

# Don't Trust. And Verify

## A Security Architecture Stack for the Cloud

This article is for security architects whose companies are moving critical systems to the cloud. Whenever technology architecture changes, it's worth revisiting the assumptions that went into security architecture and design. This is particularly relevant

for security and cloud computing; although many organizations rate security as a critical issue in moving to the cloud, few know what to do about it. Here I show some of the main trends that are driving new security technologies to cope with the security challenges that the cloud brings to the enterprise. In particular, I look at four promising technology “patterns” that form a stack of security architecture services for cloud computing.

### The Need for a New Security Model

There are many definitions of cloud. Here, I use Tim O'Reilly's definition:

Everything we think of as a computer today is really just a device that connects to the big computer that we are all collectively building.<sup>1</sup>

From this simple statement, it's obvious that the security model that most organizations use—network firewalls and Secure Sockets Layer (SSL)—is well past its sell-by date.

Now that organizations are moving to cloud-based services, connecting everything to everything else, security is naturally high on the list of concerns. Or-

ganizations have historically relied on the notion of a network perimeter. However, reality has diverged from this model when systems are connected using various cloud architectures:

- In *software as a service* (SaaS), customer, product, order, and other sensitive data is stored and processed off premises in applications and data that the enterprise doesn't control.
- In *platform as a service* (PaaS), platform rules dictate how security policy is created, managed, and enforced.
- In *infrastructure as a service* (IaaS), cloud provider rules dictate how network, physical, and host security is managed.

The “inside the firewall and outside the firewall” paradigm doesn't work when your company and its competitors and customers are all hosting data, applications, and identity in the same data store.

### Infrastructure, Infostructure, and Metastructure

The cloud changes security's role. Security no longer just provides structural boundaries at the infrastructure level. Instead, security is

an active participant in a dynamic, fluid environment. So, security must extend beyond infrastructure and into the *infostructure* (applications and data) and *metastructure* (policy).<sup>2</sup> Figure 1 illustrates these three levels.

Infostructure concerns include

- *service bindings*: service protocols and message formats,
- *service mediation*: services that mediate access to applications and data,
- *message and communication encryption*: confidentiality services at the data and transport level,
- *message and data integrity*: tamper-proofing messages and data, and
- *malicious usage*: dealing with asset abuse.

Metastructure concerns include

- *security token exchanges*: the ability to validate and issue security tokens;
- *security policy management*: policy definition, enforcement, and lifecycle management;
- *policy enforcement points*: mapping name spaces, resources, uniform resource identifiers, channels, and objects;
- *policy decision points*: the workflow for determining access;
- *message exchange patterns*: defining claims and schemas;
- *detection services*: logging and monitoring (recording and publishing events); and
- *key management processes*: key generation, distribution, and lifecycle management.

The issues at play here were

GUNNAR  
PETERSON  
Arctec Group

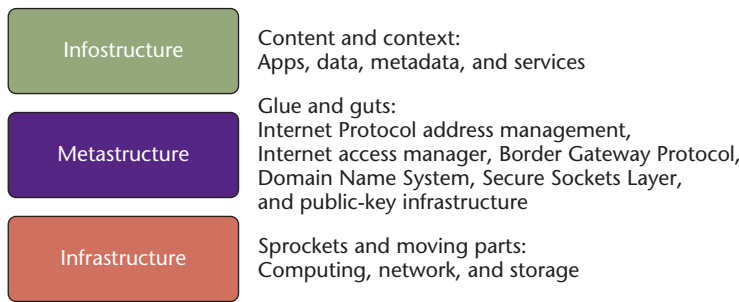


Figure 1. Christopher Hoff's layered separation of cloud architecture concerns.<sup>2</sup> Security must extend into all three layers.

