面向 AI 模型训练的 DNS 窃密数据自动生成

冯 林[1], 崔 翔[1], 王忠儒[2], 甘蕊灵[3], 刁嘉文[3], 韩冬旭[4], 姜 海[5]

[1]广州大学网络空间先进技术研究院 广州 中国 510006
[2]中国网络空间研究院 北京 中国 100010
[3]北京邮电大学网络空间安全学院 北京 中国 100876
[4]中国科学院信息工程研究所 北京 中国 100093
[5]北京丁牛科技有限公司 北京 中国 100081

摘要 近年来, 借助 DNS 协议良好的隐蔽性和穿透性实施数据窃取已成为诸多 APT 组织青睐的 TTPs, 在网络边界监测 DNS 流量进而精准发现潜在攻击行为已成为企事业单位急需建立的网络防御能力。然而, 基于 DNS 的 APT 攻击所涉及的恶意样本存在难获取、数量少、活性很低等现实问题, 且主流的数据增强技术不适合移植到网络攻防这个语义敏感领域, 这些问题制约了 AI 检测模型训练。为此, 本文基于 DNS 窃密攻击机理分析, 并结合了大量真实 APT 案例和 DNS 工具, 提出了一种基于攻击 TTPs 的 DNS 窃密流量数据自动生成及应用方法, 设计并实现了 DNS 窃密流量数据自动生成系统—MalDNS, 以生成大规模、高逼真度、完备度可调的 DNS 窃密数据集。最后, 通过实验验证了生成流量数据的有效性, 以及对检测模型训练的有效支撑。

关键词 DNS 窃密; 数据自动生成
中图法分类号 TP393.08 **DOI 号** 10.19363/J.cnki.cn10-1380/tn.2021.01.01

Automatic Data Generation of DNS-Based Exfiltration for AI-Model Training

FENG Lin[1], CUI Xiang[1], WANG Zhongru[2], GAN Ruiling[3], DIAO Jiawen[3], HAN Dongxu[4], JIANG Hai[5]

[1] Cyberspace Institute Advanced Technology, Guangzhou University, Guangzhou 510006, China
[2]Chinese Academy of Cyberspace Studies, Beijing 100010, China
[3] School of Cyberspace Secunty, Beijing University of Posts and Telecommunications, Beijing 100876, China
[4] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
[5] Beijing DigApis Technology Co., Ltd, Beijing 100081, China

Abstract In recent years, it has become the favorite TTPs of many APT organizations to implement data exfiltration by taking advantage of the good concealability and penetration of DNS protocol. Therefore, it's imperative for enterprises and institutions to establish the defense capacity to monitor DNS traffic at the network boundary so as to accurately detect the potential attack behavior. However, datasets of DNS-based APT campaigns involve lots of practical problems such as difficulty to obtain, small quantity, and low activity. Also, the available technology of data augmentation is not suitable for transplanting to such semantic sensitive field. These problems have restricted the training of AI detection models. Therefore, based on the analysis of DNS-based exfiltration mechanism, combined with a large number of real APT cases and DNS-based exfiltration tools, we propose a method that can automatically generate traffic data based on DNS -based exfiltration TTPs. We design and establish an automatic generation system named MalDNS to generate a target DNS-based exfiltration dataset with large-scale, high fidelity, and adjustable integrity. Finally, our experiments indicate that the generated dataset is effective and can support the training of the detection models effectively.

Key words DNS-based exfiltration; data generation

通讯作者: 崔翔, 博士, 教授, 主要研究领域为网络安全, E-mail: cuixiang@gzhu.edu.cn; 王忠儒, 博士, 高级工程师, 主要研究领域为人工智能、网络安全, E-mail: wangzhongru@bupt.edu.cn。

本课题得到广东省重点领域研发计划项目(No. 02019B010136003, No. 2019B010137004)和国家重点研发计划项目(No. 2018YFB0803504, No. 2019YFA0706404)资助。
收稿日期: 2020-9-29; 修改日期: 2020-11-24; 定稿日期: 2020-11-25

1 引言

目前, 人工智能(Artificial Intelligence, AI)技术的发展与应用进入第三次高潮,其解决实际问题的能力和应用潜能已得到广泛认可。AI 助力网络安全, 给网络安全防御方(后文简称“防御方”)带来了无限机遇和挑战; AI 助力防御的应用潜能需要不断被发掘, 从而有效提升防御方势能。在 AI 算法趋于成熟、算力已经得到大幅提升的情况下, 数据集已经成为限制网络安全类 AI 模型(后文简称“AI 模型”)性能的一个重要因素。

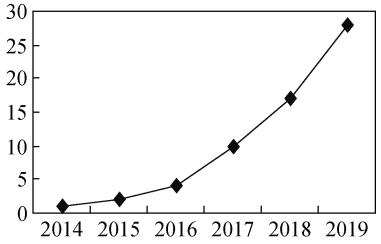

图 1 基于 MITRE ATT&CK 的 DNS 窃密事件统计
Figure 1 The Statistics of DNS-based exfiltration cases based on MITRE ATT&CK

为了深入分析当前 AI 模型面临的数据集问题, 需要选取特定的攻击类型。本文基于 MITRE ATT&CK[1]知识库调研发现, 恶意利用标准应用层协议被越来越多的 APT 组织所采用, DNS 是被恶意利用较多的标准协议之一。进一步统计 2014—2019 年间恶意利用 DNS 完成数据窃取(后文简称“DNS 窃密”)的攻击案例情况如图 1, 统计结果表明 DNS 窃密攻击事件逐年增多、上升趋势明显。

2014 年, ESET 公开了有关 Linux/Ebury 后门软件[2]的技术分析报告, 详细分析了该后门软件中所实现的 DNS 窃密技术细节。DNS 窃密技术逐渐被 APT34(又名 OilRig)、APT18(又名 Wekby)、APT32、APT41、FIN7 等 APT 组织所采用, 整理 2014–2019 年间部分代表性 DNS 窃密攻击案例见表 1。其中, APT34 是广泛使用 DNS 窃密技术的代表性 APT 组织, 安全人员综合该组织相关攻击活动所使用的基础设施信息, 确定 APT34 代表伊朗政府展开网络攻击活动。国内外安全团队持续追踪并公开了多起由 APT34 组织、实施的攻击活动, 代表性的攻击样本有 Helminth 、 ISMAgent 、 BONDUPDATER 、 QUADAGENT、Glimpse 等, 这些攻击样本中均实现了 DNS 窃密技术。2019 年, Unit42 公开的一篇专题技术分享[3], 综合整理了 APT34 的代表性攻击样本及其 DNS 窃密技术的实现细节。

表 1 2014—2019 年间公开报告中的 DNS 窃密概况
Table 1 Overview of DNS-based exfiltration in public reports from 2014 to 2019

序号	恶意样本名称	关联威胁组织	DNS 窃密概述
1	Linux/Ebury[2]	——	通过 DNS 窃密方式将截获的敏感内容, 发送到指定 IP 的远程服务器
2	Pisloader[4]	APT18	恶意利用 DNS 隐蔽泄露受害主机系统、驱动等信息
3	Helminth[5]	APT34	恶意利用 DNS 泄露击键记录、剪贴板内容等
4	Glimpse[6]	APT34	恶意使用 DNS 窃取受害主机的指定数据内容
5	ISMAgent[7]	APT34	恶意利用 DNS 上传控制命令的执行结果、进行数据泄露
6	Cobian RAT[8]	Cobalt Group	恶意利用 DNS 回传目标数据
7	Matroyshka[9]	CopyKittens	恶意利用 DNS 完成数据窃取
8	Denis[10]	APT32	恶意利用 DNS 隐蔽泄露已收集到的主机信息、敏感数据等
9	ISMDoor[11]	Greenbug	恶意利用 DNS 泄露已收集到的数据, 以及回传控制命令执行结果
10	POWERSOURCE[12]	FIN7	恶意利用 DNS 完成关键信息内容的回传

综合上述调研和分析结果, 本文选定具备较高研究价值的 DNS 窃密攻击为突破口, 深入分析可用数据集紧缺、完备度不足的痛点问题。网络安全领域的特殊性, 使得可用 DNS 窃密数据集紧缺的问题尤为突出。从 AI 模型用户(后文简称“用户”)的角度出发, 大多面临一个共同的挑战: 难以大批量获得一份高质量的数据集, 来支撑 AI 模型的训练和测试。

针对 AI 模型训练阶段面临的数据集问题, 本文明确了攻击机理在数据生成中的重要作用, 提出基于攻击 TTPs(Tactics, Techniques, Procedures)的数据自动生成及应用方法。首先, 基于大量公开的、内容详实的案例分析报告, 参考开源项目和已有技术积累, 梳理得到 DNS 窃密攻击 TTPs。然后, 以攻击 TTPs 为理论基础, 设计并实现 DNS 窃密流量数据自动生成系统。该系统可以生成大规模、高度逼真、完备度可调的 DNS 窃密流量数据(后文简称“生成数据”)。最后, 将生成的综合流量数据集应用于检测模型的训练阶段, 有效解决可用数据规模受限、完备度不足的瓶颈问题, 从而有效提升 AI 检测模型的性能。评估实验结果表明: 本文提出的基于攻击 TTPs 的流量数据自动生成及应用方法是有效的、切实可行的; 生成数据训练所得 AI 模型性能良好, 可以检

测真实的 DNS 窃密攻击。

综上所述, 总结本文贡献如下:

