计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2018.20170451
55(6):1157-1166,2018

一种基于 HTTP/2 协议的隐蔽序列信道方法

刘政祎　　嵩　天
(北京理工大学计算机学院　北京　100081)
(liuzhengyi@bit.edu.cn)

Covert Sequence Channel Based on HTTP/2 Protocol

Liu Zhengyi and Song Tian
(*School of Computer Science, Beijing Institute of Technology, Beijing* 100081)

Abstract　Covert communication technology offers effective privacy-preserving and secure data transmission services with covertness in behavior and content. Existing covert storage channels have always been questioned about their covertness. On the other hand, covert timing channels mainly use middle and lower layer network protocols as overt channels, which usually requires complex encoding methods to reduce bit error rates. It is hard to satisfy the transmission rate requirements through current covert timing channels as well. In this paper, we present H2CSC, a new covert sequence channel approach over the next-generation application layer HTTP/2 protocol. H2CSC controls and manipulates the responses of HTTP/2 Web server to its requests, forming a kind of covert sequence from the stream IDs of those response frames. Then, H2CSC exploits combinatorial coding methods to embed covert bits into these frame sequences. It takes advantage of HTTP/2 protocol to provide channel reliability and security. We implement H2CSC method in the widely used Apache Web server as a function module, and examine the channel's effectiveness and robustness in the real system. We further evaluate the covertness of this channel by using a detection method based on logistic regression of corrected conditional entropy. The experimental results show that H2CSC could provide 574 bps of covert transmission rates with excellent robustness and covertness.

Key words　covert channel; HTTP/2 protocol; data frame sequence; combinatorial coding methods; corrected conditional entropy

摘　要　隐蔽通信技术能够为使用者提供有效保证隐私安全的数据传输服务.现有存储类隐蔽信道一直存在隐蔽安全性疑问,而时间类信道较多选择网络及以下层协议作为载体,需额外提供复杂编码方法以降低误码率,且难以提供足够的传输速率.以新一代应用层协议 HTTP/2 为基础,提出了一种新的隐蔽信道方法——H2CSC.该方法通过控制 HTTP/2 协议服务器响应的数据传输过程,通过修改待发送数据帧的发送顺序,使用组合数学编码方法在数据帧序列中隐蔽消息,充分利用了 HTTP/2 协议提供信道可靠性及安全性.H2CSC 方法在广泛使用的 Apache Web 服务器中以功能模块形式予以实现,并通过真实系统对该方法的有效性和可靠性进行测试,使用基于修正条件熵的逻辑回归分类检测方法进行安全性测试.实验证明:H2CSC 方法能够达到 574 bps 的隐蔽通信速度,具有较高的健壮性和隐蔽性.

收稿日期:2017-06-12;**修回日期**:2017-12-19
基金项目:国家自然科学基金项目(U1636119,61272510,61672101)
This work was supported by the National Natural Science Foundation of China (U1636119, 61272510, 61672101).
通信作者:嵩天(songtian@bit.edu.cn)

关键词 隐蔽信道;HTTP/2 协议;数据帧序列;组合数学编码方法;修正条件熵

中图法分类号 TP393

隐蔽信道(covert channel)是指允许进程以危害系统安全策略的方式传输信息的通信信道[1-2]. 随着网络安全防护日益严格,防火墙和网络流量审查等功能越来越多地对用户隐私构成严重威胁. 传统提供隐私数据保护的通信技术主要是以保护数据安全的加密隧道及重路由技术如 Tor 和 VPN 等为主. 此类方法的通信协议具有明显特征,网络监管部门能够有效检测其通信连接,获取传输节点,并进一步监视或截断传输节点的数据流量,故其并不能有效保护通信信道安全. 隐蔽通信技术能够提供在网络流量内容之外传输数据的功能,且其协议设计上具有较高的复杂性和多样性,使得网络监管难以检测或对抗. 因此在特定需求下,隐蔽信道可以作为一种有效保护传输内容机密性和通信信道隐蔽性的网络通信方法,而事实上该方法也已经广泛应用于网络数据安全传输领域.

HTTP/2(Hypertext Transfer Protocol version 2)协议是自 1999 年公布的 HTTP 1.1 协议以来的首个官方更新,提供了众多包括异步连接复用、头压缩和请求反馈管线化等特性,并能够与 HTTP 1.1 语义完全兼容. 目前主流浏览器均强制要求以 TLS 协议加密会话,以对抗网络中间人攻击及流量监管,具有较高的安全性.

本文关注在 HTTP/2 协议通信过程中构建隐蔽信道方法,考虑以 HTTP/2 服务器作为隐蔽消息发布节点,将消息编码至 HTTP 客户端请求访问页面的过程中,向隐蔽消息接收节点下发信息. 不同会话之间互相独立,即编码解码过程仅在同一个 HTTP/2 协议会话过程中进行计算处理. 本文重点提出一种基于 HTTP/2 协议对序列信息进行编码的隐蔽信道 H2CSC(HTTP/2 covert sequence channel). 具体来说,该方法通过利用 HTTP/2 协议中基于帧的多路复用(multiplexing)技术,将待发送的隐蔽消息分组编码至传输过程中数据帧的流编号所组成的序列信息中. 该信道采用基于组合数学的序列编码方法,将二进制数据转换为排列组合问题进行编码处理,并进行优化以实现最大提高可编码信息量. 最终本文使用基于修正条件熵的逻辑回归分类方法检测该方法隐蔽性并提出熵值抚平方法,以进一步提高本方法的隐蔽性和安全性.

1 相关研究

1.1 隐蔽信道

1973 年 Lampson[1] 第 1 次提出“covert channel”,其最初的定义是建立于多安全等级(multilevel security, MLS)的自动化信息系统(automated information system, AIS)中,主要针对在不同主体和对象间的访问控制及安全分类. 如今,其主要针对高度互联的网络,实体为网络应用程序或网络节点. 因而隐蔽信道的另一个更广泛更具适应性的定义为“策略上被拒绝但性质上允许的通信信道”. 据此定义,隐蔽信道可以延伸至网络传输中的各个协议,尤其是以 HTTP 等应用层协议为公开信道(overt channel)的隐蔽通信方法. 本文所指网络传输中的隐蔽信道,依据信息发送者和接收者访问的某个或某些共享资源的属性,一般来说分为基于存储的信道和基于时间的信道.

