BLEEM: Packet Sequence Oriented Fuzzing for Protocol Implementations

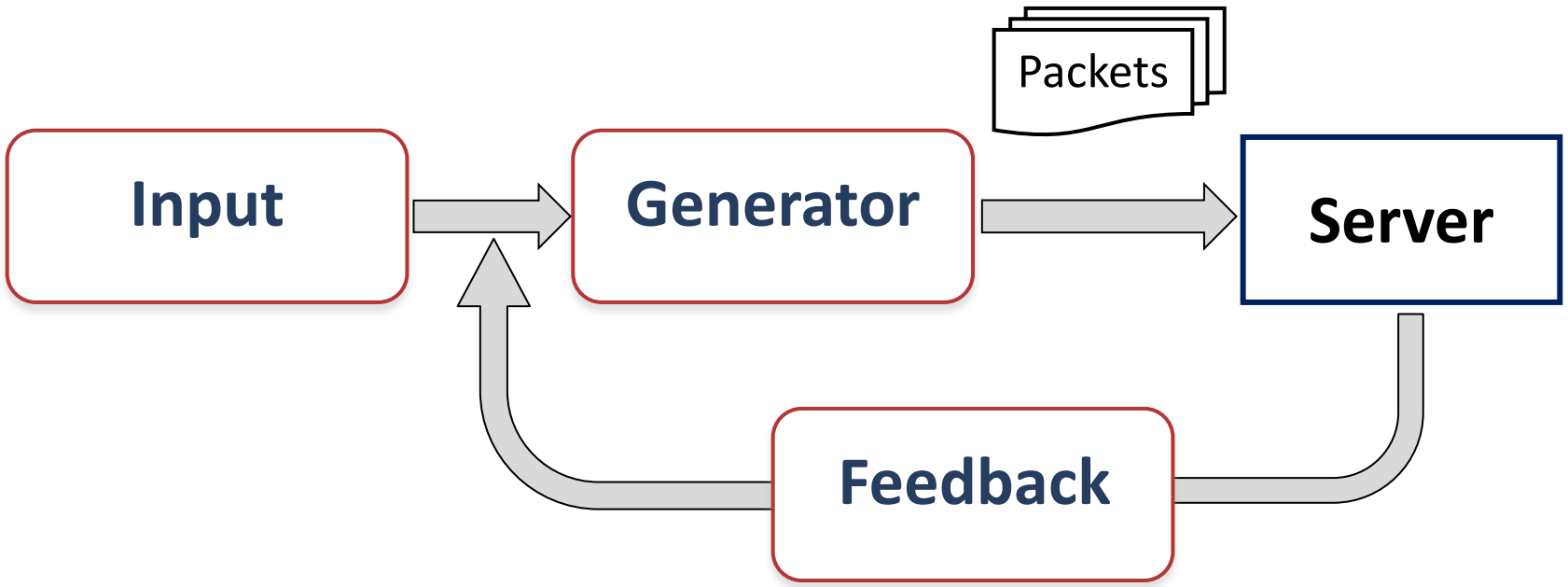
Zhengxiong Luo[1], Junze Yu[1], Feilong Zuo[1], Jianzhong Liu[1], Yu Jiang[1], Ting Chen[2], Abhik Roychoudhury[3], and Jiaguang Sun[1]

[1]Tsinghua University

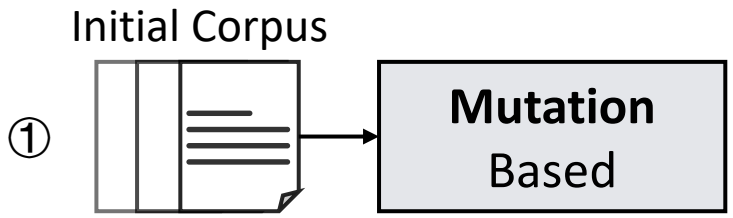
[2]University of Electronic Science and Technology of China

