Policy Overlay Networks

Christopher C. Lamb

November 16, 2011

Abstract

Abstract - TBD

1 Introduction

Current enterprise computing systems are facing a troubling future. As things stand today, they are too expensive, unreliable, and information dissemination procedures are just too slow.

Generally, such systems still do not use current commercial resources as well as they could and use costly data partitioning schemes. Most of these kinds of systems use some combination of systems managed in house by the enterprise itself rather than exploiting lower cost cloud-enabled services. Furthermore, many of these systems have large maintenance loads imposed on them as a result of internal infrastructural requirements like data and database management or systems administration. In many cases networks containing sensitive data are separated from other internal networks

to enhance data security at the expense of productivity, leading to decreased working efficiencies and increased costs.

These kinds of large distributed systems suffer from a lack of stability and reliability as a direct result of their inflated provisioning and support costs. Simply put, they large cost and effort burden of these systems precludes the ability to implement the appropriate redundancy and fault tolerance in any but the absolutely most critical systems. Justifying the costs associated with standard reliability practices like diverse entry or geographically separated hot spares is more and more difficult to do unless forced by draconian legal policy or similarly dire business conditions.

Finally, the length of time between when a sensitive document or other type of data artifact is requested and when it can be delivered to a requester with acceptable need to view that artifact is prohibitively long. These kinds of sensitive artifacts, usually maintained on partitioned networks or systems, require large amounts of review by specially trained reviewers prior to release to data requesters. In cases where acquisition of this data is under heaving time constraints like sudden market shifts or other unexpected conditional changes this long review time can result in consequences ranging from financial losses to loss of life.

Federal computer systems are prime examples of these kinds of problematic distributed systems, and demonstrate the difficulty inherent in implementing new technical solutions. They, like other similar systems, need to be re-imagined to take advantage of radical market shifts in computational

provisioning.

2 Motivation

Current policy-centric systems are being forced to move to cloud environments and build much more open systems. Some of these environments will be private or hybrid cloud systems, where private clouds are infrastructure that is completely run and operated by an organization for wider use and provisioning, while hybrid clouds are combinations of private and public cloud systems. Driven by both cost savings and efficiency requirements, this migration will result in a loss of control of computing resources by involved organizations as they attempt to exploit economies of scale and utility computing.

Robust usage management will become an even more important issue in these environments. Federal organizations poised to benefit from this migration include agencies like the National Security Agency (NSA) and the Department of Defense (DoD), both of whom have large installed bases of compartmentalized and classified data. The DoD realizes the scope of this effort, understanding that such technical change must incorporate effectively sharing needed data with other federal agencies, foreign governments, and international organizations [**?**]. Likewise, the NSA is focused on exploiting cloud-centric systems to facilitate information dissemination and sharing [**?**].

Cloud systems certainly exhibit economic incentives for use, providing

cost savings and flexibility but they also have distinct disadvantages as well. Specifically, the are not intrinsically as private as some current systems, generally can be less secure than department-level solutions, and have the kind of trust issues that therapists cannot adequately address [**?**].

To begin with, cloud technology is not currently as private as some organizations would like:

- *User Data Control* — In virtually any given Software-as-a-Service (SaaS) scenario, user data controls are sadly lacking. Once data has been committed to a specific provider, that data is completely out of the original data owners control. Furthermore, as we will see below, that data my not even be solely owned by the original owner anymore either.

- *Secondary Use* — Most consumer facing social systems extensively mine user provided data for additional business advantages. This is a common and well known secondary use for supplied data. SaaS providers again have strong incentives to examine user provided information.

- *Offshore Development* — Service users have no real control over who actually develops the systems a given service deploys. Organizations have attempted to contractually limit development and support functions companies pursue to, say, the continental United States but have had very poor results with these kinds of unsupportable arrangements.

- *Data Routing* — System providers in fact have little control over rout-

ing issues as well as system users. Prohibiting data routing through sensitive countries is a difficult task for a single organization.

- *Secondary Storage* — Most large-scale systems expect to use Content Delivery Networks (CDNs) to help manage content, and that expectation is heavily reflected in their physical system architectures. They simply cannot divorce use of CDNs from their systems for a single organization.

