2020 年全国大学生信息安全竞赛

作品报告

作品名称： 基于基因图谱的 APT 攻击检测与同源判定系统

电子邮箱： 694554872@qq.com

提交日期： 2020 年 06 月

填写说明

1. 所有参赛项目必须为一个基本完整的设计。作品报告书旨在能够清晰准确地阐述（或图示）该参赛队的参赛项目（或方案）。

2. 作品报告采用A4纸撰写。除标题外，所有内容必需为宋体、小四号字、1.5倍行距。

3. 作品报告中各项目说明文字部分仅供参考，作品报告书撰写完毕后，请删除所有说明文字。(本页不删除)

4. 作品报告模板里已经列的内容仅供参考，作者可以在此基础上增加内容或对文档结构进行微调。

5. 为保证网评的公平、公正，作品报告中应避免出现作者所在学校、院系和指导教师等泄露身份的信息。

目　　录

摘要

近年来新型恶意代码数量持续增加，对全球网络安全造成巨大威胁，恶意代码的分析与检测作为网络安全防范中的先导环节，直接影响网络防御体系的预警和响应能力。而在各类网络攻击中，又以APT攻击的威胁程度最高，检测难度也相对较大。

现在的攻击检测技术多使用特征码匹配技术或是动态沙箱分析技术，依赖于专家经验，且对变种、新型代码的检测能力较弱。同时越来越多的新型APT样本具有规避动态沙箱检测、隐藏真是行为的能力。此外，人们希望能对APT攻击进行家族聚类，以便分析APT组织内和组织间的关联性。

相比于传统APT攻击检测系统，本作品在以下几个方面有了改进。一是利用软件基因的相似性实现检测，相比于传统的基于特征码的检测技术，对变种代码和新型代码的检测能力更强。同时，直接从代码层面挖掘APT攻击的特征，解决了某些新型APT攻击能够规避动态沙箱检测、隐藏自身真实行为的难题。二是通过样本之间的基因相似性，构建了直观的基因图谱，用户可以通过阅读图谱清晰地了解到各个APT家族内和家族间的关联性。三是利用基因图谱中的家族聚类，我们可以实现对未知APT样本的同源判定，识别该样本属于哪个家族。四是能够将检测到的APT样本加入基因图谱中，因此可以在检测中不断丰富APT基因图谱，提供更完整的展示APT关联以及检测APT攻击样本的能力。

关键词：APT、基因图谱、检测、同源判定

Abstract

In recent years, the number of new malicious code continues to increase, which poses a huge threat to the global network security. As the leading link in the network security, the analysis and detection of malicious code directly affects the early warning and response ability of the network defense system. Among all kinds of network attacks, apt attack has the highest threat degree and is relatively difficult to detect.

The current attack detection technology mostly uses signature matching technology or dynamic sandbox analysis technology, which relies on expert experience, and has weak detection ability for variant and new code. At the same time, more and more new apt samples have the ability of avoiding dynamic sandbox detection and hiding real behavior. In addition, it is hoped that family clustering of apt attacks can be carried out in order to analyze the relevance within and among apt tissues. Compared with the traditional apt attack detection system, this work has improved in the following aspects. One is to use the similarity of software gene to realize detection.

Compared with the traditional detection technology based on feature code, the detection ability of variant code and new code is stronger. At the same time, mining the characteristics of apt attacks directly from the code level solves the problem that some new apt attacks can avoid dynamic sandbox detection and hide their real behavior. The second is to construct an intuitive gene map through the genetic similarity between samples. Users can clearly understand the relevance within and between apt families through reading the map. Third, by using the family clustering in the gene map, we can realize the homology determination of the unknown apt samples and identify which family the samples belong to. Fourthly, it can add the detected apt samples into the gene map, so it can enrich the apt gene map continuously in the detection, and provide a more complete display of apt Association and the ability to detect the apt attack samples.

Keywords: apt, gene map, detection, homology determination

第一章 作品概述

1.1 背景分析

近年来新型恶意代码数量持续增加，对全球网络安全造成巨大威胁，恶意代码的分析与检测作为网络安全防范中的先导环节，直接影响网络防御体系的预警和响应能力。而在各类网络攻击中，又以 APT 攻击的威胁程度最高，检测难度也相对较大。

现在的 APT 攻击检测技术多使用两种技术。一是静态的特征码匹配技术，该技术有两方面局限性：（1）只能识别已知的恶意代码变种，对新型样本检测能力弱；（2）特征片段与恶意代码的行为没有必然联系，可能是不含语义的片段，因此可以随意修改该片段从而绕过查杀，大量免杀工具正是基于该原理；并且，由于 APT 攻击具有持续性、隐秘性等特点，如果使用常规的流量分析，需要分析很长一段时间中产生的大量流量，这对存储、计算能力都是巨大的挑战。二是动态沙箱检测技术，通过使 APT 代码在沙箱环境中运行，记录样本对系统中文件、进程、网络等的操作行为。该技术对各种类型样本具有普适性，分析范围较广。而随着恶意代码技术的不断更新，出现了能够规避动态沙箱检测的恶意代码。这类代码能够检测当前运行环境，若检测到沙箱环境则会隐藏自己的真实行为使得动态分析失败。此外，人们希望能对 APT 攻击进行家族聚类，以便分析 APT 组织内和组织间的关联性。

基于以上两种检测方式的缺陷和现实需求，我们构建了一个利用基因图谱实现对 APT 代码进行检测和同源判定的系统，能够准确地识别 APT 代码，并对 APT 代码进行分类，展示 APT 家族间和家族内的关联性。

1.2 相关研究

Chen 等人[1]提出了一种简单易实现、贴近生物基因形态的软件基因划分原则。充分考虑了 APT 代码分析的需求，对造成干扰的片段进行合并，切分边界同时体现软件的执行逻辑。软件基因的边界灵活，产生的代码片段相较经典的 n-gram 方法，碎片更少，处理速度更快，所构造的检测模型性能更优。同时受生物信息学中利用隐马尔科夫模型对 CpG 岛进行检测的方法所启发，Chen 提出了基于高阶马尔科夫链的内存

中软件基因提取方法，该方法从统计的、非精确的方法入手，克服了静态反汇编的难题，真正实现了动静结合分析。

Bingling Zhao 等人[2]提出了一种基于遗传观点的新的 APT 同源性判定方法。在 Zhao 的方法中，APT 软件基因被定义为功能依赖图（FDG）的子图。首先，我们通过计算来自不同恶意软件家族的每个 API 的重要性，来选择不同 APT 家族的关键应用程序编程接口（API）。然后，将 FDG 划分为不同的子图，这些子图仅包含连接到关键 API 的路径。然后，我们使用社区划分算法将子图划分为关键子图。关键子图定义为 APT 软件基因。最后，通过图压缩对恶意软件基因进行编码。在此基础上，我们设计了一种频繁子图挖掘的算法。算法将同一族的压缩图作为比较对象，以挖掘代表性频繁子图，该子图将被视为 APT 家族的基因。

Huang 等人[3]在 2016 年提出了基于多任务深度学习框架的二进制恶

意代码分类技术，以 450 万个样本为训练集，200 万个样本为测试集，检测误报率仅为 0.358%，恶意代码家族分类失误率仅为 2.94%。

Qiao 等人[4]在 2019 年提出了基于汇编指令词向量与卷积神经网络的 APT 代码分类方法。首先逆向 APT 代码可执行文件获取汇编代码，将其中的汇编指令看作词，函数看作句子，从而将一个恶意代码转换为一个文档 ；然后对每个文档使用 Word2Vec 算法获取汇编指令的词向量，依据在训练样本集中统计的 Top100 汇编指令序列，将每个文档转换成一个矩阵 ；最后基于 CNN 构造神经网络结构，用于在训练样本上训练分类模型。本文方法可避免特征提取阶段对专家经验的依赖，在降低特征维度的同时也避免了过拟合。

Chen 等人[5]将文件的二进制形式，直接转换成了 DNA 的脱氧核糖核苷酸序列，用 ATCG 表示，进而采用生物基因比对的相关算法，进行样本之间的比较和分析。该方法丧失了程序的语义性质。APT 代码的恶意性本质，只有在语义中才能显示，而且不可规避。二进制文件其实也只是 APT 代码的一种形态、形状，不可直接作为基因来看待。

Cen 等人[6]对 APT 代码做了反编译，得到了反编译源码（text 文档形式），其核心工作是在特征提取方面，将特征分为三个粒度层级，此中包括包级（如 import java.io. File 的条数和内容），类级（class）和函数级（function），其中保留 API 作为研究的核心内容。接着使用文本处理中的 TF 来进行 API 重要性排序，实现特

征的提取。在新近许多研究中，研究者们很热门于将自然语言处理中的 TF-IDF 等文章主题模型，应用到 API 序列的研究中，通过不同的映射方法加以应用，得到良好的效果。

Itay Cohen 等人[7]搜集了约 2000 个针对俄罗斯的 APT 攻击样本，试图找出每个样本的以下信息: 作者——哪一个俄罗斯 APT 作者已知或可能写了这个恶意软件(图拉、索法基、格雷能源……) ，家族——与该恶意软件相关的常见家族名称是什么，模块——许多恶意软件系列都是以模块化的方式构建的。研究者根据不同样本之间共享的唯一和恶意代码对它们进行聚类。利用恶意软件分析技术，自动将每个二进制文件分解成数千个小的汇编代码片段，也称为“基因”。然后，通过基因的相似性来判断样本之间的关联性和完成聚类，最终生成俄罗斯 APT 代码的关联图，用于进一步研究。Itay Cohen 等人最终发现了 22000 个样本和 385 万条共享代码之间的联系。

Ishai Rosenberg 等人[8]的研究中指出：尽管各个国家的 APT 组织在发展不可同源判定的 APT 方面作出了种种努力，但仍有可能通过某些方式得出相当准确的同源判定结果。此外，不同国家的 APT 组织使用不同的 APT 开发方法，例如在同一 APT 组织内的开发人员的作品仍然足够相似，从而可以进行同源判定。Ishai Rosenberg 等人提出了一个对 APT 样本进行同源判定的有效方法，使用原始的动态分析报告作为输入，并训练一个深度神经网络作为分类器。使用原始报告具有节省手工培训和分析过程所需的成本和时间的优点，并防止丢失指示性数据和分类器的准确性。