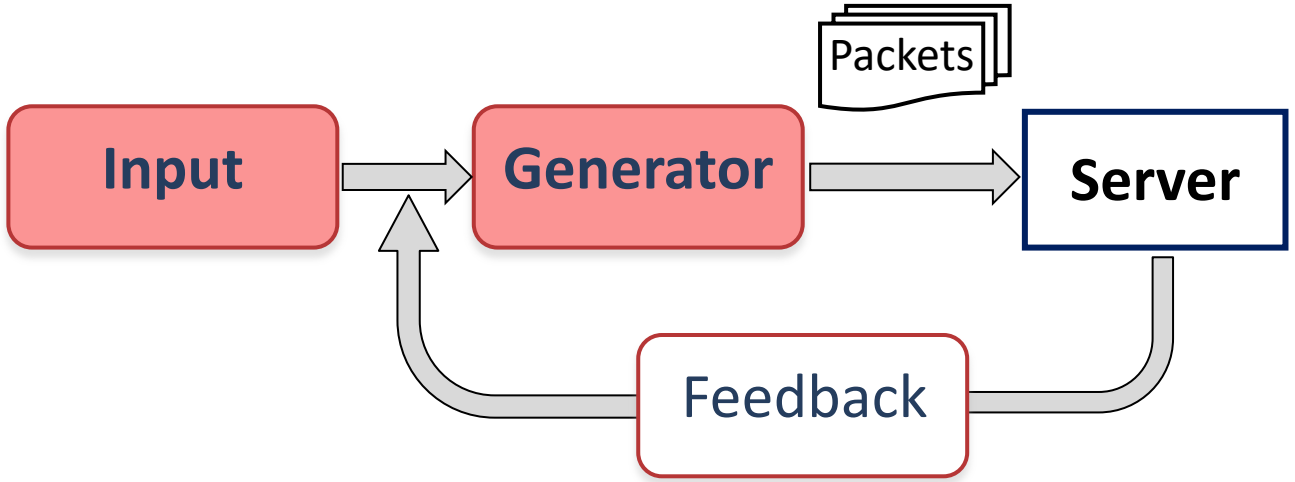
[3]National University of Singapore

Protocol Fuzzing

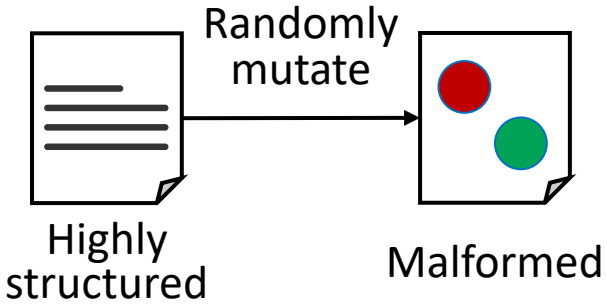


Packet Generation

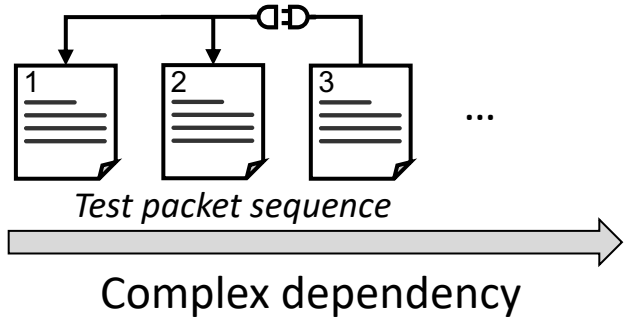


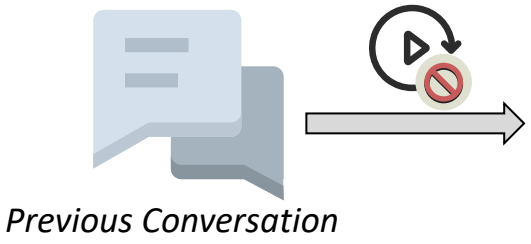


- **Syntax**



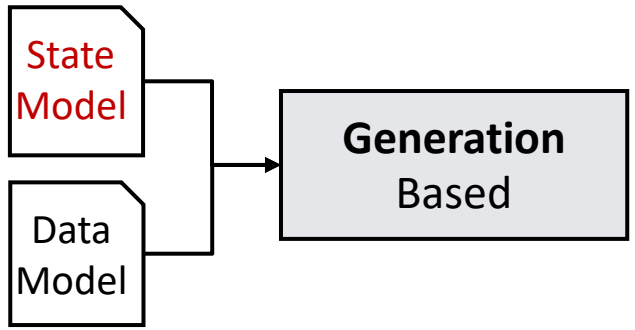
- **Semantic**

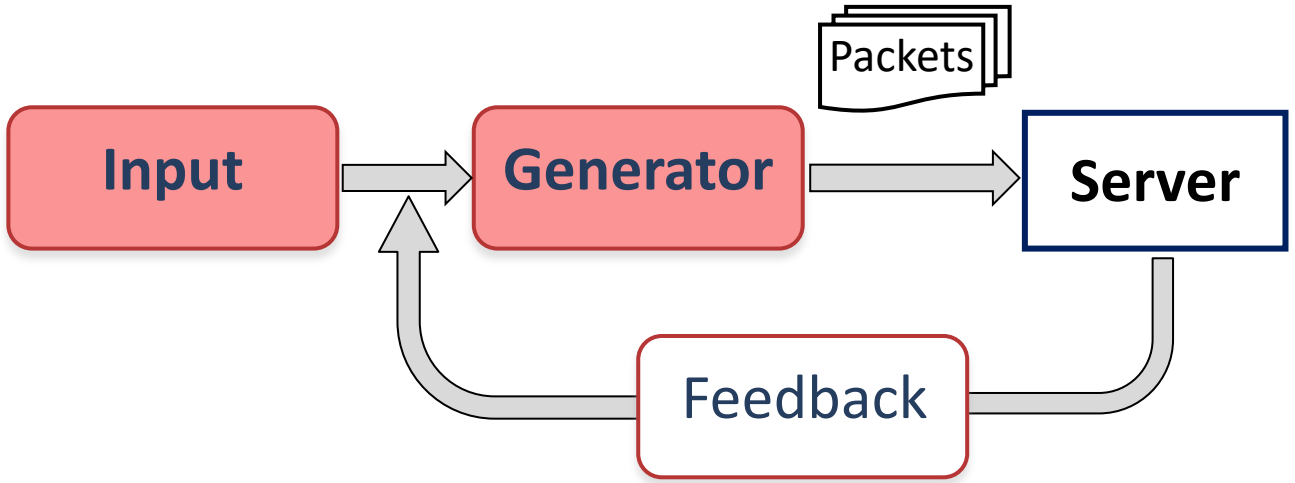


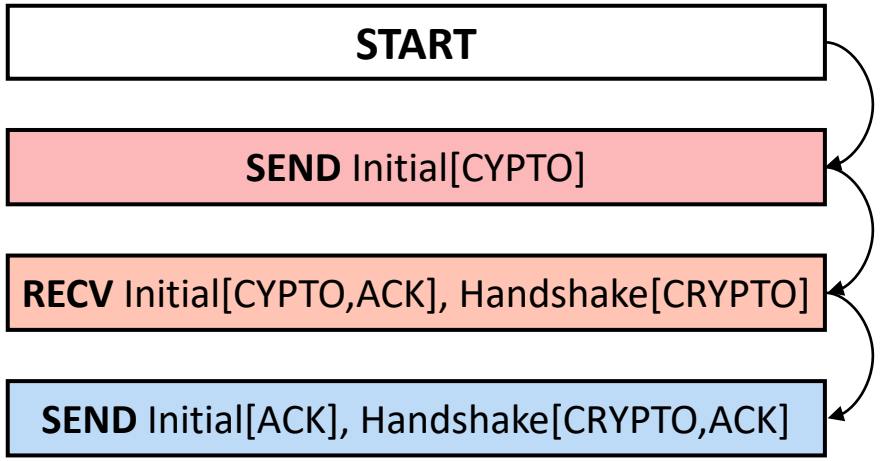


Packet Generation

② Protocol Model







An Example State Machine

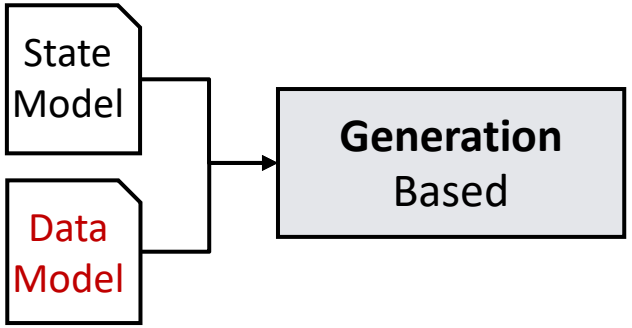
```
<StateModel name="QUIC" initialState="HANDSHAKE">
  <State name="HANDSHAKE">
    <Action type="output">
      <DataModel ref="Initial[CRYPTO]"/>
    </Action>
    <Action type="input">
      <DataModel ref="Initial[CRYPTO,ACK]+Handshake[CRYPTO]"/>
    </Action>
    <Action type="output">
      <DataModel ref="Initial[ACK]+Handshake[CRYPTO,ACK]"/>
    </Action>
  </State>
  ...
</StateModel>
```

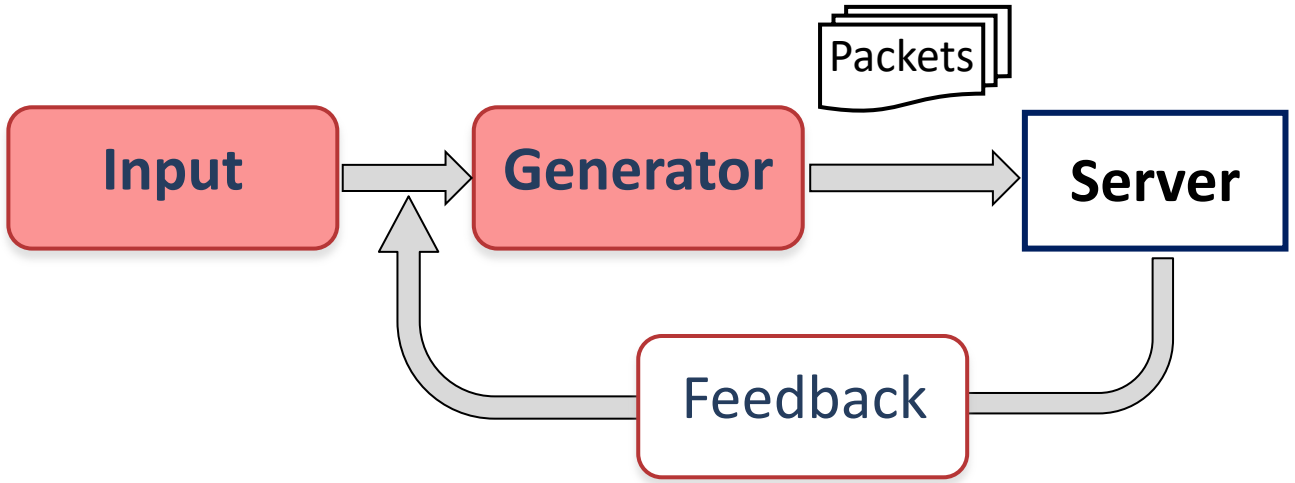
The Corresponding State Model as Peach Pit[1]

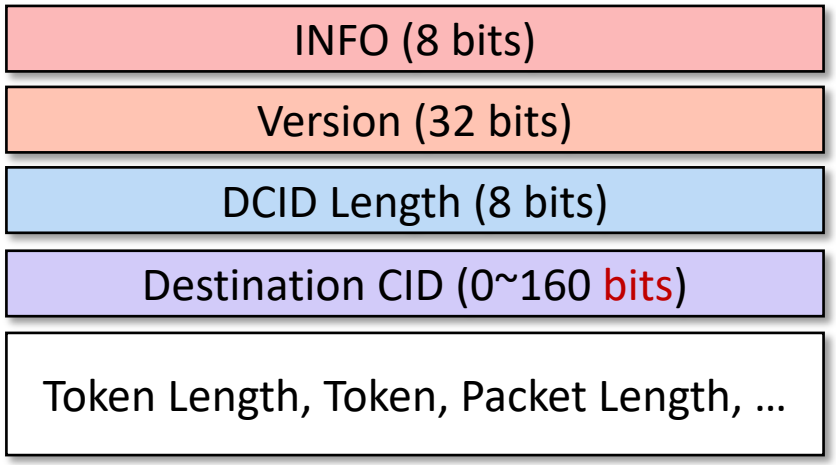
[1]Peach Pit is the configuration file for protocol fuzzer Peach

Packet Generation

② Protocol Model







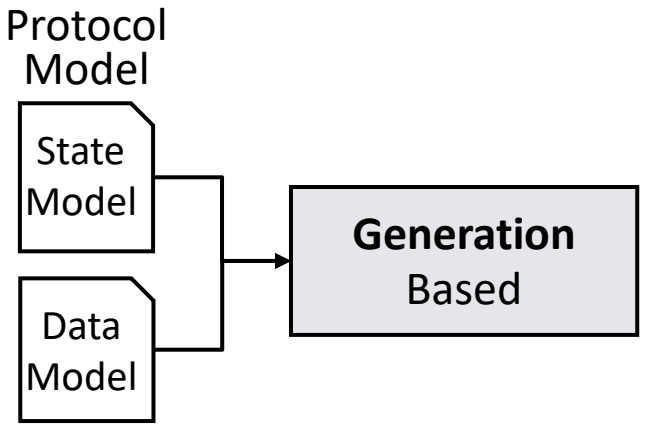
An Example Packet Format

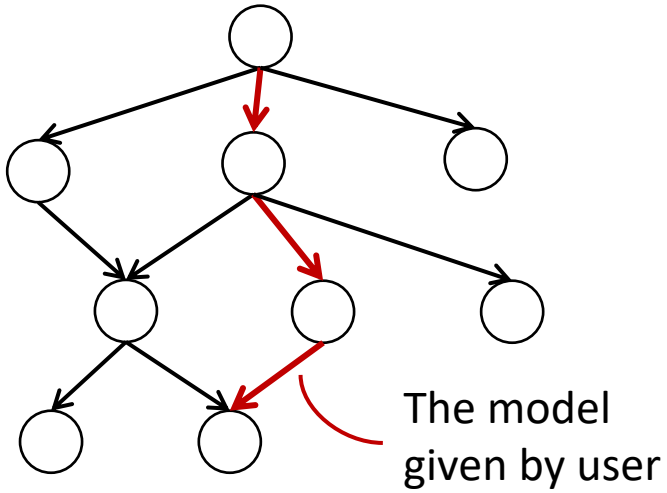
```
<DataModel name="Initial[CRYPTO]">
  <Block name="Header">
    <Number name="info" size="8" value="cd" valueType="hex"/>
    <Number name="version" size="32" value="faceb002"
          valueType="hex" token="true"/>
    <Number name="DCID_length" size="8">
      <Relation type="size" of="DCID"/>
    </Number>
    <String name="DCID" nullTerminated="false"/>
    ...
  </Block>
  <Block name="CRYPTO"> ... </Block>
</DataModel>
```

