Poster:
A Monte Carlo Ensemble Approach to Automatically Identifying Keywords in Binary Message Formats

Jared Chandler
Tufts University
jared.chandler@tufts.edu

***Abstract*—Automatic reverse engineering of unknown binary message formats is a challenging problem. We propose an automatic technique to identify keyword fields used to indicate the serialization format of individual binary messages. Our approach leverages an ensemble of six methods, each tailored to a different property of binary message, coupled with a Monte Carlo technique.**

I. Motivation

Reverse engineering message formats from static network traces is a difficult and time consuming security task [8], [10], [12], critical for a variety of purposes: bug-finding via fuzz testing, automatic exploit generation, understanding the communications of hostile systems, and recovering specifications that are proprietary or have been lost. In prior work, researchers have used message reverse engineering techniques to gain insight into the behavior of malware [1], [6], [7] and manipulate botnets during mitigation efforts [4].

Protocol reverse engineering is characterized by a pipeline with multiple steps. These steps include collecting data, clustering messages by format, inferring a state machine describing how messages are exchanged, and finally inferring semantics for each format. The ambiguous nature of binary data makes such reverse engineering difficult.

Today, we rely on human experts to manually reverse engineer the protocol used to communicate. While automatic methods have been proposed for different aspects of protocol reverse engineering [9], [3], [15], [13], [2], the gold standard remains a human expert. Access to source-code, compiled programs, and physical hardware can help experts, but aren't always available, especially in the case of adversary systems. Experts commonly reverse engineer binary protocols by examining source code, tracing the execution of a compiled program, or fuzzing a program instance with test input to observe how it reacts. When these methods don't work, or aren't feasible, experts examine messages manually to try to understand the format.

Our previous work demonstrates one approach to automatically identifying message formats in protocols and samples where there is only one format, or some portion of the format is consistent across messages [5]. Building on this work, we now focus on automatic reverse engineering of protocols with multiple message formats.

II. Research Problem

In this proposal we focus on clustering binary messages by format. Binary message formats commonly use a reserved message-type field or keyword to indicate the format of the data transmitted when multiple formats are mixed, such as with union types. Keyword fields enable messages to be deserialized quickly and unambiguously. Examples of protocols utilizing keyword fields include ARP, BGP, SMB, and TFTP. Our research focus is the automatic identification of these keyword fields from network traces. Identifying these fields allows messages to be grouped by format, greatly simplifying both further manual and automatic reverse engineering efforts. Our contributions are the following:

- We propose a Monte Carlo ensemble approach to automatically identifying keyword fields from samples of unknown binary network protocol;
- We describe our proposed ensemble of six techniques, each tuned for identifying differences between binary message groups.

III. Our Approach

Our approach identifies keyword fields using an ensemble of methods combined with a Monte Carlo technique [11]. Our approach is driven by two key insights. First, when messages are correctly grouped by type, the resulting groups have higher internal consistency than between groups. Second, when messages are correctly grouped by type, the internal consistency is significantly higher than that of groups formed entirely at random.

Our method for automatically identifying keyword fields uses an ensemble of six methods, each tailored to a different aspect of message formats. One method uses a minimum description length principle to measure gains in compression from a candidate set of message groups. Another method leverages the tendency of messages within a group to have similar if not identical message lengths. We employ three information theoretic methods based on measuring between group differences in Shannon Entropy [14] at byte-offsets. Put another way, we look for groups which have different amounts of information at the same locations as an indication that they are performing different functioning within the underlying protocol. Finally, we leverage our earlier work on a semantic approach to inferring message formats [5] to infer semantic descriptions for each group and measure the difference between

