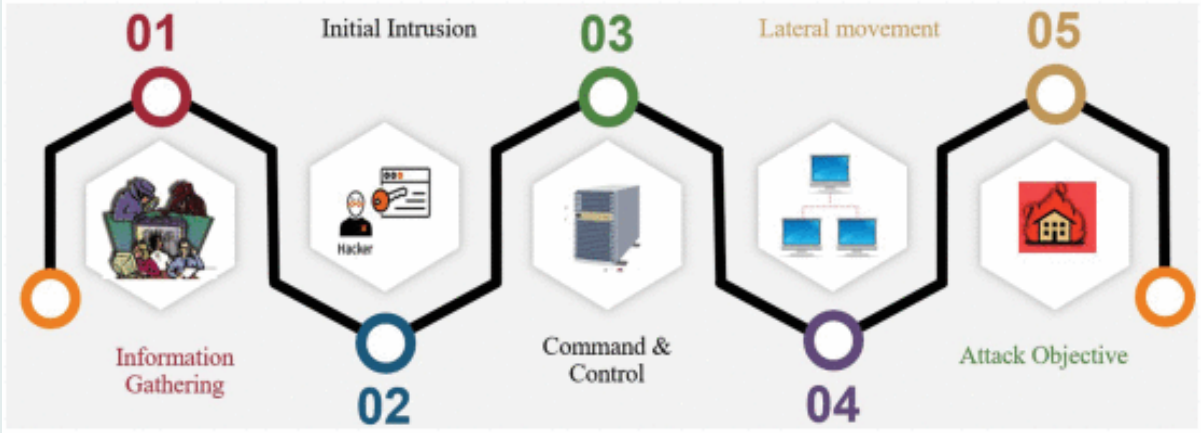
Impact: Reputation, Financial damages

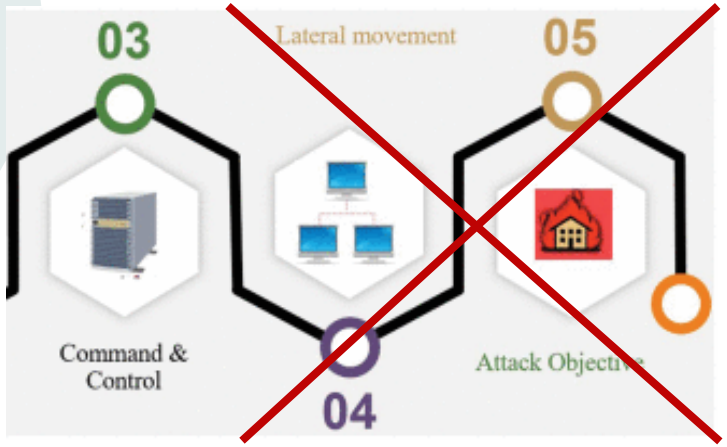
Attack Lifecycle

- Information Reconnaissance
- Initial Intrusion / Infiltration
- **Command and Control**
- Lateral Movement
- Command Execution and Data Breaches

Core Defense Strategy

- Endpoint Security (EDR / XDR)





DNS Data Exfiltration

DNS C2 – Uses DNS queries and responses to maintain covert communication with attacker infrastructure.

DNS Tunneling – Encapsulates arbitrary data within DNS packets to bypass network restrictions.

DNS Raw Exfiltration – Leaks sensitive data files directly in DNS queries.

- **Remote Code Execution (RCE)**
  - Shell code exploits
  - Script executions, File corruptions
  - Process Side channeling exploits
- Example**: Sliver C2, Hexane, APT29 (Cozy Bear), Skitnet.**
- **Persistent Backdoors**
  - Deployment rootkits, ransomwares
  - Example**: Turla** group
- **Network Pivoting (Port Forwarding)**
  - Compromised machines act as proxies to reach deeper into private infrastructure
  - Example**: Cobalt Strike, Hexane, DNSSystem**

Existing Approaches

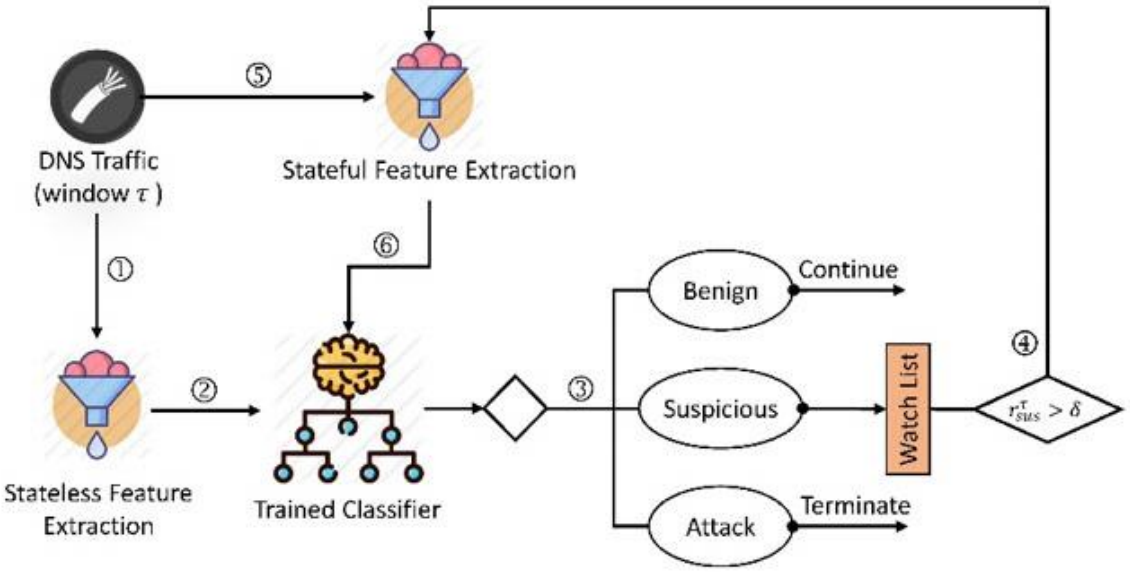
- **Semi-Passive Analysis**
  - DNS Exfiltration Security as Middleware
- **Passive Analysis**
  - Anomaly Detection
  - Threat Signatures, Domain Reputation scoring

Existing Approaches – Passive Analysis

- **Anomaly Detection:**
  - **Traffic Behavior Analysis**
    - DNS Passive Traffic Volume Analysis
    - DNS Passive Traffic timing Statistical Analysis
  - **Machine Learning-based Threat Intelligence**
    - Uses machine learning models to identify traffic anomalies.
- **Threat Signatures:**
  - DNS Domain Scoring
  - Malicious domain signature

Stateless Features – Lexical Analysis

Stateful Features – Statistical Analysis



Issues with current approaches

- **Slow Detection → High Dwell Time → More Damage**
- **Extremely slow to Advanced C2 Attacks**
  - More Damage if C2 infrastructure employs multiplayer mode (Botnet of C2 server exploiting scaled environments)
- **Dynamic Threat Patterns**:
  - Varying Throughput
  - Slow and Stealthy Rate
  - Kernel Encapsulated Traffic
  - Port Obfuscation
- **Centralized monitoring and analysis systems don't scale**
- **Ineffective over IP Masquerading & Domain Generation Algorithms**

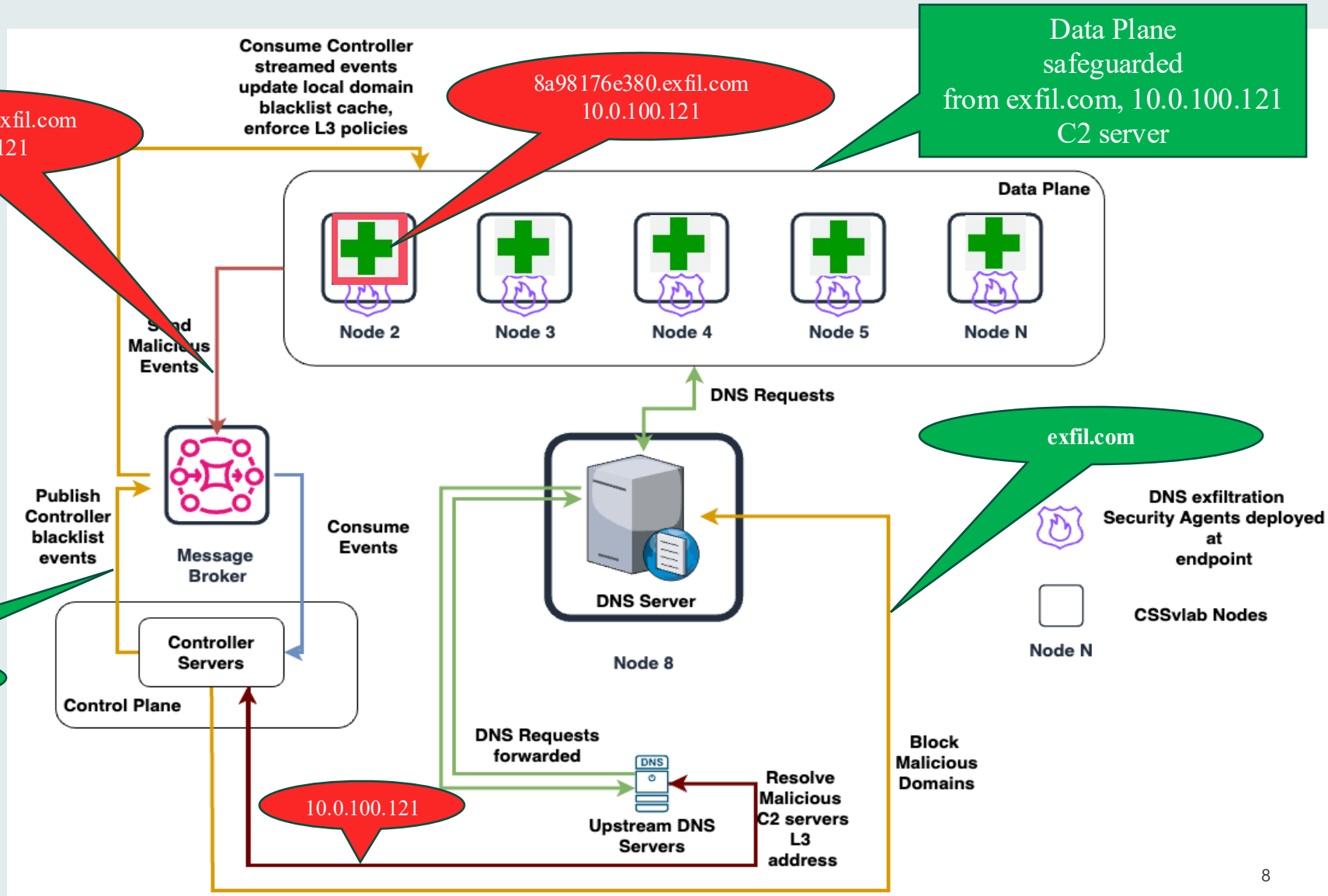
Solution:

Real-time, proactive enforcement at Ring 0 — inside the kernel.

DNS Exfiltration Security Framework Overview

Architecture Components

- **Data Plane**
  - eBPF endpoint agents
- **Control Plane**
  - Controller Servers
- **Infrastructure**
  - DNS Server
  - Message Broker



Security Framework Goals

Disrupt DNS covert C2 channel attacks, data exfiltration.

Implement in-kernel deep packet inspection and enforcement to block all forms of DNS exfiltration channels.

AI-Assisted Threat Detection

Use deep learning in userspace to detect advanced obfuscated exfiltration payloads with high accuracy aiding kernel network enforcements.

Malicious Process Aware Active Response (Threat-Hunt and Kill)

Link exfiltration attempt to parent process and kill implants processes, preventing lateral movement and further damage.