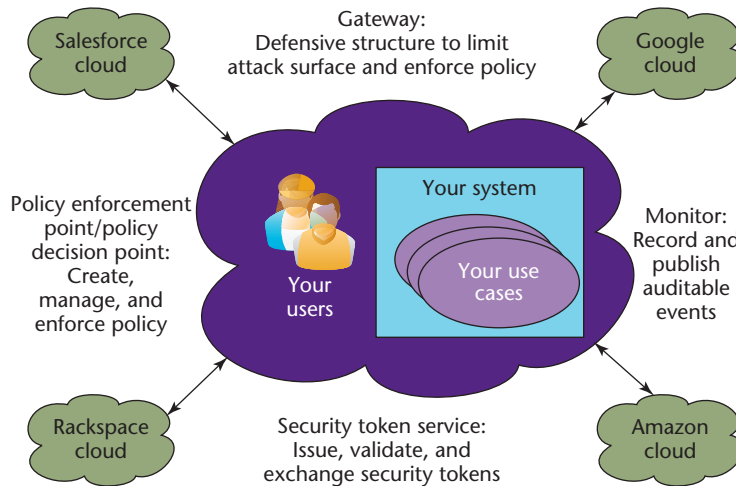


Figure 2. Security architecture services contextualize security policy for the cloud. Each pattern (gateways, monitoring, security token services, and policy enforcement points) lets security architects address security policy concerns in the metastructure and improve the runtime capabilities in the infostructure.

once small subsets of what security organizations did, but now they're the primary focus.

### The Four Patterns

The four technology patterns I mentioned in the introduction are *gateways*, *monitoring* (and logging), *security token services* (STSs), and *policy enforcement points* (PEPs). Figure 2 illustrates them in the cloud context. Each pattern lets security architects address security policy concerns in the metastructure and improve the runtime capabilities in the infostructure. Such a stack of security architecture services enables

- visibility into security events,
- attack surface reduction (I discuss attack surfaces in more detail later),
- context-specific security tokens, and
- fine-grained access control.

The combination puts security architecture groups in a more active role in the enterprise. They don't just passively pass "yea or nay" judgments on vulnerabilities; rather, they deliver protection and detection services that help the enterprise unlock value.

In a real-world deployment, these patterns don't necessarily

mean four separate pizza boxes. The services are frequently combined at runtime, and many runtime combinations and permutations are possible.

### Gateways

The gateway manages the attack surface for the cloud's entry and exit points. The attack surface comprises the data, methods, and channels that are integrated via the cloud. So, the gateway must have visibility across these layers, including data and application methods, and not merely into the communication channel, as network firewalls do. The gateway acts as a proxy for communication between the enterprise and the cloud. The other three patterns are essential security architecture elements, but the gateway remains a stylistic choice. Arguments can be made for both sides of the age-old agent-versus-proxy debate.<sup>3</sup> However, because this is a security article, I view attack surface management as a priority issue.

Gateways mediate all communication to and from cloud services, enabling more granular control of cloud use. Because management of security policy is decentralized, management costs should remain relatively low.

Like any technology, gateways should be considered in the context of the business use cases that they enable, whether this involves SaaS, PaaS, or IaaS. A service gateway lets an enterprise gracefully lose some but not all of its control over its security policy when it moves to the cloud.

Gateway security architecture responsibilities fall into five main categories.

*Communication channel security services* include transport encryption, transport authentication, transport integrity, proxy services, and protocol bridging.

*Message security services* include authentication (signing and verify-

ing a message—for example, using XML Signature), integrity (hash messages—for example, using XML), and encryption (message-level encryption—for example, using XML Encryption).

*Message processing* includes message transformation and content validation.

*Systems management* includes systems logging, administrative interfaces, and testing.

*Threat protection* includes input validation, protection against escaping data, XML denial of service (XDoS) protection, output encoding, and virus protection.

Gateways can be either hardware or software. They can serve as the delivery mechanism for any of the other security services I describe in the rest of this article.

### Monitoring and Logging

Monitoring implements network security monitoring, wiretaps, audit logs, and other tools to provide sensors for detecting security events. Because these systems are mainly passive, in cloud systems you typically can add them to the design without injuring service-level agreements.

Implementing this technology involves these questions: Where should you place the sensors? Who should operate them? What events are visible? When and how are the events published? Where are the events published to? Who should review the logs?