Nari 等人[9]在 2013 年提出了基于网络行为的恶意代码自动化分类方法，该方法通过执行恶意代码获取网络数据包，基于这些数据包构建网络交互流程图，然后从图中获取节点数、根出度、平均出度、最大出度等特征，将每个样本转换为一个向量，最后利用分类算法对恶意代码进行家族分类。

2013 年，Park 等人[10]提出了基于共有系统调用路径的恶意代码的检测与分类方法，实验结果表明误报率接近 0%。2015 年，Pascanu 等人[11]将恶意代码动态执行时的指令与 API 等视为恶意代码的语言，然后利用循环神经网络算法进行恶意代码分类，准确率达到了 98.3%。Giannella 等人[12]在 2015 年提出了基于谱聚类的恶意代码聚类方法，使用的特征包括恶意代码的执行时的调用图、HTTP 等动态信息。

Garbervetsky 等人[13]针对 C#语言编写的程序进行分析。其主要做法是针对程序中的函数调用依赖关系进行研究，并分析和推测其中变量流的类型(Type)。通过程序

的函数调用依赖关系上的差异，发现恶意代码与正常程序间的区别。

Miller 等人[14]采取了一种主动学习的方式，先实现了一款基于监督学习 SVM 算法的分类器，然后对无法明确判定正常与恶意的目标文件（结果不可靠的）采用人机交互，人工分析并打上标签后，模型重训练并获得更健壮的检测器。

David, O.E 等人[15] 提出了一种基于深度学习的恶意软件特征自动生成与分类的方法，该方法使用一个深度信任网络(DBN),由一个深度去噪自动编码器堆栈实现，生成恶意软件行为标签。

Kwon B J 等人[16] 提出了下载程序图的概念，利用它捕获主机上的下载活动，并探索了良性与恶性下载图的增长模式。通过对下载程序图表的分析，可发现大部分恶意软件下载活动。

1.3 特色描述

- 软件脱壳技术

借鉴了 Xabier U 等人[17] 和 Alessandro M 等人[18]的文献，采用团队内部的软件脱壳工具，实现了对大多数 APT 样本的脱壳，对于少量无法完成脱壳的样本采用人工分析。

- 基因图谱展示样本关联性

以基因图谱的方式展现 APT 样本间的关联性以及 APT 家族聚类。关联性用样本节点间的连线表示，不同家族用不同的颜色加以区分，使得用户能够方便地从基因图谱中获得想要的信息。

- APT 代码检测

直接从汇编代码级别挖掘 APT 样本的“基因”，作为 APT 代码的特征，解决了动态分析可能无法触发代码真实行为的难题，提高了对变种代码、新型代码的检测能力。

- 利用基因图谱实现同源判定

通过基因图谱中包含的 APT 家族聚类信息和样本基因信息，我们实现了对新 APT 样本的同源判定。通过 KNN 算法判别样本属于哪个 APT 家族，在对 APT 进行检测的基础上，更进一步地找出了其隶属的 APT 家族，为分析 APT 组织内和组织间的关联性做出了贡献。

- 通过检测不断完善基因图谱

当一个待检测样本被判定为APT攻击之后,它将作为一个新节点加入基因图谱中。因此我们可以在检测中不断丰富 APT 基因图谱，提供更完整的展示 APT 关联以及检测 APT 攻击样本的能力。

- 友好的 Web 界面设计

Web 页面向用户展示当前的基因图谱，用户可与基因图谱进行交互，如点击节点可查看其属于的家族、与哪些样本间存在多大的关联性。同时用户可从上传页面上传代码进行 APT 检测和同源判定，Web 向用户返回 APT 攻击检测结果以及 APT 攻击分类结果。

1.4 应用前景分析

随着互联网的不断发展，APT 攻击对于企业和国家的威胁已经成为信息安全防护中越来越突出的问题。人们在生活的方方面面都不可避免地需要接触互联网，如果缺乏相应的攻击检测机制，不仅会造成巨额的经济损失，甚至会泄露国家的安全机密，对社会稳定产生巨大威胁。目前，对于一般的恶意代码已经具备了较成熟的检测和分类手段，但对 APT 攻击等能够规避常规检测方式的新型攻击手段还缺乏有效的检测方法。同时人们对世界范围内 APT 组织的恶意攻击行为的协同性还不甚了解。

因此，在网络安全越来越重要的时代背景下，研发一种 APT 攻击的检测和同源判定机制是迫在眉睫的。考虑到新型恶意代码往往具有规避动态分析的能力，研究如何提取 APT 代码的特征，也是十分必须的。

本项目是一个 APT 攻击检测与同源判定系统。由于使用了基因图谱技术，直接从汇编代码层挖掘 APT 样本的特征，无需通过运行样本触发其恶意行为，解决了由于样本规避动态分析而无法获得其真实行为的难题。同时，利用基因图谱进行 APT 检测和同源判定，相比于常规方法，对变种和新型样本的检测能力更强。

本项目可分析各 APT 的基因关联图谱以及整体 APT 的基因关联图谱，了解 APT 间技术的关联性。可以判定一个恶意代码是否是 APT 攻击意图。同时由于被判定为 APT 的样本会加入基因图谱，因此可以在检测中不断丰富 APT 基因图谱，提供更完整的展示 APT 关联以及检测 apt 攻击样本的能力，具有优异广阔的应用前景。

1.5 文章组织框架

本文重点研究基于基因图谱技术的 APT 攻击检测与同源判定，实现带前端 Web 页面的 APT 攻击检测与同源判定系统。本文组织如下：

第一章概述了项目背景，简要地介绍了当前网络安全形势的严峻，以及 APT 攻击的危害性，介绍了国内外研究现状，分析了本项目的创新点和应用前景。第二章着重介绍了本项目的总体构思和核心算法、技术以及具体实现。在第三章中我们对系统做了全面的实验评估，并对系统做出评价。第四章总结了项目中的创新点。最后在第五章对项目进行总结。

第二章 作品设计与实现

2.1 系统方案

系统总体框架如图 1 所示，系统共包含五大模块：脱壳、基因提取、基因图谱、APT 检测与同源判定以及 Web 前端。样本集中带有 APT 组织标签的样本首先进入脱壳模块进行脱壳处理，该模块使用基于 Unipacker 开发的脱壳工具。脱壳成功后样本进入基因提取模块，在该模块我们使用 Radare2 反汇编工具得到样本完整汇编代码，利用基因提取算法抽取样本基因，接着对基因进行去噪清洗得到 APT 样本的特征基因，最后将特征基因存入 APT 基因数据库中。基于该数据库，我们构建了基因图谱模块，根据样本间的基因共用情况对样本进行关联；我们还设计了 APT 检测模块，根据样本基因构造样本向量，结合机器学习算法实现对未知样本的 APT 攻击检测，对于检测为 APT 的样本，进一步利用基于基因公用的 KNN 算法对其进行 APT 同源判定，即预测该样本所属的 APT 组织。最终，我们利用 Web 前端展示可交互的基因关联图谱，用户可通过图谱获取 APT 组织内部、APT 组织之间的关联性信息；同时可接收用户上传的未知样本，进行 APT 攻击检测与同源判定。

由于针对 Linux 系统的恶意代码较少，故本系统的分析对象集中于 Windows 系统的 PE 文件。

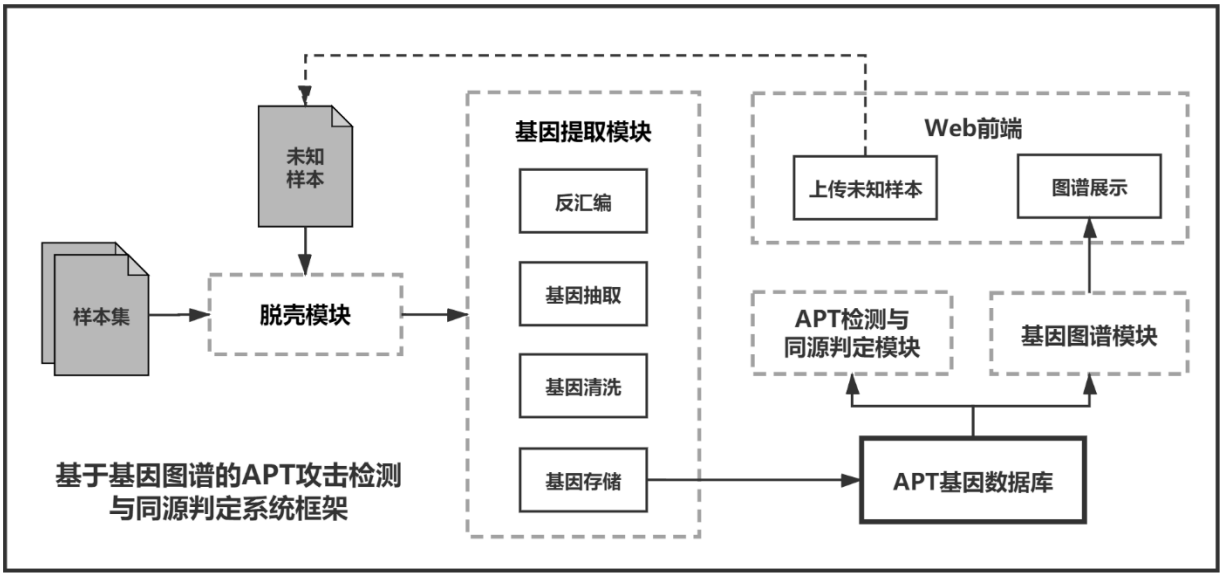

图 1 系统总框架

2.2 系统设计与实现

2.2.1 反汇编

在 X86 平台下使用的汇编指令对应的二进制机器码为 Intel 指令集一 Opcode，Intel 指令手册中描述的指令由 6 部分组成：

（1）Irsinucton Prefixes 指令前缀；

（2）Opcode 指令操作码；

（3）Mode R/M 操作数类型；

（4）SIB 辅助 Mode R/M

（5）Displacement 计算偏移地址；

（6）Immedate 立即数。

指令前缀是可选的，作为指令的补助说明信息存在，主要用于以下 4 种情况。

（1）重复指令，如 REP、REPE\REPZ;

（2）跨段指令，如 MOV DWORD PTR FS:[XXXX], 0;

（3）将操作数从 32 位转为 16 位，如 MOV AX,WORD PTR DS:[EAX];

（4）将地址从 16 位转为 32 位，如 MOV EAX,DWORD PTR DS:[BX+SI].

