A Structural Approach to Modeling Encrypted Connections

Anthony Kasza

FloCon 2020
JANUARY 6-9, 2020 | SAVANNAH, GA

corelight

Overview

- Sequence of Lengths (SOL) Background
- Zeek Background
- Applications of SOL using Zeek (pictures and graphs)
  - SSH
  - RDP
  - SSL
- Future work

Sequence of Lengths (SOL) Background

- Inspired by
  - SPLT feature in *Deciphering Malware's use of TLS (without Decryption)* [1]
  - Implementation of feature extraction - Joy [2]
- Generalizes across protocols - works on encrypted ones, too
  - Lengths of data
  - Order of data
  - Direction of data
- Combines well with rule based expert systems - maybe ML, too

Zeek Background

Zeek : "a powerful network analysis framework that is much different from the typical IDS you may know." [6]

- SDN approach to network security monitoring
  - Event-driven scripting language
- Protocol parsing
  - Analyzers detach once encryption begins to save resources
  - SOL can be used instead of full parsing
- Logging, file extraction, intel matching, and more

Applications of SOL using Zeek

Within Zeek, sequences can be represented as vectors

vector : "An associate array that maps from one set of values to another… its indices are non-negative integers, starting from zero"[4]

- Originator message lengths are positive
- Responder message lengths are negative
- Order is preserved using vector indices

Applications of SOL using Zeek

```
v: vector of int = {24, -24, 48, -12, 36, -42, 24, -124, -12, 96, -48, -48, 48}
```

Applications of SOL using Zeek

```
v: vector of int = {24, -24, 48, -12, 36, -42, 24, -124, -12, 96, -48, -48, 48}
```

First message length

First originator message length

Applications of SOL using Zeek

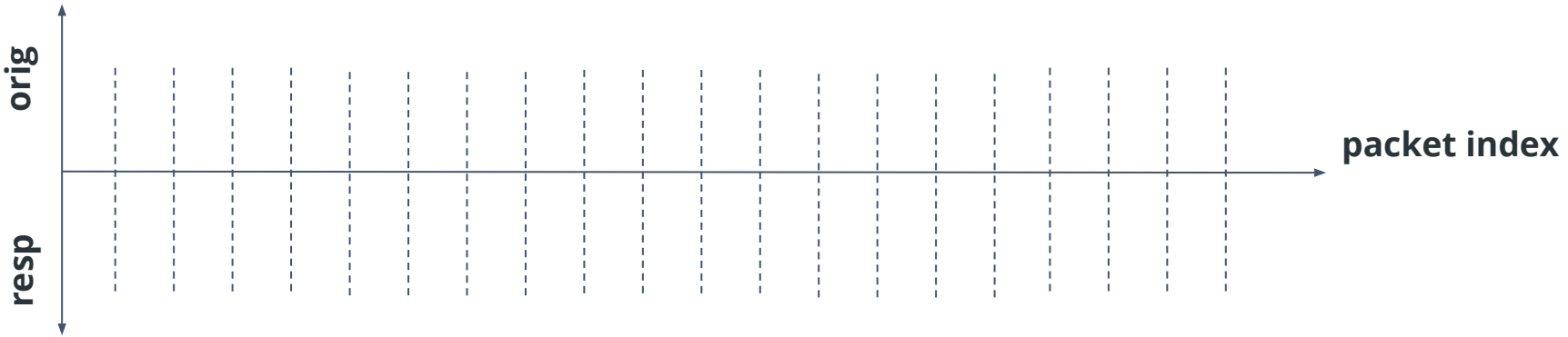
```
v: vector of int = {24, -24, 48, -12, 36, -42, 24, -124, -12, 96, -48, -48, 48}
```

Second message length

First responder message length

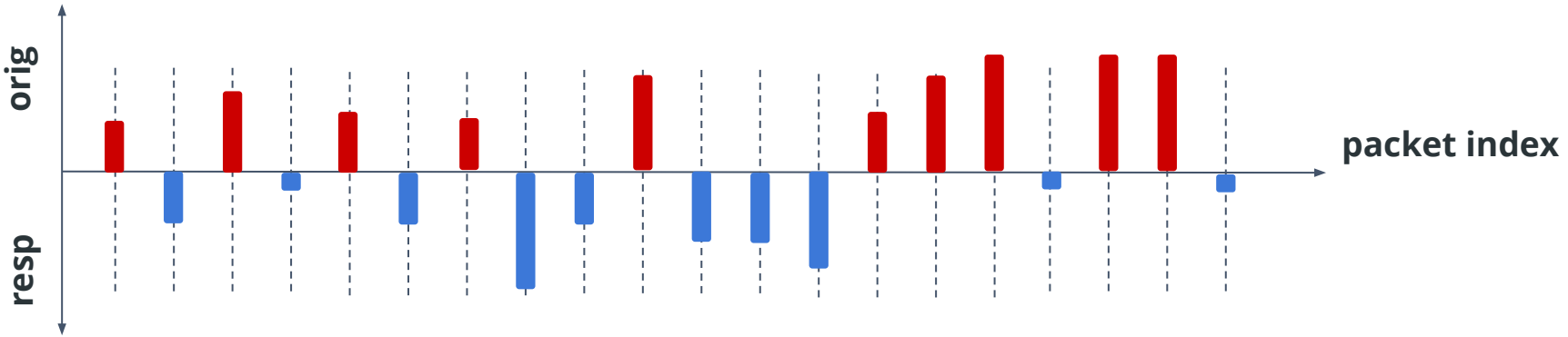
Applications of SOL using Zeek

```
v: vector of int = {24, -24, 48, -12, 36, -42, 24, -124, -12, 96, -48, -48, 48}
```



Applications of SOL using Zeek

```
v: vector of int = {24, -24, 48, -12, 36, -42, 24, -124, -12, 96, -48, -48, 48}
```



Applications of SOL using Zeek

Useful vector operations for rule building:

- Index slicing (heads and tails)
- Summary statistics (max, min, mean, range, etc)
- "Runs"
  - Increasing
  - Decreasing
  - Repeating
- PCR [7]
- Find first, second, third occurrence of...
  - Positive
  - Negative
  - "run"

Applications of SOL using Zeek: SSH

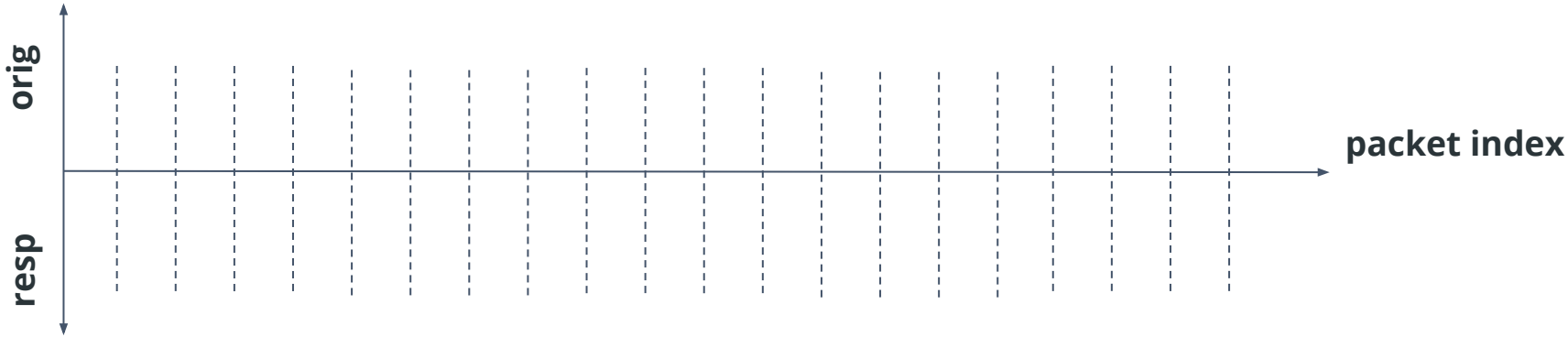
- SSH consists of 3 sub-protocols
  - Transport
  - Authentication
  - Connection
- SSH cleartext handshake/negotiations
- SSH PDUs are called messages
  - One or more messages are formatted in an SSH "packet" struct