- *Bankruptcy and Data Ownership* — Ownership and obligation to maintain expected data arrangements for a given company is unestablished under bankruptcy [**?**, **?**, **?**].

Security issues also emerge from utility computing infrastructures:

- *Data Access* — System users have very little control over who, in the system provider's organization, is able to access their data and systems.

- *Data Deletion* — Most savvy organizations have procedures in place to sanitize old storage elements like disk drives or backup tapes. System users have very little control over if and how this is done when computing services are treated as a utility.

- *Backup Data Storage* — Backup media is very difficult to encrypt, and most system providers still use tape systems as preferred media solutions for backup and storage needs. These tapes, or copies of them, are

generally stored offsite to support disaster recovery scenarios. Security of these types of systems has been spotty to date [**?**, **?**, **?**].

- *Intercloud Standardization* — Cloud computing systems do not have any standardized way to transfer computational units or data between systems. Any protocols used for this kind of thing must be developed by customers themselves. Due to the desire of providers to lock-in customers, this will likely not change as any standard development is strongly counter-incentiveized.

- *Multi-tenancy and Side-Channels* — Multi-tenant architectures in which multiple customers simultaneously use the same systems open those customers to covert side-channel attacks.

- *Logging and Auditing* — Logging and auditing structures, especially for inter-cloud systems, are non-existent.

Finally, such systems suffer from internal and external trust issues:

- *Trust Relationships* — Trust is difficult to establish between individual cloud providers long-term.

- *Consumer Trust* — Service users are still not entirely trusting of cloud system providers.

How to address these issues is an open research question. Organizations ranging from cloud service providers to the military are exploring how to

engineer solutions to these problems, and to more clearly understand the trade-offs required between selected system architectures [**?**]. The problems themselves are wide ranging, appearing in a variety of different systems. Military and other government systems are clearly impacted by these kinds of trust and security issues, and they also have clear information sensitivity semantics. This, coupled with the fact that these organizations have been dealing with these issues in one form or another for decades make them very well suited for prototypical implementation and study.

The current standard in place dealing with these issues in this environment is managed by the Unified Cross Domain Management Office (UCDMO). UCDMO stakeholders range from the DoD to the NSA. The current standard in place and governed by the UCDMO to deal with this kinds of issues are *guard-centric cross domain architectures*.

2.1 Current Solutions

Current and near-future proposed solutions endorsed by the UCDMO include system architectures assembled by the NSA, Raytheon, and Booz — Allen — Hamilton (BAH). The NSA has been active in this area for decades as a logical extension of their role in signals intelligence collection and processing. Raytheon and BAH have been engaged over the past few years to provide an alternative voice and design approach to these kinds of systems, an effort met with limited success.

These cross-domain solutions are intended to enable sensitive information

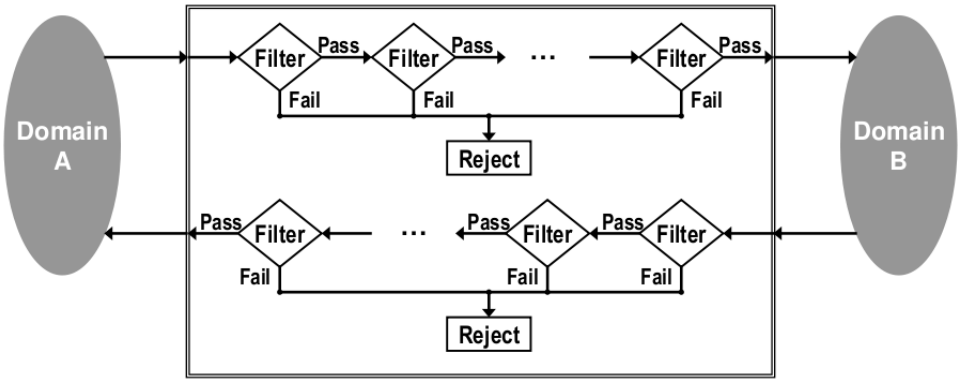

Figure 1: NSA Legacy Notional Architecture Model

to easily flow both from a higher sensitivity domain to a lower sensitivity domain, and from lower to higher as well. They generally act over both primary data (say, a document) and metadata over that primary data as well. Note that in these system, in most cases, human intervention is still required to adequately review data prior to passing into lower security domains.

