**Dridex Traffic Analysis**

**Introduction:** Dridex is a long-running banking trojan that first appeared around 2011, primarily spread through malicious email attachments and macros. It is significant because it evolved from simple credential theft into a modular malware platform capable of delivering ransomware and other payloads, often used by organized cybercriminal groups. Understanding its network traffic, especially encrypted C2 communications, helps defenders develop detection strategies that remain effective even when attackers attempt to hide behind TLS encryption.

**Objective:** Analyze actual Dridex traffic from Palo Alto Unit42 PCAP dataset.

**Tools used:** Wireshark, Suricata,MITRE ATT&CK framework.

**Focus areas**:

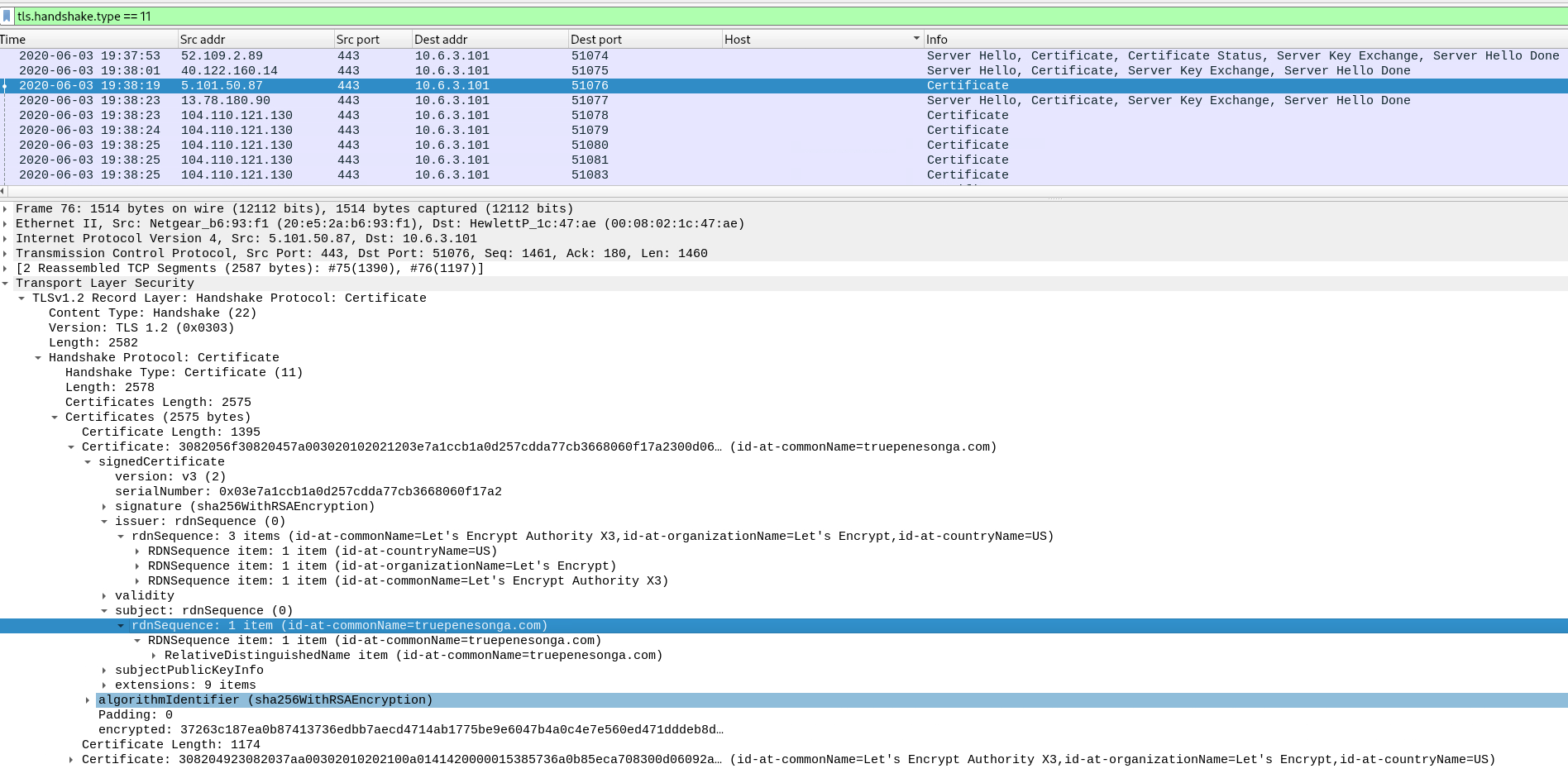
* Identify malicious TLS/SSL traffic patterns.
* Examine suspicious certificates and handshake behavior.
* Write detection rules that do not rely on payload decryption.
* Map observations to adversary TTPs.

