StealthyFlow：一种对抗条件下恶意代码动态流量伪装框架

韩 宇[1] 方滨兴[1),2)] 崔 翔[2)] 王忠儒[1),3)] 冀甜甜[1)] 冯 林[2)] 余伟强[4)]

[1)](北京邮电大学可信分布式计算与服务教育部重点实验室 北京 100876)

[2)](广州大学网络空间先进技术研究院 广州 510006)

[3)](中国网络空间研究院 北京 100010)

[4)](北京丁牛科技有限公司 北京 100081)

摘 要 恶意代码问题使国家安全面临严重威胁.随着 TLS 协议快速普及,恶意代码呈现出流量加密化的趋势,通信内容加密导致检测难度的进一步提高.本文提出一种恶意代码流量伪装框架 StealthyFlow,以采用加密流量进行远控通信的公共资源型恶意代码与 GAN 结合,对恶意流量进行不影响攻击功能的伪装,旨在实现伪装后的对抗流量与良性流量的不可区分性,进而绕过基于机器学习算法的分类器.StealthyFlow 具有如下优势:根据目标流量的变化动态调整对抗流量,实现动态流量伪装;伪装在恶意代码层面进行,保证攻击功能不被破坏;绕过目标不参与训练过程,保证恶意代码不会提前暴露.实验结果表明,StealthyFlow 产生的攻击流量与良性流量相似度极高,在对抗环境中可以绕过机器学习分类器.因此,需要对此种恶意代码提起注意,并尽快研究防御对策.

关键词 恶意代码;加密流量;StealthyFlow;绕过;动态流量伪装

中图法分类号 TP309 **DOI 号** 10.11897/SP.J.1016.2021.00948

StealthyFlow: A Framework for Malware Dynamic Traffic Camouflaging in Adversarial Environment

HAN Yu[1)] FANG Bin-Xing[1),2)] CUI Xiang[2)] WANG Zhong-Ru[1),3)] JI Tian-Tian[1)]
FENG Lin[2)] YU Wei-Qiang[4)]

[1)](*Key Laboratory of Trustworthy Distributed Computing and Service (BUPT), Ministry of Education, Beijing University of Posts and Telecommunications, Beijing* 100876)

[2)](*Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou* 510006)

[3)](*Chinese Academy of Cyberspace Studies, Beijing* 100010)

[4)](*Beijing DigApis Technology Co., Ltd, Beijing* 100081)

Abstract Malware emerges endlessly, which not only causes economic losses to enterprises and individuals, but also poses serious threats to national security. During the Gulf War in 1991, the United States publicly used malware attack technology to obtain major military benefits for the first time. Since then, malware attacks have become one of the most important intrusion methods for information and network warfare. In recent years, malware based on legitimate services has spread. The traffic of this kind of malware is mixed with the traffic of legitimate services and is not easy to be detected. At the same time, the use of TLS poses new challenges to traffic detection

收稿日期:2020-07-14;在线发布日期:2021-01-06.本课题得到广东省重点领域研发计划(2019B010137004,2019B010136003)和国家重点研发计划(2018YFB0803504,2019YFA0706404)资助.**韩 宇**,硕士研究生,主要研究方向为网络安全.E-mail:hany@bupt.edu.cn.**方滨兴**,博士,中国工程院院士,主要研究领域为计算机体系结构、计算机网络、信息安全.**崔 翔**(通信作者),博士,教授,主要研究领域为网络安全.E-mail:cuixiang@gzhu.edu.cn.**王忠儒**(通信作者),博士,高级工程师,主要研究方向为人工智能、网络安全.E-mail:wangzhongru@bupt.edu.cn.**冀甜甜**,博士研究生,主要研究方向为网络安全.**冯 林**,硕士研究生,主要研究方向为网络安全.**余伟强**,硕士,主要研究方向为网络安全、人工智能.

because the content can no longer be analyzed due to encryption. The combination of public resources and encrypted traffic makes "the traffic generated by malware flows to normal websites, and its communication content is based on encrypted protocols and cannot be checked", which further increases the difficulty of detection. In order to ensure the security of network communication, researchers have conducted in-depth explorations on the detection of encrypted traffic. Due to the advantage of discovering unknown attacks, machine learning algorithms have become the mainstream detection method, but there is a risk of failure when malicious traffic and benign traffic are indistinguishable in the features focused by machine learning systems. In order to study the possibility of confronting machine-learning-based traffic detection system, we propose a dynamic traffic camouflaging framework named StealthyFlow. StealthyFlow combines Generative Adversarial Networks with malware that uses legitimate services for backdoor command and control, to realize traffic camouflaging without affecting the attack function. It consists of two modules, GAN module and malicious code module, which are responsible for feature generation and traffic generation respectively. It aimed at realizing the indistinguishability between traffic after disguise and benign traffic, and then bypass classifiers based on machine learning algorithms. StealthyFlow has the following advantages. First, it can dynamically adjust the traffic flow according to the change of the target flow, which means dynamic flow camouflaging. Second, it makes changes at the malware level instead of directly modifying the flow, which can ensure that the attack function is not destroyed. Third, the target being bypassed does not participate in the training process, ensure that malware is not exposed. Experiment results show that the traffic generated by StealthyFlow is very similar to benign traffic, and can bypass the machine-learning-based classifiers in an adversarial environment. The result questions the robustness of the encryption traffic detection method based on machine learning algorithms. Finally, from the perspective of the attacker, the new malware based on StealthyFlow will bring new security threats to the defense work. This not only requires the attention of security researchers, but also requires a lot of effort in the future to establish anti-encrypted-malware defense system as soon as possible.

Keywords　malware; encrypted traffic; StealthyFlow; adversarial; dynamic traffic camouflaging

1　引　言

近年来，随着国家间攻防对抗不断升级，恶意代码在传播感染、命令与控制(Command and Control，C&C 或 C2)及数据泄露阶段均发生了重大变化. 其中 C2 阶段的变化尤为明显，攻击者开始利用社交网站等公共资源构建信道进行通信，目的是提高恶意代码的隐蔽性. 统计 MITRE ATT&CK① 的数据发现，HTTPS 是最常被用做恶意代码隐蔽信道的标准应用层协议之一，仅次于 HTTP，并且增长速度高于后者[1]. 而公共资源型恶意代码大多采用 HTTPS 通信，具有传播范围广泛且难以被防御者控制和关闭的特点，一旦被利用，将对网络环境造成极大威胁，因此本文以公共资源型恶意代码为切入点对采用加密协议通信的恶意代码进行研究.

对于各种公共资源服务，例如：社交网站(Facebook、Twitter 等)、云平台(Dropbox、Mediafire 等)及图床(Upload、Catbox 等)、博客(Wordpress、CSDN 等)、便签(Pastebin 等)、代码托管(Github 等)内容共享网站等，由于可以由用户定义页面内容，它们适合用于发布命令或传递窃取的信息，常被攻击者作为隐蔽信道使用[2]. 自 20 世纪末至今，加密协议普及，上述服务类型已经基本实现通信加密. 公共资源与加密流量的结合，使得"恶意代码产生的流量流向正常网站，其通信内容基于加密协议而无法检查"，进一步增加了检测难度.

在恶意代码的隐蔽性方面，利用公共资源进行 C2 通信的方式使恶意代码实现了很好的隐蔽，具体原因为：(1) 受害主机在被入侵之前极有可能建立

① Mitre ATT&CK. https://attack.mitre.org

过与相应公共资源的通信,通往公共资源的流量几乎不被怀疑,攻击流量很容易隐藏在正常流量中;(2)公共资源的存在使得攻击者便于在公共可访问的互联网上发布和接收信息,为构建隐蔽信道提供了基础设施;(3)即使防御者具有发现异常的能力,但采取屏蔽对应的 IP 或域名的措施意味着无法使用该合法服务,是一种不合理的做法.另外,公共资源通常支持 SSL/TLS 加密,也为攻击者提供了额外的保护①.可以预见,以公共资源做 C2 服务器是未来恶意代码远控通信的发展趋势.

与此同时,加密流量检测的研究也迅速发展.检测方法可归纳为基于规则的检测和启发式检测两种手段.基于规则的检测方法本质上都是解析识别方法,需要根据设定好的规则来识别流量,不具有智能性.启发式检测方法不依赖流量的局部特征解析,根据宏观特性对流量的统计行为特征进行识别,通过分类技术实现流量分类,更加智能化,可用于识别未知恶意代码.机器学习是启发式检测的常用算法.