2.1.1 NSA, Filtered

The NSA conducted initial work in this area. Their standard-setting efforts culminated in a reasonable conceptual system architecture, using groups of filters dedicated to specific delineated tasks to process sensitive information. [**?**].

In the scenario portrayed in figure 1, *Domain A* could very well be a private cloud managed by the U.S. Air Force, while *Domain B* is a public operational network of some kind shared by coalition partners in a joint operation.

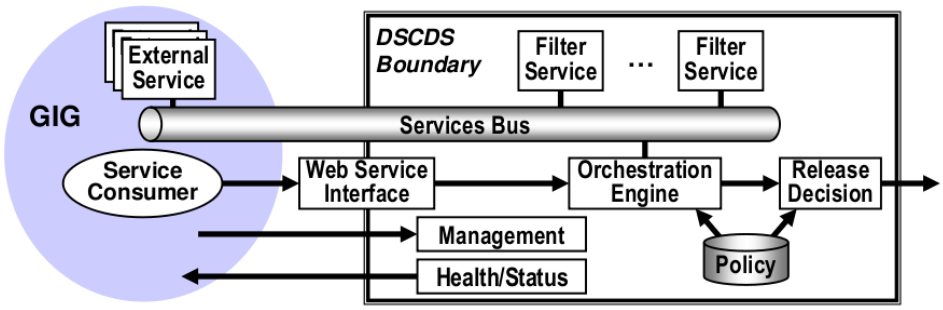

Figure 2: NSA Service-Oriented Model

A system user attempts to send a *data package* consisting of a primary document and associated metadata from *Domain A* to *Domain B*. At some point, that submission reaches a *guard*, which contains at least one *filter chain*. Each filter chain then contains at least one *filter*. Individual filters can execute arbitrary actions over a submitted data package and have access to any number of external resources as required. At any point, a filter can examine the data package and reject it, at which point it will frequently wait for human review. If a filter does not reject a data package, it passes that package onto the next filter or submits it for delivery to Domain B.

2.1.2 NSA, Services

In recent years, the NSA has extended the legacy system architecture for cross-domain information sharing to exploit service-oriented computing styles [**?**]. Visualized in 2, this model incorporates more modern conceptual elements and componentry.

In the view in Figure 2, we see on the left the *Global Information Grid*,

or *GIG*. On the right, we have the *Distributed Service-oriented Cross Domain Solution*, or *DSCDS*. The GIG is not a truly open system — rather, it is a loosely coupled collection of computational services handing data at a variety of levels of sensitivity, federated to provide stakeholder timely access to relevant information [**?**]. The DSCDS is essentially the embodiment of the NSA's cross-domain vision applied to service oriented computing. This model fuses various technology choices with previous cross-domain thinking.

Indicative of this more modern system design thinking, we have a variety of services and service consumers attached to a common service bus within the GIG. Within the DSCDS, we have groups of filters implemented as services inspecting transferred data when moved over the bus. Finally, all of this interaction is managed by a management interface and controlled by an orchestration engine accessing a centralized group of policies.

Note that here we have begun to access a common policy repository for various types of security metadata regarding primary data elements.

2.1.3 Raytheon

In the past few years, Raytheon has offered a new model for cross domain use influenced by the NSA service-oriented model [**?**]:

The model in Figure 3 is more grounded in the actual technical environment this kind of solution would be embedded within. Here, we have the Non-secure Internet Protocol Router Network (NIPRNet) as one domain, and the Secret Internet Protocol Router Network (SIPRNet) as the other.