**Dataset:** Palo Alto Unit42 Dridex PCAPs (https://github.com/pan-unit42/wireshark-tutorial-Dridex-traffic)

**Methodology:**

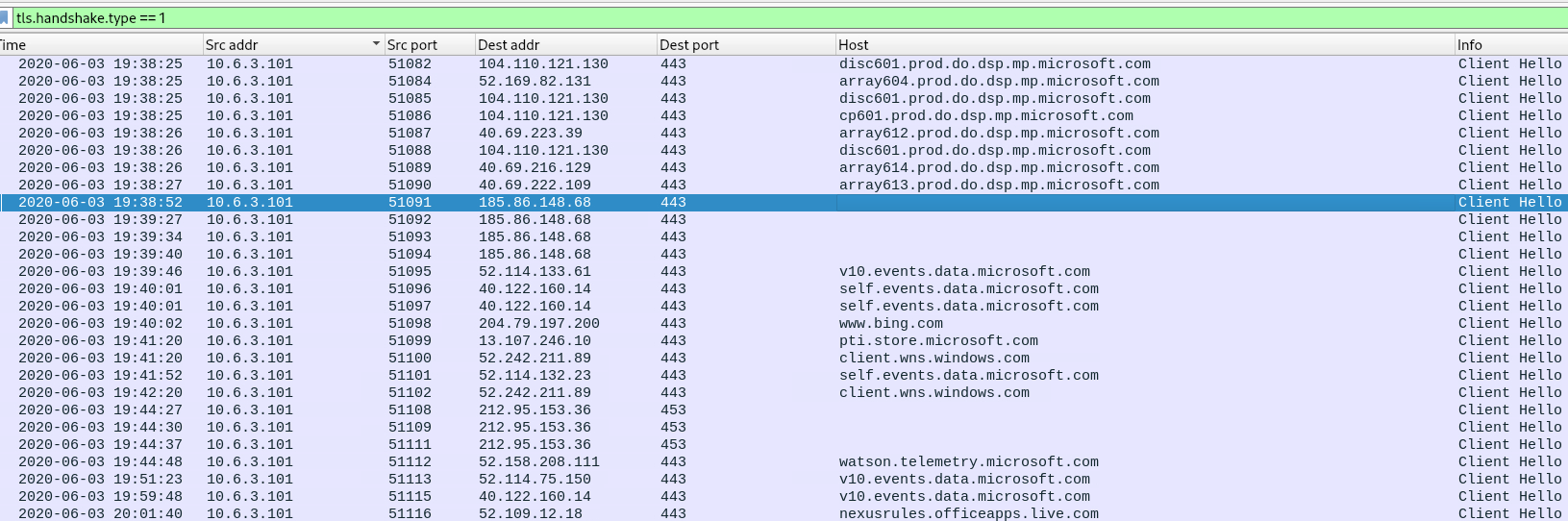
**Wireshark Analysis:**

* Customized layout for HTTP/TLS traffic (exported configuration profile)



Let’S Encrypt CA not malicious/suspicious on it’s own but paired with a known malicious domain here (truepenesonga.com)