机器学习检测固然有效,但并非不可绕过.本文认为,基于机器学习的加密流量检测算法,本质是通过流量的特征差异对良性流量和恶意流量加以区分,因此只需向恶意流量中添加不影响攻击功能的微小扰动即可绕过.以 Liu 等人[3]于 2019 年提出的加密流量检测算法为例,该算法从良性和恶意两种加密流中提取包特征、TLS 协议特征和证书特征,作为正负样本训练基于在线随机森林算法的分类器,利用训练好的分类器对加密流量进行检测.针对该检测方法,若将攻击流量的上述三类特征的统计特性进行调整,例如修改数据包到达时间、增加支持的 TLS 扩展、调整自签名证书为 CA 证书等,就可以使该检测算法失效.这种方法称为流量伪装技术,其目的是改变流量的形态以达到预期效果.

在攻击场景中,流量伪装技术将一种流量的特征伪装成另一种流量,降低基于流统计特征方法的识别准确率[4].如果不借助人工智能技术,通过设计算法同样可以进行流量伪装以绕过检测,但该方法有三个缺陷:其一是由于部分用户实体的行为习惯可能严重偏离主流行为习惯,这样生成的流量就无法做到自适应用户特性;其二由于伪装后的流量(称为对抗流量)与良性流量仍然具有差异性,因此仍有被检测到的可能;其三是针对一种场景设计的伪装算法不能适用于另一种场景,迁移性较差.

随着人工智能的快速发展,许多类型的生成对抗网络(Generative Adversarial Networks,GAN)结构分别被用于解决不同领域的问题.生成器和鉴别器相结合的结构使得 GAN 具有自我反馈的功能,在数据生成方面具有极大优势.将 GAN 应用于流量伪装,以良性流量为学习对象生成对抗流量,将对基于机器学习的检测算法提出极大挑战.

本文提出一种流量伪装框架,称为 StealthyFlow,将 GAN 与传统的公共资源型恶意代码(以下简称 OrigMalware)进行结合,可以调整 OrigMalware 的通信模式,实现与良性流量的不可区分性.基于 StealthyFlow 框架的新型恶意代码(以下简称 SFMalware)具有如下特点:以公共资源作为 C2 服务器,实现通信行为的合理性;采用加密流量进行通信,保证通信内容不被解析、攻击意图不被发现;动态模仿良性流量对恶意流量进行伪装,实现对抗流量与良性流量的不可区分性.

机器学习检测算法以统计特征分布的差异性作为正负样本的区分标准,针对这一原理,设计使对抗流量与良性流量的统计特性趋于一致,那么在控制误报率的前提下,将无法区分这两种流量,因此可以绕过机器学习分类器的检测成功实现通信.为准确对 StealthyFlow 进行描述,给出以下定义:

定义 1. 公共资源型恶意代码.凡是利用互联网公共资源构建 C2 信道的恶意代码,都称为公共资源型恶意代码.其 C2 模型如图 1(a)所示,运行在被控主机的恶意代码与公共资源为载体的 C2 服务器通信实现命令与控制.公共资源型恶意代码是 StealthyFlow 的恶意代码部分的原型.

定义 2. 对抗环境.在被控主机侧的网络边界可能存在 IDS(Intrusion Detection System,入侵检测系统)/IPS(Intrusion Prevention System,入侵防御系统)、NTA(Network Traffic Analysis,网络流量分析)、UEBA(User and Entity Behavior Analytics,用户及实体行为分析)等对抗恶意流量的防御措施,这种被控主机环境称为对抗环境.对抗环境中设置的基于机器学习算法的防御工具(简称为机器学习分类器),是 StealthyFlow 的绕过对象.

定义 3. 动态流量伪装.流量伪装代表根据目标流量生成与之相似的对抗流量,动态流量伪装则代表在目标流量不断更新的情况下,生成的对抗流

① Steve M, Peyton S. Rise of legitimate services for backdoor command and control. https://www.anomali.com/resources/anomali-labs-reports/rise-of-legitimate-services-for-backdoor-command-and-control

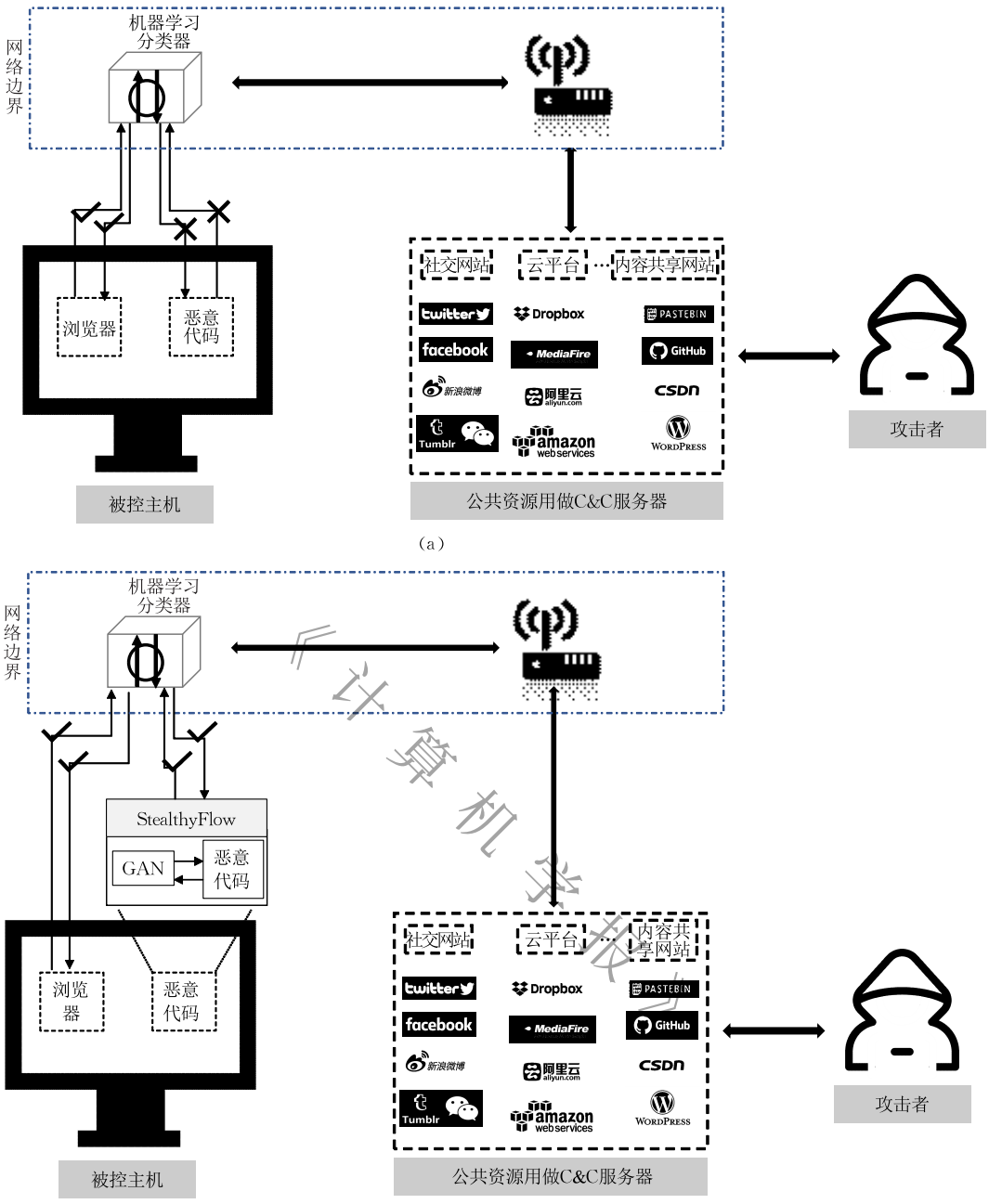

图 1　公共资源型恶意代码远程命令与控制模型

量能够学习到目标流量的变化. 动态流量伪装是本文提出的新型恶意代码的关键技术.

本文的贡献包括以下三个方面：

(1) 提出了一种恶意代码动态流量伪装框架 StealthyFlow，根据目标流量的变化动态调整对抗流量，实现与目标流量的不可区分性；

(2) 以 WGAN-GP 模型和真实恶意代码的结合为例实现了基于 StealthyFlow 框架的 SFMalware，并以真实用户访问 Github 的流量对 SFMalware 进行了训练；

(3) 在对抗环境下，针对四种机器学习分类器进行绕过，验证 StealthyFlow 绕过检测的有效性.

据我们所知，本文是首次提出以加密型恶意代码为研究目标，对其产生的加密攻击流量进行伪装的工作. 为了保证攻击功能不被破坏，本文提出在恶意代码维度实现流量伪装，并通过实验证明攻击功能的完整性.

实验结果表明，基于 StealthyFlow 的动态流量伪装可以实现与目标流量的不可区分性，能够绕过基于机器学习的检测系统. StealthyFlow 的提出使得攻击流量绕过流量监控实施攻击行为成为可能，因此需要尽快提出防御对策.

本文第 2 节对公共资源型恶意代码远程命令与控制模型进行总结;第 3 节对恶意代码框架 StealthyFlow 的设计与实现进行讨论;第 4 节介绍数据集收集和实验评估;第 5 节概述流量模仿相关工作;最后对于这种恶意代码提出防御方案并进行总结和展望.