The Corresponding Data Model as Peach Pit

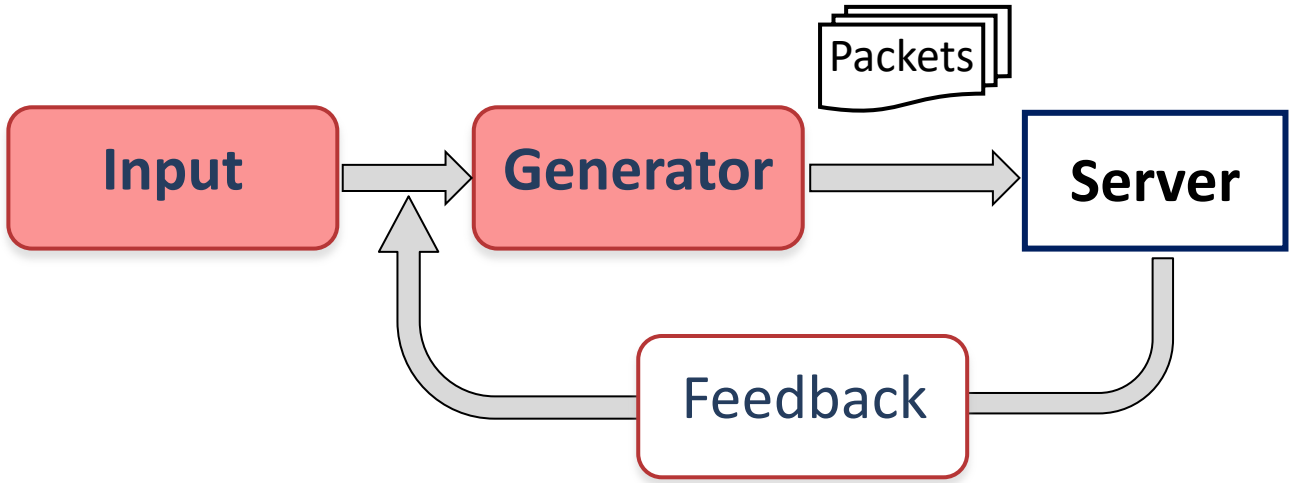
Packet Generation

②



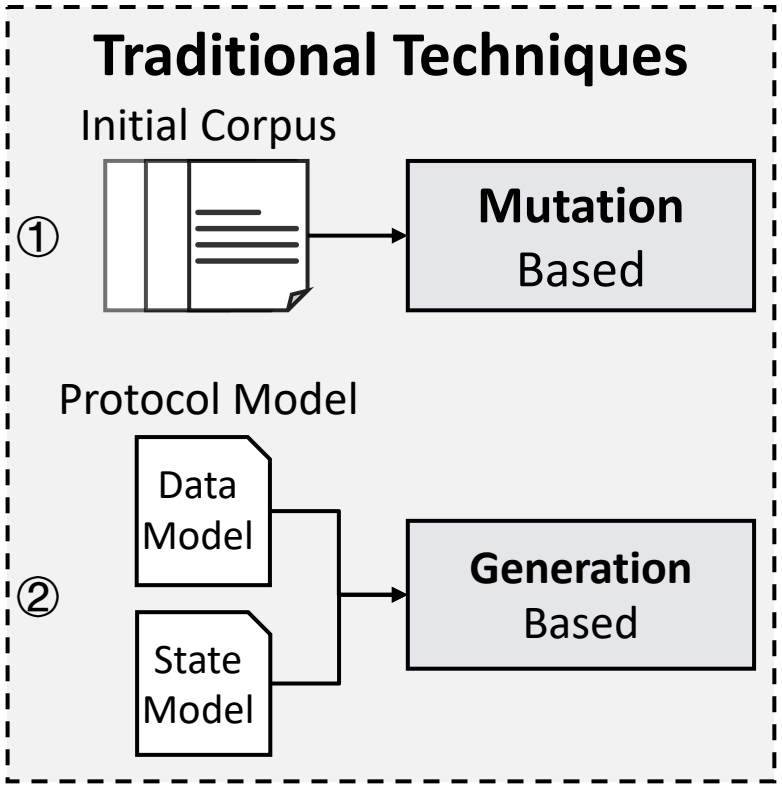


Entire Protocol State Machine

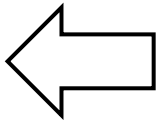


- Conduct testing by **strictly following the actions in the protocol model**
- New behaviors beyond the user-defined model are ignored

Packet Generation



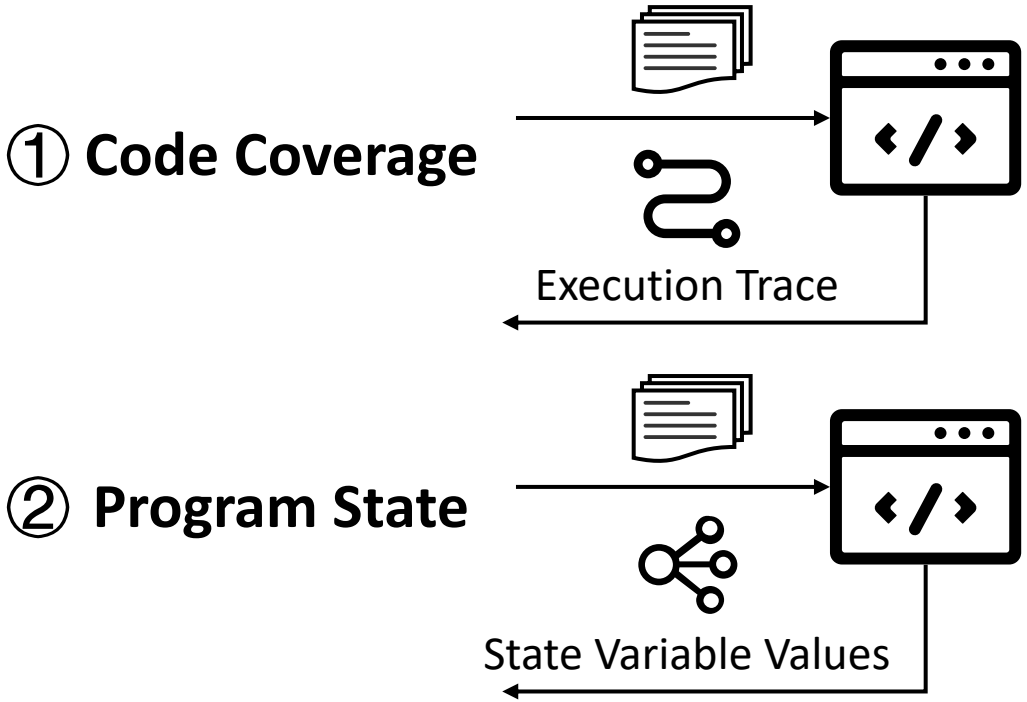
Act as a client

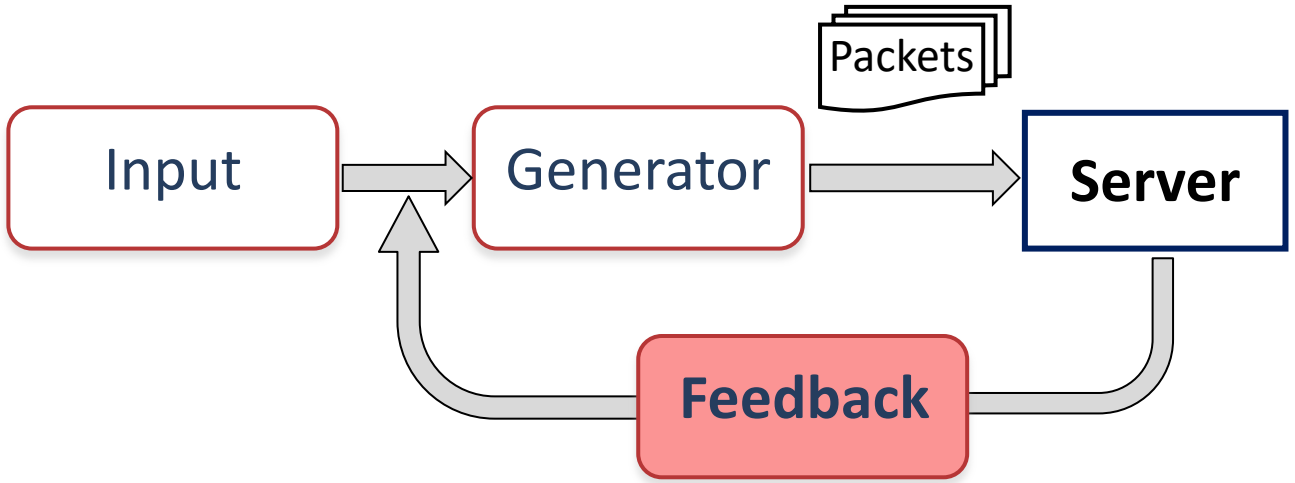


Expert protocol knowledge

Challenge 1: Highly complex protocol logic

Feedback for Fuzzing Optimization





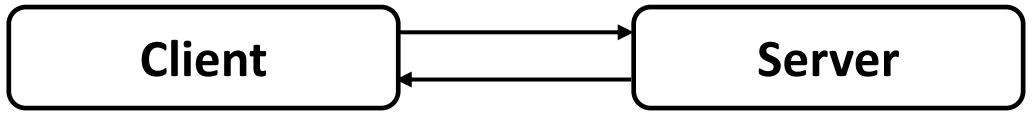
Challenge 2: Lack of a scalable feedback mechanism

- Existing methods require source code or binaries and do not apply to black-box fuzzing

