The Multipolicy Paradigm for Trusted Systems

Hilary H. Hosmer
Data Security Incorporated
58 Wilson Road
Bedford, Massachusetts 01730

Abstract

This paper describes shortcomings in the current paradigm for multilevel secure (MLS) systems, summarizes requirements for an alternate paradigm, and describes the Multipolicy Paradigm. The Multipolicy Paradigm is useful whenever there are multiple security goals such as confidentiality, privacy, availability, integrity, or weapons release control; whenever users with different values and traditions must share a common system; whenever a system is composed of separately- evaluated pieces, and whenever policies must adapt to changing circumstances. The paper suggests shifts in thinking about multilevel secure (MLS) systems, and raises important multipolicy issues: policy flexibility, policy conflict resolution, adding user security policies to commercial off-the-shelf (COTS) products, evaluating and certifying multiple policy systems, and passing sensitive data across policy boundaries.

1 Introduction

In this paper we consolidate and extend the results of our research into multipolicy systems [1,2,3,4,5] sponsored by the Air Force Electronic Systems Center. An example is presented illustrating (1) the interaction of metapolicies with security domains and (2) flexible security policies using user policy ranking.

1.1 Overview

The Multipolicy Paradigm permits a multilevel secure (MLS) system to enforce multiple, sometimes contradictory, security policies. Metapolicies, policies about policies, coordinate the enforcement of the multiple security policies. Policy domain codes on data indicate which security policies to enforce on the data, and multiple label segments supply the attributes needed for each policy.

The Multipolicy Paradigm permits natural modelling of the multipolicy real world. It permits possibly inconsistent security policies, such as confidentiality and integrity, to operate together. It may provide a vehicle for users to add their own security policies to a system without disrupting or invalidating existing evaluated policies. It may ease policy integration problems by preserving the original classification of data when data is passed across policy boundaries. Finally, if implemented in high-speed parallel processing architecture, it may improve trusted system performance.

Commercial applications include medical, financial, reservation, library, investigative and other systems that cross policy domains. Military applications include multiservice logistics, the multiservice Strategic Defense Initiative and Command, Control, and Communication (C3) systems in multinational battle theaters, like the Persian Gulf War.

1.2 Rationale

Integrating security policies on today's multilevel secure (MLS) computer systems is a difficult, sometimes impossible problem. When the security policies themselves cannot be integrated, the systems built to implement these policies cannot be integrated either. Sometimes the only way to solve impossible problems is to transcend them. For example, when Copernicus developed a new model of the planetary system with the sun at the center, his paradigm simplified planetary astronomy and initiated waves of discovery by others. Thomas Kuhn documents a number of these ground-breaking paradigm shifts in his book, The Structure of Scientific Revolutions. Hoping for similar breakthroughs, computer security founder Dr. Willis Ware has called for a new MLS paradigm which will make networking and integration of MLS systems easier.

Although the USA standards for trusted systems call for a unified system security policy, Data Secu-

rity Inc. has proposed a new security paradigm based upon multiple, perhaps contradictory, security policies. Multiple security policies may be necessary if:

1. There is more than one security goal, such as privacy, confidentiality and integrity;
2. The system serves diverse constituents with individual goals and plans, such as the United Nations (U.N.), European Community (EC), and other federations;
3. The system is composed of separately evaluated pieces, such as an MLS Database Management System (Trusted DBMS) and MLS Operating Systems (Trusted OS).
4. Policies must adapt to changing circumstances, like moving from peace to war.

2 The Current Paradigm

The current US paradigm is based upon three standards documents described below: the TCSEC, the TNI, and the TDI. The current paradigm may evolve significantly because of the ITSEC, the 'harmonized' European criteria, and the Federal Criteria, a standards document focused on commercial system security, now under development at NIST with NCSC support.

The TCSEC. The Department of Defense Trusted Computer System Evaluation Criteria (TCSEC), embodies the United States' security paradigm. The TCSEC prescribes a unified "system security policy" made up of subpolicies such as Mandatory Access Control (MAC) and Discretionary Access Control (DAC) which all cohere together to form a single system security policy. The unified policy drives the choice of security mechanisms and is the foundation of most assurance efforts.

The single-policy paradigm works well with standalone systems but causes problems when systems must be networked or combined and security policy integration is required. For example, when MLS products with slight variations in policies (such as Operating System (OS), Database Management System (DBMS), and user applications) must work together, there are usually policy integration difficulties as well as other interoperability issues [3]. The policy integration problems are even more evident when systems enforcing different policies, such as U.S.A. Department of Defense (DOD), North Atlantic Treaty Organization (NATO), and France, must interact and share classified data.

The TNI. The Trusted Network Interpretation (TNI) of the TCSEC enlarges the single-policy paradigm so that multiple policies may coexist on computer networks. It permits each node on a network to have its own nodal security policy, but stipulates that the network as a whole must have an overall global network security policy which is the basis for evaluating the security of the network.

The TDI. The Trusted Database Interpretation (TDI) addresses the problem of composing systems out of separately developed and evaluated trusted software products. The trusted computing base (TCB) of each separate component is called a TCB subset. Each TCB subset can enforce a different security policy, such as MAC or DAC. The TDI assumes, however, that these subpolicies cohere into a single consistent overall security policy.

The ITSEC. The draft International Technology Security Evaluation Criteria (ITSEC) permits a user to specify a security policy, select a system meeting site needs, then request a certified evaluation center to do an evaluation to provide the necessary assurance that the selected system is able, in fact, to carry out the user's security policy. There is no restriction on what functionality could be in the user's policy. The policy could include integrity, availability, non-repudiation, and confidentiality, for example. The ITSEC follows the TCSEC lead in requiring users to integrate multiple separate policies into a single coherent system security policy.