2 公共资源型恶意代码威胁模型

2.1 模型定义

在传统恶意代码的 C2 过程中,攻击者通常利用自建的一台或多台服务器来实现与受害主机的通信.该模型的不足之处在于,一旦防御者发现 C2 服务器,可采用屏蔽 IP 或域名沉没等方式关闭信道,导致攻击过程被打断.为弥补传统攻击模型的这一缺陷,本文采用的是一种基于公共资源的恶意代码控制信道模型,并基于此模型设计实现了名为 StealthyFlow 的流量伪装框架,以应对传统恶意代码 C2 信道被发现甚至屏蔽或关闭的风险.

公共资源型恶意代码远程命令与控制模型如图 1(a)所示.由图可见,公共资源型恶意代码运行在被控主机上,须先通过网络边界的流量监控设备方可实现与 C2 服务器的通信.恶意代码已经运行在对抗环境前提下,模型中主要包含四种元素,分别是被控主机、网络边界、公共资源服务器(作为恶意代码 C2 服务器)和攻击者.以下针对四个元素分别介绍.

(1)被控主机.被控主机指代被植入恶意代码并且可以访问指定公共资源的主机.

(2)网络边界.在对抗环境下,网络边缘可能存在机器学习分类器等流量检测装置.

(3)公共资源服务器.支撑各种公共资源提供服务的计算机,这里被用作恶意代码的 C2 服务器.

(4)攻击者.攻击者以公共资源服务器作为 C2 信道向恶意代码传递控制命令,指导恶意代码在被控主机上实施攻击行为甚至进行数据回传等操作.

相比于传统恶意代码 C2 模型,公共资源型恶意代码 C2 模型在隐蔽性、健壮性和可扩展性上都表现更好.具体原因是:

(1)隐蔽性.基于公共资源的 C2 信道不同于传统信道,通信对象是互联网可访问的、受信任的服务,从而可将恶意代码产生的攻击流量伪装成正常流量,避免引起网络边缘检测设备告警,因此可有效提高隐蔽性.

(2)健壮性.即使 C2 信道被发现,防御措施也仅限于清除远控信息和关闭用户账号,考虑到公共资源本身的合法性不会关闭整个服务.作为攻击者可以注册大量账户构造冗余信道,因此具有健壮性的特征(参见本文第 3 页脚注①).

(3)可扩展性.基于公共资源命令与控制模型的恶意代码构建信道并不局限于单一类型公共资源,只需添加插件即可适应新的公共资源,有效提升可扩展性.

2.2 信道构建与利用

2.2.1 信道构建与利用流程

如图 2 所示,公共资源型恶意代码信道构建与利用过程可以概括为寻址、命令与控制和结果回传三个阶段.

(1)寻址阶段.旨在利用攻击者预先设定的方式获取控制命令所在的地址或地址列表.公共资源型恶意代码常用 URL 生成算法(URL Generation Algorithm,UGA[5]),结合时间、用户名等元素动态生成 URL,定位控制命令所在位置,建立连接获取命令.

(2)命令与控制阶段.被控主机通过寻址阶段定位控制命令所在资源,对资源发起请求并根据既定规则从中解码得到控制命令并执行.为了及时获取控制者下发的命令,恶意代码需要不断发起请求.

(3)结果回传阶段.在命令与控制阶段恶意代码获取并执行控制命令,命令执行结果中需要回传的部分,一般通过命令与控制阶段给出的回传地址进行上传操作.

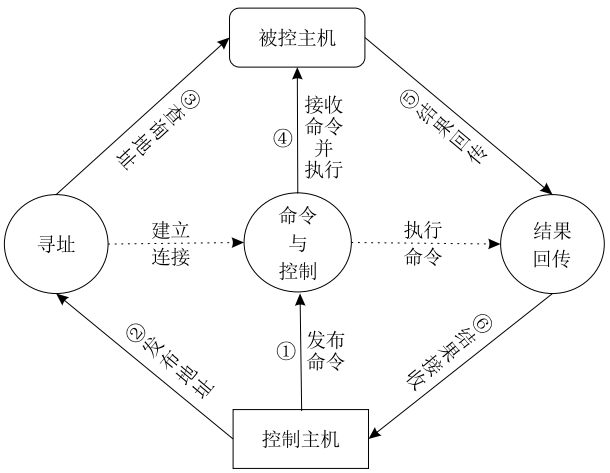

图 2 信道构建与利用

2.2.2 恶意代码实例分析

本文对 APT3 后门 RAINYDROP 和 Github 开源项目 canisrufus 样本进行分析并对攻击机理进

行了总结.

(1) RAINYDROP[①]. 使用 Github 仓库的问题页(Issues)进行命令传递和结果回传，所有信息都采用 base64 编码. 首先攻击者持有 Github 账号并在该账号下创建了用于传递命令和回传结果的仓库，并将用于传递命令的问题页地址硬编码在恶意代码程序中. 当恶意代码在被控主机中启动，会根据硬编码 URL 进行域名解析并不断向网页资源发起请求获取已编码的命令，本地解码后执行命令. 如命令执行结果需要回传，仍将结果回传至仓库问题页.

(2) canisrufus[②]. 使用 Github 仓库的代码页(Code)以新建文件的形式进行主机报活、命令传递和结果回传，文件内容均采用十六进制编码，文件命名方式与文件功能有关，如：发布命令则文件名为“job”开头，回传文件则文件名为“file”开头等. 控制者持有 Github 账号并在该账号下创建了用于传递命令和回传结果的仓库，并将 Github 账号和仓库名信息硬编码在恶意代码程序中. 恶意代码一旦在被控主机上成功运行就会向仓库中上传一个文件表明上线状态，随后定期根据硬编码的寻址信息进行域名解析并对网页资源发起请求获取新的已编码的命令，本机解码后执行命令并将执行结果上传至代码页.

2.3　流量特性

传统恶意代码与公共资源型恶意代码区别之一在于对 C2 服务器的控制权. 传统命令与控制通信中 C2 服务器通常为攻击者可控的服务器，控制端与服务端何时建立连接、何时断开连接、每个数据包传输何种内容均可控；而公共资源型远控通信中，C2 服务器仅为攻击者的访问提供端口和有限的控制权限，导致控制端与服务端之间的连接不完全可控，远控过程与正常通信仍有很大相似性，这使得某些用以检测传统恶意代码流量的特征对公共资源型恶意代码的检测无效.

表现在网络流量上，当对公共资源发起 HTTPS 请求时，整个通信流程如图 3 所示：① 根据域名发起 DNS 请求，获得服务器 IP；② 向服务器 IP 地址 443 端口请求 TCP 连接，与服务器进行三次握手；③ 与服务器进行 TLS 握手并对服务器身份进行认证；④ 认证成功后连接建立，向服务器发起 GET、POST 等请求；⑤ 服务器根据请求内容向客户端返回数据；⑥ 客户端请求断开连接，四次挥手，通信结束.

在与良性流量的相似性上，DNS 请求的域名为

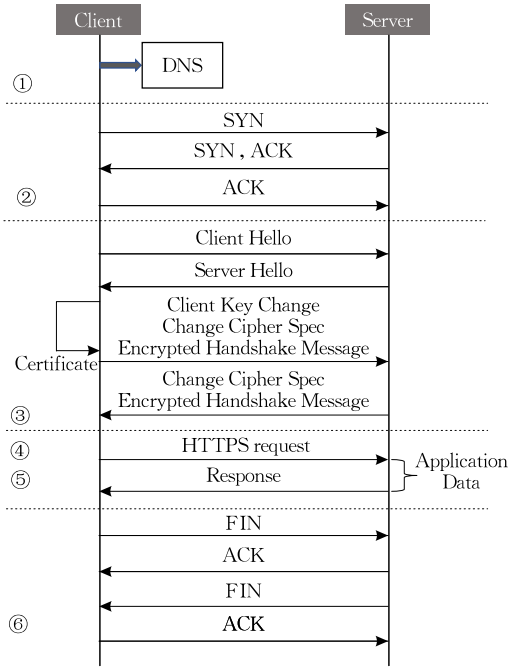

图 3　HTTPS 请求流程

正常域名，建立连接并进行三次握手的对象是公共资源服务器，HTTPS 请求内容被加密封装，通信结束后正常进行四次握手，这些过程与良性流量的通信过程并无不同. 另外，由于公共资源服务器不受攻击者控制，对某一资源发出请求后，一旦进入第⑤阶段，无论是普通用户还是恶意代码的宿主机，收到的数据都是相同的，在该阶段攻击流量与良性流量也具有相似性.

在与良性流量的差异性上，分析公共资源型恶意代码实例发现，在 TLS 握手阶段恶意代码支持的 TLS 版本及其他加密套件与良性流量中表现有所不同. 另外，恶意代码在访问公共资源时，出于在指定页面获取命令或泄露数据的目的，在请求频率等特征上表现异常，形成恶意代码的通信模式，这也为流量检测提供了突破口.