Dynamic Cross-Layer Policy Enforcement

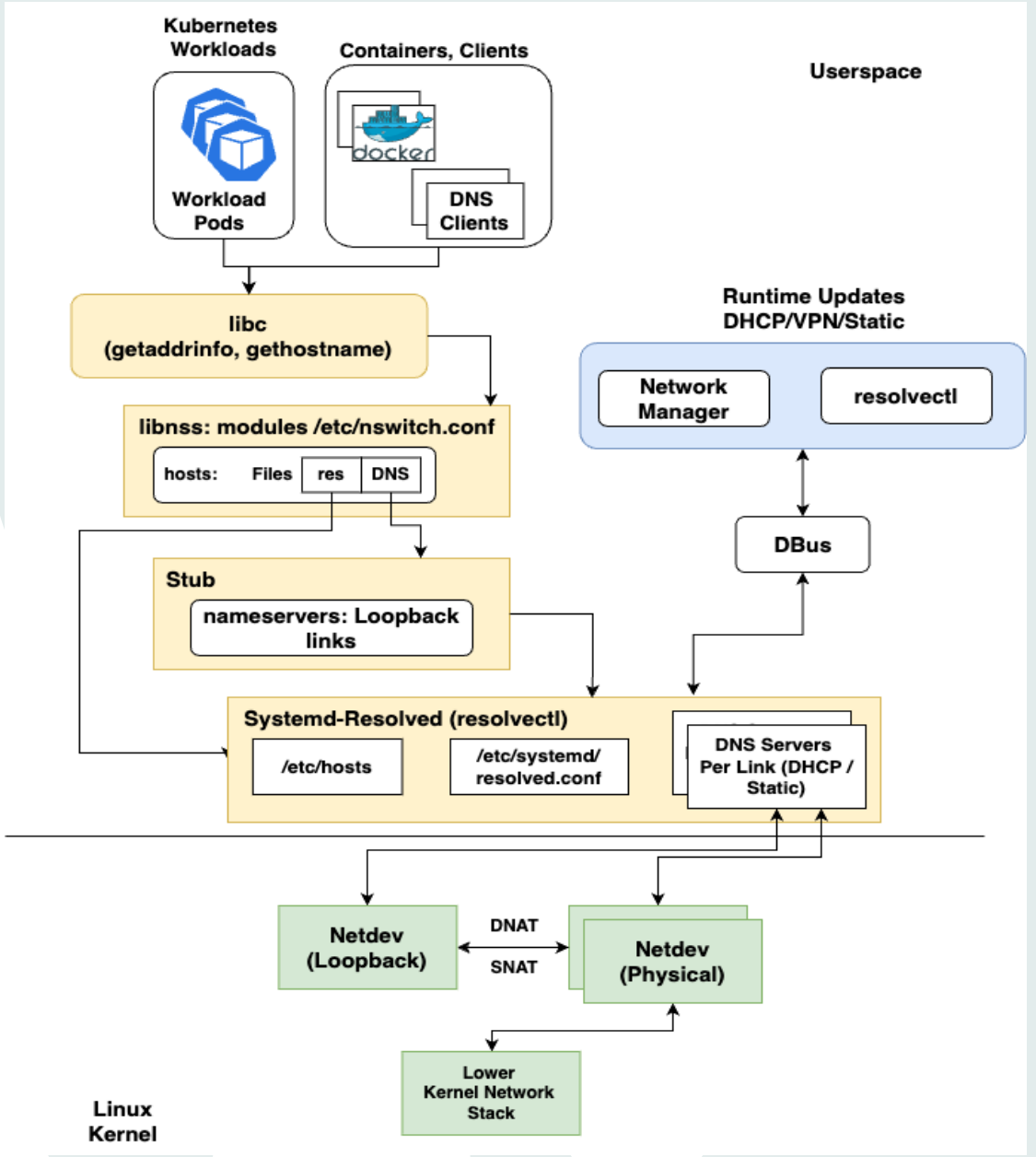
Enforce in-kernel L3 network policies adaptively and domain blacklisting on DNS server to combat DGA.

Scalable Multi-Cloud Deployment

Ensure framework's horizontal scales for real-world production cloud environments.

Systemd-Resolved

- **Userspace**
  - Libc (dns_utils)
  - Libnss (nss modules (nss-dns, nss-myhostname)
  - Systemd-resolved (resolvectl)
  - System Daemons
    - Network Manager (DHCP)
    - Dbus
- **Kernel**
  - Network Stack each netdev (east-west, north-south traffic)



Kernel Enforced Endpoint Security

Agent based Endpoint Security

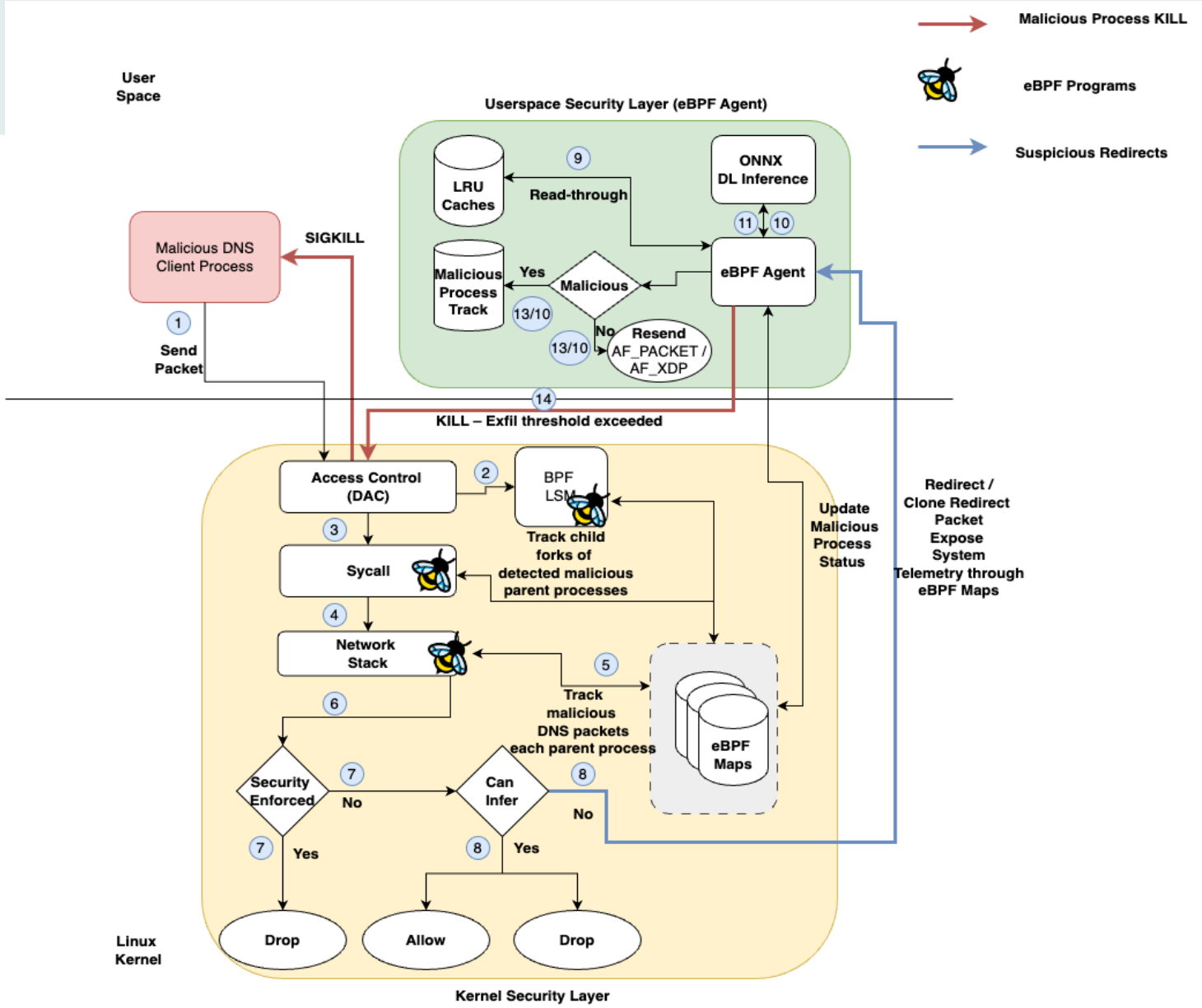
Continuous Security Enforcement Event Loop

Userspace

- eBPF Agent
- eBPF Agent LRU Caches
- ONNX Quantized Deep Learning Model
- Kernel malicious metrics export

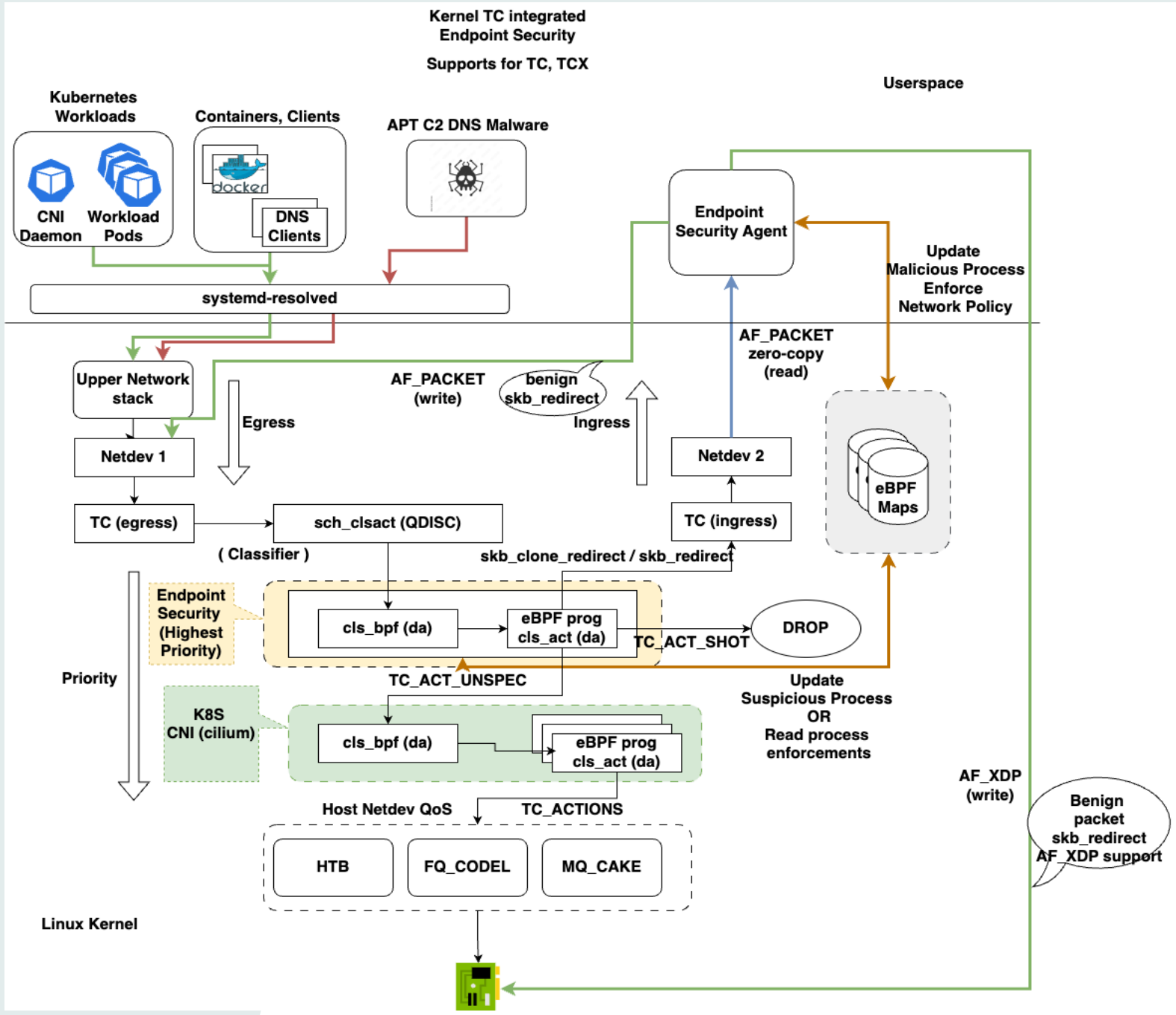
Linux Kernel

- **Inference Unix Domain Sockets**
- **eBPF Ring Buffers (malicious events)**
- **Network Stack (eBPF programs)**
  - Socket Layer
  - Traffic Control
- **Access Control Layer (eBPF programs)**
  - Security Modules (eBPF LSM)
  - Syscall (eBPF Tracepoints)



Kernel Datapath Enforcement Layer

- Sockets
- TCP/IP Stack
- Netfilter
- Traffic Control (QoS)
- Network Drivers (XDP)



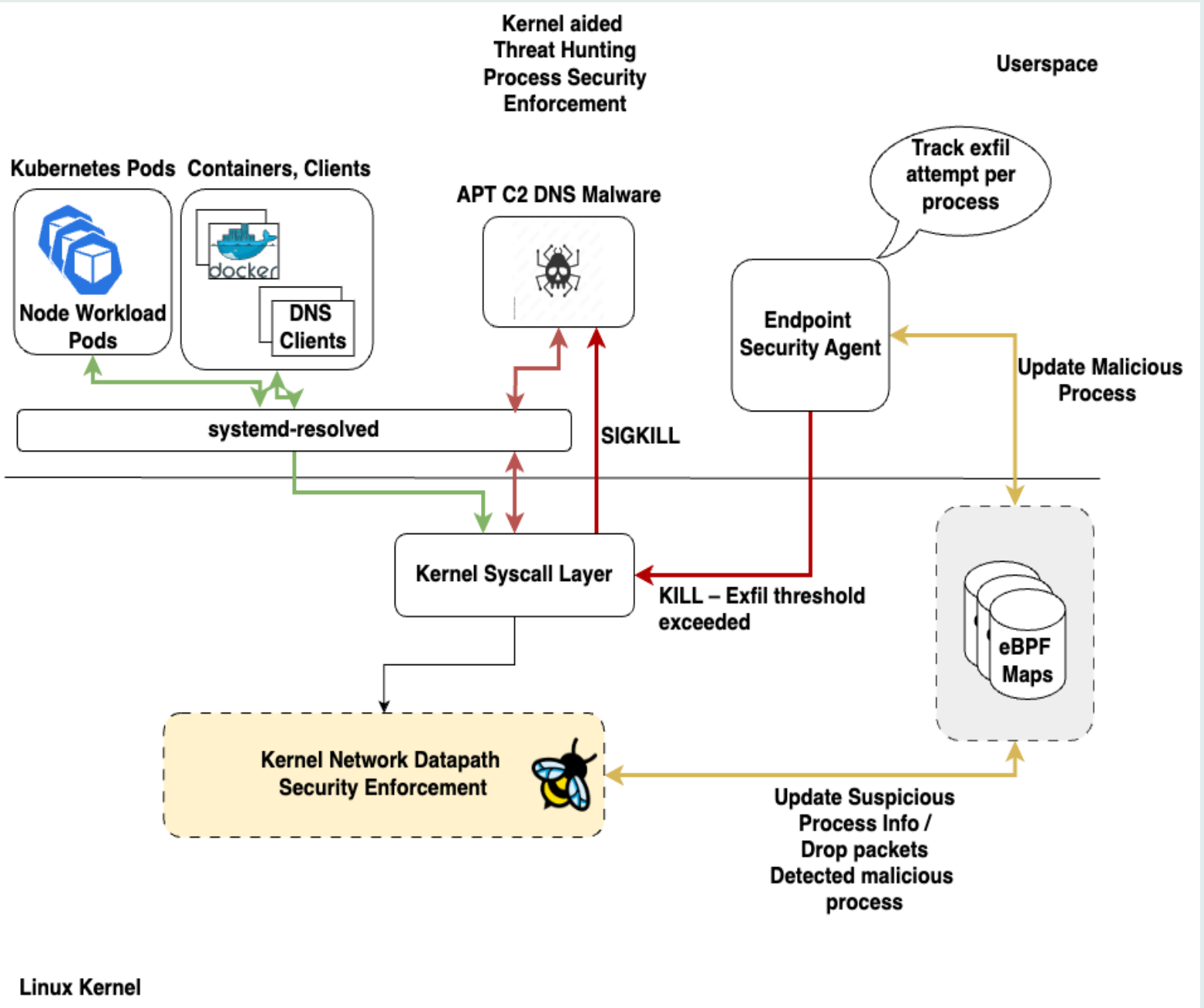
Process Enforcement Layer

Userspace

- Send SIGKILL to malicious process

Linux Kernel

- Kills the malicious implant instructed by userspace endpoint security agent.