[3]

Applications of SOL using Zeek: SSH

Cleartext (negotiations)

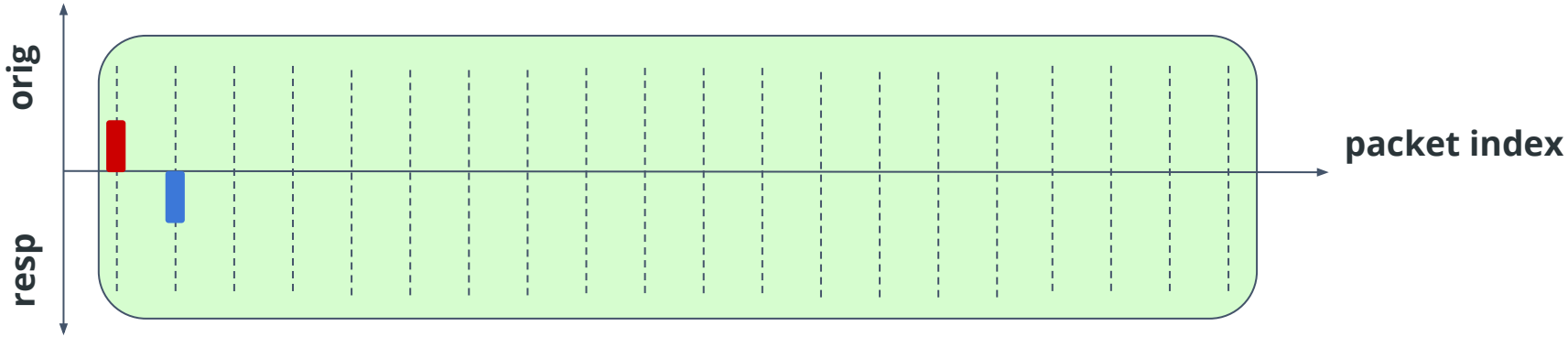
```
v: vector of int
```



Applications of SOL using Zeek: SSH

Cleartext (negotiations) | Transport (encryption)

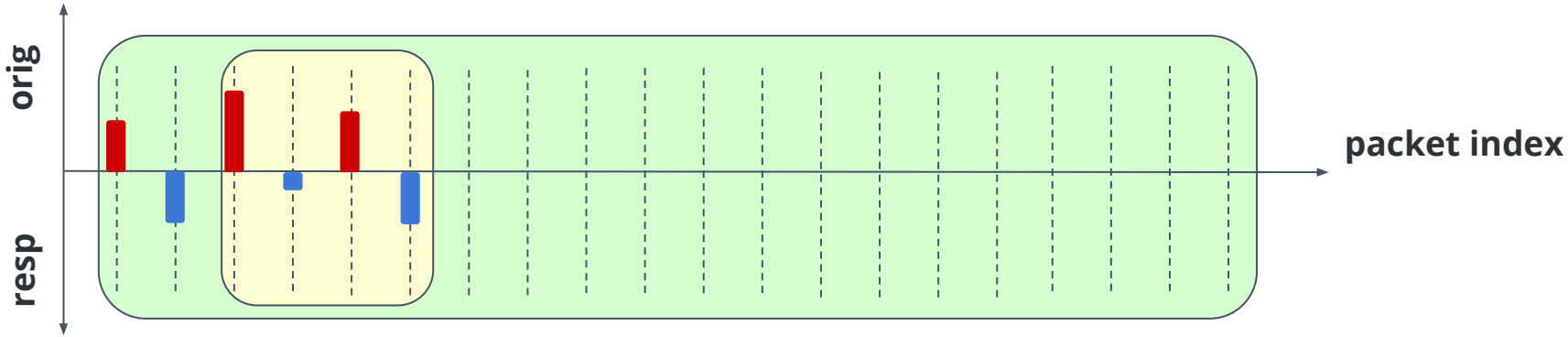
```
v: vector of int = {24, -24,
```



Applications of SOL using Zeek: SSH

Cleartext (negotiations) | Transport (encryption) | Authentication

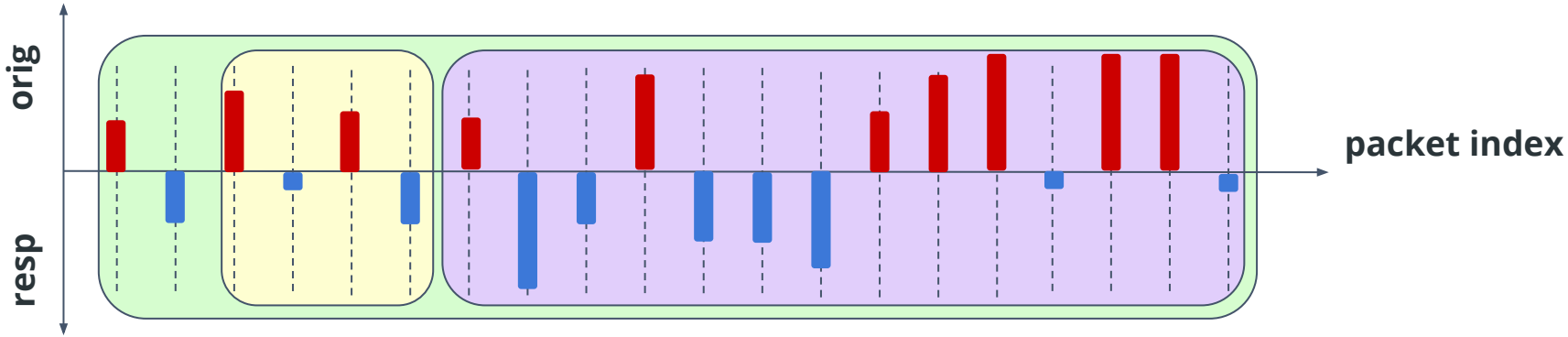
```
v: vector of int = {24, -24, 48, -12, 36, -42,
```



Applications of SOL using Zeek: SSH

Cleartext (negotiations) | Transport (encryption) | Authentication | Connection

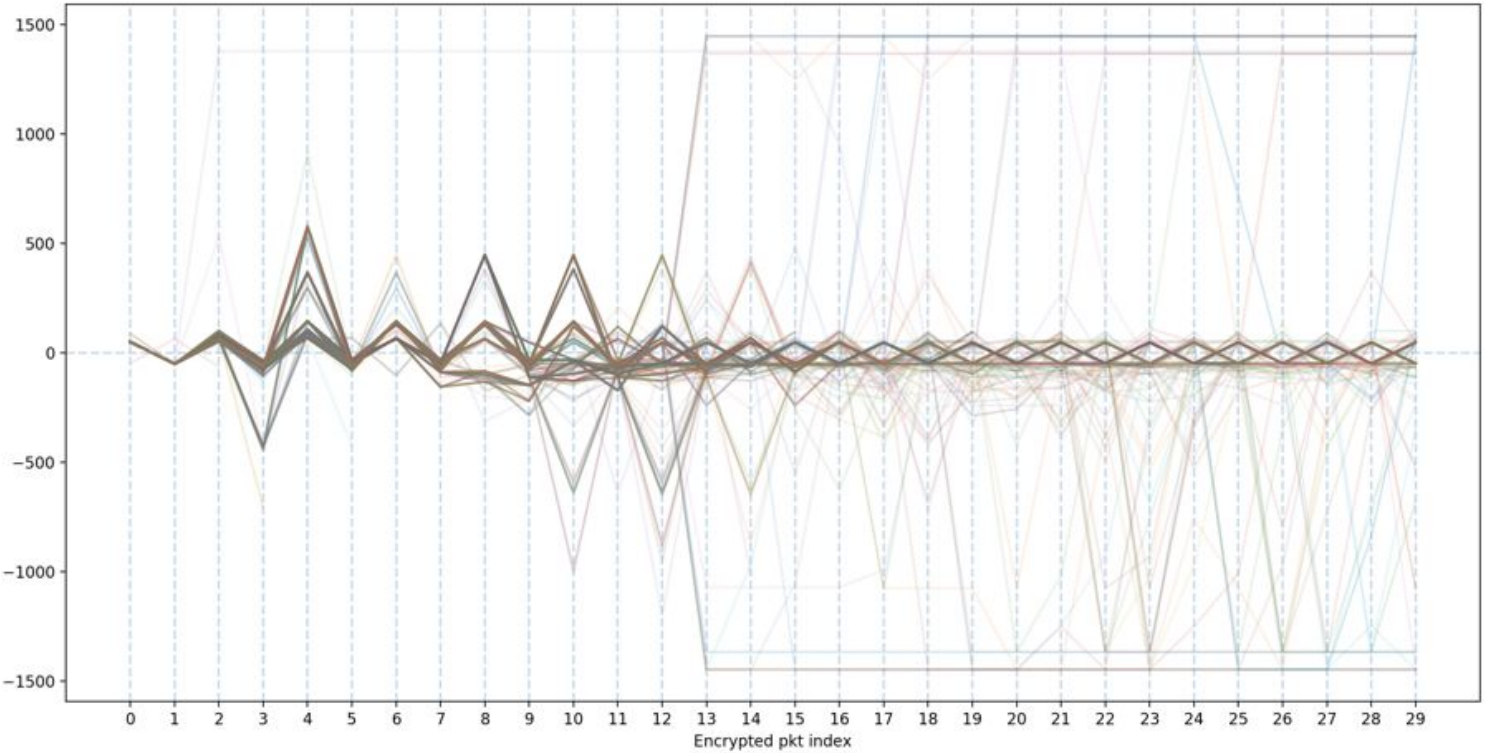
```
v: vector of int = {24, -24, 48, -12, 36, -42, 24, -124, -12, 96, -48, -48, 48}
```