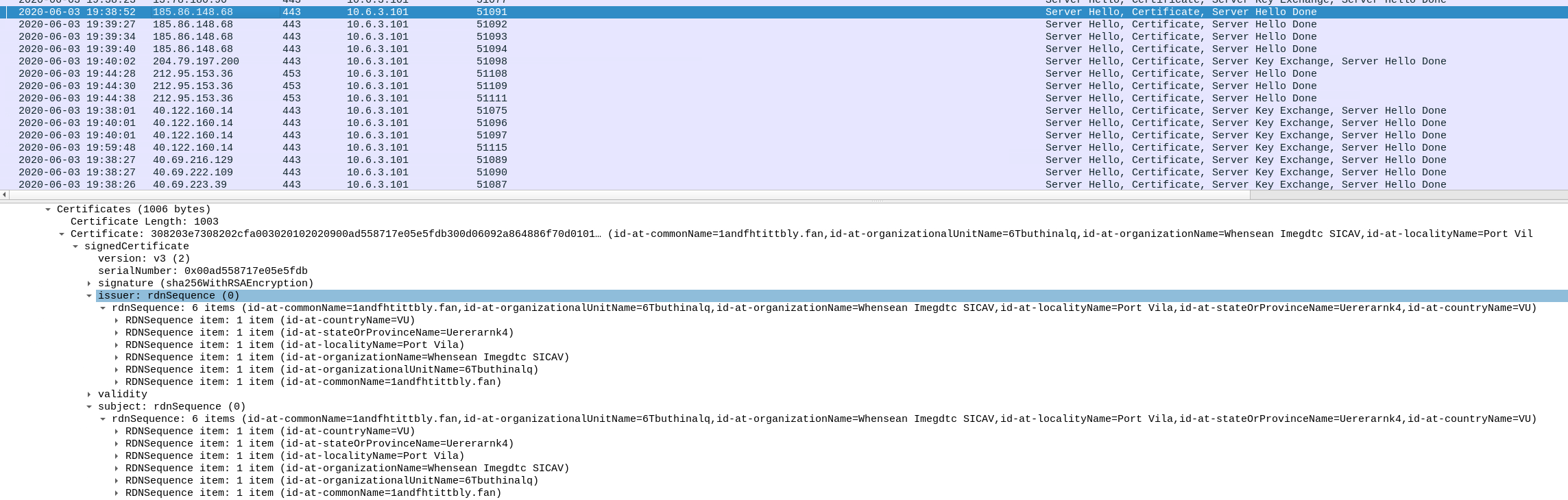
This is likely the Dridex loader being downloaded as the client reaches out to this malicious domain



Subsequently we see the client reaching out to two I addresses that do not have associated domains. This warrants suspicion.

When we examine the certificate issuer and subject data for these domains, we see multiple red flags:

There appear to be multiple fields with random alphanumeric values



**JA3 Fingerprint Analysis**

During analysis of the Dridex infection traffic, I examined the TLS ClientHello messages to extract **JA3 fingerprints**. JA3 is a method of creating a hash based on values in the ClientHello (cipher suites, extensions, elliptic curves, etc.), providing a fingerprint of the TLS client implementation. Malware families often use non-standard or consistent TLS stacks, making JA3 values useful for correlation and detection.

**Observed Values:**

* **3b5074b1b5d032e5620f69f9f700ff0e**  
  – Seen in TLS connections to the suspicious domain truepenesonga[.]com.  
  – Represents the TLS stack used by the initial dropper or installer.
* **51c64c77e60f3980eea90869b68c58a8**  
  – Seen consistently in subsequent TLS connections to the C2 IPs 185.86.148[.]68 and 212.95.153[.]36.  
  – No Server Name Indication (SNI) was present in these ClientHello messages, consistent with malware contacting raw IP addresses.  
  – This JA3 remained stable across multiple C2 endpoints, indicating the same malware TLS implementation.