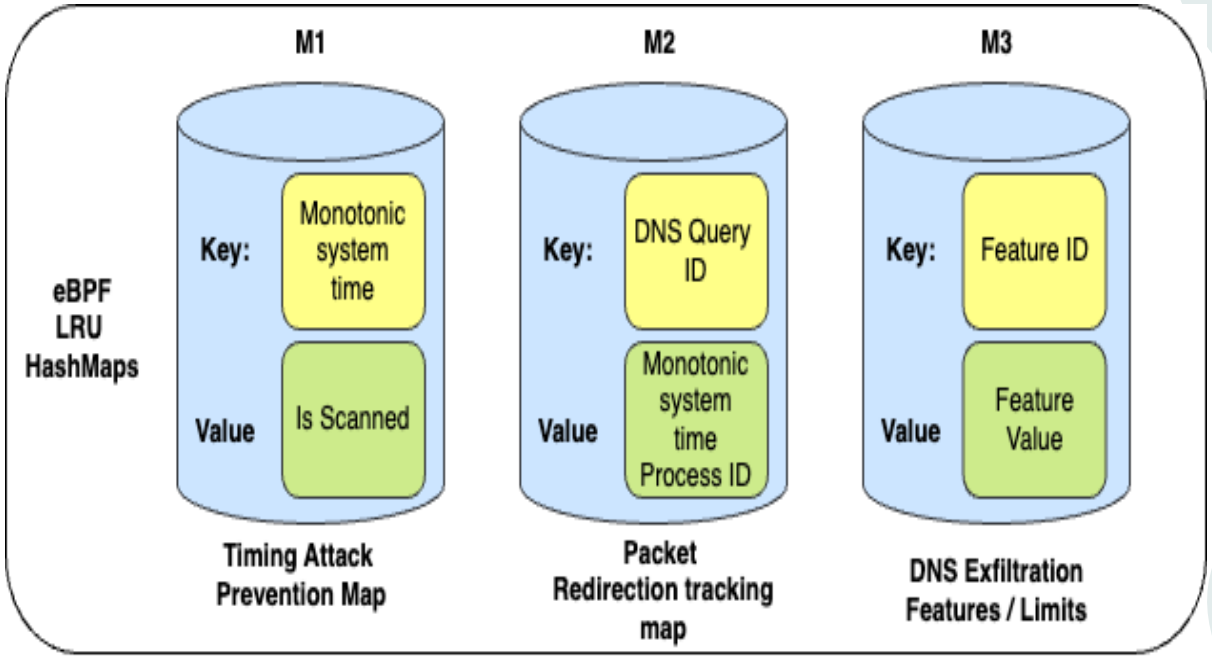
eBPF Endpoint Agent Operations Modes

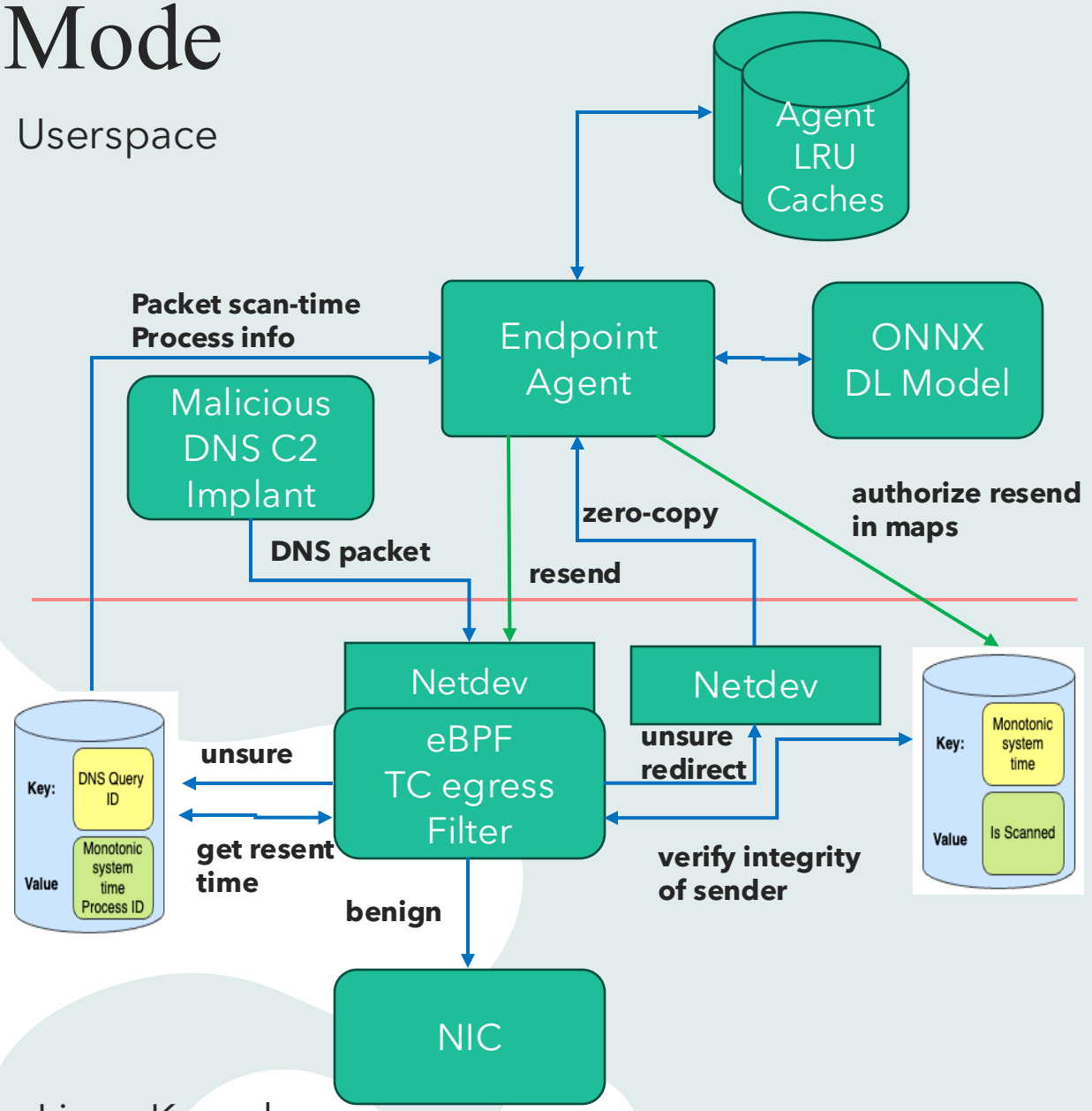
eBPF Agents in Data Plane handle DNS exfiltration over UDP

Mode	Goal	Requirement	Security Enforcement Process
Strict Enforcement Active Mode	Kill C2 Implants, ensure zero data loss and C2 command execution.	DNS Traffic over UDP ports (53, 5353,5355), for encapsulated and non-encapsulated traffic.	• **Kernel**: Live Redirects suspicious DNS packets to userspace. • **Userspace** Trace malicious process exfiltration count and terminates it, resend benign packets.
Process-Aware Passive Threat Hunt Mode	Kill C2 Implants, ensure negligible data loss and minimal C2 command execution.	DNS Traffic over random UDP ports.	• **Kernel:** Allow suspicious traffic passthrough. In Kernel start threat hunting process tied to malicious DNS packets. • **Userspace:** Trace malicious process and terminates it.

Strict Enforcement Active Mode

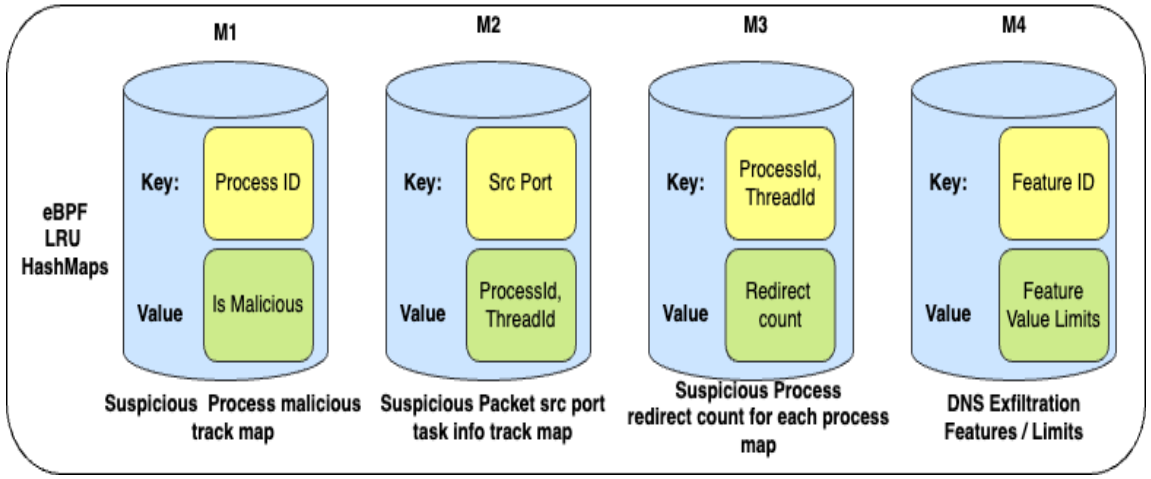
- **eBPF program deep parse of suspicious DNS packets from SKB**
- **Real-time verdict from kernel DPI eBPF program**
- **Userspace DL model aids classification**
- **eBPF timing-brute-force attack checks on resend**
- **Per-process exfil attempts tracked in userspace**

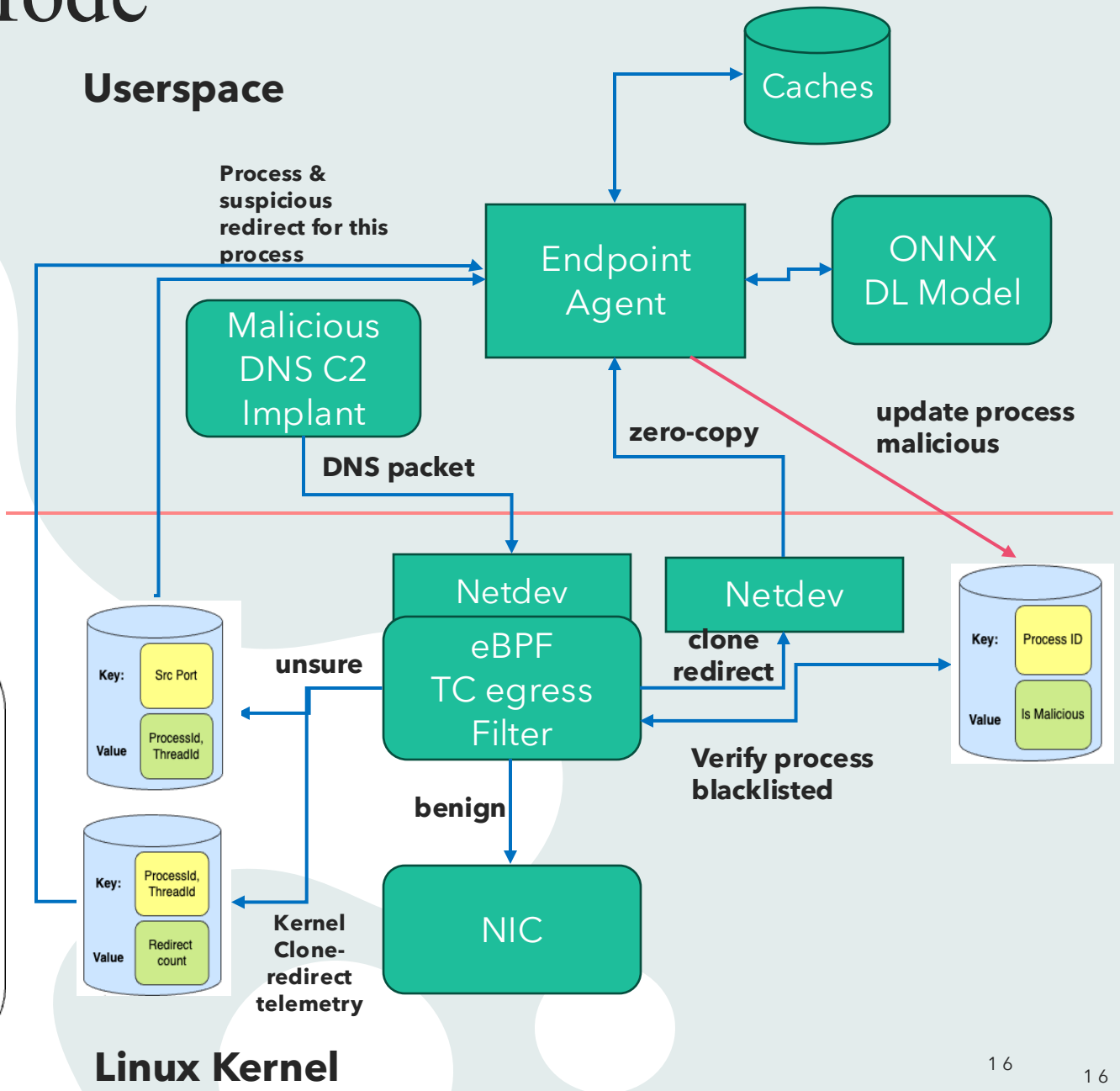




Process-Aware Passive Mode

- eBPF kernel program deep parses DNS from skb
- Suspicious packets cloned to userspace
- DL model classifies and tags process
- Malicious process flagged and eBPF maps updated
- eBPF kernel program drop packet from malicious process + clone for exfiltration attempt telemetry



