ExecScent：在真实网络下使用自适应控制协议模板挖掘新的C&C域名

Terry Nelms 1,2, Roberto Perdisci3,2, Mustaque Ahamad 2,4

1Damballa公司 2佐治亚理工学院 3佐治亚大学 4纽约大学阿布扎比分校

**摘要** 在本文中，我们展示了ExecScent，一个旨在从真实的企业网络流量中挖掘新的、从未出现过的C&C域名的新系统。ExecScent从已知的C&C通信样本中自动地学习控制协议模板(CPT)。这些CPT会匹配它们所部署的网络流量。我们的目标是生成混合型的模板，它可以对具体的部署方案自动地进行调节，从而在一个给定的网络环境下产生更好的检出(TP)和误报(FP)之间的取舍。据我们所知，ExecScent是第一个使用这种自适应C&C流量模型的系统。

我们实现了ExecScent的一个原型版本，并且将它部署到三个不同的大型网络中运行两周。在部署期间，相比于使用一个大的实时更新的商业黑名单，我们发现了许多新的、从未出现过的C&C域名以及数以百计的感染主机。此外，我们将ExecScent发掘出的新的C&C域名部署在6个大型ISP网络中，发现了至少25000个新感染主机。

**1** 引言

代码重用是恶意软件常用的做法。通常情况下，新的恶意程序只是之前样本的简单打包，或者在之前的版本上增加几个新功能。此外，成熟的恶意程序的源代码在黑市上出售或者泄露已经很常见了，它们会被其他的恶意程序使用者所重用。

大多数的现代恶意程序，特别是僵尸网络，包括(至少)两个功能模块：一个客户端，它在感染主机上运行，以及一个控制服务端，它由恶意软件的所有者管理。由于代码重用适用于各个部分，所以这导致了大多数的恶意程序样本使用同一个常用的传输控制(C&C)协议，甚至不同的主控者虽使用不同的C&C域名和IP，但是却使用同一种控制服务器实例。

在本文中，我们展示了ExecScent，一个旨在从真实的企业网络流量中挖掘新的、从未出现过的C&C域名的新系统(见图1)。从已知的C&C通讯的种子列表和恶意程序生成的相关域名出发，ExecScent旨在利用不同恶意样本中C&C协议的共性发现新的C&C域名。更确切地说，我们指的C&C协议是一组实现恶意程序控制应用逻辑规范的集合，相比于C&C通讯用到的传输层协议(如TCP或UDP)和应用层协议(如HTTP)，它定义了一个更高层次的抽象。ExecScent旨在从已知的C&C通讯的种子中自动地学习给定的C&C协议的独特特征并导出一个控制协议模板(CPT)，该模板可以部署在网络的边缘用来检测新的C&C域名的流量。

ExecScent构建了自适应模板，它也可以从部署模板的网络流量中学习。我们的目标是产生混合型的模板，它可以自适应每种不同的具体部署方案，从而在一个给定的网络环境下产生更好的检出(TP)和误报(FP)之间的取舍。这样做的灵感来自于不同的网络有着不同的流量轮廓(例如，一个金融机构和技术公司产生的网络流量肯定截然不同)。因此就可能出现这种情况：一个CPT可能在一个给定的网络中产生很高的误报，但是在其他的网络中就可以没有误报地正确检测出C&C域名。我们采用一个实用的方法，旨在自动地识别这些情况并且当CPT部署在时，可以降低CPT的判断“信心”。这使我们可以降低总体的误报风险，同时在其他的网络中保持一个高的检出率。我们会在第3节介绍自适应模板的使用。

ExecScent关注基于HTTP的C&C协议，因为有研究表明大量的恶意软件家族和几乎所有的已知的移动僵尸网络使用基于HTTP的C&C通讯。此外，很多企业的网络出口管控和防火墙的过滤规则阻止了一切的非web流量。这迫使针对企业网的恶意程序选择使用HTTP(或HTTPS)作为通讯协议。同样需要着重注意的是，许多现代的企业网络部署了web代理并使用了SSL中间人(SSL-MITM)技术。因此，企业网可以将ExecScent的模板部署在web代理处来发掘新的C&C域名，甚至是使用了HTTPS通信手段的C&C域名。

我们的系统不同于之前提出的基于URL的C&C签名生成系统。不同于以往的工作，我们建立了可以适应部署网络的模板并且它可以对所有的HTTP请求进行建模，而不只局限于URL字符串。我们会在第5节说明，相比于只对C&C请求的URL进行“静态”建模，对所有HTTP请求建模使得我们可以得到一个更好的真正和假正之间的平衡。

基于异常的僵尸网络检测系统通常需要监控网络中的多个受到同类恶意程序感染的主机，这些系统对大型网络的扩展能力不强，并且不能将可疑的C&C通讯归于一个特定的恶意程序家族。不同于这些系统，ExecScent可以在只有一台感染主机的情况下低误报地检测出C&C通讯，并且可以将新发现的C&C域名归于一个已知的恶意程序家族或恶意程序的主控者(也就是恶意操作背后的犯罪组织)。ExecScent标记的C&C域名反过来也会部署在现有的基于DNS流量的恶意程序的轻量级监测中。在非常大的网络环境(如ISP网络)下，监控所有流量是不切实际的，这可以有助于探测和溯源恶意程序的感染情况。

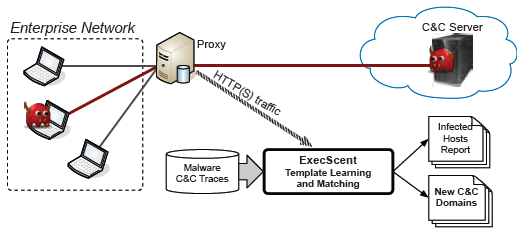


图1 ExecScent系统部署整体示意图。自适应控制协议模板从恶意程序的网络痕迹和现实的网络流量中学习，学习到的自适应模板被用来探测新的网络流量中的C&C域名

目前，我们通过利用已知的C&C域名黑名单来找出ExecScent用来学习控制协议模板的已知C&C流量的种子。基于动态分析的C&C发现系统例如Jackstraws也可能用于此目的。但是，不同于ExecScent，Jackstraws可以用来发现C&C流量的“种子”，但它的基于system-call的检测模型不能在真实的网络流量中检测新的C&C域名。

总之，我们做出了以下贡献：

* 我们提出了ExecScent，一个在真实网络流量下挖掘新的恶意C&C域名的新系统。ExecScent自动地在它部署的网络流量下学习C&C流量模型。这种“自适应”的方法使得我们在保持一个很高检出率的同时极大地降低了误报率。据我们所知，ExecScent是第一个使用这种自适应C&C流量模型的系统。
* 我们实现了ExecScent的一个原型版本，并且将它部署到三个不同的大型网络中运行两周。在部署期间，相比于使用一个大的实时更新的商业黑名单，我们发现了许多新的、从未出现过的C&C域名以及数以百计的感染主机。
* 我们将ExecScent发掘出的新的C&C域名部署在6个大型ISP网络中，发现了至少25000个新感染主机。