Opcode 为机器码中的操作符部分，用来说明指令语句执行什么样的操作，如某条汇编语句是 MOV、JMP 还是 CALL。Opcode 为汇编指令语句的主要组成部分，是必不可少的，对 Opcode 的解析也是反汇编引擎的主要工作。每一种操作都会对应一条 Opcode 码，但由于操作数类型不同，所占长度也不相同，因此对于非单字节指令来说，解析一条汇编指令单凭 Opcode 是不够的，还需要 Mode R/M、SIB、Displacement 的帮助，才能够完整地解析出汇编信息。

Mode R/M 用于辅助 Opcode 解释操作数类型。R 表示寄存器，M 表示内存单元。Mode R/M 占一个字节的固定长度，第 6、7 位可以描述 4 种状态，分别用来描述第 0、1、2 位是寄存器还是内存单元，以及 3 种寻址方式。第 3、4、5 位用于辅助 Opcode.

SIB 用于辅助 Mode R/M，计算地址偏移。其寻址方式为基址+变址，如 MOV EAX,DWORD PTR DS:EBX+ECX*2]，其中的 ECX、乘数 2 都是由 SIB 来指定的。SIB 占 I 个字节大小,第 0、I、2 位用于指定作为基址的寄存器；第 3、4、5 位用于

指定作内变址的寄存器；第 6、7 位用于指定乘数，由于只有两位，因此可以表示 4 种状态，这 4 种状态分别表示乘数为 1、2、4、8.

Displacment 用于辅助 SIB，如 Mov EAX.DWORD PTR DS(EBX+EC*2+3] 这条指令，其中的“+3”是由 Displacement 来指定的。

Immediate 立即数用于解释指令语句中操作数为一个常量值的情况。

反汇编引擎通过查表将由以上 6 种方案组合而成的机器指令编码，解释为对应的汇编指令，从而完成了机器码的转换工作。在本项目中使用反汇编引擎 Radare2 实现对脱壳后 APT 样本的反汇编。

2.2.2 基因提取

软件基因的灵感来源于生物基因，它与生物基因的形式以及作用相似，是一段表达软件行为或功能的二进制序列。软件基因的概念建立在基本块概念上。在编译原理中，一个基本块是一段顺序执行的以入口语句开始、以出口语句结束的代码序列，执行该序列时只从入口进入，从出口退出。软件基因是一个或若干个基本块的组合，如果一个基本块的最后一条指令为库函数调用（exit 类除外），则该基本块与地址紧邻的下一基本块合并；当无法继续合并时，得到的代码块为软件基因。基本块与软件基因的示例如图 2 所示。由此定义得到的软件基因中包含库函数调用，

图 2　Radare2 基本块、编译原理基本块、软件基因示例图

这是合理的，因为库函数的功能均为已知，可将其调用视为一条功能复杂的指令；对于功能未知的用户自定义函数，如果将其调用包含在软件基因中，会造成该基因

功能描述困难，故不宜包含在软件基因中。

使用 Radare2“pdbj @@b”命令可获取当前函数的所有基本块，Radare2 中定义的基本块和编译原理中所定义的基本块不同，如图 2 所示，Radare2 的基本块包含了函数调用指令。因此只需根据库函数调用对基本块进行拆分即可得到基因。基因抽取伪代码如表 1 所示。

表 1 基因抽取算法

基因抽取伪代码

输入：二进制可执行文件

输出：提取得到的基因集合 genes

算法过程：

```
function_list ⇐ r2.cmdj('aflj')          # 获取所有函数
basic_blocks = []                        # 基本块
genes = []                               # 基因集合
for function in function_list:
    r2.cmd('s {}'.format(function))      # 跳转到函数位置
    basic_blocks ⇐ r2.cmd('pdbj @@b')    # 获取函数的基本块
for basic_block in basic_blocks:
    gene = []                            # 一个基因
    for code in basic_block:
        if code 为库函数调用:             # 组成了一个基因
            gene.apeend(code)
            genes.append(gene)
            gene = []
        gene.append(code)
    genes.append(gene)
```

由上述步骤得到的基因是完整的汇编代码，为了使后续的基因清洗以及相似性计算等操作更加高效，我们使用 opcode 序列来表示基因，如图 3 所示。汇编语言中一条指令包含操作码（opcode）和操作数，操作数为被操作的地址，可以是内存地址或寄存器等。指令的操作数变化很大，相同的代码片段的编译获得汇编代码后，操作数也可能不同。opcode 序列就是由基因中每条汇编指令的操作码，排列形成。操作码是简单的操作指令，能反映代码的行为模式。黑客软件的实现过程中，编写人员习惯于参考其他代码资源，相互学习，导致代码间存在相同的行为模式，从而形成相同的 opcode 片段。

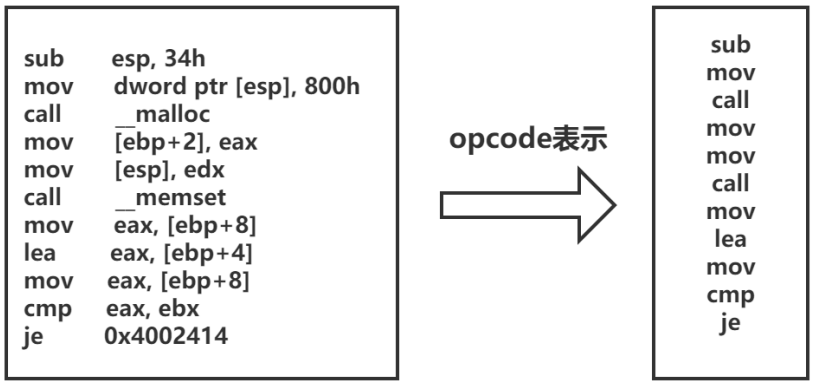

图 3 使用 opcode 序列表示基因

2.2.3 基因清洗

由节 2.2.2 得到的基因还存在大量噪声，存在许多与恶意代码核心行为或功能无关的基因。为了获得能表征某类 APT 特征的基因，采用 TF-IDF 算法，该算法的核心思想是选出在某类样本中频率较高，而在其他类中频率较低的基因作为该类的特征基因。TF-IDF 值求解过程如下：

对于一个 APT 样本中的基因x，计算 IDF 值如下，

$$IDF(x) = \log\frac{N+1}{N(x)+1} + 1$$

其中N为 APT 类型总数，$N(x)$为样本集中包含基因x的样本数，

计算 TF 值如下，

$$TF(x) = \frac{A\,(x)}{A}$$

$A(x)$，包含基因x的该类 APT 样本数；A，该类 APT 样本总数。

从而该样本中基因x的$TF-IDF$值为：

$$TF-IDF(x)=TF(x)*IDF(x)$$

利用以上公式求解得到的 TF-IDF 值越高，说明基因 x 越能表征该类 APT 独有的特征。我们对每个样本求出其所有基因的 TF-IDF 值，并排序，选取排名前 400 的基因作为该样本的特征基因。

2.2.4 基因存储和基因图谱

由节 2.2.3 获得的基因能表达某类 APT 的特征，我们需要存储这些基因，以构建 APT 基因数据库。我们使用 Neo4j 数据库存储基因，Neo4j 是一个高性能的非关系型图形数据库，它将结构化数据存储在网络上而不是表中。我们需要存储的基因序列长度各不相同，且每个样本所包含的基因数也不同，使用关系型数据库存储较困难，故使用图数据库存储。Neo4j 包含两个重要的对象，结点与关系，在对象中能自定义若干键值对用于存储对象信息。APT 基因存储结构如图 4 所示，数据库中包含两类结点：样本结点和基因结点。样本结点存储了样本 Md5 值和所属 APT 名称，基因结点存储了基因序列。数据库中包含两种关系，样本与基因间存在 CONTAINS 关系，代表样本包含该基因；样本与样本之间存在 SHARE 关系，代表两个样本共享了基因，SHARE 关系中保存了共用基因数量。如图 4 中样本 1 与 2 都拥有基因 3，因此两个样本存在 SHARE 关系。

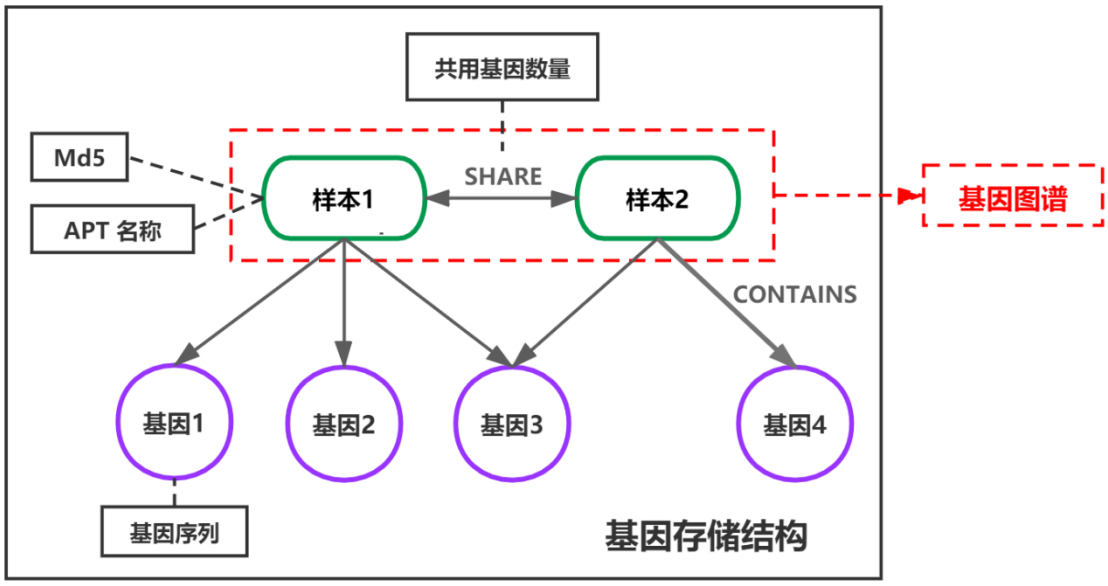

图 4 基因存储结构

由此得到的 APT 基因数据库中样本共用基因情况较多，共用基因数量越大，代表两个样本相似度更高，即恶意代码存在相似的功能或行为。我们选取基因数据库中所有样本结点以及样本之间的 SHARE 关系，构成基因关联图谱，该图谱清晰地展示了所有 APT 样本间地关联情况，同时也能表达不同 APT 组织间的关联性。