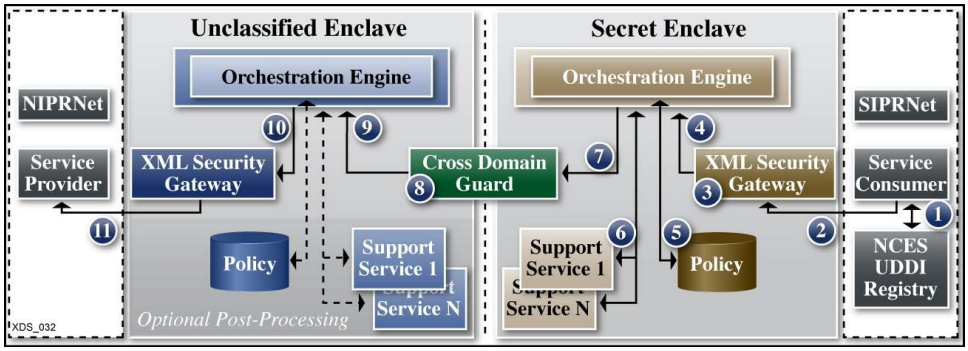

Figure 3: Ratheon Model

Here, NIPRNet is the lower security domain (lowside), and SIPRNet the higher security domain (highside). This particular view shows the motion of data from the high side (SIPRNet) to the low side (NIPRNet).

Here, a data request is submitted from SIPRNet first two the *XML Security Gateway* which calls into the *Orchestration Engine* for policy validation. The Orchstration Engine then coordinates calls into a *Policy Repository* as well as to a collection of external *Support Services*. Once rectified against these elements, the request is passed into the *Cross Domain Guard* which routes the request into the *Unclassified Enclave* in NIPRNet. Here, the request is passed directly through the lowside *XML Security Gateway*, without rectification, onto the *Service Provider*. The response from the Service Provider is then passed back to the requester via the inverse path.

This model also begins to use a centralized policy repository, just as the NSA Service Model. It also uses a single cross domain guard to transfer information from both the highside to the lowside, and vice-versa.

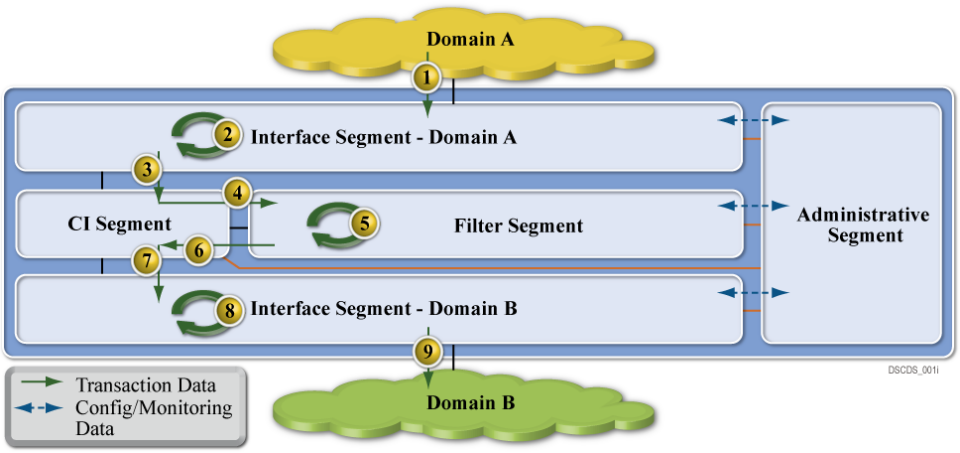

Figure 4: Booz — Allen — Hamilton Model

2.1.4 Booz — Allen — Hamilton

BAH submitted a competing model, also in 2009 [**?**]. In fact, both Raytheon and BAH presented their models under competitive contract to the UCDMO at the same conference, so the domain application is not coincidental.

Figure 4 embodies BAH's thinking with respect to cross domain information management. Note here, we have a *Domain A* as a high security domain, and *Domain B* as a low security domain. Here, we again have dataflow from the highside to the lowside through the cross domain management system.

While not as detailed as the Raytheon proposal, this does have similar elements. Here, we data first travels from Domain A into the *Interface Segment for Domain A*, similar to the secret enclave used in the Raytheon model From there, it moves into the *CI Segment*, which in turn submits the transferring ata into the *Filter Segment*. From there, the package is moved into the *Interface Segment for Domain B*, and then onto *Domain B*. The

Administrative Segment provides managment and oversight of the system as a whole.

Note the absence of specific policy-centric elements. This system is reliant on specific filters as well.