**2** 系统概述

ExecScent的基本目标是从已知的恶意程序生成的基于HTTP的C&C通信中生成控制协议模板(CPTs)。然后我们用这些CPTs去识别新的、从未出现过的C&C域名。

ExecScent自动地寻找不同恶意样本的C&C协议间的共同的特点，接着将这些共性编码进CPT集合中。每个模板都标有恶意程序家族的名称或者与该C&C流量相关的犯罪组织(如果有的话)。一旦将CPT部署在网络的边缘(如图1)，任何与该模板匹配的HTTP(S)流量就会被认为是C&C流量。与该流量相关的域名就会被标记为C&C域名，并且会归于恶意程序家族或者与该CPT关联的组织。

图2展示了ExecScent生成和标记CPT的整体过程。我们在本节简单地介绍不同系统组件的功能，详细的会在第4节介绍。

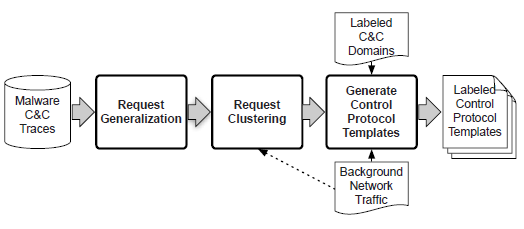


图2 ExecScent系统概述

给定一个大的恶意软件生成的网络痕迹，我们首先重构每个恶意样本的HTTP请求。接着，我们进行一个“请求泛化”的过程，这里我们将请求的一些参数(如URL参数值)替换为它们的类型和长度，如图3所示。注意，ExecScent将所有HTTP请求都考虑在内，不仅仅是URL(见3.2节)，而且泛化过程应用于请求头的每个部分。进行泛化这一步的动机是提高“请求聚类”过程的精确度，我们希望将相似的C&C协议的恶意程序请求聚在一起。

当恶意程序的请求被聚类后，我们进行模板学习过程，在这里我们推导出CPT。本质上讲，一个CPT高度概括了聚在一起的HTTP请求，并且记录了一系列重要的特征如URL结构、请求头的集合、每个恶意软件访问过的IP地址等等。此外，这些模板与恶意程序家族标签一一关联(详细说明请见4.4节)。

在真正部署模板前，我们先把CPT放在应该部署到的网络流量中观察一段时间。特别是对于模板的每一个组件(如泛化的URL路径、user-agent字符串、请求头集合等等)，我们计算各个组件在部署网络中出现的频率。在背景流量中“活跃”的那些组件会在该网络中得到一个低的“匹配信任”。另一方面，在流量中那些不经常(甚至是从来没)出现的组件会得到较高的信任。我们是指这些“罕见”的组件在部署的网络流量中有着高度的特殊性。这样做的原因和动机会在下节中详细讨论。

部署后，如果一个HTTP请求被CPT以很高的相似性和特异性所匹配，那么它就会被标记为C&C请求。这就是说，一个请求与CPT描述得很接近并且匹配的CPT的组件在特定的部署网络中有着很高的特异性(即不经常出现)。

**3** 方法的直觉和动机

在本节中，我们讨论建立自适应控制协议模板背后的直觉和动机。此外，我们讨论了使用整个HTTP请求而不像以往的工作那样仅限于URL字符串发掘C&C的优点。

**3.1** 为什么是自适应模板？

与大部分的流量模型相似，从流量中推导出来并匹配C&C的ExecScent CPT可能并不完善并且会产生一些误报。为了尽量减少这种风险，ExecScent构建了自适应控制协议模板，它除了从已知的恶意程序的C&C流量中学习之外，也从模板所部署到的网络流量中学习。我们主要发现不同的企业网络有着不同的流量轮廓。一个金融机构(例如一个大银行)产生的网络流量与一个制造公司(例如一个汽车生产商)或者科技公司的(人软件开发公司)可能截然不同。所以就会导致了CPT可能在网络环境下产生很高的误报，而在其他网络中有着很好的效果。直观地，我们的目标是自动识别这些情况，当模板匹配到网络时，降低它的“判断把握”，而在部署到其他情况时，保持它的“判断把握”。

例如，假设是一家美国银行，它的主机很少或者从不访问中国的IP。如果发现了一个指向中国IP地址的HTTP请求，这就是一个异常事件。直觉上，CPT同样匹配这样的请求，我们有着很高的把握认为这是C&C通信。另一方面，假设是一个与中国有着合作关系的汽车生产厂家，中的主机就会联系得很频繁。如果一个中的HTTP请求符合CPT的条件但是它指向的IP地址是合作伙伴的，那么我们匹配的把握值就应该下降。

更具体地说，考虑以下假设的情况。假设我们有一个模板，在和中以相似性得分*s*匹配到了HTTP请求。为了简单起见，假设*s*在和中是一样的。同样假设与匹配流量相关的服务器IP(或它的 /24前缀)在中是，中是。同样，假设在中很常见，而在中有着非常低的知名度，因为它从来没有被网络中的主机访问过。因为在中很常见，这意味着中的一大部分主机(例如超过50%)曾经访问过该域名，它很可能是模板偶然间匹配到了良性的流量，从而可能在中产生大量的误报。另一方面，因为在中不常出现，这个匹配对于它来说很可能是一个正确的决定，或者在中的任何情况下会产生相当低(可能只有一个)的误报。因此，基于在和观察到的最近的流量，当在中匹配时，我们需要降低的把握值，但在中就不一样。换言之，需要自动地“适应”以便降低误报。与此同时，在中保持中很高的把握值意味着我们已经可以检测匹配的C&C通信并同时降低误报的风险。除了目标IP，我们将这种方法推广到ExecScent模板的各个组件中，例如URL的结构、user-agent字符串以及其他请求头等等。

总之，给定网络，我们匹配模板的把握值取决于两方面的因素：

* 相似性：HTTP请求与密切关系的度量。
* 特异性：对于的流量，的组件稀缺性的度量。

当一个HTTP请求同时以很高的相似性和很高特异性匹配到一个CPT时它就会被标记为C&C请求。我们会在第5节说明这种方法优越于不将这些特殊性考虑在内的C&C模型。

**3.2** 为什么考虑所有的请求内容？

恶意软件的C&C请求通常需要携带足够多的信息，感染主机上的恶意客户端需要与C&C服务器进行鉴权。直观地讲，C&C服务器希望确保它是在与其中的一个bot对话，从而避免暴露它的真实性质。有些功能性的爬虫或安全研究者可能会将探测服务器作为调查的一部分。这通常是通过使用一组特定的嵌入C&C请求的URL的参数和值来实现的。以往的关于自动URL签名生成的研究表明在这种情况下有很多可喜的成果。但是一些恶意软件(如TDL4)与C&C交流信息时首先会加密它，接着对加密结果进行编码(如使用base-64编码)，然后将它嵌入进URL路径中。此外，鉴权字符串同样也可以嵌入到其他区域诸如user-agent中(例如一些恶意样本使用它们的MD5哈希值作为user-agent名称)，编码进其他请求头中(例如在referrer中)以及在POST请求的内容里。因此，根据我们的实验结果来看(第5节)，只考虑URL可能不足以精确地为C&C请求建模以及检测新的C&C域名。

