Composition Trust Bindings in Pervasive Computing Service Composition

John Buford
Panasonic Digital Networking Lab
Princeton, NJ, USA
buford@research.panasonic.com

Rakesh Kumar
Polytechnic University
Brooklyn, NY, USA
rkumar@utopia.poly.edu

Greg Perkins
Panasonic Digital Networking Lab
Princeton, NJ, USA
gmp@research.panasonic.com

Abstract

In pervasive computing, devices or peers may implement or compose services using services from other devices or peers, and may use components from various sources. A composition trust binding is a prescriptive set of rules which defines the combination of allowable components for a particular service or application. Composition trust bindings can be used to protect both the service invocation path as well as the content handling path. The subsidiary relationships addressed by a composition trust binding are typically transparent today, but represent potential security exposure in pervasive computing systems because the subsidiary services or components may have security vulnerabilities. We define the composition trust binding and illustrate its use in the context of rights management and distributed search in personal content publishing. We compare this approach to existing authentication and authorization methods in service composition.

1. Introduction

A node in a pervasive computing environment may offer a service to other nodes. The service may be composed or aggregated from services from other nodes, and any service may be composed of software components provided by different parties. By aggregating service facilities across nodes, a collection of limited-resource devices may be able to offer services that would otherwise not be available. However, nodes which invoke or participate in these services may be concerned about the integrity and trustworthiness of the various components that are combined to provide these services.

Conventional methods for assuring trustworthiness of software components include software audits and digitally signed software binaries; these methods are typically used to convey trustworthiness to the end-user or developer, and there is no explicit representation of trust between components. Further, these methods do not explicitly validate composite services that may be created from different service sources.

The approach presented here allows a node which creates, uses or participates in service delivery to enforce its service composition trust requirements by creating an explicit trust binding between the components that may participate in a service composition. We refer to this set of rules as a Composition Trust Binding (CTB). A CTB is a set of rules which define the collection of allowable components for a particular service. The primary interpretation of a CTB is: these components are permitted to be used in the combinations specified for implementing a service interface or processing specific content.

A service-invoking node can distribute a CTB to the service-providing node which is then expected to enforce the CTB policy as to which components are permissible for use in delivery of the service. Similarly a content-owning node can distribute a CTB to a service which processes the content, which is then expected to enforce the CTB policies during access to that content. Further, a service-providing node can publish a CTB for its composite services to aid nodes during service discovery to find services that satisfy the node's composition trust policy.

Conceptually the CTB extends the practice of digitally signed software, which is used to provide software component trust. The digital signature is a secure and verifiable indicator by the source or third-party validator of the component, and thus is a statement of the integrity of the component. Digitally signed software assures the component to the platform in which the component is installed, to other applications installed on that platform, and to users of applications on that platform. However this assurance is invisible to remote applications which invoke services on the platform which in turn use these

0-7695-2520-2/06 $20.00 © 2006 IEEE

components. Similarly, content owners whose content has been transferred to the platform for processing have no way to obtain assurance about the components processing the content by using digitally signed software alone.

The enforcement of a CTB provides additional assurance in networks where nodes in multiple administrative domains share computational resources and may be used to process information which is under an access control policy. This assurance can be given both dynamically and statically, in that the enforcement can be performed at the initial use of a service or periodically during the service use. A CTB validator is an agent that verifies that a particular execution combination of components, services and/or platform is valid according to the CTB required by the invoking node or application or user of the combination, or according to the CTB required by the content owner or licensor which is processed by the combination. CTB requires a mechanism for enforcement, such that the CTB and the enforcing agent can not be compromised.

In the next section we survey previous work addressing security and trust issues in service composition. Section 3 identifies and describes four categories of distribution composition. Section 4 presents the generic CTB, and Section 5 describes examples using CTB in service composition. Section 6 reviews representation issues in the CTB and section 7 describes techniques by which a CTB validation agent can be implemented. Section 8 concludes the paper.

2. Related Work

Service composition is the construction of complex services from primitive ones, thus enabling rapid and flexible creation of new services. Work specifically related to security in service composition is summarized below.

The SAHARA service composition framework [1] addresses the use and enforcement of service level agreements between multiple service providers. SAHARA includes an authorization control framework that enables an authorization decision based on local authorization rules and credentials from external domains.