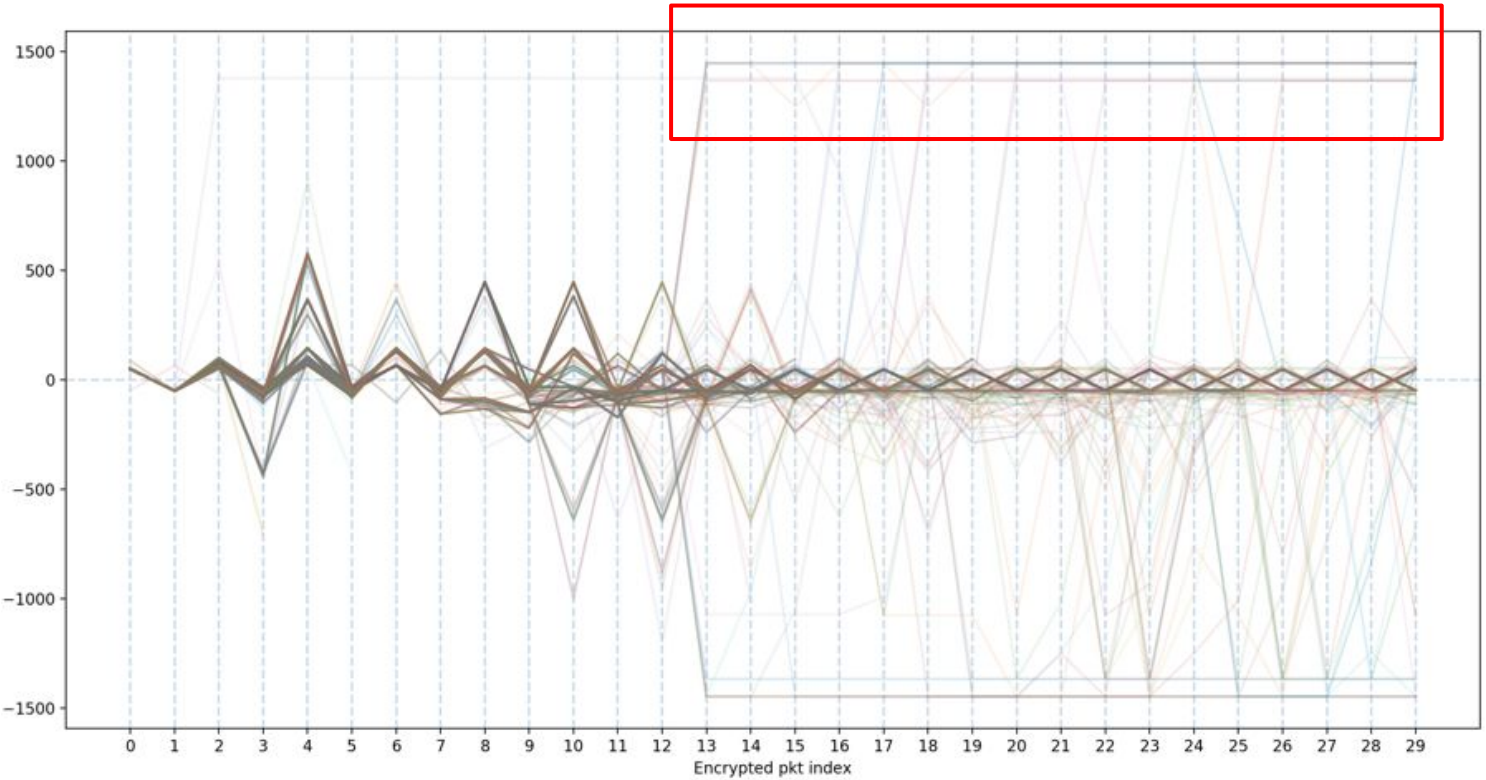
Applications of SOL using Zeek: SSH

- Authentication
  - Bruteforce guessing
  - Interactive vs automated authentication
    - Password vs pubkey
- Mode of use
  - File transfers vs keystrokes
    - Timing profiles
    - Counting keystrokes
      - Root password lengths, oh my!
- State machine transitions
  - Authentication bypass exploits
    - *do not pass authentication, do not collect $200*
  - Protocol is SSH in the clear and something different once encrypted