**4** 系统详细信息

我们现在详细介绍ExecScent的内部实现。系统的整体概述请参见第2节。

**4.1** 输入的网络流量

正如我们在第1节中提到的，ExecScent主要关注基于HTTP的恶意程序，即那些将HTTP(或HTTPS)作为恶意程序控制传输的一个基本的网络协议。为了达到这一目的，ExecScent将恶意程序产生的HTTP流量痕迹作为输入(在我们的实验中，我们使用专业从事恶意软件防御的知名公司提供的数据)。

值得注意的是虽然一些恶意程序使用HTTPS作为通信方式来躲过检测，但是这并不代表我们的部署方案有着难以逾越的障碍(见图1)。事实上，很多我们需要ExecScent的企业网络已经部署了可以执行SSL-MITM的web代理服务器，它可以将明文的HTTP请求转发到ExecScent的模板匹配模块，例如使用ICAP协议(RFC 3507)。另外我们也会在有着SSL-MITM代理的运行样本的(虚拟)机器和路由器间的这样的沙箱环境下重新分析使用HTTPS的恶意样本。毕竟，当SSL-MITM存在时不支持或选择不运行的基于HTTPS的恶意软件也将无法在有着相似配置的企业网络中运行，我们也不研究这种情况。

**4.2** 请求泛化

在下面几节中，我们将讨论为了得到高质量的控制协议模板，我们首先需要将类似的C&C请求聚在一起。为了达到这一目标，我们在使用聚类算法前需要定义好一个适当的相似距离。以前的以URL为中心的聚类系统主要是基于字符串的相似性。从本质上讲，两个URL被认为是相似的条件是它们有相近的编辑距离，或是包含有一段相同的子串(或tokens)。然而这些系统没有考虑到这样一个事实，相比于考虑特定的字符串序列，URL包含的变量的数据类型的相似性更容易被测量和判断。考虑图3中两个假设的C&C请求。图3a中的两个请求由于在字符串中存在很多不同的字符，所以他们的距离相对来讲很大。为了避免这种情况，ExecScent使用了一系列启发方法来检测表示特定类型的字符串，并相应地使用包含数据类型和字符串长度的占位符标签来替换它们(如图3b)。

例如，我们将认为“fa45e”为小写的十六进制因为它包含的数字和字母都是有效的十六进制数字。我们现在支持的数据类型有整数、十六进制(大写，小写以及混合)、base64(标准的以及“URL”安全的)和字符串(大写，小写以及混合)。另外，对于整数、十六进制数和字符串我们可以识别出加上其他符号的数据类型例如“：”或“.”(例如192.168.1.1就会被认为是一种长度为11的整数+点的数据类型)。此外，我们的启发方法可以轻松地扩展到诸如IP地址、MAC地址、MD5哈希以及版本号等情况。

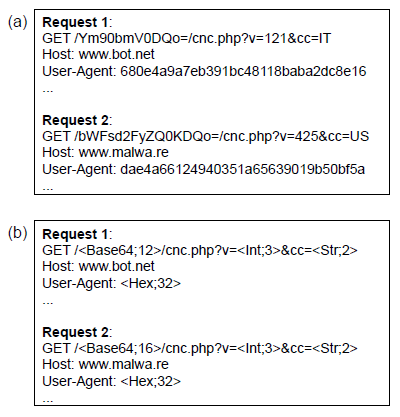


图3 C&C请求的示例：(a)原始请求；(b)泛化后的请求

泛化的过程使得我们可以定义一个更好的相似距离(4.7节)，这有助于得到更高质量的C&C请求聚簇。注意到以往的工作如[23,30]只关注URL字符串，而ExecScent将整个的请求头都考虑在内。例如，在图3中user-agent字符串是MD5，它可以被泛化后的有着特定数据类型和长度的MD5字符串所替换。

**4.3** 请求聚类

在提取模板前，我们将相似的C&C请求聚在一起。聚类这一步主要是为了帮助自动CPT生成算法提高生成高效和高产的模板。高产的意思是足够通用，可以在新流量中匹配相似的C&C通信；高效的意思是足够精确，只有非常低的(甚至没有)误报情况发生。

我们分两个阶段来进行C&C请求聚类过程。第一个阶段，我们基于它们的目的IP粗略地对C&C请求进行聚类。具体地讲，如果两个C&C请求的目的IP的C段有着“一段DNS关系”我们就将它们聚在一起。也就是说，如果在30天内曾经有一个域名解析到了两个不同的C段上，我们就认为这两个C段有这种“关系”。为了找到这种“关系”，我们需要大型的被动DNS数据库。

在第二阶段，我们考虑在第一阶段得到的粗粒度聚簇，并将它们依照内容相似性函数进一步进行聚类。我们在粗粒度聚簇内使用一个分层聚类算法对C&C请求聚类，这些泛化的URL特征包括相似的user-agent字符串，相似的HTTP请求头的项数以及各自的值等等。当计算两个请求的距离时，我们将请求内容的“相似性”和“特异性”都考虑在内。其中特异性(或者说“低出现性”)可以通过最近从不同网络中收集到的流量数据集来计算。关于聚类阶段提到的相似性的详细定义，请读者参考4.7节。

**4.4** 生成CPT

当聚类完C&C请求后，每个聚簇就会产生一个控制协议模板。在这一阶段，我们只考虑那些至少包含一个对已知的C&C域名发出HTTP请求的聚簇。每一个模板是一个聚簇中的所有C&C请求的高度概括，它包括以下部分(如图4)：

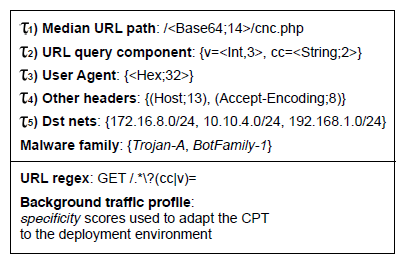


图4 CPT示例

) *中间URL路径*：中间路径字符串是使得到所有请求的URL路径的编辑距离的总和最小的字符串。*选择原因*：尽管可能每个恶意软件产生的URL路径都各不相同，我们观察到在很多情况下，对于一些特殊的恶意程序家族或者操作，路径中存在固定的部分。

) *URL查询组件*：包括每个URL查询的参数名、参数值以及值的长度。*选择原因：*URL的参数往往被恶意软件用来传递感染主机的信息，诸如它的操作系统版本，感染主机的唯一标识符等等。

) *User-agent*：在请求中发现的所有不同的(泛化后的)user-agent字符串。*选择原因：*user-agent是最常被恶意软件滥用的HTTP头之一，而且它有时用作一种弱身份验证。

) *其他请求头*：在请求中发现的其他请求头的集合。对于每个头部，我们也存储它的值的字符串长度。*选择原因：*有的恶意软件家族有着独特的请求头，包括它们的顺序和值。