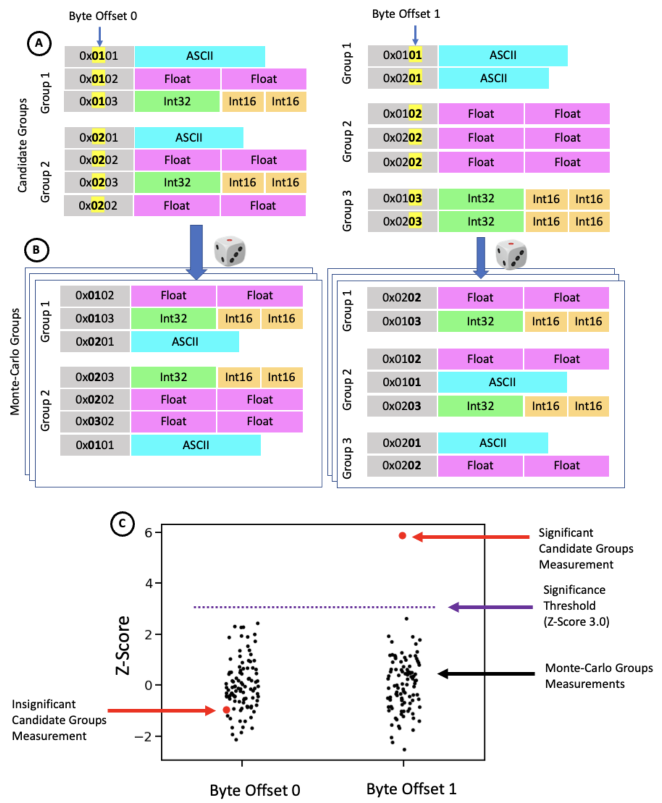


Fig. 1: Illustration of Monte Carlo approach to identifying keyword fields for a single ensemble method. Candidate groups are formed in (A) and measurements calculated for each byte-offset. Random groups are formed and their measurements calculated in (B). Keyword fields are identified in (C) by looking for candidate measurements which are significantly greater than those of the random groups.

them. The intuition here is that if the candidate groups are in fact different, the semantic descriptions we infer should be quite different. In contrast, when the message across candidate groups are similar, we should expect the semantic descriptions inferred to be similar.

To determine whether a candidate byte-offset contains a keyword field, we assign messages to groups based on their individual byte-values at that byte-offset as illustrated in Figure 1. We then calculate candidate values for each of our ensemble methods. We next use a Monte Carlo method to move messages randomly between groups, and again calculate values using our ensemble. Each method votes whether a candidate byte-offset is in fact a keyword field by comparing the value calculated from the original grouping to those formed by the Monte Carlo method. If the original values are significantly different than the Monte Carlo values, the method votes that the byte-offset is in fact akeyword field. We consider the byte-offset with the highest number of votes to be the best candidate and infer that as a keyword field.

Acknowledgments

This material is based upon work partly supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-19-C-0073. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

References

[1] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. Understanding the Mirai botnet. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1093–1110, 2017.

[2] Ignacio Bermudez, Alok Tongaonkar, Marios Iliofotou, Marco Mellia, and Maurizio M Munafo. Automatic protocol field inference for deeper protocol understanding. In *2015 IFIP Networking Conference (IFIP Networking)*, pages 1–9. IEEE, 2015.

[3] Georges Bossert, Frédéric Guihéry, and Guillaume Hiet. Towards automated protocol reverse engineering using semantic information. In *Proceedings of the 9th ACM Aymposium on Information, Computer and Communications Security*, pages 51–62, 2014.

[4] Jared Chandler, Kathleen Fisher, Erin Chapman, Eric Davis, and Adam Wick. Invasion of the botnet snatchers: A case study in applied malware cyberdeception. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*, 2020.

[5] Jared Chandler, Adam Wick, and Kathleen Fisher. BinaryInferno: A Semantic-Driven Approach to Field Inference for Binary Message Formats. In *Proceedings of the Symposium on Network and Distributed System Security (To Appear: NDSS'23)*, 2022.

[6] Chia Yuan Cho, Eui Chul Richard Shin, Dawn Song, et al. Inference and Analysis of Formal Models of Botnet Command and Control Protocols. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, pages 426–439. ACM, 2010.

[7] Lorenzo De Carli, Ruben Torres, Gaspar Modelo-Howard, Alok Tongaonkar, and Somesh Jha. Botnet Protocol Inference in the Presence of Encrypted Traffic. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pages 1–9. IEEE, 2017.

