

Polyglot: Automatic Extraction of Protocol Message Format using Dynamic Binary Analysis

Juan Caballero, Heng Yin, Zhenkai Liang
Carnegie Mellon University

Dawn Song
*Carnegie Mellon University
& UC Berkeley*

Protocol reverse-engineering

- **Process of extracting the application-level protocol used by a program, without access to the protocol specification**
- **Encompasses extracting:**
 - the Protocol Message Format
 - the Protocol State Machine
- **Challenges:**
 - No protocol specification
 - » Many closed application protocols
 - Automatic
 - » Samba, ICQ, Yahoo Messenger took years

This paper is about the Protocol Message Format

Outline

- **Introduction**
- **Techniques**
 - **Direction field extraction**
 - **Separator extraction**
 - **Keyword extraction**
 - **Multi-byte fixed length fields**
- **Evaluation**

Automatic message format extraction

- **Extract the Protocol Message Format, one message at a time**
- **Problem: Automatically extract the message format from an unknown message given a program binary implementing the protocol**
- **Message Format = field sequence**
 - **Field: smallest contiguous sequence of application data with meaning**

Use dynamic binary analysis to extract message format

Message Format Applications

- **Extracting the message format is important for multiple problems:**
 - Protocol analyzers
 - Application dialog replay
 - Intrusion detection
 - Fingerprint generation
 - Detecting services running on non-standard ports

Challenges

Main challenge is to identify the field boundaries:

- 1. Variable-length fields boundaries**
 - **Extract the direction fields (e.g. length fields)**
 - **Extract the separators**
- 2. Fixed-length fields boundaries**
 - **Avoid concatenating two consecutive fields**
 - **Avoid splitting a single field into multiple ones**

Another important challenge is to:

- 3. Find the keywords present in the message**
 - **Allow to map traffic to the protocol**

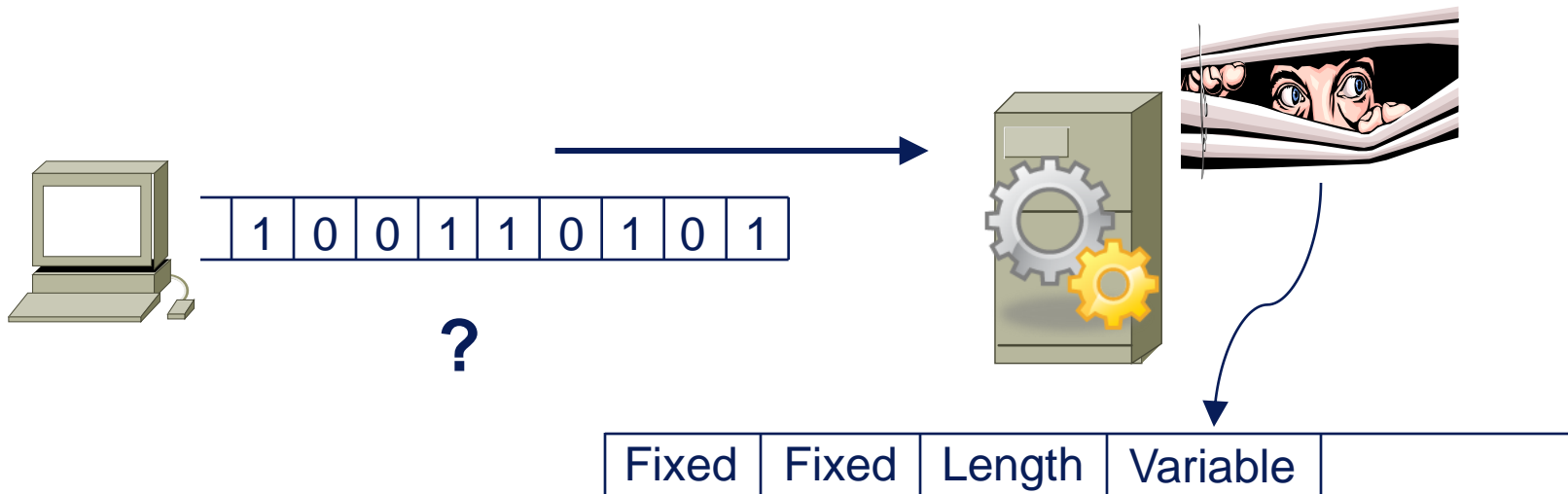
We propose techniques to address these challenges

