面向 APT 家族分析的攻击路径预测方法研究

陈伟翔[1], 任怡彤[1], 肖岩军[2], 侯 锐[3], 田志宏[1]

[1]广州大学 计算机科学与网络工程学院 广州 中国 510006
[2]绿盟科技集团股份有限公司广州分公司 广州 中国 510006
[3]中国科学院信息工程研究所 信息安全国家重点实验室 北京 中国 100093

摘要 近年来, 针对政府机构、工业设施、大型公司网络的攻击事件层出不穷, 网络空间安全已成为事关国家稳定、社会安定和经济繁荣的全局性问题。高级持续威胁(Advanced Persistent Threat, APT)逐渐演化为各种社会工程学攻击与零日漏洞利用的综合体, 已成为最严重的网络空间安全威胁之一, 当前针对 APT 的研究侧重于寻找可靠的攻击特征并提高检测准确率, 由于复杂且庞大的数据很容易将 APT 特征隐藏, 使得获取可靠数据的工作难度大大增加, 如何尽早发现 APT 攻击并对 APT 家族溯源分析是研究者关注的热点问题。基于此, 本文提出一种 APT 攻击路径还原及预测方法。首先, 参考软件基因思想, 设计 APT 恶意软件基因模型和基因相似度检测算法构建恶意行为基因库, 通过恶意行为基因库对样本进行基因检测, 从中提取出可靠的恶意特征解决可靠数据获取问题; 其次, 为解决 APT 攻击路径还原和预测问题, 采用隐马尔可夫模型(HMM)对 APT 恶意行为链进行攻击路径还原及预测, 利用恶意行为基因库生成的特征构建恶意行为链并估计模型参数, 进而还原和预测 APT 攻击路径, 预测准确率可达 90%以上; 最后, 通过 HMM 和基因检测两种方法对恶意软件进行家族识别, 实验结果表明, 基因特征和 HMM 参数特征可在一定程度上指导入侵检测系统对恶意软件进行识别和分类。

关键词 APT 攻击; 恶意行为基因库; HMM; 攻击路径还原及预测; 恶意软件家族分类
中图法分类号 TP393 **DOI 号** 10.19363/J.cnki.cn10-1380/tn.2023.01.01

A Research on Attack-path Prediction Method for APT Organization

CHEN Weixiang[1], REN Yitong[1], XIAO Yanjun[2], HOU Rui[3], TIAN Zhihong[1]

[1] School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou 510006, China
[2] NSFOCUS Technologies Group Co., Ltd, Guangzhou 510006, China
[3] State Key Laboratory of Information Security, Institute of Information Engineering of Chinese Academy Sciences, Beijing 100093, China

Abstract In recent years, attacks against government agencies, industrial facilities and large corporate networks have emerged one after another. Cyberspace Security has become an overall issue related to national stability, social stability and economic prosperity. Advanced persistent threat (APT) has gradually evolved into a complex of various social engineering attacks and zero-day vulnerability exploitation, and has become one of the most serious cyberspace security threats. The current research on APT focuses on finding reliable attack features and improving detection accuracy. Due to the complex and huge data, it is easy to hide APT features, it makes it more difficult to obtain reliable attack features. How to find APT attacks as soon as possible and attribute to the source of APT family is a hot issue for researchers. Based on this, this paper proposes an APT attack path restoration and prediction method. Firstly, referring to the idea of software gene, the APT malware gene model and gene similarity detection algorithm are designed to construct the malicious behavior gene library. The samples are genetically detected through the malicious behavior gene library to extract reliable malicious features and solve the problem of reliable data acquisition. Secondly, in order to solve the problem of APT attack path restoration and prediction, hidden Markov model (HMM) is used to restore and predict the attack path of APT malicious behavior chain. The characteristics generated by malicious behavior gene library are used to construct the malicious behavior chain and estimate the model parameters, and then restore and predict the APT attack path. The prediction accuracy can reach more than 90%. Finally, the family identification of malware is carried out by HMM and gene detection. The experimental results show that the gene characteristics and HMM parameter characteristics can guide the intrusion detection system to identify and classify malware to a certain extent.