In [2] Gu et al. propose a *service oriented* P2P system called *P2P service overlay* where peers can provide not only media files but also a number of application service components. The proposed service overlay avoids security problems in dynamic code uploading but does not address the mutual trust of components of services.

DisCo [3] addresses secure deployment of distributed applications across administrative domains. It provides a pluggable interface for component authorization that can support different access control models. In addition, components are deployed in an execution environment that enforces access control permissions for each component. DisCo addresses component security through continuous enforcement of authorization and access control of each component. It does not specifically distinguish between policies that satisfy users, resource-providers, and content-owners. Run-time enforcement of access rights of components is a complementary security technique to the prescription of which components instances are permitted in a service composition.

The Secure Service Discovery System (SSDS) [4] provides wide-area service discovery using a hierarchy of servers to provide scalability. Centralized authentication servers provide authentication for service lookup. SSDS is primary focused on the security of the service discovery method.

SPKI (Simple Public Key Infrastructure) [5] defines a digital certificate in which authorization rights are described in an s-expression in the certificate. The resource owner maintains an access control list (ACL). The certificate binds the key holder to an authorization right; the ACL contains the resource owner's binding of keyholders and authorization rights. Peers providing the services (the resource owners) can use SPKI to authorize other peers' use of the services. A service-invoking peer or a peer providing protected content might construct SPKI certificates with s-expressions representing their respective composition trust binding. However from the perspective of the service-providing peers, these certificates represent authorization requirements from the requesting peers and content owners rather than authorization rights for a local resource. Whereas in SPKI, access control is a local decision with the certificate serving to bind the remote party with the access right, in this scenario a remote decision must be made and confirmed. This mechanism is currently not defined in the SPKI model.

3. Service Composition

Categories of service composition that are important for pervasive computing include:

- Virtual devices
- Multimodal interfaces
- Computational concurrency or load distribution

A device's software and hardware components can be packaged as services and combined in arbitrary ways. Many consumer electronics (CE) devices are specialized for specific uses. Due to form factor and cost considerations, devices vary in capability. With sufficiently high bandwidth network interfaces on these devices, such as 802.11 and UWB, it is practical for sets of networked devices to share functionality.

Combining different devices can extend the user interface of the device and aggregated devices can be composed into new virtual devices:

- A video camera networked to a cell phone can use the cell phone's IMP software (Figure 1, left) to send instant messages
- A video camera networked to a cell phone or car audio receiver can augment the memory of such devices by storing information from either device on its SD card (Figure 1, middle)
- A video camera networked to a car flipdown video display and a cell phone can use the former to display its user interface and video playback, and the latter as an input device for keypad input (Figure 1, right)

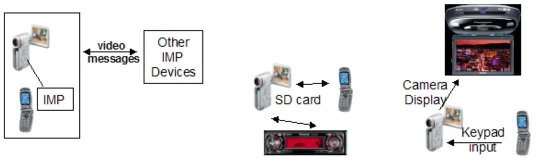

Figure 1 Examples of device composition

Multimodal user interfaces can be created by combining and coordinating user input/output from multiple devices. For example, geographic maps and location awareness from the car navigation system can be combined with streaming video about nearby landmarks to a camera display and speech input from a cellphone.

Decomposition of computation across multiple nodes can be used for concurrency or distributing computational load, as in grid computing. Applications for computation concurrency with personal CE devices include content-based retrieval, semantic search, image analysis, information fusion, and simulation.

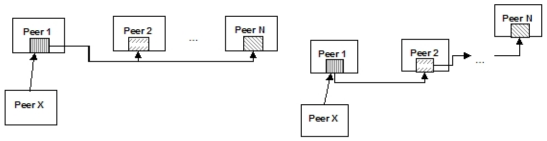

Figure 2 Service composition patterns

Networked devices can offer hardware and software components as services which can be dynamically discovered using various service discovery protocols. For example (Figure 2, left) a service might invoke services provided by several other peers as part of performing the indicated service, and service composition might be nested (Figure 2, right).

The selection of a component in such distributed computations can be done in a number of different ways, such as using a service discovery protocol, prescription by the invoking node, or configuration. To make the dynamic composition explicit in such scenarios, service descriptions can be augmented to identify sub-interfaces used (Figure 3).

```
Service [
    interface1 […]
    interface2 [ … ] uses
         interface3 [  … ] or
         { interface4[…] and interface5[…]}
]
```

Figure 3 Simplified service interface definition in which components of interface2 are explicitly identified

While service composition enables a number of interesting applications and provides additional flexibility, it creates new vulnerabilities. These are possible because any component in a composition is a potential exposure to backdoors, trojan horses, security holes, and masquerading components.

