第 27 卷第 3 期　　四川理工学院学报(自然科学版)　　Vol. 27　No. 3
2014 年 6 月　　Journal of Sichuan University of Science & Engineering(Natural Science Edition)　　Jun. 2014
文章编号:1673-1549(2014)03-0059-06　　DOI:10.11863/j.suse.2014.03.14

基于 SSL 证书认证中间件的安全性分析

何　旭[1], 鲜乾坤[2], 雷建平[3]

(1. 达州职业技术学院, 四川　达州　635001; 2. 四川理工学院计算机学院, 四川　自贡　643000;
3. 61902 部队, 四川　宜宾　644000)

摘　要:安全套接字层 SSL 证书认证具有信任链验证、主机验证、证书撤销及证书扩展机制。定义了抽象化 SSL 协议栈内部 SSL 库的使用,扩展了用于连接的缺省值设置方法,阐述了信任链验证的 OpenSSL 应用程序接口具有不同有效主机名组成限制及证书列表匹配要求。在分析 OpenSSL 内部数据结构及数据运输层建立 HTTPS 连接机制的基础之上,揭示了 JSSE 不执行自身主机验证建立 SSL 连接的缺陷,并提出通过形式验证技术和编程来实现自动检查应用程序是否正确使用 SSL 库的关键选项与参数的解决策略。

关键词:安全套按字;证书认证;信任链;安全超文本传输协议;安全漏洞
中图分类号:TP393.08　　**文献标志码**:A

引　言

Web 浏览器的 SSL 已经成为网络安全的事实标准,SSL 主要目的是针对中间件攻击提供端到端的安全保障,即使网络受到像 DNS 入侵或者路由器被控制等这样的威胁,SSL 也能够保证客户机与服务器之间通信的机密性、真实性、完整性[1-2]。

认证服务器是 SSL 连接建立的关键部分,其公钥证书身份验证发生于 SSL 握手阶段。同时,为了保证 SSL 连接是安全的,客户端必须确认证书是有效证书,即既没有过期也没有被撤销,用以检查证书名称、客户端连接的匹配性[3-4]。

Web 浏览器对 SSL 功能的实现可以通过加入补丁保证安全性,这样许多与 SSL 相关的浏览器漏洞就可以得到修复[5-6]。然而,SSL 也广泛用于非浏览器软件,如基于云计算虚拟基础设施的远程管理以及发送本地数据到云存储,以及 Android、iOS 等移动应用程序的验证服务。这些程序通常不实现 SSL 本身,主要依靠 OpenSSL、JSSE 与 CryptoAPI 等 SSL 库以及更高层次数据传输库作为 SSL 封闭,在基于 Web 服务软件用额外抽象层引入 Web 服务中间件。

入侵者可能控制路由器、交换机、无线 AP 或 DNS,也可以控制一个或多个拥有 SSL 证书的服务器。当 SSL 客户机试图连接到一台合法服务器时,入侵者可以通过 DNS 挟持诱导服务器,并欺骗用户连接到一台入侵者控制的服务器。

1　SSL 证书认证机制

SSL 握手协议是一个更高层的客户端 SSL 协议,该协议是用来协商会话的安全属性,握手消息提供给 SSL 记录层,并封装在一个或多个 SSLPlaintext 数据结构内,用以处理与传输当前活动会话状态。SSL 连接开始于客

收稿日期:2014-03-19
基金项目:四川省教育厅科研项目(14ZA0330)
作者简介:何 旭(1969-),男,四川平昌人,副教授,硕士,主要从事信息安全、网络计算、网络关键技术方面的研究,(E-mail)cnhexu@163.com

户端和服务器之间的握手,这样 SSL 客户端的“确认服务器认证”容易作为攻击目标。SSL 握手协议由 RFC6101[7] 给出了完整描述,其握手过程如图 1 所示。

