Stateful Firewalls

Hank and Foo

Types of firewalls

- Packet filter (stateless)
- Proxy firewalls
- Stateful inspection
- Deep packet inspection

Packet filter (Access Control Lists)

- Treats each packet in isolation
- Operates at network layer (layer 3) of the OSI model
- Filters based on header information in packet (e.g. src/dst IP address, port)

- Advantage is speed, application independence, scalability
- Easy to trick – spoofing, fragmenting, etc

Proxy firewalls

- Client doesn’t actually communicate directly with server
- Proxy receives request from client and makes a request to server and returns information to client
- It can filter the request from client and filter information returned to client
- Considered application layer filter
- Slower than packet filter, but more secure
- Another disadvantage: application specific

Stateful inspection

- Deals with the *state of connections*
- *State* here is vaguely defined as “the condition of the connection”, which varies greatly depending on application/protocol used
- Stores the states of legitimate connections in a *state table* (state information usually stored as hash to make matching faster)
- Filters packets by matching to valid states in the state table
- Usually takes more time during setup of a new connection (application layer inspection performed usually only at setup), compared to after

Possible state information

- Src/dst IP address, ports
- Protocol, flags, sequence, acknowledge numbers
- ICMP code and type numbers
- Secondary connection information communicated in application layer headers
- Application layer specific command sequences (GET, PUT, OPTIONS, etc)

How it works

- Spends most of the time examining packet information in transport layer (layer 4) and lower
- Can examine application layer information (layer 7), usually during new connection setup
- If new packet is permitted based on firewall rules/security policy, a new entry is added in the state table
- After new connection is setup, because later packets match an entry in the state table, there is no need for application layer inspection

Advantages

- More secure than basic packet filtering
- Faster than proxy firewalls
- Performs application layer filtering to a certain degree (e.g. FTP session)

- E.g. *iptables* classifies each packet as either NEW, ESTABLISHED, RELATED, INVALID
- For FTP protocol, a control connection is first established
- When data is transferred, separate connection is established, and *iptables* will knowingly classify the first packet as RELATED instead of NEW

Disadvantages

- Possibly less secure than proxy firewalls (does not perform true content filtering)
  1. Abbreviated application-level inspection (e.g. application-level inspection of initializing packet only allows for malicious application-level behavior in subsequent packets)
  2. Lack full application support (e.g. monitors FTP session for *port* command, but lets other non-FTP traffic pass through FTP port)
- Slower than basic packet filtering
- Vulnerable to new attacks (e.g. SYN flood – overflows state table so no new connection can be made)