(1) 参考大量 DNS 窃密攻击案例报告、开源工具分析等, 梳理总结了 DNS 窃密攻击的 TTPs 及其关键技术。

(2) 首次提出基于攻击 TTPs 的流量数据自动生成及应用方法, 不仅可以满足 AI 模型训练阶段对海量数据的需求, 还能够有效提升数据集的完备度。

(3) 设计并实现了 MalDNS 系统, 该系统不仅能高度还原已有案例报告中的 DNS 窃密攻击, 还能进行原理范围内的预测生成(MalDNS 拟开源到 Gitee 和 GitHub 平台)。

(4) 针对生成的 DNS 窃密流量数据的实验评估围绕有效性和训练所得模型性能两个方面展开, 生成数据训练所得模型对真实攻击的检测准确率高于 99.85%, 误报率为 0。实验结果表明: 生成的 DNS 窃密流量数据是有效的, 而且完备度得到提升的数据集可以有效提升训练所得模型性能。

2 相关工作

国内外研究人员在流量生成领域开展了大量研究, 但已有工作[13–15]的研究侧重点各不相同, 大多服务于网络设备模拟研究、流处理系统和设备的性能评估等目标。从 AI 模型训练对数据集的需求特点分析, 流量生成的已有研究工作并不能有效提升数据集规模和质量, 即与本文工作目标不一致。本文是首次面向 AI 模型训练需求而展开攻击流量数据自动生成方面的研究工作, 因此本章将以用户所用数据集的来源分类展开介绍。

• 源于真实网络的公开流量数据集

在实际的应用过程中, 若拥有或易于储备大规模、高质量的真实攻击流量数据, 必然可以作为 AI 模型训练阶段的可用数据集, 然而并未发现专门针对 DNS 窃密攻击的数据集。深入调研发现, AI 模型用户普遍面临这一挑战, 即难以获取大规模、有效的、高质量的数据集, 来支撑 AI 模型的训练和测试。

参考 DNS 相关的检测研究工作, 整理了 5 个常用公开数据集见表 2。其中, CTU-13、ISOT 和 ISCX 已被广泛应用于学术研究中的模型训练阶段。深入分析发现这类数据集的问题主要有:

(1) 大型公开数据集通常为综合型攻击数据, 然而 DNS 窃密流量数据规模极小, 不足以支撑 AI 模型的训练和测试; 若使用小样本数据进行训练, 会导致模型过拟合问题。

(2) 数据集更新时间滞后严重, 陈旧数据对应的原攻击样本甚至已失活, 导致数据失去了其主要价值, 而且这类陈旧的数据无法跟上攻击技术的迭代发展。

(3) 这类数据主要来源于专业安全团队对网络攻击的持续追踪和分析, 由于团队利益、企业核心竞争力、用户隐私等因素, 这类真实的攻击流量数据极少会公开分享, 致使普通用户难以大批量、公开获取。

表 2 常见的 DNS 相关攻击流量数据集概况

Table 2 Overview of publicly available datasets related to DNS-based attacks

数据集	概要说明	规模(GB)	最后更新
Bot-DAD	由印度塔帕大学(Thapar University)的研究人员收集, 2016 年 4 月至 5 月随机 10 天中超过 4000 个活跃用户的校园 DNS 网络流量, 格式为 pcap 流量文件	9.9	2019.5
CTU-13	捷克理工大学捕获的僵尸网络流量数据, 包含 13 个不同僵尸网络样本的流量数据, 格式为 pcap 流量文件	74.2	2014
ISOT	包含 ISOT Botnet Dataset 和 ISOT HTTP Botnet Dataset 两个数据集, 格式为 pcap 流量文件	10.9	2017.6
ISCX	僵尸网络 pcap 流量数据集, 包含使用不同类型通信协议的恶意软件流量数据	13.8	2016.1
KDD Cup 1999	KDD 竞赛在 1999 年举行时采用的数据集, 主要内容是网络流量及主机行为数据	1.1	1999.10

除此之外, 也有研究人员提出可以基于真实流量进行审计与分析, 使用已有工具标记攻击流量数据。如彭丹等[16]提出了一种整合不同工具进行网络攻击数据标记的方法, 主要思想是从教育网骨干网络的核心路由器采集原始流量数据, 使用入侵检测工具、告警分析方法, 对流量数据进行分析并标记攻击流量, 从而形成攻击流量数据集。这类攻击数据虽然来源于真实环境, 但数据集的质量极其依赖系统部署位置、提取和标记规则、数据采集周期长短等因素; 此外, 源于真实网络场景的流量数据, 用户隐私问题已经成为数据共享、流通的主要阻力之一。

• 复现攻击样本和开源工具的流量数据

普通研究人员在无法获取到真实攻击数据的情况下, 为了回避没有数据集可用的问题, 学术研究工作中通常会自行构建临时数据集作为替代方案。整理近几年高度相关的检测类研究工作所用数据集

情况如表 3; 由表 3 分析发现常见的构建方法是: 捕获某个网络的日常 DNS 流量作为背景(良性)流量; 复现某几个特定攻击样本或开源工具, 捕获其流量数据作为数据集的恶意部分。

表 3 近几年高度相关研究工作所用数据集概况

Table 3 Overview of datasets used in highly relevant researches

近年工作	刊出来源	数据集概要说明	公开状态
Naotake Ishikura 等[17]	2020, ICIN	良性流量: 作者所在(大阪府立大学)实验室 31 天的 DNS 流量 恶意流量: 自行搭建 DNS 隧道场景, 捕获 Dnscat2 隧道工具的流量	未公开
Jawad Ahmed 等[18]	2020, TNSM	良性流量: 研究机构、合作公司网络 14 天的日常流量 恶意流量: DET 和 Iodine 两个开源工具的流量, 以及 17 条真实的恶意 DNS 查询记录(公开报告)	已公开
RiChard Preston[19]	2019, HST	良性流量: 某大学网络经过简单过滤后的日常 DNS 流量数据 恶意流量: DNS 隧道工具生成的流量, Denis、FrameworkPOS 攻击样本流量	未公开
Chang Liu 等[20]	2019, IPCCC	良性流量: 请求解析 top 1000000 正常域名的 DNS 流量数据约 50W 恶意流量: 开源工具 Iodine、Dns2tcp、Dnscat2、OzymanDNS、ReverseDNShell 的流量, 近 50W	未公开
Nadler Asaf 等[21]	2019, CS	良性流量: Akamai 科技某子网一星期的日常 DNS 流量 恶意流量: 开源工具 Iodine、Dns2Tcp 的 DNS 隧道流量, 以及复现 FrameworkPOS、Win32.Denis 的流量数据	未公开

由表 3 分析发现, 研究人员在论文工作中自行构建的实验数据集基本不公开, 仅简单描述数据集的来源和组成部分; 这非常不利于其他研究人员对已有成果的性能验证和进一步探讨。从数据质量的角度分析, 自行构建过程中所使用的开源工具种类和真实攻击样本极少, 例如表 3 中就仅仅涉及 Denis、FrameworkPOS 两个真实攻击样本。对普通用户而言, 自身不具备专业安全团队的技术积累和资源支撑, 不仅存在真实攻击样本批量获取的困难, 复现运行真实攻击样本也极具挑战, 因为很多攻击样本中广泛使用了诸如对抗沙箱、反虚拟化环境等对抗技术。以表 3 为例的众多研究工作中, 数据集的恶意流量则主要来源于开源工具, 然而这些开源工具与真实攻击差距较大。以 DET[22](Data Exfiltration Toolkit)为例, 该工具实现的 DNS 窃密方式存在很多缺陷, 例如窃密成功率低、DNS 请求方式不符合实际攻击趋势等。因此, 自行构建数据集中的恶意流量数据局限性明显, 数据集的完备度受限于恶意样本、开源工具的多样性。

与此同时, 基于已有攻击样本, 有研究人员提出采用自动化样本分析和执行技术, 来获取目标攻击样本的流量数据。如陈家浩等[23]提出了一种基于 Python 符号执行的自动化网络攻击流量获取方法; 该方法针对当前网络上可获取的 Python 网络攻击脚本, 采用 Python 符号执行技术和强制执行技术, 来自动化获取输入脚本对应的攻击流量数据, 旨在解决大量攻击场景复现困难的问题。值得注意的是, 该方法需要事先收集 Python 攻击脚本, 然而, 获取真实攻击的代码文件是极其困难的, 很多攻击代码甚至永远也不会公开, 即并没有实际解决普通用户获取困难的问题。

- **其他领域类似问题的研究工作**

AI 模型训练数据不足的问题在其他领域同样存在, 本文也简要梳理了其他领域的类似研究工作, 主要有数据增强 (Data Augmentation) 和 GAN(Generative Adversarial Networks)方法生成。

数据增强技术是基于有限数据产生更多的等价数据来人工扩展训练数据集的技术, 被认为是克服训练数据不足的有效手段。目前针对图片的数据增强技术, 通过旋转、缩放、裁剪等简单操作即可完成, 且增强的数据验证是显而易见的。例如, 一张“猫”的图片通过旋转、缩放后, 形成的图片仍然是一只“猫”, 则可以认为增强的数据是有效; 即通过旋转、缩放等操作, 可同时完成数据规模增大和标签化工作。更高级的图片数据增强技术研究也取得了一些成果, 如 Zhu 等[24]使用条件 GAN 可以将夏季风景照转换为对应的冬季风景照。Luan 等[25]提出了一种基于深度学习的照片风格转化方法, 能一定程度扩展图片数据。若应用上述方法来增强 DNS 窃密流量数据, 首先需要满足增强保证性假设的要求, 即增强后的 DNS 窃密数据仍然可以完成既定攻击, 且符合 DNS 窃密的基本攻击原理。对于由 0、1 表示的二进制数据, 自动化修改并验证数据、程序的功能, 这类技术目前尚待突破; 因此, 数据增强暂不适用于增强 DNS 窃密流量数据。