基于存储的隐蔽信道,主要选取将网络传输协议中的某些次要字段进行隐蔽消息填充的方式以实现隐蔽通信,如将某些无用、保留或随机字段替换为指定数据,如谭庆丰等人[3] 设计实现的基于 P2P 协议 DHT 网络键值对的 StegoP2P 信道. 基于 HTTP 协议的此类信道可对包括 URI, Cookies, Entity-body, HTML 页面代码[4] 等内容进行替换,或使用 HTTP 请求头域中不同字段的顺序交错实现编码[5] 等方法实现隐蔽信道的构建. 此类方法一般能够达到较高的隐蔽数据传输速率,但由于填充了实际信息,将这类流量与正常流量进行对比将产生概率上分布偏差,易被协议分析方法识别,且这类识别引擎易于部署,识别率较高[6-8]. 此外 Aggarwal 等人[9] 将隐写术(steganography)应用于 HTML 页面代码,列举包括是否存在空标签或无用空格等多种编码隐蔽信息的不同方法. 对于此类基于隐写术的隐蔽信道可通过相关针对性方法实现该信道的识别及消除[9-10].

基于时间的隐蔽信道主要利用网络传输数据包的时间属性信息加以编码,如选择单位时间片内有无数据包发送或到达,或判断 2 个数据包间隔时间是否在某个区间内以实现编码. 此类信道最初于 2004 年由 Cabuk 等人[11] 提出,他们设计了一种基于 IP 协议以判断时间段内是否发送数据包实现编码

的时间类隐蔽信道,被称作 IPCTC(IP covert timing channel),类似的还有基于 TCP 协议的时间类隐蔽信道[12]. 随后 2005 年 Berk 等人[13]设计了一种利用数据包间隔时间(inter-packet delays),令编码比特 1 的时间间隔为编码比特 0 的时间间隔的 2 倍. 钱玉文等人[14]将时间间隔应用至 HTTP 协议实现双工通信,以进一步提高此类信道的鲁棒性. 一般情况下此类隐蔽信道具有较高的隐蔽性,且在通信双方网络距离较近时稳定性较好,但在相距较远时会受到较大干扰,导致误码率较高,并且其隐蔽数据传输速率较低. 在 HTTP 协议中构建时间隐蔽信道,除了基本网络层传输层构建方法以实现外,Kolegov 等人[15]介绍了一种基于 HTTP 协议 Cache 字段的时间类隐蔽信道. 另外 Ji 等人[16]设计一种基于传输数据报文长度的隐蔽信道并实现在 HTTP 协议上,此方法的数据传输速率相比其他时间类信道较好,但其需要相应的网络先验数据,且仅以发送者生成的正常网络数据作为基准,并不能保证特定网络环境的特征. 针对此类方法一般采用两大类检测方法,即形状(shape)参数检测和规律性(regularity)检测[8]. 其中形状参数检测主要分析统计数据包括均值、方差和概率分布等相关一阶统计参数,如 Kolmogorov-Smirnov 方法和熵值检测方法[8,17]. 规律性检测主要分析统计数据的数据间相关性,如基于修正条件熵(corrected conditional entropy)及 Regularity 的相关测试方法[17-19].

除上述两大类隐蔽信道外,还有一些基于通信行为以组合数学思想实现编码的隐蔽信道构建方法. 如 2012 年 Luo 等人[20]提出的一种基于 TCP 的枚举组合的 Cloak 隐蔽信道,其将多个 TCP 流合并在一起构成了符合 The 12-Fold Way 模型的数据编码方法. 该方法是第 1 个引入组合数学思想实现编码的隐蔽信道构建方法,其能够较好地平衡信道的隐蔽性与传输速率,但需要多个 TCP 连接用于构建信道,反而在一定意义上因行为异常而降低了隐蔽性. Shen 等人[21]在 2015 年提出 LiHB,一种基于 HTTP 行为建立隐蔽信道的方法,其利用连续的 HTTP 请求组合成流以其不同序列进行编码,即每 1 组隐蔽消息可对应编码为 1 组若干特定 HTTP 请求流的排列. 此方法隐蔽性相对较高,然而该方法隐蔽数据传输速率随参数呈对数增长趋势,实验中也仅最高达到 201 bps,传输速率相对不足. 对于此类隐蔽信道检测方法一般通过对通信数据提取合适的特征信息,并带入上述针对时间类隐蔽信道的形状参数及规律性等检测方法以实现.

1.2 HTTP/2 协议

超文本传输协议第 2 版由互联网工程任务组(IETF)的 Hypertext Transfer Protocol Bis 工作小组进行开发. HTTP/2 标准于 2015 年 5 月以 RFC 7540[22]正式发表. W3techs 网站统计,截至 2017 年 11 月,已有 20.6%的网站支持 HTTP/2 协议[23].

HTTP/2 协议向前保留了 HTTP 1.1 协议包括方法、状态码、标头字段、URI 等高级语法,但加入了在客户端和服务器之间构建格式化数据和传输数据等新特性. 对于一次完整页面请求,其包含了对于页面主 HTML 代码及相关资源文件的全部请求过程,HTTP/2 协议中使用同一个 TCP 连接完成全部请求,为 1 个 HTTP/2 会话(session). 对于 HTTP 1.1 协议中使用不同 TCP 连接进行的页面内容请求,即每 1 个页面资源的 HTTP 请求,为 1 个 HTTP/2 流(stream). 在 1 个流中,因传输内容不同、拥塞控制及传输效率等原因,使得该请求的数据内容不能一次全部发送,故将其分为若干帧(frame)进行传输. HTTP/2 标准中共定义了 10 种帧格式,包括 HEADERS,DATA,GOAWAY 等.

在 1 个 HTTP/2 流中,首先客户端向服务器发送 HEADERS 帧,包含 HTTP 1.1 协议中 GET 请求协议头字段内容;服务器接收请求后将依次返回给客户端包含响应内容的 HEADERS 帧和包含请求资源数据的 DATA 帧. 在 1 个 HTTP/2 会话中,可同时存在多个流,其中,属于不同流的数据帧可以互相交错发送,但同属于 1 个流的数据帧之间顺序不能修改,如图 1 所示. 当会话正确完成时,客户端向服务器发送 GOAWAY 帧,结束会话.