BLEEM Overview

Insights

- Protocol is a bipartite system
- Utilize the logic encoded in parites



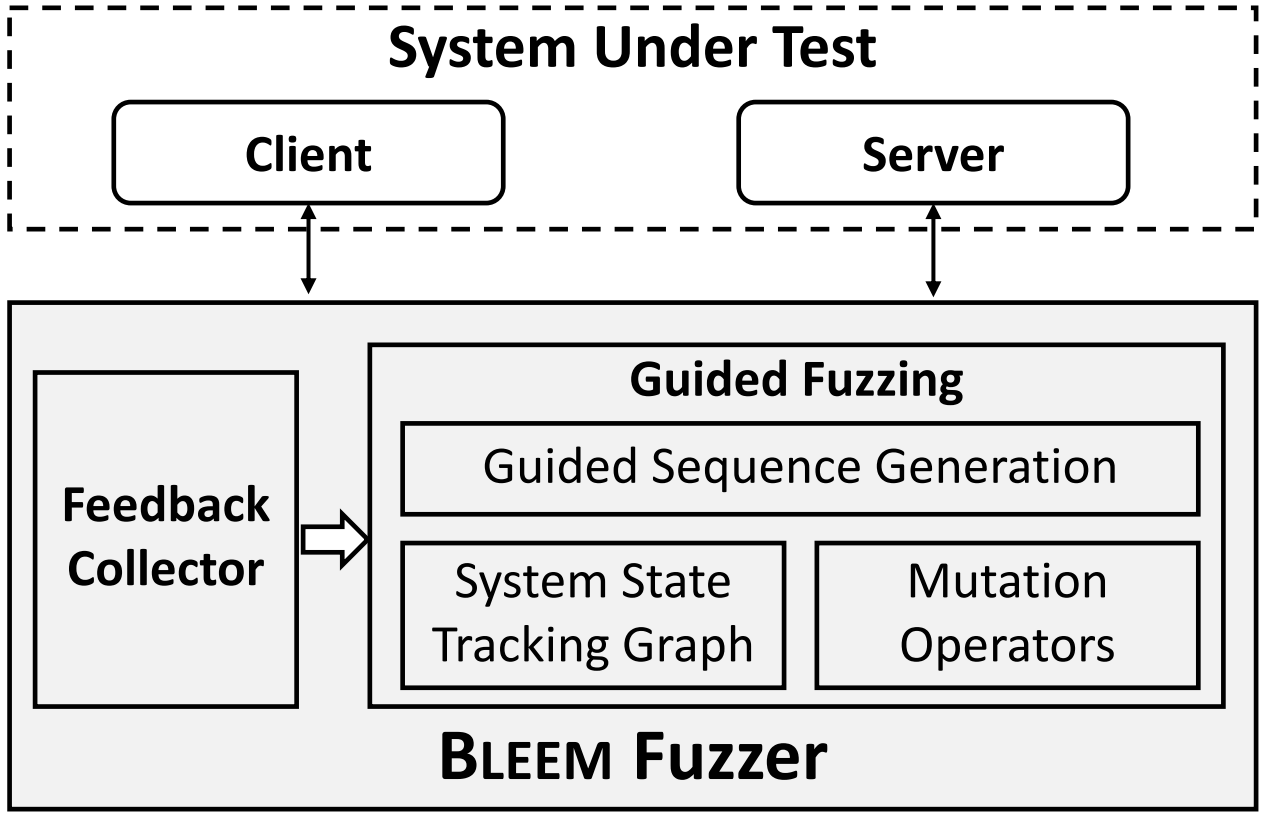
BLEEM Overview

Insights

- Protocol is a bipartite system
- Utilize the logic encoded in parites

BLEEM – Black-Box Protocol Fuzzer

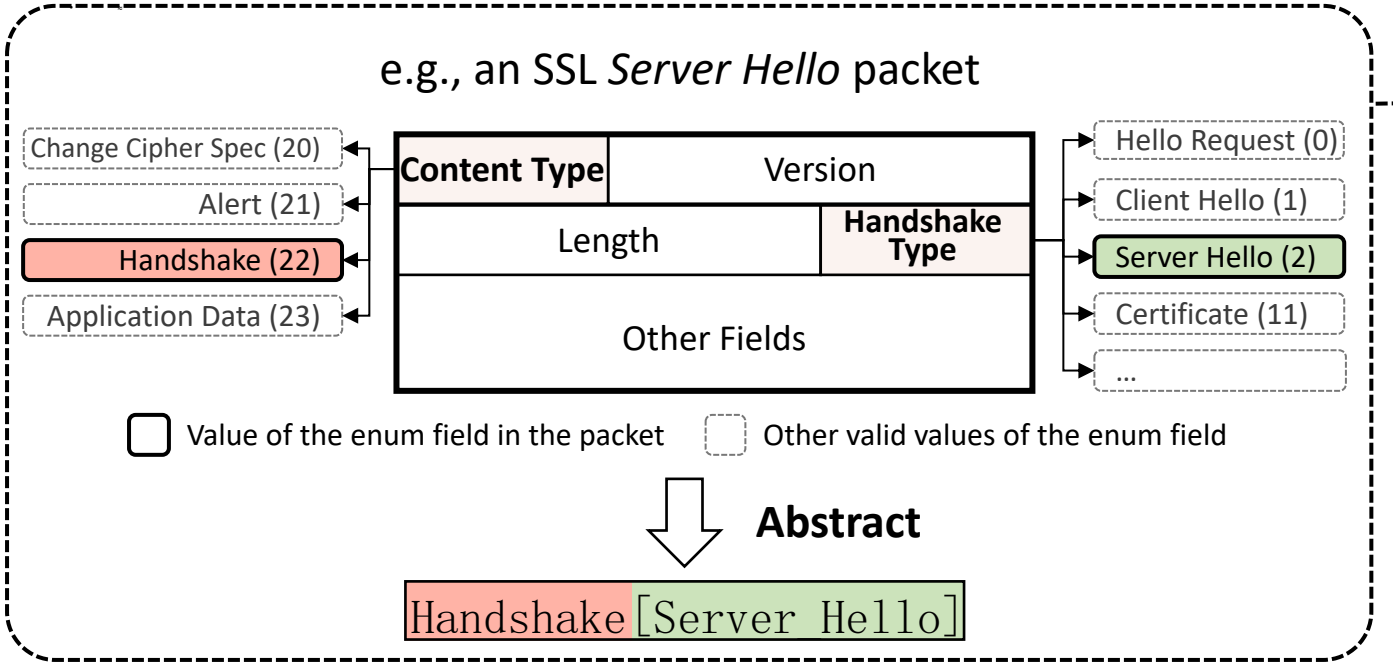
- Feeback Collector
- Guided Fuzzing
  - Mutation Operators
  - System State Tracking Graph
  - Guided Sequence Generation