### Security Token Services

An STS has two main interfaces: it can issue a security token and validate a security token. The combination of these interfaces is the main reason STSs are critical for cloud computing. The ability to first validate and then issue security tokens means an STS can exchange tokens. Most enterprises have internal security tokens such as Kerberos tickets from Active Directory or X.509 certificates from the

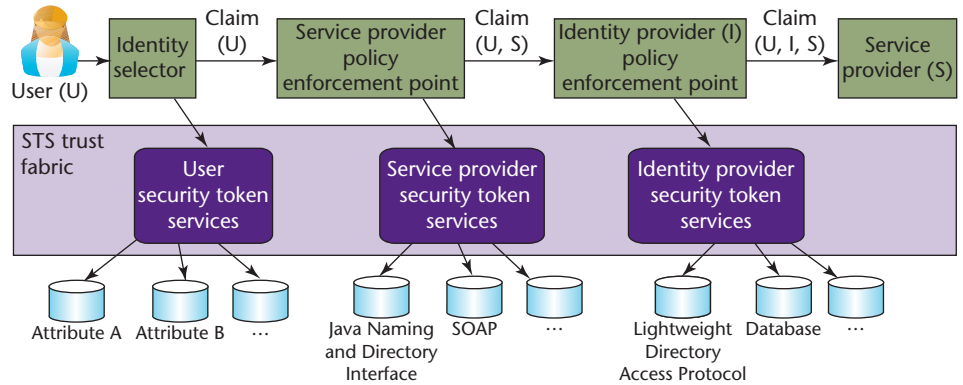


Figure 3. Service-provider-initiated security token services. These services' capabilities are similar to access management.

public-key infrastructure. However, the Web comprises OpenId and OAuth 2.0 bearer assertions, whereas cloud providers implement Security Assertion Markup Language (SAML) assertions and proprietary token types. The result is demand for a policy-based way to exchange tokens that ensures the secure exchange of attributes and session data.

You can deploy an STS in two main ways: on the identity provider (IDP) or service provider (SP) side. An STS's location dictates its function.

An IDP STS's responsibilities include

- subject-claim mapping,
- policy-based map requests and responses to tokens,
- policy-based route and transform requests and responses, and
- policy-based payload access.

In addition, the IDP must communicate with authentication systems, user stores, and directories such as those following the Lightweight Directory Access Protocol. The IDP might also be required to support any number of multifactor authentication systems. So, the STS can dynamically gather attributes and session data required by the relying party on the SP side.

An SP STS's responsibilities include:

- mapping of objects or resources to claims,
- policy-based mapping requests and responses to tokens,
- policy-based route and transform requests and responses, and
- policy-based payload access.

In this case, the SP STS must know about the managed objects—for example, Java Naming and Directory Interface trees, Java Database Connectivity connections, databases, and Web service methods.

In cloud scenarios, the enterprise end user can easily be in a situation where either deployment type is appropriate. So, it's important to remember that an STS varies by where and how it's used. On the SP-initiated side (see Figure 3), the capabilities are similar to access management—defining and enforcing access control policies. On the IDP-initiated side (see Figure 4), the capabilities are similar to an identity management suite; integration with directories and user stores is at a premium.

To enable the STS trust fabric, the federated identity standards must be agreed to—whether this means support for SAML or other federated standards. The STS's role is simple and straightforward—support, validate, issue, and exchange tokens—but the enterprise gains a composable security protocol.

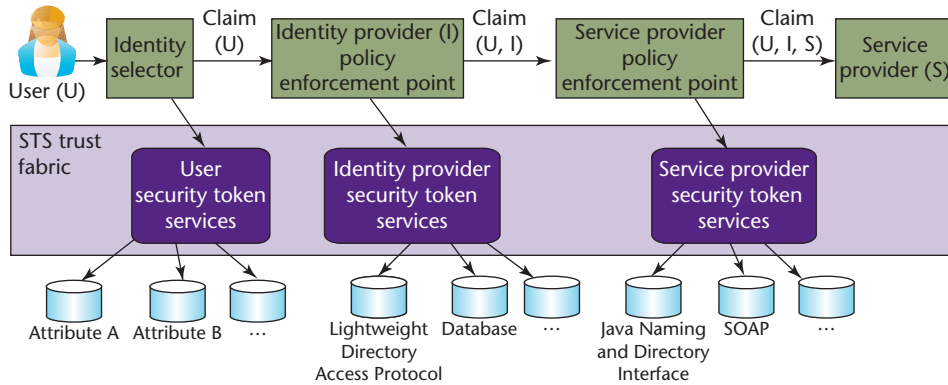


Figure 4. Identity-provider-initiated security token services. These services' capabilities are similar to an identity management suite.