2.4　基于 StealthyFlow 的恶意代码命令与控制模型

针对恶意代码在通信模式上表现出的脆弱性，防御方可以在网络边界通过 IDS[6]/IPS、NTA、UEBA 等手段展开检测. 上述三种检测手段分别关注南北向(跨越网络边界)流量、南北向＋东西向(网络中横向移动)流量、日志信息，旨在发现网络通信中的潜在威胁. 应用本文提出的流量伪装框架，在被控主机

① Sean W. The case of getlook23: using Github Issues as C2. https://oalabs.openanalysis.net/2016/09/18/the-case-of-getlook23-using-github-issues-as-a-c2/, 2016

② Canisrufus project. https://github.com/maldevel/canisrufus, 2017

侧，恶意代码与服务器通信前先进行动态流量伪装，以良性流量为目标，对目的流量进行伪装，改变恶意代码的通信模式，使其与正常流量无法区分，即可导致上述检测手段失效. 体现在模型上，如图 1(b)所示，SFMalware 将恶意代码程序与 GAN 结合，在受害主机侧进行流量伪装，将具有明显通信模式的恶意流量伪装成对抗流量，不再被通信链路上的检测手段阻塞，可以实现与 C2 服务器的隐蔽通信.

3 StealthyFlow 框架设计与实现

本文提出了 StealthyFlow——一种利用 GAN 做流量伪装的框架. 其结构如图 4 所示，由 GAN 模型和恶意代码程序两部分构成，在受害主机上以 GAN 模型文件和恶意代码程序文件的组合形式存在.

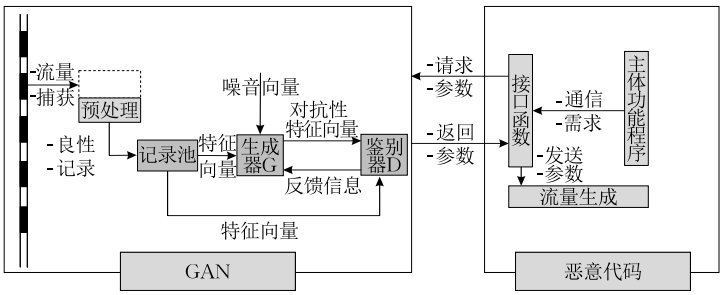

图 4 StealthyFlow 框架结构

StealthyFlow 的运行流程分为三个阶段，数据收集、接口调用和流量生成.

(1)数据收集. 为了实现动态流量伪装需实时捕获受害主机上的流量，过滤出公共资源流量，提取特征作为 GAN 模型的训练样本. 该过程采用进程注入的方法，将流量监听进程注入被控主机. 由于数据收集过程仅收集流量统计特征而非流量本身，因此不会产生过多资源消耗. 参考 ATT&CK 知识库中的诸多案例和反病毒厂商近年报道可见，进程注入方法是成熟的，已经被应用到攻击案例之中.

(2)接口调用：当恶意代码需要与 C2 服务器通信时，会通过接口函数调用 GAN 模型请求数据，后者向前者以向量列表的形式返回若干对抗性样本，向量列表如式(1)所示：

$$\begin{pmatrix} (x_{11},\cdots,x_{1j},\cdots,x_{1n}) \\ (x_{21},\cdots,x_{2j},\cdots,x_{2n}) \\ \vdots \\ (x_{i1},\cdots,x_{ij},\cdots,x_{in}) \\ (x_{m1},\cdots,x_{mj},\cdots,x_{mn}) \end{pmatrix} \tag{1}$$

其中 n 为特征数；m 为向量个数；x_{ij} 为流量某一特征统计特性的取值，例如流持续时间；$(x_{i1},\cdots,x_{ij},\cdots,x_{in})$为一个向量，代表一个数据流的 n 个特征值，例如取流长度、流持续时间、流间隔时间和 TLS 版本号四个特征时某一数据流的特征向量为(6263,0.0004,122.5960,1).

(3)流量生成. 恶意代码根据接口调用阶段获取的向量列表顺序取得一个向量，根据向量值对流量的相应统计特性进行调整后发起网络请求. 当向量列表中的数据依次取尽后，重复接口调用阶段操作.

在接下来的三个小节，我们分别从特征选择、GAN 模型、恶意代码程序三个角度详细阐述 StealthyFlow 原理.

3.1 特征选择

加密流量不同于 TCP 或 HTTP 流量，其数据内容摒弃明文采用密文，因此许多统计特性对于第三方不可见. 为了选择合适的特征进行流量伪装，本文首先总结检测系统常用特征，如表 1 所示.

表 1 已有加密流量检测工作中采用的特征总结

实例	检测对象	特征	数据集
[3]	TLS	包特征、TLS 特征、证书特征	CTU-13、MCFP
[7]	TLS	流特征、包特征、TLS 特征	自建
[8]	HTTP、HTTPS	流特征、TLS 特征、主机特征	自建
[9]	HTTPS	流特征、域名特征	自建
[10]	TLS	流特征、包特征、TLS 特征、主机特征、字节分布	自建
[11]	HTTPS	流特征、包特征、证书特征	CTU-13、MCFP
[12]	HTTPS	包特征、证书特征、HTTP 请求、域名特征、字节分布	自建
[13]	HTTPS/Tor	流特征、证书特征	DARPA
[14]	SSH、HTTPS 及非加密应用	流特征、包特征	GMU

由表 1 可见，当前主流的加密流量检测采用的特征可以归为流特征、包特征、TLS 特征、证书特征、主机维度特征五类. 实质上，上述维度都可由流长度、流持续时间、流间隔、包长度、包持续时间、包间隔、TLS 加密套件、TLS 扩展、证书链、自签名特征排列组合获得. 例如，可以选取流长度、流持续时间、流间隔、平均包长度、平均包持续时间、平均包间隔、TLS 加密套件数量、TLS 扩展数量、TLS 版本号作为特征构建检测模型. 为了实现隐蔽通信，绕过基于机器学习算法的黑盒检测器，需要提高黑白样本关于以上特征的不可区分性. 对于本文的研究对象公共资源型恶意代码而言，C2 服务器的特殊性令其在包长度、包持续时间、包间隔、证书链、自签名等特征上与良性流量天然具有不可区分性. 例如，公共资源作为 C2 服务器时，在接收到恶意代码所在客户端的请求后，其响应包的大小和包间隔、包持续时间等特征不受攻击者控制，仅取决于服务器相关特性，具体如 2.3 节所述. 因此，在进行流量伪装时，不必将这些特征考虑在内. 另外，对于 TLS 的加密套件和扩展，可以通过支持所有类型实现流量伪装.

综上所述，本文仅需着重在流持续时间、流间隔、流长度和 TLS 版本四个特征进行流量伪装，即可实现已有检测算法常用特征的伪装. 以 RAINYDROP 和 canisrufus 产生流量和正常用户使用 Github 产生的流量分别作为良性和恶意流量，对上述特征进行对比.

对比样本以双向流为单位，分别在良性和恶意流量中选取随机一周流量，由于样本量纲差异较大，因此对其进行了对数归一化的操作，结果如图 5 所示. 由图可见，良性和恶意流量在流持续时间、流间隔、TLS 版本和流长度四个特征下分别表现出一定的差异性. 以流持续时间为例，良性流量的流长度分布随着长度增加呈现阶梯式减少，而 RAINYDROP 产生的恶意流量的流长度分布随着长度增加呈现断崖式减少，canisrufus 产生的恶意流量的流长度分布随着长度增加先高后低，三者的分布各不相同. 在后续章节中，本文将基于 StealthyFlow 结构，针对流持续时间、流间隔、TLS 版本和流长度四个特征对恶意流量进行流量伪装，若伪装后的流量与良性流量实现不可区分性，则表明 StealthyFlow 的有效性. 另外，须知本文所选特征仅为示例，用以证明 StealthyFlow 结构的有效性，在实际部署过程中，可以对特征进行灵活调整，仅需改变恶意代码产生流量的接口即可.