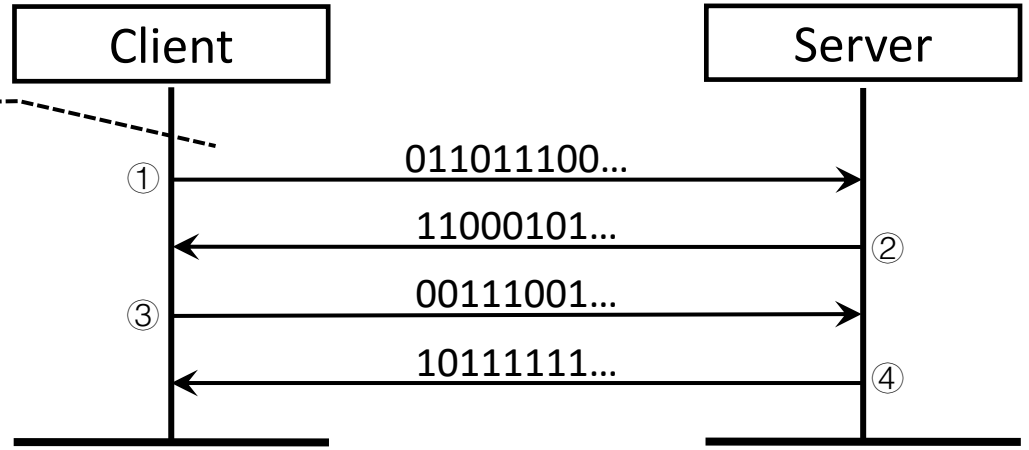


Feedback Collector

The output packets can indicate the inner protocol state

- Use concrete packets can cause confusion => Focus on crucial type fields

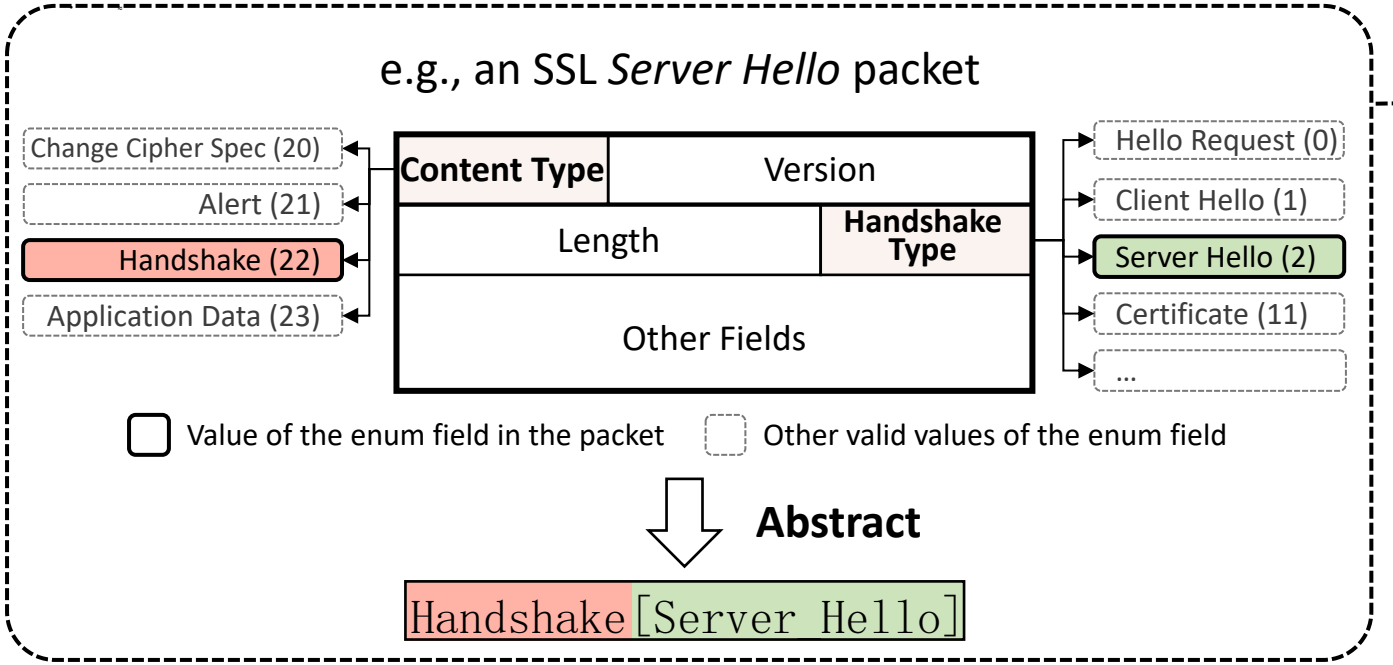


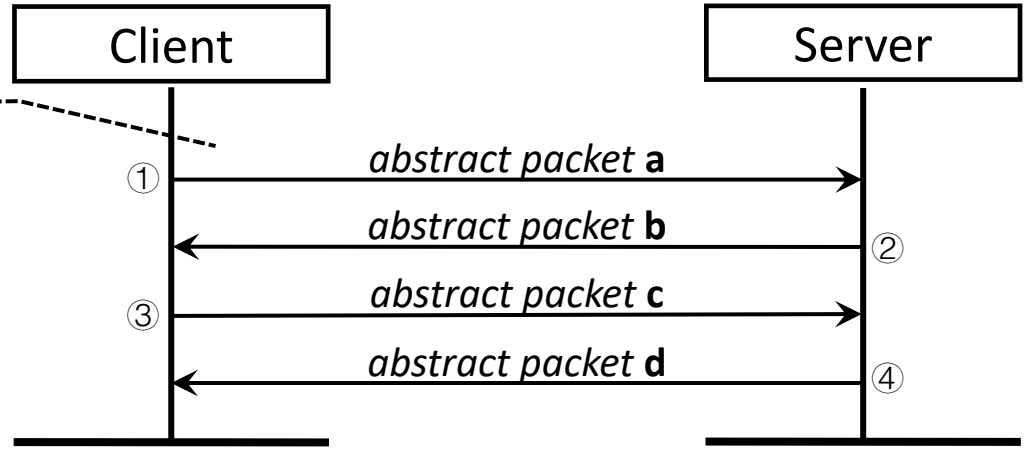


Feedback Collector

The output packets can indicate the inner protocol state

- Use concrete packets can cause confusion => Focus on crucial type fields

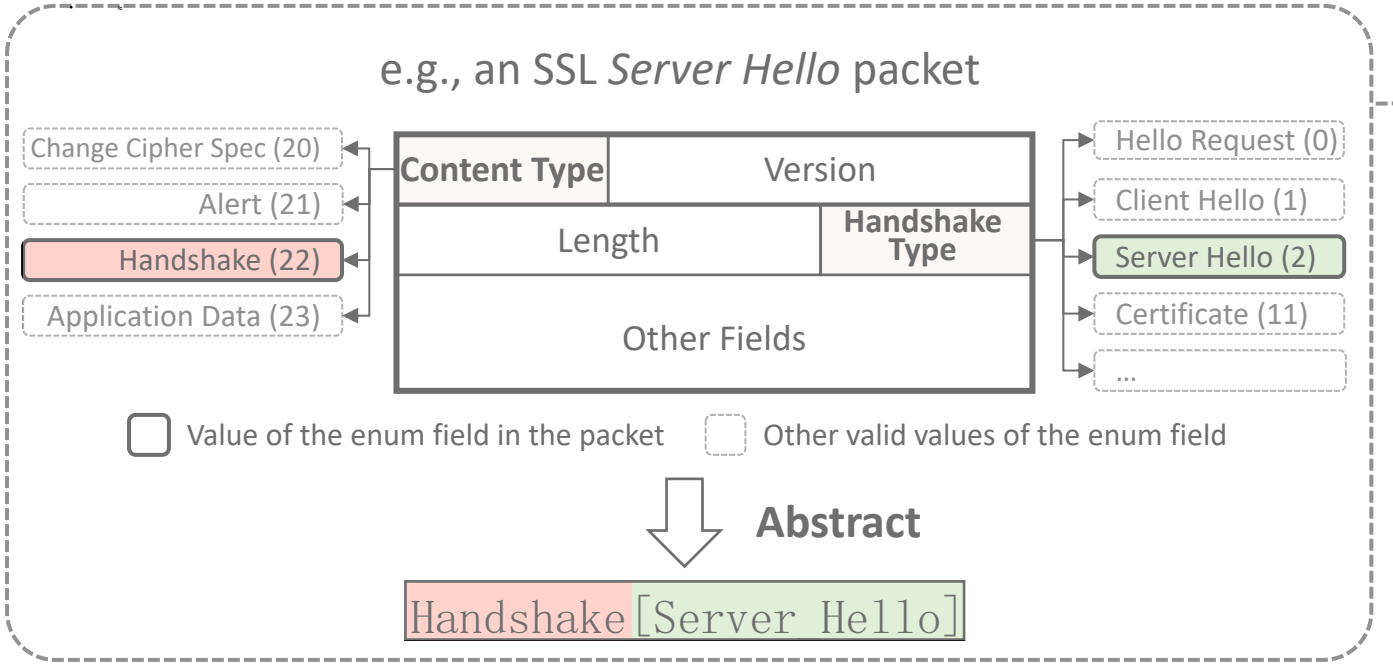


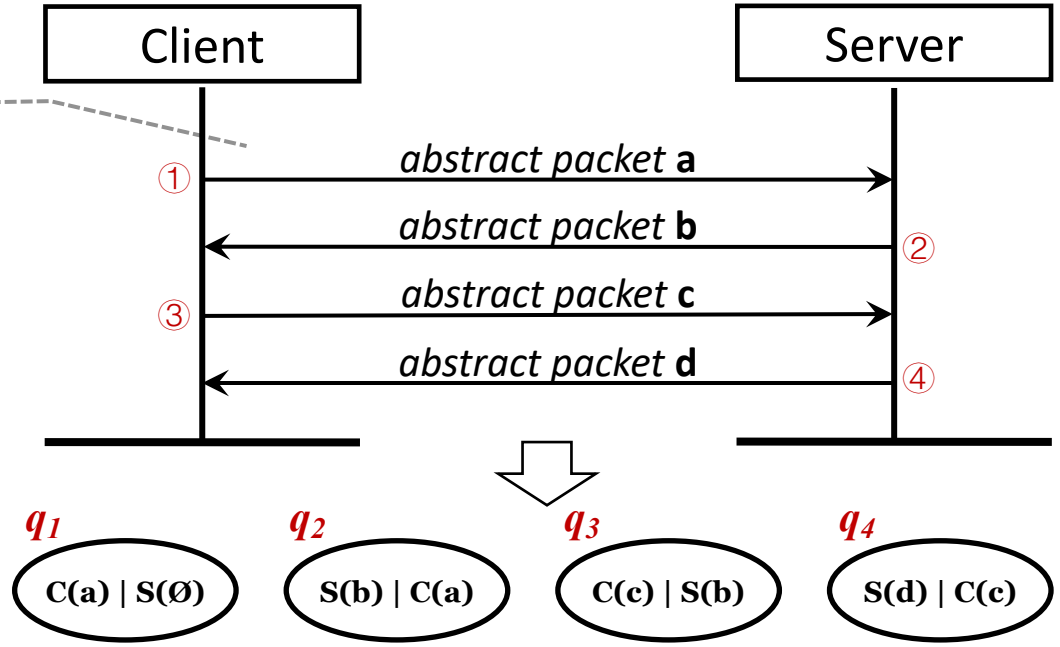


Feedback Collector

The output packets can indicate the inner protocol state

- Use concrete packets can cause confusion => Focus on crucial type fields



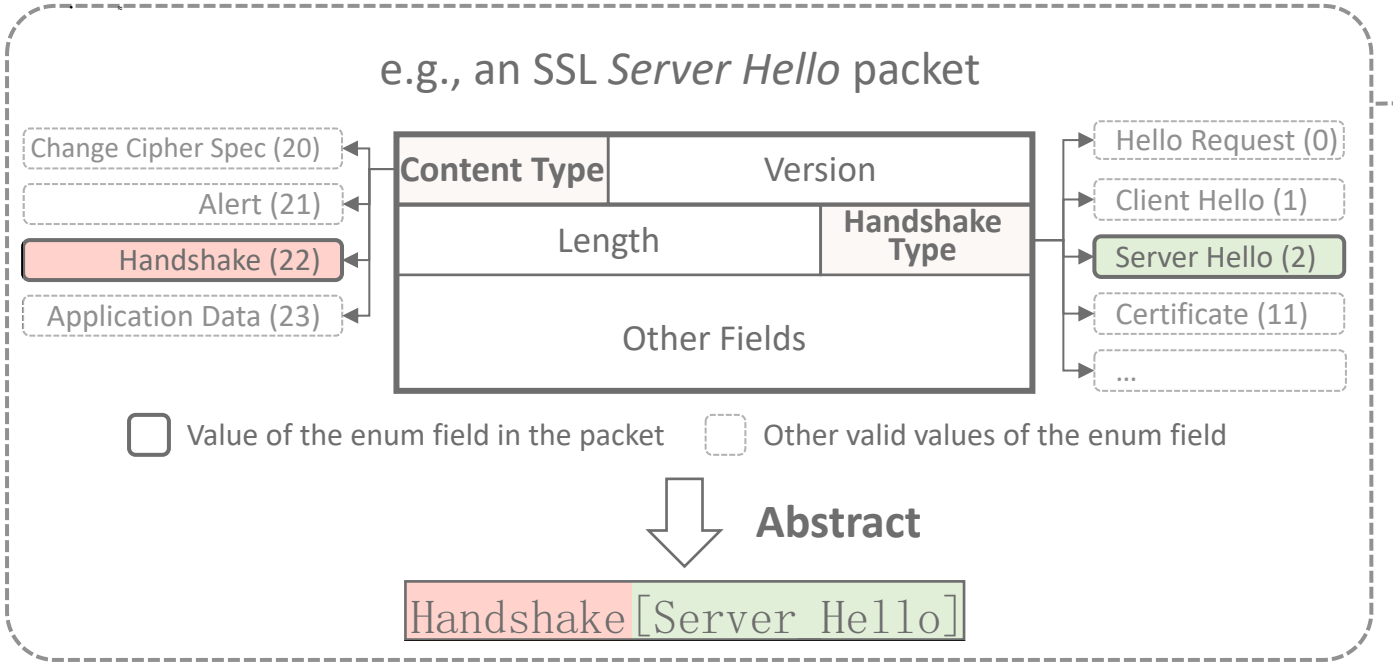


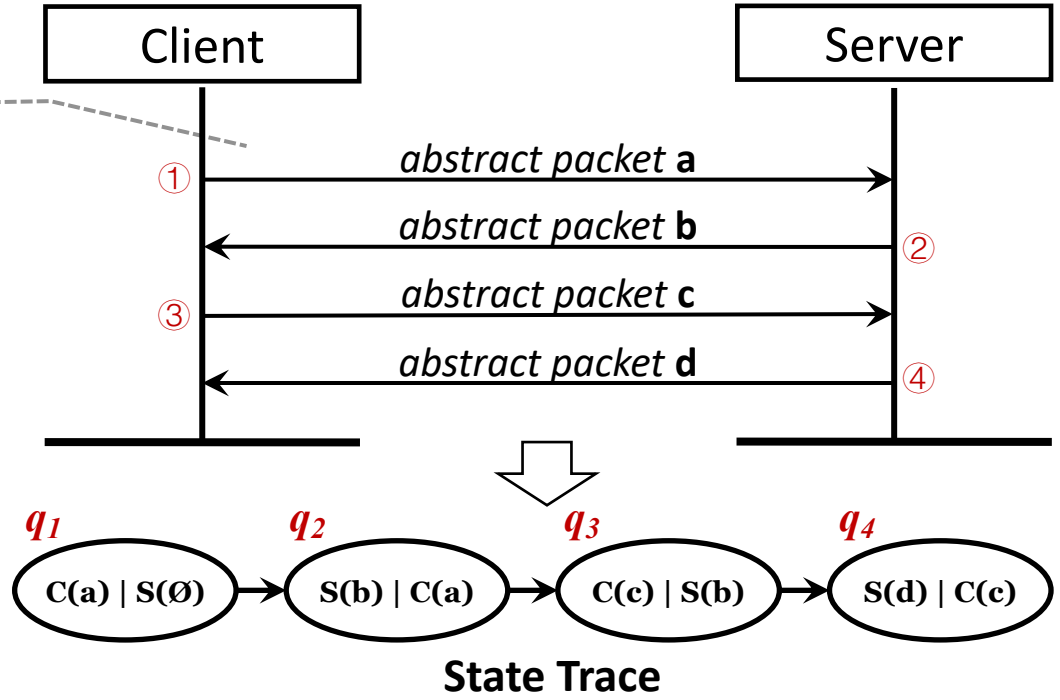
Model the full state using bi-directional communication information