2.1 Problems with the Current Paradigm

The paradigm of a unified security policy has some major shortcomings which are becoming apparent as multilevel secure systems are fielded.

1. It's inflexible. If a user wants to modify built-in aspects of the system security policy, the whole system must be reevaluated.
2. Exchanging sensitive data with systems with other security policies is difficult or impossible in real-time. Guards are needed at all interfaces, and mapping rarely can translate security levels from one policy to the other without upgrading.
3. It's unrealistic. The real world has multiple coexistent security policies. A computer security officer creating an automated security policy must often integrate diverse and contradictory security policies together into a single coherent policy to meet TCSEC criteria. Canada's experience trying to integrate the national privacy policy with

the national disclosure policy into a single policy lattice illustrates the real difficulties users face.

4. Performance is poor. Adding security to existing systems seriously slows down throughput.

The current paradigm must be enlarged or shifted to meet the needs of a more interrelated and integrated world. With a few significant enhancements, the single-policy paradigm can be extended into a more flexible, more interoperative, better-performing multipolicy paradigm.

3 Requirements for a New Paradigm

What must a larger and more inclusive paradigm do? It should:

- Handle COTS system construction. Facilitate the integration and tailoring of commercial off- the-shelf products to meet the end-user's system security policies.
- Separate the policy from the enforcement mechanism. The system security policy should not be such an integral part of the system that it is impossible to change policies without reevaluation.
- Ease sharing data with other policy systems. In multinational conflicts like the Persian Gulf, US DOD users need to share classified data with allied computers that implement different service, national or international security policies.
- Enforce the originator's security policy. Current strategies for sharing data across security policy boundaries (Guards, Man-in-the-loop) frequently must upgrade or downgrade data, thus losing the original classification. Even if the multinational situation is one of cooperation rather than conflict (for example, divisions of a multinational corporation, or international electronic funds transfer), it is desirable to guarantee enforcement of the originator's security policy while sharing data across computer systems.
- Permit contradictory policies to operate in parallel. For example, different states have passed different laws about releasing AIDS data. If an AIDS patient from Connecticut is in a New York hospital, which state's disclosure laws should apply to the release of data? In the European Community health system, the varying disclosure laws of 12 different countries must be implemented and maintained.
- Improve the performance of trusted systems. Adding security to a system usually degrades its performance significantly, largely because of auditing and access control checks.
- Other. The list above is not exhaustive. As more multilevel systems are implemented, we will become aware of more difficulties and requirements. Solving these problems is essential to widespread user acceptance of MLS systems.

4 Related Work

Many researchers have addressed aspects of these problems. To give just a few examples, Biba and Clark and Wilson established the importance of integrity policies. DEC built a multipolicy operating system SEVMS that enforces both confidentiality and (Biba) integrity, showing the commercial feasibility of multiple policy systems. The European Computer Manufacturers Association (ECMA) developed a conceptual framework for security across multiple domains with multiple authorities, raising hopes for international standards.

Multiple policies frequently conflict. Dobson and McDermid discovered that Integrated Programming Environments (CASE tools) require three distinct security policies , and "it is critical to articulate and resolve the policy conflicts." Dobson has been studying the source of many conflicts by documenting organizational security policies and modelling enterprises. Trusted Information Systems (TIS) documented that the Aegis Combat System requires three sometimes conflicting policies (information disclosure, information modification, weapon release), while the Nuclear Command, Control, and Communications system requires four policies (weapon release, denial of service, information disclosure, and information modification). Secure Computing Technology Corp. (SCTC) and Georgia Tech Research Corporation (GTRC) in their Assured Service policy work identified ways that various availability mechanisms both complement and conflict with secrecy policies. Rae Burns raised the inherent secrecy/integrity conflict, and Oracle Corp. addressed methods for resolving it. Tradeoffs between competing policies are often required, and often only the ultimate users can determine which policy to emphasize. From all this work, it is clear that in a multipolicy automated information system environment, it is critical to specify how policy conflicts should be resolved.

Multiple policies may be more complex than single policies. Some researchers are studying the fundamental properties of policies, which may help to reduce this complexity. Feiler and Dowson explored the relationships between policies and processes, discovering that policies may conflict and that policies about policies may be necessary. Moffet and Sloman explored management policies and the need for explicit control authority in the commercial arena. They also explored how to represent and manipulate policies and came to view policies as objects which can be created, destroyed, queried; and which can interact with each other. The Policy Workbench project at George Mason University(GMU) studied intentions implicit in policies, incompleteness in assumptions underlying security policy models, and ways to represent security policies, including activity role charts, Petri nets, data flow diagrams, and structural diagrams.

Several researchers aim for policy flexibility. Grenier, Hunk, and Funkenhauser differentiated policies from mechanisms. The Planning Research Corporation (PRC) proposed rule-based policies as a way to escape the inflexibility of built-in policies and demonstrated that assorted rule-bases can be plugged into the same system. MITRE's General Framework for Access Control (GFAC) group asserted that all policies can be expressed as rules specified in terms of attributes and other information controlled by authorities.

Our earlier work introduced several concepts which are incorporated into this paper. [1] proposed a Multipolicy Machine which enforces multiple, possibly contradictory security policies using Metapolicies, a term introduced in [3] and expanded in [2]. [4] proposed shared labels to save space, and parallel processing of policies and policies on ROM chips to improve performance and standardization.