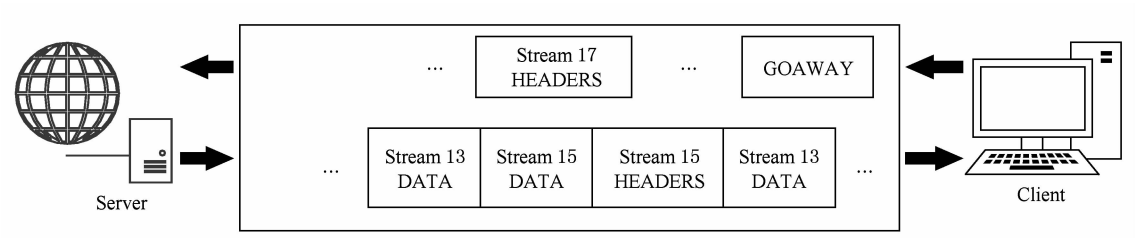

Fig. 1 The process of normal HTTP/2 session

图 1 正常 HTTP/2 会话过程

在 HTTP/2 协议标准规定及具体实现中,除客户端请求发送顺序外,存在多种因素能够影响共享资源数据帧的排列组合.因页面设计带来的页面元素显示的先后顺序依赖关系将使得 HTTP/2 流具有优先级(priority)关系,具有高优先级的流将优先进行处理发送,无优先级关系的流之间互不影响.HTTP/2 协议中拥塞控制也会影响数据帧的发送顺序,当接收端可用滑动窗口大小不足时,选择流中部分数据组成帧并发送,或发送其他流以保证传输速率.其他还有包括服务器推送(server push)、动态页面交互、服务器处理速度等因素亦能影响数据帧序列中流编号的排列.本文即利用不同流的数据帧之间的顺序交替规律,以其流编号构成序列,进行编码实现隐蔽通信.

2 H2CSC 方法原理

本节主要介绍 H2CSC 的设计目标和威胁模型,并详细描述 H2CSC 的方法原理.本文的出发点是建立一种可靠的隐蔽通信模型,能够在保证隐蔽通信数据的隐蔽性和信道稳定性的基础上,提供较高的传输速率.

隐蔽信道需要选择合适的共享资源,并在此基础上寻找提供高数据传输速率及数据准确性的编码方以满足隐蔽性原则、安全性原则和可靠性原则设计目标.其中隐蔽性原则指网络监管设备不能准确判断或区分目标是否有意发布或接收特殊数据;安全性原则指通信内容对于任何的第三方均是不可见的;可靠性原则指在通信过程中该信道需保证传输数据准确.

本文假设所对抗的网络流量监管设备,可能会对全部的网络流量进行协议分析及内容审计,尤其是从监控范围内发出的网络连接.其主要可以分为 2 类:主动监管设备和被动监管设备.被动监管设备能够检查全部网络数据流量,并使用不同方法检测隐蔽信道的存在,或存储数据以供后续协议分析方法进行分析.而主动监管设备除具备上述能力外,还可能尝试篡改数据帧中非必要的内容,如修改 HTTP 协议头中部分域;或调整其通信属性,如延迟或人为丢包等.

通过对已有方法的研究及设计目标的考虑,本文选择 HTTP/2 协议作为公开载体信道,以其中 HTTP 会话数据帧中流编号信息组成序列作为共享资源,使用相应序列编码方法对隐蔽消息进行编码.H2CSC 隐蔽信道工作在如图 2 所示的网络环境中,是以下行隐蔽数据传输为主的基于 HTTP/2 通信行为的隐蔽信道.

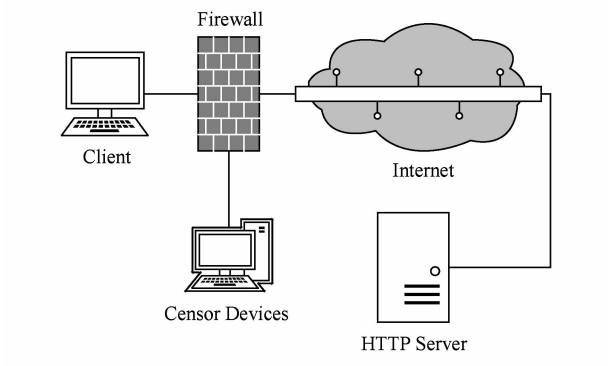

Fig. 2 H2CSC working network environment

图 2 H2CSC 工作网络环境

隐蔽消息接收者可以是 1 个正常的 HTTP 客户端程序,仅需保证其能够正确获取响应的序列信息,这样可以在接收 HTTP 页面响应的同时对隐蔽消息进行解码.消息发送者控制 HTTP 服务器,并将隐蔽数据部署在多个不同的网页中,此过程不受网络监管者监控.受控 HTTP 服务器为开启 NGHTTP 模块、TLS 模块支持的 Apache HTTP 服务器,其源码经过修改以满足操控接收 HTTP 请求或发送 HTTP 响应的目的.对于正常的页面访问请求,其能够正常传输未添加隐蔽消息的响应信息;对于隐蔽消息接收者的页面访问请求,其能够将该页面对应的隐蔽消息加载在响应的过程中,以实现隐蔽通信.

对于静态页面而言,在 HTTP/2 会话建立时,客户端将首先请求主页面代码,一般在主页面全部获取完成之后,其才会继续请求其他页面资源文件.H2CSC 将操纵全部资源文件 HTTP/2 流响应的数据帧发送顺序,通过 3.1 节编码方法,将原待发送队列中数据帧进行分组并重新排序,随后发送至客户端.最终客户端通过分析接收到的数据帧序列以进行解码.排序时并不打乱同属于 1 个流的数据帧的先后顺序,仅在属于不同流的数据帧之间调换顺序.

在隐蔽信道构建过程中,涉及隐蔽数据、网络环境等若干假设,本文在此处一并给出:

假设 1. 假设对于 HTTP/2 会话中的全部流,其数据帧均只有 1 帧,即令页面中每个资源文件大小均小于单数据帧默认最大限制 16 384 B,且在编解码过程中舍弃后续数据帧,此假设是为简化通信模型;

假设 2. 假设 HTTP/2 会话中,除主页面流外的全部流均做隐蔽消息编码,此假设是为最大程度提高单个 HTTP/2 会话隐蔽消息编码量;

假设 3. 假设任意访问该隐蔽消息载体页面的 HTTP/2 请求均为消息接收者请求,并进行处理,此假设是考虑对于任意接收者即使接收页面内容均一致,但除接收者外第三方均无法正确解码隐蔽数据;

假设 4. 假设网络流量监管设备能够通过某种手段获取 HTTP/2 协议传输数据内容,此假设是考虑存在 HTTP 客户端提供 h2c(HTTP/2 cleartext)服务,即运行于非加密通道之上的 HTTP/2 协议服务.