2.1.5 Shortcomings of Current Systems

Having reviewed the current state of the art of these kinds of cross domain solutions, they still have clear similarities, and in fact have not progressed far beyond the initial notions of how these kinds of systems should work. The still, for example, all use some kind of filter chaining mechanism to evaluate whether a given data item can be moved from a classified to an unclassified network. Both NSA models used filters explicitly, as did the BAH model. They all use a single guard as well, a sole point of security and enforcement, providing perimeter data security, but nothing else. In each of these current system architectures, users are only allowed to exchange one type of information per domain. The physical instantiations of these models are locked by operational policy to a single classification level limit. Users cannot, for example, have Top Secret material on a network accredited for Secret material. Finally, these models violate the end-to-end principle in large service network design, centralizing intelligence rather than pushing that intelligence down to the ends of the system [**?**].

2.1.6 Characteristics of Future Systems

Future systems on the whole will demonstrate decentralized policy management capabilities, infrastructural reuse, the ability to integrate with cloud systems, and security in depth. Policy management is decentralized and integrated within the fabric of the system. The system is both more secure and resilient as a result, better able to control information and operate under stressful conditions. Multi-tenancy can lower costs and increase reliability and is furthermore a common attribute of cloud systems. An appropriately secured system facilitates integration of computing resources into multi-tenant environments. The ability to handle multi-tenant environments and to reliably secure both data at rest and data in motion leads to computational environments deployable in cloud systems. Finally, systems must operate under *all* conditions, including when they are under attack or compromise [**?**]. Ergo, they must provide protection to sensitive data in depth.

2.2 Related Work

3 Proposed Taxonomy

A clear taxonomic organization of potential steps in approaching finer grained policy based usage management helps in describing the difficulties inherent in developing potential solutions as well as aiding in planning system evolution over time. Here, we have five distinct types of integrated policy-centric usage

Name	*Description*
ϕ	FooFooFooFooFooFooFooFooFooFooFooFooFoo FooFooFooFoo Foo
α	Bar
β	Bar
γ	Bar
δ	Bar

Table 1: Proposed Usage Management Taxonomy

management systems, as shown in Figure 1. Of these five, only the first two levels are represented in current system application.

In this taxonomy, it is not required that systems pass through lower levels to reach higher ones. This taxonomy represents a continuum of integration of usage management controls. Systems can very well be designed to fit into higher taxonomic categories without addressing lower categories. That said however, many of the supporting infrastructural services, like identification management or logging and tracing systems, are common between multiple levels.

The taxonomy itself starts with the current state, integrating policy evaluation systems into the network fabric gradually, moving away from filters, then by adding policy evaluation into the routing fabric, then the computational nodes, and finally by incorporating evaluation directly into content.

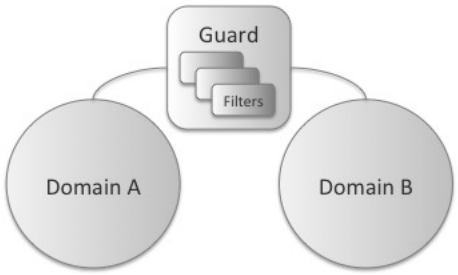

Figure 5: Taxonomy (ϕ)

3.1 ϕ — Single Guard, No Policies

The ϕ classification consists of systems like the initial NSA and BAH notional models in Figures 1 and 4.

These systems consist of two distinct domains, separated by a filter-centric single guard. The initial NSA system model is clearly of this type, separating two domains with a guard using filter chains. The BAH model is also of this type, using a Filter Segment to evaluate data packages transmitted between interface segments attached to specific domains.

Generally one of the domains supports more sensitive information than

the other, but that is not always the case. In the models we have examined this has certainly been true, but classified information for example is commonly stored in *compartments* which are separated by clear *need-to-know* policies generally enforced by access lists and classification guides. These kinds of compartments contain information at similar levels of classification, but contain distinct informational elements that should not be combined.

In these kinds of systems, specific rules regarding information transfer and domain characterization are tightly bound to individual filter implementations.

3.2 α — Single Guard, Policy Integration

The α category begins to integrate policy-centric management rather than using strict content filtering. The Raytheon model shown in Figure 3 and the service-oriented NSA model in Figure 2 are α architectures as they both use policies to guide usage management decisions.

Here, we again have at least two domains, Domain A and Domain B, though we could potentially have more. ϕ type systems require domain specific information to be tightly coupled to the filter implementations. Separating the permissions, obligations, and other constraints from the filters and incorporating them into a specific separate policy entity frees the Guard from this coupling and provides additional flexibility to the system.

