JavaSE6基于JSR105的XML签名之理论篇 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/252/2021_2022_JavaSE6_E5_9F_BA_c104_252487.htm XML签名技术，这项在W3C建议中指定的XML签名语法及处理方法（XML-Signature Syntax and Processing），成为解决SOA开发中消息级安全性方案的基础。被普遍接受的OASIS标准WS-安全性（WS-Security）正是构建在这一技术(以及XML加密)基础之上。JSR-105规范又对在Java平台应用XML签名技术进一步标准化，并且将成为即将到来的Java SE 6发行版本的一个组成部分。本系列文章（《理论篇》与《实践篇》）将基于Java SE 6的试发行版本对JSR-105作入门性介绍；在第二篇（即《实践篇》）中，我们将讨论一个具体的应用案例。 一、 数据一致性和消息认证 XML数字签名的主要目的是确保数据一致性。RFC 2828，因特网安全词汇表（Internet Security Glossary），把"一致性"定义为"在一种未授权的或偶然方式下确保数据没有改变、破坏或丢失的属性"。在这种意义上，与一个校验和一起存储或传递数据就可以实现数据的一致性。严格地说，XML签名能够实现比这种一致性更为丰富的内涵-它能够为在RFC 2828中所谓的消息认证提供支持。 二、 签名元素结构 实质上，XML签名使用XML语义描述一个数字签名。下列层次捕获顶级的元素和属性以及它们之间的结构化关系。＜Signature ID?＞ ＜SignedInfo＞ ＜CanonicalizationMethod/＞ ＜SignatureMethod/＞ (＜Reference URI? ＞ (＜Transforms＞)? ＜DigestMethod＞ ＜DigestValue＞ ＜/Reference＞) ＜/SignedInfo＞ ＜SignatureValue＞ (＜KeyInfo＞)? (＜Object

ID?＞)*＜/Signature＞ 在这个示例中，?表示零个或一个出现，表示一个或多个出现，而*表示零或多个出现。在此，所有的元素和属性被定义于命名空间http://www.w3.org/2000/09/XMLdsig#。 在此，Reference担当连接要签名的数据对象与一个XML签名之间的桥梁作用(通过URI属性)。一个应用程序选择这里的digest方法来计算一个数据对象的digest值，并且把这二者作为相应的Reference元素的一部分。 对于digest方法，W3C建议实现对SHA-1的支持。而且，这种实现通常还支持其它交互式单向哈希函数-例如SHA-256，SHA-512和RIPEMD 160。 实际上，我们很少直接从数据对象本身计算一个digest值。通常，一个应用程序需要首先对数据对象应用一些转换。例如，我们可以使用XPath来从一个XML文档中仅提取关键元素以实现签名；或者，我们也可能在使用XSLT经过一些转换后对一个XML文档进行签名。这样的转换是在Transforms元素中指定的-其中包含一个有关实现转换算法及其它相关信息的Transforms的有序列，当然也包括在相应的Reference元素之内。 SignedInfo是数字签名算法实际应用的元素。该算法通过SignatureMethod元素捕获；W3C建议中要求实现对DSA_SHA1，RSA_SHA1和HMAC_SHA1(由JSR-105所注释)的支持。前两个是基于公钥的，而HMAC是一种对称密钥密码学算法。 