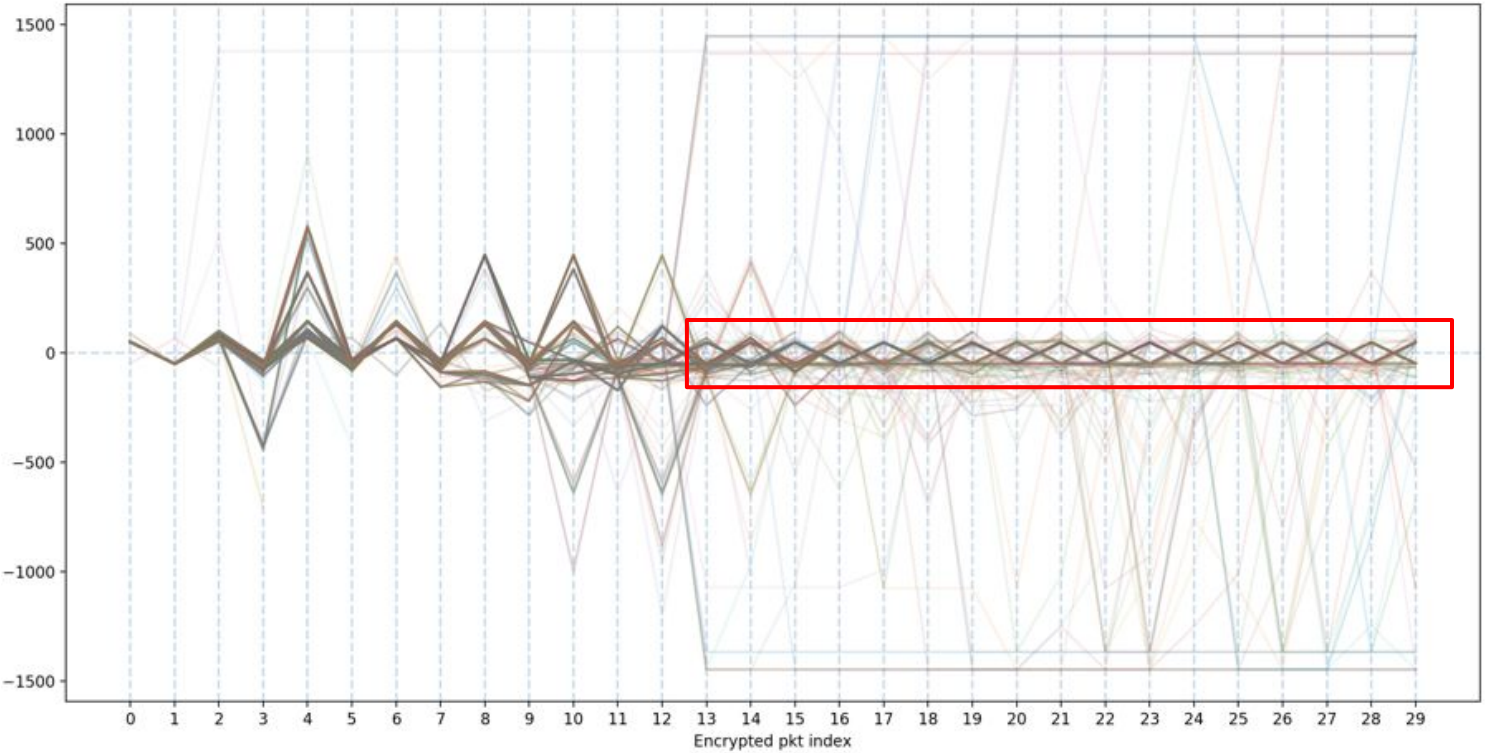
Applications of SOL using Zeek: SSH



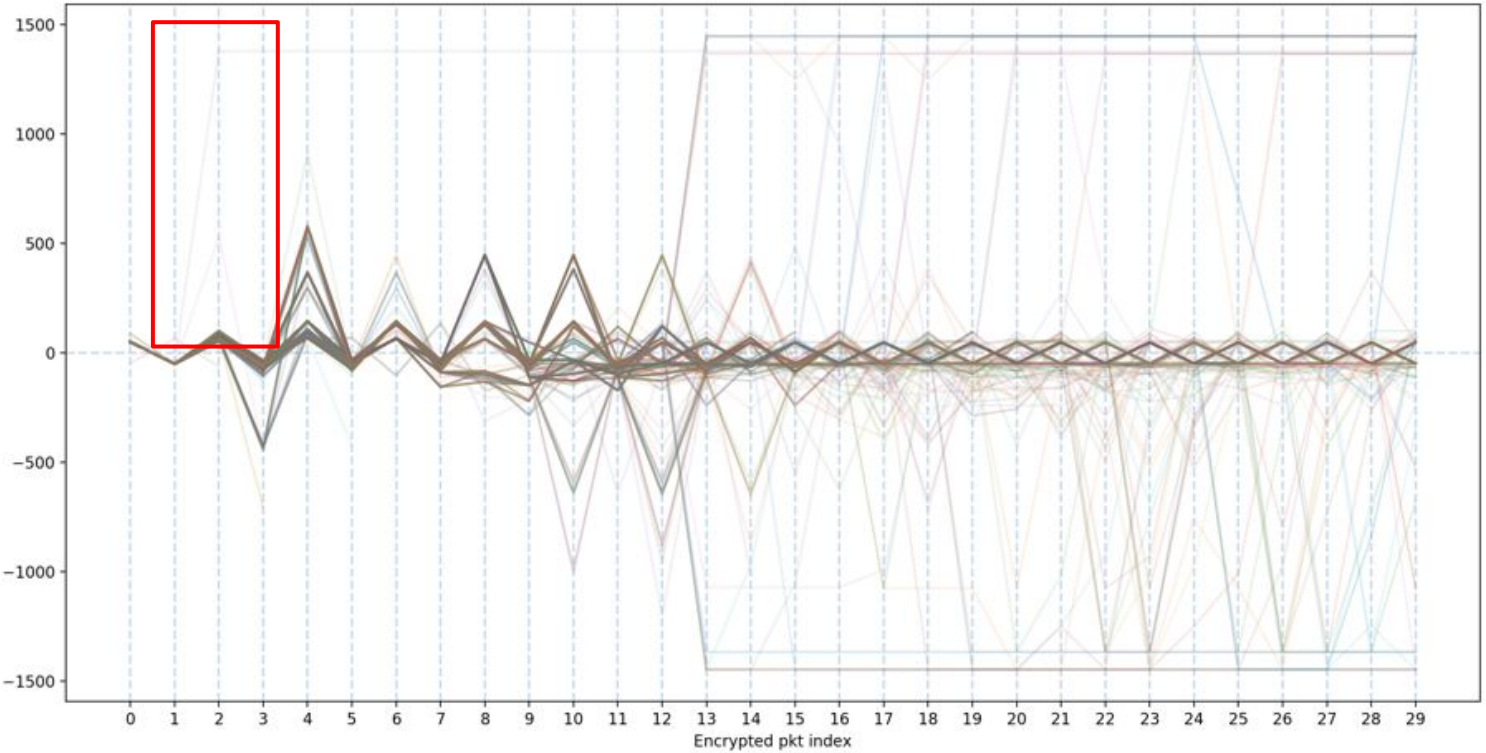
Applications of SOL using Zeek: SSH



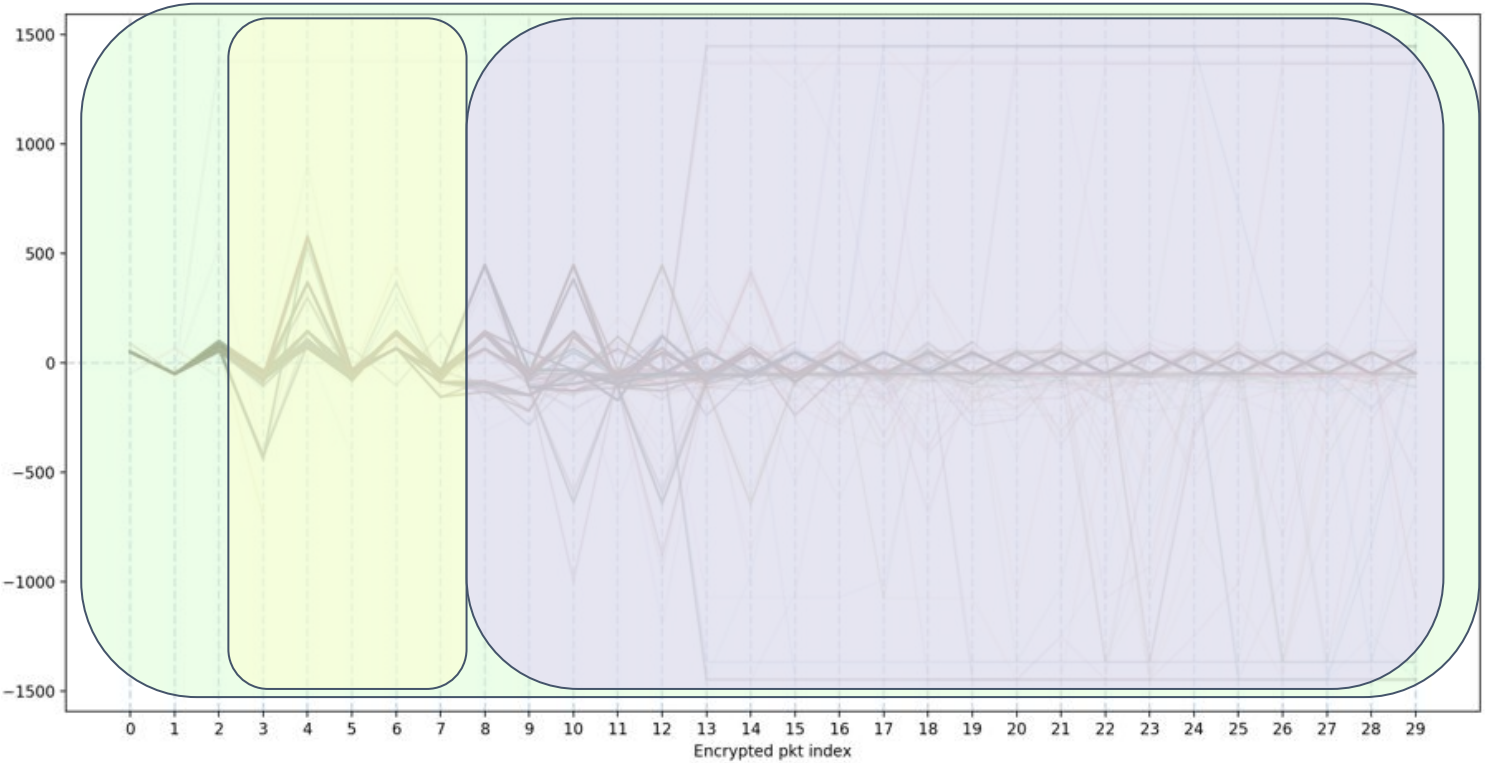
Applications of SOL using Zeek: SSH



Applications of SOL using Zeek: SSH



Applications of SOL using Zeek: SSH



Applications of SOL using Zeek: RDP

- non-NLA
  - native crypto
  - Unauthenticated clients can do things
  - Channels opened before authentication + encryption
    - Can monitor for `MS_T120` (Bluekeep) channel opens
- NLA
  - TLS
  - Authenticate client before anything else
  - Channels opened after authentication + encryption
    - Bluekeep exploits
      - Requires valid creds
      - Occurs after encryption begins