2014 年, Goodfellow 等[26]首次提出 GAN, 主要

由生成器(Generator)和判别器(Discriminator)组成; 生成器的目标是欺骗判别器, 使其不能正确分辨生成数据和真实数据。随着 GAN 在样本对抗中的出色表现, 也有研究人员尝试利用 GAN 网络来解决数据集紧缺这一问题。Lee 等[27]提出了使用 WGAN 来生成自相似的恶意流量, 解决训练数据不平衡的问题。深入分析发现, 若应用GAN网络生成DNS窃密数据, 首先需要积累一定规模的原始 DNS 窃密流量; 其次, GAN 生成方法的关键在于生成器和判别器的设计, 生成数据质量由生成器的泛化能力直接决定, 最终难点是设计一个能够完成攻击目的和数据有效性验证的判别器。目前, 依据参数矩阵中的各项数值来完成自动化攻击目的和攻击原理验证, 是当前技术无法完成的。

3 DNS 窃密攻击 TTPs

在国内外众多安全专家、团队的共同努力下, 公开的、内容详实的案例分析报告已经初具规模。参考大量 DNS 窃密攻击案例分析报告, 本章将梳理DNS窃密攻击TTPs, 作为DNS窃密流量数据自动生成框架的理论支撑, 也可供相关研究人员作为技术参考。

3.1 范围界定

如图 2 所示, 本文所研究的 DNS 窃密是指: 基于标准 DNS 协议自行设计规则, 将目标主机(受害者)上的特定目标内容(如高价值文件、截获的敏感内容、命令执行结果等)传送到攻击者控制的服务端, 且通信流量基本符合 DNS 协议规范。

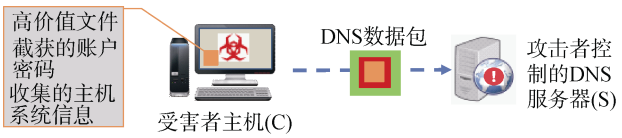

图 2 DNS 窃密概况

Figure 2 Overview of DNS-based exfiltration

在实际的攻击活动中, 攻击者通常综合使用多种攻击技术来完成一系列攻击动作, DNS 窃密技术在各种攻击活动中也有不同体现。参考 ATT&CK 知识库对数据泄露(Exfiltration)的分类, 典型的 DNS 窃密主要包含:

(1) 独立于命令与控制信道(后文简称“C&C 信道”), 特意为执行 DNS 窃密而制作的恶意代码段或工具, 如 APT34-Glimpse、DNSExfiltrator 等;

(2) 直接使用构建完成的 DNS C&C 信道(基于 DNS 构建的 C&C 信道)执行 DNS 窃密任务, 例如使用 DNS C&C 信道回传收集到的主机信息、截获的敏感信息、控制命令执行结果等, 都属于本文所研究的“DNS 窃密”范畴。值得注意的是, 通过 DNS 响应特定 IP 指代特定攻击命令、受害主机发送特定 DNS 请求更新 Payload 等, 暂不属于本文所指 DNS 窃密技术的范畴。

(3) 已成功建立 DNS 隧道后, DNS 隧道客户端向服务端回传内容的部分, 也属于本文所讨论的 DNS 窃密范畴。同样的, 由 DNS 隧道服务端向客户端传送数据的部分, 暂不属于本文所指 DNS 窃密的范围。

综合大量 DNS 窃密攻击案例分析报告可知, 三类 DNS 窃密方式都需要基于标准 DNS 协议构建自定义的数据传输规则, 因 DNS 协议规范的限制, 三类 DNS 窃密方式需要将欲窃取内容嵌入 DNS 请求的域名中。因此, 上述三类 DNS 窃密形式的基本原理是一致的, 本文将以第一类 DNS 窃密实现形式为例, 来阐述 DNS 窃密攻击 TTPs。为了便于后续的分析讨论, 本文特别指定以下说法:

目标内容: 特别指代攻击者拟窃取的内容, 可以是文件内容、控制命令执行结果、截获或收集的用户敏感信息等。

附属信息: 特别指代与目标内容相关的其他信息, 如目标文件名标识、归属主机系统标识、目标内容校验信息等。

窃密数据: 按预定义的编码转换方案, 处理目标内容和附属信息后, 最终形成的需要传送的数据, 本文用“窃密数据”来特别指代。

特制域名: 特别指代由攻击者设计的、具有特定组成格式的域名, 一般包含窃密子域和攻击者预置的二级域名(Second Level Domain, SLD)。

3.2 DNS 窃密机理

DNS 窃密攻击中, 攻击者会事先注册至少一个二级域名用于执行 DNS 窃密任务, 并且配置攻击者控制的 DNS 服务器(后文简称为“窃密服务端”)作为权威服务器来解析该 SLD 及其所有子域。例如, 攻击者可配置由攻击者控制的 DNS 服务器, 来解析 maldns.club 及其所有子域; 那么经过公共 DNS 系统的解析查询机制, 所有 maldns.club 及其子域的解析请求最终会到达窃密服务端。

DNS 窃密攻击以成功执行窃密任务为基本要求, 目标内容需要通过 DNS 解析请求传送到窃密服务端, 总结 DNS 窃密攻击的基本流程如图 3, 总结如下:

(1) 窃密客户端对目标内容、附属信息等, 按预定义的内容加工流程和方法进行处理, 从而服务于

攻击者的特定意图, 例如执行压缩以提升效率、进行加密避免明文传输等。

(2) 受限于 DNS 对标签、域名长度的限制, 窃密客户端将窃密数据分片, 并构建形成一系列携带窃密数据分片的特制域名。

(3) 窃密客户端发起 DNS 请求, 依次解析特制域名, 即通过 DNS 实际传送窃密数据分片。经过 DNS 请求解析流程, 窃密 DNS 请求数据包会最终到达由攻击者控制的窃密服务端。

(4) 配置为全时段工作的窃密服务端, 从所有 DNS 解析请求中, 按预定义规则识别并筛选出窃密 DNS 请求; 除此之外, 窃密服务端通常还会进行策略响应。

(5) 窃密服务端依据预定义的特制域名结构, 从窃密 DNS 请求的域名中, 分别提取和暂存窃密数据分片、辅助信息等。当窃密传送完成后, 依据辅助信息将所有窃密数据分片重组, 得到完整的窃密数据。

(6) 窃密服务端依据客户端使用的内容加工处理方法和流程, 对窃密数据进行相逆的处理和转换, 从而恢复得到目标内容及其附属信息, 进行简单校验后, 以存储到服务端本地等形式反馈给攻击者。

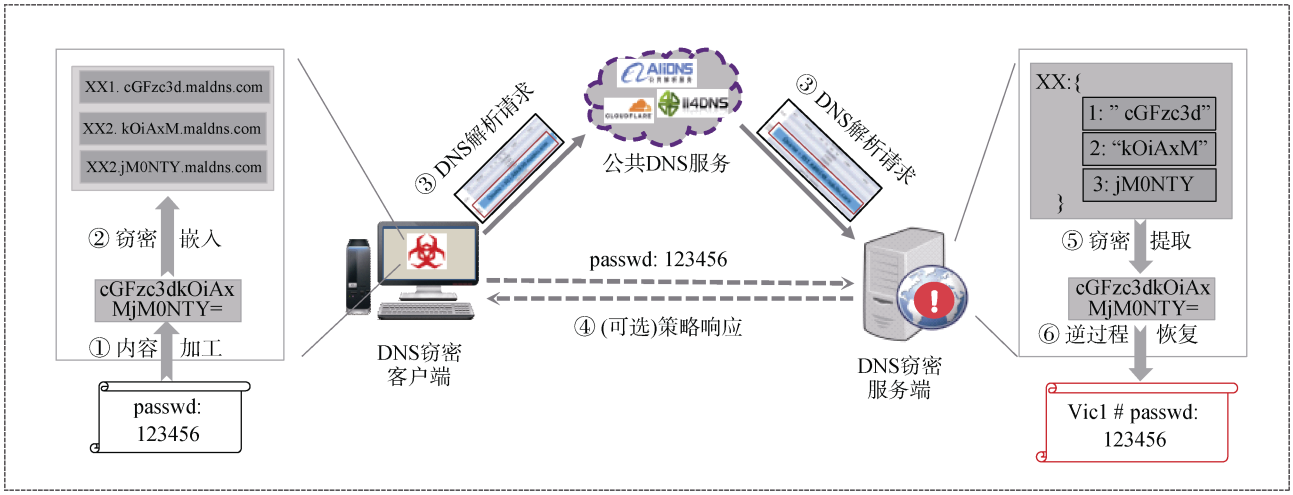

图 3 DNS 窃密的基本流程

Figure 3 The principle of DNS-based exfiltration

3.3 DNS 窃密关键技术

- **数据嵌入与恢复技术**

综合前文对 DNS 窃密的界定和 DNS 窃密机理的研究, DNS 窃密实际是通过 DNS 请求数据包来完成的。参考 DNS 协议规定的请求报文格式详情, DNS 查询请求主要包含 DNS 首部和查询问题区域两个部分。DNS 首部包含 6 个字段, 每个字段与 DNS 数据报文紧密相关, 不同取值对应在 DNS 协议规范内都具有特定含义。查询问题区域中的 QCLASS 明确 DNS 查询请求的地址类型, 通常取值为 1, 代表互联网地址; 而 QTYPE 的取值则对应 DNS 查询请求的资源类型, 例如 QTYPE 字段取值为 1 则表示请求解析 A 记录。DNS 查询请求数据报文中的 QNAME 部分通常是请求解析的域名, 三类典型 DNS 窃密攻击形式的窃密数据嵌入位置均是 QNAME 部分, 这是因为 QNAME 部分是嵌入窃密数据分片的最佳位置。