3 编码机制

本节将详细介绍 H2CSC 所采用的编码机制,以及如何根据隐蔽消息设计页面并选取合适参数以建立该信道.

3.1 信息编码方法

隐蔽通信编解码与一般通信协议编解码过程类似. 信源的原始数据首先经过压缩加密处理得到待编码数据 U,然后该二进制序列经隐蔽信道编码器编码,转换为离散序列 V 发送至隐蔽消息发送端;接收端访问获取页面,同时获取该离散序列,记为 V',使用同样方法逆向过程进行解码得到二进制序列 U',最终经过解密解压缩得到原始数据. 由于本文方法底层由 TCP 协议提供数据可靠性,即在传输过程中序列顺序不会发生改变,故 $V'=V$,仅需保证解码方法能够无错解码使得 $U'=U$ 即可.

依据所选择共享资源的特点,将待编码的二进制数据 U 分为若干组,每组记为 U_i,其所包含的位数记为 $|U_i|$. 在正常 HTTP/2 请求响应过程中,得到其正常响应待发送数据帧序列,本编码方法将对此序列做再排序操作. 为保证数据帧处理响应位置不发生大的改动,将此正常序列切割为若干组,每组共包含 n 个数据帧,分属 m 个 HTTP/2 流,每个流在当前位置存在有 m_j 个数据帧待发送,记该序列为 V_i,则 V_i 组中不同数据帧顺序共存在 $|V_i|$ 种互不相同情况,其计算为

$$|U_i|_{\max}=\lfloor \mathrm{lb}\,|V_i| \rfloor. \tag{1}$$

由实际情况可得出条件 $m\leqslant n$ 且 $m_j\geqslant 1$. 故此编码问题可简化为 n 个有序位置、m 种不同小球,每种分别有 m_j 个小球的组合数学问题. 依据本文假设 1,即 $m_j=1$;若满足每组包含 n 个数据帧,共需 n 个流,即 $m=n$. 则最大可编码数据帧排列情况数为

$$|V_i|=\prod_{i=1}^{n} i. \tag{2}$$

此情况下其编码过程即为康托展开式(Cantor expansion)的逆向过程. 康托展开是一种特殊的散列函数,其是对 n 个数的排列进行状态的压缩和存储. 将数据序列 U_i 以先输出位为高位转为十进制数 u_i,通过 u_i 计算逆向康托展开结果序列,依据该序列为 n 个数据帧依据流序号大小进行排序,完成编码. 康托展开公式为

$$X=a[n]\times(n-1)!+a[n-1]\times(n-2)!+\cdots+a[i]\times(i-1)!+\cdots+a[1]\times 0!, \tag{3}$$

其中,$a[i]$为整数,表示当前元素在所有未出现的元素中排在第 i 个,并满足 $0\leqslant a[i]<i$,其中 $1\leqslant i\leqslant n$.

为提高数据传输量,可在编码过程中,将在 U_i 基础上额外添加 1 位,判断添加此位后的结果是否溢出,若未溢出,则添加此位进行编码,若溢出则此次仅编码 $|U_i|$ 位,以此扩充序列编码范围至 $|V_i|$,提高数据传输率.

由于 TCP 协议保证数据准确性,即 HTTP/2 帧经服务器发出后顺序不再改变,而且编解码过程中亦无随机变量引入,故本方法能实现无错译码.

3.2 参数选取及编码

H2CSC 隐蔽信道对每个 HTTP/2 会话均独立看待,即每 1 个新的 HTTP/2 会话建立,其所传输的隐蔽消息均与其他会话无关. 对于 1 次 HTTP/2 会话不能完全加载的隐蔽消息,需要额外数据分片重组处理机制,不在本文讨论范围内.

当给定隐蔽消息数据大小后,可依据其数据总量及组内帧数 n 等参数确定页面设计所需资源文件总数. 如给定隐蔽消息总大小为 1 KB,且选取 $n=4$ 时,每组最大可传输 $|U_i|=\lfloor \mathrm{lb}\,24 \rfloor=4$ b,则共需要 8 192 个流,即 8 192 个页面资源文件;当选取 $n=10$ 时,每组最大可传输 $|U_i|=\lfloor \mathrm{lb}(10!) \rfloor=21$ b,则此时共需要 3 901 个流,即 3 901 个页面资源文件. 本方法支持自定义 $|U_i|$ 取值.

RFC 7540 中规定 HTTP/2 会话最多包含 2^{31} 个流,且由客户端建立的流编号必须使用奇数,即最多能够建立 2^{30} 个流. 当 $n=4$ 且每个流有且仅有 1 个数据帧时,每组最大可传输 $|U_i|=4$ b,故理论上 1 个 HTTP/2 会话最多可传输约 128 MB 数据.

本节将进一步举例描述二进制数据编码至数据帧流编号序列过程. 假设隐蔽消息为“Hello World”时;且主页面代码在第 13 号流中完成传输,共包含 120 个资源文件;选取参数 $n=4$. 表 1 为此时 V_i 输出序列与 U_i 序列的对应关系示例.

Table 1 Encoding 4-bit Data into Sequence

表 1 4-bit 数据序列编码

U_i	V_i	U_i	V_i
0000	1,2,3,4	1000	2,3,1,4
0001	1,2,4,3	1001	2,3,4,1
0010	1,3,2,4	1010	2,4,1,3
0011	1,3,4,2	1011	2,4,3,1
0100	1,4,2,3	1100	3,1,2,4
0101	1,4,3,2	1101	3,1,4,2
0110	2,1,3,4	1110	3,2,1,4
0111	2,1,4,3	1111	3,2,4,1

取隐蔽消息第 1 个字符‘H’,其 ASCII 码二进制高 4 位为 0100,则对应 V_i 为 1,4,2,3. 若后续 4 个数据帧的流编号原发送顺序为 15,17,19,21,对其重新排序后的数据结果即为 15,21,17,19. 同理,‘H’的二进制低 4 位为 1000,对应 V_i 为 2,3,1,4,若此时下一组数据帧流编号按顺序依次是 23,27,25,29,则舍弃原顺序并依据大小关系重新排序得到输出的数据帧流编号序列为 25,27,23,29. 随后,依据该顺序执行数据帧发送,后续依次读取隐蔽消息数据并编码序列,直至全部数据发送完毕.