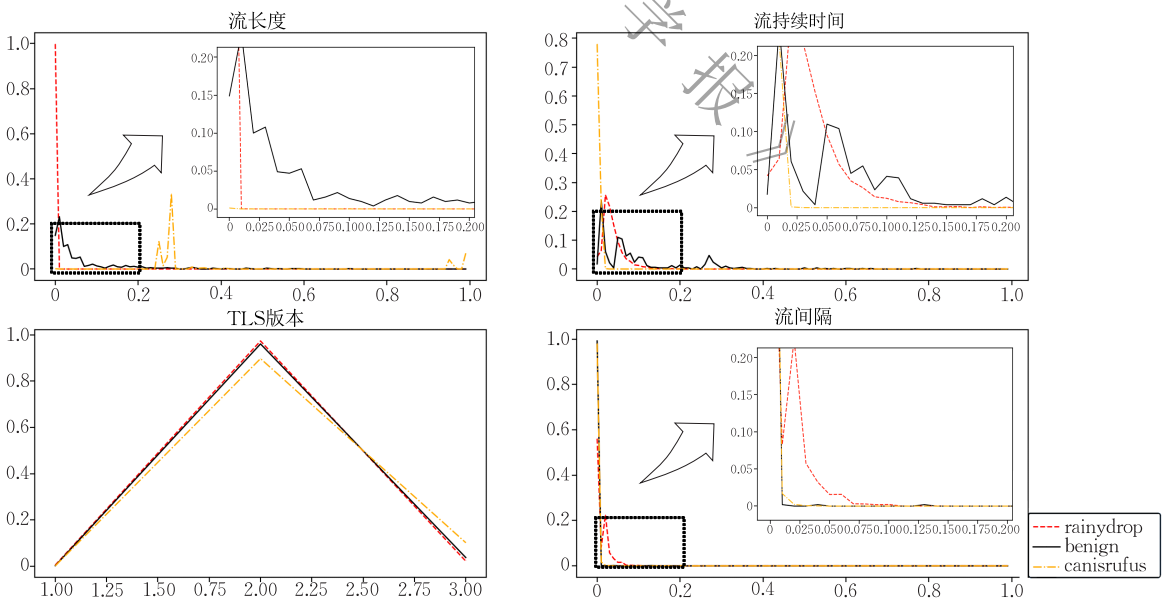

图 5　良性和恶意流量特征对比(其中红色短划线表示 RAINYDROP 流量，橙色点划线表示 Canisrufus 流量，黑色实线表示良性流量，统计结果以百分比形式呈现. 另外，流长度、流持续时间和流间隔子图中的嵌套子图分别是对应子图黑色虚线框部分的放大显示)

3.2　GAN 模型

生成对抗网络最初由 Goodfellow 等人[15] 于 2014 年提出，是训练生成模型的一种框架，其主要思想是在生成器(Generator，简称为 G)和鉴别器(Discriminator，简称为 D)之间进行博弈，以得到最佳生成样本. GAN 在图像、声音和文本生成领域均表现良好，信息安全领域也已经存在相关研究成果. 2017 年提出的 WGAN-GP[16] 中以梯度惩罚(gradient

penalty)的思路解决了梯度消失和梯度爆炸的问题,训练更稳定、收敛速度更快,一直备受研究者重视. StealthyFlow 的 GAN 模型部分采用 WGAN-GP 模型,在给定良性流量特征向量集合的基础上,生成与其相似的特征向量样本. 本文中 WGAN-GP 模型基于 TensorFlow 框架进行实现.

生成器的训练集存放在一个记录池中,该记录池中包含固定时间间隔阈值内主机上与公共资源通信产生的所有 M 条良性流量特征向量记录,在本文中,取两周为记录池的时间间隔阈值. 记录指以受害主机与指定公共资源间通信产生的双向加密流为基础,提取的流持续时间、流间隔、TLS 版本和流长度的统计值向量. 一条记录由 4 个特征的统计值组成,表示为(x_{i1},x_{i2},x_{i3},x_{i4}),记录池表现为 $M\times 4$ 的矩阵. 如图 6 所示,被控主机上实时捕获的流量经预处理后形成特征向量,对记录池进行实时更新.

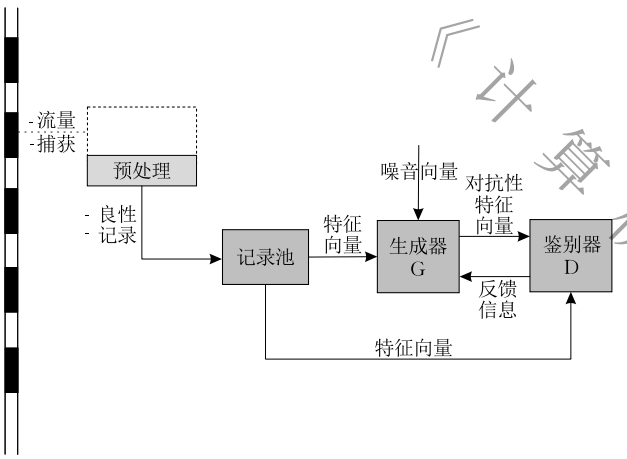

图 6 部署在被控主机的 GAN 模型

控制记录池时间间隔阈值的原因有二:其一是出于空间考虑,维持一个记录池需要的存储空间过大时,会增加恶意代码暴露的风险. 其二是出于生成器学习效果考虑,将数据控制在一定时间范围内并实时更新,可以保证用于生成器的数据集的实时性,同时有助于学习用户的活动模式(例如用户在一天中的某个时间段活动较为频繁),对于绕过 UEBA 有良好的效果.

记录池中的记录可能存在分布不均的情况,例如某个特定时间段的记录数据量极大或极小,这些都是用户实体行为习惯的正常表现. 某时段内数据量极小(可能为零)时,说明用户在该时段内不活跃,为隐蔽起见,这种不活跃的趋势应该加以保留,因此需要降低恶意代码在该时段内的通信频率或停止恶意代码活动. 某时段内数据量极大说明用户在该时段内表现活跃,通信频率极高,在此基础上,恶意代码的通信请求会更快被处理,此时通信频率的上限取决于恶意代码的通信需求.

如图 6 所示,生成器(G)以随机噪音向量为输入,在记录池中取当前时间上下误差一小时的所有数据作为训练集,生成对抗性特征向量. 鉴别器(D)使用生成器生成的对抗性特征向量和生成器训练集中的特征向量进行训练,并将训练结果反馈生成器,辅助生成器调整参数以得到最佳生成样本.

3.3 恶意代码程序

恶意代码程序是 StealthyFlow 的重要组成部分. 如图 7 所示,StealthyFlow 实施流量伪装首先需要恶意代码程序的主体功能程序部分发起通信请求,然后由接口函数调用 GAN 模型获取指导流量生成的参数,最终由流量生成模块产生流量. 伪装对象是数据流的统计特征,并不涉及对通信内容的修改,因此对于所有具有网络通信功能可进行远程命令与控制的恶意代码具有普适性. 为了验证 StealthyFlow 的有效性,本文使用 RAINYDROP 和 canisrufus 两个基于 Github 进行 C2 的恶意代码为原型,保留主机功能部分,修改通信函数代码使其调用 WGAN-GP 模型获取数据并调整自身网络行为,伪装数据流的流持续时间、流间隔、TLS 版本和流长度.

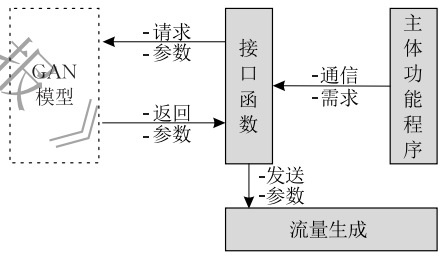

图 7 恶意代码程序

具体实现方法为:通过在流持续时间参数指定的时间值内保持连接建立,可以控制当前流的持续时间;通过开启多个线程,可以保证即使仍有 TLS 流未断开连接,当前流也可以顺利建立;调整连接的上下文参数可以指定 TLS 版本;对 HTTPS 请求进行数据填充,可以控制 TLS 流总字节长度等. 通过对上述特征统计值的修改,可以构造出近似于正常流量的对抗流量.

3.4 StealthyFlow 绕过原理分析

StealthyFlow 可以绕过基于特征工程的黑盒机器学习检测系统,包括 IDS/IPS、NTA、UEBA 等. 以下对绕过原理进行分析:

(1) IDS/IPS. 关注点在于网络边界即客户端与服务端通信流量. 它使用流量统计特征的机器学习分

类算法发现异常. StealthyFlow 改变了被控主机和公共资源服务器之前的通信流的统计特征，隐藏了客户端和服务器之间的流量异常，因此可以绕过检测.

(2) NTA. 关注点结合网络边界流量和网络中横向流量，即客户端与服务端、客户端与客户端、服务端与服务端的通信流量都在监控范围之内. 但本文仅讨论被控主机处于对抗环境中的情况，因此不涉及服务端与服务端的通信流量. 另外，公共资源的客户端与客户端之间也没有直接的通信信道. 因此 StealthyFlow 只需绕过客户端与服务端的通信监控，其原理与 IDS/IPS 绕过原理一致.

(3) UEBA. 对正常用户行为模式建模，当流量偏离了正常模型，则判定为异常. StealthyFlow 控制恶意流量的统计特性，使其在保留攻击功能的同时，在行为模式上与正常流量趋于一致，因此也可以实现绕过.