**Interpretation:**

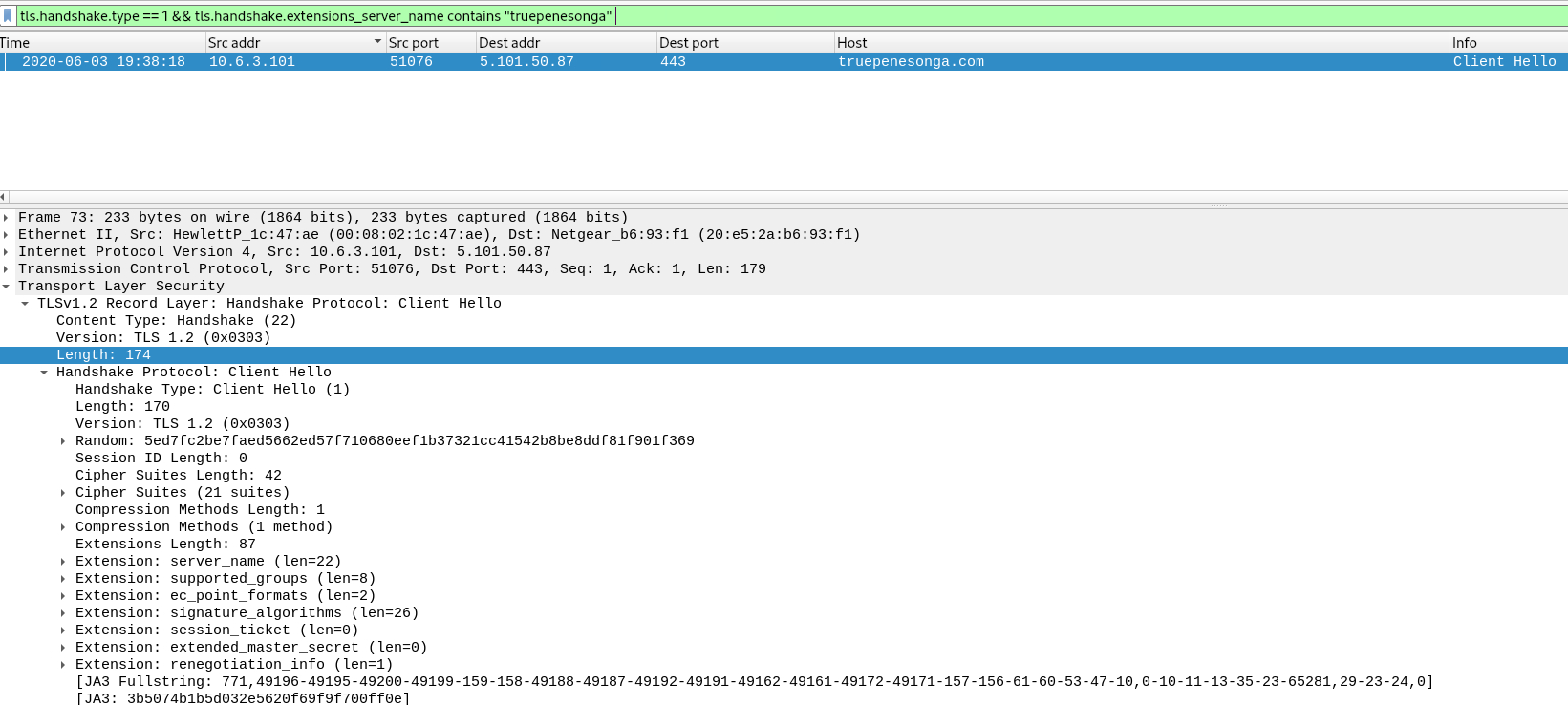
* The host first contacted truepenesonga[.]com using JA3 3b5074…. Immediately afterward, the same host initiated TLS sessions to bare IPs using JA3 51c64c7….
* This pivot from one JA3 to another strongly suggests a **process shift**: the initial installer executed, and the Dridex payload began C2 communication using its own TLS stack.
* The consistency of 51c64c7… across multiple C2 servers further supports attribution, as legitimate applications rarely switch TLS stacks mid-session chain.