客户端解码过程与之相反,首先分组并读取到数据帧流编号序列为 15,21,17,19,依据表得到原二进制数据为 0100,随后再计算下一组,最终将全部解码数据合并,即可得到加载的隐蔽消息.

实现 H2CSC 方法的 Web 服务器工作流程如图 3 所示. 其中正常 Web 服务器将对客户端请求依次进行处理,并依次发送处理完成的数据帧,如图 3 中左侧实线所示. 服务器实现 H2CSC 方法流程如右侧虚线所示,依据页面读取待传输的一段隐蔽消息后,依据 3.1 节方法获得顺序序列 V_i,并对待发送的数据帧序列重新进行排序,并最终发送回客户端.

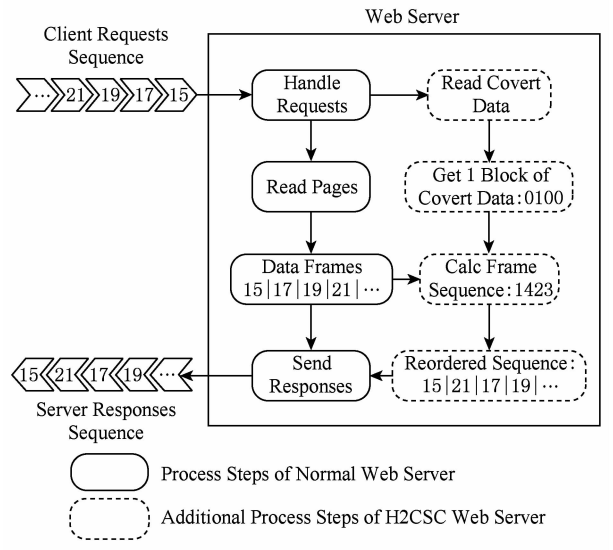

Fig. 3 Web server internal flow chart

图 3 Web 服务器内部工作流程图

4 修正条件熵检测方法

隐蔽信道的分析对抗工作包括 3 类,分别是信道检测、信道限制和信道消除.

本文认为网络监管设备处理 H2CSC 方法时会选择中断该 HTTP/2 协议连接,或重定向使用 HTTP 1.1 协议,此时会导致 H2CSC 方法无效. 因此本文着重关注 H2CSC 隐蔽信道的信道检测工作.

客户端请求顺序一般依据页面解析得到的资源 URI 顺序和优先级依次请求获取. 本方法中隐蔽消息载体页面为静态页面,由于页面设计存在优先级关系,即每行图片资源第 1 个图片优先级最低,最后进行处理传输;其他图片资源优先级相同,依次进行请求处理. 另本方法在服务器设置中不引入包括服务器推送等功能,故最终所能影响数据帧序列的因素包括服务器 HTTP/2 处理速度、拥塞控制及优先级关系等.

由于最终数据帧传输序列存在规律性和非规律性因素,故本文选择基于熵值的检测方法进行检测. 以获取的相邻 2 个数据帧流 ID 之差作为输入数据,以熵值分析该差值的变化规律,用以表明数据帧序列的变化规律,其取值应均为正负整数. 由于熵率是一个极限概念,是无穷序列的条件熵,表示无穷序列的不确定性,利用有限采样数据进行估计无法真实反映数据帧序列的随机性质,故本文最终选择修正条件熵(corrected conditional entropy, CCE)用来解决有限数据采样的问题,其计算公式为

$$CCE(X_j|X_{j-1})=CE(X_j|X_{j-1})+perc(X_j)\times EN(X_1), \quad (4)$$

其中,$j\leqslant n$;$CE(X_j|X_1,X_2,\cdots,X_{j-1})$为经验概率密度条件熵;$perc(X_j)$为采样中长度为 j 的序列只出现 1 次的比例;$EN(X_1)$是子序列长度 $j=1$ 时的熵,即一阶熵. 实际计算时,需将获取 1 次 HTTP/2 会话中全部数据帧序列流 ID 差值,并进行数据正规化后作为 CCE 计算输入,计算得到 j 的所有取值对应的修正条件熵后,取最小值作为其隐蔽性评分,如

式(5)所示:

$$CISC=\overline{ER}=\min_{i=1,2,\cdots,j}(CCE(X_i|X_{i-1})). \quad (5)$$

计算得到 CCE 熵值后,需要对加载隐蔽消息页面与正常页面进行分类识别.由于计算得到的 CCE 熵值为一维数据,故选择使用逻辑回归(logistic regression, LR)分类器对其进行分类,通过对训练集学习以得到参数的合适取值.

5 实验与结果

如第 2 节所述,本方法选择客户端为支持 h2c 方式 HTTP/2 协议客户端程序,支持 HTTP/2 帧级信息输出;选择服务器为支持 h2c 方式 HTTP/2 协议服务器程序,支持 HTTP/2 帧级可控或提供源码以实现定制化修改.本文隐蔽信道是建立在可靠网络通信协议基础上,故实验以测试该信道性能、数据传输速率和信道隐蔽性为主.

5.1 实现细节

隐蔽消息接收者在接收隐蔽消息时,行为与访问正常 Web 页面完全一致;在结果处理中,不需要进行页面展示,而是将获取数据帧的流编号组成序列,分组解码即可.故选择 nghttp2-client 程序作为客户端程序,通过 python 脚本对结果输出进行解码处理.

隐蔽消息服务器由于较少存在配置粒度达到可直接操控 HTTP/2 帧处理的服务器程序,选择采用 nghttp2 库和 Apache Httpd 服务器程序以修改源码方式实现.由于本文假设 1 全部 HTTP/2 流均有且仅有 1 个数据帧,故本文修改程序使得服务器在 HTTP/2 会话处理过程中,另额外建立流发送队列,独立于原流处理优先级队列,当流发送队列中流发送条件不满足时,取出原队列流执行发送处理.在发送数据帧时,确保流发送队列中出队流第 1 个数据帧发送后,即将其出队;解码时仅取该流的第 1 个数据帧所在位置进行解码处理,忽略后续数据帧.