TCP

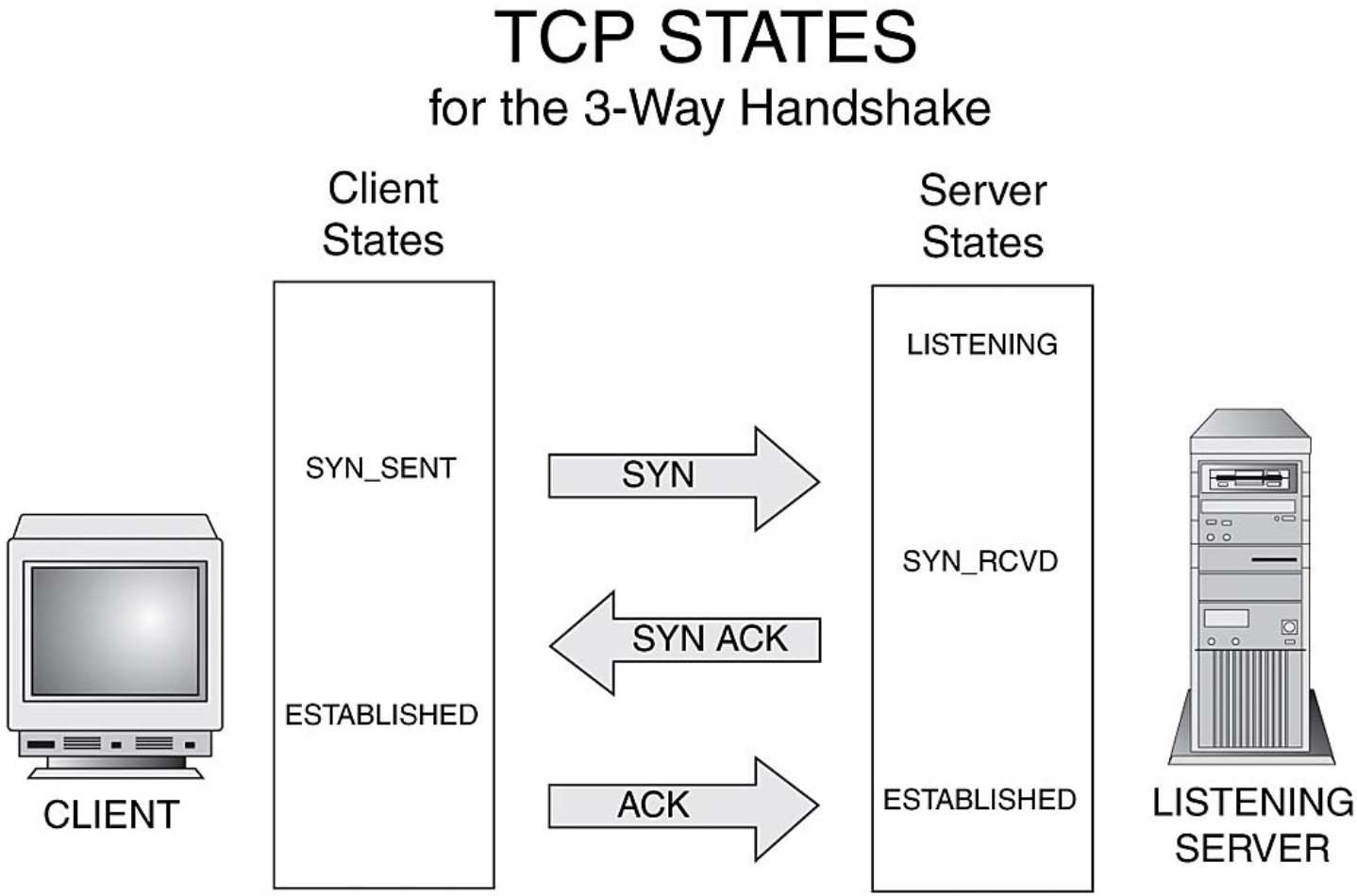
Connection-oriented protocol

1. Beginning/end of a session is well defined
2. State of connections tracked with flags