2.2.5 检测模型

基于 2.2.4 构建的 APT 基因数据库，我们建立了 APT 检测框架，如图 5 所示。我们使用 UPGMA 聚类算法对数据库中所有基因进行聚类，其中基因距离定义使用 Smith-Waterman 算法，由此得到 512 个基因聚类，根据该聚类构造长度为 512 的样本向量，最后使用机器学习模型进行检测。以下将详细介绍样本向量构造方法以及机器学习模型。

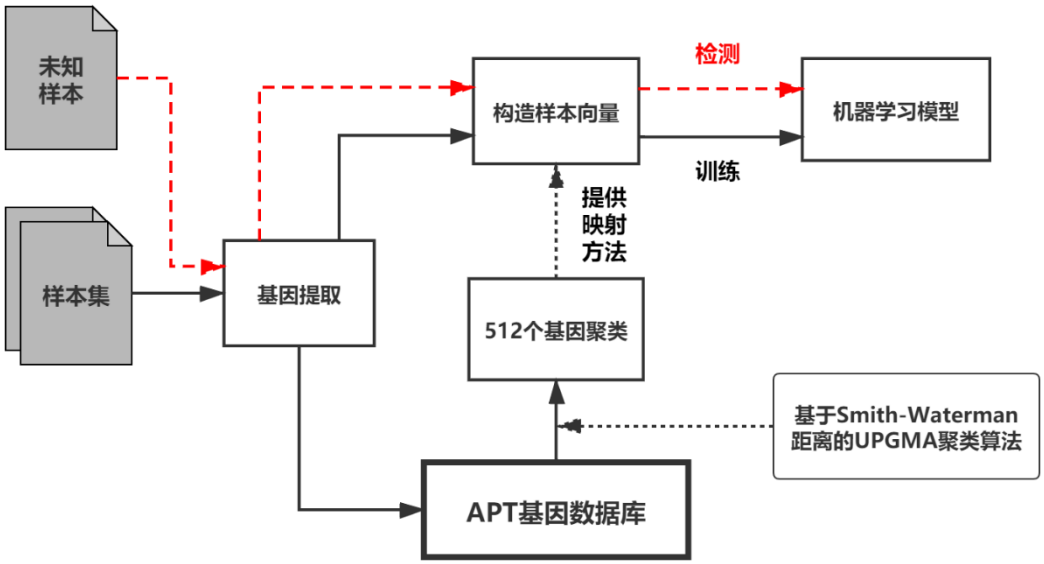

图 5 APT 检测框架

由于功能相似基因一定程度上也会在形态上呈现相似的序列，因此高度相似的基因可认为是功能基本一致的基因，可归为一个类别。基于软件基因这一特性，我们对基因集合进行聚类，实现降维。聚类算法的关键部分是距离定义，我们使用 Smith-Waterman 算法定义两个基因之间的距离。Smith-Waterman 算法是最长公共子序列问题的一种解法，采用动态规划的思想。设计思想如下：

给定两个 opcode 表示的基因序列 A=a1, a2, …, an 和 B=b1, b2, …, bm，其中 ai 与 bi 均为操作码，从两个序列的起始元素开始匹配，使用矩阵记录 A 前 i 个元素构

成的子序列与 B 前 j 个元素构成的子序列的最高分值，计算公式为：

$$H_{ij} = \max\{H_{i-1,j-1} + \delta(i,j), \max_{k\geq 1}\{H_{i-k,j} - W_k\}, \max_{i\geq 1}\{H_{i,j-k} - W_l\}, 0\}$$

其中$1 \leq i \leq n, 1 \leq j \leq m$.$W_k$为长度 k 的子序列不匹配扣除的分值权重。$\delta(i,j)$是序列中$a_i$和$b_j$匹配成功时加上的分值，当该值设定为 1 时，$H_{nm}$恰好等于序列 A 和 B 的最长公共子序列的长度。序列 A 和序列 B 的距离定义为：

$$d_{AB} = \max\{n,m\} - H_{nm}$$

基于基因距离度量方法，我们采用了优化的 UPGMA 聚类算法对相似的基因进行聚类，再映射到样本向量中，实现降维，同时聚类使得各维度的相关性降低，符合机器学习对样本向量的要求。

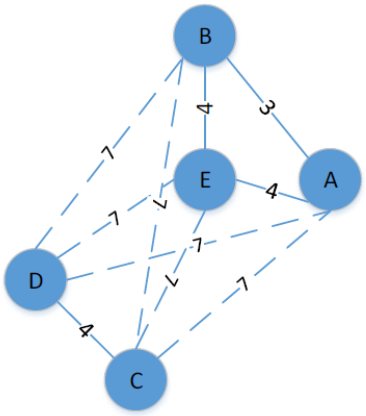

图 6 UPGMA 算法示例图

UPGMA 算法每次选取所有基因结点中距离最近的两个进行合并，在图 6 中，AB 边距离为 3，在图中最小，故选择 AB 进行合并，形成一个聚类（视为一个新结点），该聚类到其他结点的距离为合并前两个基因各自到其他基因距离的平均值。直到结点数到达最初设定的聚类数目时停止。我们设置聚类数目为 512，由此可得到 512 个聚类。对于一个样本的所有基因，检查其是否存在某一个聚类中的基因，若存在，则对应向量位置置 1，否则置 0，从而构造长度为 512 的样本向量。

样本向量构造完成后，需选择合适的机器学习算法。针对一维的向量，可使用基于全连接神经网络的分类器，如图 7 所示，长度维 512 的样本向量作为输出层输入，经过隐藏层后在输出层输出一个长度为 2 的向量，分别作为 APT 样本与非 APT 样本的分数，实现 APT 检测器。图 8 是全连接网络的基本单元结构，其中：

- a1~an 是输入向量的 n 个分量
- w1~wn 为各分量权重

• b 为偏置量(bias)

• 激励函数 f 为非线性函数，常用的有 Sigmoid，Tanh，ReLU

• y 为神经元输出，若激励函数为 f，y 计算公式如下

$$y = f(\sum_{k=1}^{n} a_k w_k + b)$$

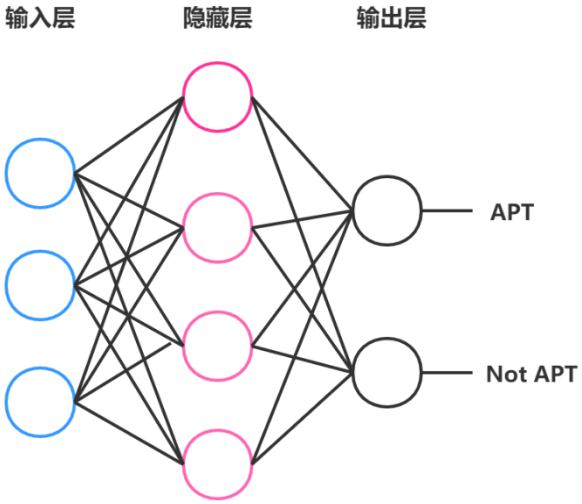

图 7 神经网络示意图

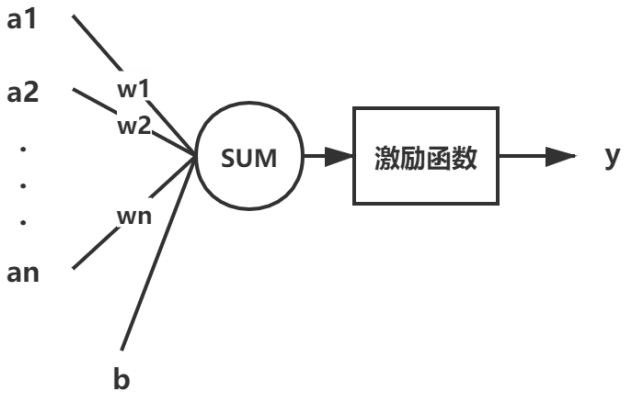

图 8 神经网络基本单元

假设损失函数为$E(y)$，记 $z = \sum_{k=1}^{n} a_k w_k + b$ ，则损失函数对各权重分量的偏导为：

$$\frac{\partial E}{\partial w_i} = E'(y) * f'(z) * a_i$$

由此可按如下公式对各权重进行更新，优化损失函数的值。lr 为学习率。

$$w_i \leftarrow w_i - lr * \frac{\partial E}{\partial w_i}$$

此外，我们也使用了传统机器学习算法随机森林作为检测模型进行实验。算法流程图如图 9 所示。

随机森林实质上由很多决策树组成。一个决策树在构建时，通过将数据划分为具有相似值的子集来构建出一个完整的树。决策树上每一个非叶节点都是一个特征属性的测试，经过每个特征属性的测试，会产生多个分支，而每个分支就是对于特征属性测试中某个值域的输出子集。决策树上每个叶子节点就是表达输出结果的连续或者离散的数据。随机森林算法随机构建一个森林，由相互不关联的决策树组成，各决策树独立地学习并作出预测，这些预测最后结合成一个单一的预测作为结

