

Specifying Security Selection Policies Using SCIPAD

Presented By

Jim Hoagland

hoagland@cs.ucdavis.edu

Research With

Dr. Baiju Patel

Intel Architecture Labs, Intel Corporation

baiju@mailbox.jf.intel.com

Security Lab Seminar, 28 April 1999

1 of 26

■
James A. Hoagland
Department of Computer Science
University of California, Davis
hoagland@cs.ucdavis.edu

Outline

- Introduction
- SCIPAD
- Composing SCIPAD policies
- Using SCIPAD
- Conclusion

2 of 26

■
James A. Hoagland
Department of Computer Science
University of California, Davis
hoagland@cs.ucdavis.edu

Jim's Policy Languages

LaSCO

- work with Raju Pandey and Karl Levitt
- primary dissertation policy language
- access control policy language for (possibly) distributed systems

SCIPAD (presented here)

- work with Baiju Patel
- developed while interning at Intel Corporation summer '97
- selection policy language for indicating what measures to take in different situations

3 of 26

■
James A. Hoagland
Department of Computer Science
University of California, Davis
hoagland@cs.ucdavis.edu

Motivating Problem

- we are provided security goodies for communications
- this is good
- but, ... when do we use them?
- and how?
- what are our security needs?

- these questions are answered by security policy
- how do we state this policy conveniently and suitably for automated use?

4 of 26

■
James A. Hoagland
Department of Computer Science
University of California, Davis
hoagland@cs.ucdavis.edu

Policy Use in Communications

Some roles of policy in communications:

- ❑ decide how to use secure communications protocol options such that policy is upheld
- ❑ select a proposal (protocol configuration) from several offered
- ❑ determine if policy is violated
- ❑ explore situations permitted by policy under a set of protocol options

5 of 26

Outline

- ✓ Introduction
- ➔ SCIPAD
- ❑ Composing SCIPAD policies
- ❑ Using SCIPAD
- ❑ Conclusion

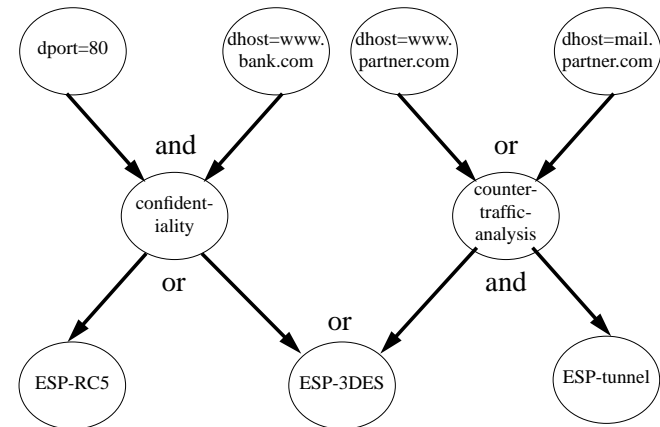
6 of 26

Our Approach

- ❑ SCIPAD (Security Concepts In a Policy Augmented DAG) is a language to state selection policies
 - i.e., it can specify what protocol options to use under different situations
- ❑ SCIPAD specifies the policy in terms of a DAG
 - nodes represent security concepts
 - edges group these concepts together
- ❑ designed-in ability to compose separately specified policies together into an overall policy

7 of 26

Example Policy Over IPSEC



8 of 26

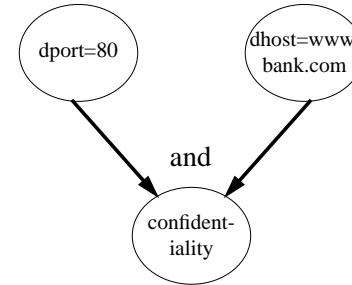
SCIPAD Overview

- nodes represent security concepts
 - a root node represents an aspect of the domain (i.e. that the destination port is 80)
 - a leaf node represents an aspect (primitive) of the security-providing environment (i.e., a security mechanism)
 - an interior node can represent some arbitrary security-relevant notion in between
- directed edges show what nodes are parents and children
- “and” and “or” annotations of a node for its parents and for its children determine the nature of relationship
- set of domain aspects is a *situation*
- set of primitives is a *configuration* or *proposal*

9 of 26

Semantics: Parents of a Node

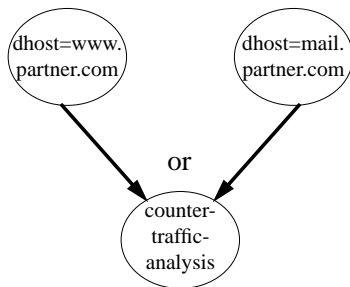
Parents of a node state when the node is needed