Related work

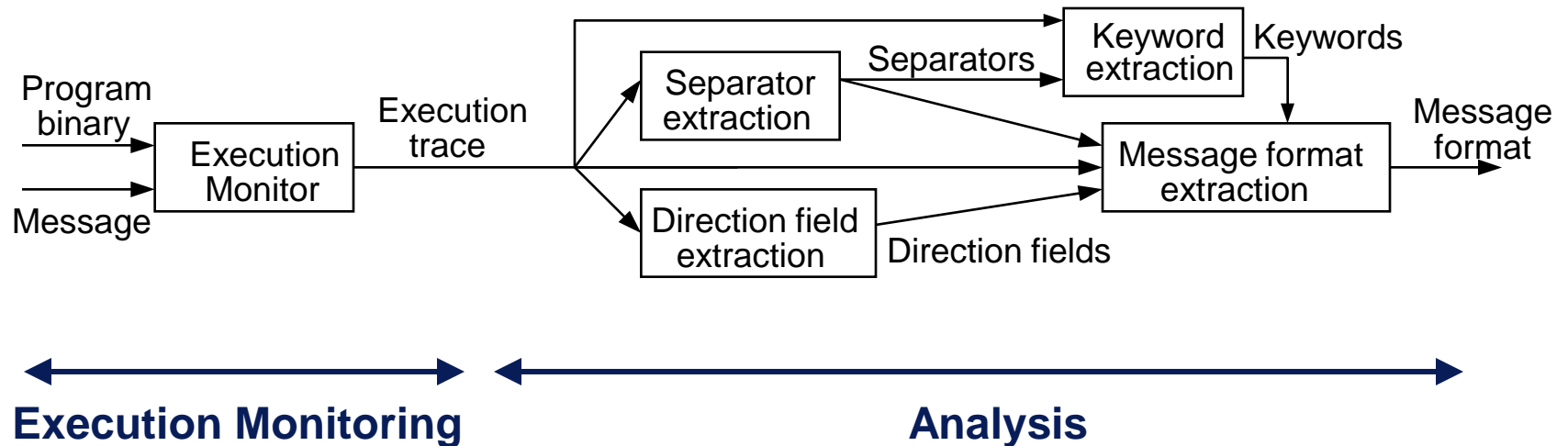
- **Previous work focuses on** network traces
- **Fundamental limitation:** lack of protocol semantics
- **Protocol Message Format:** [Cui06,Cui07,Leita05,Leita06]
 - Need complex heuristics to identify direction fields
 - Need assumptions about separator values & data encoding
 - Cannot identify consecutive fixed-length fields
- **Keywords / Application Signatures:** [Haffner05,Ma06,Cui07]
 - Need tokens to appear at same position across messages
 - Problems: variable-length fields, floating fields
- **Some previous work using binary analysis:** [Lim06]
 - Assumes knowledge of program's output functions

Approach

- **Shadowing: Monitor how a binary implementing the protocol processes a received message**
 - How the message is parsed
 - How the message is used



Polyglot: system architecture



- **Input: program binary and received message**
- **Output: message format**
- **Two phases:**
 - **Execution Monitoring** → generate execution trace
 - **Analysis** → extract the message format

Execution monitoring phase

- **Run the binary inside an emulator**
- **Implements dynamic taint analysis**
[Chow04,Suh04,Costa05, Newsome05, Crandall06]
 - **Received network data marked as interesting**
 - **Marking propagates as data is used**
 - **Marking represents offsets in input stream**
- **Outputs an execution trace which contains**
 - **Executed instructions**
 - **Operand content during execution**
 - **Taint information**