### Policy Enforcement Points

In the security policy life cycle, the security architect must create, enforce, and manage policy. But in cloud architectures, these policy decisions are often mobile and dynamic. PEPs enable fine-grained, decentralized security policy decisions through languages such as Extensible Access Control Markup Language (XACML). These languages associate subject and object policy targets with rules specifying authorized conditions and actions. The security policy manager bundles these decisions into standards-based XML documents that can be transported and consumed across many disparate parts in the system.

This lets a PEP query *policy decision points* (PDPs) to make authorization decisions in a highly distributed way. For example, cloud services can use a PEP to identify where to bind to PDPs to make authorization decisions, embedding access control rules based on verifiable attributes in an object that's occasionally connected, such as a mobile device.

### Don't Trust. And Verify

The trends I've noted are all about enabling the business to derive cloud architectures' economic and scale benefits. This means the enterprise security program loses some control, but security architecture

isn't a zero-sum game. Wisely used, the technologies I've discussed can yield cost-effective protection and detection security services.

Security architects can learn something from Bill Gross's investment advice:

*Just last week Bank of England Governor Mervyn King said that it would be difficult to cut government spending quickly, but that there needs to be a clear plan for doing so. Not good enough, Mr. King. Don't care. Show investors the money, not vice versa. An investor's motto should be, "Don't trust any government and verify before you invest."*<sup>11</sup>

The main trends that will drive security architecture are visibility and verification, which we can pithily sum up as "Don't trust. And verify."

Enterprises are often told, even by security luminaries, that they must trust the cloud, but that's bunk. Sure, they must rely on some access control and other security services that are beyond their control. However, this can be partly mitigated by visibility services offered by gateways (chokepoints) and monitoring (audit event logging). In other words, a nickel's worth of visibility trumps a dollar of access control.

Basically, verification involves

good engineering. STSs can verify and issue the proper credential at the proper time, and PEPs can answer dynamic, fine-grained verifiable attribute queries at runtime. This will allow outsourcing of the commodity sprockets, while the key information remains verifiable with the enterprise authority. In cloud terms, the infrastructure is outsourced and inherently not trustable. The infostructure is responsible for verifying what it receives, and the metastructure defines where and how to perform the verification.

Many enterprise systems have two security modes: untrusted and fully trusted. Cloud security requires a partial-trust model. The four technology patterns I've examined allow partial trust through detection services and granular protection. □

### References

1. "What Is Cloud Computing?" video, Joyent, 2008; [www.youtube.com/watch?v=6PNuQHUIV3Q](http://www.youtube.com/watch?v=6PNuQHUIV3Q).
2. C. Hoff, "Incomplete Thought—Cloudanatomy: Infrastructure, Metastructure & Infostructure," blog, 19 June 2009; [www.rationalsurvivability.com/blog/?p=1070](http://www.rationalsurvivability.com/blog/?p=1070).
3. M. Ranum, "What Is 'Deep Inspection'?" [www.ranum.com/security/computer\\_security/editorials/deepinspect](http://www.ranum.com/security/computer_security/editorials/deepinspect).
4. W. Gross, "Don't Care," Pacific Investment Management Company, Mar. 2010; [www.pimco.com/Pages/Investment%20Outlook%20March%202010%20Bill%20Gross%20Dont%20Care.aspx](http://www.pimco.com/Pages/Investment%20Outlook%20March%202010%20Bill%20Gross%20Dont%20Care.aspx).

Gunnar Peterson is managing principal of Arctec Group. Contact him at [gunnar@arctecgroup.net](mailto:gunnar@arctecgroup.net).

**cn** Selected CS articles and columns are also available for free at <http://ComputingNow.computer.org>.