Feedback Collector

The output packets can indicate the inner protocol state

- Use concrete packets can cause confusion => Focus on crucial type fields

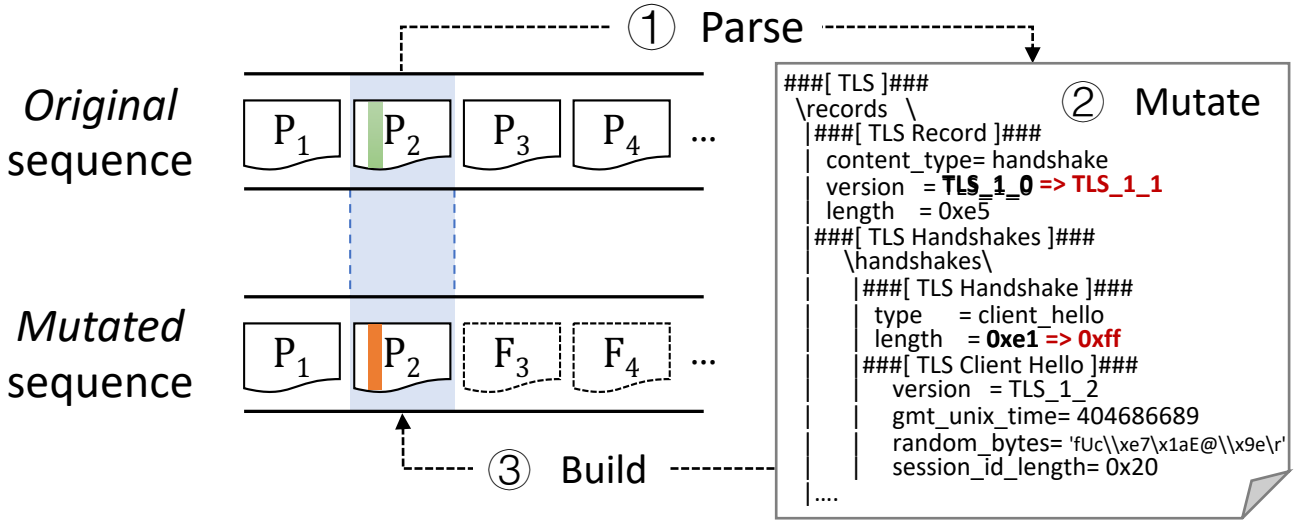




Model the full state using bi-directional communication information

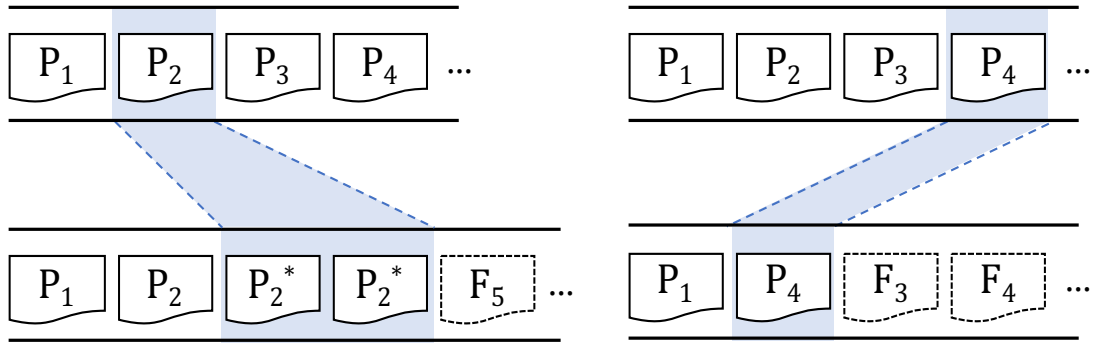
Protocol-Aware Mutation Operator

(1) Packet-Level Mutation



Structure-aware mutation

(2) Sequence-Level Mutation



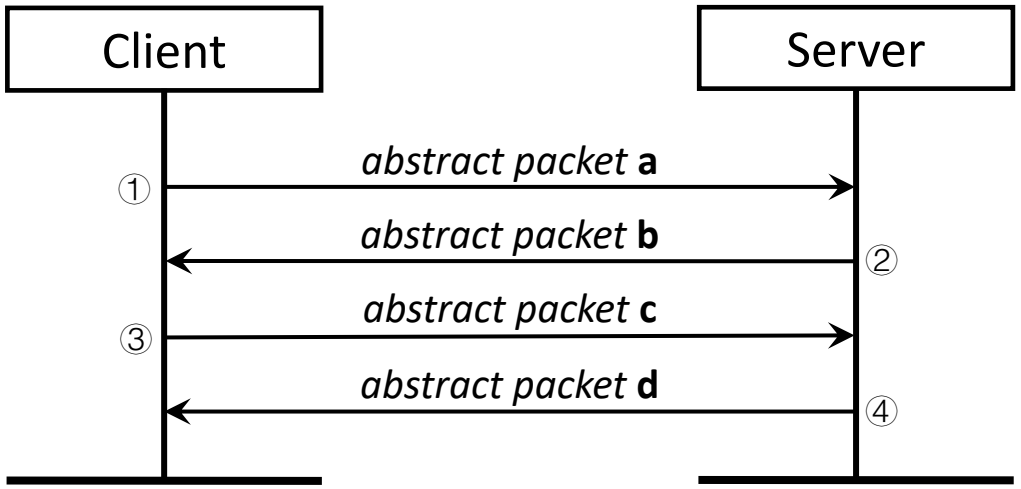
(a) Packet duplication

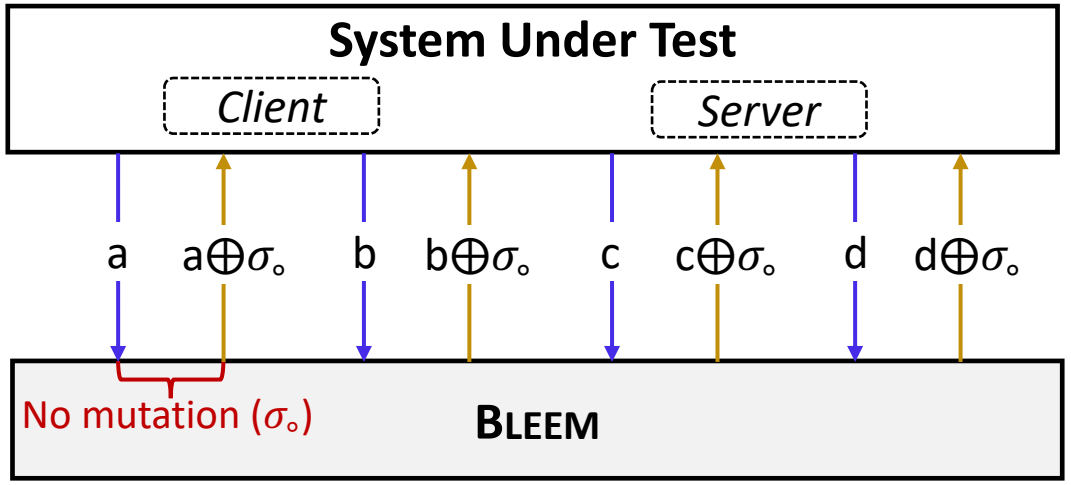
(b) Packet disordering

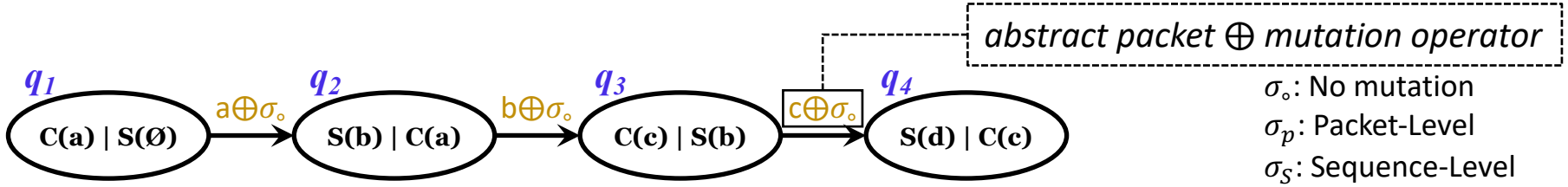
System State Tracking Graph (SSTG)