如 3.1 节所述，加密流量检测的已有工作关注的特征都可由流长度、流持续时间、流间隔、包长度、包持续时间、包间隔、TLS 加密套件、TLS 扩展、证书链、自签名特征排列组合获得. 对于任意一个基于机器学习算法的黑盒分类器，如果能够对这些特征进行伪装，就能实现攻击流量与良性流量的不可区分性，因此可以绕过黑盒分类器.

4　实验验证

4.1　数据集建立

众所周知，TLS 协议已经在近几年广泛应用，ATT&CK 数据显示，至少有 39 个知名 APT 组织或攻击工具利用过基于 TLS 协议的公共服务资源进行远程命令与控制，研究者应该对此类恶意代码进行深入研究[17]. 本文针对公共资源型恶意代码原型开展研究，实验涉及三个数据集，以下详细介绍如何选择和获取这些数据集.

(1) 恶意数据集(以下简称黑样本). 目前绝大多数恶意流量数据集仅包含 HTTP、TCP 协议流量，包含 HTTPS、TLS 协议攻击流量的数据集极少，加密流量检测领域已有工作中的数据集一般取自 VirusTotal①、MCFP② 数据集[18]或自建数据集. 由于 VirusTotal 并非开源，且 MCFP 数据集中缺乏基于公共资源的恶意代码，为了获取恶意数据集，本文采取自建数据集形式. 笔者通过对 RAINYDROP 二进制样本进行静态分析，得到其实施 C2 行为的 Github 页面；通过将二进制样本在隔离网络下专用物理主机中运行，捕获并过滤得到承载攻击行为的恶意流量，以此作为恶意数据集的一部分. 另外，canisrufus 是 Github 开源项目，通过部署被控端和控制端环境，执行项目代码在被控端捕获攻击流量作为恶意数据集的另一个组成部分. 最终得到 8736 个 RAINYDROP 数据流和 29 474 个 canisrufus 数据流作为恶意数据集.

(2) 良性数据集(以下简称白样本). 本文关注对象是僵尸网络类型的远控型恶意代码而非远控木马(Remote Access Trojan). 两者的区别是：前者的命令控制过程是异步的、有延迟的，上行流量往往小于下行流量，如同网页浏览；后者的命令控制过程是同步的、实时的，上行流量往往大于下行流量，如同文件/屏幕等数据上传. Github 是 APT 中远控型恶意代码常用的公共资源，因此选取 Github 日常访问流量作为学习对象. 对实时性要求较高的通信或数据传输量较大的通信，应选取符合要求的目标流量作为学习对象. 例如，对于远控型木马，则应选取诸如邮件发送、文件上传类上行流量大于下行流量的服务进行学习.

本文实验中良性数据集由正常用户访问 Github 产生的流量构成. 为了收集良性数据集，需要捕获实验主机与 Github 通信流量. 由于流量基于 TLS 协议，无法根据报头信息识别. 目前已有的针对 TLS 流量进行特定服务识别的研究，无法做到百分百准确. 本文的做法是，以查询方式获取 Github 网站所有域名及其下属子域名对应的 IP 地址，根据这些地址对实验主机捕获的流量进行过滤. 最终得到用户访问 Github 产生的 825 个数据流作为良性数据集.

(3) 生成数据集(以下简称灰样本). 以 RAINYDROP 和 canisrufus 为原型，应用 StealthyFlow 框架，得出的新型恶意代码(SFMalware)，在专用物理主机上运行并收集攻击流量，最终得到以 RAINYDROP 为原型的 SFMalware 流量共计 570 个数据流和以 cainsrufus 为原型的 SFMalware 流量共计 621 个数据流.

4.2　数据预处理

数据预处理的目的是将流量包以 TLS 流的形式聚合并进行特征提取、数据清洗和数据归一化三个步骤/操作. (1) 特征提取是将数据由流量转换为向量的重要步骤，提取的特征包括流持续时间、流间

① Virustotal. com. https://www. virustotal. com/gui/home/upload

② Stratosphere Lab. https://www. stratosphereips. org/datasets-malware

隔、TLS 版本和流长度；(2) 数据清洗操作处理特征提取步骤所得向量中的无效向量(例如未完成三次握手的数据等)，并将特征提取得到的向量进行数值转换；(3) 数据归一化用于 GAN 结构生成样本过程，其目的是避免不同特征的量纲或量纲单位不同而影响生成样本的效果. 为了降低少数异常值对训练结果的影响，本文采用 RobustScaler 标准化函数进行归一化处理.

在 GAN 结构指导样本生成过程中，三个步骤缺一不可；而在黑盒检测器检测恶意流量过程中，仅需进行特征提取和数据清洗两个步骤.

4.3 模型训练及实验结果

训练参数是多次实验对比得出的，能令 GAN 模型快速收敛的取值. StealthyFlow 选取的批尺寸为 10，共训练了 100 个 epoch. 生成器和鉴别器的学习率为 0.0001，指数衰减率为 0.5，训练集的维度为 4，噪声向量的维度为 2.

为了全面、深入地评估模型的能力，实验中使用了四种机器学习算法构建黑盒检测器. 根据入侵检测的相关研究，实验中采用的机器学习算法包括决策树(Decision Tree，DT)、随机森林(Random Forest，RF)、自适应增强(Adaptive Boosting，AdaBoost)和梯度树提升(Gradient Tree Boosting，GTB). 在对 StealthyFlow 进行评估之前，已经以黑白样本数据集中部分数据对黑盒检测系统进行了训练.

对于实验指标，测量了逃逸率以展示 StealthyFlow 的性能. 逃逸率(Escape Ratio，ER)反映了黑盒检测系统未能正确检测到的恶意流量记录占所有恶意流量记录的比例，直接显示了模型逃避检测能力和检测系统的鲁棒性. 其计算方法为

$$ER=\frac{\text{错误分类的非良性流量记录数}}{\text{非良性流量记录总数}}=\frac{FN}{TP+FN} \tag{2}$$

逃逸率的计算公式与机器学习中常用的测量指标召回率具有紧密联系. 恶意流量召回率($recall$)的计算公式为

$$recall=\frac{\text{正确分类的非良性流量记录数}}{\text{非良性流量记录数}}=\frac{TP}{TP+FN} \tag{3}$$

逃逸率与召回率的关系公式为

$$ER=1-recall \tag{4}$$

逃逸率和召回率均只关注非良性流量的分类结果，排除了良性流量与非良性流量数据不平衡的场景下良性流量对整体结果带来的影响，相比召回率，逃逸率更加直观地展示了 StealthyFlow 的绕过效果，因此选择逃逸率作为测量指标. 原始逃逸率和对抗逃逸率分别表示对原始恶意流量记录逃逸率和对恶意流量记录逃逸率.

实验分两部分进行，分别称为初始实验和对抗实验.

(1) 初始实验. 初始实验的目的是测量机器学习分类器对黑白样本的分类能力. 以黑白样本作为数据集对分类器进行训练，黑白样本比例为一比一，训练集和测试集分别占数据集的 80% 和 20%，测量得到各种机器学习算法下黑样本的逃逸率以及整体的准确率. 如表 2 第 2～5 行所示，黑样本的逃逸率最高为 1.54%，表明机器学习分类器能正确分类黑样本；整体准确率最低为 97.08%，表明分类器对于黑白样本的分类均有很好的效果.

(2) 对抗实验. 以黑白样本作为训练集，以白样本和灰样本作为测试集，测量灰样本的逃逸率. 实验结果如表 2 第 6～9 行和图 8 所示，经过 StealthyFlow 调整流量后，逃逸率增加到 94% 以上，准确率大大降低，这意味着分类器几乎将所有灰样本错误分类为白样本，即与黑样本相比，白样本与灰样本的相似度更高.