5 The Multipolicy Paradigm

5.1 Components

Most security models built after Bell and LaPadula's classic model include:

1. Subjects
2. Objects
3. Security Policy
4. Sensitivity attributes for subjects and objects
5. Policy Enforcer to mediate subjects' operations on objects in accordance with the policy.

Several additional components are required to handle multiple policies:

1. Multiple security policies;
2. Multiple security policy enforcers;
3. Multiple policy coordinators (metapolicies);
4. Assignments to specify which policies apply to which subjects and objects.

Two optional components are needed to provide flexibility and performance:

1. A means to control policy changes and updates;
2. A design to avoid policy-enforcement processing bottlenecks.

Each component is described below. We describe abstract concepts, then suggest concrete ways to implement those concepts.

A policy is a set of constraints established by an accepted authority to facilitate group activity. A policy may be explicit or implicit, broad or narrow in scope, mandated or optional.

Security policies are those policies whose goals are protecting the confidentiality, integrity, and/or availability of people, resources, and information.

Automated security policies [11] protect information within computer systems, and require security policies that are much more explicit and formally specifiable than policies intended for people. Automated security policies typically include: a) definitions of subjects and objects; b) definitions of allowable operations; c) policy rules, and d) data for implementing the policy, such as a lattice for ordering sensitivity levels, integrity levels, compartments, etc. Automated security policies must be tamperproof and are, by definition, part of the trusted computing base.

In the Multipolicy Paradigm, a computer system can enforce more than one policy. The Multipolicy Paradigm permits multiple:

1. Types of security policies (Eg. integrity, MAC, DAC, Chinese Wall);
2. Variations of security policies (Eg. integrity by Biba and by Clark- Wilson);
3. Combinations of policies (Eg. hierarchical, independent, coordinated);

4. Sources of policy (Eg. user, administrator, government, standards body);

5. Means of changing policies (Eg. locally, remotely, at sysgen).

Unlike the current TCSEC paradigm, the Multipolicy Paradigm does not require that a unified system policy be developed, or even that the policies be consistent. Canada, for example, can implement separate privacy and confidentiality policies on one system[13], and the EC may keep several separate health and financial information privacy policies.

Policies in current systems are usually implemented as instructions in code using user-supplied data from tables for access control, with instructions in the kernel TCB for system security policies and instructions in applications programs for user policies. In the Multipolicy Paradigm, complex data structures will be needed to implement each policy and its associated metapolicies. This implementation method can provide both flexibility and assurance.

5.2 Multiple Enforcers

Security Policy Enforcers Security policy enforcers implement the rules of a policy on the subjects and objects within the policy domain. Each enforcer is trusted to protect and enforce the policies in its domain correctly and must be tamperproof. Enforcers may be implemented in several ways. However, it is critical that the policy NOT be built into the enforcer, as it now is in most reference monitor implementations. One enforcer may enforce multiple independent policies, or multiple policy enforcers may enforce multiple different policies, or multiple versions of the same policy, or multiple subsets of the same policy.

Metapolicies Metapolicies are policies about policies. They provide a framework for clarifying policies, explicitly stating the assumptions about policies and the organization's control process for policies. They also coordinate the interaction between policies, explicitly specifying order, priority, and conflict-resolution strategies. Metapolicies clarify underlying policy assumptions and relationships, facilitate expression of the variety, richness, and multiplicity of security policies, and permit the controlled interaction of policies and subpolicies, making complex policy systems possible [2]. Metapolicies specify who can set policy, who can change policy, and the procedures for changing policies. They also include rules about developing, verifying, and protecting security policies and rules about the interaction of multiple security policies, especially where they conflict. The Multipolicy Paradigm permits multiple distinct security policy domains, administered by different organizational entities each with complete policy autonomy in its domain, to be modeled in a computer system. Metapolicies control the interactions of the multiple policies.

5.3 Multiple Domains

A policy domain is a logical construct defining the area of responsibility of an authority. The U.S. federal government, for example, takes responsibility for regulating interstate commerce (the federal domain), while the states take responsibility for regulating intrastate commerce (the 50 state domains). NCSC, OSI, ISO, ECMA, DOD, and NATO are a few of the well-known security domain authorities.

Each security domain may be autonomous, with its own authority, subjects, objects, policies, and policy enforcement mechanisms. Others may be part of a hierarchical structure, like Air Force Base (AFB), Air Force System Command(AFSC), Air Force (AF), and Dept. of Defense(DOD). In hierarchical structures, the authority and policies of the top domains must be incorporated by the subordinate domains. Under the unified-policy model, the base, system command, AF and DOD policies would be integrated and implemented as a single automated security policy. However, under the Multipolicy Paradigm, each of the individual policies in the hierarchy - the DOD policy, the AF policy, the AFSC policy and the AFB policy - would be separate policies, and a policy domain code would be required for each. This gives the AFB security administrator the flexibility to change local base policy while leaving national DOD and AF policies untouched.

Domains may overlap each other, so that subjects or objects may belong to more than one domain and fall under more than one policy. Patients who fall under the confidentiality policies of multiple states, and military information which comes under both national and international confidentiality policies, are members of overlapped domains.

Policies in different domains may conflict. However, there must be means to resolve the conflicts as they occur. For example, if a national and an international policy are in conflict, which takes precedence? A later section addresses conflict resolution techniques. Policies within the same domain must not conflict, because logical inconsistencies may create exploitable holes. Research is needed to see if policy conflicts are possible between subdomains.

6 Multipolicy Problem

A multicultural scenario is used to illustrate different policy domains, policies in conflict, and conflict resolution by metapolicies.