Therefore considered a stateful protocol

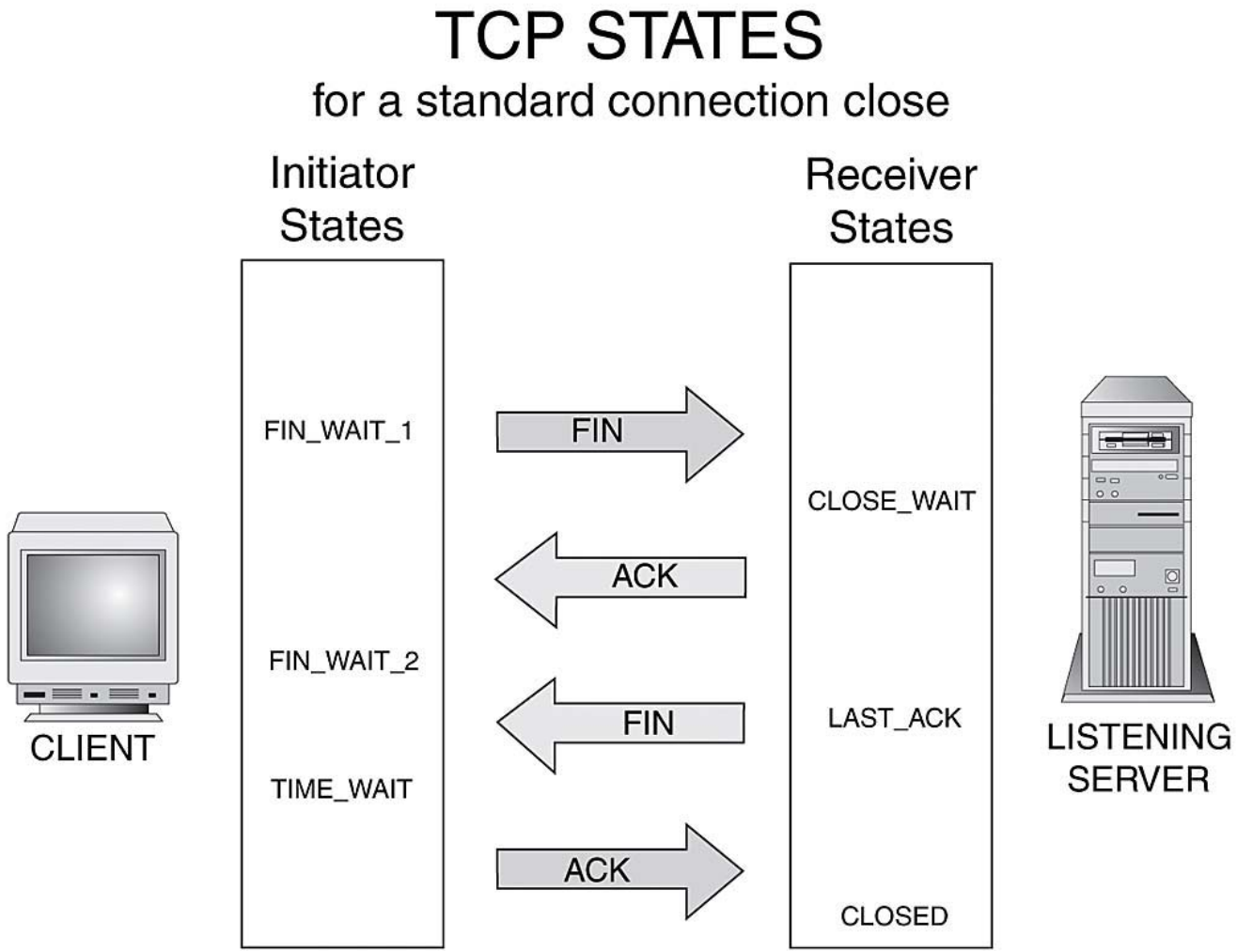
The connection can be in 1 of 11 states, as defined in RFC 793