) *目的网络*：聚簇中C&C请求的目的IP地址的C段的集合。选择原因：在一些情况下，C&C服务器可能被重新分配一个在同一网段的新IP地址。

* 恶意程序家族：聚簇中与已知C&C请求相关的恶意程序家族的集合。

此外，每个CPT包括以下与部署相关的信息：

* + *URL正则*：为了提高模板匹配阶段的效率(4.6节)，每个模板包含一个自动从请求中的URL字符串集合中生成的正则表达式。URL正则被有意设计成一个非常通用的，并且可以在部署过程中过滤掉那些极不可能匹配整个模板的流量，从而减少在真实流量和模板下计算HTTP请求相似度的成本。
  + *背景流量轮廓*：过去W(W是系统参数)天内从部署环境的流量中得到的信息。它被用来计算CPT组件的特异性，从而使我们将CPT适应其部署的网络，4.5节会详细介绍。

注意到CPT扮演了推导了它的聚簇的“质心”的角色。为了判断一个新请求是否与一个给定的聚簇足够相似，我们只需要将它与CPT比较，而不必与聚类中的所有C&C请求比较。因此，CPTs提供了一个有效的计算新的请求与已经聚类好的恶意样本的相似度的方法。

**4.5** 适应部署网络

如3.1节所述，CPT部署后，当一个HTTP请求同时以很高的相似性和很高特异性匹配到一个CPT 时它就会被认为是C&C请求。为此，我们首先需要计算每个部分的特异性得分(其中k=1,…,5，见图4与4.4节)，来看看该元素在部署网络的流量轮廓中是多么得罕见。

例如，为了计算的特异性得分，我们首先计算中每个user-agent字符串的基于主机个数的得分。我们考虑过去的W天内(W是一个可配置的时间窗口参数)在部署网络下产生包含的HTTP请求的主机的个数。定义，其中分母是在部署网络的流量中观察到的user-agent字符串个数的最大值。同样，我们计算基于域名个数的得分，它是包含的HTTP请求的目的域名个数，定义。这样做是因为只有当user-agent字符串请求多个主机和域名时它才会被认为是常见的。另一方面，如果有着很高的主机个数(如 “Windows-Update-Agent”)但是却有着很低的域名个数(如 唯一使用过的域名就是 microsoft.com)我们就不认为它很常见。最后，我们定义的特异性得分。利用同样的方法，我们为中的每个请求头计算特异性得分。

为了计算的特异性得分，我们计算中的每个网络前缀的请求主机个数，接着计算每个前缀的得分。

4.5.1 URL特异性

由于网络中充斥着太多各种各样的URL，计算URL部分的特异性更加复杂。为了解决这一问题，我们依靠监督分类的方法。首先，给定一个网络下收集到的所有流量的集合，我们提取所有的URL，并学习URL的词频map，其中词通过符号化URL获得(即提取URL路径，文件名，查询字符串中的元素)。接着，对于一个给定的新URL，我们将其转化为特征向量，其中统计特征描述了符号化URL中单一“字词”的平均频率、查询参数中词的2-gram的平均频率、文件名的平均频率等等(为了提取URL中每个词的频率，我们需要去查询之前建好的词频map)。

在我们将背景流量下的URL都转换为特征向量后，我们训练一个可以标记新的URL是否为“活跃”或“不活跃”的SVM分类器。为了准备训练集，我们进行如下操作。我们首先基于域名个数来对URL进行排序(也就是说，我们认为出现在不同域名多个站点的请求的URL更为“活跃”)。接着我们取这个排名的顶部和底部的URL作为“活跃”和“不活跃”的样本。我们用这些标记了的数据集去训练SVM分类器，并且我们利用SVM使用的max-margin方法去生成一个可以推广到不在训练集中的URL的模型。

在运行阶段(当部署了训练好的SVM分类器后)，给定一个URL ，我们首先将转换成对应的特征向量，并将输入到SVM分类器中。分类器可以判断是“活跃”还是“不活跃”的。在实际中我们不考虑这些分类的结果，我们只考虑SVM认为它是“活跃”的分类(把握)得分。因此，SVM的输出可以这么理解：相对于在背景流量中观察到的URL，的得分越高，它就越有可能是“活跃”的URL。最终，该URL的特异性的得分为，其中是SVM对的输出。

现在我们回过头来考虑模板和它的两个与URL有关的部分和(见图4)。我们首先通过连接中间URL路径()和(排好序的)泛化的参数名称和值来构建“中间URL”。接着令特异性得分，其中是的特异性得分。

**4.6** 模板匹配

模板匹配有两个阶段。如上文所述，每个模板包括从聚簇的C&C请求中提取来的URL正则表达式。对于一个新的HTTP请求r，为了测试该请求是否匹配模板，我们首先将r的URL与的URL正则相匹配。值得注意的是，如4.4节提到的，URL正则构建得非常通用，它只用来过滤掉那些极不可能匹配模板的流量。此外，我们检查r的目的IP是否在中的前缀里(即在内)。如果URL正则和目的IP都不匹配，我们就认为r不匹配模板。否则，我们接下来考虑r请求的全部内容，根据请求泛化过程(4.2节)将r转化，并计算(泛化后的)请求r与模板的整体匹配得分。

概括地讲，得分S是通过计算请求r与模板每个部分的相似性得到的。这些相似度量会根据它们的特异性进行加权，匹配得分就是通过取这些加权后的相似度量的平均值来计算的。关于相似性函数的定义以及特异性在的计算中起到一个什么样的作用，我们会在4.7节详细描述。

如果超过了一个检测阈值，那么请求r就会被视为C&C请求，与r有关的域名(假设r没有使用硬编码的IP地址)就会被认为是C&C域名并且标记为与有关的恶意程序家族。此外，发起r请求的主机也会被标记为感染了恶意程序的主机。

**4.7** 相似函数

4.7.1 CPT匹配得分

为了判断一个新的HTTP请求r是否匹配一个CPT，我们这样计算匹配得分：

(1)

其中是将r中与中的各自比较的相似函数，是一个动态权重(定义会在下面给出)，它是中第k个部分的相似性和特异性的函数。的分母介于0和1之间。

元素是请求r的目的域名d的特异性，计算方法为，其中是部署网络的流量中查询域名d的主机个数，是查询最“活跃”域名的主机个数。因此，我们使用来减少低特异性域名的匹配得分(也就是说那些被大量主机查询的域名)。这样做是因为企业网络中只有相对一部分主机感染了特定的恶意程序，大多数现代的恶意程序通过隐蔽下载和社会工程学攻击传播得相对较慢。所以，在一个监控网络下一个新的C&C域名不太可能在一个短暂的时间内(例如 1天)被绝大部分(例如 50%)的主机所查询。

接下来，我们描述等式(1)中相似函数的详细计算方法。此外，我们还详细介绍了当算出的值后，每个部分的特异值是如何选取的(特异性的相关定义请参见4.5节)。

1. 给定与r相关的URL路径，我们计算该路径和CPT的中间URL路径的归一化编辑距离。URL路径特异性计算方法见4.5节。
2. (a)我们计算r的URL请求参数名称集合与的名称集合的Jaccard similarity。参数名称的特异性等同于(见4.5节)。