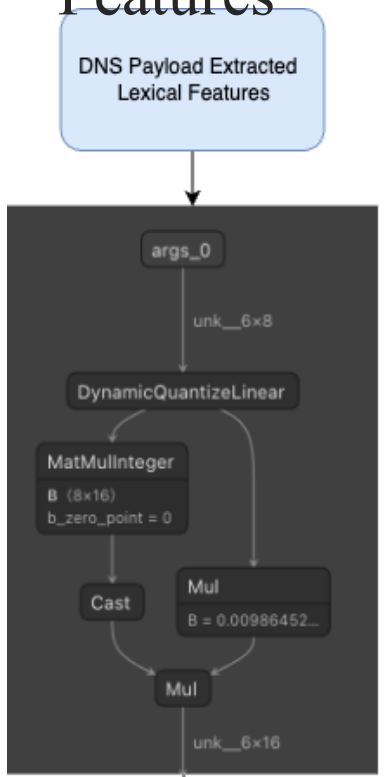


16

DNN based DNS Data Obfuscation Detection

- Model Architecture ONNX
- Sample Malicious Data
- Input Features

DNN Features



Relu

Feature	Description
total	
max	
la	

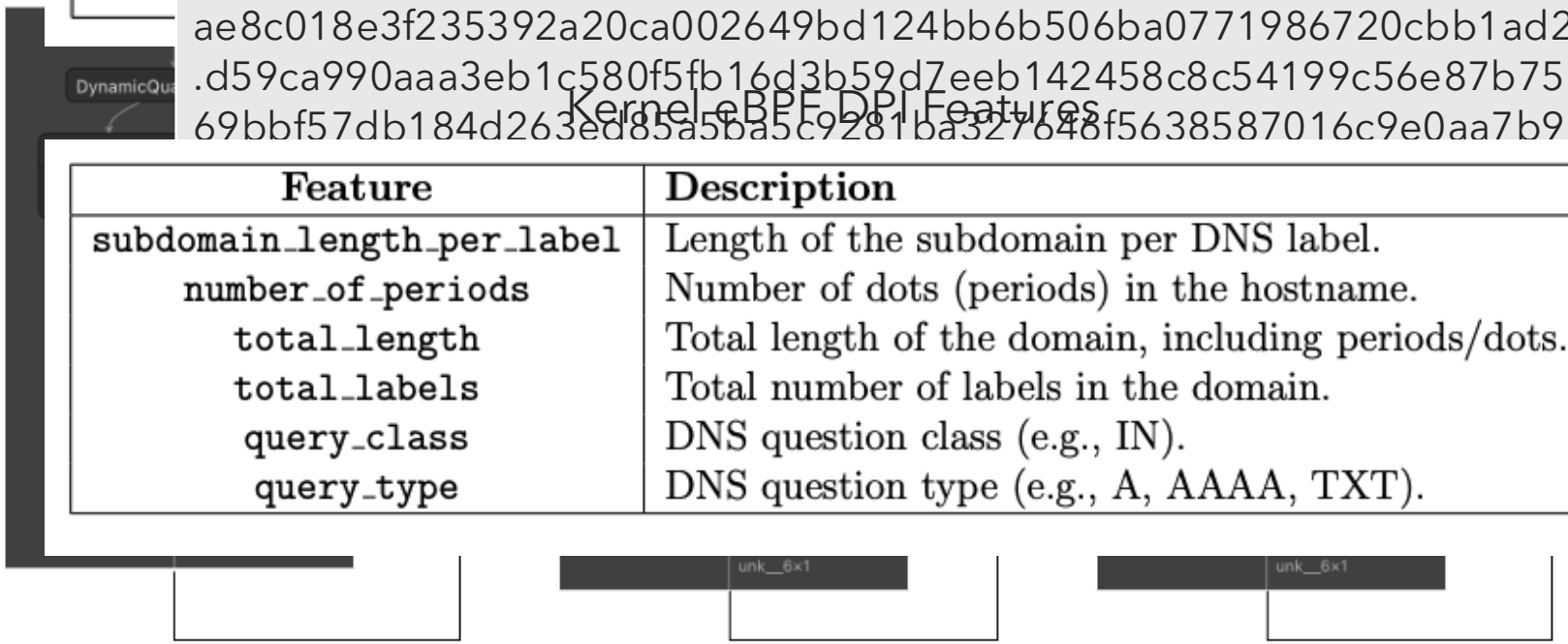
Malicious Exfiltrated data DNS queries

381c018e3f5d05b78e3f6a026381e0f3476c066e8017be6ba9f5a9d758ef.d
04bc3e0fc58e5a2401da590f3ee268a6af637eaafd210e58060a41082dc.92
d594840bcb32a6500f39248db646e4e602f8547294692d83a4b4680223.b
4d0ce0ec94abc9b6821cea90561aac558a6ba30b53e6b.bleed.io

ae8c018e3f235392a20ca002649bd124bb6b506ba0771986720cbb1ad2e2
.d59ca990aaa3eb1c580f5fb16d3b59d7eeb142458c8c54199c56e87b751c.
69bbf57db184d263ed85a5ba5c9281ba327648f5638587016c9e0aa7b9b8.

Kernel eBPF DPI Features

Feature	Description
subdomain_length_per_label	Length of the subdomain per DNS label.
number_of_periods	Number of dots (periods) in the hostname.
total_length	Total length of the domain, including periods/dots.
total_labels	Total number of labels in the domain.
query_class	DNS question class (e.g., IN).
query_type	DNS question type (e.g., A, AAAA, TXT).



Input Layer

Hidden Layer 1

Hidden Layer 2

Hidden Layer3

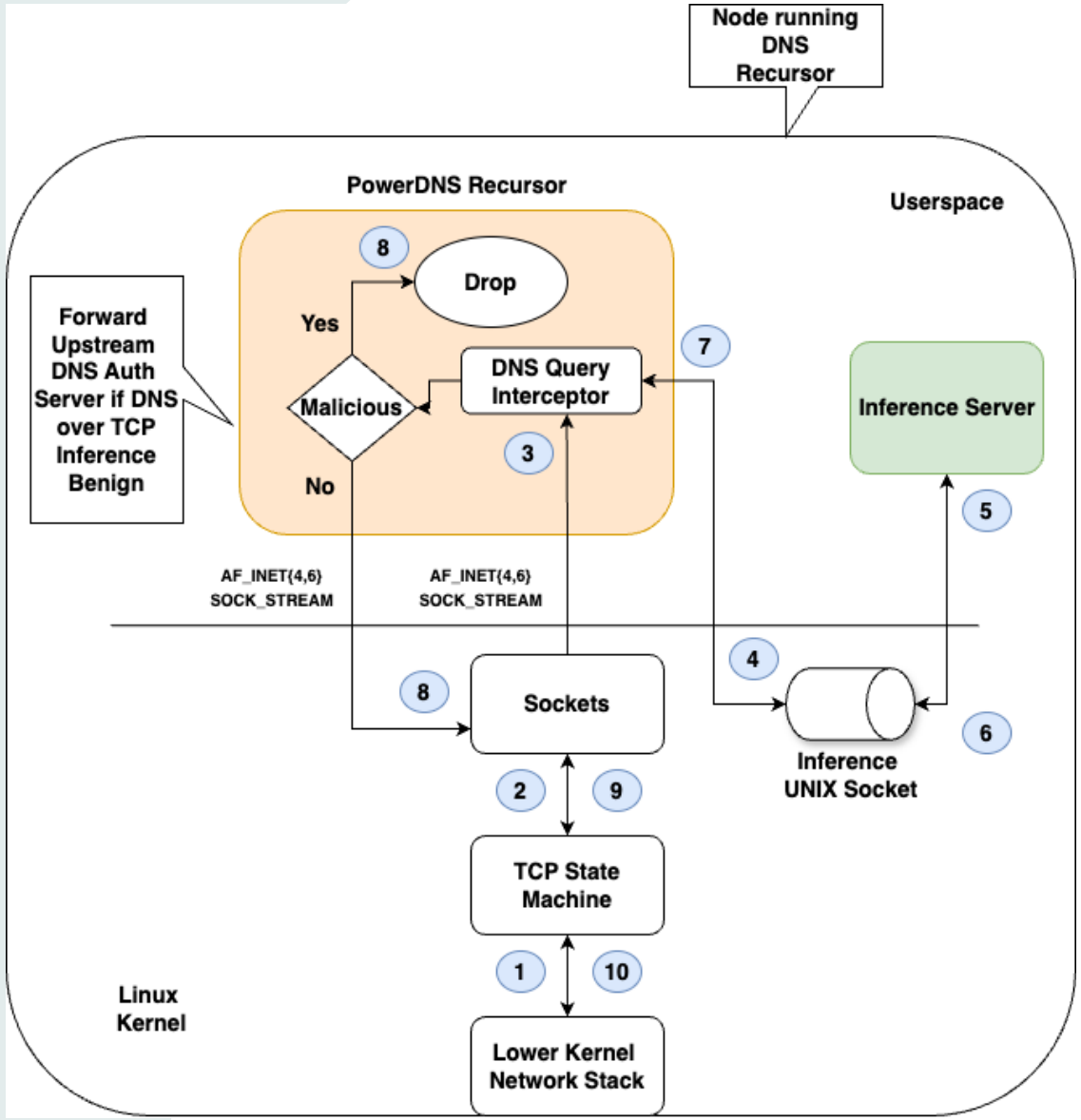
Datasets

Dataset Type	Source / Characteristics	Size	Primary Goal
Trusted Benign Cache	Top 1M Cisco Second-Level Domains (SLDs)	1 Million	Reduce inference on known-good traffic.
ISP-Captured DNS	Live-sniffed ISP DNS traffic [Ziza et al.]	50 Million	Provide real-world benign & malicious baseline.
Synthetic Exfiltration	Custom-generated (DET, DNSCat2, Sliver, Nuages, Custom Scripts, etc.);	2.4 Million	Malicious samples use varied obfuscation across file formats
Final Combined Dataset	Synthetically formed	3.8 millions	Balanced dataset w/ obfuscated payloads across file formats

DNS Exfiltration over TCP

Prevent DNS Exfiltration over TCP

- Runs on
  - PowerDNS Recursor
- Relies on
  - PowerDNS recursor Query Interceptors
  - Inference UNIX domain sockets



Results and Evaluation

- Model Metrics
- Throughput comparisons (Active mode)
- Response Time per Exfiltration attempt
- Kernel DPI time (raw parse DNS protocol from SKB)
- Resources
  - Memory Usage
    - Security Agent memory usage at endpoints in data plane
  - Endpoint Agent Flame Graph