Applications of SOL using Zeek: RDP Connection Sequence

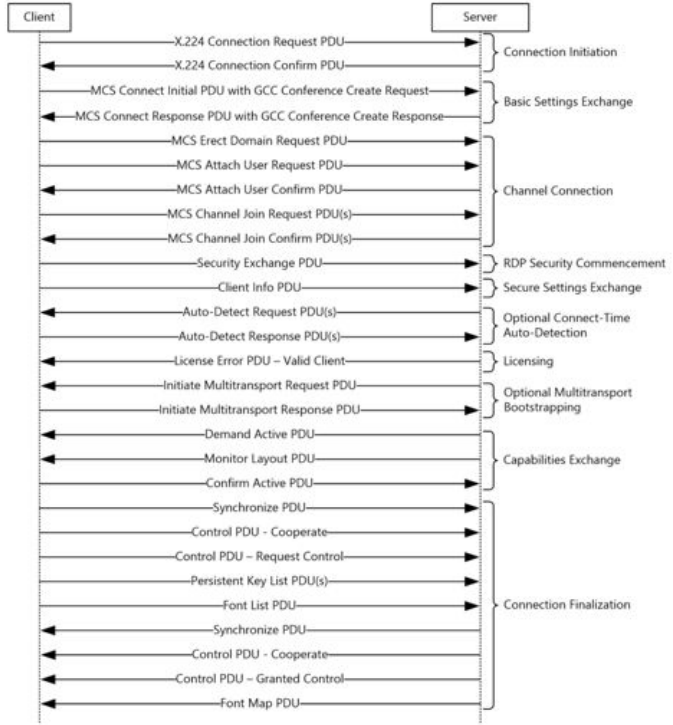


Figure 1: Remote Desktop Protocol (RDP) connection sequence

[5]