(b)我们比较泛化后的URL查询字符串参数的数据类型和值的长度(见4.2节)。对于查询中的每一个元素，如果r的数据类型和中的数据类型相匹配，我们计1分。此外，我们计算r和中值的长度的比值。最终等于这些得分的平均值，即匹配的数据类型和长度越多，相似性得分就越高。

1. 我们计算r中的泛化user-agent和集合中每个字符串的归一化编辑距离。令是这些距离中最小的，也就是说m是最接近模板user-agent的字符串。定义，令特异性。
2. 给定r中剩余的其他请求头部，我们分别计算每个的相似性。首先，我们计算r的头部集合和集合的Jaccard similarity j。接着，我们考虑r和中头部元素出现的顺序。如果顺序匹配，我们令变量,否则。最后，对于每一个头部，我们分别计算r和中值的长度的比例。相似性得分是所有这些部分得分的平均值(也就是说j,o和长度比例)。我们令特异性得分,其中是请求头。
3. 令为请求r的目的IP。如果在中任意一个前缀网络中，我们令1，否则令0。假设有前缀(这种情况1)。在这种情况，我们令特异性。

动态权重函数的计算方法如下：

(2)

其中是一个静态权值(即常量)，n是一个配置参数。注意，并且这些权值由等式(1)的分母进行归一化，所以(因为另外依据定义，)。

动态权重的灵感来源于我们希望给予那些既在相似性也在特异性上都很好匹配了CPT 的请求r中的部分一个高的权值。事实上，当相似性和特异性都等于1时权重获得最大值，当相似性和特异性其中一个趋于0时权重获得最小值。

综上所述，相似性判断两个值有多像，而特异性判断它们在所属网络下是多么的独特和唯一。动态权重使我们能突出CPT和请求之间的罕见结构元素，从而我们可以将它们用作检测的主要特征。由于这些罕见的元素在不同的恶意程序家族中有着不同的重要性，通过突出这些“独特的特征”，我们能够检测和区分不同的恶意程序。

4.7.2 聚类阶段的相似函数

在4.3节，我们描述了C&C请求的聚类过程。在本节我们定义用来计算HTTP请求对之间的相似性函数。

给定两个HTTP请求和，我们使用等式(1)计算他们的相似性。此时读者可能发现等式(1)是用来计算一个HTTP和CPT的相似性的，而不是计算两个请求。我们使用等式(1)的原因是我们可以将一个CPT看成是只从一个HTTP请求中提取而来的。此外，如果我们想把用于计算动态权重的特异性得分包括进来，我们可以使用先前从一个或多个网络收集的流量的数据集（参见图2中的虚线箭头）。

**5** 评估

在本节中，我们描述用来评估ExecScent的数据(5.1节)，系统是如何启动来执行实验的(5.2节)，展示在不同真实网络下的实验结果(5.3节)。此外，我们量化了使用整个HTTP头而不是只考虑URL的优势，以及使用自适应模板相比于“静态”C&C模板的优势(5.4节)。我们展示了通过将由ExecScent发现的新的C&C域名部署到大型ISP网络中的得到的可喜成果(5.5节)。

**5.1** 评估数据

5.1.1 恶意程序的网络痕迹

我们获得了商业公司的威胁情报数据(由一家知名的安全公司提供)，并用它来生成控制协议模板(CPT)。通过它我们收集到了包括HTTP流量的每天约8000条的恶意程序网络痕迹。每条网络痕迹都标记有生成网络活动的恶意软件的可执行文件的哈希，以及(如果已知)相关恶意软件系列名称。

5.1.2 真实的网络流量

为了评估ExecScent，我们拿到了三个大型生产网络的实时流量，称为UNetA,UNetB和FNet。UNetA和UNetB是两个位于美国的不同的学术网络，FNet是一个大型北美金融机构的计算机网络。表1报告了在这三个网络中观察到的网络业务的统计。例如，在UNetA我们观察到每天有平均7893个不同的活跃IP。平均来说，这些网络主机每天生成超过34.8M的HTTP请求，这些请求有平均149,481个不同的域名。

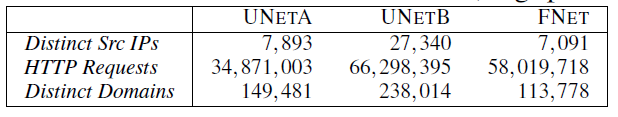


表1 平均每天真实网络的流量统计

5.1.2 参考标准

为了评估精度和误报，我们使用以下数据：

* CCBL：我们从一家知名的安全公司获得了一个包括数十万个C&C域名的黑名单，我们称之为CCBL。值得注意的是，CCBL与大多数公开可用的域名黑名单有所不同，原因有二：1)这些C&C域名由专业的威胁分析团队仔细审查过；2)这些域名有标记了其对应的恶意程序家族以及运营名字(如果有的话，它是运营C&C的犯罪组织的标识符)。
* AWTL：我们从Alexa(alexa.com)的前100万的全球域名中生成一个大的良性域名白名单。在这些100万的域名中，我们过滤掉那些被认为是“effective”的顶级域名，如那些与动态DNS服务相关的域名(如dyndns.org, no-ip.com等等)。接下来我们剔除掉在过去的一整年的90%时间里没有进前100万的域名。为此，我们在过去一年中每天都整理一个更新的顶级域名列表，并且只将一直出现在前100万个域名列表中的域名视为良性。这个过滤阶段的目的是移除那些短时间内变得热门的恶意域名的那些噪音。经过这个精简后，我们留下了大约45万个常用域名。
* PKIP：我们也维护了一个停靠IP列表，PKIP。即那些与停靠域名服务相关的IP地址(例如由注册服务商暂时接管的过期或未使用的域名的IP)。我们使用这个列表来修正ExecScent的模板。事实上，CPT是自动地从恶意程序产生的网络痕迹的HTTP请求中提取出来的，由于那些网络痕迹的相关域名在CCBL中，所以它们被标记为了C&C通信。然而，CCBL中的一些域名可能已经过期，并且可能现在指向了一个停靠站点。这可能导致在恶意程序的追踪中将一些HTTP请求错误地标记为C&C请求，从而在ExecScent的CPT中引入噪音。我们使用PKIP来过滤掉这种噪音。
* 威胁分析：显然，当我们评估ExecScent时，获取到完整的真实网络下所有流量参考标准是不现实的。为了弥补这一点并得到一个更好的精度和误报率(相比于只使用CCBL和ATWL)，我们在专业的威胁情报的帮助下对我们的实验结果进行了广泛的手动分析。

**5.2** 启动系统

为了进行评估，我们实现并部署了一个基于Python的ExecScent概念验证版本。在本节中我们讨论了将系统部署到真实网络前的准备过程。

5.2.1 聚类参数

正如4.3节所说，为了生成CPT，我们首先需要将请求进行聚类。这一步的主要目的是提高CPT学习过程的效率。聚类阶段依赖于层次聚类算法，它将需要切割的树状图(即由聚类算法生成的“距离树”)的高度作为输入，从而将HTTP请求分割为请求聚簇。