The guard can continue to use filters to process data. These filters however are now more generic and decoupled from the specific domains it man-

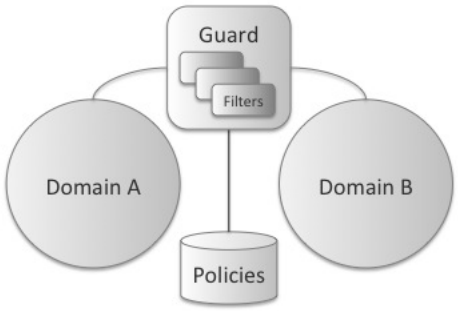

Figure 6: Taxonomy (α)

ages. The choice of using a specific filtering model rather than some other kind of construct is a design detail level to implementers. That said however, filters will be remarkably different and still need to understand the ontologies over which specific licenses are defined.

The policy repository is key to the implementation and differentiation of this taxonomy category. This repository can be implemented as a separate repository keyed into via a data artifact's unique URI, for example. It could also represent a policy sent in tandem with a data artifact in a data package.

The policy repository may be implemented as some kind of external service, and as such, represents the first such external service explicitly used

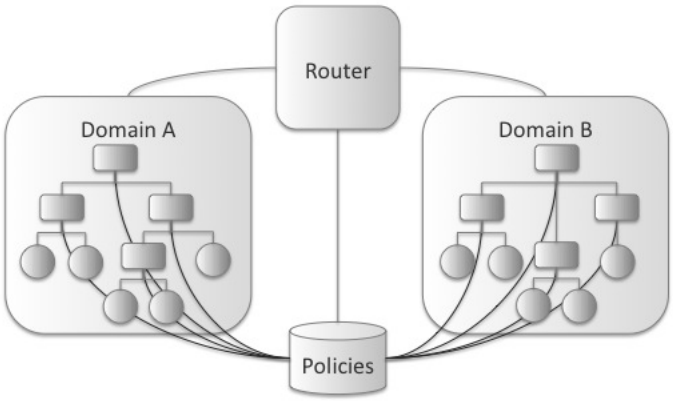

Figure 7: Taxonomy (β)

in this taxonomy. Other external services may well exist and be used to adjudicate information transfer decisions as well.

3.3 β — Router Guards

The β taxonomic category begins to integrate policy-centric processing with router elements in a given network. While this work is centered on using overlay technology to illustrate and implement these concepts, it is important to note that this kind of distributed policy-centric processing could very well be distributed into the physical routing fabric of a given network as well.

In this model we can as well host multiple domains as a result of flexible policy-based content examination. Each domain hosts a network of some kind, though that hosted network could very well be a degenerate network of a single system. Each network hosted in a domain is hierarchical, with specific computational nodes embodied by workstations, tablet computers or mobile devices, and routing points embodied by routers or switches of some kind.

Policy evaluation in this model has begun to penetrate into the routing elements of the specific domain networks. Here, note that we have started to penetrate into the routing fabric of the network by doing content evaluation at router points. Content-based switching networks have been successful in other domains, and such techniques can be used here to provide policy evaluation capabilities.

Certain types of traffic are easier to evaluate than others however. For example, HTTP requests and responses are easier to examine that TCP packets. When examining TCP packets, systems generally require additional context to select an appropriate packet window (e.g. the number of packets cached for examination). HTTP traffic does not usually require this kind of flexibility.

This migration of policy evaluation into the routing fabric provides for enhanced data security and better network management, especially if part of a network is compromised. Now that policy decisions can be made at the router level in a given network, we are starting to have network security