> Two passengers aboard ship wish to be married. Should the Captain marry them?
>
> Sally Jones is female, American, single, Protestant, 28 years old, and is heading to Kuwait to live. Ibrahim Mohammed is male, Kuwaiti, married, Muslim, 35 years old, and resides in Kuwait. Ibrahim has one wife, although up to four are allowed under Muslim law.
>
> At sea, under Maritime Law, the Captain has the discretion to do anything. However, the couple does not plan to stay at sea forever, so the Captain is concerned about national and religious law as well. If he marries the couple, will they be able to live in peace once they reach land?

There are three major policy areas in this example: Religious traditions, National laws, and Maritime law. The policy domains are: Muslim religious marriage policy, Protestant religious marriage policy, American law, Kuwaiti law, and high seas maritime law. Some of the policy- makers were the Prophet Mohammed, Henry the Eighth, the U.S. Congress, and the King of Kuwait. Some of the policy authorities are the leading ayatollahs, the church committees on marriage and family, and the U.S. Supreme Court. Policy implementors include imams, pastors, justices of the peace and the ship's Captain.

Sally comes under three domains: Protestant marriage policy, American marital law, and, because she is now aboard ship, high seas maritime law. Ibrahim also comes under three domains, but they are different domains: Muslim marriage policy, Kuwaiti marital law, and high seas maritime law.

To enforce a policy on an object may require knowing some attributes of the object. For example, to enforce the Muslim religious policy on Ibrahim, it must be known whether Ibrahim is Muslim, male or female, married or single, and if married, whether there are three or fewer spouses.

Under Kuwaiti law and the Muslim religion, a male may have as many as four wives, so Ibrahim can marry Sally. However, Sally's religion and American law permit only one spouse, so Sally cannot marry Ibrahim. This is a policy conflict, in need of resolution.

6.1 Resolution by Metapolicies

Several conflict resolution metapolicies may be invoked to settle this problem. For example:

If the couple are of the same religion, then that religious policy prevails.

- If the couple come from the same country, then the national law of that country prevails.
- If the couple are going to live in the same country, then the law of the land where they will reside prevails.
- If nothing else works, the captain may decide.

The first two metapolicies don't help, since Sally and Ibrahim are from different religions and different countries, but the third resolves the issue. Sally and Ibrahim both will reside in Kuwait, so Kuwaiti law prevails.

7 Multipolicy Issues

7.1 Conflict Resolution

Strategies for resolving conflicts between policies include:

Resolve the conflicts manually and automate the integrated results. This is the strategy taken by most vendors and most user organizations. The information security officer manually integrates multiple, possibly contradictory policies into a coherent system security policy. This is a difficult process, since each policy has its own source or owner, its own enforcement authorities, and its own evolutionary time frame. Developing consensus takes a long time, especially if policies reflect deeply held values.

This would be analogous to trying to get Christians and Muslims to adopt the same marital policy. Compromises, such as allowing a man or woman to have two spouses, may offend both groups.

Resolve by dominance. If the policies are hierarchically structured, then the policy higher in the hierarchy predominates. If the policies are ranked by their importance, the most important predominates. Or, if the policies reflect the ranking of the authorities who created them, then

the policies of the dominant authority predominate. This strategy is appropriate in the military and other hierarchically-structured organizations. The problem is that the resolution may unravel as soon as it leaves the hierarchy.

The sea captain's authority dominates only while at sea. The marriage must be blessed by authorities on land as well.

Translate policies into a common form. Dr. Bell advocates this strategy using policy conversion logic on a Universal Lattice Machine. He showed that multinational sharing, Clark and Wilson, dynamic separation of duty and ORCON can all be implemented with the Universal Lattice Machine. Apparent contradictions or differences may disappear when the policy is a common form.

Run in separate policy domains. John Rushby's Separation Machine, as implemented by Amdahl's Multiple Domain Facility, allows seven different policies on one machine but no communication between domains. Parallel processing of policies is possible but resolving conflicts between policies must be done outside the Separation Machine.

Use additional enforcement mechanisms to implement custom user policies in addition to DAC, MAC , etc. Type enforcement like that implemented in SCTC's Logical Coprocessor (LOCK) provides considerable user flexibility.

Use rule-based access control. John Page[27], Marshall Abrams[28], Leonard LaPadula , and others' showed how rule bases (rules with one-to-one correspondence with the operations of the system) handle many kinds of access control policies. LaPadula proposed a voting technique to resolve rule conflicts which we adopt in Figure 2.

Adjudication. In case of conflict, develop a solution which reflects the tradeoffs and weights of the users on the system. If there are multiple applications which weigh things differently, accommodate the various weights. The use of metapolicies, or 'policies about policies', to sort out precedence and to identify and resolve policy conflicts is illustrated in Figure 2.

Outside mediation. When two security policies contradict each other, the decision about what to do may be best left to a human who understands the content and the context, as in downgrade decisions.

A combination of these techniques can be powerful. Figure 2 shows that the conflict resolution process can be simple and elegant, no matter how many different policies are included. If parallel processors are used to implement multiple policies [1], the decision-making time could be kept close to that of a single-policy machine.

7.2 Policy Assignments

Implementation of multiple policies and metapolicies on information systems requires a way to tell what policies should be enforced on what data. Currently, mandatory access control policies (MAC) use sensitivity labels to describe security attributes, and implicitly assume that the MAC policy is the one to be enforced.