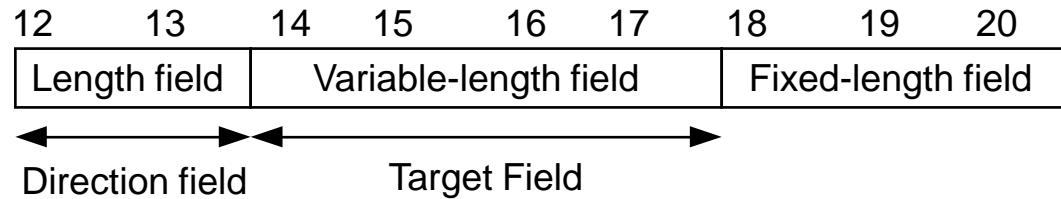
Outline

- **Introduction**
- **Techniques**
 - **Direction field extraction**
 - **Separator extraction**
 - **Keyword extraction**
 - **Multi-byte fixed length fields**
- **Evaluation**

Direction fields

- **Direction Fields: store information about the location of another target field in the message**
- **Target field has variable-length**
- **Three types of direction fields**
 - **Length fields**
 - » **Encode length of target field**
 - **Pointer fields**
 - » **Encode displacement relative to another position**
 - **Counter fields**
 - » **Encode position of field in a list of items**

Direction field extraction



- **Detect direction fields when used to increment the value of a pointer**
- **Two methods to increment the pointer:**
 - **Direct:** increment computed from field and added to pointer
 - » Indirect memory access that 1) accesses a tainted memory position, and 2) where the destination address has been computed from tainted data
 - **Indirect:** increment computed using a loop
 - » Need to find loops and check if they have tainted conditions

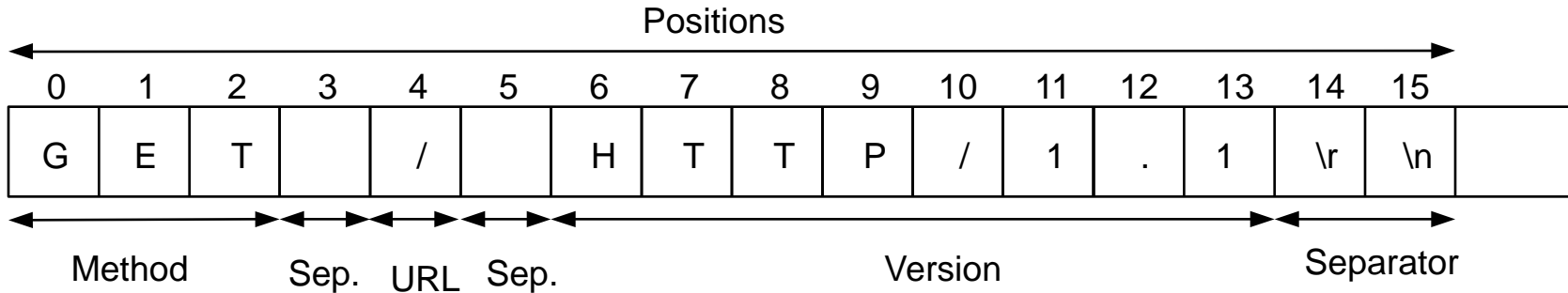
Direction field extraction

- **No assumptions about field encoding used**
 - » number of bytes/words/quadwords,
 - » integer / string
- **Variable-length fields:**
 - Direction fields need to appear before variable-length fields
 - Variable-length field: the sequence of bytes between the last position belonging to the direction field and the end of the target field

Outline

- **Introduction**
- **Techniques**
 - **Direction field extraction**
 - **Separator extraction**
 - **Keyword extraction**
 - **Multi-byte fixed length fields**
- **Evaluation**

Separators



- **Used by protocols to mark the boundary of variable-length fields**
- **Separator = Constant + Scope**
 - Constant marks the boundary
 - Scope contains list of positions in the message where the separator is used
- **Two separator types depending on scope: Field, In-fields**

Separator extraction

- Program compares the separator value against all positions in the separator scope until it finds it
- Intuition: find tokens that are compared against consecutive positions in the received data
- Three step process:
 1. Summarize tainted comparisons (comparison table)
 2. Extract byte-long separators
 3. Extend byte-long separators into multi-byte separators

Offset Positions

	0	1	2	3	4	5	6
0x0a (\n)	F	F	F	F	F	F	F
0x47 (G)	T						
0x45 (E)		T					
0x54 (T)			T				
0x2f (/)					T	F	F

Comparison table

No assumptions about separator value or encoding

Outline

- **Introduction**
- **Techniques**
 - **Direction field extraction**
 - **Separator extraction**
 - **Keyword extraction**
 - **Multi-byte fixed length fields**
- **Evaluation**