与此同时, DNS 协议规定域名的标签最大长度为 63 个 ASCII 字符, 完整域名的最大长度为 253 个 ASCII 字符。受限于标签和域名长度的限制, 窃密数据通常需要进行分片后, 分别通过一系列的 DNS 查询请求序列完成传送。

为了能够识别、筛选出窃密 DNS 请求, 并按序恢复各分片数据, 每个窃密 DNS 请求的域名需要包含必要的辅助字符串, 常用辅助字符串比如分片序号、分片归属标识信息等。然后将窃密数据分片、辅助字符串等, 按照预定义的方式和结构构造, 从而形成携带窃密数据分片的特制域名, 整理部分代表性攻击案例中所使用的特制域名结构如表 4。如表 4 所示, 攻击案例中实际使用的特制域名结构特征因不同的攻击样本实现而有所差异, 例如表 4 中的 8 个攻击案例中所使用的特质域名结构均不同。对比分析发现, 不同特制域名结构都包含“分片序号”、“标识符”等关键组成部分, 不同点大多只是次要字符串的增改以及各组成部分所处位置的调整。

表 4 攻击案例中常见的 DNS 窃密特制域名结构

Table 4 Common structure of crafted subdomains in DNS-based exfiltration campaigns

序号	样本名称	DNS 窃密特制域名结构
1	Helminth	00<系统标识符><文件名><分片序号><随机数><窃密数据>.go0gIe.com
2	Conmmunicator Dot	<随机数>.IDID.<系统标识符>.<分片序号>.<总包数>.<窃密数据>.<文件名>.newuser.tk
3	ISMAgent	<窃密数据><分片序号>.d.<GUID>.ntpupdateserver.com
4	BONUPDATER	<随机数>4<分片序号><系统标识符>B007.<窃密数据>.poison-frog.club
5	QUADAGENT	<窃密数据>.<分片序号>.acrobatvesrify.com
6	APT34-Glimpse	<GUID><数据包标识>2<随机字符>C<Data 偏移指示><操作类型偏移量指示>. myleftheart.com
7	APT34-RDAT	<分片序号><AES 密钥><窃密数据>.tacsent.com
8	DET	<任务标识><窃密数据>.<SLD>

- **编解码转换技术**

在 DNS 窃密攻击案例中, 内容编解码转换技术被广泛使用, 主要是针对目标内容和附属信息进行内容形式或格式的转换。编解码转换是指按预定义的编解码方法和流程来转换目标内容的呈现格式或形式; 并且可以按照对应相逆过程和方法解码得到原始内容。显然, 上述内容编解码转换方法包含标准的 Base64、Base32 等方法, 攻击者也可以自定义内容转换和方法, 例如按一定流程组合使用常用编码方法。在实际攻击活动中, 内容编解码转换的设计和技术实现, 通常与域名语法规范、攻击者意图紧密相关。

内容编解码转换的首要目标是使得转换后的内容符合域名语法规范。DNS 协议规定, 域名由层级结构的多级标签构成, 只能由字母、数字和连字符组成, 且开头和结尾只能为字母或数字。然而, 攻击者欲窃取的目标内容可能存在非法字符, 显然不能完全符合域名语法。因此, 攻击者必须对目标内容执行内容编解码转换, 使所得窃密数据基本符合域名语法要求, 最简单方法可直接使用标准 Base64URL(基于 Base64 编码方法的改进, 将域名中不能出现的“+”、“/”分别替换为“-”、“_”, 并去处尾部的“=”)、Base32 等方法。

除此之外, 内容编解码转换还可能服务于攻击者的更多意图。例如, 由于 DNS 协议设计为明文传输, 较多攻击者在实际的攻击活动中, 还会对目标内容进行加密, 从而避免明文传输、逃避内容检测。

- **DNS 窃密传送技术**

DNS 窃密传送技术直接控制攻击活动的网络流量表现, 主要是依据特制域名构建 DNS 请求数据包, 并直接控制 DNS 请求相关的其他参数。

攻击者在设计和实现 DNS 窃密传送技术时, 首先需要重点规避各级 DNS 服务器的缓存机制。一般情况下, 经过编解码转换处理后的窃密数据片, 出现 2 个以上完全相同数据分片的可能性较小。然而, 在实际的攻击活动中, 通常还会在特制域名的固定位置加入随机的冗余部分, 从而确保每个窃密 DNS 查询请求都能最终到达窃密服务端。

DNS 窃密传送技术直接影响窃密流量数据的多个方面, 而且不同攻击者的关注点和实际需求会有所差异, 常见的包括窃密 DNS 请求间隔、DNS 数据包构建方法等。以 DNS 请求发起方式为例, 攻击者通常会依据受害主机系统特点来选取 DNS 请求发起方式; 例如 ISMAgent 攻击样本是调用系统 API 的 DnsQuery_A 函数来发出 DNS 解析请求, 而 QUADAGENT 攻击样本则使用 nslookup.exe 可执行程序来发出窃密 DNS 请求数据包。

- **策略响应技术**

依据 DNS 窃密的基本原理分析, 针对窃密 DNS 请求的响应并不是执行窃密传送所必须的, 但若大量的 DNS 请求无响应则极为异常, 为了对抗检测攻击者通常会按既定策略进行响应。

策略响应技术就是攻击者处理大量窃密 DNS 查询请求时, 服务于攻击活动而制定的响应策略及其实现技术。避免大量 DNS 查询请求无响应的异常情况是策略响应技术的关键因素之一, 但具体的响应策略会因不同攻击者及其目的不同而差异明显。以针对 A 记录的响应策略为例, 常见的响应策略有:

(1) 基础伪装类: 响应某一固定 IP、随机 IP 或规律变化 IP, 而响应的 IP 没用指定用途。这是攻击者选取的一种较为简单的响应策略, 可以解决大量 DNS 查询请求无响应的这一明显异常问题。比如 APT34-Helminth 在执行窃密任务时就响应固定 IP “172.16.107.128”。

(2) 强化伪装类: 出于伪装目的, 攻击者预置一个或多个具有欺骗性的响应结果; 例如 Helminth 样本中, 对于窃密 DNS 请求的响应可以伪装成“google.com”服务器的 IP 之一。

(3) 稳定增强类: 在对应响应中携带有利于实现更稳定传送的信息; 例如携带已成功收到的窃密数据分片序号, 从而通知客户端已经传送成功的窃密数据分片。由于这种方式需要攻击者设计完整的重传机制, 因此目前只在极少数攻击案例中被采用。

DNS 窃密攻击的关键技术点是攻击者在实现过程中难以避免的, 或者放弃使用关键技术点需要付出巨大代价, 例如牺牲窃密传送效率、传送成功率等, 因而防御方可以基于这些关键技术要点制定更加有效的检测、防御措施。基于上述攻击 TTPs 分析, 针对 DNS 窃密攻击的防御思路可以从以下几个方面开展:

(1) 特制域名: 携带窃密数据分片的一系列特制域名具备固定的结构特征; 数据分片序号、各类标识呈现出的规律等。

(2) DNS 窃密传送: 受限于 DNS 包的大小限制, 存在大量窃密所用 SLD 的子域的 DNS 解析请求, 且窃密传送间隔、DNS 请求和响应主机间的固定规律等, 均是 DNS 窃密技术实现在网络流量上的体现。

4 数据自动生成的框架设计与实现

4.1 基于攻击 TTPs 的数据生成及应用设计

众所周知, AI 模型训练阶段需要足够规模的流量数据来支撑模型训练; 而且训练数据集的质量越高, 训练所得模型性能越好。本文的首要目标是解决 AI 模型训练阶段面临的数据集紧缺、完备度不足的问题, 即面向 AI 模型训练的流量数据自动生成的主要目标是:

(1) 突破 DNS 窃密流量数据的规模局限性, 解决数据有效性验证的难题, 为用户生成大规模、有效的 DNS 窃密流量数据。

(2) 广泛覆盖 DNS 窃密攻击案例, 拓展未知变体空间, 从而有效提升生成流量数据集的完备度。

本文从攻击者的角度出发, 明确了攻击原理的重要作用, 提出基于攻击 TTPs 的流量数据自动生成及应用方案, 具体设计见图 4。

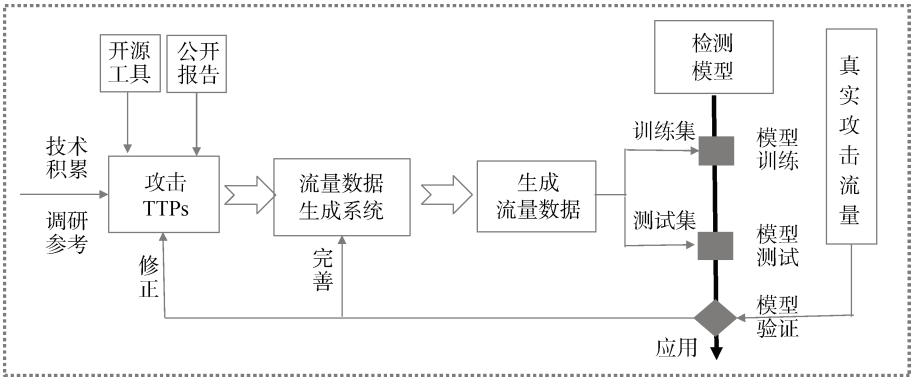

图 4 基于攻击 TTPs 的流量数据自动生成及应用方案

Figure 4 Automatic data generation and application scheme based on attacks' TTPs

攻击 TTPs 作为样本数据自动生成的理论基础, 是生成流量数据有效性、完备度的重要保证。在国内外安全研究人员的共同努力下, 曝光攻击案例的分析报告已经初具规模, 内容详实的分析报告基本还原了攻击样本的技术实现、攻击策略等。基于已有分析报告和开源项目, 可以梳理得到 DNS 窃密的攻击 TTPs。