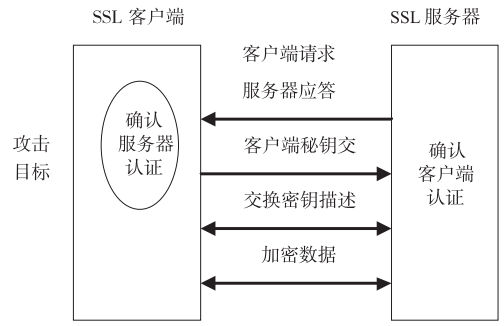

图 1 SSL 握手协议描述

信任链验证。每个 X.509 证书有一个“issuer”属性,其中包含 CA 的名称,每台 SSL 客户端配置可信根 CA 证书列表。除自己的证书外,还有该服务器发送的发行 CA 认证。如果签发的 CA 不是根 CA,服务器还发送一个更高级别通向根 CA 的证书列表。客户端试图建立一个从底部服务器证书开始的链,链中每个证书必须立即由上级 CA 签署,根 CA 必须是客户信任的 CA 之一,客户验证证书没有过期、中间 CA 认证以及设置有“Basic Constraints”字段。完整的 X.509 证书认证算法由 RFC5280[3] 与 RFC2818[4] 给出了描述。

主机验证。信任链建立后,客户端必须验证服务器的身份。RFC2818[4] 建议使用“SubjectAltNames”作为服务器标识符主要来源以及支持“Common Name”向后兼容,但大多数软件是先检查“Common Name”。构建服务器标识列表后,客户端尝试匹配所有符合 DNS 域名的被请求服务器标识符。如果客户端在服务器标识符列表中找到一条精确匹配,就通过简单字符串比较进行验证,客户端也可以查找一个匹配规则复杂的通配符名字标识符的列表[4,8]。

证书撤销。OpenSSL 的一些 SSL 库实现证书撤销[9],但要求应用程序提供证书撤销列表(CRL)。虽然 JSSE 库要求应用程序检查自身 CRL 的有效性,但大多数应用程序不用检查,如有些 SSL 库(Python 的 SSL)不公开 CRL 的检查方法。

X.509 扩展。一些 X.509 证书扩展包含关键安全信息,如 key useage(CA 允许使用该秘钥签署证书)、name constraints(限制子 CA 能够签署的认证)以及 RFC2527[10] 描述的证书策略。虽然 CA 可以分配不同的信任级别给子 CA,但应用程序必须提供利用这些信息的策略,实际上这些扩展在很大程度上可以被忽略,即或是目前的 OpenSSL 也没有正确验证名字约束,而 cURL 甚至没有接口用于指定应用程序证书策略。

2 SSL 抽象化安全分析

根据需求,应用程序可以插入到不同层次抽象的 SSL 内,但在最低层次有多种使用不同属性、许可、硬件请求来实现通用 SSL,如 OpenSSL、JSSE、CryptoAPI 等。因此,为了避免自己解析 HTTP,所以使用 SSL 的 HTTP 应用程序通常不直接使用,而使用 HTTPS 在内部使用 SSL 库,SOAP 或基于 REST 的 Web 服务应用程序在其上层使用额外中间件 HTTPS 或 Web Socket,如图 2 所示。

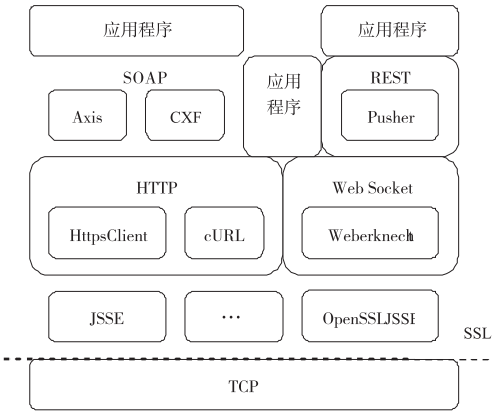

图 2 使用 SSL 的协议栈

2.1 SSL 库

OpenSSL。OpenSSL 只提供信任链验证,而应用程序必须支持自身主机验证。不同应用层协议(HTTPS,LDAP)有不同有效主机组成限制和证书列表匹配要求,因此主机验证必须通过应用程序自身管理或数据传输层封装。OpenSSL 通过提供返回函数或修改配置允许应用程序定制信任链验证[11],如配置“验证深度(verify depth)”和“验证模式(verify mode)”(图 3)。