果。

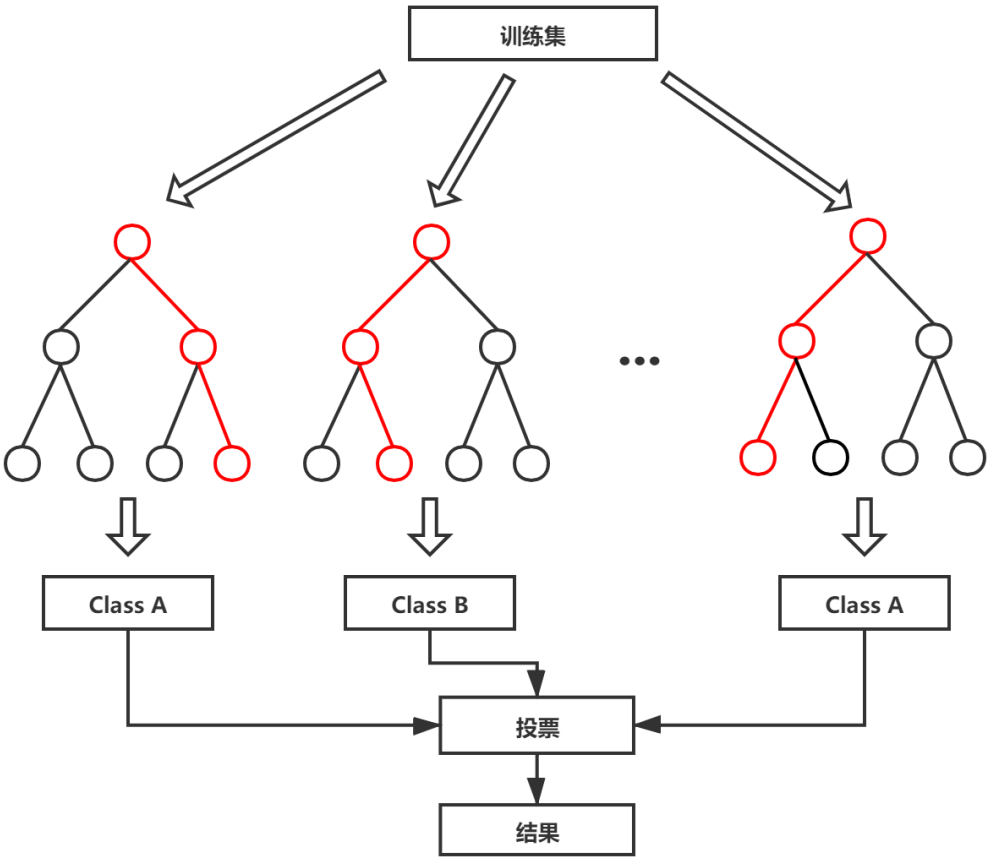

图 9 随机森林算法流程图

2.2.6 同源判定模型

由节 2.2.4 中构建的基因关联图谱基于样本间的基因共用情况。来源于同一个 APT 组织的恶意样本往往具有部分相似的行为或功能，基因能够表达行为，因此属于相同组织的样本的基因共用数量往往较大。基于这一特点，我们使用 KNN 算法进行 APT 同源判定。

KNN 算法思路是：如果一个样本在特征空间中的 K 个最邻近的样本中的大多数属于某一个类别，则该样本也划分为这个类别。在图 10 中，假设 K 取 5，与点 Xu 最近的 5 个点中有 4 个为ω1 类，只有 1 个为ω3 类，故该点被划分到ω1 类。

KNN 算法依赖于 K 的取值以及距离的定义。当 K 值较小时，噪声的存在会对预测产生较大影响；当 K 值较大时，与输入样本较远的样本也会对预测起作用，使预测发生偏差，此外，若原始样本分布不均匀，也会造成较大的误差。K 值的选取需要实验进行测试，找到预测准确率与 K 值的关系。距离用来衡量两个样本间的邻

近关系，距离的度量方法常用的有欧几里得距离、余弦值、相关度等。在本例中，我们使用共用基因数来衡量邻近关系，若共用基因数越大，则认为两个样本更近，这也与基因关联图谱相契合。

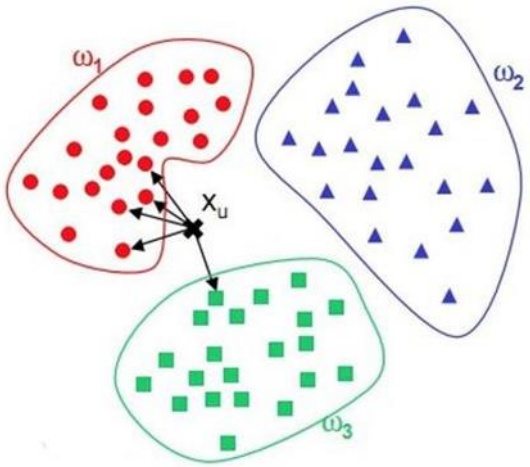

图 10　KNN 算法示例图

2.2.7 Web 页面

2.2.7.1 Web 设计原理

本项目被部署于云服务器上，并设计了相应的基于 Flask 的 Web 页面展示窗口。本项目的 Web 展示页面主要分为三个模块：一是基于 python 的 neo4j 数据库读取模块；二是基于 echarts.js 的基因图谱绘制部分；第三是在线 APT 检测及同源判定模块。

在第一模块——neo4j 数据读取模块，我们使用了关联 neo4j 和 python 之间的管道工具——py2neo，其具有快速，便捷调取数据库数据的特点。其提供的语句形式可以直接以 Cypher 查询语句的形式，查询相应数据并以 json 数据格式返回结果。通过 python 字符串处理结果后的数据，将放置于\graph 目录下待用。

第二模块——基于 echarts 的基因图谱绘制模块。本部分所使用的 Web 框架是 Flask：相比于 django，它更轻量化，也更加方便，易于承载我们的基因图谱。首先，网页会调用来自于 python 读取并分化好的数据，对其进行进一步的类目划分，包括使同一家族的基因样本聚在一起，并对每一个家族绘制不同颜色并分配位置。

其后，他会读取家族之间的联系大小，并将其求和后，以不同宽度的连线形式直观表现其相关程度。

第三个模块是在线 APT 检测及同源判定模块。我们设计了一个在 Web 框架上增设了一个上传连接，客户可以通过其上传存疑文件，之后 flask 会调用我们自己设计的函数接口在线对存疑文件进行检测，并给出其属于不同家族的概率。

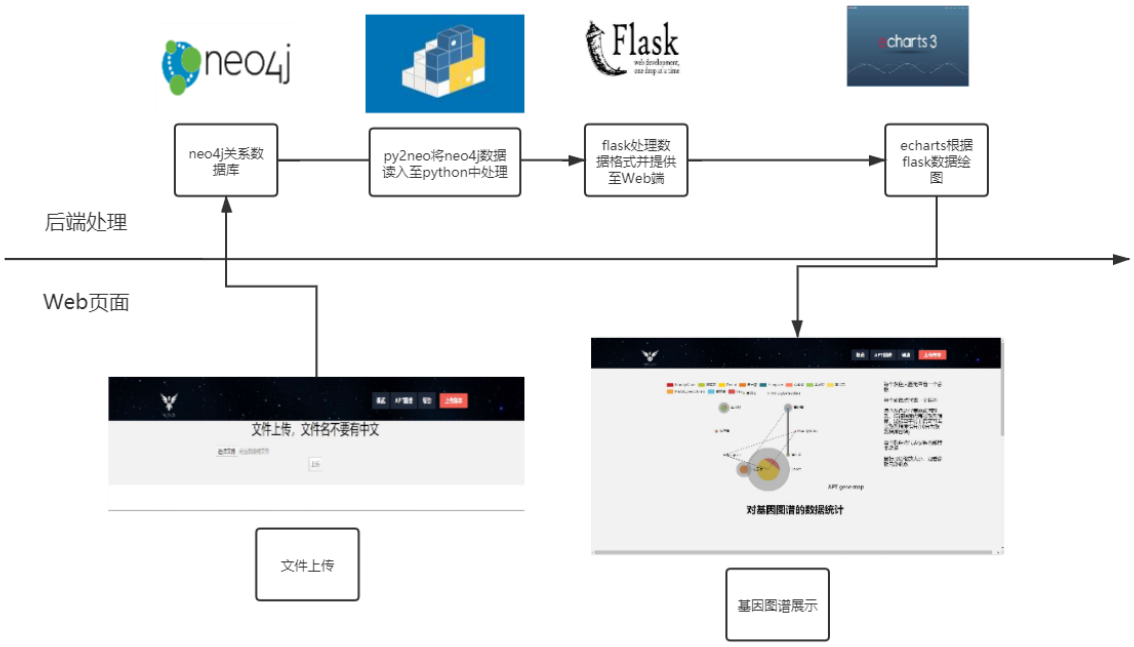

图 11 Web 端基因图谱绘制流程

2.2.7.2 Web 客户端操作

使用者在登录 Web 页面后，会进入 Web 界面的主页，即概览部分。概览部分首先简单介绍了 APT 家族的定义；其后，列举了 10 种较为常见的 APT 家族简介，每一个家族都超链接到了权威组织中对该组织的分析界面；同时，在家族简介后方，还附有本系统中所有的恶意样本的 md5 值，并链接到了 virustotal 网站中以供查看。

单击右上角的基因图谱标签后，其中每一个大圆都代表着一个恶意软件家族，其大小代表了各个家族中含有样本的数量，将鼠标防止于各个家族所在的大圆之上，可以看到各个家族更加详细的信息，大圆内的小圆则是代表了一个个不同的恶意样本。家族内部的线代表了同一家族不同样本之间的联系，而家族之间的边则代表了不同家族之间的关联，而边越粗，代表其相互之间的关联越强。同时，图谱上方的不同颜色的方块代表了不同家族的标签，使用者可以通过单击各个标签以隐藏

或显示各个家族。基因图谱下方则是两个环形图，分别统计了各个家族中样本的数量及样本中新旧样本（原本拥有的样本和用户提交的样本）的数量。更下方是我们对现有基因图谱的一些分析。

在基因图谱的标签后，则是帮助标签，其有三个小标签，分别是如何使用，系统介绍和关于我们。如何使用中超链接了前两个标签，并简要的概括了其内容；系统介绍中则是绘制出了我们的系统框架；而“关于我们”中则是列举了我们小组的三维成员。

同时，本 Web 页面还设计了恶意软件上传连接：通过单击“上传连接”标签，使用者就会跳转到上传页面，并可以上传其恶意样本。经过一段时间的等待后，系统就会返回对其的分析，并打印出结果。