在攻击 TTPs 的指导下, 设计并实现 DNS 窃密流量数据生成系统。该系统通过定制配置文件的方式实现高扩展性, 即定制化生成大量逼真的、完备度可调的 DNS 窃密流量数据; 而且生成流量数据的有效性可以依据窃密任务执行结果进行直接验证。

除此之外, 通过跟进最新的攻击案例报告, 已有攻击 TTPs 可以得到持续修改和完善, 保证专家知识体系的完备度和时效性。持续更新、完善的攻击 TTPs, 可以指导生成系统拓展实现最新攻击技术, 从而保证生成数据的完备度和时效性。

生成流量数据完成有效性验证后, 即可作为训练集和测试集, 应用于 AI 模型的训练阶段。为了检验训练所得模型的实际检测性能, 辅以少量真实攻击数据来验证模型的真实检测性能。与此同时, 检测结果反馈也可以指导生成系统的持续完善, 进一步提高生成数据的质量。

本文设计的流量数据自动生成及应用方案, 不仅满足了 AI 模型训练对目标类型攻击流量数据的大规模需求, 还可以通过完善数据集完备度来有效提升检测模型性能。区别于其他流量生成方法, 所述方法将攻击 TTPs 作为理论支撑, 保证了流量数据生成

框架的合理性; 实现的流量数据自动生成系统需要验证能否实际完成既定攻击任务, 因而生成流量数据与真实攻击流量数据之间的差异性不可区分。

4.2 MalDNS: DNS 窃密流量自动生成框架

遵循 4.1 节的数据自动生成方案, 基于 DNS 窃密攻击 TTPs 设计 MalDNS 系统, 该系统实现了完整的 DNS 窃密框架, 用于大规模生成 DNS 窃密流量数据; 生成流量数据的有效性易于直接验证, 而且其完备度可以通过配置文件进行调控。得益于可控环境下的完整窃密框架, MalDNS 生成流量数据的过程可以等效为参数可调的真实攻击, 从而保证了生成流量数据与真实攻击流量数据之间的差异性不可区分。

MalDNS 系统设计如图 5, 遵循 DNS 窃密攻击的基本流程, 包含 DNS 窃密客户端和服务端, 客户端对应真实攻击中的受控主机, 而服务端则对应由攻击者控制的 DNS 服务器。如图 5 所示, MalDNS 系统执行 DNS 窃密任务的主要流程可概括为:

(1) DNS 窃密客户端按配置文件描述的流程对目标内容进行加工处理, 形成窃密数据; 然后依据特制域名相关参数, 将窃密数据分片、对应辅助字符串等嵌入特制域名; 最后调用指定方式构建 DNS 请求数据包并发出。

(2) 从客户端发出的窃密 DNS 请求数据包, 遵循常规的 DNS 解析流程, 最终会到达窃密服务端。由 DNS 窃密攻击 TTPs 可知, DNS 窃密传送使用通用 DNS 解析服务而无需特别实现, 调用客户端系统支持的 DNS 解析请求方式即可; 因此, MalDNS 系统对窃密 DNS 解析不做定制化设计。

(3) 窃密服务端将识别并筛选出窃密 DNS 请求数据包, 从 QNAME 中提取并暂存窃密数据分片、辅助字符串等。当窃密任务传送完成后, 服务端则依据辅助信息重组所有窃密数据分片, 得到完整的窃密数据。最后依照与客户端相逆的内容转换处理流程恢复目标内容, 并以指定形式反馈或存储。

(4) 服务端对窃密 DNS 解析请求执行策略响应, 如避免大量 DNS 请求无响应的异常、辅助实现更稳定的 DNS 窃密传送等。

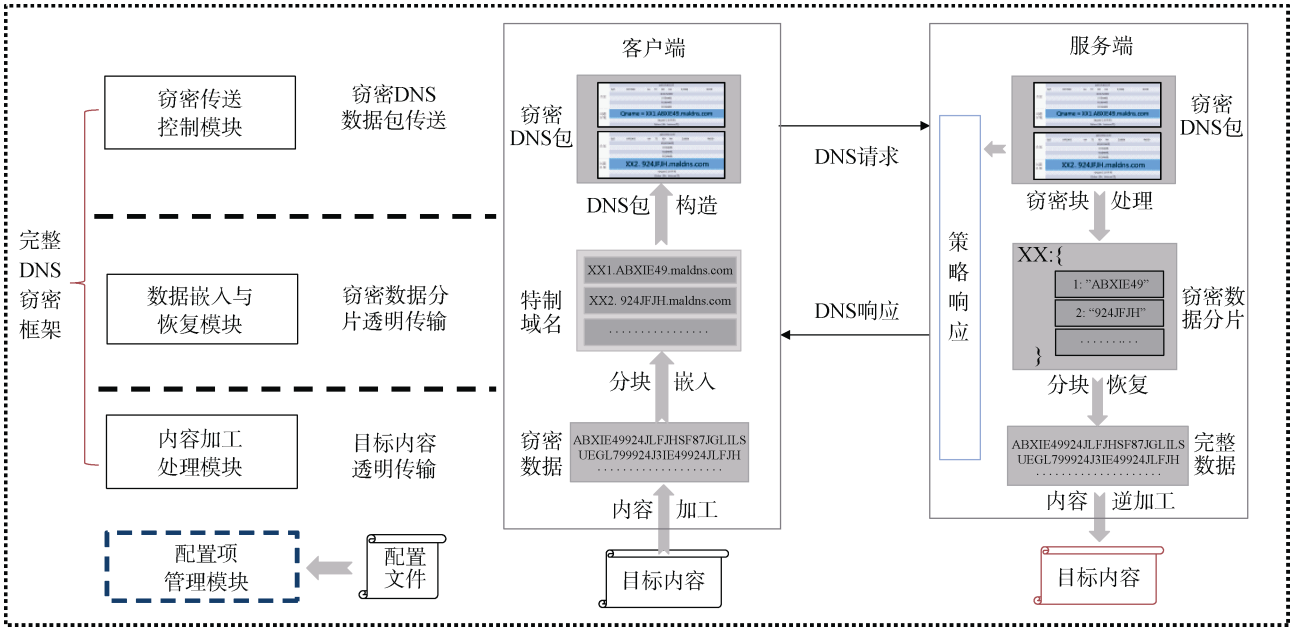

图 5 MalDNS 系统框架设计

Figure 5 The framework of MalDNS

表 5 DNS 窃密关键技术矩阵

Table 5 Key technologies of DNS-based exfiltration

内容加工	数据嵌入/恢复	DNS 窃密传送	*策略响应
编解码转换	辅助子串	请求频率	无响应
*压缩	子域结构	记录类型	基础类响应
*加密	长度限制	包构建方法	伪装类响应
	*握手子域		稳定传输类

MalDNS 系统的配置项参考 DNS 窃密攻击 TTPs 进行设计, 旨在提升生成流量数据的多样性和完备度。结合生成流量数据的主要影响因素, MalDNS 系统拟覆盖 DNS 窃密的关键技术矩阵如表 5, 表中的*项为非必须实现的技术要点。具体的, 针对表中每个技术要点设计一组配置项, 通过编辑不同配置项的值来还原该技术要点在不同攻击案例中的实现。例

如, 不同的攻击案例使用的编码方法不同, 常见的有 Base16、Base32、Base64URL 及其组合使用等。

参照表 5 设计的配置近 60 项, 以 json 格式存储为配置文件; 而且配置文件是直接面向用户的, 即用户可以在攻击原理范围内对各配置项进行定制化编辑, 如针对性修改加密、编码相关的配置如下:

```
    "content": {
        "compress_or_not": 1,
    "encrypt_method": {
            "active": 1,
            "method": "AES",
            "key": "THISISMALDNSKEY"
        },
        "encode_method": {
            "active": 1,
            "method": "BASE32"
    }
    }
    "domain_structure": {
        "SLD": "ntpupdateserver.com",
        "format": "<target_content>.
<seq_number>.d. <system_ID>",
        "max_label": 63
    }
```

示例的配置项组合将配置 MalDNS 系统在编解码目标内容时, 使用 AES 加解密方法, 且对称密钥为“THISISMALDNSKEY”; 启用的编码方法则是 Base32。关于特制域名的配置将窃密所用二级域名设置为“ntpupdateserver.com”, 特制域名最大标签长度为 63 个 ASCII 字符, 特制域名子域的组成格式为“ <target_content>.<seq_number>.d.<system_ID><pack_type>”(其中, target_content 将嵌入窃密数据分片, seq_number 则是分片序号的嵌入位置)。

面向用户的配置项设计是 MalDNS 系统生成流量数据多样性的重要方式, 可以实现两类定制化流量数据生成:

(1) 案例还原生成: 参考案例报告编辑各配置项的值, 目标是高度还原分析报告中所描述的 DNS 窃密攻击, 使得生成流量数据与真实攻击流量数据之间差异不可区分; 从而配置 MalDNS 系统生成与目标攻击案例高度相似的 DNS 窃密流量数据。

(2) 预测生成: 在 DNS 窃密攻击原理范畴内, 编辑各配置项的值来描述未来可能被攻击者使用的 DNS 窃密变体; 即配置 MalDNS 系统预测生成未知的、符合攻击原理的 DNS 窃密流量数据。