Data Security proposed in [4] to extend the sensitivity label to accommodate multiple policies. The European Computer Manufacturers Association (ECMA) [17] has proposed security domain codes on security labels which indicate under which label convention the label is formatted, for example, the International Standards Organization (ISO). We propose security policy domain codes as a mechanism to indicate which policy domains apply to this subject, object, or policy. Whenever policy decisions are made, these policy domain codes would be checked first so that the proper policy enforcers can be invoked. Figure 1 illustrates domain codes incorporated into security labels.

7.3 Sensitivity Label Formats

Note that labels with multiple policy attributes and multiple security domain codes may get very long. A paper published last year, "Shared Sensitivity Labels" [4] describes an indirect addressing technique which permits subjects and objects with the same sensitivity levels to share a single version of the label.

7.4 Conflict Resolution Process

1. The 'Subject' wants to operate on the 'Object', but the request must be mediated by the 'Policy Enforcer'.

2. The Policy Enforcer passes the request to the 'Policy Decider' along with the subject and object policy domain codes. The Decider consists of multiple 'Policies' operating in parallel, one for each policy implemented by the system.

3. Based upon policy domain codes, the request is routed to the proper Policy.

Single Policy Label Format:
OBJECT // POLICY DOMAIN CODE / SECURITY ATTRIBUTES /
Single Policy Example:
The string of bits representing patient Jones' data release permissions will be interpreted in accordance with the New York privacy policy.
Patient Record for John Jones // Privacy-NY / 100101/

Figure 1: Single Policy Domain Code Example

Multiple Policy Label Format:
OBJECT// POLICY CODE / ATTRIBUTES / POLICY CODE / ATTRIBUTES / etc
Multiple Policy Example:
Patient Smith, who lives in Connecticut, is hospitalized in New York and then sent for consultation to a teaching hospital in Massachusetts. The privacy policies for all three states apply to him and his hospital record has three sets of privacy attributes.
Patient Sam Smith // Priv-MA / 01011 / Priv-CONN / 01010 / Priv-NY / 11010

Figure 2: Multiple Policy Domain Codes in Trusted Labels

4. Using rules and decision data to evaluate the request, each Policy Decision-Maker sends its Policy Precedence Ranking and a Vote (e.g. Yes', 'No', 'Don't Care', 'Undecided' or a fuzzy logic number on a continuum) to the Metapolicy.

5. The votes of all the individual policies (Vote 1 and Vote 2 in this example) are combined by the Metapolicy and weighed according to its rules as well as the precedence ranking of each policy (Rank 1 and Rank 2 in this example).

6. The resulting 'Yes' or 'No' vote is sent back to the Policy Enforcer which then permits or denies the requested operation.

7.5 Flexible User Security Policies

One of the frustrations experienced by users is their limited ability to modify the security policies which are built into COTS products. Most current systems allow changes to the policy data (eg. the contents of the lattice), but not to the policy rules. The problems are:

1. the rules are built in;

2. assurance depends upon a stable policy;

3. changes may introduce security flaws.

User ranking. A simple scheme is to allow user authorities to rank the multiple policies in their system. The policies do not have to be changed, but user authorities can prioritize policies to emphasize those that are most important to their organization's goals. For example, availability is critical to reservations systems, patient monitoring, manufacturing processes. As several people have noted, the military will usually emphasize confidentiality over integrity, while commercial institutions like banks, insurance companies, retailers, and manufactories are likely to emphasize integrity over anything else. See Figure 4.

Tailoring Commercial Off-The-Shelf (COTS) systems. Users complain frequently about how difficult it is to tailor any current trusted system to their own needs. A vendor designing a trusted multiple policy system tries to meet the needs of the largest market groups, but cannot anticipate the total needs of every eventual user. For the vendor, it is much easier to include many policies in the system and market one multilevel secure product to diverse customer communities which prioritize the multiple policies to best meet their own goals. See Figure 5.

Adaptive policies. User priorities may suddenly change. For example, when the threat level changes, the military quickly changes security policies. Cars at a base must be parked further away from buildings, and data may be classified at the next higher level. In many military systems, confidentiality is the goal during peacetime, but availability is the goal as soon as war breaks out.

Current systems do not handle change well. Built-in policies are the keystone of building and evaluating today's system. A change in policy means rebuilding

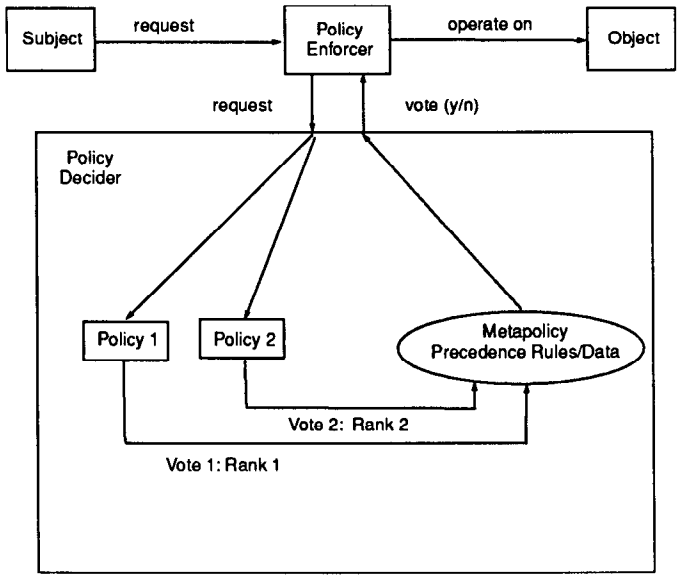

Figure 3: Multipolicy conflict resolution

Military System		**Commercial System**	
Policy	*Rank*	*Policy*	*Rank*
Confidentiality	80	Confidentiality	25
Weapon Release	95	Weapon Release	0
Integrity	30	Integrity	85

Figure 4: Policy ranking in different environments

and reevaluating the trusted system. However, if the policies are already in the system, it is a simple matter to change the rankings so that there is a switch in policy. See Figure 6.