第三章 作品测试与分析

基于前文所述 APT 基因图谱构建以及 APT 检测方法，我们进行了实验与测试。实验所使用的恶意代码数据集如表 2 所示。我们使用 265 个来自 10 个 APT 组织的样本构建 APT 基因数据库并绘制 APT 基因关联图谱，并将这些 APT 样本以及 745 个普通恶意样本用于检测模型的训练。我们使用 80 个来自 6 个 APT 组织的样本以及 200 个普通恶意样本进行系统测试，首先利用训练好的 APT 检测模型进行预测，并测试模型的性能。最后将检测为 APT 的样本加入关联图谱，并展示图谱的变化。

表 2 实验与测试数据集

样本类型	所属 APT 组织	构建库样本量	测试样本量
	Donot	110	50
	Hangover	6	
	MalluCyberSoldiers	7	
	MuddyWater	6	
	双尾蝎	10	2

APT 样本	毒云藤	47	10
	海莲花	13	2
	盲眼鹰	26	6
	蔓零花	9	
	黑凤梨	31	10
	总计	265	80
非 APT 样本		745	200

3.1 基因图谱展示

3.1.1 APT 内部基因关联图谱

根据 2.3.3 所述方法，我们绘制了各个 APT 组织内部的基因关联图谱，图 12 和图 13 分别为海莲花和双尾蝎组织内部的关联图谱，可见来源于同一个 APT 组织的

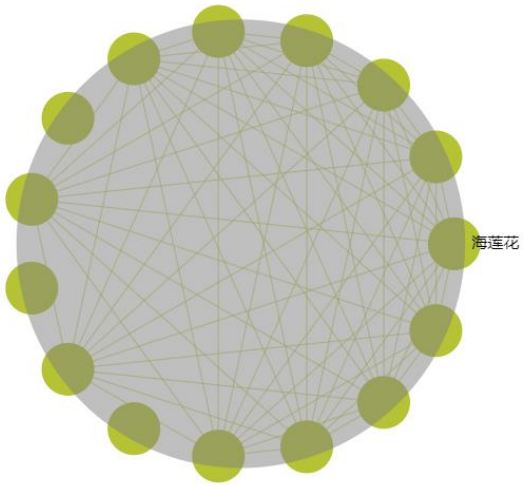

图 12 海莲花组织基因图谱

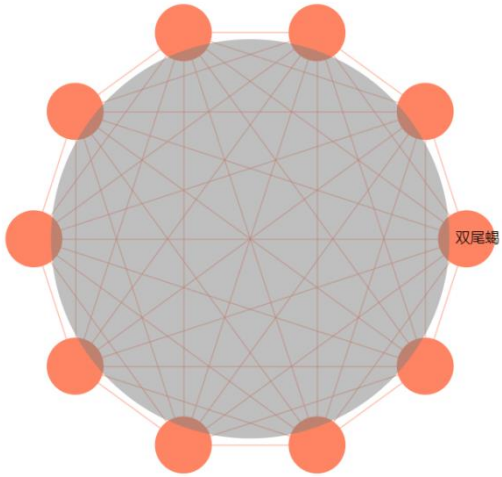

图 13 双尾蝎组织基因图谱

恶意代码之间存在明显的基因共用现象，说明这些代码互相借鉴的部分较多，且含有很多相似的行为与功能。

3.1.2 APT 间基因关联图谱

在 APT 内基因关联图谱的基础上，我们绘制了 APT 组织之间的关联图谱。对于两个不同的 APT 组织，若存在样本共享基因，则在两个 APT 组织间连线，线的粗细反映了共享基因的样本数大小。完整的基因关联图谱如图 14 所示，可见 Donot、Hangover、MuddyWater 组织间存在基因关联性，毒云藤、盲眼鹰、海莲花也存在基因共用的情况，且盲眼鹰与海莲花的基因共用程度较高。

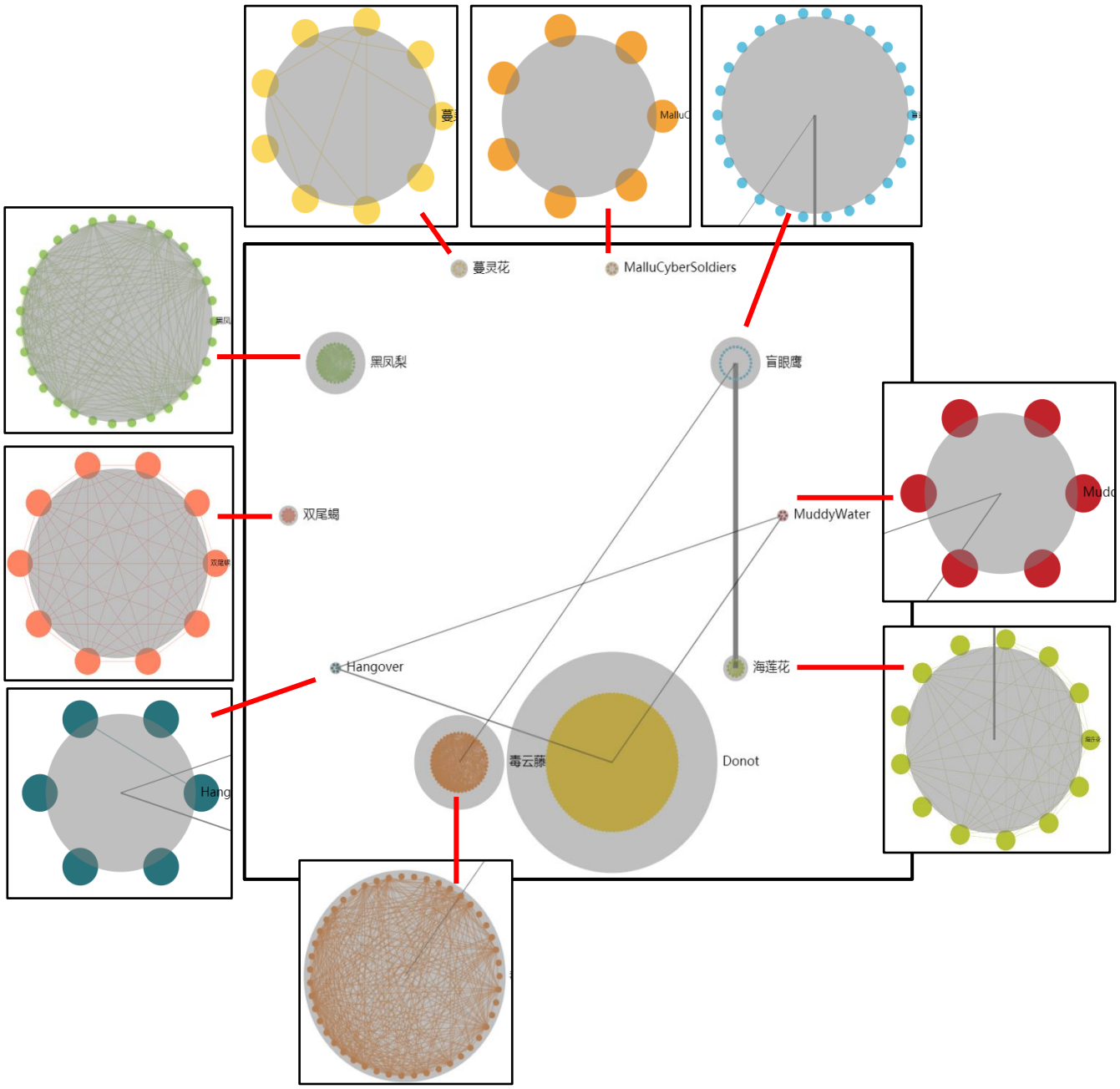

图 14　APT 间基因关联图谱

3.2 APT 检测与同源判定

3.2.1 APT 检测

基于 2.2.5 所述检测模型，我们使用表 1 所示数据集进行模型训练。训练集共 1010 个样本，包含 265 个 APT 样本和 745 个非 APT 样本；测试集共 280 个样本，包含 80 个 APT 样本和 200 个非 APT 样本。测试指标包括准确率(Accuracy)、精确率(Precision)和召回率(Recall)，指标计算公式如下。

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

其中 TP,TN,FP,FN 为表 3 中所示对应的样本数量。

表 3 混淆矩阵

Confusion Matrix		真实值	
		APT	非 APT
预测值	APT	TP	FP
	非 APT	FN	TN

我们首先使用随机森林算法在数据集上进行训练，模型测试结果如表 4 所示。

表 4 模型测试结果

准确率	精确率	召回率
96.8%	97.5%	91.8%

可见模型对 APT 与非 APT 的分类有较高的准确率，且针对 APT 样本的检测有较高的精确率及召回率。

其次，我们构建了神经网络分类器进行实验，网络结构如图 15 所示，训练环境以及所使用的深度学习框架如表 5 所示。

表 5 实验环境配置

操作系统	Linux 3.10
GPU	GeForce GTX 1080Ti
框架	Pytorch 1.1

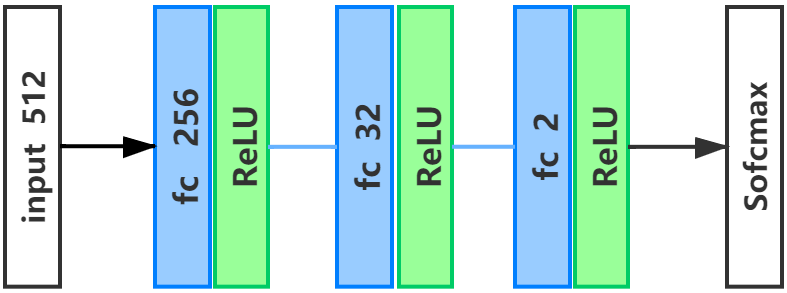

图 15 神经网络结构

训练参数如表 6 所示，使用 Adam 优化器，并设置初始学习率为 10^{-4}，损失函数使用交叉熵(CrossEntropyLoss)。训练过程中在验证集准确率与损失函数变化曲线如图 16 所示，训练 25 轮后在验证集上的正确率达到最高，为 97.5%，损失为 0.35，

表 6 训练参数及结果

训练参数			训练结果				
优化器	损失函数	学习率	轮次	损失值	准确率	精确率	召回率
Adam	交叉熵	10^{-4}	25	0.35	97.5%	97.9%	92.5%

