Northwestern Lab for Internet and Security Technology

NetShield: Towards High Performance Network-based Vulnerability Signature Matching

Zhichun Li, Gao Xia, Hongyu Gao, Yi Tang, Yan Chen, and Bin Liu
Northwestern University and Tsinghua University (China)

Problems

Currently, the regular expressions used by NIDS for signature matching have low accuracy because fundamentally regex cannot capture the vulnerability condition well. On the other hand, vulnerability signatures are much more accurate, but may have performance problems.

	Regular Expression	Vulnerability
Accuracy	Relative Poor	Much Better
Speed	Good	??
Memory	OK	??
Coverage	Good	??

Shield [sigcomm'04]

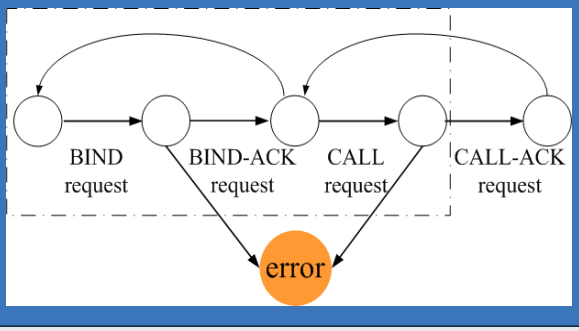
Focus of this work

Goal: Build a high speed vulnerability signature based IDS!

Our approach

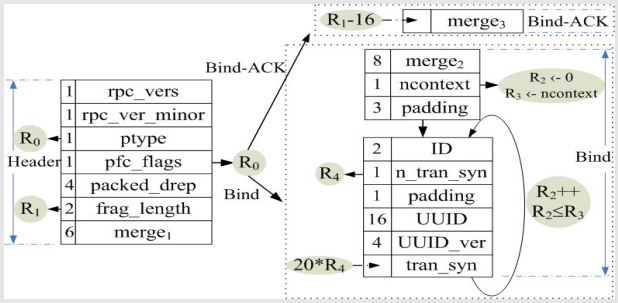
Signature Example

```
BIND:
  rpc_vers==5 && rpc_vers_minor==1
  && packed_drep==\x10\x00\x00\x00
  && context[0].abstract_syntax.uuid
  ==UUID_RemoteActivation
BIND-ACK:
  rpc_vers==5 && rpc_vers_minor==1
CALL:
  rpc_vers==5 && rpc_vers_minor==1
  && packed_drep==\x10\x00\x00\x00
  && stub.RemoteActivationBody.actual_length
  >=40 && matchRE(stub.buffer,/^\x5c\x00\x5c\x00/)
```

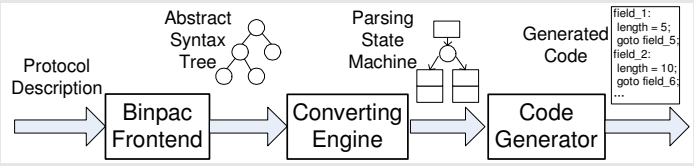


High speed parsing

Lightweight parsing state machine (Simplified WinRPC example)



Automatic parser generation



High speed matching

Tabular representation of signatures

RuleID	RB	Matcher 1 Method ==	Matcher 2 Filename ==	Matcher 3 Filename RE	Matcher 4 VARS ==RE	Matcher 5 Headers ==LEN
1	1	DELETE	*	*	*	*
2	1	TRACE	*	*	*	*
3	1	POST	header.php	*	*	*
4	2	*	ads.cgi	*	name="file"; value ~ ".*\.\./"	*
5	2	*	awstats.pl	*	name="configdir"; value ~ ".*7C"	*
6	2	*	fp40reg.dll	*	*	name="host"; len(value)>300
7	3	*	*	.*\.id[aq]$	*	*
8	4	*	*	*	name="name"; value ~ ".*GLOBAL"	*
9	5	*	*	*	*	name="User-Agent"; len(value)>512

Candidate Selection algorithm

- Pre-computation decides the rule order and matcher order. Given that most matchers are good rule filters, we only keep track of a few matching candidates for one connection.
- For each matcher, match rules in parallel.
- Iteratively combine the candidate sets for multiple matchers.

Evaluation

- High speed parsing: 3.6~19.9 Gbps for different protocols (HTTP, WINRPC, DNS)
- High speed matching: HTTP, 791 vulnerability signatures at ~2Gbps
- Multi-core implementation further boosts the throughput to ~11Gbps
- Prototype was deployed on live-network and achieves ***higher accuracy and speed*** than Snort
- Memory usage: 2.3MB for HTTP matching structure and no more than 27B per connection