Key words APT attack; software gene; HMM; attack path reconstruction and prediction; malware family classification

通讯作者: 田志宏, 博士, 教授, 博士生导师, Email: tianzhihong@gzhu.edu.cn。
本论文得到国家自然科学基金项目(No. U20B2046), 广东省高校创新团队项目 (No. 2020KCXTD007), 广州市高校创新团队项目(No. 202032854)资助。
收稿日期: 2021-06-07; 修改日期: 2022-02-25; 定稿日期: 2022-11-03

1 引言

高级持续性威胁(Advanced Persistent Threat, APT)是一种长周期、多阶段攻击模式, APT 攻击者结合社会工程学, 0day 漏洞等攻击手段对目标实行精准攻击。由于 APT 攻击破坏性极强、隐蔽性极高, 已经逐渐成为网络空间安全的主要威胁, 如何对 APT 家族溯源分析已经成为了安全领域关注的焦点[1]。APT 攻击过程通常持续数周甚至数月, 这种长周期、多阶段的攻击方式使得 APT 攻击具有极高的隐蔽性, 即使其中一环被发现, 受害者也很难知晓攻击者的战略意图。

目前对 APT 攻击进行防御主要依赖于入侵检测系统(Intrusion Detection Systems, IDS)[2], 通过匹配已知的攻击模式(例如基于签名、特征)或观察异常活动来检测 APT 攻击 [3], 但由于 APT 攻击的多阶段攻击模式, 其攻击意图易被海量告警日志淹没, 且 APT 的任一阶段都可能伪装成一种良性行为, 又极大提高了检测难度, 尤其当 APT 攻击持续数月甚至数年的情况下, 被攻击方可能在毫无知觉情况下被窃取大量数据。因此, IDS 不仅需要改良传统手段检测 APT 攻击, 也要致力于关联 APT 生命周期内的数个阶段, 以此提高检测攻击能力。

综上, 参考软件基因思想, 提出了一种基于 APT 恶意行为基因的可靠数据获取方法, 采用隐马尔可夫模型(HMM)还原并预测 APT 攻击路径, 保证了攻击路径的可解释性, 力求在早期发现攻击, 为有效制定防御策略提供支撑。最后, 使用基因检测和 HMM 两种方法对恶意软件进行家族识别。图 1 为整体研究框架。

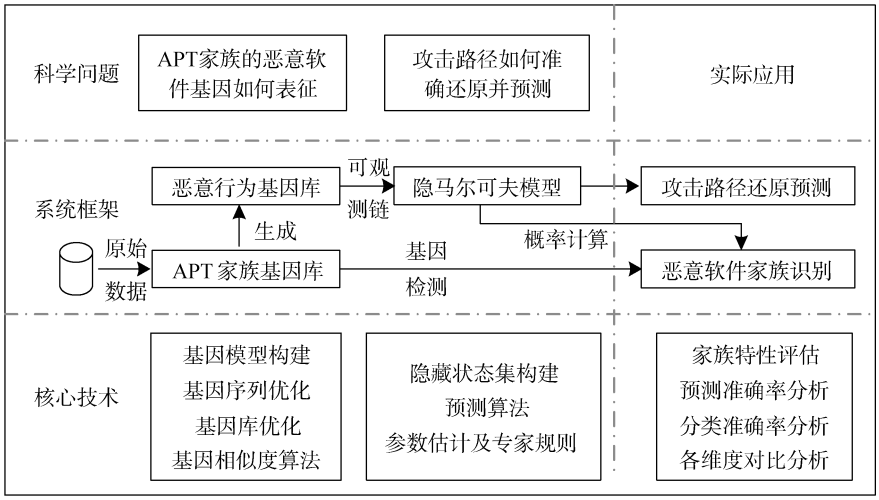


图 1 研究框架

Figure 1 The research framework

2 APT 恶意软件基因模型与基因库

2.1 APT 恶意软件基因模型

APT 恶意软件基因模型定义如下:

(*MD5,env,object_1,object_2,action_name*)

上述模型中, *MD5* 表示恶意软件名称; *env* 为软件运行环境; *action_name* 代表恶意软件的执行动作; *object_1* 与 *object_2* 则分别代表软件执行动作的对象和路径。通过将恶意软件运行时产生的行为按照基因模型抽取并按时间顺序排列, 便可以通过一系列的基因序列描述恶意软件行为, 这些基因序列在本文称为一个软件的基因组, 简称为基因。同一个 APT 家族下恶意软件基因集合, 称为一个 APT 家族的基因库。

2.2 APT 家族基因库

2.2.1 APT 基因库构建规则

APT 家族基因库由该家族中所有恶意样本中所提取基因序列构成, 同时增加下列规则对每个基因库进行精简优化:

1) 保证每条基因的独立性;

2) 保证每条基因不含无效信息;