In summary, in a multiple policy system user authorities will rank policies for several reasons.

1. Ranking policies permits users to tailor COTS products to their needs;
2. Ranking permits trusted system vendors to market the same trusted product to diverse communities;
3. Ranking permits quick switching between policies to adapt to changes.

There are several other ways to provide policy flexibility. For example:

1. Domain administrators are responsible for the policies in their domain. Domain administrators can "securely download new policy modules, send new firmware chips for installation by each System Security Officer (SSO), or simply give orders to the SSOs to make changes. Metapolicies will restrict policy changing to authorized personnel.

2. COTS vendors can offer customers a set of evaluated policy options, clusters of commonly-desired combinations, to choose from when the product is ordered. Trusted software can be used to tailor policies further. Modifiable aspects, such as label size, number of compartments, and which policies are selected for enforcement, must be carefully limited to maintain the integrity of the evaluated system.

System M Policies	User 1 Rank	User 2 Rank
Confidentiality	75	25
Chinese Wall	0	60
Integrity (Biba)	0	0
Integrity (Clark and Wilson)	40	70
Weapon Release	0	0

Figure 5: Selecting multiple policies by ranking

Policy	Rank Jan 1	Rank Jan 8	Rank Jan 9
Threat Level1	50	0	0
Threat Level 2	0	80	0
Threat Level 3	0	0	90
Threat Level 4	0	0	0

Figure 6: Switching policies by changing rankings

3. The vendor could tailor the system before shipping, or the System Security Officer (SSO) could tailor it at system generation. The vendor should ship any trusted system with conservative defaults selected to err on the side of caution.

4. Vendor-provided options, such as audit policy options and default options, can be set by the SSO at any time.

Adding user policies. Although inconceivable with today's built-in policies, user authorities should be able to add or delete policies from their systems at any time. Standardized policies and labels may be distributed on ROM firmware [4] or protected software modules and would include metapolicies which describe the policies and their interrelationships. Metapolicies which coordinate the policies must be customized to the user's needs when each policy is installed.

Currently, user policies are coded many times into applications programs. It is desirable for integrity and control to get the policy out of the application and into the system where the same policy can be invoked by many programs. Ideally, a System Security Officer (SSO) should be able to enter entire user policies via trusted software into an isolated area of the TCB where their interactions with applications programs and other policies are carefully mediated by the appropriate metapolicies.

The capacity to absorb multiple user policies (representing multiple nations, multiple divisions, or several kinds of integrity policies) without reevaluating the whole system is an integral part of the Multipolicy Model. However, evolution of policies raises the issues of reevaluation and recertification.

7.6 Evaluation and Certification Issues

How does one evaluate and certify a system with multiple flexible policies? If policies change, whether at sysgen or on-the-fly, when must the system be reevaluated or recertified? There are many questions and problems.

Today, evaluators determine whether or not a system correctly implements a policy with what degree of assurance. The Multipolicy Paradigm requires a shift

in thinking so that evaluators determine that the system has mechanisms which will correctly implement whatever policy it is given. The evaluators will examine the mechanisms for interpreting and enforcing a variety of policies rather than just one.

Evaluators cannot determine in advance that every possible policy which might come along will work correctly in the system. Certifiers will study the installation of specific policies on a specific system. Certifiers will also check the metapolicies which are set by the user, such as those that prioritize policies in conflict, Certifiers will need to check the interaction of multiple security policies and to handle the problem of too many possible combinations to test them all.

If evaluators evaluate systems separately from polices, who evaluates the policies? Since there are likely to be so many different policies, commercial evaluation centers, like the ones doing evaluations in Europe, would be appropriate for policy evaluation. If policies have been developed in different places by different authorities and evaluated in different places with different levels of assurance, common standards need to be developed. For example, for any desired certification level, each policy must have the minimum level of assurance for that certification level. Metapolicies which describe and control the policies must be evaluated as well.

7.7 Crossing Policy Boundaries

Crossing policy boundaries is one of the most difficult problems in trusted computing. When classified data leaves one policy system, such as the US Military, to go to another, such as NATO, a trusted person or process must sanitize and relabel the data and approve the transfer. In the civilian world, privacy laws require that the patient or the patient's guardian give written approval for the transfer of medical data to another hospital. In both military and civilian life the data owner wants assurance that the data will be treated in the new policy realm in sufficient accordance with the owner's policies.

In a multipolicy system, it is possible for an object to go from one multipolicy machine to another without leaving its original policy domain. There are several important assumptions:

1. The Multipolicy Machines follow standards for labelling objects which preserve the integrity of labels and policy domain codes.

2. The Multipolicy Machines follow standard policies about handling objects from different policy systems. For example, if the label is checked and it doesn't match any policy in the system, the object is inaccessible to any users on the system. However, the inaccessible object may be passed on intact to another system which follows the standards for multipolicy systems.

The sending and receiving systems may implement a homogeneous, an overlapping, or a heterogeneous set of policies. As long as the receiving system is trusted to implement the policies indicated in the label associated with the object, there is no boundary-crossing problem.

If the receiving system does not enforce the policy or policies marked on the object, it must either pass the object on to another machine which enforces the appropriate security policy, hold on to the object without permitting any access, or, as is done now, request human intervention. Which choice is made could depend upon instructions which accompany the object, or on the metapolicy for the receiving computer.