在隐蔽信道载体页面设计中,需要考虑元素显示的先后顺序而导致的页面资源文件获取优先级问题.服务器页面设置为以表格控件为主,以若干图片切片资源作为表格每 1 项,以此组成服务器隐蔽消息载体静态页面.实验中共实现 8 个页面,由均分图片矩形切片构成,全部页面资源为以表格形式可组成具有编号的同一大图片页面.其页面 URI、总大小及该页面加载隐蔽消息参数如表 2 所示:

Table 2 Information of 8 Covert Pages

表 2 8 个隐蔽页面信息

URI	Page Size/MB	Resource Amount	n	$\lvert U_i\rvert$
1.html	1.07	120	4	4
2.html	0.98	240	5	6
3.html	1.04	480	6	9
4.html	1.05	960	7	12
5.html	1.19	1920	8	15
6.html	1.49	3840	9	18
7.html	2.89	7680	10	21
8.html	4.35	10000	10	21

5.2 服务器性能影响

本文使用的 1 核、1 GB 内存、1 Mbps 带宽的云主机作为 HTTP/2 服务器,其物理主机位于中国广东省广州市.客户端所在主机位于中国北京市海淀区,其与服务器平均 RTT 约为 44 ms.

对每个页面分别计算加载隐蔽消息和未加载隐蔽消息情况下访问 100 次的平均耗时,所加载的隐蔽消息为 ASCII 码英文字母及数字字符串,每次访问间隔时间为 1 s.所得平均访问时间如图 4 所示.对于同一个页面,其中加载隐蔽信道与普通页面访问时间平均相差均不超过 10 ms,该时间间隔相比整体页面访问时间约占 1%或更少,可以忽略不计.结果表明 H2CSC 方法对于 Apache HTTPD 服务器的性能影响较小.另外 1~8 号页面之间访问时间相差的主要原因在于其页面总体大小的逐渐递增.

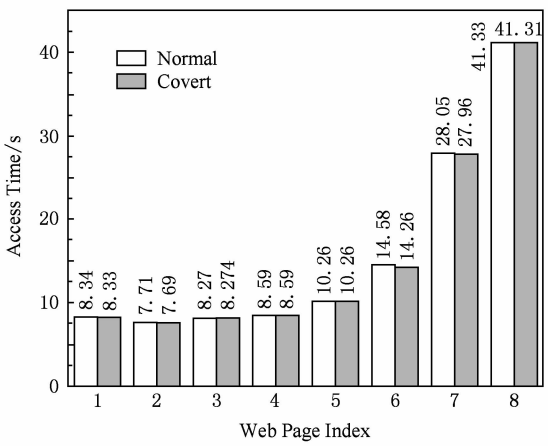

Fig. 4 Page access time

图 4 页面访问时间

5.3 数据传输速率

实验所用 8 个页面分别加载长度内容均不同的隐蔽消息以进行数据传输速率测试.由于本文选择使用 Adobe Photoshop 软件对原始图片进行切片,

该软件最大支持 10 000 个图片切片,且由于图片效果、切片数量等原因使得各页面总大小不同且较难减小.作为目标传输页面,其所加载隐蔽消息均使用全部页面资源进行编码,实验中测量每个页面访问 100 次的平均用时,以此计算隐蔽数据传输速率.

访问各页面对应隐蔽信道速率如图 5 所示.当服务器带宽确定时,此隐蔽信道数据传输速率主要和 HTTP/2 编码相关页面资源文件大小和编码参数选取相关.比较页面 1～5,其页面总大小相差较小.随分块帧数 n 取值的增大,可传输数据量大致呈指数增长趋势.页面 6～8 比较可以看出,隐蔽数据传输速率受页面整体传输时间影响较大,可以通过提高服务器带宽或降低页面整体大小以提高传输速率.在确定编码方式 n 和 $|U_i|$ 取值情况下,其与 HTTP/2 服务器带宽成正比,与页面总大小成反比.另外,随页面资源文件个数的增加,HTTP/2 协议中其他类型帧等额外传输数据也将增加,这些额外开销将导致隐蔽信道数据传输速率略有下降.

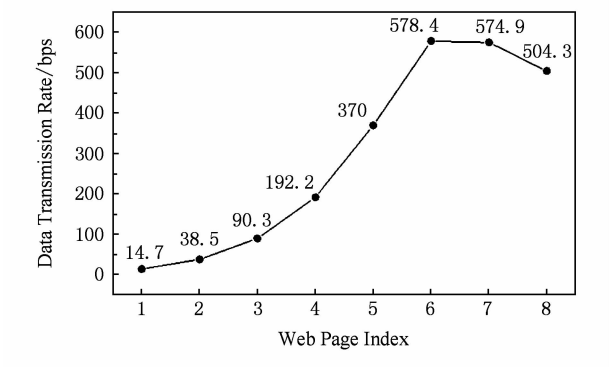

Fig. 5 Covert data transmission rate of each page

图 5 各页面隐蔽数据传输速率

本文选择目前具有较高的传输速率和安全性的基于时间和基于通信行为以组合数学思想实现编码的 JIBC[15],Cloak[20],LiHB[21] 方法和本文 H2CSC 进行速率比较,如表 3 所示.基于多 HTTP 请求的 LiHB 在实验中达到约 200 bps,但该方法传输速率对于页面对象取值增长缓慢.基于多 TCP 流排列组合的 Cloak 隐蔽信道实验中最多达到 450 bps,其在 20 个 TCP 流中排列组合 40 个数据包,且需要数据包具有先后顺序,该方法占用网络资源较多且实际使用环境较小.

Table 3 Transmission Rates of Covert Channels

表 3 隐蔽信道数据传输速率

Method	Experimental Rate/bps
H2CSC	578
Cloak	450
JIBC	295
LiHB	201

而本文的基于 HTTP/2 协议数据帧序列的隐蔽信道能够达到 578 bps 的隐蔽数据传输速率,当信道带宽提高或页面大小降低后仍能获得极大提高,本方法传输速率具有较高的提升空间.其计算为

$$V_{\mathrm{H2CSC}}=\frac{N_i\times|U_i|\times Bandwidth}{n\times PageSize_i},\tag{6}$$

其中,N_i 表示第 i 个页面的资源文件总数,$PageSize_i$ 表示第 i 个页面的页面总大小,$Bandwidth$ 表示服务器带宽.