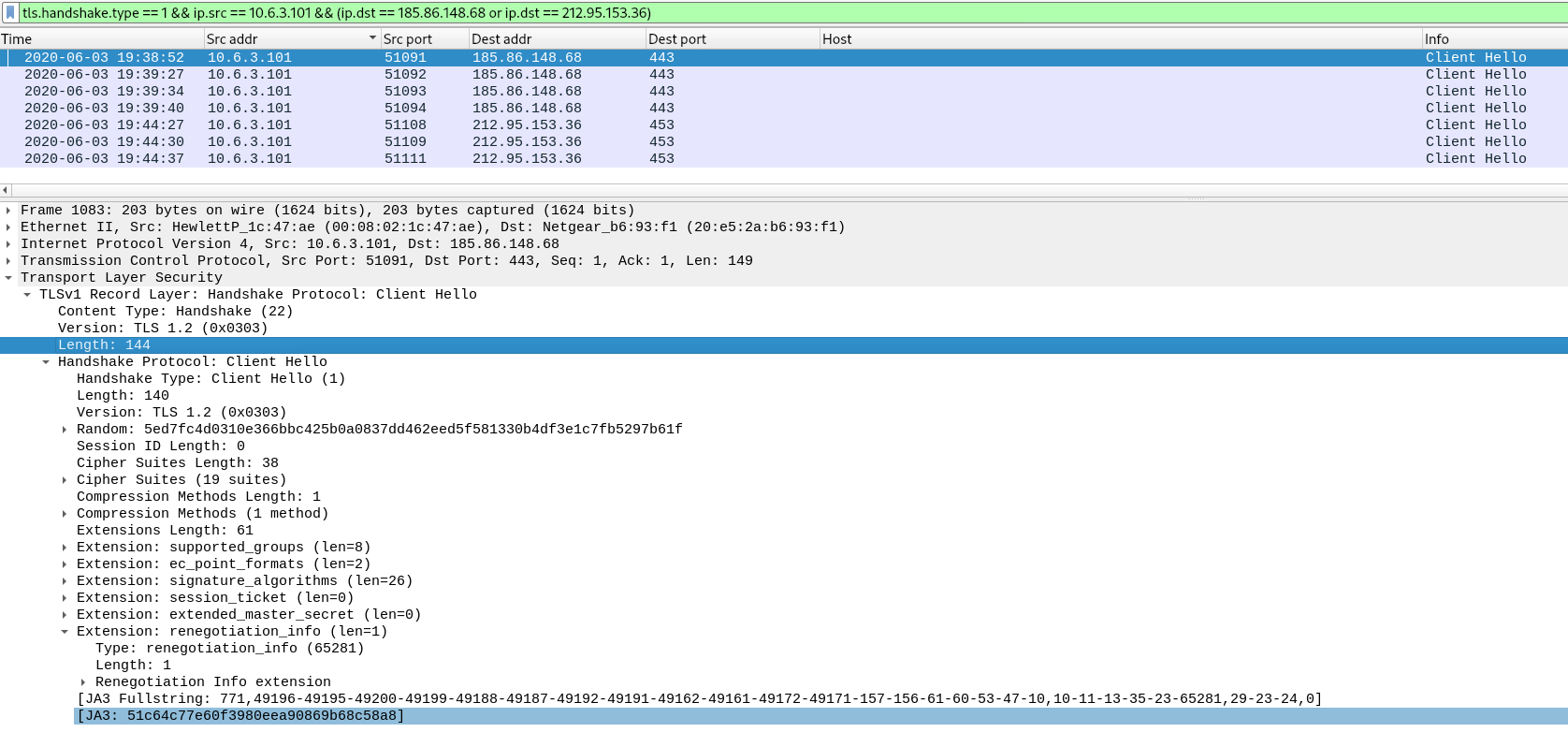
**Detection Opportunity:**

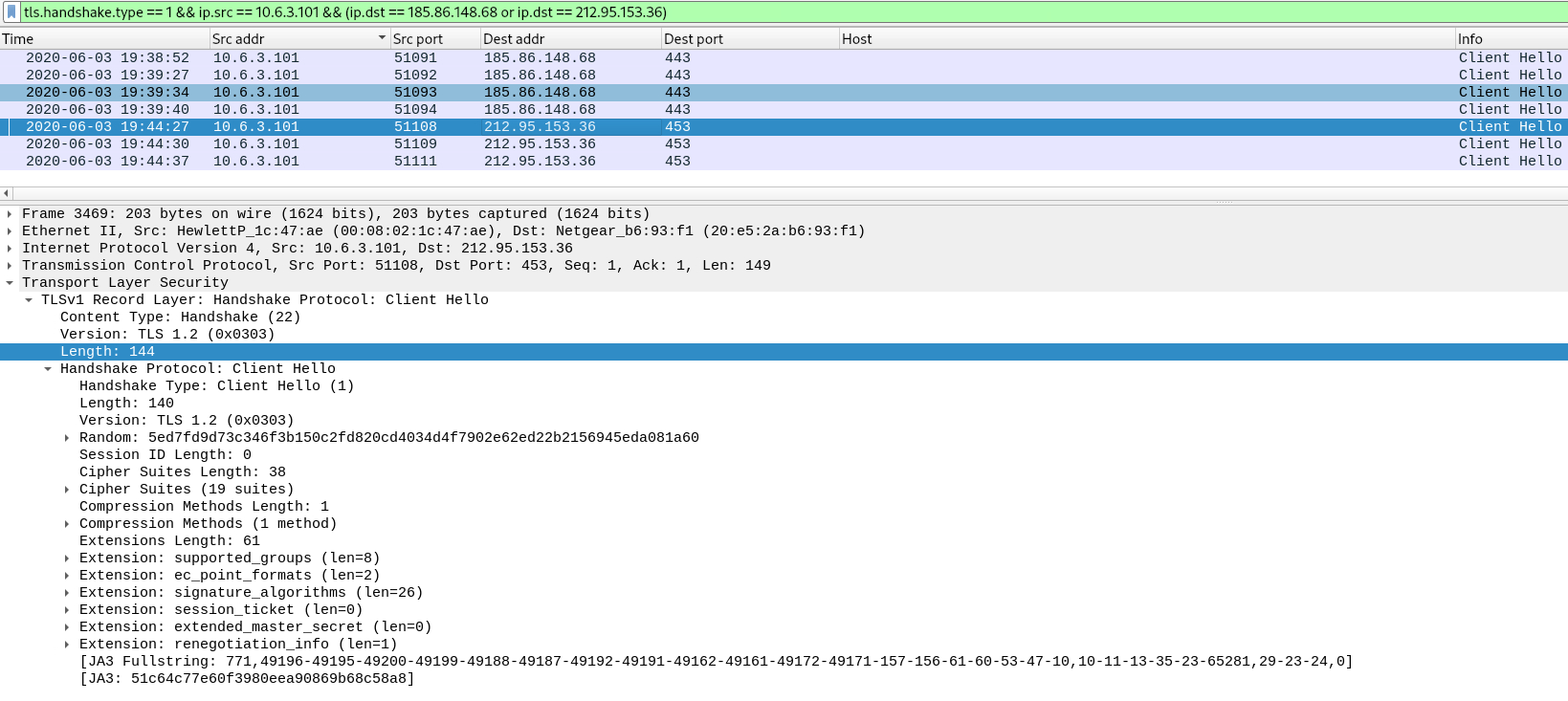
* The JA3 51c64c7… can be used in IDS/IPS (e.g., Suricata) or SIEM correlation rules to detect similar Dridex C2 activity, especially when combined with:
  + Absence of SNI in TLS handshakes
  + Direct TLS connections to bare IPs
  + Anomalous certificate issuer/subject data

**Conclusion:**  
JA3 analysis provided a reliable way to correlate the initial dropper activity with subsequent Dridex C2 traffic. Even though the payloads are encrypted, the change in TLS fingerprint and the reuse of a distinct JA3 across multiple C2 IPs offer high-confidence indicators of malicious activity.

“JA3 represents the TLS ClientHello fingerprint (cipher suites, extensions, groups, formats) hashed to a short digest. It effectively encodes the cryptographic preferences of the client TLS stack. When a host switches from JA3 A → JA3 B in minutes, it usually means a different process or library initiated the TLS connections (e.g., an installer/browser then a malware loader/C2 client).”

“The victim first connected to truepenesonga[.]com using JA3 3b5074b1b5d032e5620f69f9f700ff0e, then pivoted immediately to bare-IP TLS sessions (185.86.148[.]68, 212.95.153[.]36) with JA3 51c64c77e60f3980eea90869b68c58a8, indicating a process shift from the downloader to the Dridex malware C2 client.”  
JA3s are **not globally unique identifiers of malware**, but they are **consistent fingerprints of the TLS stack in use**. When rare JA3s appear in suspicious contexts — especially showing up across multiple C2 servers — they provide high-confidence signals of malicious activity. 