Merge the *state trace* to **track the explored state space**

Label each transition to **obtained the reproducibility**



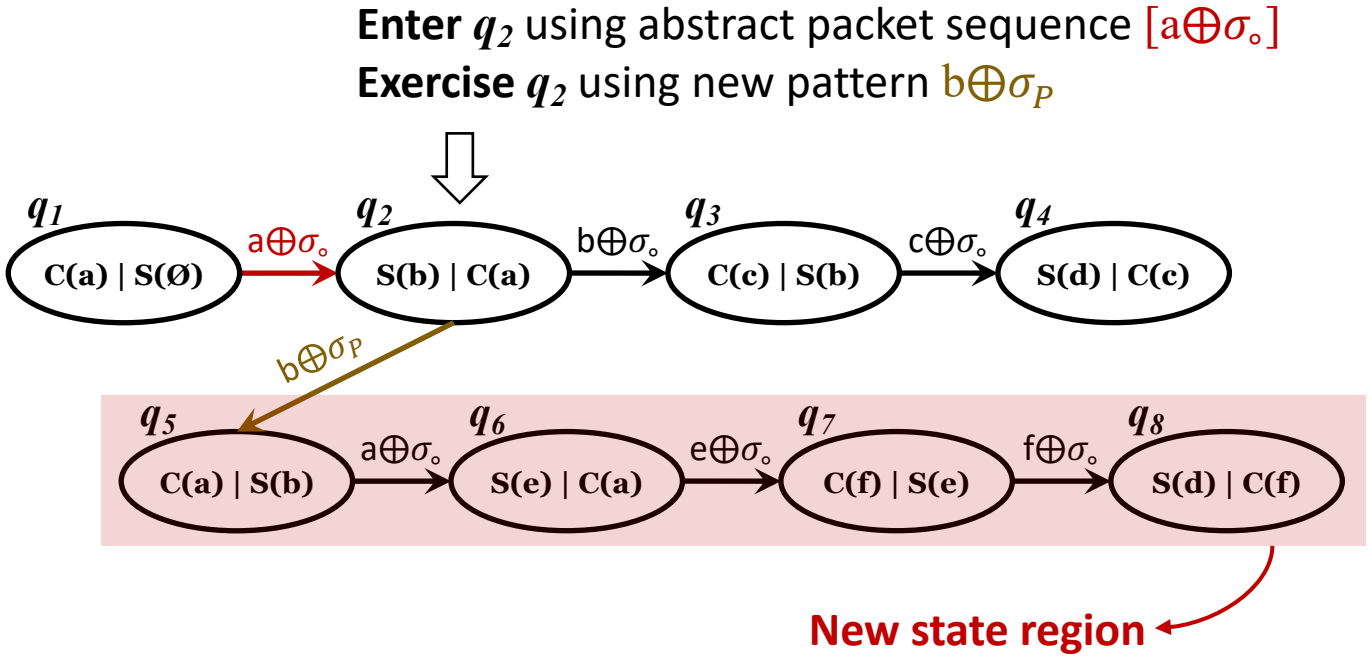




System State Tracking Graph

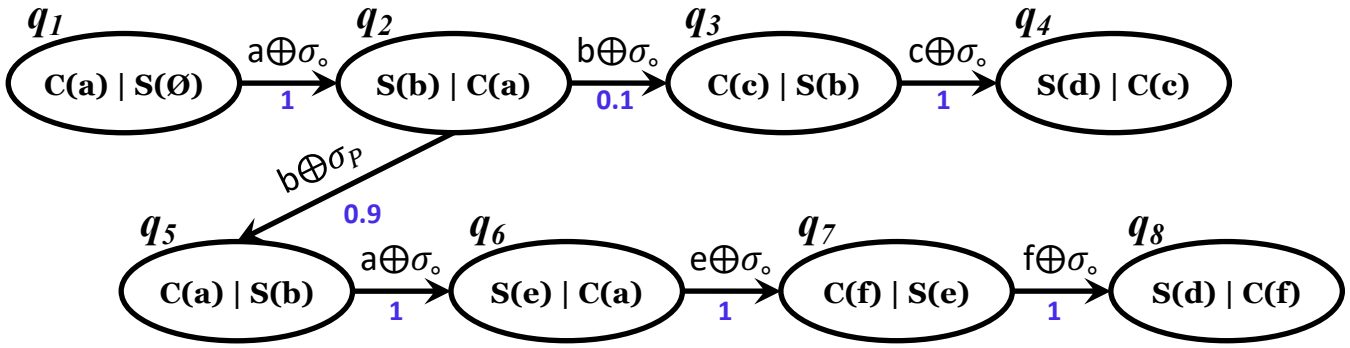
Guided Sequence Generation

- Stress test each SSTG state with **diverse inputs**



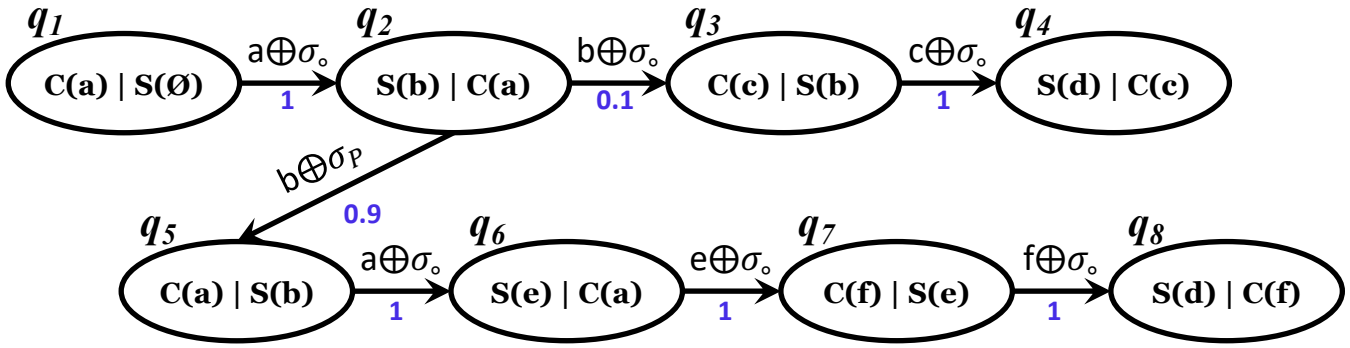
Guided Sequence Generation

- Stress test each SSTG state with **diverse inputs**
- Facilitate comprehensive SSTG traversal by **steering towards low-density regions**



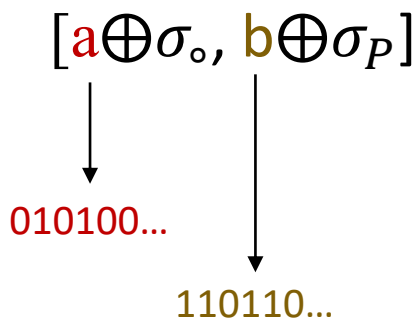
Guided Sequence Generation

- Stress test each SSTG state with **diverse inputs**
- Facilitate comprehensive SSTG traversal by **steering towards low-density regions**



Instantiation of Abstract Packet

- Utilize the intercepted packets exchanged between the client and server during the live session



Evaluation: Coverage Analysis

We use **branch coverage on the server side** for a fair comparison

BLEEM achieves **higher coverage** than existing protocol fuzzers

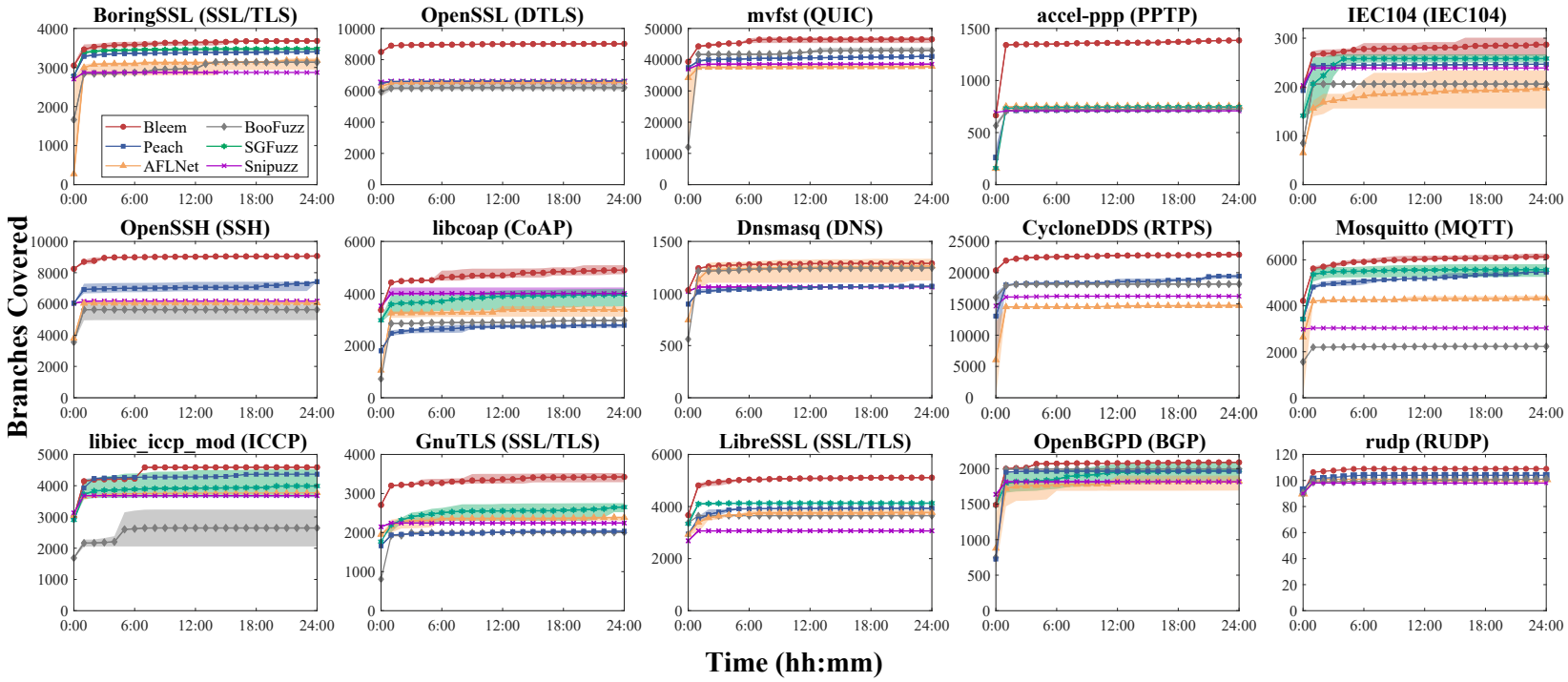
Peach (+28.5%)

BooFuzz (+48.9%)

SGFuzz (+23.4%)

AFLNet (+35.7%)

Snipuzz (+40.3%)



Evaluation: Guided Sequence Generation

Compare with BLEEMRand – a variant using random sequence selection

- BLEEM achieves **more complex SSTGs** and **5.7% more branches** than BLEEMRand