[8] Julien Duchene, Colas Le Guernic, Eric Alata, Vincent Nicomette, and Mohamed Kaâniche. State of the art of network protocol reverse engineering tools. *Journal of Computer Virology and Hacking Techniques*, 14(1):53–68, 2018.

[9] Stephan Kleber, Henning Kopp, and Frank Kargl. NEMESYS: Network message syntax reverse engineering by analysis of the intrinsic structure of individual messages. In *12th USENIX Workshop on Offensive Technologies (WOOT 18)*, 2018.

[10] Stephan Kleber, Lisa Maile, and Frank Kargl. Survey of protocol reverse engineering algorithms: Decomposition of tools for static traffic analysis. *IEEE Communications Surveys & Tutorials*, 2018, 2018.

[11] Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.

[12] John Narayan, Sandeep K Shukla, and T Charles Clancy. A survey of automatic protocol reverse engineering tools. *ACM Computing Surveys (CSUR)*, 48(3):1–26, 2015.

[13] Johannes Pohl and Andreas Noack. Automatic wireless protocol reverse engineering. In *13th USENIX Workshop on Offensive Technologies (WOOT 19)*, 2019.

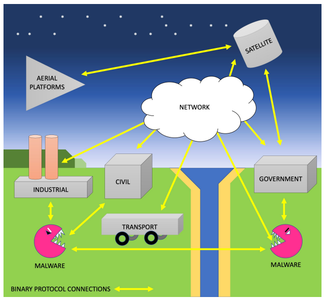
[14] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, July 1948.

[15] Yapeng Ye, Zhuo Zhang, Fei Wang, Xiangyu Zhang, and Dongyan Xu. NetPlier: probabilistic network protocol reverse engineering from message traces. In *Proceedings of the Symposium on Network and Distributed System Security (NDSS'21)*, 2021.

A Monte Carlo Ensemble Approach to Automatically Identifying Keywords in Binary Message Formats

Jared Chandler | Jared.Chandler@tufts.edu | **Tufts University, Department of Computer Science**

Problem Overview: Protocol Reverse Engineering



Why Reverse Engineer Binary Protocols?

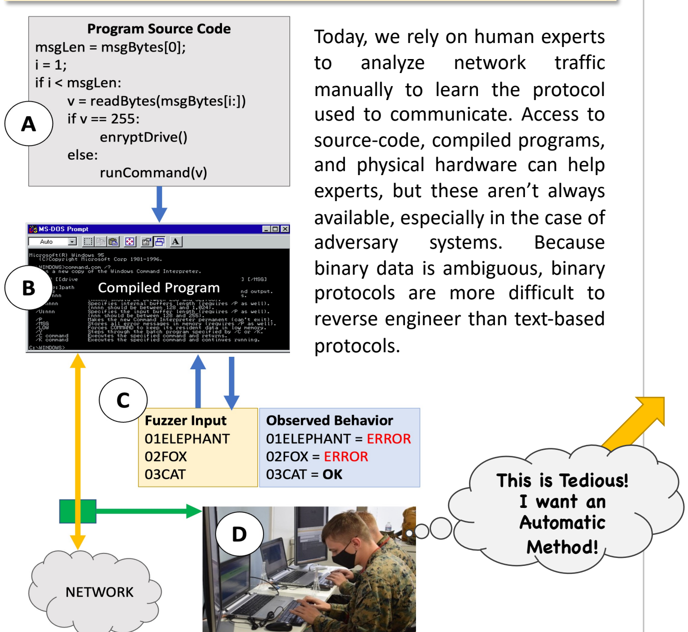
1. Find Vulnerabilities
2. Generate Exploits
3. Validate Specifications
4. Build Firewall & IDS Rules
5. Improve Fuzzing
6. Understand Adversary Communications

Binary Protocols of Security Interest

- Computer Networking Protocols & Applications
- Industrial Control Systems & Critical Infrastructure
- Internet of Things (IoT) and Smart Home Devices
- Vehicle Telemetry and Onboard Control Systems
- Malware Command and Control (C2) / Botnets
- Legacy & Proprietary Systems with Missing Specifications

How are Protocols Currently Reverse Engineered?