4.3 MalDNS 关键功能设计与实现

MalDNS 系统设计为 C/S 模式, 窃密客户端和服务端在对应阶段存在较大关联性, 如图 5 所示每一阶段都完成了不同的透明传输任务。因此, MalDNS 系统划分为 4 个核心功能模块, 分别是配置项管理、内容加工处理、数据嵌入与恢复、DNS 窃密传送模块, 各部分的功能及模块间协同工作情况如图 6。其中, 配置项管理部分的作用是处理面向用户的配置文件, 并转化为 MalDNS 系统参数; 而内容加工处理、数据嵌入与提取、DNS 窃密传送 3 个模块则是一个完整的 DNS 窃密框架, 按配置文件描述的方式执行 DNS 窃密任务, 从而大批量生成 DNS 窃密流量数据。

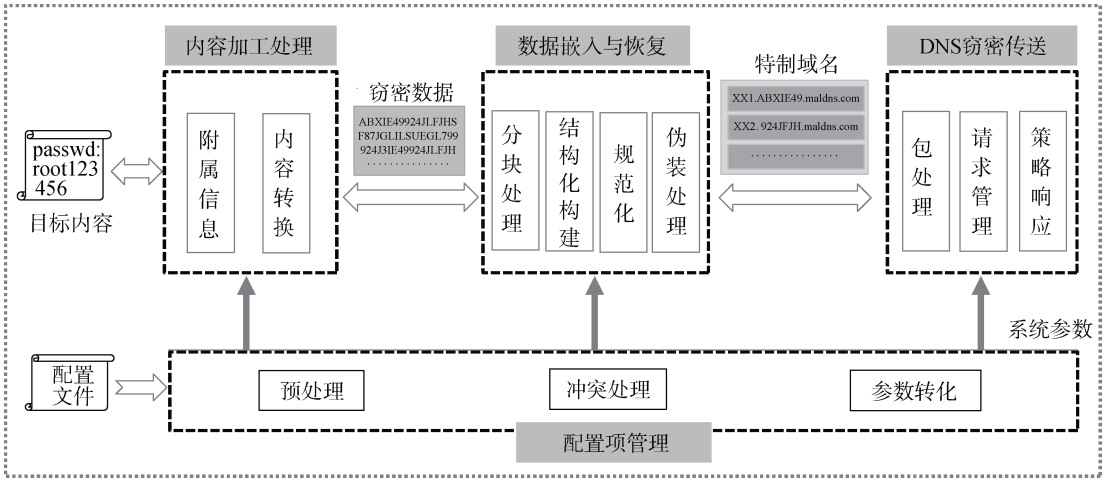

图 6 MalDNS 系统的功能模块设计

Figure 6 Design of function modules for MalDNS system

• **配置项管理**

配置项管理的主要功能是管理直接面向用户的众多配置项, 并基于各配置项的值形成定制化的 MalDNS 系统参数, 用于指导 MalDNS 系统生成所需的 DNS 窃密流量数据。配置项管理主要实现 3 个功能:

(1) 预处理: 对配置文件进行预处理, 例如读取配置项后执行分类提取、初始转化等, 即基于众多配置项的值形成初始的系统参数集合。

(2) 冲突处理: 为了便于对照攻击报告进行定制化编辑, 各配置项不可能做到完全独立, 那么存在相关性的配置项之间可能存在一些冲突的情况。冲突处理则对这些关联配置项的正确性进行必要的确认并反馈。例如, 由用户编辑配置的子域标签最大长度为 63 字符, 又指定窃密数据分片长度大于 63 字符; 在未配置多级标签的情况下, 就出现了参数冲突的情况, 因此配置项管理中的冲突管理是很有必要的。

(3) 参数转化: 确认初始参数配置的正确性以后, 配置项管理需要基于初始参数进行计算和转换, 形成可供 MalDNS 系统直接使用的参数集合, 便于生成系统在不同阶段直接调用。

• **内容处理加工**

内容加工处理的主要功能是执行目标内容与窃密数据之间的编解码和转换处理, 而且不同攻击案例使用的转换方法和流程可参照分析报告自行设计。从窃密数据传输的角度来看, 内容加工处理则是完成了目标内容的透明传输。参考 DNS 窃密攻击 TTPs, MalDNS 系统在内容加工处理阶段需要重点关注的技术要点有:

(1) 附属信息: 服务于窃密数据传送、目标内容的后续使用, 通常需要额外传送目标内容相关的一些附属信息。常见的例如: 内容校验信息可用于校验窃密数据的完整性; 而目标内容的文件名、所属受害主机标识等, 则服务于目标内容的进一步使用。为了便于窃密传送和恢复, 通常需要将目标内容、选定的附属信息、分隔符等, 按照预定规则进行结构化处理。

(2) 内容转换: 受限于 RFC 1034、1035 等 DNS 相关标准文档的规定, 或服务于特定的攻击意图, 窃密内容通常需要执行内容转换处理。常用的如压缩、加密、编解码等。其中, 编码转换在实际的攻击案例中是必不可少的, 这是因为 RFC 1034 中规定: 域名只能由数字、字母、连字符组成, 然而原始目标内容不能完全符合域名的语法规范。

值得注意的是, 窃密服务端的内容转换处理与客户端完全对应, 即遵循与客户端相逆的内容编解码和转换方法, 可以恢复得到目标内容及其附属信息。

• **数据嵌入与恢复**

数据嵌入与恢复主要完成窃密数据与窃密特制域名之间的转换, 具体的窃密数据分片嵌入、提取和重组将基于相关配置项来完成。由于窃密特制域名直观体现在 DNS 流量数据上, 其扩展性和还原能力是决定生成 DNS 流量数据多样性的重要因素之一。因此, 数据嵌入与恢复的主要功能点有:

(1) 分块处理: 参考 DNS 标准协议有关域名、标签长度的规定, 结合良性域名长度的统计分布, 将窃密数据通过多个 DNS 请求完成窃密传送; 即将窃密数据进行分块后对应编号, 然后分别嵌入到多个 DNS 请求中。

(2) 结构化处理: 为了便于服务端恢复、提取窃密数据分片及其辅助字符串, 窃密特制域名需要遵循预定义的结构进行构建, 攻击案例中常使用的特制域名如表 4。MalDNS 系统的目标之一是高度还原大量攻击案例中所采用的特制域名, 即通过相关配置项值的编辑, 能够还原各案例实际采用的特制域名结构。除此之外, MalDNS 还能通过配置项的修改执行预测生成, 即基于 DNS 窃密攻击发展趋势预测未来可能被采用的特制域名结构, 从而进一步扩展可能的变体空间。

(3) 规范化和伪装处理: 为了使携带窃密数据的特制域名基本符合域名规范, 弱化特制域名与正常域名之间的不必要差异, 需要执行必要的域名规范校验和伪装处理。例如构建的特制域名中避免出现非法字符, 将域名的数字占比调整为正常域名的统计均值等。其中, 域名规范化大多可以在设计内容编解码转换方案时同步考虑, 然后在这一阶段进行非法字符校验和反馈即可。

分析实际攻击案例发现, 分块和结构化是影响服务端提取、恢复目标内容成功率的关键因素。而规范化和伪装处理, 则是为了最大化 DNS 窃密的隐蔽性和穿透性, 有效提升逃避检测的能力, 也是实际 DNS 窃密攻击中的重要因素。

• **窃密传送控制**

窃密传送控制主要涉及窃密 DNS 请求数据包的传送, 以及服务端按预定义策略进行响应。在靶场环境下, 通过捕获窃密客户端和服务端之间的窃密 DNS 流量, 即为所需的生成流量数据。窃密传送控制阶段需要重点实现包处理、请求管理、策略响应 3 方面的功能:

(1) 包处理: 包处理在客户端主要依据窃密特制域名相关配置, 构建 DNS 解析请求数据包并发出; 相应的, 在服务端识别并筛选窃密 DNS 数据包。参考攻击案例中的包处理方法, MalDNS 不需要特别定制窃密 DNS 数据包的格式, 调用目标系统的 DNS 解析系统函数即可。

(2) 请求管理: 对窃密 DNS 请求的管理直接影响生成流量数据的多个方面, 如 DNS 窃密数据包发送频率控制、窃密传送工作时间、DNS 请求方式等。在 MalDNS 系统中, 窃密传送控制主要依据配置项进行请求管理, 还可以通过配置项进行持续补充、扩展。

(3) 策略响应: 服务于攻击者的不同意图, 需要按配置文件描述的响应策略, 对收到的窃密 DNS 解析请求执行策略响应。

5 实验评估

5.1 数据说明

本文设计和实现的 MalDNS 系统, 既能参考案例分析报告进行还原生成, 也可以在 DNS 窃密原理范畴内进行预测生成; 而且现有系统基本能够还原已有案例分析报告中呈现的 DNS 窃密攻击模式(具有相似数据结构、处理流程的, 则属于同一种 DNS 窃密模式)。

MalDNS 系统的实现了完整的 DNS 窃密框架, 即首先需要在靶场环境下分别部署 MalDNS 客户端和服务端, 按配置文件实际执行 DNS 窃密任务。用户使用 MalDNS 系统生成 DNS 窃密流量数据时, 按需定制配置文件和指定目标内容集即可; 然后捕获系统执行 DNS 窃密活动的通信流量, 即为所需的生成流量数据。其中, 定制多种 DNS 窃密模式的配置文件可以提升生成流量数据集的完备度, 主要依据拟还原生成的目标案例或预测性的攻击变体; 目标内容集(可对应实际攻击活动中拟窃取的目标内容集合)则根据流量数据规模需求自行指定, 使得生成流量数据的规模完全可控; 同时也可以规避数据集共享时的隐私问题。