5.4 基于修正条件熵的 LR 检测方法

此项测试以页面 4 为基础页面,总计测试 5 组.其中,第 1 组 4-Normal 不加载任何隐蔽消息;第 2 组 4-ASCII 加载以 ASCII 码编码的英文数字隐蔽消息;第 3 组 4-Random 加载随机二进制隐蔽消息;第 4 组 4-UTF8-CHS 加载中文 UTF-8 编码的隐蔽消息;第 5 组为网络中支持 HTTP/2 协议的正常页面.

从图 6 中可以得出,对于基础页面,未加载隐蔽消息时,CCE 熵值较低基本保持在 0.6 以下,表明其数据帧流编号序列基本保持不变,主要原因在于其页面资源文件大小均较小,不会因拥塞控制机制调整顺序,故大部分数据帧流编号依据客户端请求顺序依次递增.另外页面资源布局导致的优先级结构也相对简单,不能显著影响数据帧响应发送顺序.

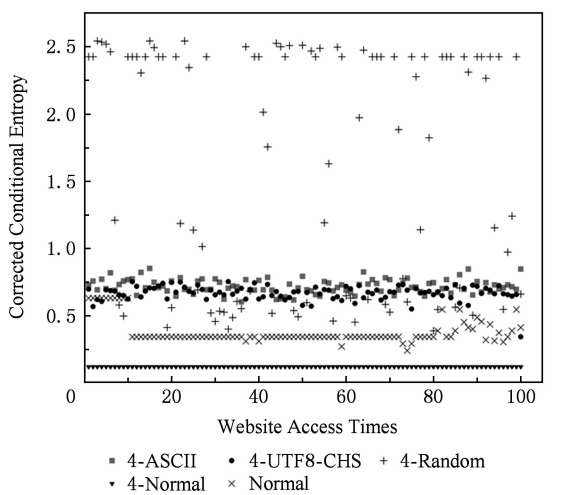

Fig. 6 CCE of different pages

图 6 不同页面修正条件熵

而对于加载 ASCII 编码和 UTF-8 编码的隐蔽消息而言,其 CCE 值变化相似,表明 2 种编码分块后数据分布基本类似;而对于加载随机二进制数据的 CCE 值变化即相对随机,变化范围是 0.4～2.5.

从图 6 中可以较清晰地观测到,H2CSC 方法加载消息页面与正常页面提取的 CCE 熵值具有一定区别,可以使用修正条件熵作为特征提取方法以进行识别区分.

本文考虑对于网络监管设备,实时监控目标访问网页并预先进行熵值计算方式并不可取,其可选择预先训练检测模型,随后直接计算目标访问页面熵值并直接带入检测分类程序得出结果.故实验中选择页面资源文件较多、网站访问量较大的 100 个正常页面,计算访问过程的 CCE 熵值,将其均匀分为训练集和测试集,并保证其页面资源文件分布情况及总和大致相当,用于学习训练及后续测试.并以同样方式处理访问以页面 1 为基础平均加载文本和随机数据的 100 个页面所得到的 CCE 熵值.每个页面以访问 10 次的平均 CCE 熵值作为该页面特征.

本文选择使用 Python scikit-learn 内建逻辑回归分类器进行实现.使用该逻辑回归函数对测试集进行分类检测所得准确率结果矩阵如图 7 所示.其中识别正常会话准确率为 90%,识别 H2CSC 隐蔽信道会话准确率为 91%.

	(Result) Overt	(Result) Covert
(True) Overt	90%	10%
(True) Covert	9%	91%

Fig. 7 Detection method accuracy

图 7 检测方法准确率

5.5 CCE 熵值优化处理方法

基于 5.4 节的检测结果,考虑可行的对抗检测方法为降低每组传输位 $|U_i|$ 取值,以缩小每组修改后数据帧序列与正常会话序列编辑距离,从而降低熵值.为测量比较多种取值情况下熵值的变化规律,选择页面 3 为目标页面,加载同一文本隐蔽消息,不同 n 和 $|U_i|$ 取值及相应的熵值如表 4 所示:

Table 4 Optimize Parameters for CCE

表 4 CCE 熵值参数优化

n	$\lvert U_i\rvert$	CCE	Covert Data/b
		0.2127	0
5	1	0.3845	96
5	2	0.5235	192
5	3	0.7317	384
5	4	0.8622	768
5	5	1.4673	1536
4	1	0.5057	120
6	1	0.3173	80

由表 4 可知,当 n 取值确定时,$|U_i|$ 取值越小,则熵值越小;$|U_i|$ 取值确定时,n 取值越大,则熵值越小;其余部分给出部分熵值在阈值范围内或接近阈值边界的 n 与 $|U_i|$ 取值.由于加载的隐蔽消息不同会使得页面熵值不同,实际使用时可依据需要加载的隐蔽消息数据量大小及对应页面熵值以调整 n 与 $|U_i|$ 的取值.以选取 $n=5$,$|U_i|=1$ 为例,即每 5 个数据帧传输 1 b 数据,会话以最大容量编码隐蔽数据,其熵值约为 0.384 5.此情况下使用上述基于修正条件熵的检测方法进行识别,得到其被识别为正常通信行为,故针对对抗性参数取值 $n=5$ 且 $|U_i|=1$ 时,基于熵值的检测方法将存在极大误差,其也表明使用降低 $|U_i|$ 取值可以有效降低信道被识别率.相应数据传输速率可通过 5.1 节中数据及参数取值和调整后参数取值相比对进行数学计算.实际使用时亦可依据数据传输速率要求及安全性要求选取合适的参数.