表 2 初始实验和对抗实验中机器学习分类器对于流量的分类效果(以逃逸率和准确率为测量指标)

测量指标	恶意代码种类	DT/%	RF/%	AdaBoost/%	GTB/%
原始逃逸率	RAINYDROP	1.54	0.62	1.54	0.92
	Canisrufus	1.44	0.96	0.96	1.91
原始准确率	RAINYDROP	98.00	98.46	97.08	97.85
	Canisrufus	98.09	98.80	98.33	98.33
对抗逃逸率	RAINYDROP	94.05	96.89	96.48	98.27
	Canisrufus	95.10	97.11	95.72	98.00
对抗准确率	RAINYDROP	6.93	3.96	4.95	3.11
	Canisrufus	5.94	4.06	4.99	2.97

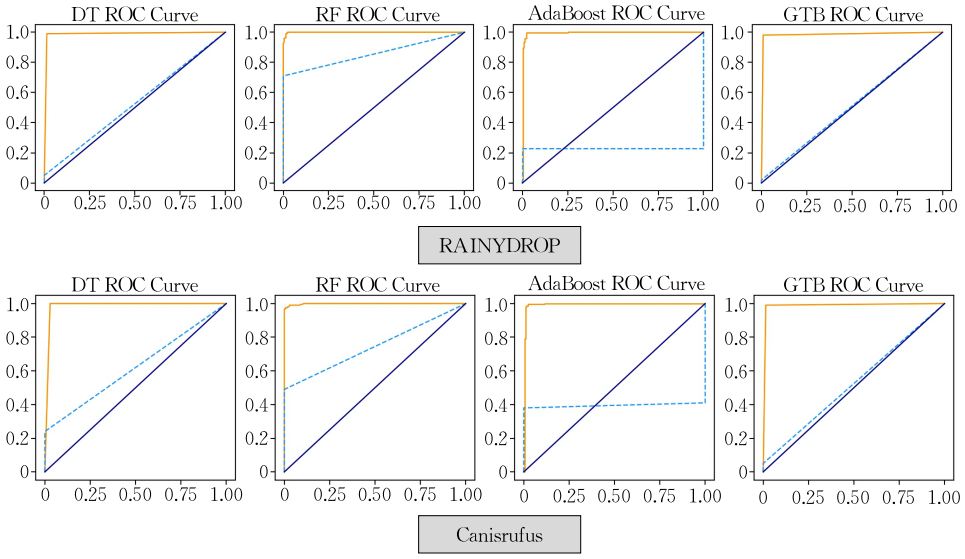

图 8　四种机器学习算法下初始实验和对抗实验的 ROC 曲线图(上下两个部分分别代表两种恶意代码的实验结果. 其中橙色实线代表初始实验 ROC 曲线,蓝色虚线代表对抗实验 ROC 曲线. 由图可见初始实验检测率大于对抗实验检测率)

为了直观展示流量伪装的效果,以时间为轴,分别统计一周时间内良性、恶意、对抗性流量的加密流的分布情况. 由图 9 可见,良性流量在一周中活动时间分布不均匀,每次活动持续时间不超过五小时,夜晚几乎完全没有活动痕迹. 未经过伪装的流量在全天 24 小时活动频率几乎无间断,因此容易被发现;而经过 StealthyFlow 伪装的对抗性流量,在活动时间特性上与良性流量接近一致,因此不容易被发现.

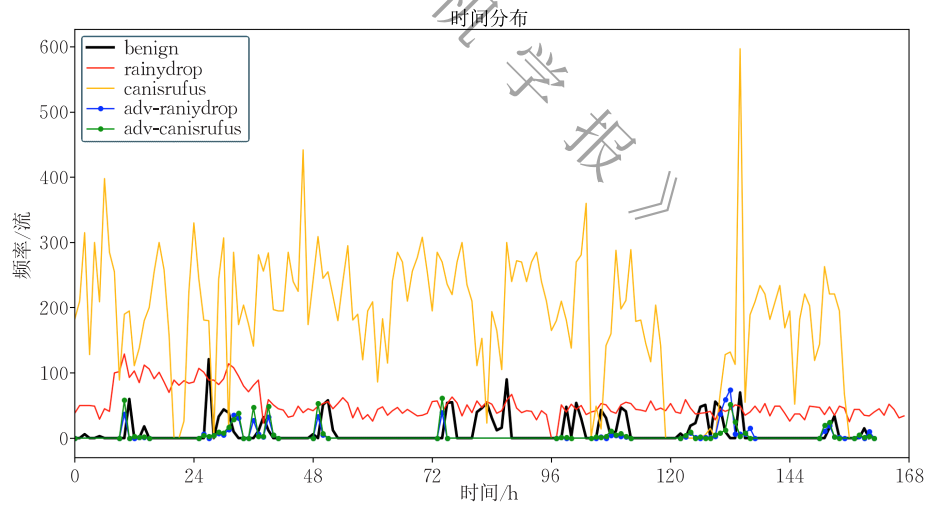

图 9　时间分布对比(横坐标表示时间,以小时为单位,纵坐标表示频率,以流为单位,其中黑色折线代表白样本,红色和黄色折线分别代表 RAINYDROP 和 canisrufus 黑样本,蓝色和绿色点状实线分别代表以 RAINYDROP 和 canisrufus 为原型产生的灰样本)

4.4　实验讨论

过拟合问题:由于实验数据集较小,实验结果可能存在过拟合问题,但是在时间分布实验中可以明确看出经 StealthyFlow 伪装的流量与良性流量表现相似,不再具有原始攻击流量的明显异常. 因此,过拟合问题并不会对本文的研究结果造成影响.

攻击功能问题:由于本文中用于流量伪装的特征是数据流的浅层特征,而非恶意代码的固有属性,因此对其进行调整并不影响恶意代码攻击功能的正常执行. 同时,我们也运行了基于 StealthyFlow 的恶意代码,验证其执行了恶意代码原型会执行的恶意操作,即攻击功能未受影响.

信道效率问题:传统公共资源型恶意代码存在通信频率过高且规律性强的特征,因此容易被发现. 针

对这种情况，StealthyFlow 将通信的平均频率降低以避免异常，同时每间隔一段时间有一个请求集中点可以实现对实时性要求较高的通信，从而保证攻击行为的正常进行.

5 相关工作

为了提高攻击流量的隐蔽性，降低流量检测的准确率，研究人员对流量伪装技术进行了深入研究.

2018 年，Rigaki 等人[19]提出一种流量伪装技术，修改恶意软件源代码，调整恶意软件网络行为，学习合法应用程序流量，成功绕过网络边界检测. 模型实现采用生成对抗网络，以真实的 Facebook 聊天流量作为白样本进行训练，生成流的总字节数、持续时间和流时间间隔三个统计特征值. 恶意软件根据以上三个参数调整其流量，与 C2 服务器进行通信，并根据流量是否被 IPS 阻塞对 GAN 进行反馈. 结果表明，只需使用少量数据训练模型，就可以生成模拟 Facebook 聊天流量的对抗流量，该流量在保持恶意软件正常通信功能的基础上，可以通过 IPS 的检测. 此研究是恶意代码通过流量伪装适应检测的第一次尝试，但这项研究存在几个问题：(1) 实验中的 IPS 为自建白盒检测模型，无法适应其他流量检测工具；(2) IPS 参与了模型的训练过程，可能导致恶意代码提前暴露；(3) 实验最终生成非加密流量，对于利用加密流量进行隐蔽通信的恶意代码不具有普适性.

同年，Lin 等人[20]提出了 IDSGAN 框架，以 NSL-KDD 为数据集，在确保攻击行为正常进行的前提下，通过对攻击流量的非功能性特征进行修改，将原始恶意流量转换为对抗性恶意流量，以绕过黑盒检测器. 该实验以 Wasserstein GAN 为基础框架，利用黑盒检测器检测结果作为标签训练鉴别器，使鉴别器模拟黑盒检测器的检测结果. 这项工作为流量伪装研究提供了新的思路，然而在真实场景中，黑盒检测器检测到恶意流量后可能触发其他操作打断恶意代码的攻击过程甚至将恶意代码从受害主机上移除，将黑盒检测器引入训练过程会使实验变得不可控. 另外，实验在流量特征维度展开，直接对特征参数进行修改，对于修改后的特征能否生成实际攻击流量欠缺考虑.

2019 年，Li 等人[21]提出了 FlowGAN，将目标流量变形为正常流量，进而绕过互联网审查. FlowGAN 的核心思想是自动学习正常网络流的特征，采用生成对抗网络模型，生成器使用累计行为表征(CUMUL)指导流量生成，鉴别器以传出数据包总数 Nout、传入数据包总数 Nin、传出数据包 Sout 的字节总和、传入数据的字节总和数据包 Sin、累积字节 Cum、数据包 AvgInter 之间的平均间隔共六个特征进行分类，最终通过本地代理发送流量绕过 ISP 审查. 这项工作可以实现针对任意目标流量进行学习的流量伪装，是一项很有价值的工作. 但流量伪装直接针对流量进行，无法验证攻击功能是否得以保持. 另外，文中提出的动态流量伪装，仅代表可以学习各种目标流量，但未提及目标流量实时变化的情况，因此并非完全意义上的动态流量伪装.

针对流量伪装技术的研究目前仍在起步阶段，相关工作较少，我们选取了上述三个具有代表性的工作进行总结，可以发现，生成对抗网络在对抗性样本生成领域具有的极大优势，为恶意代码攻防研究提供了一种新思路[22]. 上述研究表明，利用生成对抗网络进行流量伪装，可以生成逼近于良性流量的样本，实现匿名通信，绕过流量检测.

然而与此同时，当前的流量伪装技术研究面临一个共同的问题，即结果验证问题. 可用于验证流量伪装效果的开源检测器不足，研究成果无法以某一检测器为基准进行对比，多数研究通过自建检测器的方法解决. 本文提出的 StealthyFlow 结构，以 GAN 模型和恶意代码相结合，利用良性流量统计特征进行动态流量伪装，可以实现对抗流量与良性流量的不可区分性，进而绕过基于机器学习的黑盒分类器，涉及效果验证部分，也通过自建多个不同算法的机器学习分类器进行. 这一现状，也对检测研究提出了更高的期望.