使用 OpenSSL 的程序可以通过调用 SSL_connect 函数执行 SSL 握手。证书验证错误信息通过 SSL_connect 函数返回值提供,而其他错误由 SSL_connect 函数设置内部 verify result 标识,应用程序必须调用 SSL_get_verify_result 函数以检查是否发生此类任何错误[12]。

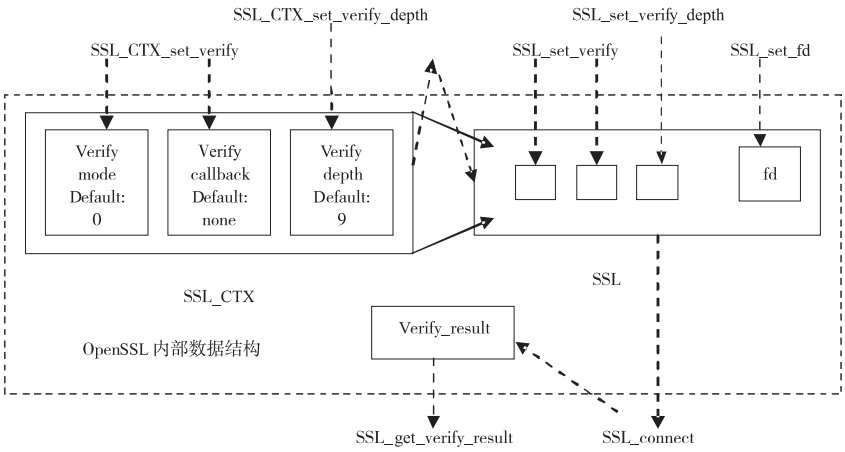

图 3　使用缺省信任链验证 SSL 连接设置的 OpenSSL API 图

JSSE。Java 安全套接字扩展(JSSE)提供大量接口给 Java 应用程序(包括 Android 移动应用程序)建立 SSL 连接。低层通过 SSLSocketFactory 创建的 SSL 客户端 API 可能不执行主机验证。

下面的代码段来自于 X. 509 的 TrustManagerImpl. checkIdentity。

```
private void checkIdentity(String hostname,
X509Certificate cert, String algorithm)
throws CertificateException {
if (algorithm! =null&&algorithm.length()! =0){
....
if (algorithm.equalsIgnoreCase("HTTPS")){
HostnameChecker. getInstance (HostnameChecker.
TYPE_TLS).match(hostname,cert);
} else if (algorithm.equalsIgnoreCase
("LDAP")) {
HostnameChecker. getInstance (HostnameChecker.
TYPE_LDAP).match(hostname, cert);
} else {
throw new CertificateException("Unknown
identification algorithm: " + algorithm);
                }
        }
}
```

上述代码段的功能是实现对主机名与认证的确认。如果 algorithm 设置为 HTTPS 或 LDAP,checkIdentity 方法就抛出一个异常,其不同于 OpenSSL 的返回值,即使验证失败也认为应用程序会检查该值。在 HttpsClient 和 HttpsURLConnection 等 JSSE 应用程序中,当创建 SSL 客户端时就会调用 SetHostnameVerification 方法并设置为 HTTPS,其结果是调用 HostnameChecker 和验证证书。