“and” means we need all the parents

10 of 26

Semantics: Parents of a Node

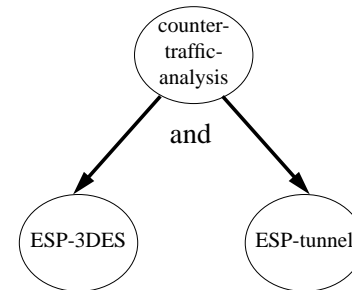


“or” means that one of the parents is sufficient for the node to be relevant

11 of 26

Semantics: Children of a Node

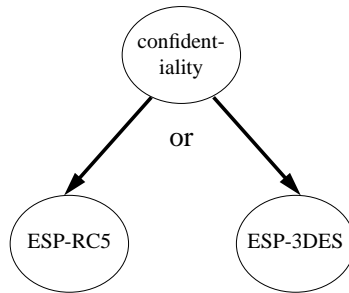
Children of a node state how to achieve the node’s concept



“and” means we need all the children

12 of 26

Semantics: Children of a Node



“or” means only we only need one child to enforce the parent

13 of 26

Node Meaning

The meaning of the security concept a node represents is independent of SCIPAD

- SCIPAD's only “knowledge” of these concepts is what the parents and children are and how the “and”s and “or”s relate them
- interpretation of nodes is left to its users
- this enables SCIPAD to be used in different ways for applications

14 of 26

Outline

- ✓ Introduction
- ✓ SCIPAD
- Composing SCIPAD policies
- Using SCIPAD
- Conclusion

15 of 26

Composing Policies

Composition is bringing together separately specified policies

- horizontal composition relates peer policies
- vertical composition connects parts of a policy stated at different levels

16 of 26

Vertical Composition

Policy levels example

- IPSEC WG says what the packet format is for IPSEC concepts
- a security expert says how to achieve security properties using these concepts
- site has policy for when different security properties are required
- user and application determines when aspects of site's policy domain is relevant

These are part of an overall policy but might be separately specified.

- separate specification is useful since each specifier might "think" in terms of different input and output primitives

17 of 26

Rules for SCIPAD Composition

- nodes representing same concept get merged
- independently specified groups of children of a node are combined with "or" semantics
- independently specified groups of parents to a node are combined with "and" semantics
- sometimes intermediate nodes need to be introduced

18 of 26

Outline

- ✓ Introduction
- ✓ SCIPAD
- ✓ Composing SCIPAD policies
- ➔ Using SCIPAD
- Conclusion

19 of 26

6 Things to do with a SCIPAD

Under a SCIPAD policy:

- given a situation
 - generate a requirements expression for a situation
 - find sets of acceptable configurations for a situation
- given a situation and a configuration
 - find acceptability of the primitive set for the situation
- given a configuration
 - find sets of situations where configuration is acceptable
- manipulation
 - split a SCIPAD into sub-PADs where only one configuration may result

Given multiple SCIPAD policies:

- given a situation
 - find mutually acceptable primitive sets

20 of 26

Deriving Acceptable Proposals

Apply a situation to a policy to obtain a requirements expression

- determine relevance of each node to the given situation
- ignore non-relevant nodes and parent-combining ands and ors
- combine resulting trees with “and”
- requirements expression is the corresponding logical “and”s and “or”s in terms of primitives

From the requirements expression, derive proposals

- determine the relevant solutions to the expression
- each solution is a primitive set
- each primitive set describes a state of the security environment
- determine proposals from the primitive sets
- what is done with the proposals is protocol-specific

21 of 26

Finding Mutually Acceptable Proposals

- each side derives a set of acceptable proposals for its policy and the situation
- find the intersection of these sets

Alternately, check a proposal against a requirements expression

22 of 26

Other Possible Uses of SCIPAD

Broader utility of SCIPAD promoted by independence of language and application

- response policies
- IDS configuration
- ...

23 of 26

Outline

- ✓ Introduction
- ✓ SCIPAD
- ✓ Composing SCIPAD policies
- ✓ Using SCIPAD
- Conclusion

24 of 26

Current Status

- ❑ basic syntax and semantics have been defined
- ❑ text notation for SCIPAD policies has been defined
- ❑ SCIPAD interpreter implemented in Perl
- ❑ interface to ISAKMP in progress (?)
- ❑ some work on primitive prioritization (preference for proposals) completed

25 of 26

Future Work

- ❑ explore other uses for SCIPAD
- ❑ more detailed semantics for some operations
- ❑ more detailed guidelines for writing primitive and domain aspect definitions
- ❑ further work on primitive set prioritization
- ❑ making the graphs smaller for efficiency
 - transitive and single edge compaction
 - application of partial situations

26 of 26