Establishing TCP connection



Obtained from *Inside Network Perimeter Security: Stateful Firewalls*

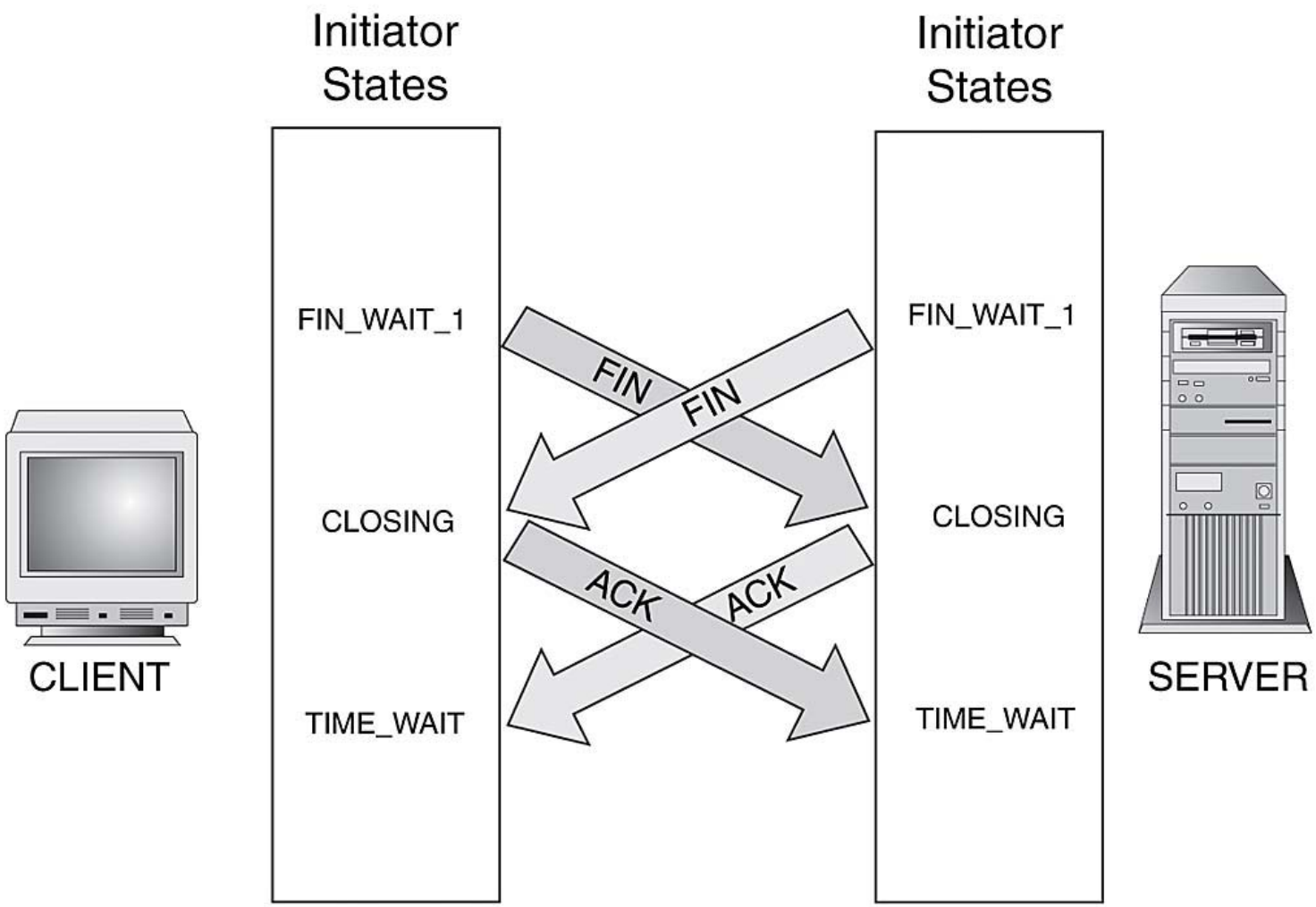
Tearing down TCP connection



Obtained from *Inside Network Perimeter Security: Stateful Firewalls*

TCP STATES

for a simultaneous connection close



Obtained from *Inside Network Perimeter Security: Stateful Firewalls*

UDP

- Connectionless transport protocol have no defined state
- Pseudo-stateful tracking
- UDP has no sequence numbers or flags
- So IP addresses and port numbers used
- Ephemeral ports are somewhat random, differ for different connections from same IP
- No set method for connection teardown, so timeout value used to remove entries in state table

UDP

- Cannot correct communication issues by itself, relies entirely on ICMP for error handling
- Therefore ICMP also important when tracking UDP states
- E.g. Host 2 may send a ICMP source quench message to host 1 to slow down transmission, firewall must know that this ICMP message is related to the UDP session

ICMP

- Like UDP, not stateful protocol
- ICMP sometimes used in a request/reply format (e.g. ping echo request, echo reply)
- This can be tracked
- For one-way ICMP messages (like error messages) that are precipitated by messages from other protocols, it is more difficult