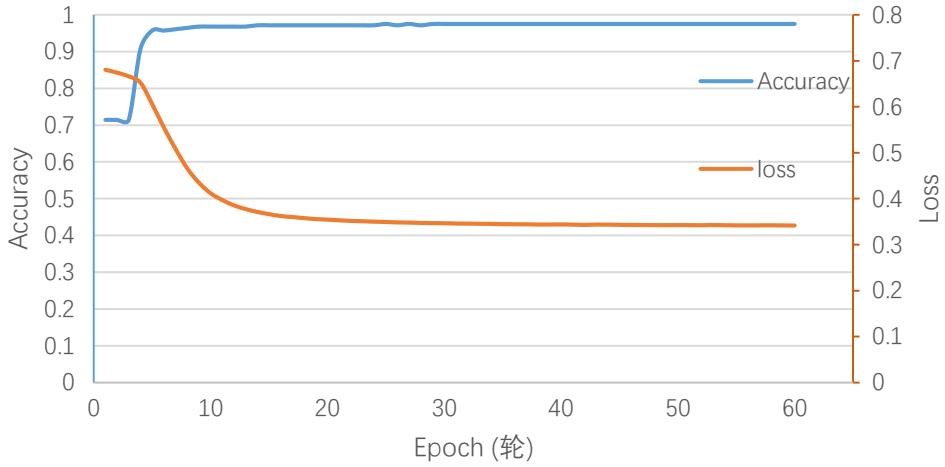

图 16 验证集准确率与损失变化曲线

训练过程中在验证集精准率与召回率变化曲线如图 17 所示，训练 25 轮后在验证集上的精准率达到最高，为 97.9%，召回率为 92.5%。

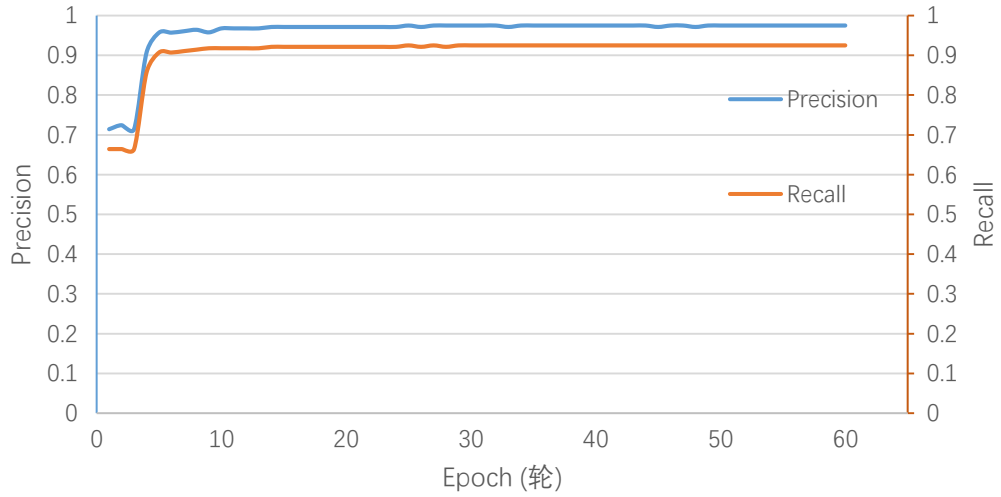

图 17 验证集精准率与召回率变化曲线

由此可见，使用神经网络方法得到的测试集准确率(97.5%)，精准率(97.9%)以及召回率(92.5%)比随机森林得到的更高，最终将该神经网络模型部署至系统中。

3.2.2 APT 同源判定

基于 2.2.6 所述同源判定模型，我们利用表 1 中的 80 个 APT 测试样本，在 265 个 APT 样本基因形成的基因库上进行同源判定测试。KNN 算法的核心参数是 K 的

取值，我们测试了 K 值对同源判定准确率的影响，如图 18 所示。

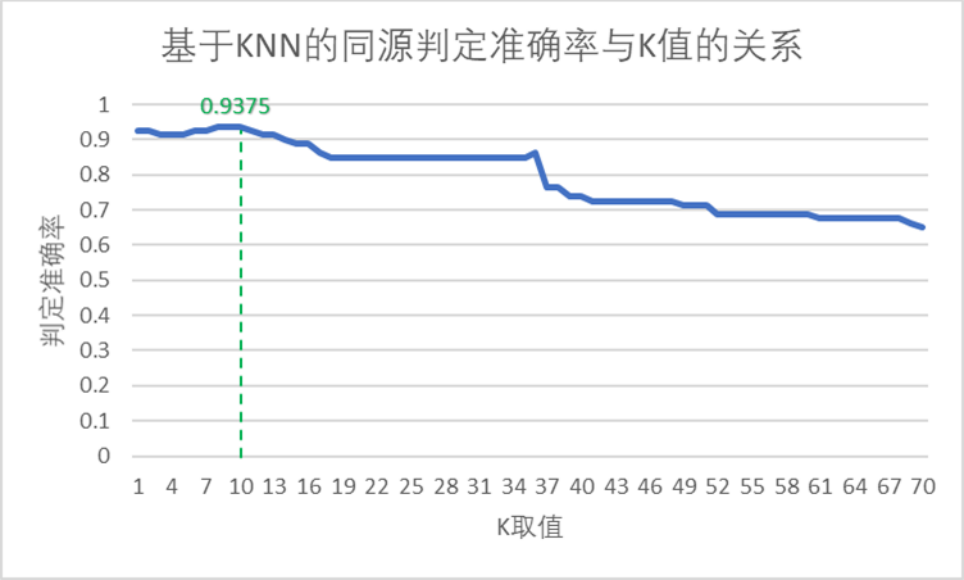

图 18　KNN 同源判定准确率与 K 的关系

当 KNN 算法中的参数 K 选取为 10 时，同源判定准确率达到最高，为 93.75%。误判的样本主要来源于双尾蝎，这是由于 KNN 算法的分类判定由最近的样本中频率最高的类别决定，若在原始数据中每类样本数极少，会造成该类样本判定准确率较低。此外，K 值越大，同源判定准确率持续降低，这也是因为基因库中某些 APT 样本数相对偏少，就算 K 个最邻近的样本中包含了该类的全部样本，也比其他类别少，造成准确率的降低。

3.2.3 APT 家族更新后的基因图谱展示

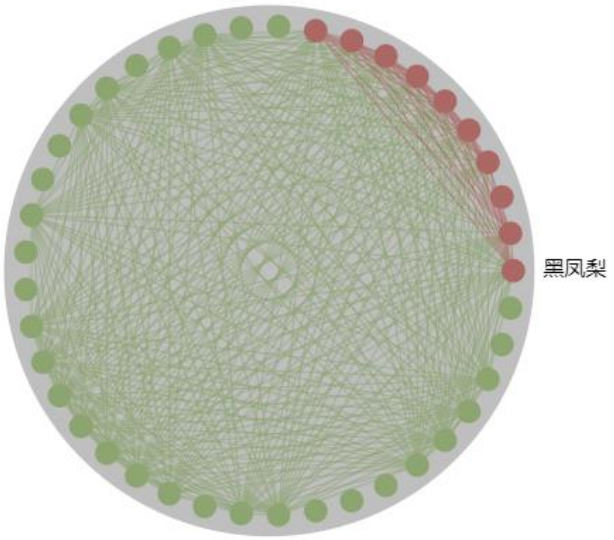

图 19 更新后的“黑凤梨”恶意家族

所有新加入的节点，其 type 属性都为’new’。在绘制新图谱的时候，所有 type 为 new 的样本将在布局结束后，加入新的一个 new 的类目中，绘制基因图谱时，他们将在布局不变的情况下，被分配到同一个新的颜色。如图 19 所示，绿色的样本点是原本的旧的样本点，而红色的样本点则是更新后新的“黑凤梨”家族的样本点。

3.3 结果分析总结

通过对绘制出来的 apt 基因图谱和检测与同源判定结果的分析，我们可以得出如下结论：

同一家族内部的 APT 样本具有很强的关联性，这体现了 APT 组织内的协同性。同时，由于 APT 攻击的复杂性，一次完整的 APT 攻击需要各部分的协调配合，这也使得 APT 家族内部的样本之间需要存在较强的关联性。

同时，可以看到不同的 APT 家族之间的关联性较弱，这是由于 APT 攻击多由各个组织独立制作，基本不存在共享代码。基于 APT 的这一特性，我们的同源判定

模型有较好的表现，能够准确地判定 APT 样本属于哪一个 APT 组织。通过对 APT 样本的同源判定，我们可以通过对该样本隶属 APT 组织的了解来指定针对性的检测、防御措施。

特别的，个别 APT 家族之间存在较强的关联，表明这些 APT 家族对应的 APT 组织之间可能存在合作与协同性。可将这些 APT 组织合并在一起研究它们共同的特征。

3.4 产品对比分析

与现有的APT检测产品相对比，我们的项目拥有诸多优势，下面通过与国内具有代表性的信息安全企业之一——启明星辰的APT检测产品的详细对比来说明优势。

启明星辰APT检测系列，是一款针对恶意代码等未知威胁具有细粒度检测效果的专业安全产品，可实现包括对：未知恶意代码检查、嵌套式攻击检测、木马蠕虫病毒识别、隐秘通道检测等多类型未知漏洞（0-day）利用行为的检测。

系统采用动静结合的检测方式，使用虚拟机环境运行被检测文件，检测文件打开后的各种行为和系统环境等以确定文件是否具有恶意行为，动态分析的特点是准确率高、误报率低。同时通过一定的特征比对或算法对被检测文件的二进制内容进行匹配或计算的检测方法，静态检测并不真实的运行被检测文件。静态检测的方法有很多种，天阗APT检测系统使用虚拟Shellcode执行、暴力搜索隐藏PE等多种方式对被检测文件的文件内容进行静态检测，以此来确定文件是否为恶意文件。

只需要为系统添加入侵检测与管理系统功能模块，就可以实现已知威胁加未知威胁的全面检测，包括但不限于：病毒、蠕虫、木马、DDoS、扫描、SQL注入、XSS、缓冲区溢出、欺骗劫持等攻击行为以及网络资源滥用行为、网络流量异常等威胁具有高精度的检测能力。