In general, these threats concern not only the specific node providing the service but also any node or peer invoking the service (control path) and any node or peer whose data or content are processed by the service (data path). Consequently we claim that existing techniques for software integrity such as digitally signed software or peer authentication are insufficient because they fail to address component security from the perspective of the service invoker or content provider. A way to securely prescribe the allowed components in these types of compositions is needed.

4. Composition Trust Binding and Service Composition

A CTB is a set of rules which define the collection of allowable components for a particular service. The primary interpretation of a CTB is: these components are permitted to be used in the combinations specified for implementing a service interface or processing specific content. Due to space limitations, we omit the CTB template definition but show an example in the next section. The CTB contains the following elements:

- An id by which the CTB can be identified

- The identity of the owner of the CTB
- The service description the CTB applies to
- The content object(s) which the CTB applies to
- One or more component rules, each specifying the permitted components, component suppliers, component validators and expiration time of this prescription. The component rule can list components in various boolean combinations

The CTB identifies permitted combinations of components, services, operating system, hardware, and may include other information such as decryption keys for parameters or content, content identifiers, identifiers for the source of the binding, user identification, and license identification. In a CTB, the component supplier could be a third party, OS vendor, or the content issuer. There may be multiple possible suppliers/validators for a given component. If there are several components, acceptable combinations of components can be associated, each combination in a separate rule. A component may be a software library, application, service interface, operating system, or platform

Components, services, hardware, and OS may have unique immutable identification, version information, and other descriptive attributes which are cryptographically protected and verifiable, such as digitally signed software module or hardware or OS digital certificate; these identifications are used by the CTB validation agent to determine if the available components match the binding requirements.

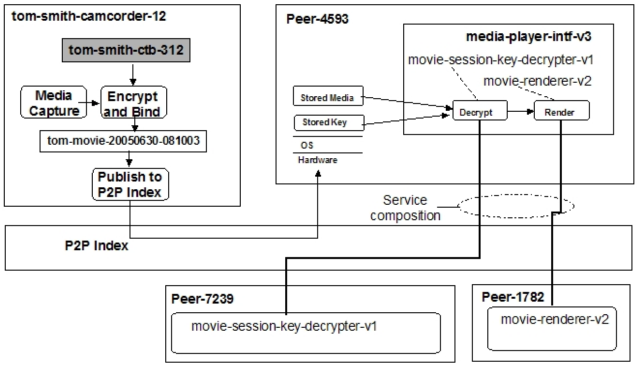

Figure 4 Use of CTB in personal content publishing in a peer-to-peer example

5. Example

This example of the use of CTB is divided into the data path and control path. The data path example (Figure 4) illustrates personal content publishing in a peer-to-peer environment. In this example, the camcorder used to capture the media also immediately encrypts the media, applies the owner's rights management policy, and prepares it for publication to a wide-area peer-to-peer index. Additionally, the camcorder incorporates a CTB (labeled "tom-smith-ctb-312") which is either encrypted into the content file or encrypted with the license file for the content.

Once the content is published in the peer-to-peer (P2P) index, any peer may retrieve it using existing methods for keyword search in peer-to-peer file sharing systems [6]. In Figure 4, peer-4593 has retrieved the media file "tom-movie-20050630-081003" from the P2P index. Peer-4593 uses a local service (media-player-intf-v3) which plays the content, assuming the peer also has the appropriate license. In addition, this media player service uses two components which may be either local or provided by other peers. In this simplified example, the components provide two key functions of the media player: media decryption and media rendering. Peer-7239 and Peer-1782 have previously registered services in the P2P index which correspond to these interfaces. Peer-4593 can discover these services and use them to perform the necessary function.

Because the content owner Tom Smith has associated a CTB with the encrypted media file, the CTB is used to enforce Tom Smith's policies about which components can be used to implement the media-player service.