Applications of SOL using Zeek: RDP Connection Sequence

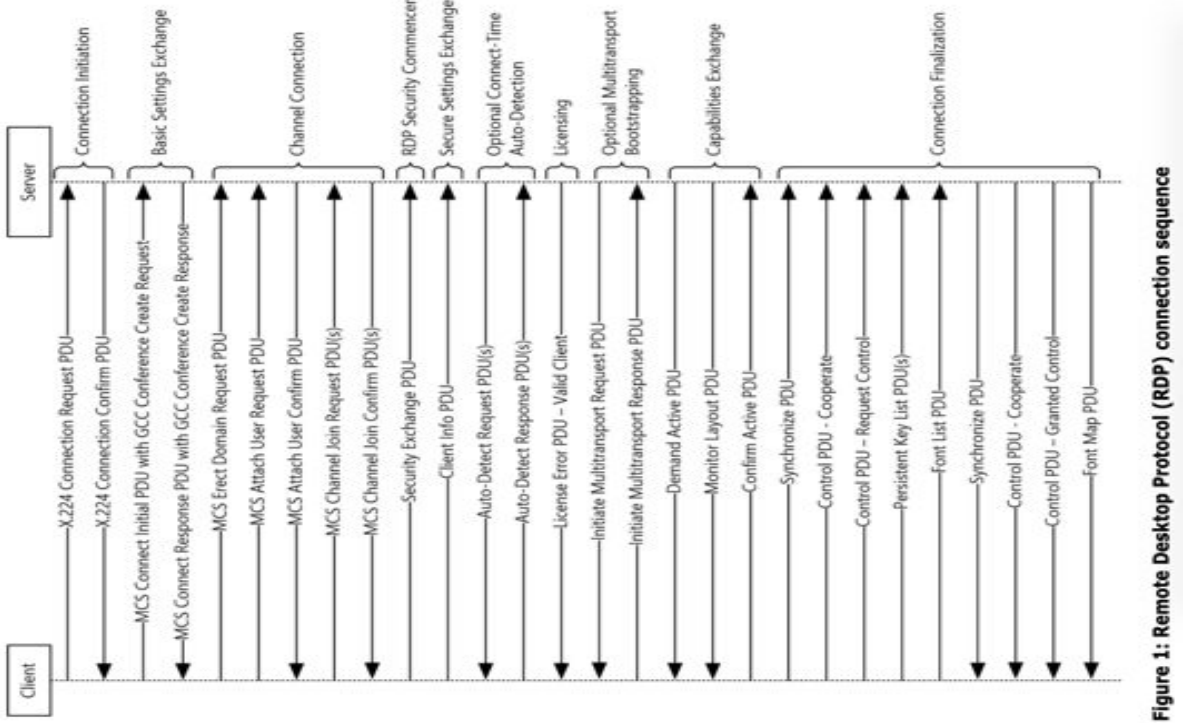


Figure 1: Remote Desktop Protocol (RDP) connection sequence

Applications of SOL using Zeek: RDP Connection Sequence

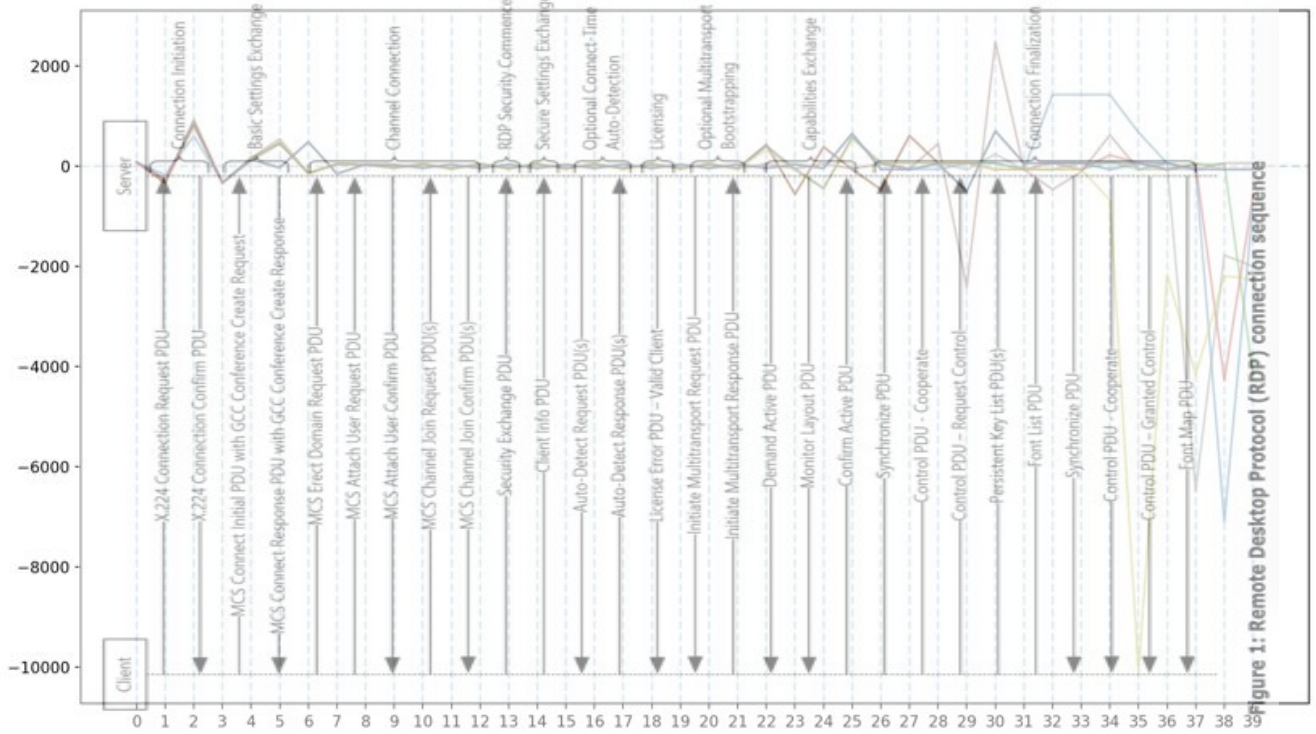
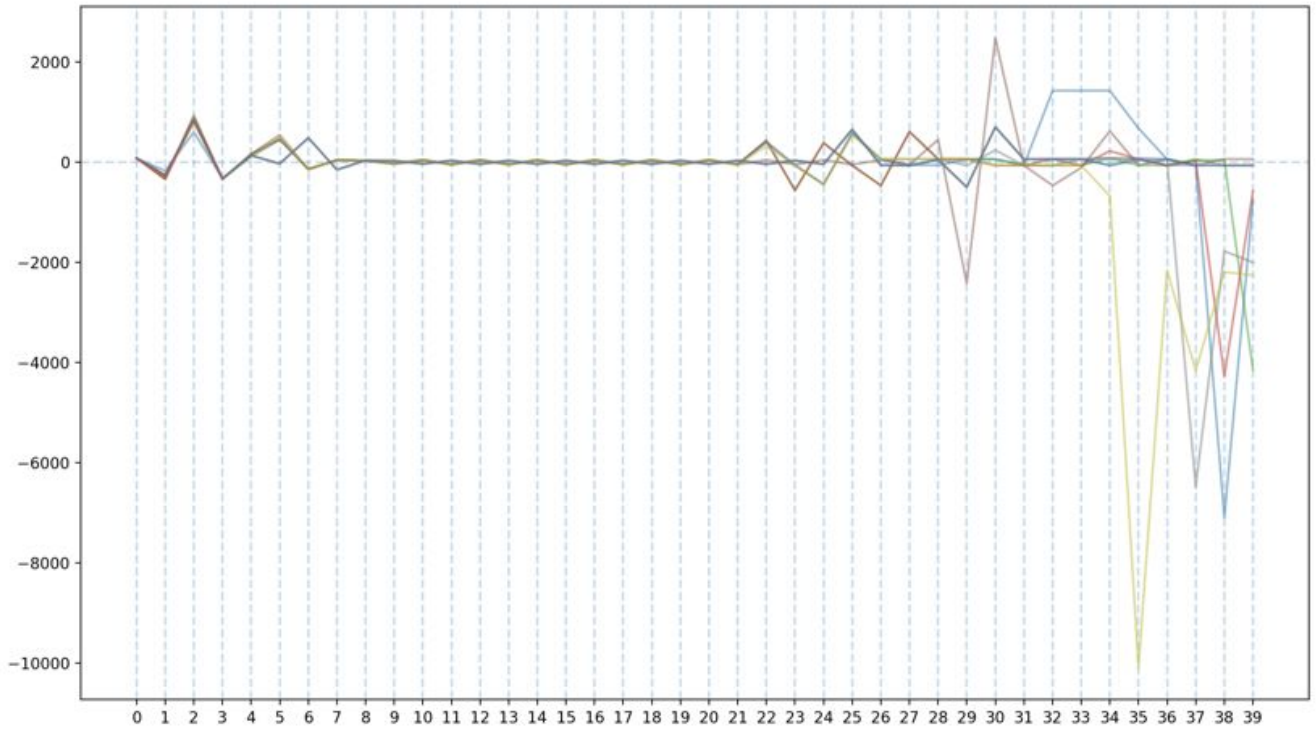
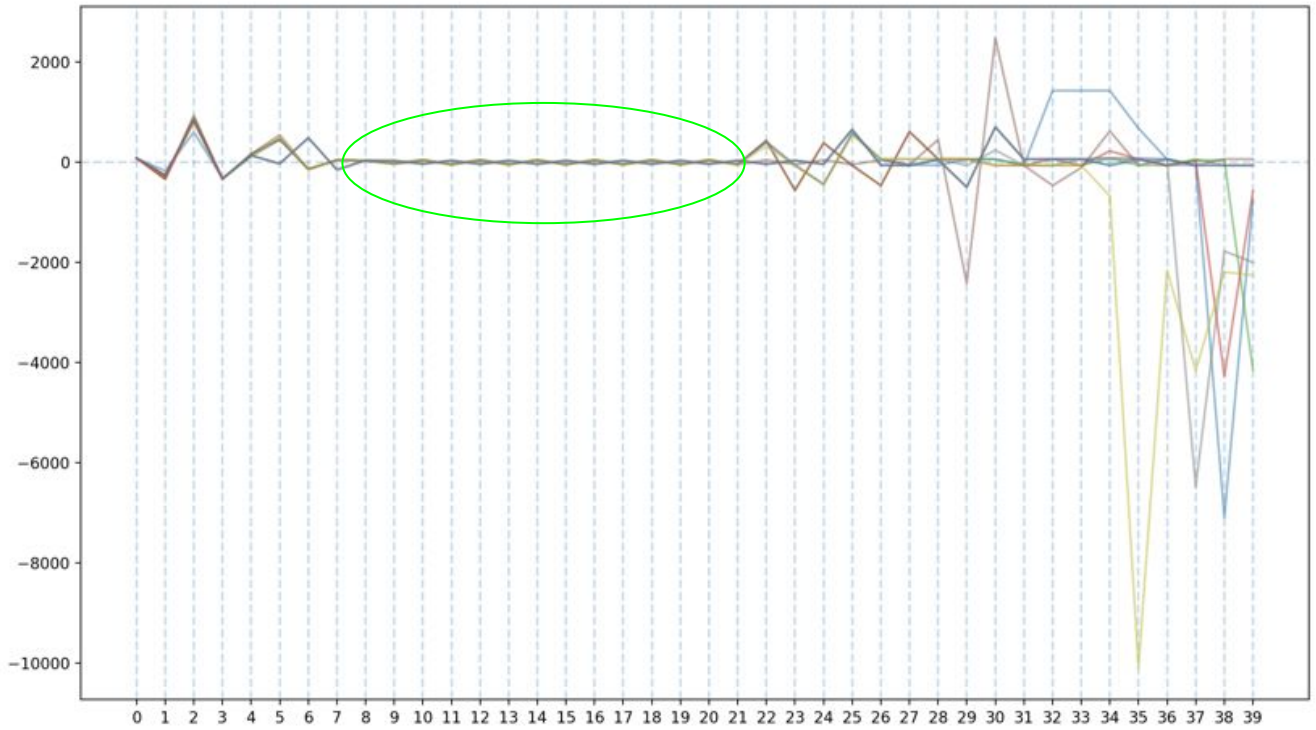