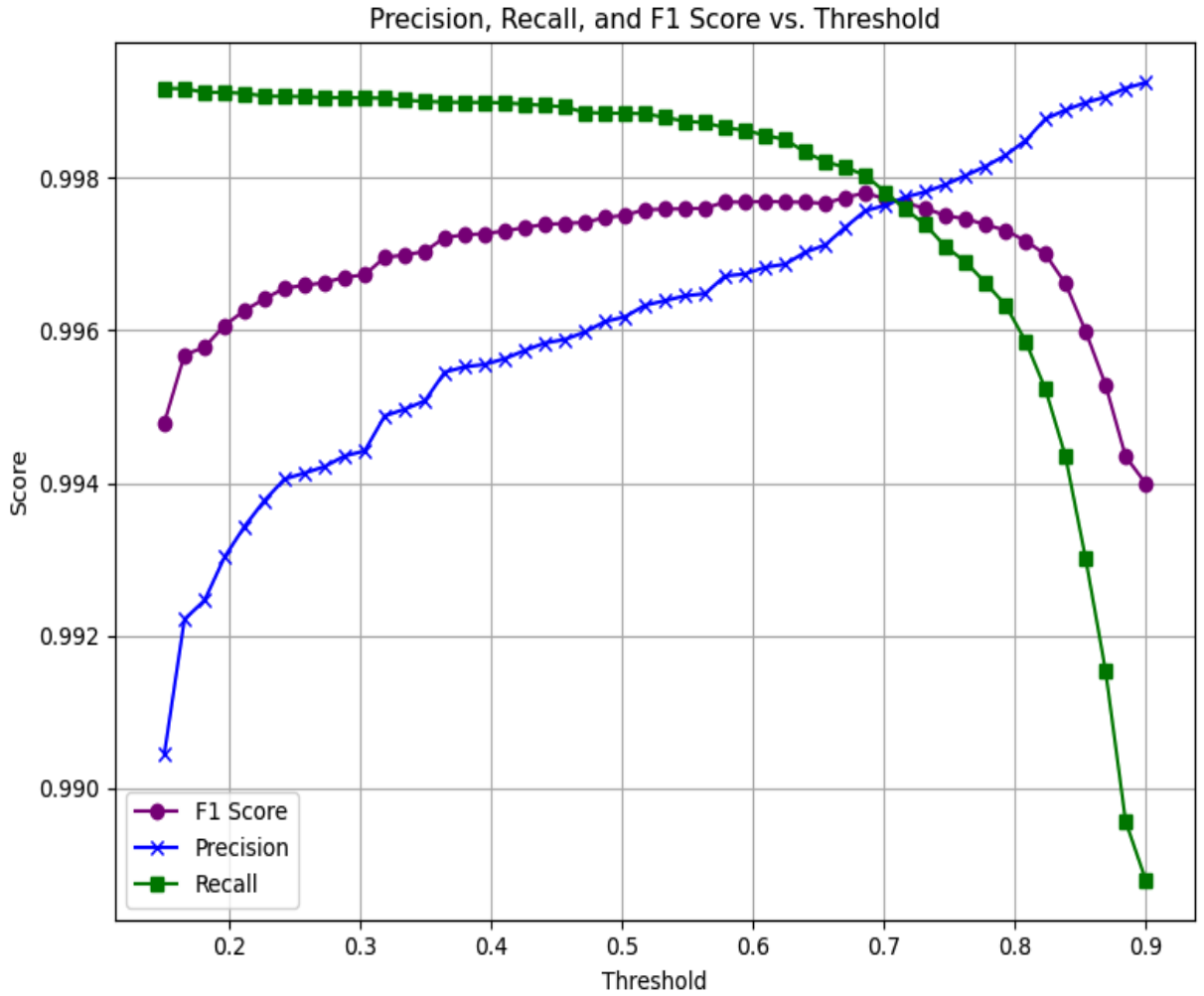
Test Bench
CPU: Intel Xeon 6130
Memory: 8 GB
Linux Kernel: 6.12.4
Network Driver: netvsc
Bandwidth: 100 Gb/sec
Root QDISC: FQ_Codel
Queues: 8 RX / TX

DNN Model Metrics

Metric	Training	Validation
Accuracy	0.9973	0.9997
AUC	0.9997	0.9997
Loss	0.0099	0.0091
Precision	0.9959	0.9959
Recall	0.9987	0.9988