8 Implementation Options

Throughout this paper we have suggested several reasonable approaches to implement a Multipolicy Paradigm:

1. Multiple sets of rule-based policies, as seen in Figure 3, [27] and [35];

2. Multiple co-processors, like SCTC's LOCK and Sidearm [33];

3. Distributed processors: each node has a local policy and a master node has them all; [1]

4. Parallel processors or policies in ROM chips to improve performance [4];

5. Multidomain machines, like Amdahl's Multidomain Facility [32];

6. Hybrids of the above

Other approaches are possible as well, but we do not wish to focus here on implementation options. More implementation option information appears on our "The Multipolicy Model; A Working Paper" [5].

9 Applications

The Multipolicy Paradigm is useful whenever multiple security policies are involved, especially when

normal security goals are extended beyond DOD confidentiality to include privacy, availability, integrity, weapons release control or other policies and wherever users with different values and traditions must share a common system.

Military multipolicy applications include: multinational battle management, multinational command and control centers, logistics involving multiple services, and multinational communications systems. The Strategic Defense Initiative is a classic case of multiservice policy interaction, as was the Persian Gulf War. An application common to ordinary military systems would be to define peacetime, threat alert, and wartime security policies and shift from one to the other, rather than 'loosening' the peacetime policy[20] when war starts.

There is no single standard security policy, like that of the DOD, in the commercial world, so a trusted system, to be marketable, must be able to adapt to multiple policies. Although the TCSEC unified policy paradigm can adapt to a wide range of needs [Bell, 31], the Multipolicy Paradigm will facilitate expression of users' diverse, unanticipated, and contradictory security policies.

Commercial applications for multipolicy machines are numerous. Multinational banks, multinational corporations, international non-profit activities such as the Red Cross and CARE, merged corporations with multiple corporate cultures, colleges and companies which cross state borders, international telecommunications systems, are all candidates for multipolicy systems.

In non-military government sectors there is even more potential for the multipolicy paradigm. Almost every system developed by the European Community needs multiple policies to express the different values and varying traditions of the nations involved. For example, a multipolicy international health system that permits different nations to control security policies for their own citizens is more practical than requiring twelve nations to come up with a unified confidentiality and privacy policy.

10 Conclusion

This paper identified shortcomings in the TCSEC/TNI/TDI paradigm for multilevel secure systems and summarized some of the requirements for an alternate paradigm. It briefly described other researchers' work in the area, then wove many contributions into a Multipolicy Paradigm.

The Multipolicy Paradigm supports multiple, perhaps contradictory security policies and has many applications and uses. Multiple contradictory security policies may be necessary if:

1. There is more than one security goal, such as privacy, confidentiality and integrity;
2. The system serves diverse constituents with individual goals and plans, such as the EC;
3. The system is composed of separately evaluated pieces, such as MLS DBMS and OS.
4. The system's policies must adapt to changing circumstances, such as peace and war.

Multiple policy systems will be more flexible, but much more complicated in many ways than single policy systems. The paper addressed strategies for solving many of the key multipolicy issues and showed that

- Policy conflicts can be resolved;
- Changes in ways of thinking are needed to evaluate and certify flexible multipolicy systems.
- There are many strategies for getting policy flexibility while preserving assurance.
- Users can add user security policies to commercial off-the-shelf products.
- Multipolicy systems may ease the old problem of how to pass sensitive data across policy boundaries.
- The Multipolicy Paradigm can be successfully implemented in many ways.

The Multipolicy Paradigm will provide greater flexibility for users who need to add their own security policy specifics to the security policy of an existing system. It will make it easier to transfer data to systems in other security policy domains. It will let users model complex real world security policies more easily and permit contradictory policies to operate in parallel. Parallel processing may permit an improvement in trusted system performance, as well.

The Multipolicy Paradigm is now just a concept with potential. Much more work needs to be done to make it a reality.

Acknowledgements

This work was produced under a Small Business Innovative Research grant, contract number F19628-91-C-0157, under the sponsorship of the Electronic Systems Division of the Air Force Systems Command, Hanscom Air Force Base, Bedford, MA. Several colleagues contributed helpful suggestions, critiqued the multipolicy ideas as they evolved, or provided support for spreading the word about multipolicies: Marshall Abrams, Victoria Ashby, David Bell, Dennis Branstad, Rae Burns, Rowena Chester, Dorothy Denning, Jon Graff, Tom Haigh, Grace Hammonds, Jody Heaney, Eric Leighninger, Bret Michael, and Bhavani Thuraisingham.

References

[1] Hosmer, H.H. "The Multipolicy Machine: A New Paradigm For Multilevel Secure Systems," *Proceedings of Standard Security Label for GOSIP, an Invitational Workshop*, April 1991, NISTIR 4614, June 1991.

[2] Hosmer, H.H., "Metapolicies I," ACM SIGSAC Data Management Workshop, San Antonio, TX, December 1991, ACM SIGSAC Review 1992.

[3] Hosmer, H., "Integrating Security Policies," *Proceedings of the Third RADC MLS DBMS Workshop*, Castile, NY. June 1990, MITRE Technical Paper MTP 385.

[4] Hosmer, H.H., "Shared Sensitivity Labels," *Database Security, Status and Prospects*, North-Holland, 1991.