Figure 1: Remote Desktop Protocol (RDP) connection sequence

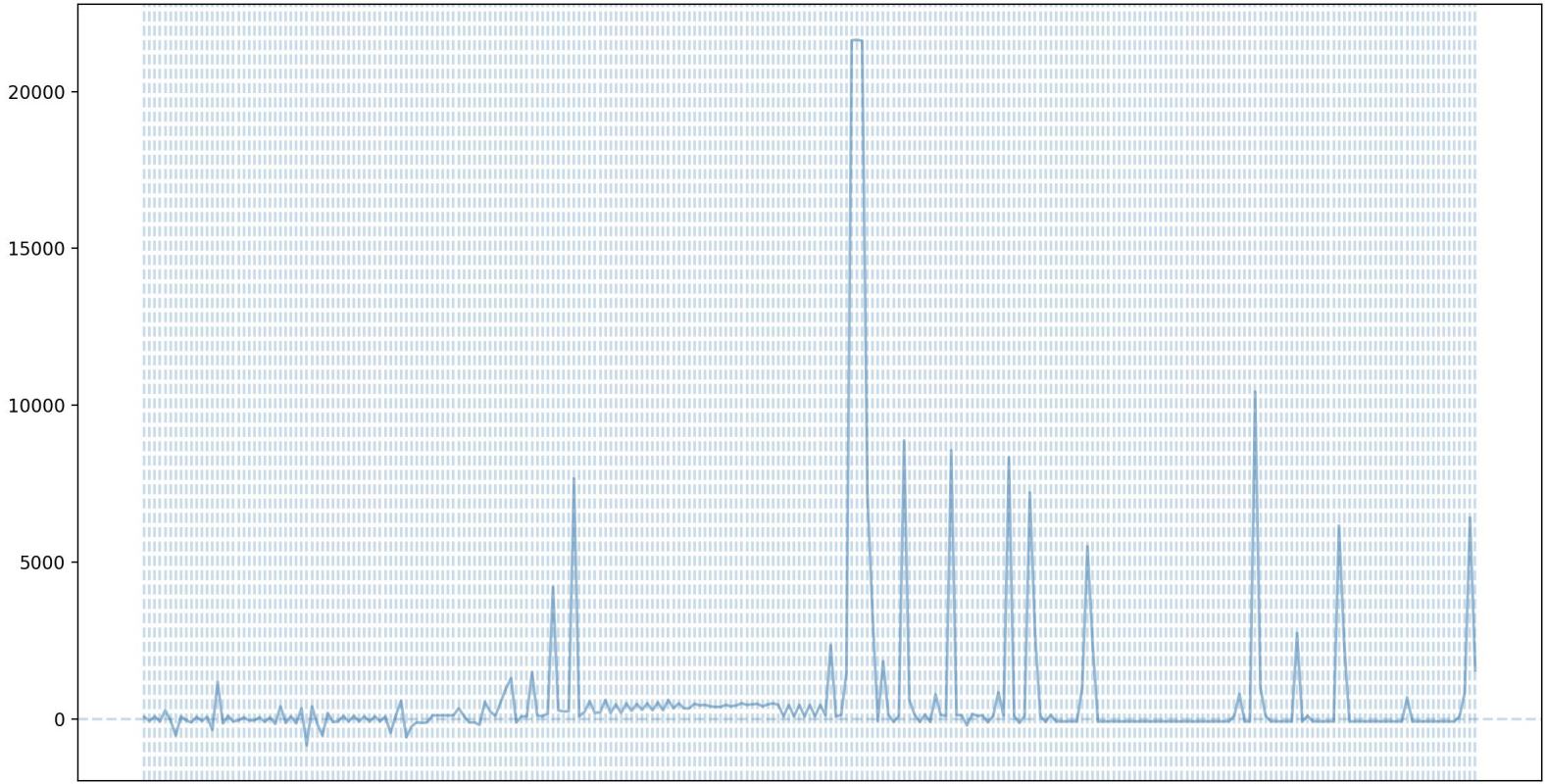
Applications of SOL using Zeek: RDP (NLA - TLS, ~10)



Applications of SOL using Zeek: RDP (NLA - TLS, ~10)

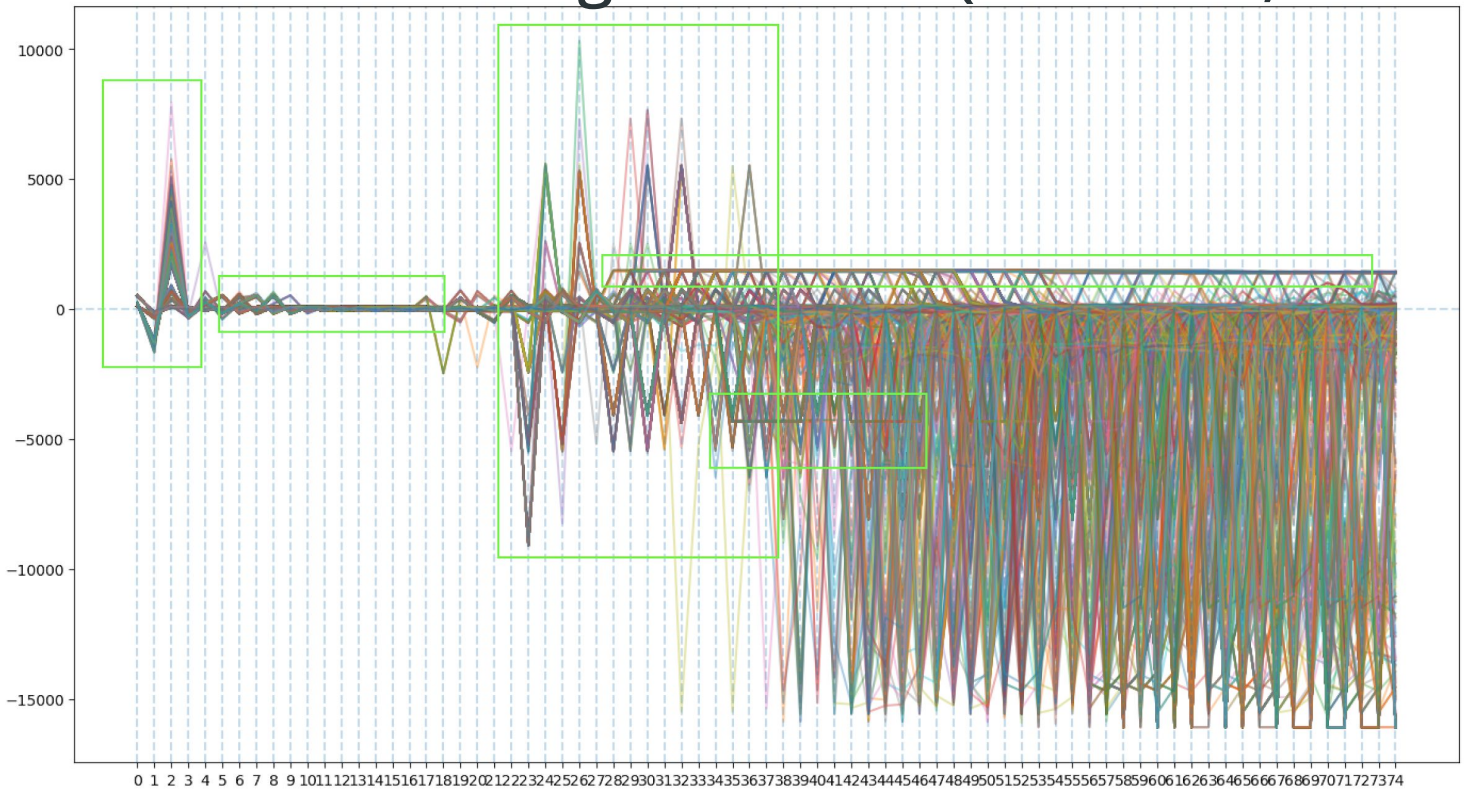


Applications of SOL using Zeek: RDP over SSH (NLA - TLS, 1)