如果客户端字段是 NULL 或空串,checkIdentitysilently 跳过主机验证且不抛出异常。该作法是为了适应基于证书协议实现而重用 JSSE 的默认值 HTTPS 或 LDAP 以验证信任链。Java7. 3 的证书验证代码与 Java6 不同,如果字段是 NULL 或空串,checkIdentity 根本就不调用。

```
private void checkTrusted(X509Certificate[] chain,
String authType, Socket socket, boolean isClient)
throws CertificateException {
...
String identityAlg = sslSocket.getSSLParameters().
getEndpointIdentificationAlgorithm();
if (identityAlg! =null&&identityAlg.length! =0){
String hostname = session.getPeerHost();
checkIdentity(hostname, chain[0], identityAlg);
        }
}
```

上述代码段实现对信任链的验证功能,验证链路中各节点主机间的信任关系。在 SSL 客户端使用 SSLSocketFactory,如果该字段为空 JSSE 就不执行主机验证,主机验证工作就由运行在 JSSE 上层的软件完成,该功

能在 JSSE 参考指南有警告说明。

在发送任何数据前使用 SSLSockets/SSLEngines 应该检查各节点证书,因为 SSLSocket 和 SSLEngine 类不自动验证与 URL 匹配的域名验证,如果不进行主机验证应用程序,这样就可以利用 URL 欺骗。Java 软件使用 SSLSocketFactory 创建 SSL 客户端但不执行主机验证,说明开发人员并没意识到该特性,而 JSSE 接口执行主机验证也增加了其混乱性。

2.2 数据运输层的安全机制

实际上大多数应用程序依赖数据运输层建立 HTTPS 连接,这些在内部使用 SSL 库架构通常是不透明的应用程序。

Apache HttpClient。Apache HttpClient 是基于 JDK 的客户端 HTTP Java 库。Apache HttpClient 广泛用于 Web 服务中间件,是因为本地化 JDK 不支持 SOAP Web 服务,且在发送 HTTP POST 请求等功能时,Apache HttpClient 比 JDK 提供了更好性能。Apache HttpClient 使用 JSSE 的 SSLSocketFactory 建立 SSL 连接,即 Apache Http 客户端必须执行自己的主机验证,这样对于基于旧版本的 HttpClient 不验证主机就会导致很多漏洞。Apache HttpClient 使用 HttpHost 数据结构描述 HTTP 连接,而 HttpHost 并没有任何内部一致性检查,如允许连接到 443 端口。

cURL。cURL 是一个从远程服务器获取数据的工具和库。从 7.10 版本开始,cURL 默认验证 SSL 证书。内部 cURL 使用 OpenSSL 验证信任链和主机,其功能由参数 CURLOPT_SSL_VERIFYPEER(默认值 true)和 CURLOPT_SSL_VERIFYHOST(默认值 2)控制,VERIFYPEER 参数是布尔类型,而 VERIFYHOST 参数是整数。开发人员经常误解这些参数,而且将 CURLOPT_SSL_VERIFYHOST 设置为 TRUE,从而改变其值为 1,这样如果意外禁用主机验证就会带来不安全的后果。

PHP。PHP 提供多种方法建立 SSL 连接,打开远程服务器套接字的 fsockopen 能够通过在 URL 中加入“SSL://”连接到 SSL 服务器。尽管 fsockopen 不执行任何证书检查,PHP 应用程序开发人员通常用它来建立 SSL 连接。PHP 也提供 cURL 绑定,使用 cURL 默认设置建立 SSL 连接以适应证书验证,开发人员可能设置 cURL 不正确选项改变默认值,从而破坏证书验证。

3 非浏览器软件 SSL 的分析

云客户端 API。云端 API 是连接云端与客户端的桥梁,它包括面向各不同协议客户端的 API 和面向第三方的 API。前者的作用是云端与客户端之间的交互,使用 RTMP 或 RTMPE 协议传输;后者允许定制特定函数输出给第三方,使用 RTMP 或 RTMPE 传输,或通过本地的 PHP 中转为 HTTP 传输。对于第三方云计算平台的运行在客户端 SDK,可以通过第三方软件传送用户数据到基于云的存储、管理用户基于云的计算以及访问其他云服务。如 Amazon 在 Java、PHP、Python 和 Perl 中提供的 EC2 API 工具,以及 Apache 提供的独立库访问多个云的 Libcloud。