Keywords

- **Protocol constants that:**
 - appear in the received application data
 - are supported by the implementation
- **Keywords can be strings or numbers**
 - “Content-type” in HTTP
 - Version number
- **Useful to match traffic to protocols**

Keyword extraction

- Program compares the supported keywords against the received data
- Intuition: analyze true comparisons that include tainted data
- Two phases
 - Populate the comparison table
 - Extract true comparisons at consecutive positions
 - » Break keywords at false comparisons or separators

Offset Positions

	0	1	2	3	4	5	6
0x0a (\n)	F	F	F	F	F	F	F
0x47 (G)	T						
0x45 (E)		T					
0x54 (T)			T				
0x2f (/)					T	F	F

Tokens

Comparison table

Outline

- **Introduction**
- **Techniques**
 - **Direction field extraction**
 - **Separator extraction**
 - **Keyword extraction**
 - **Multi-byte fixed length fields**
- **Evaluation**

Multi-byte fixed length fields

- **Intuition: program operates on the entire field**
 - All field bytes need to be considered for comparisons, arithmetic operations...
- **Most fields are independent**
 - Exceptions: direction fields, checksums
- **Detection: for each instruction extract the list of positions it operates on**
 - If not direction field, then create a new multi-byte fixed field
- **Limitation: can only extract fields up to the system's word size → 4 bytes in 32-bit architectures**
 - But, still better than previous work that separates each byte

Outline

- **Introduction**
- **Techniques**
 - **Direction field extraction**
 - **Separator extraction**
 - **Keyword extraction**
 - **Multi-byte fixed length fields**
- **Evaluation**