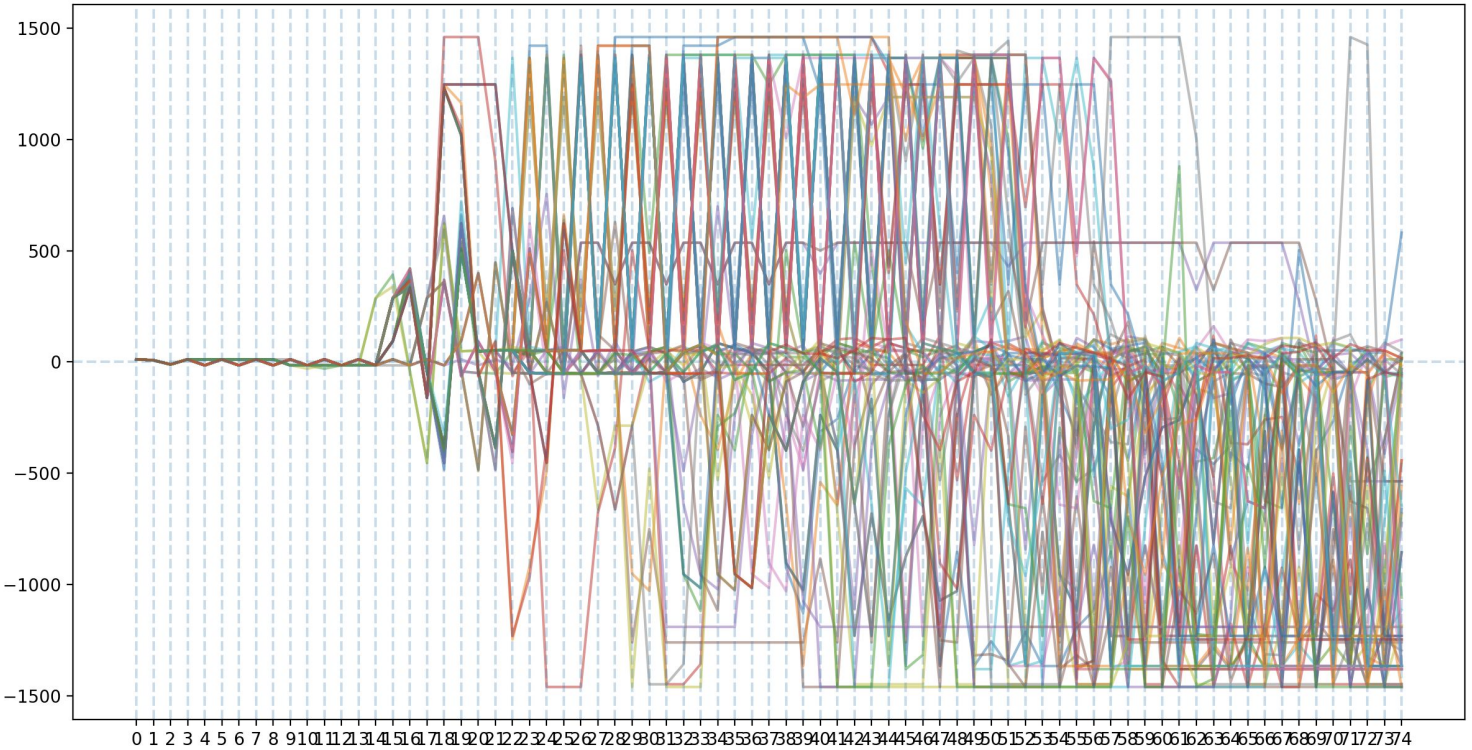


See also: [8]

Applications of SOL using Zeek: RDP (NLA - TLS, ~2300)



Applications of SOL using Zeek: RDP (non-NLA - native, ~70)



Applications of SOL using Zeek: TLS

SPLT can be used to identify malware communications over TLS.

Can it conceptually be applied to identify other application layer protocols?

Applications of SOL using Zeek: TLS

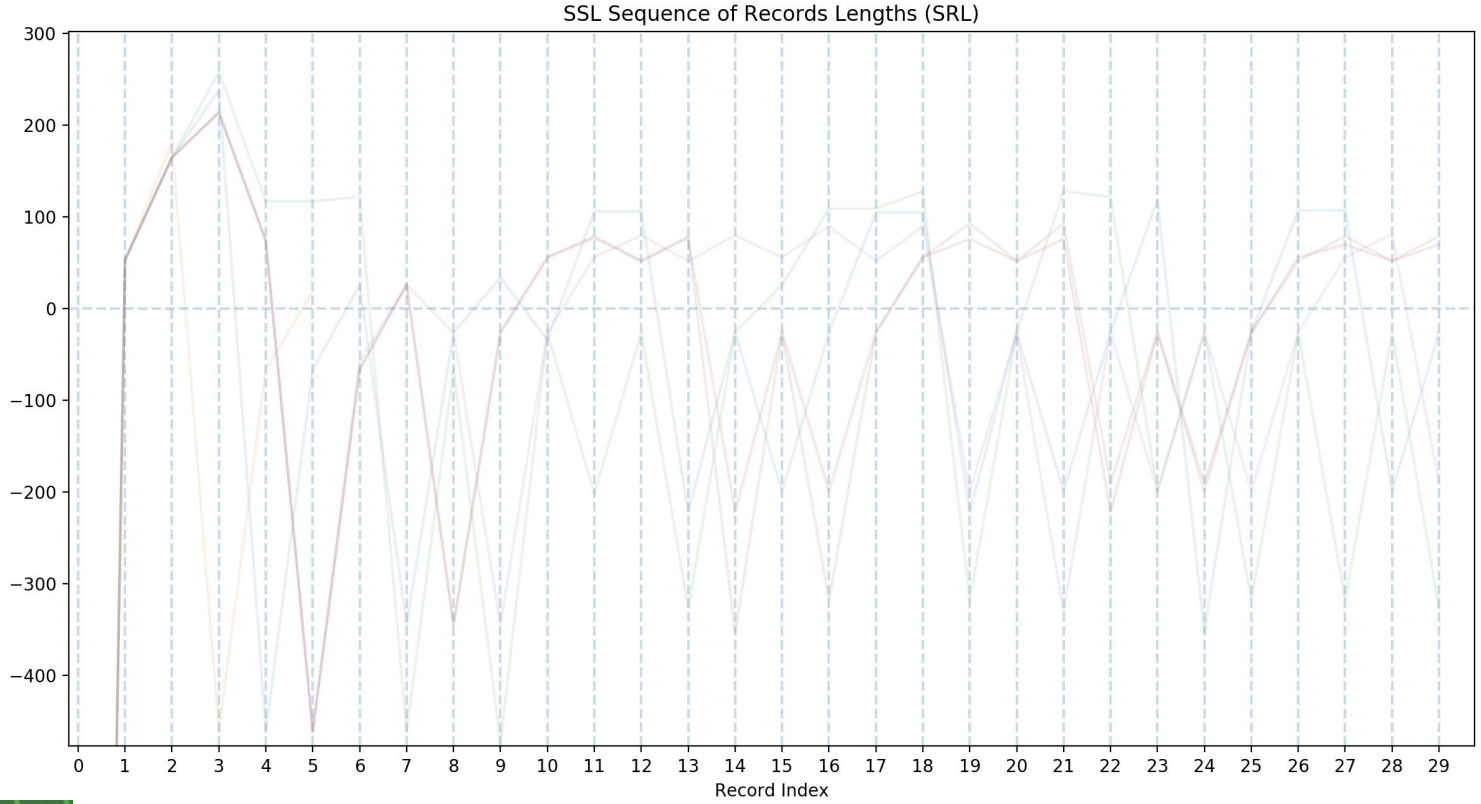
SPLT can be used to identify malware communications over TLS.

Can it conceptually be applied to identify other application layer protocols?

I think so.

DNS has intrinsic size ceilings/floors and expected PCRs [7]

Applications of SOL using Zeek: TLS (DoH POSTs and GETs)



[9]

Future Work

- Apply LSTM RNNs to investigate application layer protocols' SOLs
- Incorporate timing deltas (sequence of times/deltas)
  - Timing of specific pkts or states of a protocol can be insightful
- Generically identifying TCP proxies:
  - Align sequences of two connections (within a time window)
  - If one sequence is a multiple of the other, it may be a tunnel

References and Resources

1. https://arxiv.org/pdf/1607.01639.pdf
2. https://github.com/cisco/joy
3. https://corelight.blog/2019/05/07/how-zeek-can-provide-insights-despite-encrypted-communications/
4. https://docs.zeek.org/en/stable/script-reference/types.html#type-vector
5. https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdpbcgr/023f1e69-cfe8-4ee6-9ee0-7e759fb4e4ee
6. https://www.zeek.org/
7. https://qosient.com/argus/presentations/Argus.FloCon.2014.PCR.Presentation.pdf
8. https://www.fireeye.com/blog/threat-research/2019/01/bypassing-network-restrictions-through-rdp-tunneling.html
9. https://isc.sans.edu/forums/diary/Is+it+Possible+to+Identify+DNS+over+HTTPs+Without+Decrypting+TLS/25616/