相比于启明星辰APT检测系统，我们的项目存在以下两大优势：

第一，更加有效的的检测方式。APT样本往往具有较强的反动态分析能力，导致虚拟环境下运行无法触发真实的样本行为。同时，启明星辰APT检测采用特征比对技术，虽然速度快，但是攻击者可以通过添加花指令、输入表免杀等技术使得特征匹配失效从而绕过检测。而我们的检测方式则是直接通过反汇编提取样本中具有特定功能的汇编代码片段，即软件基因，挖掘APT样本更加底层的特征，而这些特征是难以通过常

规的混淆手段加以隐藏的。与常规静态分析相比，我们的方法虽然牺牲了一些时间成本，但大大提高了对未知样本、变种样本的检测能力以及抗混淆能力，同时检测准确率、误报率不输于常规动态分析。

第二：同源判定的能力。启明星辰APT检测系统能够对APT样本按恶意行为进行分类，但不具有对APT样本进行同源判定、挖掘APT家族的能力。我们的项目通过绘制APT家族基因图谱，可有效地对APT样本进行同源判定，发现样本隶属的APT家族，以及发现新的APT家族。通过对APT样本的同源判定，我们可以对APT样本进行溯源，同时不断丰富对各APT组织的认识，有助于对APT组织的分析研究。

除此之外，我们还将本项目与国内其他多家企业的APT检测产品做了对比，如蓝盾、明朝万达、科来、安恒信息、阿里云等。这些企业的APT检测产品无一例外地使用基于动态/静态分析的方式进行APT检测，都具有上文提到的传统分析方式存在的缺陷。并且这些产品都不具有APT同源判定的能力，无法对理解APT组织攻击行为的协同性做出贡献。个别企业使用了全流量审计的方式分析流量中的可疑行为，以此检测APT攻击。这种方式需要计算机强大的存储能力和运算能力，成本较高，不适合大范围推广。

总的来说，我们的项目提供了一种更加有效的APT检测方法，解决了目前常规动、静态分析存在的缺陷。同时从二进制汇编代码层挖掘APT组织内的代码重用以及 APT 组织间的代码共用，发现APT样本之间的关联性，对APT样本进行溯源，能够更好理解当前世界 APT 组织的恶意攻击行为的协同性，这是现有产品所不具备的功能。

第四章 创新性说明

(1)基于基因图谱的 APT 攻击检测

直接从汇编代码级别挖掘 APT 样本的基因，作为 APT 代码的特征，解决了动态分析可能无法触发代码真实行为的难题，同时提高了对变种代码、新型代码的检测能力。在实际测试中，检测准确率达到了 92%，即绝大部分 APT 代码都可以被成功检测，证明了本系统有较强的现实 APT 攻击检测能力。

(2)利用基因图谱实现样本分类溯源

利用基因图谱中的 APT 家族，我们实现了对 APT 样本的分类工作。由于使用了 KNN 算法，用公用基因的数量来定义样本之间的距离，与传统基于特征码的识别方式相分类效果更好，抗变种能力更强。通过对 APT 样本的同源判定，我们实现了对 APT 样本的溯源工作，即该 APT 样本隶属于哪一个知名 APT 组织。以此我们可以不断丰富对 APT 组织的了解，同时可根据不同 APT 组织的特点制定针对性的防御措施。更进一步，可以通过对大量样本的检测，挖掘未知的 APT 恶意家族，从而发现新的 APT 组织并研究其攻击特点。

(3)基因图谱的可扩展性

在一个样本被判定为 APT 攻击之后，它将加入到基因图谱中，成为新的节点。通过这样的方式，我们可以在检测中不断丰富、完善基因图谱，从而形成正反馈，提供更完整的展示 APT 关联以及检测 APT 攻击样本的能力。同时，这样的方式能够使得不断吸收最新 APT 样本的基因特征，从而增强对变种代码、新型代码的检测能力。

(4)挖掘APT家族间的独立性与关联性

我们收集了多达20个APT家族的样本，绘制了首个APT家族基因图谱。我们发现APT家族间往往具有较强的独立性，家族之间的代码关联不强。同时，我们发现盲眼鹰和海莲花APT组织间存在较强的关联，说明这两个APT组织间存在代码共享和协同。

第五章 总结

当今计算机互联网技术迅速发展，人与人之间的交流变得越来越便利，但随之而来的，是各种新型网络攻击的出现。其中 APT 攻击作为一种新型的网络攻击方式，对企业和国家的网络安全造成了巨大威胁。

传统的 APT 攻击检测技术使用两种方法。一是特征码匹配技术，通过人工确定恶意代码，并手工确定其代码中的共性字节片段。但该技术有诸多局限性，例如对新型恶意代码样本识别能力弱；特征片段与恶意代码的行为没有必然联系，不含语义的字节片段会干扰恶意代码的识别等。二是动态沙箱检测技术，但新型恶意代码往往具有规避沙箱分析、隐藏自身真是行为的能力。因此，本项目针对传统特征码匹配技术泛用性差的问题，以及动态分析难以触发样本真实行为的问题，参考生物学上的基因技术，基于对恶意软件间关联性的绘制，提出了一种通过基因图谱识别恶意软件的方法。

本项目旨在实现绘制一个大型 APT 攻击样本基因图谱，提取出各种 APT 代码之间的基因特点，并以之检测新的可能的 APT 攻击。项目分为 3 个过程，包括对已有的恶意软件脱壳，对脱壳后的软件划分基因，并对划分出的基因绘制基因图谱，以及对新的恶意软件与基因图谱之间的关联技术。其中，基因图谱能够有效地标识出 APT 样本基因之间的特点，并以之检测出新的 APT 攻击，并将新样本加入基因图谱，在检测中不断丰富 APT 基因图谱，提供更完整的展示 APT 关联以及检测 APT 攻击样本的能力。

经过对已有脱壳APT样本的基因划分与基因图谱绘制，并以测试的样本进行测试，可以判断出此项目基本实现了设计目标，能够较为有效的对现有的样本做出检测。但同时，此设计中的脱壳系统对少部分的软件无法进行有效的脱壳，因此难以对其进行检测。因此，在未来，项目成员将进一步对基因脱壳部分进行改进，考虑更加有效地完成脱壳工作，以增加检测程序的泛用性。此项目相信能在APT攻击相互关联性日益增大的今天，提供一种新的APT攻击检测方向，增加对APT攻击的检测力度。

为扩大产品的受众，方便用户上手和操作，本产品为用户提供了友好的 Web 操作页面。Web 页面在设计时主要考虑两大功能：可交互的基因图谱的展示，以及对

用户上传样本的检测和分类。用户可通过与基因图谱的交互，了解 APT 组织内和组织间的关联性，同时可以上传样本，检测是否属于 APT 攻击，以及属于哪一家族。

参考文献

[1] Chen Y.H: Research on Bioinformatics-Inspired Malware Detection Model Strategic Support Force Information Engineering University 2019

[2] Zhao B, Shan Z, Liu F, Zhao B, Chen Y.H, Sun W.J.(2019): Malware homology identification based on a gene perspective

[3]Huang W, Stokes J W. MtNet: A Multi-Task NeuralNetwork for Dynamic Malware Classification New York: Springer, 2016: 399-418.

[4]Qiao Y.C, Jiang Q, GU L, et al. Malware Classification Method Based on Word Vector of Assembly Instruction and CNN [J]. Netinfo Security, 2019, 19(4): 20-28.

[5]Chen Y, Narayanan A, Pang S, et al. Malicioius Software Detection Using Multiple Sequence Alignment and Data Mining[C]// IEEE, International Conference on Advanced Information NETWORKING and Applications. IEEE, 2012:8-14.

[6]Cen L, Gates C S, Si L, et al. A Probabilistic Discriminative Model for Malware Detection with Decompiled Source Code[J]. IEEE Transactions on Dependable & Secure Computing, 2015, 12(4):400-412.

[7]Itay C & Omri B.B(2019): Mapping the connections inside Russia' APT Ecosystem. https://research.checkpoint.com/2019/russianaptecosystem/

[8]Ishai R, Guillaume S, Eli D: DeepAPT: Nation-State APT Attribution Using End-to-End Deep Neural Networks. ICANN 2017, Part II, LNCS 10614, pp. 2017：91–99.

[9]Nari S, Ghorbani A A. Automated Malware Classification Based on Network Behavior[C]//IEEE. Proceedings of the 2013 International Conference on Computing, Networking and Communications (ICNC), January 28-31, 2013, San Diego, California, USA. New York: IEEE, 2013: 642-647.

[10]Park Y, Reeves D S, Stamp M. Deriving Common Malware Behavior through Graph Clustering[J]. Computers & Security, 2013, 39(6): 419-430.

[11]Pascanu R, Stokes J W, Sanossian H, et al. Malware Classification with Recurrent networks[C]//IEEE. Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Apirl 19-24, 2015, South Brisbane,Queensland, Australia. New York: IEEE, 2015: 1916-1920.

[12]Giannela C, Bloedorn E. Spectral Malware Behavior Clustering[C]//IEEE. 2015 IEEE International Conference on Intelligence and Security Informatics (ISI), May 27-29, 2015,Baltimore, MD, USA. New York: IEEE, 2015: 7-12.

[13]Garbervetsky D, Zoppi E, Livshits B. Toward full elasticity in distributed static analysis: the case of callgraph analysis[J]. 2015.

[14]Miller B, Kantchelian A, Tschantz M C, et al. Reviewer Integration and Performance Measurement for Malware Detection[M]// Detection of Intrusions and Malware, and Vulnerability Assessment. 2016.

[15]David, O.E., Netanyahu N.S.: DeepSign: deep learning for automatic malware signature generation and classification. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN), pp. 2015: 1-8

[16]Kwon B J, Mondal J, Jang J, et al. The Dropper Effect:Insights into Malware Distribution with Downloader Graph Analytics[C]// Acm Sigsac Conference on Computer & Communications Security. 2015

[17]Xabier U, Davide B, Igor S, Pablo G: SoK: Deep Packer Inspection: A Longitudinal Study of the Complexity of Run-Time Packers IEEE Symposium on Security and Privacy. 2015

[18]Alessandro M, Simone A, Xabier U, Alessio M, Davide B: Prevalence and Impact of Low-Entropy Packing Schemes in the Malware Ecosystem. Network and Distributed Systems Security (NDSS) Symposium 2020, San Diego, CA, USA. 2020