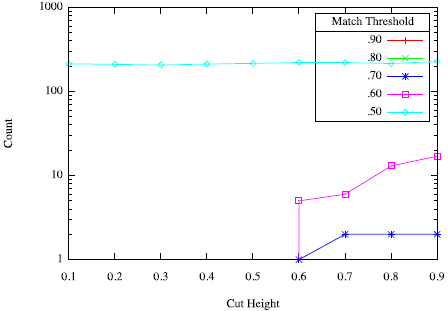


图5 不同树状图切割高度的影响(FP)

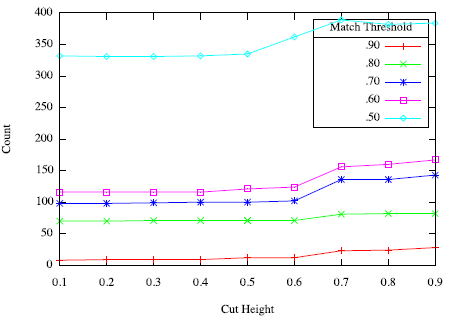


图6 不同树状图切割高度的影响(TP)

为了选择树状图的切割高度，我们进行如下操作。我们考虑收集到的一天的恶意程序痕迹(大约8000条不同的恶意痕迹)。接着我们对这些痕迹进行聚类，并通过在不同高度切割树状图来产生不同的聚类结果。对于这些不同的聚类结果中的每一种情况，我们提取相关的CPT集合，并用不同的匹配阈值在下一天的恶意程序痕迹中测试这些CPT。图5和图6分别统计了得到的误报数，即将良性域名分类错误(使用ATWL测试)和检出数，即成功辨认出C&C域名(使用CCBL测试)的情况。注意到虽然在这一阶段我们使用恶意程序生成的网络痕迹来测试CPT，由于一些恶意软件会在联系C&C域名的同时查询大量的良性域名，所以依旧会有误报的情况出现。

如图5和图6所示，对于每一个固定的匹配阈值，改变树状图切割高度不会显着改变误报和检出的个数。换句话说，CPT的匹配结果对聚类参数的具体数值不是特别敏感。最终我们决定将切割高度定为0.38(接下来的实验中一直沿用)，因为它在CPT生成步骤中在保持很高的CPT质量的同时，也能提供不错的效率。

5.2.2 CPT生成

为了生成用于在真实网络(5.3节)下评估ExecScent的CPT，我们最初使用从威胁情报中获取到的两个星期的恶意程序痕迹。为了在恶意程序的痕迹中标记C&C的HTTP请求，我们使用CCBL黑名单。我们还使用停靠IP列表PKIP来修正CPT相关的停靠C&C域名，如5.1.3节所述。一旦部署了这组初始CPT，我们就会持续从数据源中收集新的恶意痕迹，并且每天向CPT集合每天添加新的从这些后添加的恶意痕迹中提取出的CPT。更确切地说，令为首次将CPT集合部署在真实网络的那一天，为初始的CPT集合。是从的前两周收集到的恶意痕迹中生成的。CPT集合接着会在那一整天用来检测新的C&C域名。与此同时，在那一天我们从当天产生的恶意痕迹中生成了额外的CPT，我们会将它们添加到集合中去。因此，那一天结束时我们有了CPT的扩展集，我们将它部署在天中，以此类推。在部署阶段结束后，我们有了超过4000个不同的CPT。

为了将这些CPT与每个部署的网络流量相适应(4.5节)，我们使用一个类似的方法。我们基于在前两天内每个部署的网络上观察到的所有HTTP流量构建了一个背景流量轮廓，并用这个轮廓去适配CPT的初始集。然后，我们每天都根据当天观察到的新实时流量更新流量轮廓统计信息，并使用此信息进一步调整所有CPT。注意，部署到不同网络的CPT集合是不同的，因为它们针对每一部署网络都进行了不同的适应(利用网络的背景流量轮廓)。

**5.3** 真实网络部署结果

为了评估ExecScent，我们将它部署在三个不同的网络UNetA, UNetB和FNet下运行两周。我们总共使用4周的恶意程序产生的网络痕迹(部署前有两周，其余部署的两周每天都会更新)来生成上面提到的(5.2.2节)自适应的CPT集合。CPT匹配引擎部署在每个网络的边缘。

检测阶段的步骤如下。对于每一个网络，我们记录下只要有其中一个CPT匹配得分的所有HTTP请求，并连同请求的目的IP地址、相关域名、产生请求的源IP地址自己实际的得分这些信息一起记录。这样对于不同的检测阈值，我们可以在检出数和误报数中得到一个取舍出来。具体来说，令为匹配得分在检测阈值之上的请求，令*d*为与*h*有关的域名。于是我们将*h*标记为C&C请求，并将*d*认为是C&C域名。我们接着利用CCBL和ATWL列表以及人工分析(在专业威胁分析的帮助下)去确认*d*是是一个正常的检出，也就是说*d*的确是一个C&C域名；还是一个误报，即*d*不是C&C域名。

图7总结了在三个不同的真实网络中两周内得到的检出数和误报数的汇总，表2展示了针对一组有代表性的检测阈值不同网络的分析结果。例如表2中，考虑检测阈值为0.65时的UNetA。在两周的部署中，我们总共发现了66个C&C域名，其中34个是从未出现过在我们商业黑名单CCBL中的新域名。这66个C&C域名与17个不同的恶意程序家族有关。总体上我们检测出105台感染主机，其中有90个是与这34个未出现过的新C&C域名有关的新感染的主机。这意味着仅依靠CCBL黑名单有90个感染主机(86%)不能被检测出来。

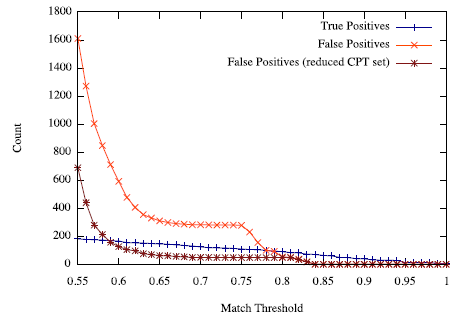


图7 不同检测阈值下的CPT检测结果

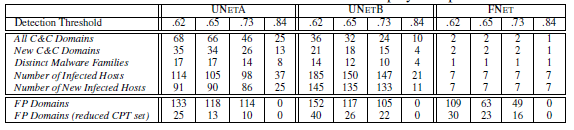


表2 两周的部署时间内真实网络下的检测结果

这些CPT产生了118个误报，即我们将域名错误地认成了C&C域名。我们注意到绝大多数误报都由两个CPT产生(这两个CPT在所有的网络中产生了最多的误报)。通过剔除由于这两个“噪声”CPT产生的误报，我们只剩下了13个误报，如表2的最后一行所示。在图7中这些标记为“reduced CPT set”的误报也与没有这两个CPT的结果有关。总之，在两周的测试中ExecScent产生的误报在可控的范围内，通过专业的威胁分析我们可以在几个小时内就将这些误报的C&C域名剔除掉。