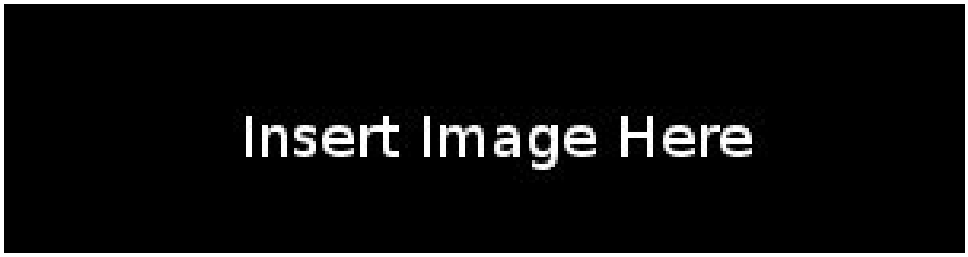

Figure 8: Taxonomy (β) Compartments

in depth rather than simple perimeter protection. This not only provides the ability for additional information protection, but also allows for different compartments holding information at different need-to-know levels to be created ad-hoc under different routing segments. Figure 9 shows this kind of configuration within a typical information domain. In cases of network compromise, this kind of dynamic policy enforcement can also allow for quick node excision as well.

3.4 γ — Router and Node Guards

The γ compartment has integrated policy evaluation with compute and routing nodes. Here, policies can be evaluated against content at all network levels — nodes emitting requests, nodes fielding requests, and all routing elements in between.

In Figure 10, we see that the policy repository is supplying services to all computational elements in both domains. This gives us increased granularity with respect to data compartmentalization by integrating information security into each network element. At this point, the network can create

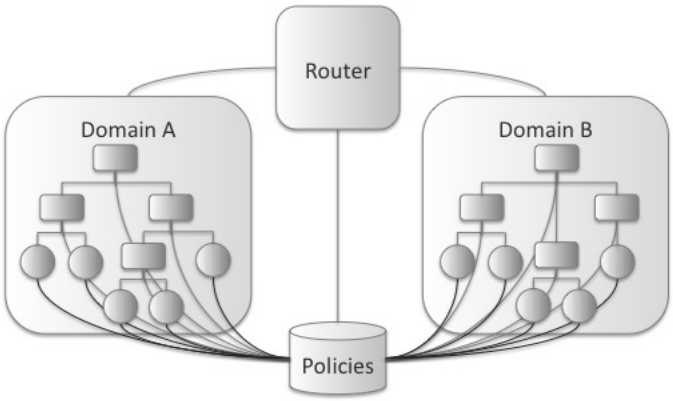

Figure 9: Taxonomy (γ)

compartments of single nodes, while previously in β level systems compartments could only be created under specific routing elements. At this level, we can also provide services revoking data access based on policy evaluation decisions when needed.

Furthermore, individual node exclusion is possible as well. β classified systems could excise network elements under specific routers by dynamic policy application. Now, we can apply the same functionality to individual compute nodes. For example, if a networked device like a smart phone is compromised, that device can be removed from access quickly or used to feed mis-information to adversaries.

3.5 δ — Continuous Evaluation

At this level of the taxonomy, we are not passing data packages around which we then examine and make judgments against. Rather, we now have a system based around a *smart license* where decisions regarding use and routing are asked of the data package itself.

Smart licenses simplify the deployment environment. They are essentially packages of mobile code, and as such require an environment in which to run but that environment does not need to support all the system elements that it would otherwise need to integrate. Usage history, management decisions, policy evaluation, and other components would be packaged into this smart license and distributed. Hosts would need to provide the appropriate environment in which these licenses run, but little else.

These kinds of licenses are able to manage dynamic context more effectively. As usage history accompanies each smart license, such usage no longer needs to be managed by an external service. Furthermore, offline use is much easier to implement as much of the information is omnipresent in the smart license package.

The cost for these advantages is additional license development complexity. The license needs to support offline use via some kind of caching mechanism and needs to be able to interact with the local environment as well as the larger enterprise environment. It is no longer a data-centric package of artifacts and meta-data, but is rather an actual program moving through various systems of the network.

4 System Architectural Implications

5 Scope and Contribution of Work

The unique contribution of this work is a quantitative analysis of policy-centric overlay network options, associated taxonomies of use, and prototypical technology proofs-of-concept. The process of examining and developing this system will address:

- *Overlay Options* — This includes various types of overlay networks and associated strengths and weaknesses addressing centralized and decentralized models
- *Taxonomies of Use* — Depending on the specific usage management requirements and context, different overlays have different applicability; this work will provide guidance on suitability
- *Prototypical Technologies* — Examples and proofs-of-concept will be required to appropriately analyze various architectural alternatives

This work will focus on the α, β, and γ taxonomic categories, excluding δ and ϕ classed systems.

6 Conclusions