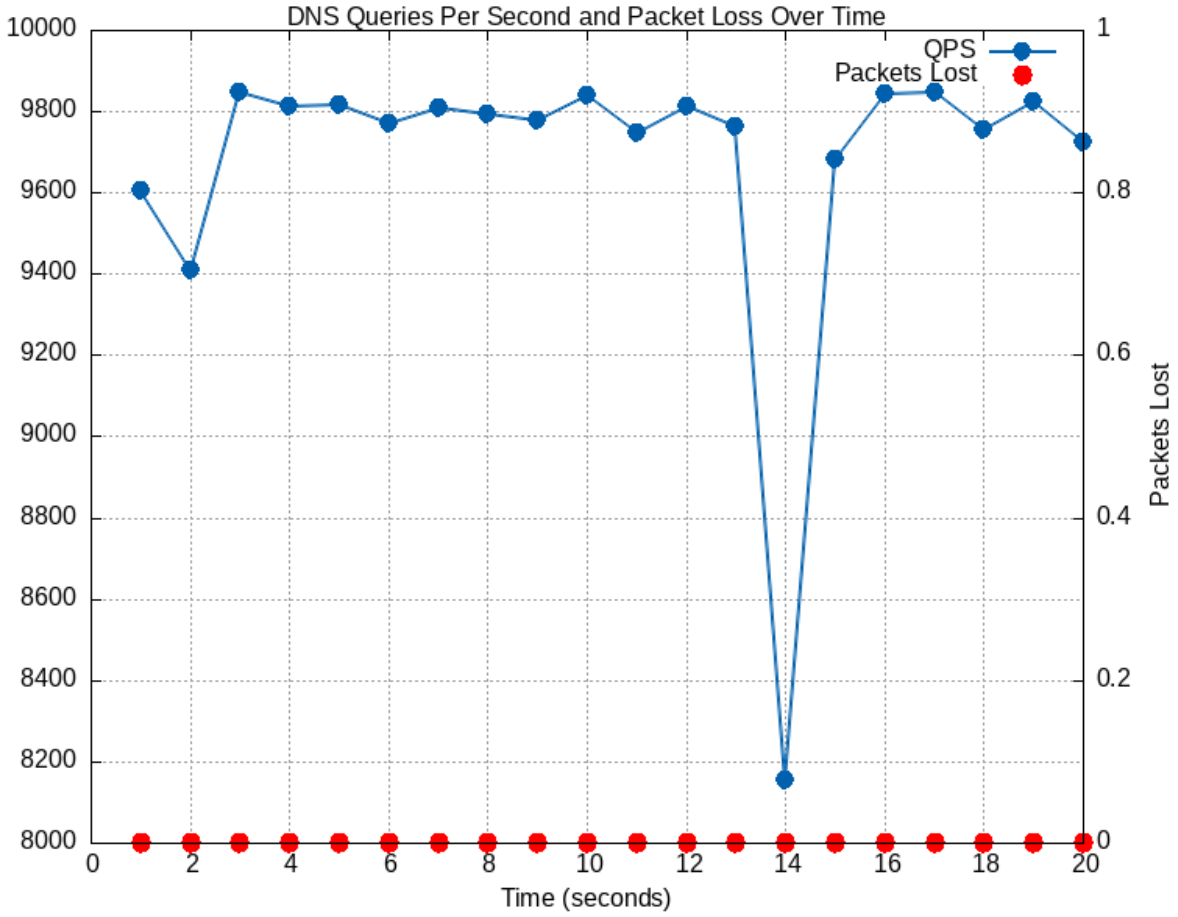
Table 5.1: Model Evaluation Metrics

Model Performance

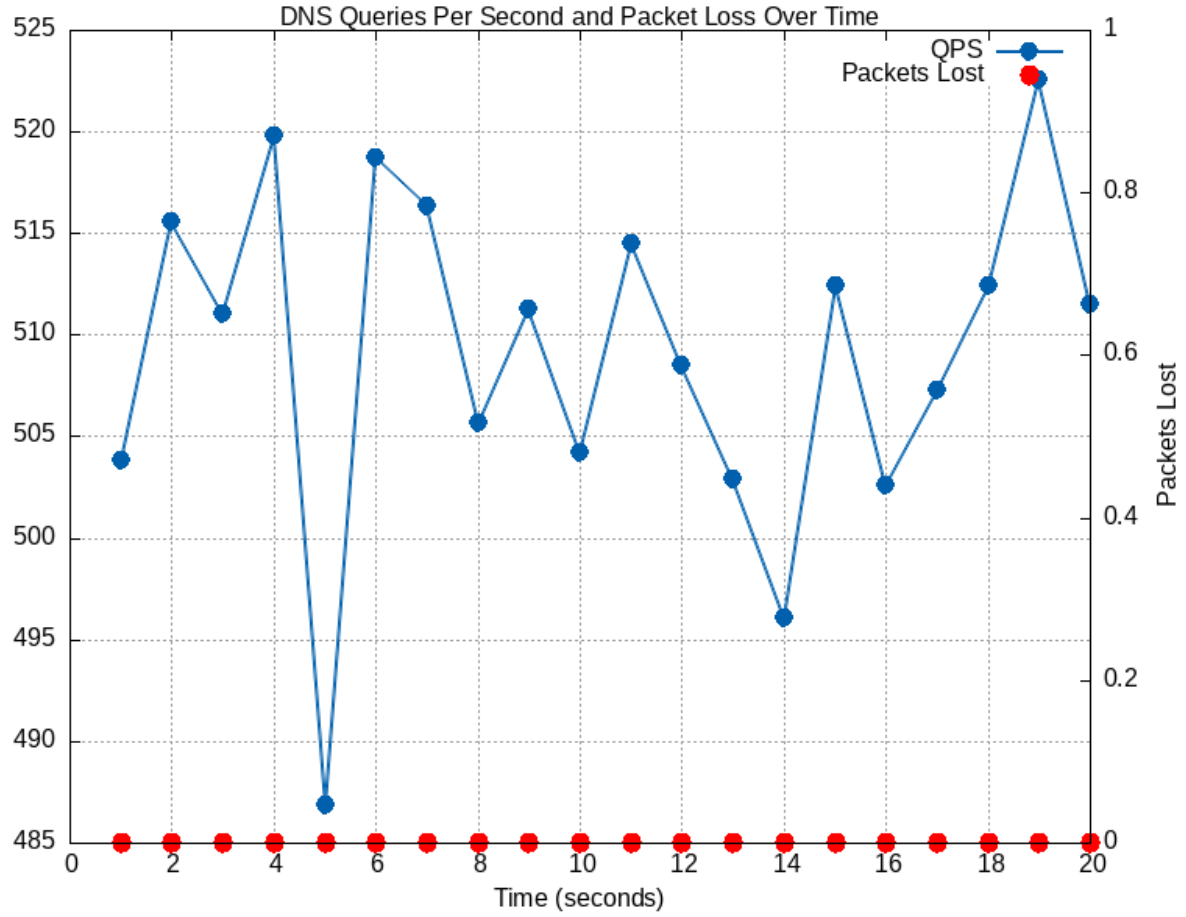


Model Scores

Throughput comparisons – Active Mode

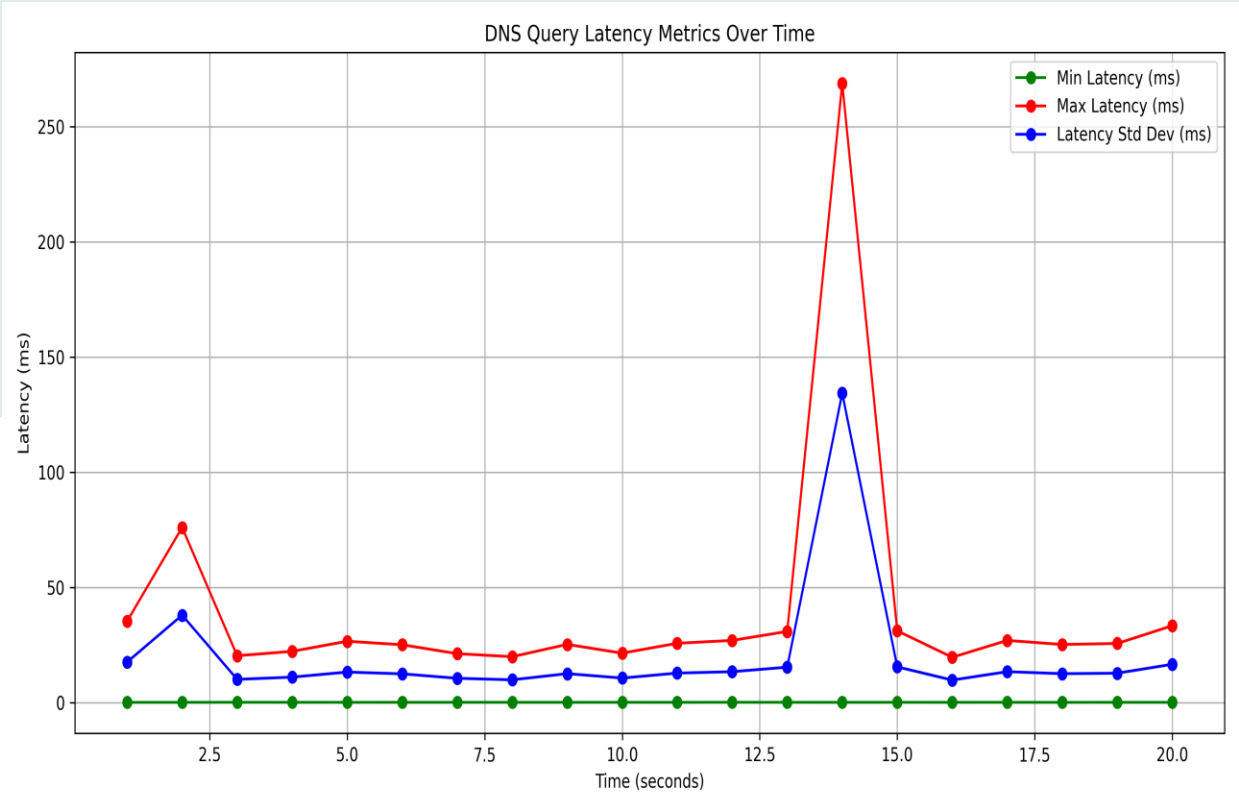


Agent LRU Cache Read-Through Hit 10k DNS req/sec

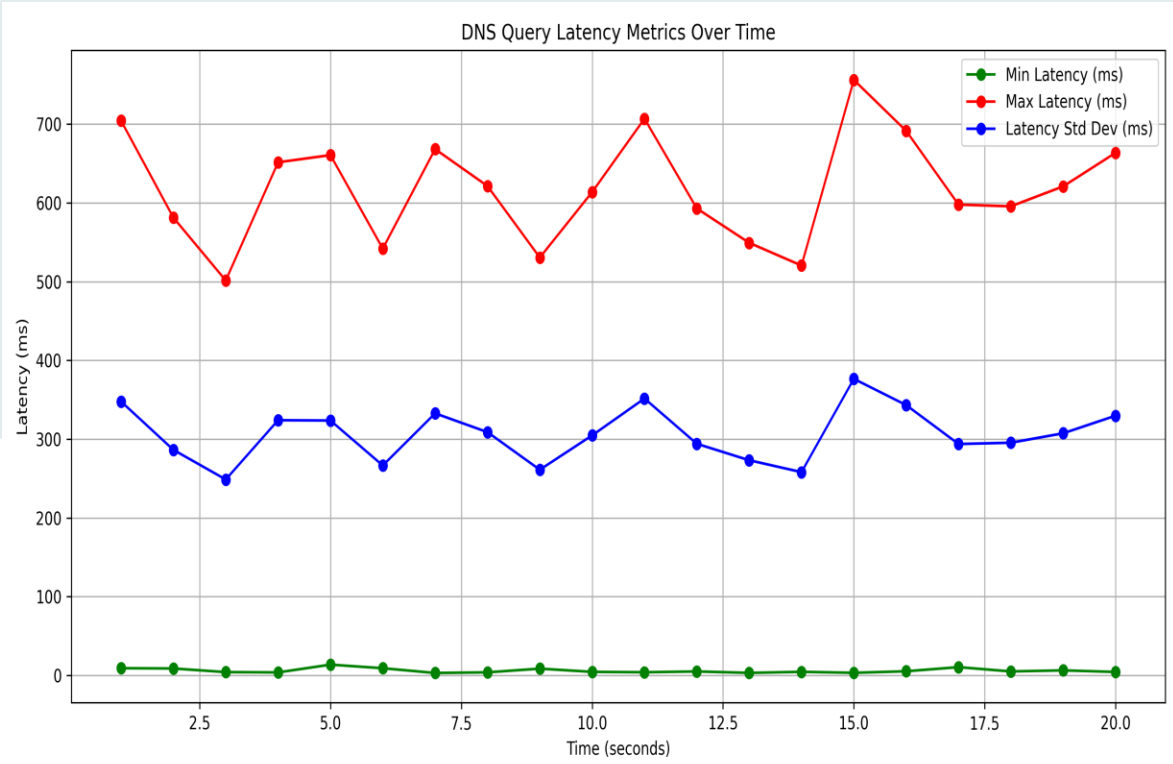


ONNX Live Inferencing 10k DNS req/sec

Throughput comparisons – Active Mode (continued)



Agent LRU Cache Read-Through Hit

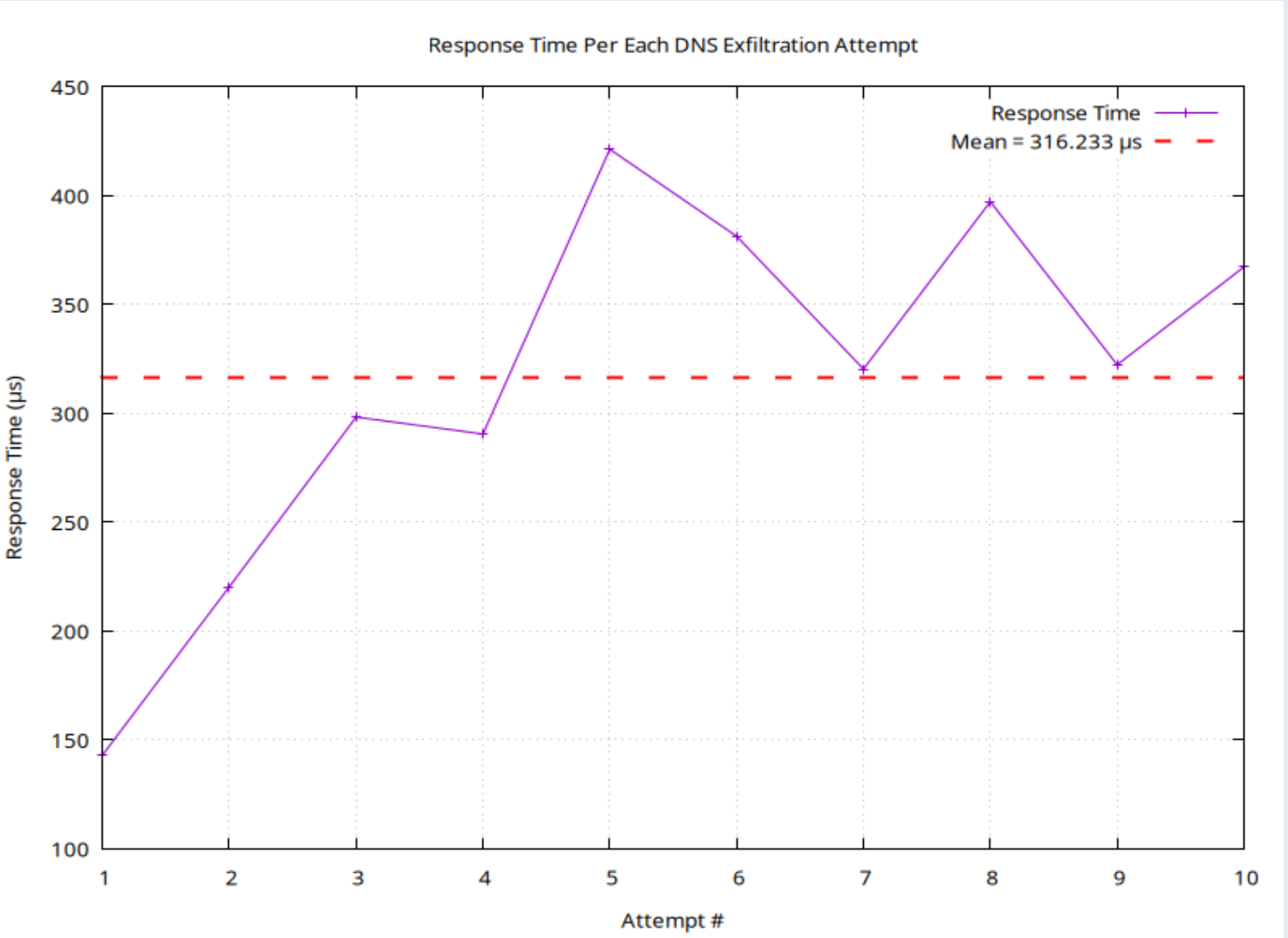


ONNX Live Inferencing

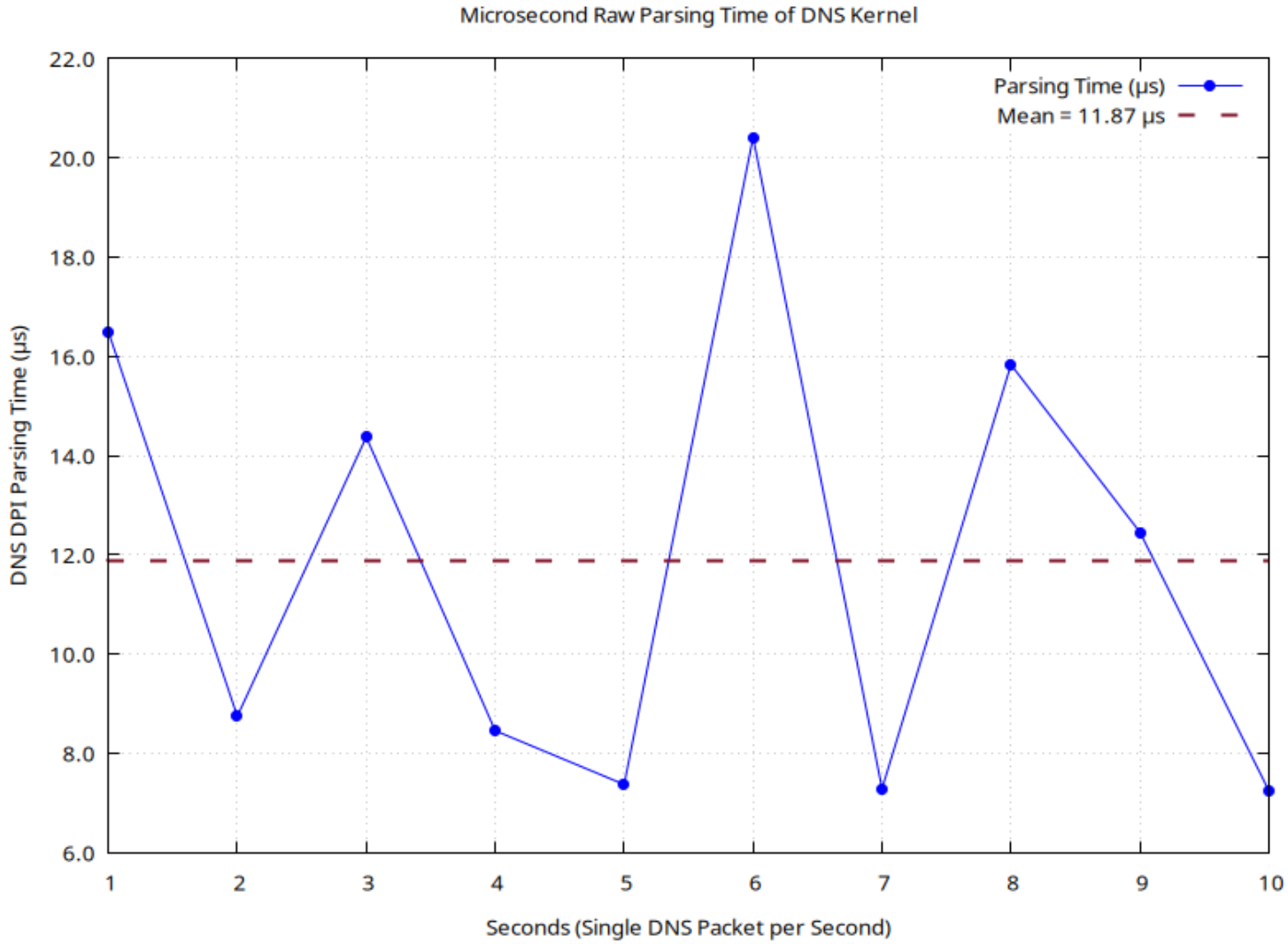
Response Speed - Active Mode

Response Speed Before Implant Eventually Killed

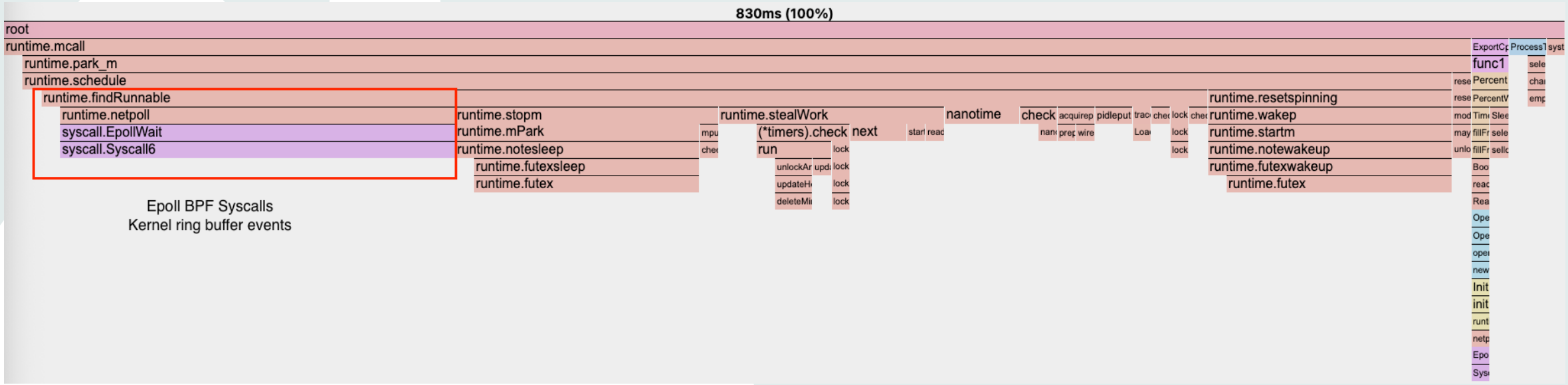
- Each Exfiltration Attempt

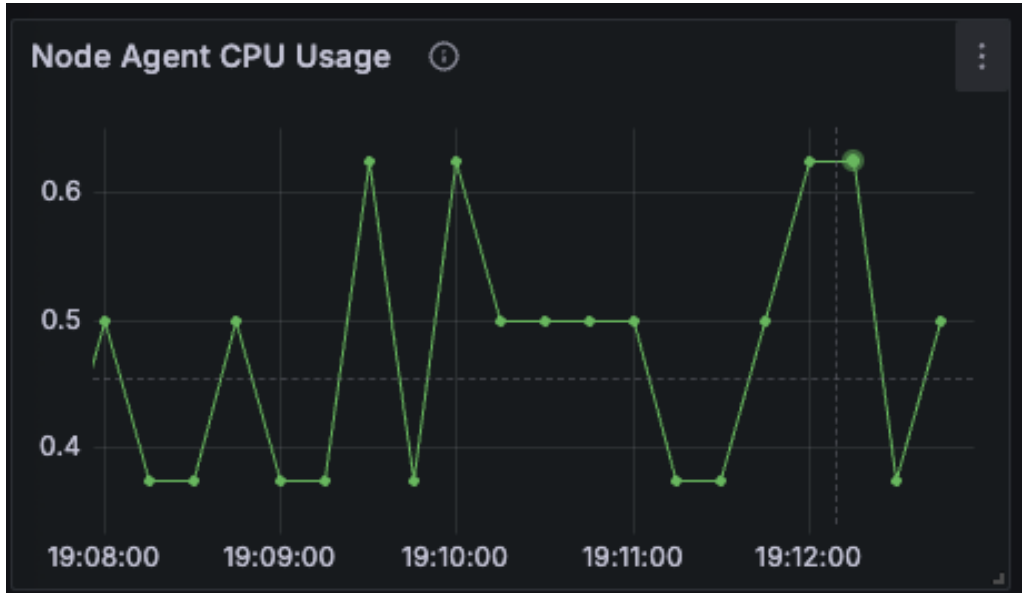


Kernel eBPF Program DNS DPI Time (SKB Parsing)