注意到在FNet网络中检测出了很少的新的C&C域名(只有两个)。实际上，FNet是一个非常敏感的金融机构，它有很多层的网络安全机制从而用来防止恶意程序的感染。但是我们的结果表明了即使是这么强大的网络依旧不堪一击。

5.3.1 Pushdo Downloader

需要说明的是，图7和表2报告的所有结果是在剔除由某一CPT检测出的域名后得到的，该CPT容易导致数以百计的误报。通过手动分析，我们很容易发现ExecScent已经正确地学习了该CPT，事实上它对PUSHDO下载变种的HTTP通信进行了建模。这一特殊的变种故意地复制它的C&C请求，并且将它们发送到大量的良性域名诱饵中去。恶意程序这么做就是想在众多的良性域名间试图隐藏真正的C&C域名。但是，虽然这么做使得在众多良性域名中发现真实的C&C域名变得异常艰难(需要一些人工分析)，但是我们很容易就能得到匹配PUSHDO CPT的这些请求的主机，它们感染了特定恶意程序的变种。

我们会在第6节进一步讨论恶意程序类似的噪音和误导。

5.3.2 UNetB部署结果

相比于UNetA和FNet，我们用了稍微不同的方式来得到UNetB的部署结果。由于UNetB中的流量较大，我们自验证的CPT匹配引擎不能轻松地跟上这些流量。这是因为我们的匹配引擎程序与其他有着更高优先级的生产软件共享硬件资源。在进行这个实验的几周后，我们实现了一个用C语言写的优化版本，它比Python的要快8倍。因此，它就能很轻松地跟上UNetB的流量。

为了弥补我们实现的原型的性能问题，在两周的部署期间内，我们只考虑每隔一天的流量。也就是说，我们只使用了UNetB中七天的流量用来匹配CPT，有效地削减了ExecScent中一半的流量。

**5.4** “静态”以及URL-Only模型

在本节中，我们将ExecScent的自适应模板的结果与静态的模板相比较。静态的模板只从恶意程序产生的痕迹中学习而不将部署网络的流量轮廓考虑在内，基于URL的C&C请求模型只使用了从URL中提取到的信息。

为了得到“静态”模型，我们只取ExecScent的CPT而舍弃特异性参数的部分。换句话说，我们令等式(1)中的特异性得分为0(除外，令它为1)，将动态权重赋予为(见4.7节)。在下文中，我们将这些静态(非自适应)模板称为“特异性-关闭”模型。

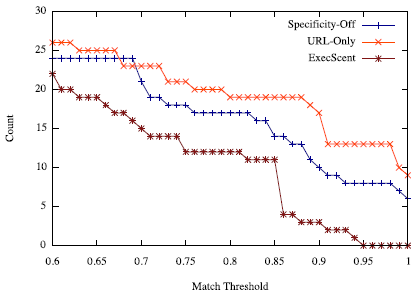


图8 C&C模型的比较-检出数

为了得到基于URL的模型，我们先关闭特异性信息，并且忽略ExecScent的CPT中除了与URL有关的所有组件。实际上，在等式(1)中我们只使用在4.7定义的相似度函数，和。我们将这些模板称之为“URL-Only”模型。

为了进行这个比较，我们将ExecScent的CPT以及“特异性-关闭”模型和“URL-Only”模型在UNetA, UNetB和FNet下运行4天。图8和图9分别统计了检测阈值时各个模型的检出数和误报数。从这些图中可以看出，下ExecScent的自适应模板比另外两个模型表现得都要好。除非我们愿意牺牲很大的一部分的检出率的损失，相比于时的数据，“特异性-关闭”模型和“URL-Only”模型会产生一个非常大的误报数(请注意图9的y轴上的刻度)。

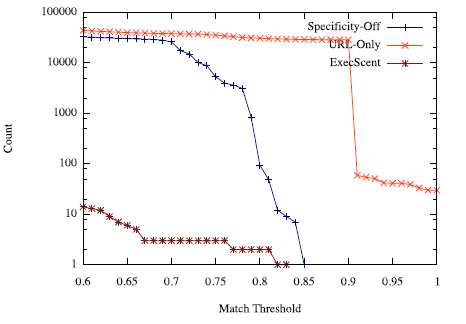


图9 C&C模型的比较-误报数

**5.5** ISP网络下的部署

我们还可以在服务几百万台主机的6个大型ISP网络下评估ExecScent的结果。我们进行如下操作：给定在5.3节描述的真实网络部署下ExecScent发现的65个新的C&C域名，我们将这些域名部署到6个ISP中一个星期，此时我们监控所有的DNS流量。我们每天计算查询这65个C&C域名中任意一个的不重复的源IP地址的个数。我们发现每天最多有25584个不同的源IP查询过这些C&C域名。换句话说，ExecScent发现的新的C&C域名使得我们可以识别6个ISP网络中的25584个新的潜在感染主机。

**6** 局限

那些知道ExecScent工作原理的攻击者可能会通过每当C&C服务器重新有了一个新的域名时就改变它的僵尸网络C&C协议这样的方法来躲过检测。一种可能的方法是在切换到新的域名之前实现一个可以部署到所有客户端和服务端的新的协议。但是这样做会显著增加僵尸网络管理的复杂性并且损伤它的灵活性。此外，对于那些中型和大型的僵尸网络，推进这样的更新需要时间，而且更新过程中的错误可能导致丢失部分或整个僵尸网络。

另一种逃避的方法是将噪声注入进C&C协议中从而使其看起来有一些不同。例如攻击者可以在发出请求时随机生成C&C URL路径或查询字符串的键值对。然而如果恶意程序客户端需要传达足够的信息用来向C&C服务器进行鉴权，则至少有一个请求是具有某种形式的“结构化”数据。由于ExecScent通过协议结构来衡量相似性并且给予独特的共享部分更大的权重，攻击者在所有部署的网络情况下逃避检测是十分重要的。事实上，在评估ExecScent期间有一些恶意程序家族就是使用这种类型的技术来尝试躲过正则表达式的检测。

攻击者还可能通过在域名匹配中注入噪声来尝试“误导”检测。例如，攻击者可以将它们发送到真实C&C服务器的C&C请求发送给许多良性域名作为诱饵。在评估期间我们发现PUSHDO恶意程序变种使用了这种方法。这种形式的恶意噪声很容易识别出来，因为在一个很短的时间内匹配到指定CPT的单一主机查询过的不重复的目的域名的数目暴露了它。因此，检测感染主机就非常容易。但是，在所有的域名中判断哪个是C&C域名就更加困难了。这种情况下我们会在把这些域名添加进黑名单前先做一个威胁评估。与此同时，安全管理员可以立即向感染主机警告，从而及时地进行补救。