6 总 结

本文提出一种基于 HTTP/2 协议的隐蔽序列通信方法,该方法利用 HTTP/2 协议数据帧中的流编号组成序列,通过该序列排列组合信息建立一个隐蔽通信协议,使隐蔽消息发送者和接收者能够秘密地交换信息,从而对抗网络安全设备.本文提取 H2CSC 方法和正常网络通信数据的同一特征指标,使用基于修正条件熵的逻辑回归分类方法进行识别,实验证明该方法具有较好的隐蔽性,且基于 HTTP/2 协议特性,具有较高的可靠性和安全性.H2CSC 方法数据传输速率能够达到一般隐蔽存储类信道水平,且易于通过增加服务器配置及页面优化等方法进一步提高.本文未来预计进一步研究该方法在上行隐蔽通信过程中的应用,并将使用更加全面的如 Regularity 测试、Kolmogorov-Smirnov 测试等进行提取特征分析,并研究能够更加准确分析 H2CSC 与正常通信差异的特征提取方法和分类识别方法.

参 考 文 献

[1] Lampson B W. A note on the confinement problem [J]. Communications of the ACM, 1973, 16(10): 613-615

[2] Gligor V D. A Guide to Understanding Covert Channel Analysis of Trusted Systems [M]. Fort Meade, Maryland: National Computer Security Center, 1994

[3] Tan Qingfeng, Fang Binxing, Shi Jinqiao, et al. StegoP2P: A hidden communication approach in P2P networks [J]. Journal of Computer Research and Development, 2014, 51(8): 1695-1703 (in Chinese)
(谭庆丰, 方滨兴, 时金桥, 等. StegoP2P: 一种基于 P2P 网络的隐蔽通信方法[J]. 计算机研究与发展, 2014, 51(8): 1695-1703)

[4] Brown E, Yuan B, Johnson D, et al. Covert channels in the HTTP network protocol: Channel characterization and detecting Man-in-the-Middle attacks [C] //Proc of the 5th Int Conf on Cyber Warfare and Security. South Oxfordshire, England: Academic Conferences International Limited, 2010: 56-64

[5] Heilman S, Williams J, Johnson D. Covert channel in HTTP User-Agents [C] //Proc of the 11th Annual Symp on Information Assurance. Albany, NY: GCCIS, 2016: 68-73

[6] Schwenk G, Rieck K. Adaptive detection of covert communication in HTTP requests [C] //Proc of the 7th European Conf on Computer Network Defense. Piscataway, NJ: IEEE, 2011: 25-32

[7] Tomar N, Gaur M S. Information theft through covert channel by exploiting HTTP post method [C] //Proc of the 10th Int Conf on Wireless and Optical Communications Networks. Piscataway, NJ: IEEE, 2013: 1-5

[8] Wang Yongji, Wu Jingzheng, Zeng Haitao, et al. Covert channel research [J]. Journal of Software, 2010, 21(9): 2262-2288 (in Chinese)
(王永吉, 吴敬征, 曾海涛, 等. 隐蔽信道研究[J]. 软件学报, 2010, 21(9): 2262-2288)

[9] Aggarwal P K, Jain P, Verma T. Adaptive approach for information hiding in WWW pages [C] //Proc of IEEE ICICT'2014. Piscataway, NJ: IEEE, 2014: 113-118

[10] Kwecka Z. Application layer covert channel analysis and detection [D]. Edinburgh: Edinburgh Napier University, 2006

[11] Cabuk S, Brodley C E, Shields C. IP covert timing channels: Design and detection [C] //Proc of the 11th ACM Conf on Computer and Communications Security. New York: ACM, 2004: 178-187

[12] Luo Xiapu, Chan E W W, Chang R K C. TCP covert timing channels: Design and detection [C] //Proc of the 38th IEEE Int Conf on Dependable Systems and Networks with FTCS and DCC. Piscataway, NJ: IEEE, 2008: 420-429

[13] Berk V, Giani A, Cybenko G, et al. Detection of covert channel encoding in network packet delays, TR2005-536 [R]. Hanover, NH: Department of Computer Science, Dartmouth College, 2005

[14] Qian Yuwen, Zhao Bangxin, Kong Jianshou, et al. Robust covert timing channel based on Web [J]. Journal of Computer Research and Development, 2011, 48(3): 423-431 (in Chinese)
(钱玉文, 赵邦信, 孔建寿, 等. 一种基于 Web 的可靠网络隐蔽时间信道的研究[J]. 计算机研究与发展, 2011, 48(3): 423-431)

[15] Kolegov D N, Broslavskiy O V, Oleksov N E. Covert timing channel over HTTP cache-control headers [J]. Prikladnaya Diskretnaya Matematika, 2015(2): 71-85

[16] Ji Liping, Jiang Wenhao, Dai Benyang, et al. A novel covert channel based on length of messages [C] //Proc of the 1st Int Symp on Information Engineering and Electronic Commerce. Piscataway, NJ: IEEE, 2009: 551-554

[17] Gianvecchio S, Wang Haining. Detecting covert timing channels: An entropy-based approach [C] //Proc of the 14th ACM Conf on Computer and Communications Security. New York: ACM, 2007: 307-316

[18] Tumoian E, Anikeev M. Network based detection of passive covert channels in TCP/IP [C] //Proc of the 30th Anniversary of the IEEE Conf on Local Computer Network. Piscataway, NJ: IEEE, 2005: 802-809

[19] Zander S, Armitage G, Branch P. A survey of covert channels and countermeasures in computer network protocols [J]. IEEE Communications Surveys & Tutorials, 2007, 9(3): 44-57

[20] Luo Xiapu, Chan E W W, Zhou Peng, et al. Robust network covert communications based on TCP and enumerative combinatorics [J]. IEEE Trans on Dependable and Secure Computing, 2012, 9(6): 890-902

[21] Shen Yao, Huang Liusheng, Wang Fei, et al. LiHB: Lost in HTTP Behaviors—A behavior-based covert channel in HTTP [C] //Proc of the 3rd ACM Workshop on Information Hiding and Multimedia Security. New York: ACM, 2015: 55-64

[22] Belshe M, Peon R, Thomson M. RFC 7540: Hypertext Transfer Protocol Version 2 (HTTP/2)[S/OL]. IETF, 2015 [2017-11-15]. https://tools.ietf.org/html/rfc7450

[23] W3techs.com. Usage Statistics of HTTP/2 for Websites [EB/OL]. [2017-11-15]. https://w3techs.com/technologies/details/ce-http2/all/all

Liu Zhengyi, born in 1993. Master. His main research interests include covert communication and network security.

Song Tian, born in 1980. PhD, associate professor in Beijing Institute of Technology. Senior member of CCF. His main research interests include future Internet architecture, network security and computer architecture.