Firewall Design: Consistency, Completeness, Compactness

Alex X. Liu
alex@cs.utexas.edu

Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188, U.S.A.

March, 2004

Co-author: Mohamed G. Gouda
Firewall is a sequence of rules that allow packets that go through it to be accepted (and so proceed) or discarded.
Firewall Example

\[ r_0 : I = 0 \land S = any \land D = s \land P = tcp \land T = 25 \rightarrow a \]
\[ r_1 : I = 0 \land S = any \land D = s \land P = any \land T = any \rightarrow d \]
\[ r_2 : I = 0 \land S = m \land D = any \land P = any \land T = any \rightarrow d \]
\[ r_3 : I = 1 \land S = h \land D = any \land P = any \land T = any \rightarrow a \]
\[ r_4 : I = 1 \land S = any \land D = any \land P = any \land T = any \rightarrow a \]
Three Firewall Design Problems

- Inconsistency
- Incompleteness
- Incompactness
Inconsistency Problem

\[ r_0 : I = 0 \land S = \text{any} \land D = s \land P = tcp \land T = 25 \rightarrow a \]
\[ r_1 : I = 0 \land S = \text{any} \land D = s \land P = \text{any} \land T = \text{any} \rightarrow d \]
\[ r_2 : I = 0 \land S = m \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow d \]
\[ r_3 : I = 1 \land S = h \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow a \]
\[ r_4 : I = 1 \land S = \text{any} \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow a \]

- This firewall accepts all incoming SMTP packets even those originated at the malicious host \( m \)
- Solution: placing rule \( r_2 \) at the beginning of the sequence of rules ahead of rule \( r_1 \)
Incompleteness Problem

\[ r_0 : I = 0 \land S = m \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow d \]
\[ r_1 : I = 0 \land S = \text{any} \land D = s \land P = \text{tcp} \land T = 25 \rightarrow a \]
\[ r_2 : I = 0 \land S = \text{any} \land D = s \land P = \text{any} \land T = \text{any} \rightarrow d \]
\[ r_3 : I = 1 \land S = h \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow a \]
\[ r_4 : I = 1 \land S = \text{any} \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow a \]

- Any packet where \( I = 0 \), \( S \neq m \), and \( D \neq s \) does not satisfy the predicate of any of the five rules \( r_0 \) through \( r_4 \).
- Solution: add the following rule immediately before rule \( r_3 \)

\[ I = 0 \land S = \text{any} \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow a \]
Incompactness Problem

\[ r_0 : I = 0 \land S = m \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow d \]
\[ r_1 : I = 0 \land S = \text{any} \land D = s \land P = \text{tcp} \land T = 25 \rightarrow a \]
\[ r_2 : I = 0 \land S = \text{any} \land D = s \land P = \text{any} \land T = \text{any} \rightarrow d \]
\[ r_3 : I = 0 \land S = \text{any} \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow a \]
\[ r_4 : I = 1 \land S = h \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow a \]
\[ r_5 : I = 1 \land S = \text{any} \land D = \text{any} \land P = \text{any} \land T = \text{any} \rightarrow a \]

- Rule \( r_4 \) can be removed without affecting the set of packets accepted by the firewall and the set of packets discarded by the firewall.
- Solution: remove rule \( r_4 \)
How to design a consistent, complete, and compact firewall?

- Start with a Firewall Decision Diagram (FDD)
- Apply 5 algorithms to the FDD
  - reduction
  - marking
  - generation
  - compaction
  - simplification
Firewall Decision Diagram

- **Consistency**: for any two outgoing edges of a node, their labels are non-overlapping.
- **Completeness**: the union of the labels of all the outgoing edges of a node is the domain of the label of the node.
Rules from FDD

- Each complete path in FDD can be represented by one rule
- For example:

\[ F_0 \in [4, 7] \land F_1 \in [2, 3] \cup [5, 7] \rightarrow a \]

- the number of rules = the number of decision paths
FDD Reduction

- Collapse “parallel edges” into one edge
- Collapse “isomorphic nodes” into one node

![Diagram showing FDD Reduction with nodes and edges labeled with numbers and relations.

PSfrag replacements:
F_0
F_1
PSfrag replacements:
F_0
F_1

Alex X. Liu
Firewall Design – p.11
Choose one outgoing edge to mark ALL
FDD Marking Algorithm

- For each node in FDD, choose one outgoing edge to mark ALL
Firewall Generation

- Generate one rule for each complete path
- Use the mark “ALL”, instead of the specified set
- Order the rules properly

\[
r = \begin{cases} 
  F_0 \in [4, 7] \land F_1 \in [2, 3] \cup [5, 7] & \rightarrow a, \\
  F_0 \in [4, 7] \land F_1 \in \text{ALL} & \rightarrow d, \\
  F_0 \in \text{ALL} \land F_1 \in [0, 9] & \rightarrow d,
\end{cases}
\]
Firewall Compaction

- A rule is redundant in a firewall iff the rule can be removed without affecting the semantics of the firewall.

- Theorem: A rule $r_i$ in firewall $(r_0, \cdots, r_{m-1})$ is redundant iff for each $j, i < j \leq m - 1$, at least one of the following two conditions holds:
  1. \langle decision \rangle of $r_j = \langle decision \rangle$ of $r_i$.
  2. No packet over the fields $F_0, \cdots, F_{n-1}$ satisfies the predicate $r_i.op \land (\neg r_{i+1}.ep \land \cdots \land \neg r_{j-1}.ep) \land r_j.ep$
     where $r_i.op$ denotes the original predicate of $r_i$ and $r_j.ep$ denotes the exhibited predicate of $r_j$.

- Firewall compaction algorithm is in paper
Firewall Compaction Example

\[ r = ( F_0 \in [4, 7] \land F_1 \in [2, 3] \cup [5, 7] \rightarrow a, \\
F_0 \in [4, 7] \land F_1 \in \text{ALL} \rightarrow d, \\
F_0 \in \text{ALL} \land F_1 \in [0, 9] \rightarrow d, \\
) \]

■ The second rule is redundant.

\[ r' = ( F_0 \in [4, 7] \land F_1 \in [2, 3] \cup [5, 7] \rightarrow a, \\
F_0 \in \text{ALL} \land F_1 \in [0, 9] \rightarrow d, \\
) \]
Firewall Simplification

- A firewall rule of the form

\[ F_0 \in S_0 \land \cdots \land F_{n-1} \in S_{n-1} \rightarrow \langle \text{decision} \rangle \]

is called simple iff every \( S_i \) in the rule is either the ALL mark or a single interval.

- \( r' = ( F_0 \in [4, 7] \land F_1 \in [2, 3] \cup [5, 7] \rightarrow a, \)
  \( F_0 \in ALL \land F_1 \in [0, 9] \rightarrow d, \)

- \( r'' = ( F_0 \in [4, 7] \land F_1 \in [2, 3] \rightarrow a, \)
  \( F_0 \in [4, 7] \land F_1 \in [5, 7] \rightarrow d, \)
  \( F_0 \in ALL \land F_1 \in [0, 9] \rightarrow d, \)
Summary of Firewall Design

A user specified FDD $f$

Algorithm 1

A reduced FDD $f'$

Algorithm 2

A marked FDD $f''$

Algorithm 3

A generated firewall $r$

Algorithm 4

A compact firewall $r'$

Algorithm 5

A simple firewall $r''$