[5] Hosmer, H.H. "The Multipolicy Model, A Working Paper," *Proceedings of the Fourth RADC Workshop on Multilevel Secure Database Systems*, Little Compton, Rhode Island, June 1991,

[6] Kuhn, T., *The Structure of Scientific Revolutions*, 2nd Edition, University of Chicago Press, Chicago, 1970.

[7] Ware, W., Comments from Computer Security Panel, AFCEA Conference, Washington, D.C., Feb. 5-7,1991.

[8] Department of Defense, *Trusted Computer System Evaluation Criteria*, DOD 5200.28-STD, December 1985.

[9] National Computer Security Center, *Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria*, 31 July 1987.

[10] National Computer Security Center, *Trusted Database Interpretation of the Trusted Computer System Evaluation Criteria*, April 1991

[11] Information Technology Security Evaluation Criteria, draft of 2 May 1991.

[12] Sterne, D., "On the Buzzword Security Policy," *Proceedings of the 1991 IEEE Computer Security Symposium on Research in Security and Privacy*, May 1991, Oakland, CA.

[13] Crawford, D.S. "Modelling Security Policy and Labelling Unclassified but Sensitive Information - A Canadian Perspective," *Proceedings of Standard Security Label for GOSIP An Invitational Workshop*, NISTIR 4614, June 1991.

[14] Biba, K.J., Integrity Considerations for Secure Computer Systems, MTR-3153, Rev. 1, Electronic Systems Division, Air Force Systems Command, United States Air Force, Hanscom Air Force Base, Bedford, MA, April 1977 (ESD-TR-76-372).

[15] Clark, D.D., and Wilson, D.R., "A Comparison of Commercial and Military Computer Security Policies," *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, Oakland, CA, April 1987.

[16] Comments made by William Wilson in San Antonio, Dec. 1991 that the SEVMS integrity portion is not well-utilized because of the absence of a standard integrity user clearance structure like the widely-implemented DOD user clearances for confidentiality.

[17] European Computer Manufacturers Association, Security in Open Systems, A Security Framework, ECMA TR/46, July 1988.

[18] Dobson, J. and McDermid, J., "A Framework for Expressing Models of Security Policy," *Proceedings of the 1989 IEEE Computer Society Symposium on Security and Privacy*, May 1-3, 1989, Oakland, CA.

[19] Sterne, D., Branstad, M., Hubbard, B., Meyer, B., and Wolcott, D., "An Analysis of Application-Specific Security Policies," *Proceedings of the 14th National Computer Security Conference*, October 1-4, 1991, Washington, D.C.

[20] Haigh, T., O'Brien, F., Endrizzi, W., and Yalamanchi, "Assured Service Concepts and Models," draft Final Technical Report, Contract Number F30602-90-C-0025, October 1991, CDRL A007, vol. 1 and 2.

[21] Burns, R.K., "Referential Secrecy," *Proceedings of the IEEE Computer Security Symposium*, Oakland, CA, 1990.

[22] Maimone, B. and Allen, R., "Methods for Resolving the Security vs. Integrity Conflict," *Proceedings of the Fourth RADC Database Security Workshop*, Little Compton, R.I. April 1991.

[23] Feiler, P., "Experiences with Software Process Models, Session Summary: Policies," *Proceedings 5th International Software Process Workshop*, Kennebunkport, ME, October 10-13, 1989.

[24] Moffett, J. and Sloman S., "The Representation of Policies as System Objects," Domino Report:B1/IC/6.1, August 20 1991; "The Source of Authority for Commercial Access Control," IEEE Computer, February 1988.

[25] Sibley, E.H., Michael, J.B., and Wexelblat, R.L., "An Approach to Formalizing Policy Management," P. Bourgine and B. Walliser, eds., *Economics and Cognitive Science*. Pergamon Press, Oxford, England, 1992.

[26] Grenier, G.-L., Holt, R., and Funkenhauser, M., "Policy VS Mechanism in the Secure Tunis Operating System," *Proceedings of the 1989 IEEE Computer Society Symposium on Security and Privacy*, May 1-3, 1989, Oakland, California.

[27] Page, J., Heaney, J., Adkins, M., and Dolsen, G., "Evaluation of Security Model Rule Bases," *Proceedings of the 12th National Computer Security Conference*, Baltimore, Maryland, 1989.

[28] Abrams, M., LaPadula, L., Eggers, K., and Olson, I., "A Generalized Framework for Access Control: An Informal Description," *Proceedings of the 13th National Computer Security Conference*, Washington, D.C., October 1990.

[29] Bell, D.E., and LaPadula, L.J., "Secure Computer System: Unified Exposition and Multics Interpretation," MTR-2997, The MITRE Corporation, July 1975.

[30] Brewer, D.F.C. and Nash, M.J., "The Chinese Wall Security Policy," *Proceedings of the 1989 IEEE Computer Security Symposium on Security and Privacy*, Oakland, CA, 1989.

[31] Bell, D.E., "Putting Policy Commonalities to Work," *Proceedings of the 14th National Computer Security Conference*, October 1-4, 1991.

[32] Amdahl Corporation, Multiple Domain Feature, General Information Manual, Amdahl MM001501001[1:10]6-89.

[33] Honeywell Inc. B-Level Design Specification for the LOCK Operating System, CDRL A009, Contract MDA 904-87-C-6011, June 1987.

[34] LaPadula, L.J., "A Rule-Base Approach to Formal Modeling of a Trusted Computer System," M91-021, August 1991.

[35] The diagram and description combine our visualization of metapolicies resolving policy conflicts [1] with Marshall Abrams' diagram of a proposed ISO conflict resolution process for access control policies (unpublished) using Leonard LaPadula's voting concept for rule-based systems.