Web services 中间件。许多程序依赖于 Web services。Web services 是一软件系统,用来支持网络主机间的互操作性,其服务由一个接口描述计算机可读的 XML 格式文档。不同系统提供不同具体实现接口,异构系统间通过发送和接收信息实现服务的互操作。

使用基于 XML 的 SOAP 或 REST 的 Web services 发送与接收消息。从客户端软件的角度来讲,Web services 可以被认为是提供远程过程调用 RPC 接口,而 SOAP 或 REST 中间件是 RPC 调用参数的封装和解封。

Web service 通过使用 SOAP 中间件与现有 Java 软件进行交互。同样地,如果一个 Android 应用程序需要实时发送通知,它可以使用客户端库连接到基于 REST 的 Pusher 服务。这些中间件框架负责发送 Web service 信息到网络,如果连接必须是安全的,中间件通常使用 SSL,而不是现在的 SSL 连接管理数据传输库代替。

4 Apache HttpClient 的 SSL API 安全性应用分析

2007 年发布并广泛使用的 Apache HttpClient 3.1 版本与较早期的版本一样,使用 JSSE 的 SSLSocketFactory 而不执行自身主机验证来建立 SSL 连接。这样 Apache HttpClient 3.* 接受任何具有有效信任链的证书,而不管其名字。

HttpClient 主机验证的缺陷在 4.0 版得到修正,在 4.2.1 版本中,有其自身的主机验证和指派 JSSE 进行信

任链验证。这样依靠 SSL 连接建立 HttpClient 实现对应用程序的安全性有较少影响。HttpClient 的使用通常隐藏内部 Web services 中间件,从而跳过主机关于 SSL 证书认证。

值得注意的是,如果用户将自定义主机验证代码添加到 HttpClient 4. * 是错误的,并且将拒绝有效证书。下面的代码来自 HttpClient 4. 2. 1:

```
String parts[ ] = cn. split( " \\. " ) ;
boolean doWildcard = parts. length > = 3 &&
parts[0]. endsWith( " * " ) &&
acceptableCountryWildcard( cn) &&
! isIPAddress( host) ;
if( doWildcard) {
if ( parts[0]. length( ) >1) {
String prefix = parts[0]. substring(0, parts. length -
2) ;
String suffix = cn. substring( parts[0]. length( ) ) ;
String hostSuffix = hostName. substring
( prefix. length( ) ) ;
match = hostName. startsWith( prefix)
&& hostSuffix. endsWith( suffix) ;
} else {
match = hostName. endsWith( cn. substring(1) ) ;
}
if( match && strictWithSubDomains) {
match = countDots( hostName) = = countDots( cn) ;
}
} else {
match = hostName. equals( cn) ;
}
```

该代码段具有计算来自 CN 的主机名验证、通配符使用处理、CN 与主机通配符忽略处理功能。其方法是计算参数 parts 的数字部分减去 2 的前缀长度,域名第一部分有效性什么也不操作而其就不具有逻辑性。如,如果证书名是 a * . < b > . < b > . com,它将拒绝 ail. < b > . < b > . com。而且最初的补丁及其衍生代码在解析 IPv4 地址通用表达式中有一个小缺陷,使其接受从 0 开始的 IP 地址,但这并不会立即导致安全漏洞。

5　结　论

通过对具有代表性的中间件应用程序使用 SSL 用于安全连接的情况分析,特别对中间件库使用 SSL 作为多层软件协议栈的一部分功能分析,发现程序在具有潜在不安全的公用网络中传输极其敏感的数据时,非常重要的一点就是必须正确使用 SSL。