本次实验所使用的目标内容集选自 Ubuntu 系统“/etc/”路径下的默认文件, 包含选定“enviroment”、“passwd”、“profile”等文档共 100 个, 文档大小共 287KB。为方便描述和区分, 由 MalDNS 系统生成的流量数据以“MDG-XXX”命名, 而且对应配置项组合以还原 XXX 窃密活动为目标。例如“MDG-Glimpse”表示对应流量数据由 MalDNS 系统 生成, 且配置文件配置项的组合以还原 APT34-Glimpse 所实现的 DNS 窃密攻击模式为目标。

由于真实 DNS 窃密攻击流量数据的紧缺, 本次实验使用的真实数据主要来源于靶场环境下的攻击场景复现。具体的, 在 APT34 Glimpse 源码泄露后, 基于收集到的 Glimpse 完整代码文件进行攻击场景复现和分析, 捕获其执行 DNS 窃密攻击活动的流量数据。与此同时, 近几年相关检测类工作常使用 DET 作为主要数据来源, 调研发现 DET 是利用 DNS 执行数据窃取的代表性开源项目, 而且 DET 能较好完成 DNS 窃密任务。因此, 本文也复现并捕获了 DET 执行 DNS 窃密活动的真实流量, 作为对比样本数据之一。

5.2 生成数据评估

本文设计的初衷是为 AI 模型用户大规模生成可用的、完备度可调的 DNS 窃密流量数据, 生成数据的有效性是基本要求。得益于部署、运行的完整 DNS 窃密框架, 窃密任务执行情况即可作为生成流量数据有效性的直接证明, 且用户易于统计、验证。

为了评估 MalDNS 系统生成的 DNS 窃密流量数据的有效性, 结合课题小组已有真实 DNS 窃密流量数据积累情况, 本节主要以还原生成 APT34-Glimpse 和开源工具 DET 的流量数据展开分析, 对应生成的流量数据分别是“MDG-Glimpse”和“MDG-DET”。然后展开生成数据与真实数据的对比分析。

针对上述指定目标内容集, 不同配置的 MalDNS 系统实际执行 DNS 窃密任务统计情况(部分)如表 6, 其中以下统计数据都是 3 次以上重复实验的均值, DNS 数据包总量是指不同变体完成窃密任务所需的 DNS 包数量, 成功数量是窃密服务端成功恢复目标内容的数量。

表 6 MalDNS 系统窃密任务执行情况

Table 6 The performance statistics of exfiltrating through MalDNS system

序号	工具变体	配置目标类型	DNS 数据包总量	成功数量	成功率(%)
1	MDG-efficent	预测生成	807	100	100
2	MDG-stealthy	预测生成	9094	100	100
3	MDG-Glimpse	还原	3438	100	100
4	MDG-DET	还原	4114	100	100

在正常网络情况下, 如表 6 所示, 不同参数配置的 MalDNS 系统能够较好执行窃密任务, DNS 窃密成功率为 100%。实验结果表明, MalDNS 系统具备良

好的预测和还原生成能力, 完全可以胜任 DNS 窃密流量生成任务, 稳定的窃密成功率证明生成数据是有效的。与已有开源项目 DET、DNSExfiltrator 相比, DNS 窃密任务执行成功率更高。分析全部实验过程中曾经出现的 1 次恢复失败的主要原因是: 极差网络环境导致了 UDP 丢包个例, 致使某窃密 DNS 请求未到达服务端; 这不属于本文关注范畴。

与此同时, 本文方法生成的 DNS 窃密流量数据主要服务于 AI 模型训练, 评估生成数据在 AI 模型视角下的表现情况也是必要的。具体思路是: 综合选用多数论文中常用特征, 分别对真实的 DNS 窃密流量数据和生成的 DNS 窃密流量数据进行特征处理, 并展开对比分析。

参考近几年 DNS 窃密检测工作中的常用特征, 本文选取其中近 20 项关键特征对 DNS 窃密流量进行特征提取。选用的域名基础类特征例如子域名长度、子域名数字占比、子域名公共字串及其占比等; 也选用了 DNS 响应 IP 规律、DNS 请求-应答比、DNS 请求频率等其他特征。

为了更好的对比生成数据与真实数据之间差异性, 使用目前主流的 ISOMAP 降维算法, 将特征处理后的生成数据集与和真实数据集降至三维空间。直观分布情况如图 7 所示, 生成数据与真实攻击数据的空间分布高度拟合, 这表明 MalDNS 生成系统具备高度还原目标 DNS 窃密攻击的能力, 在 AI 模型视角下生成数据集与真实攻击流量数据集在空间分布上高度拟合度。因此, 应用本文方法生成的流量数据训练 AI 模型, 其效果可与真实攻击数据相媲美。

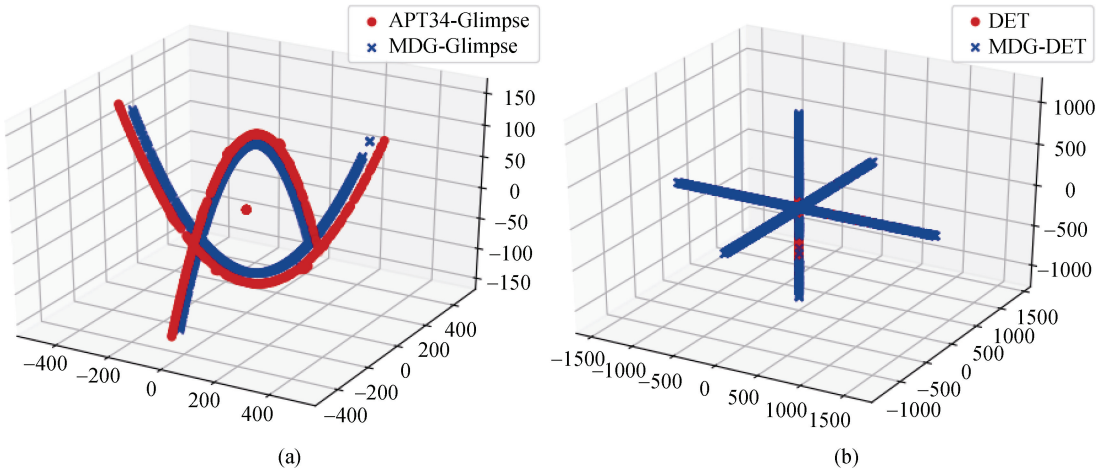

图 7 生成数据与真实数据的三维分布

Figure 7 The comparison of distribution between generated data and veritable data

5.3 生成数据训练所得模型性能评估

- **检测模型及评估概述**

在可用数据集紧缺的大背景下, 以攻击机理为理论基础, 设计并实现流量数据生成系统的方法由本文提出, 生成流量数据的应用效果可以通过训练所得 AI 模型的性能进行检验。首先, 使用生成数据训练所得模型的性能, 至少需要达到其他类似研究工作中的同等水平。理想情况下, 生成数据训练所得模型性能能够得到明显提升; 同时也可以反映 AI 模型性能受限于数据集的问题。

参考已有恶意 DNS 检测研究工作, 使用较多的机器学习检测模型有随机森林、SVM、J48 决策树、孤立森林等。其中, 随机森林是一种由多个决策树构成的分类器, 具备较好的可解释性; 而且不同决策树之间没有关联, 在近几年的恶意 DNS 检测工作中表现良好, 因而本文选取随机森林检测模型进行评估实验。

本节的实验评估思路是综合 MalDNS 系统还原生成、预测生成的所有 DNS 窃密流量数据作为训练阶段的数据集, 使用课题组积累的 APT34-Glimpse、DET 等真实 DNS 窃密攻击数据集作为验证数据集, 对训练所得模型的有效性和检测能力进行验证和评估。即这类宝贵的真实攻击数据对于 AI 模型的训练阶段而言是完全“不可见”的, 完成模型训练后才启用真实 DNS 窃密攻击数据, 用于验证模型对真实攻击的检测能力。

- **模型训练和验证阶段**

在模型训练阶段, 实验所用数据集由黑样本、白样本两部分组成。白样本主要来自于合作企业局域网的日常 DNS 流量数据, 其中 DNS 总数量达 50 多万。训练阶段使用数据的黑样本, 则是 MalDNS 系统生成的 DNS 窃密流量数据, 依据本次实验需求生成

的黑样本 DNS 总数量接近 45 万。模型训练过程的 5 折交叉验证结果如表 7, 结果表明: 使用生成流量数据进行检测模型训练, 对训练测试集的 DNS 窃密攻击具备准确的检测能力。

表 7　训练阶段 5 折交叉验证结果

Table 7　5-fold cross-validation results during the model training phase

分组	#1	#2	#3	#4	#5
准确率(%)	100	100	100	99.99	100
召回率(%)	100	100	100	100	100

验证对真实攻击的检测能力时, 验证数据集的黑样本主要 5.1 节所述的真实 DNS 窃密攻击流量。由于验证数据集对于检测模型是完全未知的, 模型对这类真实攻击数据集的检测能力, 能够实际代表所得模型对未知的真实 DNS 窃密攻击的检测能力。本次实验使用生成流量数据集训练所得的随机森林检测模型, 对验证数据集的检测结果见表 8。

表 8　生成数据训练所得模型对真实攻击的检测结果

Table 8　The detection performance of the model trained with generated data against veritable attacks

验证样本集	混淆矩阵		检测率(%)	误报率(%)
APT34	5812258	0 5342	99.85	0
DET	581225 0	0 13183	100	0
DNSExfiltrator	581225 0	0 23132	100	0