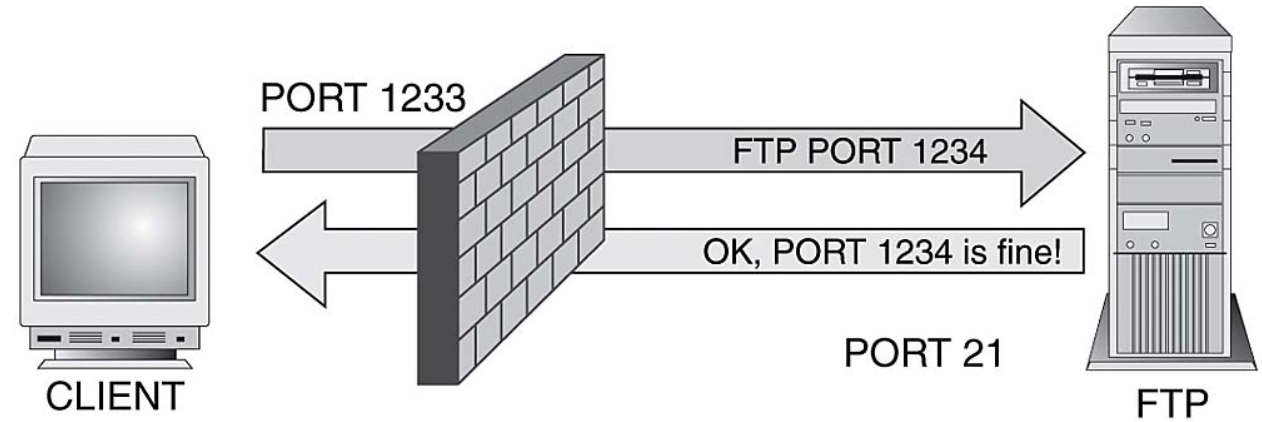
HTTP

- HTTP uses TCP in a simple manner, easy to track the state
- Can also do track application-level commands like GET

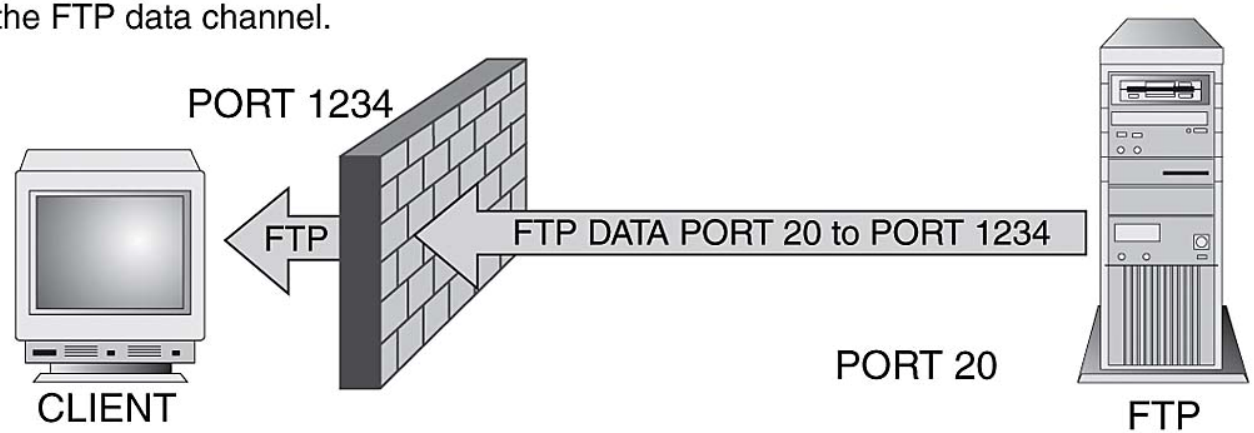
FTP

- Uses the TCP protocol in a nonstandard way
- Stateful firewall with no knowledge of FTP will not pass FTP traffic
- Because control and data connections are separate TCP sessions

FTP



The stateful firewall recognizes the FTP port command and records its value to securely facilitate the creation of the FTP data channel.



When the FTP data connection is initialized by the server, the firewall recognizes the IP address and port combination used and allows the inbound connection to pass.