6 防御方法

实验表明基于 StealthyFlow 框架的恶意代码可以有效逃避流量检测，需要引起重视并提出解决方案. 本文建议在云端以云端检测和云端阻断两种方法进行防御.

云端检测技术利用服务提供商身份对平台数据内容与流量的监管权限，分别从内容、活动模式和访问模式三个维度进行异常检测.

(1) 内容. StealthyFlow 借助加密流量逃避内容检测实现隐蔽性，但到达云端的数据一般以明文形式存储，可以采用内容审查机制发现异常. 审查范围包括：特征码、编码方式、可疑链接等.

(2) 活动模式. 控制者传输控制命令和收取命令结果以公共资源为代理，这种恶意行为常表现出某种规律性，因此监控用户向公共资源传输的内

容、传输频率、编码与否、传输时间分布等特征有助于发现异常；

(3) 访问模式. 与活动模式类似，被控主机被恶意代码驱动可能向 C2 服务器发送心跳数据、获取命令或回传结果，挖掘请求频率、请求包长度、请求发起时间等统计特性的潜在特点也有助于发现异常行为.

云端阻断采用随机人机验证技术实现. 基于 StealthyFlow 的恶意代码在流量维度实现了与正常用户的不可区分性，但恶意代码运行过程缺少人为控制，在访问公共资源时灵活性不够，可以在其向公共资源发起连接请求时进行人机验证. 考虑到用户体验，本文提出可以以概率 x(概率 x 由用户干扰度分析得出)对请求访问服务器的用户进行随机人机验证，验证失败则终止访问，有效阻断恶意代码的通信过程.

7　结　论

本文构建了一个名为 StealthyFlow 的动态流量伪装框架，将恶意代码与生成对抗网络结合，生成与良性流量难以区分的对抗流量，并利用公共服务资源作为 C2 服务器进行通信. 以已有恶意代码为原型，仅对通信函数代码进行修改使其连接 GAN 模型，即可在保持攻击行为的基础上进行流量伪装. 实验结果证明，基于 StealthyFlow 框架的恶意代码具有产生承载攻击行为的对抗流量的能力，可以在四种机器学习算法的黑盒分类器中达到 94% 以上的逃逸率.

公共资源涉及的用户和网络资产众多，据悉，仅 Github 一项公共服务就在全球拥有超过 3600 万用户. 由于用户群十分庞大，因此安全问题显得尤为重要. 针对公共资源被攻击者滥用问题，防御者不断优化流量检测方案. 然而出于利益驱使，恶意代码需要逃避流量检测，流量伪装技术势必也会有新的发展，恶意代码与人工智能结合是必然的趋势. 可以预见，基于 StealthyFlow 框架的 SFMalware 一旦被真实应用，将带来巨大的安全隐患. 因此，预先研究此类高级恶意代码的防御方案是有必要的.

参　考　文　献

[1] Čech P, Kohout J, Lokoč J, et al. Feature extraction and malware detection on large HTTPS data using MapReduce//Proceedings of the International Conference on Similarity Search and Applications. Tokyo, Japan, 2016: 311-324

[2] Sharevski F, Jachim P, Florek K. To tweet or not to tweet: Covertly manipulating a Twitter debate on vaccines using malware-induced misperceptions. ArXiv preprint arXiv: 2003.12093, 2020

[3] Liu J, Zeng Y, Shi J, et al. MalDetect: A structure of encrypted malware traffic detection. CMC-COMPUTERS MATERIALS & CONTINUA, 2019, 60(2): 721-739

[4] Pan Wu-Bin, Cheng Guang, Guo Xiao-Jun, et al. Overview and prospects of research on network encrypted traffic recognition. Journal on Communications, 2016, 37(9): 154-167 (in Chinese)
(潘吴斌，程光，郭晓军等. 网络加密流量识别研究综述及展望. 通信学报，2016，37(9)：154-167)

[5] Yang S, Chitturi K, Wilson G, et al. A study of automation from seed URL generation to focused web archive development: The CTRnet context//Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries. Washington, USA, 2012: 341-342

[6] Vinayakumar R, Alazab M, Soman K P, et al. Deep learning approach for intelligent intrusion detection system. IEEE Access, 2019, 7: 41525-41550

[7] Anderson B, McGrew D. Machine learning for encrypted malware traffic classification: Accounting for noisy labels and non-stationarity//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Halifax, Canada, 2017: 1723-1732

[8] Chen Y C, Li Y J, Tseng A, et al. Deep learning for malicious flow detection//Proceedings of the 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC). Montreal, Canada, 2017: 1-7

[9] Prasse P, Machlica L, Pevný T, et al. Malware detection by analysing network traffic with neural networks//Proceedings of the 2017 IEEE Security and Privacy Workshops (SPW). San Jose, USA, 2017: 205-210

[10] Anderson B, Paul S, McGrew D. Deciphering malware's use of TLS (without decryption). Journal of Computer Virology and Hacking Techniques, 2018, 14(3): 195-211

[11] Shekhawat A S, Di Troia F, Stamp M. Feature analysis of encrypted malicious traffic. Expert Systems with Applications, 2019, 125: 130-141

[12] Anderson B, McGrew D. Identifying encrypted malware traffic with contextual flow data//Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security. Vienna, Austria, 2016: 35-46

[13] Schatzmann D, Mühlbauer W, Spyropoulos T, et al. Digging into HTTPS: Flow-based classification of webmail traffic//Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement. Melbourne, Australia, 2010: 322-327

[14] Wright C V, Monrose F, Masson G M. On Inferring application protocol behaviors in encrypted network traffic. Journal of Machine Learning Research, 2006, 6(4): 2745-2769

[15] Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets//Advances in Neural Information Processing Systems. British Columbia, Canada, 2014: 2672-2680

[16] Gulrajani I, Ahmed F, Arjovsky M, et al. Improved training of wasserstein gans//Advances in Neural Information Processing Systems. Long Beach, USA, 2017: 5767-5777

[17] Stojanović B, Hofer-Schmitz K, Kleb U. APT datasets and attack modeling for automated detection methods: A review. Computers & Security, 2020, 92: 101734

[18] Ismail Z, Jantan A, Yusoff M N, et al. The effects of feature selection on the classification of encrypted botnet. Journal of Computer Virology and Hacking Techniques, to appear

[19] Rigaki M, Garcia S. Bringing a GAN to a knife-fight: Adapting malware communication to avoid detection//Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW). San Francisco, USA, 2018: 70-75

[20] Lin Z, Shi Y, Xue Z. Idsgan: Generative adversarial networks for attack generation against intrusion detection. arXiv preprint arXiv:1809.02077, 2018

[21] Li J, Zhou L, Li H, et al. Dynamic traffic feature camouflaging via generative adversarial networks//Proceedings of the 2019 IEEE Conference on Communications and Network Security (CNS). Washington, USA, 2019: 268-276

[22] Ring M, Schlör D, Landes D, et al. Flow-based network traffic generation using generative adversarial networks. Computers & Security, 2019, 82: 156-172

HAN Yu, M. S. candidate. Her main research interest is network security.

FANG Bin-Xing, Ph. D., academician of the Chinese Academy of Engineering. His main research interests include computer architecture, computer network and information security.

CUI Xiang, Ph. D., professor. His main research interest is network security.

WANG Zhong-Ru, Ph. D., senior engineer. His main research interests include artificial intelligence and network security.

JI Tian-Tian, Ph. D. candidate. Her main research interest is network security.

FENG Lin, M. S. candidate. His main research interest is network security.

YU Wei-Qiang, M. S. His main research interests include network security and artificial intelligence.

Background

The abnormality of traffic is a key issue in the remote command and control process of malicious code. With the current traffic detection technology, machine learning algorithms based on feature engineering can classify benign traffic and malicious traffic on the premise of having a training set. In order to study the feasibility of countering machine learning detection, traffic masquerading methods are proposed. At present, a traffic masquerading method based on deep learning has been proposed, which realizes the automation of traffic masquerading. However, the traffic masquerading methods proposed in the existing work are generally aimed at the masquerading of non-encrypted traffic. At the same time, traffic masquerading is carried out directly on existing traffic, which cannot guarantee the integrity of the attack function. In addition, existing methods cannot cope with scenarios where the target traffic changes. In response to the above problems, this paper proposes a new traffic camouflage framework StealthyFlow, which directly generates adversarial encrypted traffic by modifying the code dimension. StealthyFlow realizes dynamic flow camouflage through real-time flow capture and learning.

This work was supported by the Guangdong Province Key Field Research and Development Program under Grant Nos. 2019B010137004 and 2019B010136003, the National Key Research and Development Program under Grant Nos. 2018YFB0803504 and 2019YFA0706404.