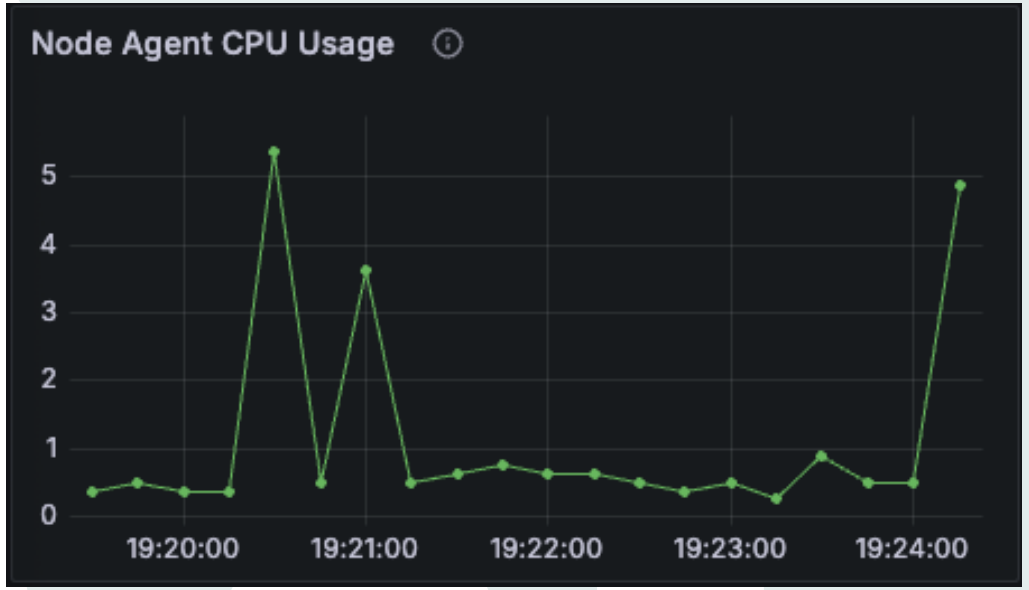


Resource Usage – eBPF Agent Flame Graph



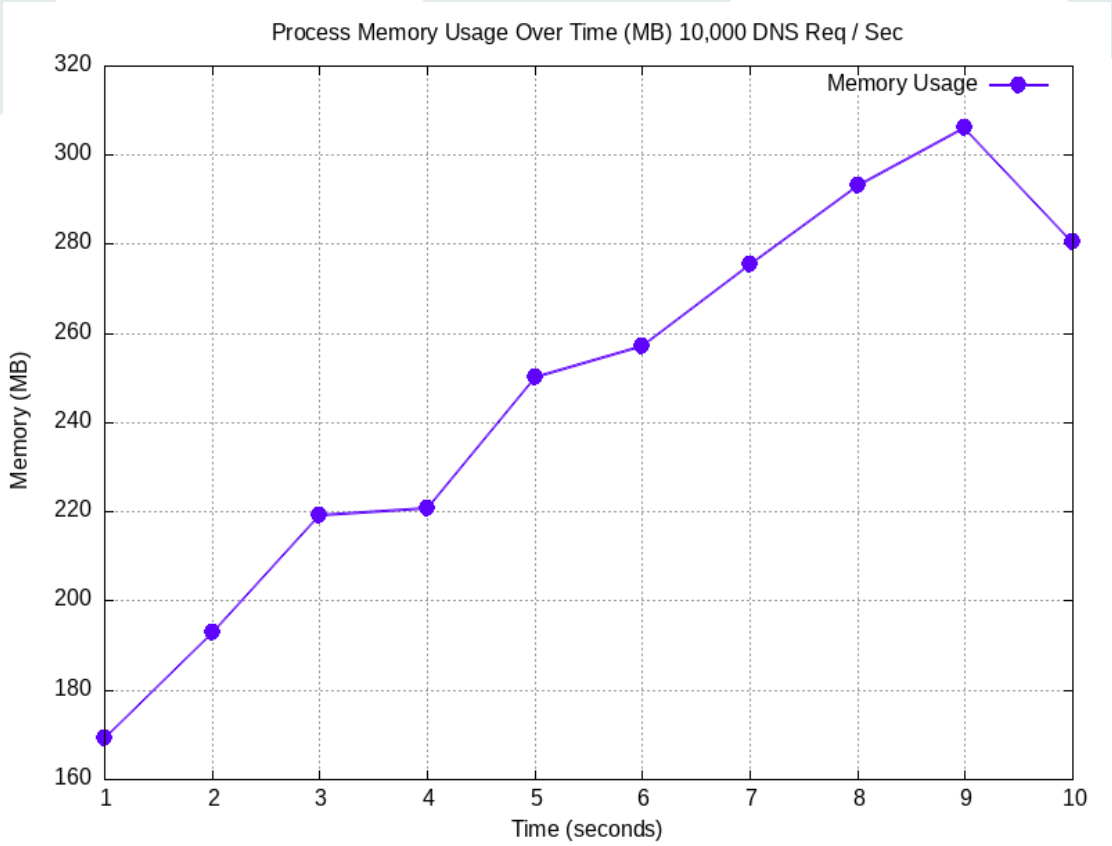


Idle Usage

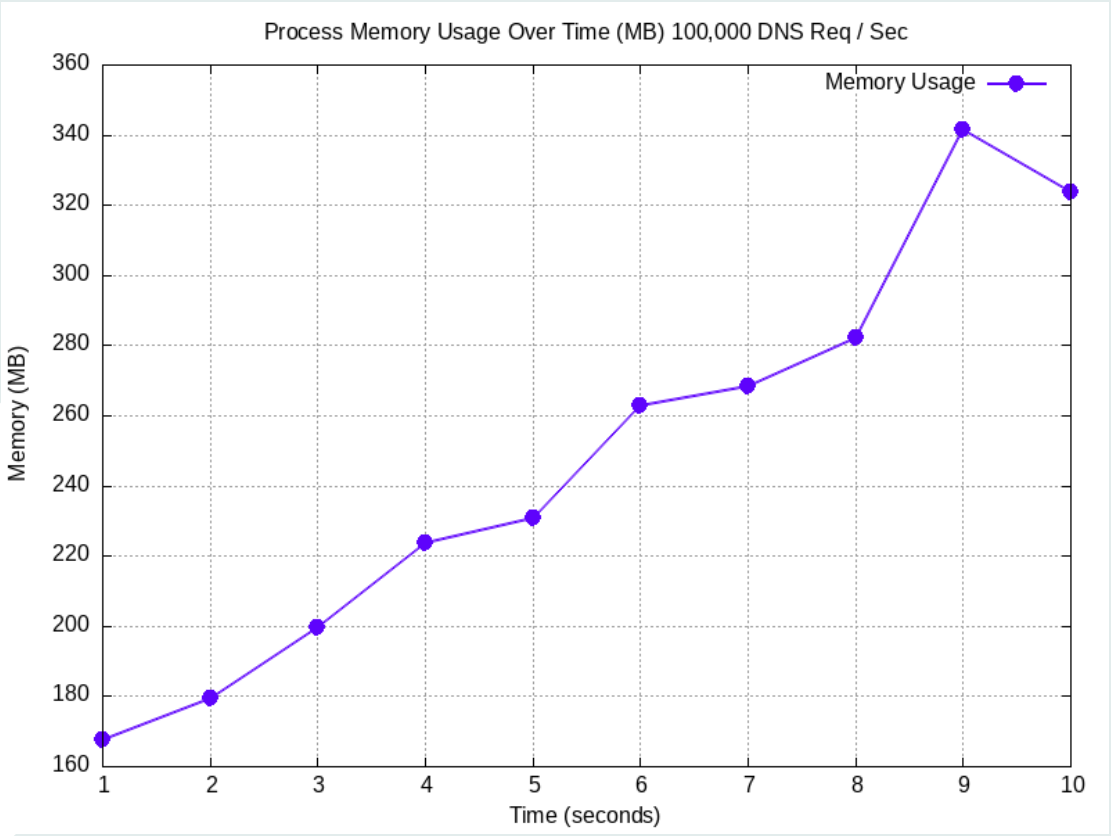


CPU Throttle Run

Resource Usage - Memory

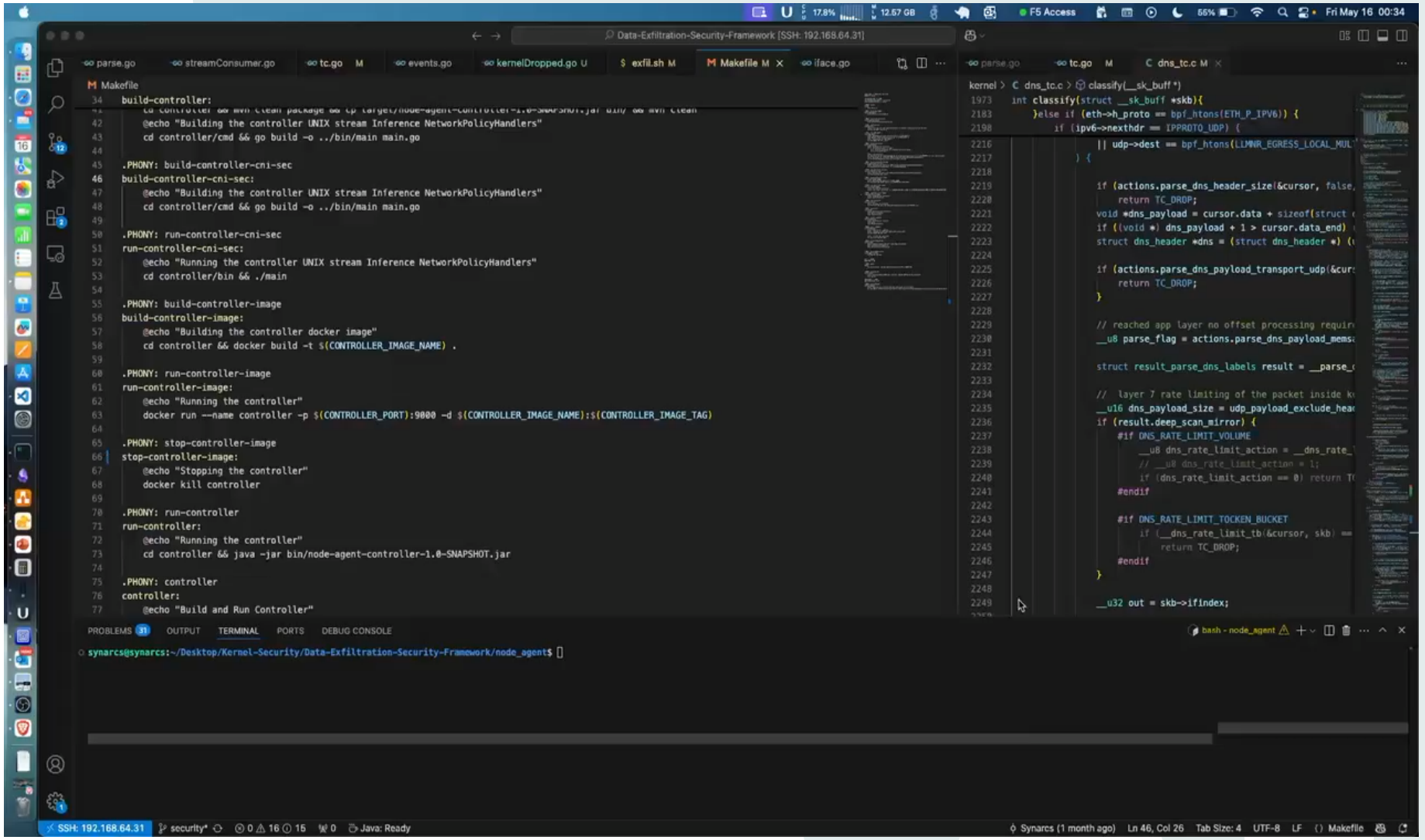


10,000 DNS Req / Sec



100,000 DNS Req / Sec

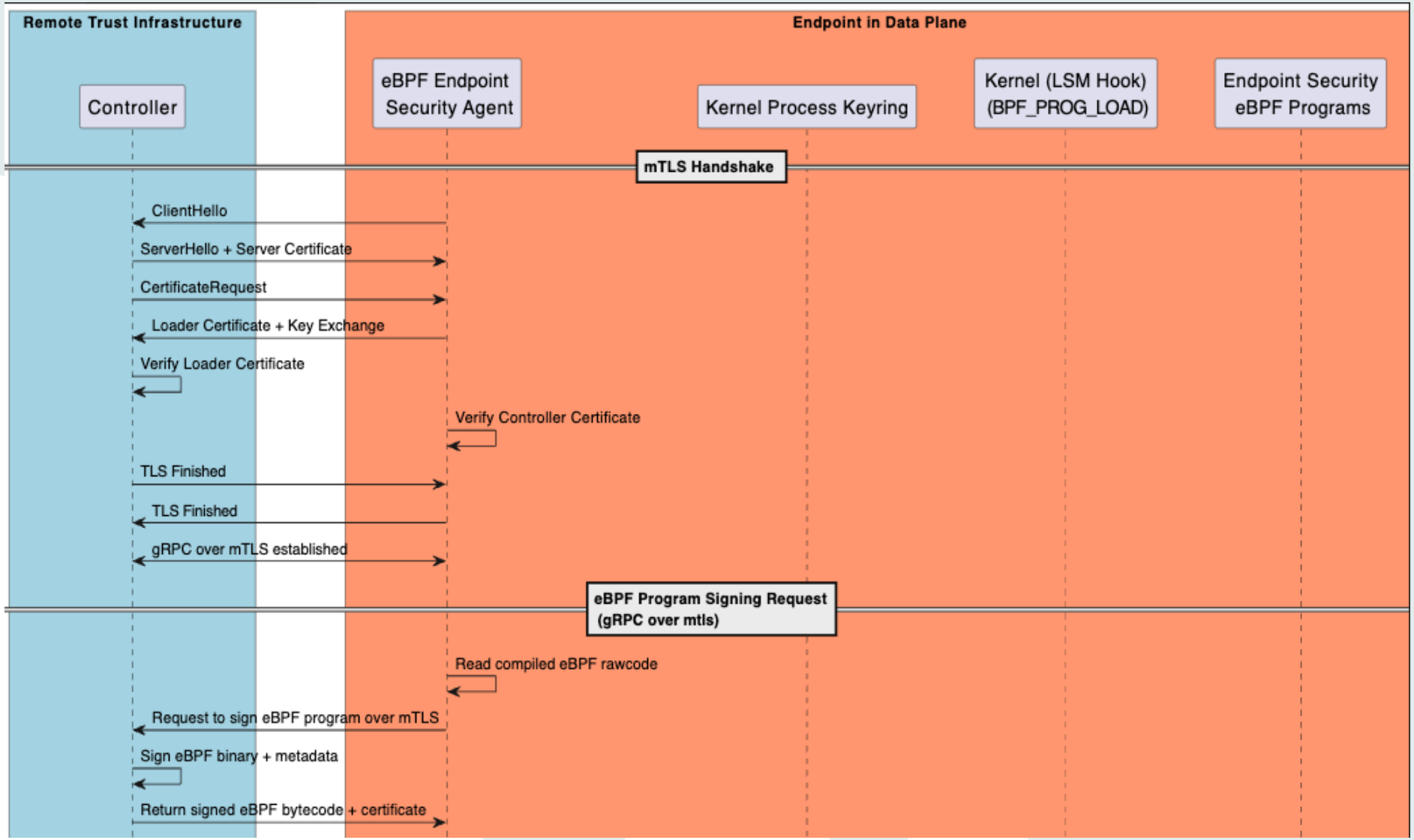
Framework Security Strength : Demo



First Trust Chain (Endpoint-Agents – Cloud Trust Infrastructure)

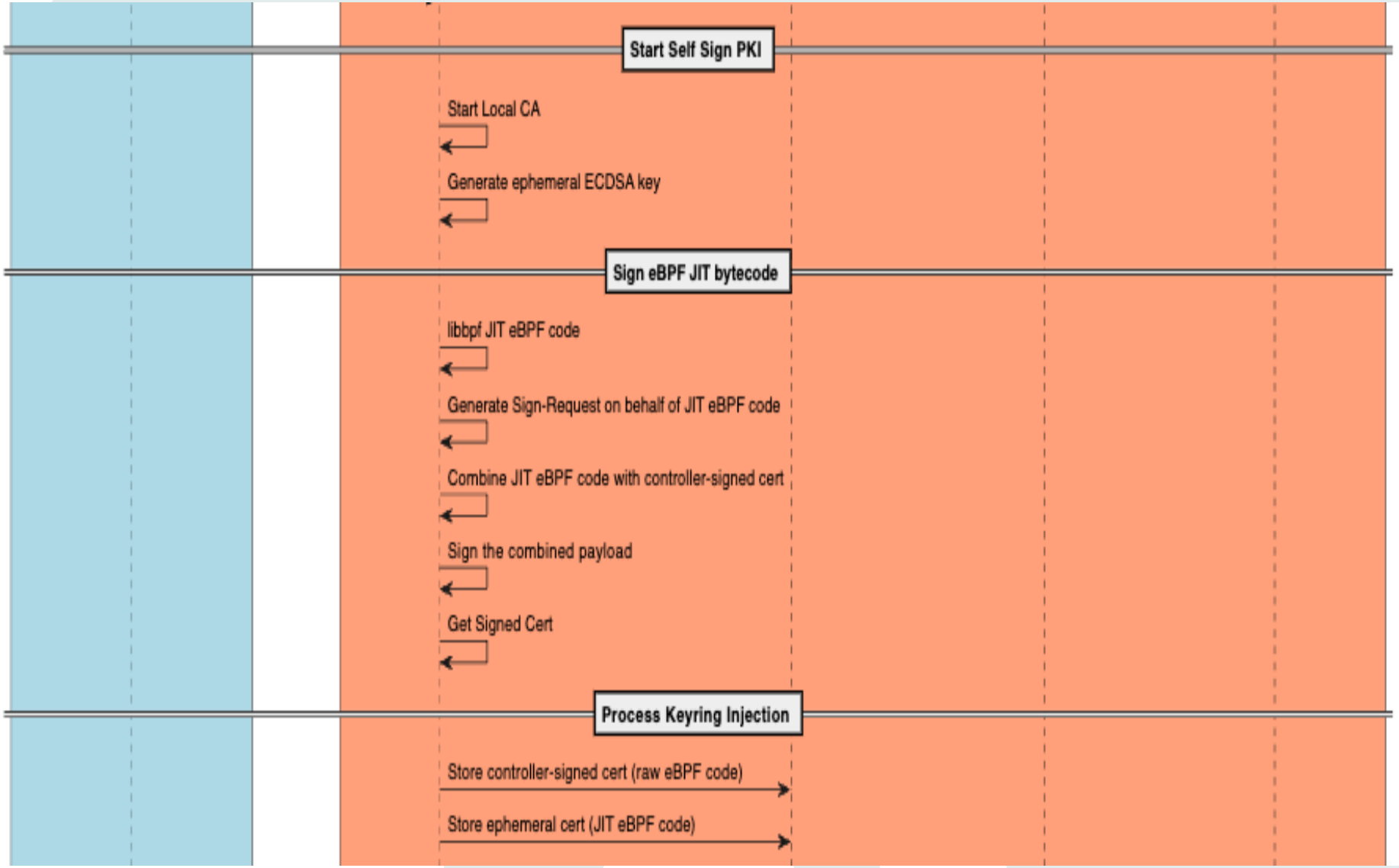
- **Endpoint Agents**
- **Controller (Remote PKI)**

- **mTLS-based identity handshake**
- **Remote-signed eBPF raw bytecode**



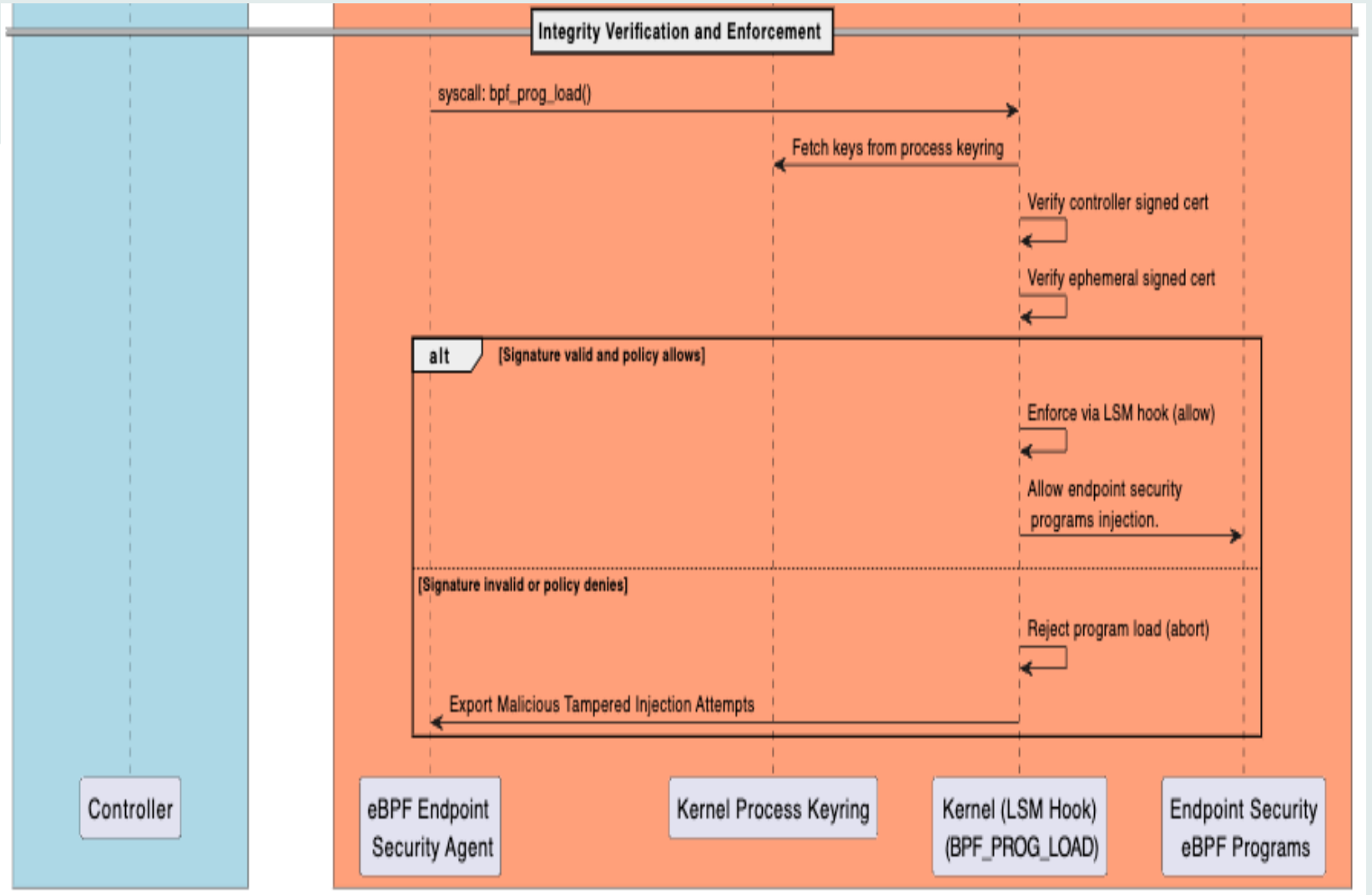
Ephemeral Runtime Signing Bootstrap PKI (JIT bytecode integrity)

- **Ephemeral PKI bootstrapping**
- **JIT eBPF bytecode signing**
- **Controller-bound cert injection**
- **Keyring-based program identity**



Second Trust Chain (Endpoint-Agents – Kernel Keyring)

- **Verify controller signed cert (raw eBPF bytecode)**
- **Verify ephemeral signed cert (JIT bytecode)**
- **Ensure 3-way strong integrity prevent eBPF program tampering.**



Summary and Enhancements

