Responding to Policies at Runtime in TrustBuilder

Michael D. Jones
mike.jones@byu.edu

Kent E. Seamons

See next page for additional authors

Follow this and additional works at: https://scholarsarchive.byu.edu/facpub

Part of the Computer Sciences Commons

Original Publication Citation

BYU ScholarsArchive Citation
Jones, Michael D.; Seamons, Kent E.; and Smith, Bryan, "Responding to Policies at Runtime in TrustBuilder" (2004). All Faculty Publications. 1285.
https://scholarsarchive.byu.edu/facpub/1285

This Presentation is brought to you for free and open access by BYU ScholarsArchive. It has been accepted for inclusion in All Faculty Publications by an authorized administrator of BYU ScholarsArchive. For more information, please contact scholarsarchive@byu.edu, ellen_amatangelo@byu.edu.
Responding to Policies at Runtime in TrustBuilder

Bryan Smith, Kent E. Seamons, and Michael D. Jones
Computer Science Department
Brigham Young University

IEEE 5th International Workshop on Policies for Distributed Systems and Networks (POLICY 2004)
June 7-9, 2004
IBM Thomas J Watson Research Center
Yorktown Heights, New York
Outline

♦ Trust Negotiation
♦ Policy Exchange
♦ Compliance Checker
  – Limitations of current implementation
  – Adaptations for demands of trust negotiation
♦ Conclusions
Trust Negotiation

- The process of establishing trust between strangers in open systems based on the non-identity attributes of the participants
- One approach: The incremental disclosure of credentials and access control policies
Trust Negotiation Example

Step 1: Alice requests enrollment service from Bob

Step 2: Bob discloses his policy P2

Step 3: Alice discloses her driver’s license credential

Step 4: Bob grants service
Type-1 Compliance Checker

Bob’s Enrollment Policy

Alice’s Disclosed Credentials

- Traditional Trust Management compliance checker
- Determines whether a set of credentials satisfy a policy
Type-2 Compliance Checker

Bob’s Disclosed Policy

Alice’s Local Credentials

- Determines whether a policy is satisfied and how a policy is satisfied, but only produces a single satisfying set.
- IBM Trust Establishment (TE) and REFEREE produce a set of local credentials that satisfy the received policy.
Trust Negotiation Example

Step 1: Alice requests enrollment service from Bob
Step 2: Bob discloses his policy P2
Step 3: Alice discloses her policy P3
Step 4: Bob responds with an empty message

Failure
**Type-3 Compliance Checker**

Bob’s Disclosed Policy

Alice’s Local Credentials

- Only PSPL [Bonatti-Samarati] and RT [Li et al.] return all the sets of satisfying credentials.
- PSPL has no available implementation.
- RT is currently under development.
TrustBuilder

- Prototype system for trust negotiation currently under develop at the Internet Security Research Lab at BYU
- Utilizes the IBM TE (Haifa Research Lab) system
  - Trust Policy Language (TPL)
  - Supports X.509v3 certificates
  - Type-2 compliance checker
Completeness in Trust Negotiation

♦ Goal: Obtain all satisfying sets with a type-2 compliance checker

♦ Two approaches
  – Policy Modification
  – Credential Set Modification
Policy Modification

P = a conjunction of all the credentials in the satisfying set

Policy language specific
Policy Modification

- Process continues until the compliance checker returns an empty set
- N+1 invocations of the compliance checker, where N is the number of satisfying sets
Performance Results: Policy Modification

♦ Test Scenarios
  – 50 local credentials
  – Policies with 4 or 5 satisfying sets
  – Each satisfying set consisting of 2 to 3 credentials

♦ Added overhead negligible
Credential Set Modification

♦ Modify the input credential set each time the compliance checker is invoked

♦ Two implementations
  – Brute Force
  – SSgen Algorithm
SSgen Algorithm: Definitions

- A *minimal satisfying set* is a set of credentials that satisfies the policy such that no proper subset also satisfies the policy.
- A *policy* $P$ is a disjunction of rules, where rules are conjunctions of credentials. A rule specifies a minimal satisfying set.
- A *compliance checker* is a function $f : \{C,P\} \Rightarrow S$
  - $C$ is a set of credentials
  - $P$ is a policy
  - $S$ is a subset of $C$ that minimally satisfies $P$, or the empty set
SSgen Algorithm

L is the set of local credentials
P is a policy
B is the set of known minimal satisfying sets, which can be empty
E is the set of sets known to contain no subsets that satisfy P, which can be empty
U is union of all sets in B
An n-subset is a subset that contains n members.
\( P(U) \) returns the power set of U

1. \( J = \text{complianceChecker}(L, P) \)
2. if \( J \) is not empty
3. \( B = B \cup \{J\} \)
4. Let \( S = P(U) \)
5. Let \( n = |U| \)
6. while \( (n > 0) \)
7. Let \( T \) be all the \( (n-1) \)-subsets \( \in S \)
8. For each set, \( D \in T \)
9. \( A = D \cup (L \setminus U) \)
10. if \( (A \) is not a superset or a subset of a set \( \in B \) and is not a subset of a set \( \in E) \)
11. \( J = \text{complianceChecker}(A, P) \)
12. if \( J \) is empty
13. \( E = E \cup \{A\} \)
14. else
15. if \( (J \setminus U \neq \emptyset) \)
16. goto line 3 because \( |U| \) will increase
17. \( B = B \cup \{J\} \)
18. if \( (\forall t \in T \ (t \cup (L \setminus U) \subseteq E)) \)
19. \( n = 0 \)
20. else
21. \( n = n-1 \)
22. end-while
23. end-if
SSgen Algorithm

- Finds all satisfying sets
- $O(2^{|U|})$ complexity, where $U$ is the union of all satisfying sets
Performance Results: Credential Set Modification

![Execution Time for the SSgen Algorithm](chart)
Utilizing a Type-3 Compliance Checker during Trust Negotiation

♦ Generate all the satisfying sets immediately
  – Sets can be ordered using a heuristic
  – Sets can be merged into a set containing unique satisfying credentials

♦ Generate some of the satisfying sets
  – Limit resources used to generate satisfying sets

♦ Generate satisfying sets one at a time
  – Avoids generating all satisfying sets unnecessarily
Contributions

♦ An trust negotiation system with the completeness property using existing trust management languages and compliance checkers.

♦ First example of a trust negotiation system that generates potential solutions and prioritizes them according to a specific criteria.
Questions?

For further information, go to http://isrl.cs.byu.edu