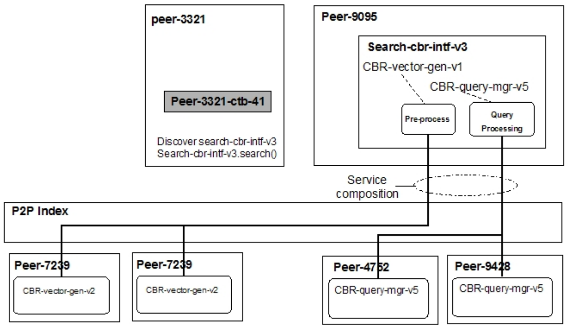

Figure 5 CTB example in control path

In Figure 5 the example is continued to show use of the CTB in the control path. In this example, peer-3321 discovers peer-9095 which offers a service for content-based retrieval (CBR). The service search-cbr-intf-v3 uses two component services, one for pre-processing the CBR vectors (CBR-vector-gen-v1) and the other for managing queries (CBR-query-mgr-v5). Peer-9095 offloads the computational load of the CBR by distributing the vector generation and query processing to other available peers.

Peer-3321 invokes the CBR service on peer-9095 and includes its CTB with the request. Peer-332-ctb-41 stipulates the components that satisfy Peer-3321's security policy with respect to operation of the CBR service.

6. CTB Representation

The utility of the CTB approach depends on the ability of the binding to represent the policies and composition scenarios of interest, and on the ability to securely enforce the CTB in a distributed context. We outline an approach to the latter issue in the next section.

Representation issues effecting the CTB include:

- updating the CTB for changing service interfaces, component interfaces, and component suppliers
- complexity of the CTB

Experience in the evolution of various software platforms such as Java, DCOM, and CORBA shows that interfaces change relatively slowly compared to implementations. Nevertheless there is the case that a service offering peer might move to a new version of an interface with a different composition model before the service user or content provider has validated these and produced a new CTB. Further, implementations that had previously been validated might be obsoleted as when the vendor no longer supports them or the vendor no longer exists. A content licensee might be left without the ability to use the content and a service user might face service interruption. These problems are not unique to CTBs (such as software that no longer runs on an upgraded OS). A solution for CTBs that cannot be updated is a backward compatible deployment of the necessary services, and for CTBs that can be updated, a mechanism by which content licensees can obtain updated CTBs as needed.

Composition complexity relates to the size of each interface, the number of component interfaces, and the nesting depth of the composition. We expect the service composition is effective if the unit of composition is an interface rather than individual methods, and that interface complexity in existing service-oriented architectures is representative of what to expect in peer-to-peer service composition. If service composition uses interfaces as the unit, then the CTB complexity will always be less than the corresponding service descriptions.

A service may be composed of other services to an arbitrary nesting level. A CTB might prescribe only the first level components, if the validated component services include CTBs that the composite CTB trusts. A CTB might prescribe component compositions to several levels, but this increases the complexity of the CTB and makes it more difficult to maintain.

The CTB also becomes sizeable if components are implemented by large numbers of software providers.

Finally, the CTB assumes that content publishers and service users are aware of services from different implementers. Past experience with component suppliers for distributed middleware such as DCOM, CORBA and Java EJB suggests that this is manageable, and online registries could be created to streamline this communication.

7. Securing and Validating Composition Trust Bindings

A CTB validator is an agent that verifies that a particular execution combination of components, services and/or peers is valid according to the trust binding required by the invoking peer or application or user of the combination, or according to the trust binding required by the content owner or licensor which is processed by the combination.

The validation of the binding may be performed by a validation agent in a component participating in the application or service, or in the operating system. The validation agent may communicate with each component, OS and hardware, and may communicate with validation agents on other platforms if components or services are located on those other platforms. The validation agent may be itself secured via a secure OS on a trusted computing base, a smart card, Java Card, or other security technology. The validation agent could be integrated with the system loader to monitor launching of components on a given platform.

In a separate paper [9] we review the security technologies such as Trusted Computing Base to secure an agent for peer-to-peer trust negotiation. Securing the validation agent is a similar problem except where the CTB constrains nested compositions which requires that validation agents coordinate with downstream agents to insure compliance. The enforcement of constraints on nested components appears to be recursively decomposible. That is, the local agent communicates with each remote agent enforcing the next level service composition. The local agent trusts that the remote agent enforces the immediate composition as well as nested constraints. Each remote agent repeats the process for its nested compositions.