同时通过对 SSL 在应用程序软件设计中的使用,详细说明了依赖标准 SSL 库的应用程序(如 JSSE、OpenSSL 等)执行 SSL 证书认证可能存在的不安全因素,这些不安全性在有些软件中是随处可见的,如 FPS、PayPal、EC2 以及其他客户端远程管理云存储和云基础设施等。这些特点对中间件攻击来讲,SSL 连接是没有安全性的。

另外,对于一些宣称是更加安全的 SSL 策略,通过分析发现,可以通过开发代码分析工具发现 SSL 连接建立逻辑错误问题。对于这类情况,可以通过形式验证技术设计和编程,支持自动检查应用程序是否正确使用 SSL 库,而不误解关键选项和参数的定义,以此来设计更好的 API 用于 SSL 或其他安全的网络协议,从而解决其不安全隐患等。

参 考 文 献:

[1] 舒 剑.一种无证书认证密钥协商协议的分析与改进[J].计算机应用研究,2012,29(1):294-296.

[2] 张延红,陈 明.标准模型下强安全的无证书认证密钥协商协议[J].四川大学学报:工程科学版,2013,45(1):125-132.

[3] HTTP over TLS[EB/OL].(2000-05-01)[2014-03-10]. http://www.ietf.org/rfc/rfc2818.txt.

[4] Internet X.509 public key infrastructure certificate and certificate revocation list(CRL)profile[EB/OL].(2008-05-01)[2014-03-10].http://tools.ietf.org/html/rfc5280.

[5] 何 旭.一种可扩展 Web 模型安全机制[J].计算机系统应用,2012,21(8):89-93.

[6] 何 旭.隐私模式浏览器的安全性分析[J].西华大学学报:自然科学版,2012,31(3):17-22.

[7] The Secure Sockets Layer(SSL)protocol version 3. 0

[EB/OL]. (2011-08-01)[2014-03-11]. http://tools. ietf. org/html/rfc6101.

[8] Representation and verification of domain-based application service identity within Internet public key infrastructure using X. 509 (PKIX) certificates in the context of Transport Layer Security (TLS) [EB/OL]. (2011-03-01) [2014-03-11]. http://tools.ietf.org/html/rfc6125.

[9] https should check CN of x509 cert[EB/OL]. (2007-04-22)[2014-03-12]. https://issues. apache. org/jira/browse/HTTPCLIENT-613.

[10] Internet X. 509 public key infrastructure certificate policy and certification practices framework[EB/OL]. (1999-03-01)[2014-03-11]. http://www. ietf. org/rfc/rfc2527.txt.

[11] Viega J,Messier M. Secure programming cookbook for C and C + +[M]. California:O'Reilly Media,2003.

[12] Moxie M. IE SSL vulnerability[EB/OL]. (2002-05-08)[2014-03-12]. http://www. thoughtcrime. org/ie-ssl-chain.txt.

Security Analysis of Middleware Based on SSL Certificate Authentication

HE Xu[1], *XIAN Qiankun*[2], *LEI Jianping*[3]

(1. Dazhou Vocational and Technical College, Dazhou 635001, China; 2. School of Computer Science, Sichuan University of Science & Engineering, Zigong 643000, China; 3. 61902 Troop, Yibin 644000, China)

Abstract: Secure Socket Layer (SSL) certificate authentication has the trust chain, host authentication, certificate revocation and extension mechanism. The use of SSL library in the abstract SSL protocol stack is defined, and the setting method of default value used to connection is expanded. Then, it expounds that the trust chain validation OpenSSL application interfaces have different valid hostname composition limit and matching requirements of certificate list. Based on the analysis of HTTPS connection mechanism built by OpenSSL internal data structure and data transport layer, the defect that JSSE does not perform its host authentication to build SSL connection is revealed, and then the solving strategy that realize automatic check whether the application is correct to use key options and parameters of SSL library by formal verification techniques and programming, is put forward.

Key words: SSL; certificate authentication; trust chain; HTTPS; security vulnerabilities