The port number used by the server initializing the data channel is actually sent to it in an FTP port command from the client, which is why application-level inspection is needed here

Obtained from *Inside Network Perimeter Security: Stateful Firewalls*

Examples of stateful firewalls

- **Check Point Firewall-1** – Check Point Software Technologies Ltd (they coined the term *stateful inspection* and patented it)
- **Cisco PIX** – Cisco Systems Inc
- **iptables** (and netfilter) – Included in all modern linux distributions

Stateful inspection is implemented differently by different vendors

iptables

- Admins create rules specifying what protocols or specific traffic types should be tracked
- Basic state table entry contains
  - The protocol being used for the connection
  - The source and destination IP addresses
  - The source and destination ports
  - A listing with source and destination IP addresses and ports reversed (to represent response traffic)
  - The time remaining before the rule is removed
  - The TCP state of the connection (for TCP only)
  - The connection-tracking state of the connection

Sample state table entry

- tcp 6 93 SYN_SENT src=192.168.1.34 dst=172.16.2.23 sport=1054 dport=21 [UNREPLIED] src=172.16.2.23 dst=192.168.1.34 sport=21 dport=1054 use=1
- [protocol name] [protocol number] [timeout] [state] [src ip] [dst ip] [src port] [dst port (initial connection tagged UNREPLIED)] [return src ip] [return dst ip]

- tcp 6 41294 ESTABLISHED src=192.168.1.34 dst=172.16.2.23 sport=1054 dport=21 src=172.16.2.23 dst=192.168.1.34 sport=21 dport=1054 [ASSURED] use=1
- After connection established, timeout increased greatly

Basic rules

- iptables -A OUTPUT -p tcp -m state --state NEW,ESTABLISHED -j ACCEPT
  - -A: append to existing rules
  - OUTPUT: for output traffic
  - -p tcp: for tcp protocol
  - -m state: use *state* module
  - -j ACCEPT: parameter to accept such traffic
- All NEW and ESTABLISHED traffic allowed out, which means no outbound traffic disallowed by this rule

Basic rules

- iptables –A INPUT -p tcp -m state --state ESTABLISHED -j ACCEPT
- Only return traffic allowed into network

- For UDP, just change previous rules to ‘–p udp’
- Same for ICMP (-p icmp), but also add RELATED

- New modules can be added when new protocols used

Deep packet inspection

- Basically stateful inspection but with visibility into the application layer
- Not just keeps track of connection information, but looks at the data too (i.e. content filtering)

- Simply a stateful firewall with limited IDS capabilities built in (NOTHING NOVEL)

Firewall clustering for scalability

Two general ways to use multiple firewalls

1. Single shared state table, possibly with a dedicated and fast communication channel between firewalls
2. Guarantee packets from the same connection reach the same firewall (using load balancers)

References

- http://dmiessler.com/study/firewalls
- http://www.wikipedia.org
- http://www.quepublishing.com/articles/article.asp?p=373431&seqNum=1 Sample chapter from book *Inside Network Perimeter Security: Stateful Firewalls*