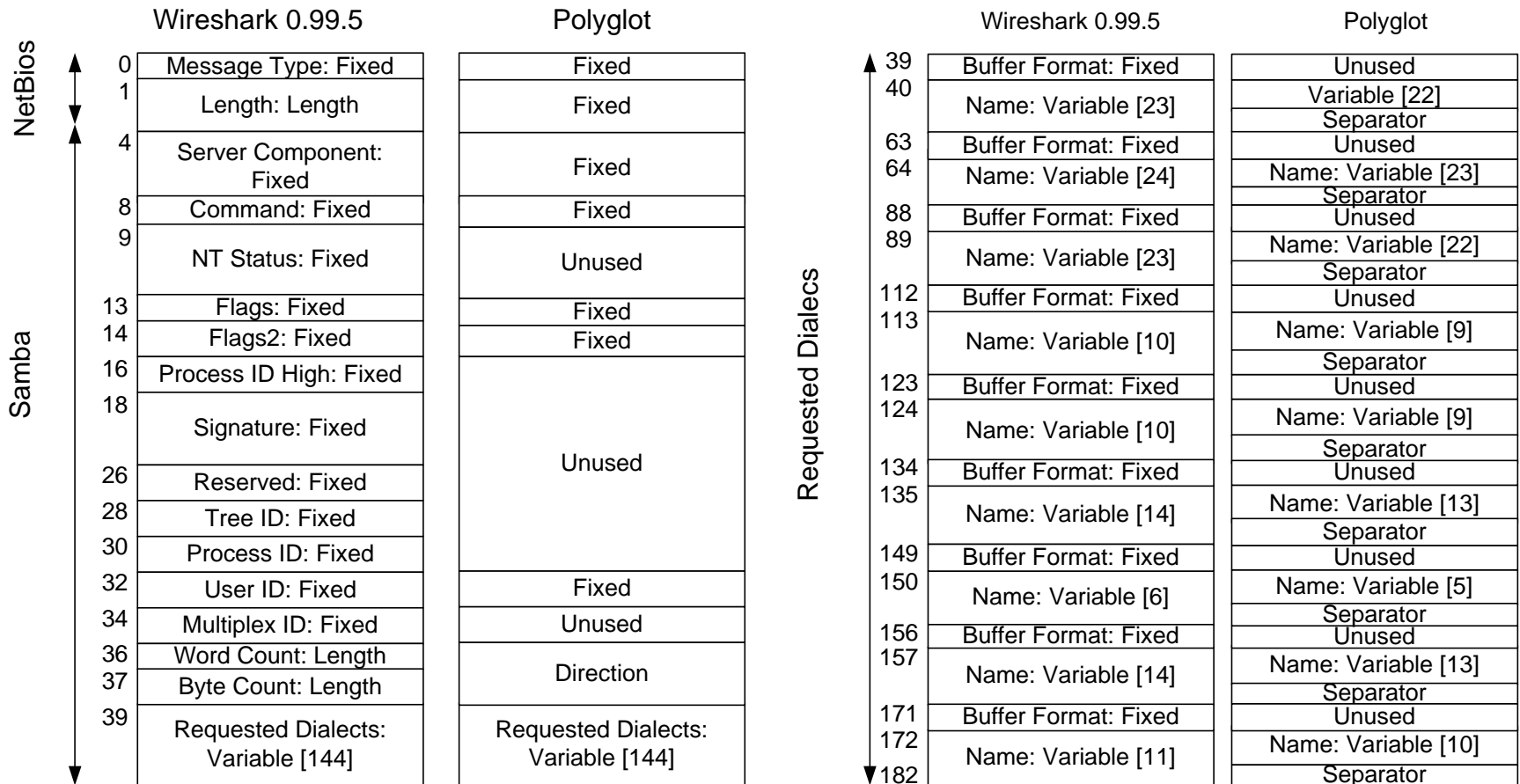
Evaluation overview

- **Evaluated on 11 programs and 5 protocols: HTTP, DNS, IRC, ICQ, Samba**
 - **Clients & Servers**
 - **Windows & Linux**
 - **Text & Binary**
- **Results compared to Wireshark**

Program	Version	Type	Size	OS
Apache	2.2.4	HTTP server	4,344kB	Win.
Miniweb	0.8.1	HTTP server	528kB	Win.
Savant	3.1	HTTP server	280kB	Win.
Bind	9.3.4	DNS server	224kB	Win.
MaraDNS	1.2.12.4	DNS server	164kB	Win.
SimpleDNS	4.00.06	DNS server	432kB	Win.
TinyICQ	1.2	ICQ client	11kB	Win.
Beware ircd	1.5.7	IRC server	148kB	Win.
JoinMe	1.41	IRC server	365kB	Win.
Unreal IRCd	3.2.6	IRC server	760kB	Win.
Sambad	3.0.24	Samba server	3,580kB	Linux

- **Only Samba, HTTP results presented for brevity**

Samba results



- **Negotiate Protocol request**

HTTP separators

Separator	Apache	Savant	Miniweb
0x0d0a ('\r\n')	Field	Field	Field
0x2f ('/')	In-field	In-field	-
0x2e ('.')	In-field	In-field	In-field
0x20 (' ')	-	In-field	In-field
0x3a20 (': ')	In-field	-	-

GET /index.html HTTP/1.1

Host: 10.0.0.1

User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.1) Gecko/20061208 Firefox/2.0.0.1

Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5

Accept-Language: en-us,en;q=0.5

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7

Keep-Alive: 300

Connection: Keep-Alive

HTTP keywords

Keyword	Apache	Savant	Miniweb
GET	Yes	Yes	Yes
Host	Yes	NS	Yes
User-Agent	NS	Yes	NS
Accept	Yes	Yes	NS
Accept-Language	Accept	Yes	NS
Accept-Encoding	Accept-	Accept-	NS
Accept-Charset	Accept-	Accept-	NS
Keep-Alive	NS	NS	NS
Connection	Yes	Yes	Yes

Server	Add. keywords
Apache	'HTTP/', 'e', 'Keep-Alive'
Savant	'HTTP/1.', 'Keep-Alive', ':'
Miniweb	-

GET /index.html HTTP/1.1

Host: 10.0.0.1

User-Agent: Mozilla/5.0 [...]

Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5

Accept-Language: en-us,en;q=0.5

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7

Keep-Alive: 300

Connection: Keep-Alive

Conclusion

- **Proposed a new approach for extracting the protocol message format using dynamic binary analysis**
- **Proposed new techniques for:**
 1. **detecting** direction fields
 2. **detecting** separators
 3. **detecting** multi-byte fixed-length **fields**
 4. **extracting protocol** keywords
- **Evaluated techniques on 5 protocols and 11 programs. Results compared to Wireshark**

Questions?