- **Endless Security Enforcement Loop:** Kernel-enforced security strengthens EDR with real-time defense against advanced C2.
- **Dynamic in-kernel security enforcement through eBPF:** Reprogram kernel with proactive defense over advanced C2 mutating nature.
- **Extend Support for DNS-over-TCP:** Implement in-kernel eBPF-based detection for DNS-over-TCP replicating TCP state machine over kernel socket layer, paired with userspace DPI via Envoy proxy.
- **Add In-Kernel TLS Fingerprinting and Encrypted Tunnels**: Use eBPF for TLS fingerprinting (e.g., JA3/JA4) to detect DNS exfiltration over TLS (DOH), DNS over TLS, WireGuard.
- **Continuous Model Evolution:** Drift detection, online learning, and confidence-based live updates to maintain precision against emerging DNS obfuscation tactics.
- **Cloud Native Security:**
  - Dynamic L3/L7 security enforcement over cloud Vnet’s / VPC via dynamic blacklist’s NACL’s.

Discussions and Q&A

Code:

https://github.com/Synarcs/DNSObelisk

WhitePaper:
https://github.com/Synarcs/DNSObelisk_Report

Endpoint Agent LOC

	Language	LOC
Kernel (eBPF)	C	3314
Userspace	Go, Python	8862

Rest Framework LOC

	Language	LOC
Userspace	Go, Java, Lua	1991