Subject	Fuzzer	SSTG Construction		SSTG Metrics			Branch Coverage
		Paths	$\overline{Len.}$	Types	Nodes	Trans.	
BoringSSL	BLEEMRand	42	4.11	32	72	84	4293
	BLEEM	75	3.82	**69**	**152**	**183**	**4549**
OpenSSL	BLEEMRand	266	6.84	73	247	397	10512
	BLEEM	256	5.96	**90**	**267**	**442**	**10614**
mvfst	BLEEMRand	2352	7.83	492	1494	2806	53942
	BLEEM	10781	7.99	**671**	**2779**	**6581**	**55575**
accel-ppp	BLEEMRand	101	6.07	**14**	**33**	**46**	1384
	BLEEM	30	4.62	11	25	31	**1385**
IEC104	BLEEMRand	39	5.57	84	113	137	279
	BLEEM	49	5.86	**88**	**149**	**164**	**321**
OpenSSH	BLEEMRand	43	5.23	20	59	82	12444
	BLEEM	97	5.58	**30**	**104**	**171**	**14579**
libcoap	BLEEMRand	10013	85.08	325	1340	3400	8292
	BLEEM	10286	83.42	**331**	**1427**	**4143**	**8530**
Dnsmasq	BLEEMRand	2958	22.60	**60**	**163**	**511**	1271
	BLEEM	1783	14.62	58	155	413	**1292**
CycloneDDS	BLEEMRand	55	4.55	18	67	132	22912
	BLEEM	139	4.26	**31**	**153**	**303**	**23710**
Mosquitto	BLEEMRand	19522	15.67	215	1037	2815	9284
	BLEEM	20652	13.09	**253**	**1085**	**3142**	**10285**
libiec_iccp_mod	BLEEMRand	116	9.05	28	96	155	6059
	BLEEM	314	9.71	**41**	**139**	**294**	**6265**
GnuTLS	BLEEMRand	50	4.66	25	70	98	5057
	BLEEM	57	4.12	**38**	**92**	**114**	**5222**
LibreSSL	BLEEMRand	209	3.93	67	248	**394**	5473
	BLEEM	196	3.78	**100**	**277**	385	**6157**
OpenBGPD	BLEEMRand	222	17.73	35	92	131	2072
	BLEEM	253	17.21	**43**	**111**	**169**	**2086**
rudp	BLEEMRand	149	5.86	21	82	151	112
	BLEEM	154	5.70	**30**	**109**	**185**	**115**

Evaluation: Guided Sequence Generation

Compare with BLEEMRand – a variant using random sequence selection

- BLEEM achieves **more complex SSTGs** and **5.7% more branches** than BLEEMRand
- SSTG complexity **correlates positively** with code coverage and packet type

Subject	Fuzzer	SSTG Construction		SSTG Metrics			Branch Coverage
		Paths	*Len.*	Types	Nodes	Trans.	
BoringSSL	BLEEMRand	42	4.11	32	72	84	4293
	BLEEM	75	3.82	**69**	**152**	**183**	**4549**
OpenSSL	BLEEMRand	266	6.84	73	247	397	10512
	BLEEM	256	5.96	**90**	**267**	**442**	**10614**
mvfst	BLEEMRand	2352	7.83	492	1494	2806	53942
	BLEEM	10781	7.99	**671**	**2779**	**6581**	**55575**
accel-ppp	BLEEMRand	101	6.07	**14**	**33**	**46**	1384
	BLEEM	30	4.62	11	25	31	**1385**
IEC104	BLEEMRand	39	5.57	84	113	137	279
	BLEEM	49	5.86	**88**	**149**	**164**	**321**
OpenSSH	BLEEMRand	43	5.23	20	59	82	12444
	BLEEM	97	5.58	**30**	**104**	**171**	**14579**
libcoap	BLEEMRand	10013	85.08	325	1340	3400	8292
	BLEEM	10286	83.42	**331**	**1427**	**4143**	**8530**
Dnsmasq	BLEEMRand	2958	22.60	**60**	**163**	**511**	1271
	BLEEM	1783	14.62	58	155	413	**1292**
CycloneDDS	BLEEMRand	55	4.55	18	67	132	22912
	BLEEM	139	4.26	**31**	**153**	**303**	**23710**
Mosquitto	BLEEMRand	19522	15.67	215	1037	2815	9284
	BLEEM	20652	13.09	**253**	**1085**	**3142**	**10285**
libiec_iccp_mod	BLEEMRand	116	9.05	28	96	155	6059
	BLEEM	314	9.71	**41**	**139**	**294**	**6265**
GnuTLS	BLEEMRand	50	4.66	25	70	98	5057
	BLEEM	57	4.12	**38**	**92**	**114**	**5222**
LibreSSL	BLEEMRand	209	3.93	67	248	**394**	5473
	BLEEM	196	3.78	**100**	**277**	385	**6157**
OpenBGPD	BLEEMRand	222	17.73	35	92	131	2072
	BLEEM	253	17.21	**43**	**111**	**169**	**2086**
rudp	BLEEMRand	149	5.86	21	82	151	112
	BLEEM	154	5.70	**30**	**109**	**185**	**115**

Evaluation: Guided Sequence Generation

Compare with BLEEMRand – a variant using random sequence selection

- BLEEM achieves **more complex SSTGs** and **5.7% more branches** than BLEEMRand
- SSTG complexity **correlates positively** with code coverage and packet type
- Total number of state traces is finite, indicating that our SSTG construction provision **prevents state space explosion**

Subject	Fuzzer	SSTG Construction		SSTG Metrics			Branch Coverage
		Paths	*Len.*	Types	Nodes	Trans.	
BoringSSL	BLEEMRand	42	4.11	32	72	84	4293
	BLEEM	75	3.82	**69**	**152**	**183**	**4549**
OpenSSL	BLEEMRand	266	6.84	73	247	397	10512
	BLEEM	256	5.96	**90**	**267**	**442**	**10614**
mvfst	BLEEMRand	2352	7.83	492	1494	2806	53942
	BLEEM	10781	7.99	**671**	**2779**	**6581**	**55575**
accel-ppp	BLEEMRand	101	6.07	**14**	**33**	**46**	1384
	BLEEM	30	4.62	11	25	31	**1385**
IEC104	BLEEMRand	39	5.57	84	113	137	279
	BLEEM	49	5.86	**88**	**149**	**164**	**321**
OpenSSH	BLEEMRand	43	5.23	20	59	82	12444
	BLEEM	97	5.58	**30**	**104**	**171**	**14579**
libcoap	BLEEMRand	10013	85.08	325	1340	3400	8292
	BLEEM	10286	83.42	**331**	**1427**	**4143**	**8530**
Dnsmasq	BLEEMRand	2958	22.60	**60**	**163**	**511**	1271
	BLEEM	1783	14.62	58	155	413	**1292**
CycloneDDS	BLEEMRand	55	4.55	18	67	132	22912
	BLEEM	139	4.26	**31**	**153**	**303**	**23710**
Mosquitto	BLEEMRand	19522	15.67	215	1037	2815	9284
	BLEEM	20652	13.09	**253**	**1085**	**3142**	**10285**
libiec_iccp_mod	BLEEMRand	116	9.05	28	96	155	6059
	BLEEM	314	9.71	**41**	**139**	**294**	**6265**
GnuTLS	BLEEMRand	50	4.66	25	70	98	5057
	BLEEM	57	4.12	**38**	**92**	**114**	**5222**
LibreSSL	BLEEMRand	209	3.93	67	248	**394**	5473
	BLEEM	196	3.78	**100**	**277**	385	**6157**
OpenBGPD	BLEEMRand	222	17.73	35	92	131	2072
	BLEEM	253	17.21	**43**	**111**	**169**	**2086**
rudp	BLEEMRand	149	5.86	21	82	151	112
	BLEEM	154	5.70	**30**	**109**	**185**	**115**

Evaluation: Bug-Detection Capability

Open-Source Targets

- **15** new bugs, only 9 of which can be exposed by existing tools
- **10** CVEs assigned

Protocol	Subject	Information	CVE ID
TLS/SSL	LibreSSL	Stack buffer overflow in the x509_constraints_parse_mailbox in libcrypto/x509/x509_constraints.c	CVE-2021-41581
TLS/SSL	GnuTLS	MD_UPDATE does not prohibit zero-length input from illegal address, causing null pointer dereference	CVE-2021-4209
TLS/SSL	BoringSSL	The server tries to write to a socket that has been shut down even after the client has sent TLS Alert	-
PPTP	accel-ppp	Stack-based out-of-bounds read in post_msg when processing a call_clear_request	CVE-2021-42870
PPTP	accel-ppp	Stack-based out-of-bounds read in the server if the client exits after authentication	CVE-2021-42054
PPTP	accel-ppp	Memory allocated to pool in ippool_init2 is not free when exits, which can cause a denial of service	-
IEC104	IEC104	Stack buffer overflow in the parameter Iec10x_Sta_Addr	CVE-2020-20486
IEC104	IEC104	Segmentation violation in the Iec104_Deal_FirmUpdate function	CVE-2020-18731
UDP	rdup	Memory leak in the rudp_delete function because memory allocated rudp is not freed	CVE-2020-20665
ICCP	libiec_iccp_mod	Heap buffer overflow in the ByteStream_readOctets in common/byte_stream.c	CVE-2020-20490
ICCP	libiec_iccp_mod	Heap buffer overflow in the parseIncomingMessage in mms/iso_cotp/cotp.c	CVE-2020-20662
ICCP	libiec_iccp_mod	Heap buffer overflow in the MmsConnection_create in mms_client_connection.c	CVE-2020-20663
BGP	OpenBGPD	Undefined behavior, a null pointer is passed as the first argument of memcpy in imsg.c	-
BGP	OpenBGPD	Undefined behavior, an incorrect bitwise shift in imsg.c	-
QUIC	mvfst	Heap buffer overflow in the EchoServerTransportFactory::make function in EchoServer.h	-
		SUM	10 CVEs

Closed-Source Targets

- **Best CVE discovery performance** on vulnerable firmware in blackbox setting

CVE ID	Protocol	Snipuzz	BooFuzz	Peach	BLEEM
CVE-2018-5767	HTTP	-	25min	34min	6min
CVE-2020-25067	UPnP	26min	36s	47s	33s
CVE-2019-14457	HTTP	-	-	652min	35min
CVE-2019-1663	HTTP	-	501min	307min	72min

Summary

- BLEEM provides a scalable feedback mechanism by analyzing the system output sequence noninvasively
- BLEEM supports guided fuzzing by resorting to state-space tracking that encompasses all parties timely
- BLEEM leverages interactive traffic to generate protocol-logic-aware packet sequences
- BLEEM outperforms the state-of-the-art and is flexible to apply to diverse protocol implementations even in a black-box setting.