By a human expert taking days, weeks, or longer.



Today, we rely on human experts to analyze network traffic manually to learn the protocol used to communicate. Access to source-code, compiled programs, and physical hardware can help experts, but these aren't always available, especially in the case of adversary systems. Because binary data is ambiguous, binary protocols are more difficult to reverse engineer than text-based protocols.

Common Ways to Reverse Engineer Protocols

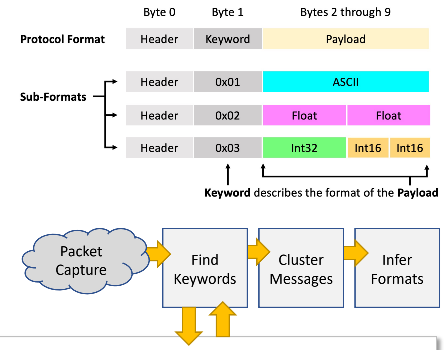
A. Static Analysis of Source Code
B. Dynamic Analysis of Executable Program
C. Interactive Fuzzing
D. Manual Inspection of Traffic

Protocol Reverse Engineering Pipeline Steps

1. Collection of Messages
2. Clustering by Message Format
3. Field Segmentation & Identification
4. State Machine Inference & Semantic Identification

Our Approach to Automatically Identifying Keywords

Our work focuses on Step 2 of the protocol reverse engineering pipeline: ***clustering messages by format***. Keywords are used when protocols transmit messages with different formats. For example: a client query format and a server response format. Identifying these keywords automatically lets the messages be exactly clustered by format. Further reverse engineering is improved by having the correct message clustering.

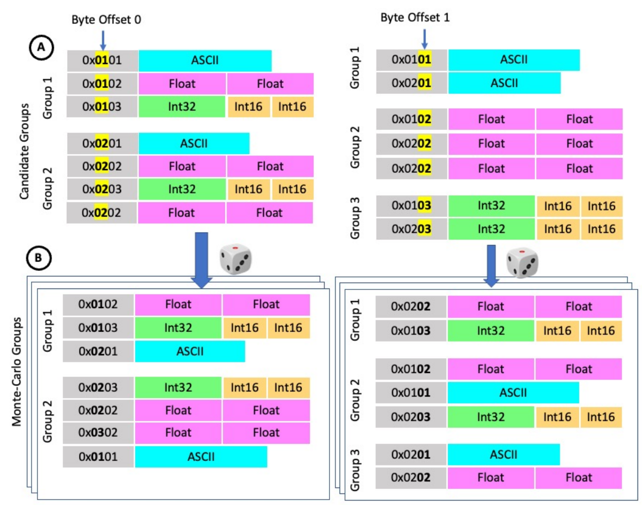


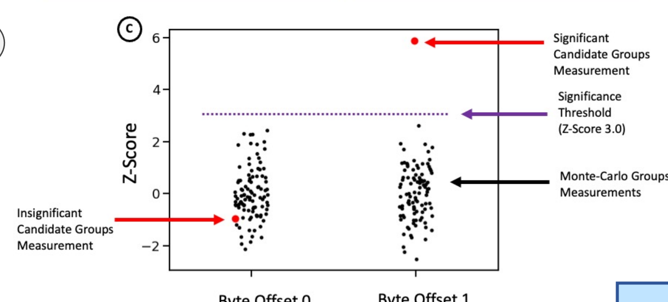
How Our Monte Carlo Approach to Identifying Keyword Fields Works:

A. Form candidate groups and calculate distance measurements for each byte-offset. We use an ensemble of distance metrics each specialized to a different property.

B. Random groups are formed, and their measurements similarly calculated.

C. Keyword fields are identified by distance measurements which are significantly greater than those of the random groups.





Advantages of Monte Carlo Ensemble Approach

- ✓ Uses Passive Network Traces
- ✓ Does Not Need Access to the Program
- ✓ Completely Automatic Method
- ✓ Does Not Require Training Data

Interested?

Jared.Chandler@Tufts.edu

SCAN →

This material is based upon work partly supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-19-C-0073. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.