融入到背景业务流量中是另一种可以避开检测的技术。例如，攻击者可以在它们的C&C协议模板中选择“常用的”数据类型和数据值。对于一些诸如URL路径的部分很容易选一个常用的值(如 “index.html’)。但是对于绝大多数的组件来说，“通用性”是相对于部署网络的流量而言的。因此，攻击者需要根据受感染机器的网络来定制协议。这可能很难做到因为在同一网络下，大多数的网络主机不能或者只能看到有限的整体流量。因此，虽然一个C&C协议可能有着一些“共性的”组件，ExecScent的自适应CPT依旧可以利用那些特异性的组件在部署的网络中检测C&C请求。

最后，ExecScent的CPT依赖于恶意痕迹以及从中提取出的标记了的C&C请求。因此在对C&C协议建模和检测前，ExecScent至少需要一些恶意程序家族提供的样本。这种情况下恶意代码的重用很大程度上帮助了我们。一些代码已经被重用(由于代码出售或泄露)的恶意样本反过来会帮助我们检测未来的恶意软件。注意多亏了请求泛化过程(4.2节)，ExecScent原则上只需要一个样本来生成CPT。话虽如此，当不止一个样本共享同一套C&C协议时，CPT的质量可以显著地提高。

**7** 相关工作

*恶意程序聚类和签名生成*：很多研究者利用从HTTP请求中提取到的特征对恶意程序进行聚类。具体地说，Perdisci等人提出了一个系统，它将请求相似URL的恶意程序样本进行聚类。此外，它从这些URL中提取token的子序列用于检测真实网络下的感染主机。还有的系统利用HTTP的请求方法和URL参数来聚类相似的恶意样本。作者将他们的聚类技术描述为手动过程，并且提到会在将来用自动化系统替换它。

最近提出的系统FRIMA基于协议的特征(如相同的URL路径)聚类恶意样本，而且对于每一个聚簇生成一组网络签名。网络签名是从字节串中创建的token集合，这些字节串是聚簇中大部分网络流量所共有的。为了减少误报，通过删除匹配作者的良性流量池中任何的通信来修正网络签名。自动化的网络签名生成也被用于检测蠕虫。生成的签名通常包括固定的字符串或token子序列，它可以部署在入侵检测系统中。AutoRE扩展了自动签名的产生过程，它可以产生匹配电子邮件中的URL的正则表达式从而检测垃圾邮件并将它们分组为垃圾邮件活动。

我们的工作聚焦于自动生成用于检测C&C通信和将这些通信归于一个已知恶意程序家族的模板。特别是，我们的主要重点不是聚类恶意软件样本本身。相反，我们应用聚类技术主要作为一个优化步骤来生成高质量的控制协议模板。此外，我们不限于仅考虑URL或提取常见的tokens集合。更重要的是，我们的C&C模板是自适应的，因为它们从将被部署的网络的流量中学习从而自我调节并自动地得到检出和误报的取舍。

*僵尸网络检测和C&C识别*：大量的研究一直在解决检测僵尸网络流量的问题。BotSniffer和BotMiner是基于异常的僵尸网络检测系统，它们寻找主机间相似的网络行为。这个想法是感染了相同僵尸程序的主机有着相同的C&C通信模式。此外，BotMiner利用这样一个事实：bot以协调的方式相应命令，并产生了相似的恶意网络活动。这种类型的检测系统需要多个感染主机在同一受控网络下。此外由于是基于异常的检测，它们不能降感染归因于特定的恶意程序家族，并且这种方法相对来说会产生较高的误报率。

我们的工作不同于这些，因为ExecScent可以在网络中只有一个bot的情况下检测出僵尸网络C&C。此外，不同于以往的工作，ExecScent使用了一个混合的检测方法，它从已知的C&C通信和部署的网络流量中学习从而生成自适应模板，它能以很高的精度和很低的误报率检测C&C域名。

Wurzinger等人提出了通过在沙箱环境下执行恶意样本从而将C&C流量从恶意和合法的混合流量中分离开来。他们提出先去识别恶意网络行为(如扫描，垃圾邮件等)，接着回过头来分析这个网络流量，发现可能代表命令送达到恶意程序的网络流，它代表了之前识别的恶意活动的启动步骤。但是，在恶意程序的网络痕迹中发现命令不是一直都可行。事实上，很多恶意程序的网络痕迹的数据集是通过运行数以千计的恶意程序样本得到的，每个样本只分配到了很短的运行时间。因此，在这么短的时间内看见一个有效的发到样本中的命令的可能性微乎其微。另一方面，恶意样本通常是一旦运行就联系C&C服务器，即使可能第一次联系时没有发出执行恶意活动的命令。因为这个原因，ExecScent并不聚焦于识别由恶意程序产生的恶意网络活动以及相关的命令。反而ExecScent利用与C&C服务器相关的任何形式的(基于HTTP)通信来学习控制协议模板，从而以后可用于识别新的C&C通信和相关的C&C域名，就算恶意活动不能被直接观察到。

Jackstraws在一个沙箱中运行恶意程序从而生成与网络通信相关的系统调用的行为图。然后将这些系统级行为图和C&C图模板进行比较从而找到新的C&C通信。ExecScent与它不同之处在于它只依赖于网络信息，并且不需要在沙箱中(例如，它可以使用“裸机”执行或者真实网络收集的痕迹)执行恶意程序来学习模板。此外，不同于Jackstraws，ExecScent学习自适应模板，这使得我们可以在真实的网络下识别新的C&C域名。

*恶意域名*：最近很多文献提出了通过监控DNS流量来识别恶意域名的方法。这些系统把域名分为恶意和良性两种，但并不把它们归于特定的恶意程序家族。同样，[2,6]主要是域名信誉系统，它将一般的恶意域名而不仅仅是C&C域名赋予一个低的信誉得分，恶意域名和C&C域名之间没有明确的界限。另一方面，[4]只关注使用伪随机的域名生成算法的恶意软件。Kopis是唯一专注于通用恶意域名的系统，但是它需要在上层的DNS中监控DNS流量，这很难做到。

不同于以上提到的基于DNS的系统，ExecScent关注于通过监控HTTP(S)流量并使用自适应的C&C协议模板在真实企业网络中挖掘新的C&C域名。

**8** 结论

我们提出了ExecScent，一个在真实企业网络流量下挖掘新的C&C域名的新系统。ExecScent从已知的C&C通讯的样本以及将要部署的网络的背景流量中学习自适应控制协议模板，从而在一个给定的网络环境下产生更好的检出(TP)和误报(FP)之间的取舍。

我们将ExecScent的一个原型版本部署到三个不同的大型网络中运行两周。在部署期间，相比于使用一个大的实时更新的商业黑名单，我们发现了许多新的、从未出现过的C&C域名以及数以百计的感染主机。我们同样将ExecScent的自适应模板与“静态的”C&C流量模型相比较。我们的结果表明ExecScent胜过那些不将部署网络的背景流量考虑在内的模型。此外，我们将ExecScent发掘出的新的C&C域名部署在6个大型ISP网络中，发现了至少25000个新感染主机。

致谢

我们感谢匿名评论者的宝贵意见。这些资料一部分受到国家科学基金会No. CNS-1149051.的支持。在本文中表达的任何观点、研究发现、结论或者建议只与作者有关，不代表国家科学基金会的观点。