表 8 的检测结果表明: 使用生成数据训练所得检测模型, 对未知、真实攻击的检测能力良好, 检测率均高于 99.85%、误报率基本为 0。进一步分析 APT34 中出现的 8 条漏报记录发现, 这 8 条 DNS 数据并未实际执行 DNS 窃密传送; 而是与 SLD 域名的普通解析相关, 并没有实际传送窃密数据分片, 可以认为它们并不属于本文界定的 DNS 窃密流量范畴。例如漏报“mulong.club”(笔者复现攻击场景时, 注册使用的二级域名)的 DNS 流量为该域名的正常 DNS 解析请求, 并未携带任何窃密数据内容。

综上所述, 本文实验在检测效果方面表现良好, 应用生成数据训练所得到的检测模型, 能够有效检测真实的 DNS 窃密攻击。

6　总结与下一步工作展望

本文在 AI 助力防御的实践过程中, 发现并分析训练阶段面临的可用数据集紧缺、完备度不足的问题。首次提出面向 AI 模型训练的、基于攻击 TTPs 的流量数据自动生成及应用方案。遵循该方案, 梳理 DNS 窃密攻击 TTPs 作为理论基础, 设计并实现了 MalDNS 系统用于自动生成 DNS 窃密流量数据。最后对生成流量数据的有效性、应用于 AI 模型训练的效果展开实验评估。

评估结果表明: 本文提出的基于攻击 TTPs 的流量数据生成及应用方法是切实可行的; 基于 DNS 窃密攻击 TTPs 设计和实现的 MalDNS 系统, 不仅能高度还原已有案例报告中的 DNS 窃密攻击模式, 还具备良好的预测生成能力。综合参数可调的案例还原生成流量和预测生成流量数据, 能有效提升数据完备度和拓展未知样本空间, 从而有效提升训练所得 AI 模型的性能。相较于传统的训练数据收集方法优势明显: (1)生成数据规模不再受限, 降低用户获取训练数据集的难度; (2)综合案例还原和预测生成的流量数据, 可以有效提升数据集的完备度; (3)通过窃密任务执行情况易于直接验证生成数据的有效性。

本文设计的数据自动生成及应用方法, 旨在降低 AI 模型用户获取可用数据集的难度, 促进 AI 技术助力网络防御。本文将持续跟进最新 DNS 窃密攻击案例报告、完善 DNS 窃密攻击 TTPs, 维持 MalDNS 系统的持续更新和扩展。同时, MalDNS 系统源码拟开源供 AI 模型用户免费使用, 也期望得到更多安全研究专家的关注, 共同改进、完善该生成系统。

最后, 本文设计的数据自动生成及应用方法, 同样适用于面向 AI 模型训练的、其他攻击类型的流量数据生成及应用。

参考文献

[1] Software. MITRE ATT&CK®. https://attack.mitre.org/software/. Sept. 2020.

[2] An indepth analysis of Linux/Ebury. WeLiveSecurity. https://www.welivesecurity.com/2014/02/21/an-in-depth-analysis-of-linuxebury/. Feb. 2014.

[3] DNS tunneling in the wild: overview of oilRig's DNS tunneling. Unit42. https://unit42.paloaltonetworks.com/dns-tunneling-in-the-wild-overview-of-oilrigs-dns-tunneling/. Apr. 2019.

[4] New wekby attacks use DNS requests as command and control mechanism. Unit42. https://unit42.paloaltonetworks.com/unit42-new-wekby-attacks-use-dns-requests-as-command-and-control-mechanism/. May. 2016.

[5] The OilRig campaign: attacks on saudi arabian organizations deliver Helminth backdoor. Unit42. https://unit42.paloaltonetworks.

com/the-oilrig-campaign-attacks-on-saudi-arabian-organizations-deliver-helminth-backdoor/. May. 2016.

[6] OilRig actors provide a Glimpse into development and testing efforts. Unit42. https://unit42.paloaltonetworks.com/unit42-oilrig-actors-provide-glimpse-development-testing-efforts/. Apr. 2017.

[7] OilRig uses ISMDoor variant; possibly linked to Greenbug Threat Group. Unit42. https://unit42.paloaltonetworks.com/unit42-oilrig-uses-ismdoor-variant-possibly-linked-greenbug-threat-group/. Jul. 2017.

[8] Cobian RAT – A backdoored RAT. Zscaler. https://www.zscaler.com/blogs/security-research/cobian-rat-backdoored-rat. Aug. 2017.

[9] Operation Wilted Tulip – Exposing a cyber espionage apparatus. ClearSky Cyber Security. https://www.clearskysec.com/tulip/. Jul. 2017.

[10] A large-scale APT in Asia carried out by the oceanlotus group. Cybereason. https://www.cybereason.com/lab-analysis-operation-cobalt-kitty. Jun. 2017.

[11] Greenbug's DNS-isms. Netscout. https://www.netscout.com/blog/asert/greenbugs-dns-isms. May. 2017.

[12] Covert channels and poor decisions: the tale of DNSMessenger. Talos Group. https://blogs.cisco.com/security/talos/covert- channels-and-poor-decisions-the-tale-of-dnsmessenger. Mar. 2017

[13] Fu C. Research and implementation of attack traffic generation technology[D]. Beijing University of Posts and Telecommunications. 2017.
(付超. 攻击流量生成技术的研究与实现[D]. 北京邮电大学, 2017.)

[14] Wang X T, Wang Y W, Li P. Design and Implementation of Self-similar Network Traffic Generator[J]. *Microelectronics & Computer*, 2016, 33(8): 54-58.
(王晓婷, 王忆文, 李平. 一种自相似网络流量生成器的设计与实现[J]. 微电子学与计算机, 2016, 33(8): 54-58.)[万方 vs 知网]

[15] Wang Y J, Xian M, Chen Z J, et al. Design and Realization of a Network Attack Generator[J]. *Computer Science*, 2007, 34(2): 64-67, 90.
(王永杰, 鲜明, 陈志杰, 等. 一种网络攻击流量生成器的设计与实现[J]. 计算机科学, 2007, 34(2): 64-67, 90.)

[16] Peng D. Research and implement of netwrok attack data generation[D]. Beijing University of Posts and Telecommunications. 2010.
(彭丹. 网络攻击数据生成技术的研究与实现[D]. 北京邮电大学, 2010.)

[17] Ishikura N, Kondo D, Iordanov I, et al. Cache-Property-Aware Features for DNS Tunneling Detection[C]. *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, 2020: 216–220.

[18] Ahmed J, Gharakheili H H, Raza Q, et al. Monitoring Enterprise DNS Queries for Detecting Data Exfiltration from Internal Hosts[J]. *IEEE Transactions on Network and Service Management*, 2020, 17(1): 265–279.

[19] Preston R. DNS Tunneling Detection with Supervised Learning[C]. *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, 2019: 1–6.

[20] Liu C, Dai L, Cui W, et al. A Byte-level CNN Method to Detect DNS Tunnels[C]. *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*, 2019: 1–8.

[21] Nadler A, Aminov A, Shabtai A. Detection of Malicious and Low Throughput Data Exfiltration over the DNS Protocol[J]. *Computers & Security*, 2019, 80: 36-53.

[22] sensepost/DET. GitHub. https://github.com/sensepost/DET. Nov. 2017.

[23] Chen J H, Wang Y J, Lü C. An Automated Network Attack Traffic Acquisition Method Based on Python Symbol Execution[J]. *Computer Applications and Software*, 2019, 36(2): 294-307.
(陈家浩, 王轶骏, 吕诚. 一种基于 Python 符号执行的自动化网络攻击流量获取方法[J]. 计算机应用与软件, 2019, 36(2): 294-307.)

[24] Zhu J Y, Park T, Isola P, et al. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks [EB/OL]. 2017: arXiv:1703.10593[cs.CV]. https://arxiv.org/abs/1703.10593

[25] Luan F, Paris S, Shechtman E, et al. Deep photo style transfer[C]. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017: 4990–4998.

[26] Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative adversarial networks[C]. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2014: 2672–2680.

[27] Lee W, Noh B, Kim Y, et al. Generation of Network Traffic Using WGAN-GP and a DFT Filter for Resolving Data Imbalance[M]. Internet and Distributed Computing Systems. Cham: Springer International Publishing, 2019: 306-317.

冯林 于 2017 年在重庆大学通信工程专业获得学士学位, 现在广州大学计算机技术专业攻读硕士学位; 研究领域为网络安全。研究兴趣包括: 网络对抗、人工智能安全。Email: 2111806016@e.gzhu.edu.cn

崔翔 博士, 现任广州大学网络空间先进技术研究院研究员, 主要研究方向为网络安全。Email: cuixiang@gzhu.edu.cn

王忠儒 博士, 现任中国网络空间研究院高级工程师, 主要研究方向为人工智能、网络安全。Email: wangzhongru@bupt.edu.cn

甘蕊灵 于 2018 年在北京邮电大学信息安全专业获得学士学位。现在北京邮电大学计算机技术专业攻读硕士学位。研究领域为网络安全。研究兴趣包括: APT 远程控制技术、流量分析。Email: lynngan@bupt.edu.cn

刁嘉文 于 2019 年在北京电子科技学院计算机技术专业获得硕士学位。现在北京邮电大学网络空间安全专业攻读博士学位。研究领域为网络安全。Email: jarvand@bupt.edu.cn

韩冬旭 于 2013 年在华北电力大学获得硕士学位。现任中国科学院信息工程研究所高级工程师, 并攻读网络安全方向博士学位。研究领域为网络攻击检测、网络安全态势感知等。Email: handongxu@iie.ac.cn

姜海 硕士, 现任北京丁牛科技有限公司 CTO, 主要研究方向为网络安全、大数据、云计算。Email: jianghai@digapis.cn