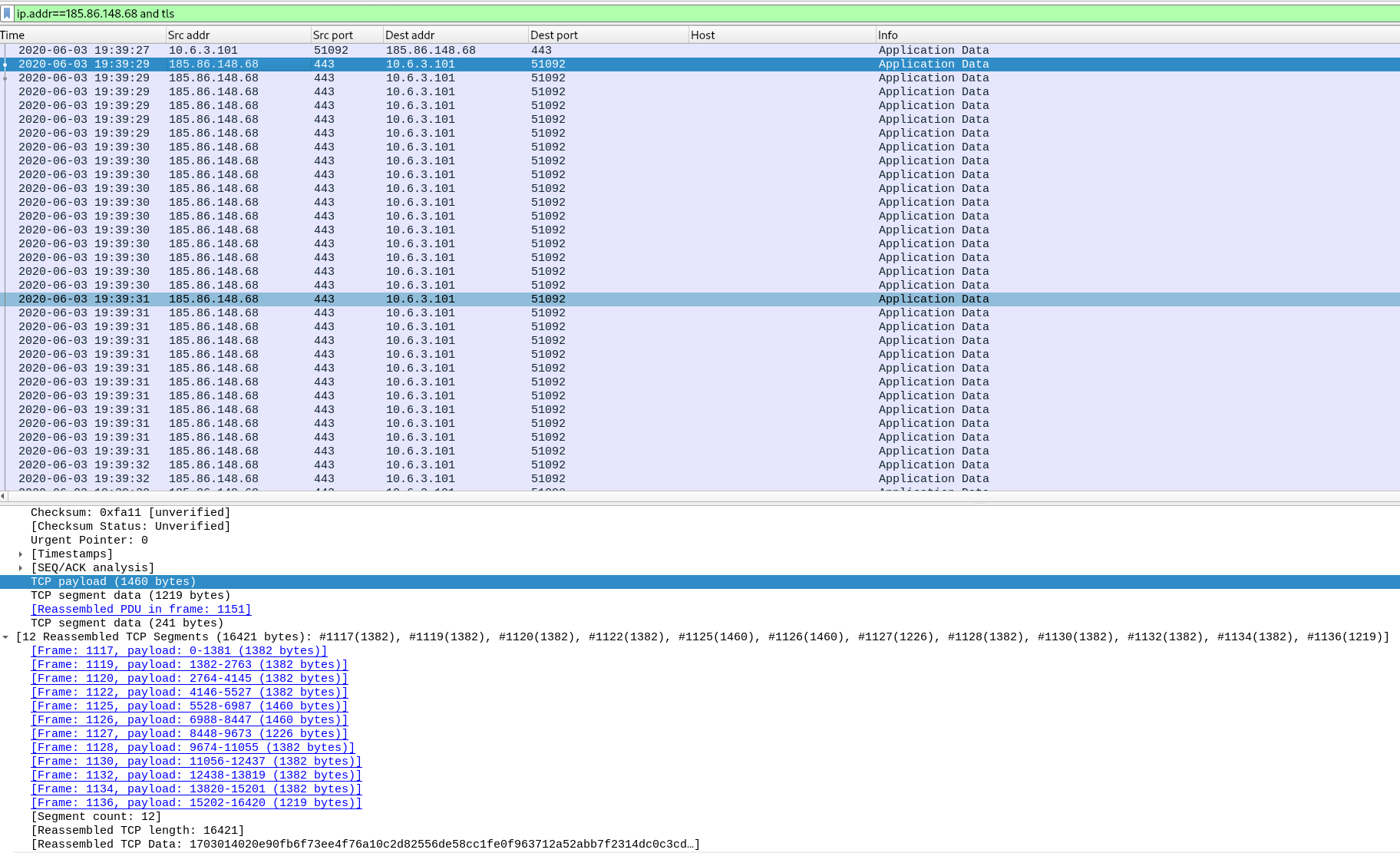




NO SNI OR SAN FOR C2 IPS:

“Post-infection TLS sessions to 185.86.148[.]68 and 212.95.153[.]36 exhibited a consistent JA3 fingerprint (51c64c77e60f3980eea90869b68c58a8), contained no Server Name Indication (SNI), and presented X.509 certificates with no Subject Alternative Name (SAN) and random issuer/subject values — a combination of anomalies strongly indicative of Dridex C2 traffic rather than legitimate browser activity.”

The burst of ~40 packets from 185.86.148.68 within 3 seconds is the result of a single large TLS Application Data record (~16 KB) being segmented by TCP into near-maximum segment size packets, consistent with bulk transfer (e.g., stage download or data exfil), not interactive beaconing.



This burst is followed by another almost immediately from 212.95.153.36. Similarly, this lasts a few seconds and also has a reassembled TCP length of over 16KB. This is almost certainly stage download. Again here, we see the initial beacon/handshake is followed by a server→ client bulk TLS application data transfer (single large record reassembled from near-MSS segments). This behavior is consistent with a stager fetching a larger stage (DLL/executable/payload) rather than small interactive C2 commands.