Packet filtering/classification

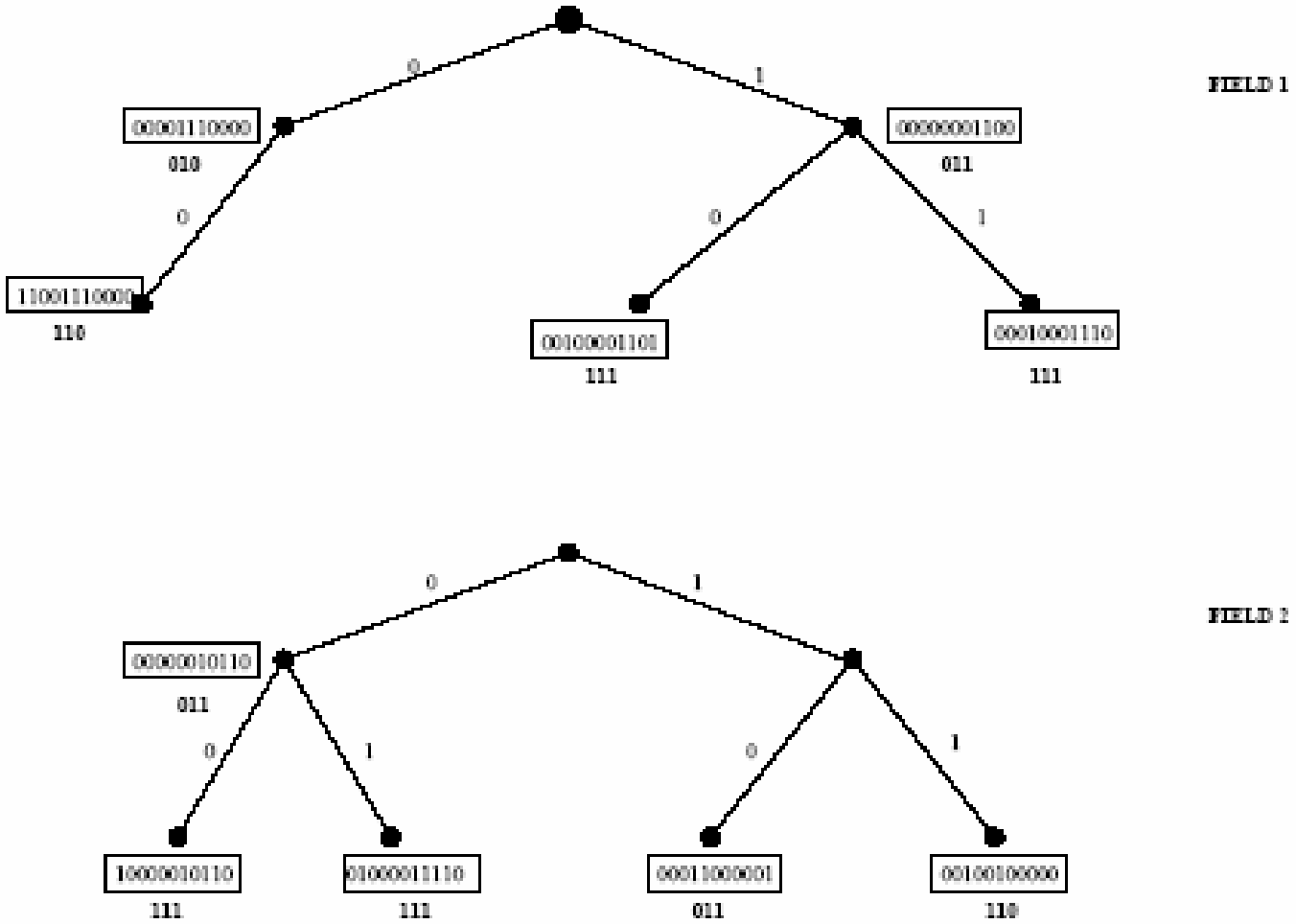
Given packet P with k fields, and N rules, find rules that P matches to.

Many different ways to do this, one way is through bit vectors.

Presented here is Aggregated Bit Vector Scheme, which builds on the Lucent Bit Vector Scheme which is Nk/w memory accesses, where w is the size of a word in memory

$Rule$	$Field_1$	$Field_2$
F_0	$00*$	$00*$
F_1	$00*$	$01*$
F_2	$10*$	$11*$
F_3	$11*$	$10*$
F_4	$0*$	$10*$
F_5	$0*$	$11*$
F_6	$0*$	$0*$
F_7	$1*$	$01*$
F_8	$1*$	$0*$
F_9	$11*$	$0*$
F_{10}	$10*$	$10*$

Figure 1: A simple example with 11 rules on two fields.


FIELD 1
0
1
00001110000
010
00000001100
011
0
0
1
11001110000
110
00100001101
111
00010001110
111
FIELD 2
0
1
00000010110
011
0
1
0
1
10000010110
111
01000011110
111
00011000001
011
00100100000
110
Aggregate Size = 4

References

- Baboescu and Varghese, “Aggregated Bit Vector Search Algorithms for Packet Filter Lookups”, http://citeseer.ist.psu.edu/cache/papers/cs/27575/http:zSzzSzwww-cse.ucsd.eduzSz~baboescuzSzresearchzSzlookup.pdf/aggregated-bit-vector-search.pdf