Each component of a binding must have a secure, unique, verifiable id. The binding must be encrypted, such as by using the peer's public key when the CTB is generated. The CTB is not intended to describe dynamic variations in component use at different points in a process lifetime. The validator may not be able to enforce or monitor all possible communication paths between possible components, and the CTB is not intended to replace access control and authorization mechanisms. The strength of the CTB is to express a known set of component relationships that have been validated through other means (e.g., software audit and integration testing) for a specified environment or platform, so that components adhering to the specified service interfaces and signed by trusted parties are expected to adhere to the desired access policy with greater reliability than component combinations that have not be validated.

A digital signature on the component indicates that the software is from the given supplier. Alternatively if the software is signed by a content issuer, it could indicate that the content issuer authenticates the software as a playback component for its content. Or, if the software is signed by a third party validator, it could be that the software has been audited or validated by the third party. The intent then is to provide assurance to the content issuer and the licensee that the component does not have a backdoor, trojan horse, or other hole that would lead to the encryption keys being exposed. The signature is an indirect statement that the supplier of the component has validated the integrity of the component for its functional purpose, whereas the CTB is a statement that a prescribed set of components from possibly many sources are considered reliable and trustworthy for the indicated service.

8. Summary

In pervasive computing, peers may implement or compose services using services from other peers, and may use components from various sources. A composition trust binding is a prescriptive set of rules which defines the combination of allowable components for a particular service or application. We have defined the composition trust binding and illustrate its use in the context of rights management and distributed search in personal content publishing.

Existing authentication and authorization methods in service composition do not address the trust requirements of distributed composition. Conceptually the CTB extends the practice of digitally signed software, which is used to provide software component trust. But the assurance of digitally signed software is invisible to remote applications which invoke services on the platform which in turn use these components. Similarly, content owners whose content has been transferred to the platform for processing have no way to obtain assurance about the components processing the content by using digitally signed software alone.

Finally, the enforcement of a CTB provides additional assurance in networks where nodes in multiple administrative domains share computational resources and may be used to process information which is under an access control policy. We have described some of the implementation techniques available to enforcing CTBs using validation agents.

9. References

[1] Raman, B., et al. 2002. The SAHARA Model for Service Composition across Multiple Providers. In *Proceedings of the First international Conference on Pervasive Computing* (August 26 - 28, 2002). F. Mattern and M. Naghshineh, Eds. Lecture Notes In Computer Science, vol. 2414. Springer-Verlag, London, 1-14.

[2] Gu, X., Nahrstedt, K., and Yu, B. 2004. SpiderNet: An Integrated Peer-to-Peer Service Composition Framework. In *Proceedings of the 13th IEEE international Symposium on High Performance Distributed Computing (Hpdc'04) - Volume 00* (June 04 - 06, 2004). HPDC. IEEE Computer Society, Washington, DC, 110-119.

[3] Freudenthal, E. and Karamcheti, V. 2004. DisCo: Middleware for Securely Deploying Decomposable Services in Partly Trusted Environments. In *Proceedings of the 24th international Conference on Distributed Computing Systems (Icdcs'04)* (March 24 - 26, 2004). ICDCS. IEEE Computer Society, Washington, DC, 494-503.

[4] Czerwinski, S. E., et al. 1999. An architecture for a secure service discovery service. In *Proceedings of the 5th Annual ACM/IEEE international Conference on Mobile Computing and Networking* (Seattle, Washington, United States, August 15 - 19, 1999). MobiCom '99. ACM Press, New York, NY, 24-35

[5] C. Ellison, et al. SPKI Certificate Theory. IETF RFC 2693. Sept. 1999.

[6] Androutsellis-Theotokis, S. and Spinellis, D. 2004. A survey of peer-to-peer content distribution technologies. *ACM Comput. Surv.* 36, 4 (Dec. 2004), 335-371.

[7] King, I., Ng, C. H., and Sia, K. C. 2004. Distributed content-based visual information retrieval system on peer-to-peer networks. *ACM Trans. Inf. Syst.* 22, 3 (Jul. 2004), 477-501

[8] Müller, W. and Henrich, A. 2003. Fast retrieval of high-dimensional feature vectors in P2P networks using compact peer data summaries. In *Proceedings of the 5th ACM SIGMM international Workshop on Multimedia information Retrieval* (Berkeley, California, November 07 - 07, 2003). MIR '03. ACM Press, New York, NY, 79-86

[9] J. Buford, I.-P. Park, G. Perkins. Social Certificates and Trust Negotiation. IEEE Computer Communications and Networking Conference (CCNC) 2006. Jan. 2006.