3) 保证基因库不含两条以上的高相似度基因。

通过去除重复、冗余的基因序列可保证基因独立性, 假设基因 a 与基因 c 属于基因库 G, 则:

$$\forall \vec{a},\vec{c} \in G, cos(\vec{a},\vec{c}) \neq 1 \quad (1)$$

公式(1)要求基因库中每一条基因所含的信息不能与其他的基因相同或者相似, 最大程度上保证每条的独特性。

保证每一条基因不含无效信息可节省存储空间、减少后续设计基因相似度算法的时间复杂度, 同时保证每条基因精准包含恶意信息, 提供更为可靠的数据源。

保证基因库中不含两条以上高相似度基因则是针对基因库的优化策略。通过实践发现, 某些基因具有高度相似性, 但由于路径、名称等属性值(*object_1* 和 *object_2* 包含的信息)不同, 无法被第一条规则去除。针对这种情况, 设计了基因库优化算法。

2.2.2 APT 恶意行为基因序列优化

家族基因库更为关注基因行为而非提供基因的个体, 故 *MD5* 作为基因头会导致基因冗余, 顺沿 *env* 作为基因头并使用替换字进行表示, *object_1* 作为基因段, 也是恶意软件执行动作的对象, 必定为一个可执行文件, 故仅保留可执行文件名称, *object_2* 不做改动, 经统计, 基因尾包含 19 种动作。具体优化规则如表 1 所示。

表 1 基因优化策略

Table 1 Genetic optimization strategy

基因序列元素	优化策略
MD5	删除
Env	使用特殊字符代替
Object_1	保留路径中的可执行文件
Object_2	全部保留
Action_name	全部保留(共 19 种)

2.2.3 基于 Smith-waterman 算法的基因库优化

Smith-waterman 算法是一种局部序列对比算法, 不同于全序列对比, 它可以找出两个序列中具有高相似性的部分。在 Smith-waterman 算法中, 设两条对比序列分别为 $A=(a_1,a_2,\cdots,a_n)$ 和 $B=(b_1,b_2,\cdots,b_m)$, 其中 n 和 m 分别为两个序列的长度。在确定置换矩阵和空位罚分后, 建立一个大小为 $n+1$行$m+1$列的得分矩阵 $\boldsymbol{H}$, 并初始化首行和首列元素, 初始化值为 0。矩阵中其他位置的以从左至右, 从上至下的顺序, 按公式(2)进行打分。

$$H_{ij}=max\begin{cases}H_{i-1,j-1}+s(a_i,b_j)\\ max_{k\geqslant 1}\{H_{i-k,j}-w_k\}\\ max_{l\geqslant 1}\{H_{i,j-l}-w_l\}\\ 0\end{cases} \tag{2}$$

上述规则中$s(a_i,b_j)$表示组成序列中的相似度得分, w_k表示长度为 k 的空位罚分, $H_{i-1,j-1}+s(a_i,b_j)$表示将a_i和b_j比对的相似度得分; $max_{k\geqslant 1}\{H_{i-k,j}-w_k\}$代表了$a_i$位于一段长度为 k 的删除末端的得分。同理, $max_{l\geqslant 1}\{H_{i,j-l}-w_l\}$表示$b_j$位于一段长度为 1 的删除末端的得分; 0 表示两者至此没有相似性。

不难看出, 在得分矩阵 $\boldsymbol{H}$ 中, 可以从得分最高的元素进行回溯, 直至回溯到分数为 0 的元素, 所经历的基因即为最高相似度的基因片段。将最高基因相似度片段记作 *list_of_Smithwaterman*(*a*,*b*), 代表 *a* 与 *b* 相似度最高的公共子串, 其中 *a* 为原始序列, *b* 为对比序列; 两条序列的相似度得分记作 *Smithwaterman*(*a*,*b*)。

在优化过程中, 依照 *object_2* 与 *action_name* 的对应关系, 设计基因合并规则, 将包含相似信息的基因序列进行合并, 并采用算法 1 对基因库进行优化。

算法 1 基因库优化算法

Algorithm 1 Gene Library Optimization

Input: gene pool

1.FOR gene *a* IN gene pool:

2. FOR gene *b* IN gene pool:

3. gene *a* = [env, *object_1*, *object_2*, *action_ name*]

gene *b* = [env, *object_1*, *object_2*, *action_ name*]

4. IF *a*[0]==*b*[0] AND *a*[1]==*b*[1] AND *a*[3]==*b*[3] AND *Smithwaterman*(*a*[2],*b*[2])> Θ:

list *c* = [*a*[0],*a*[1], *list_of_Smithwaterman*(*a*[2],*b*[2]),*a*[3]]

Output: *c*

合并操作时, 随机选取基因库中两条基因进行 *env*、*action_name* 和 *object_1* 匹配, 当前两项不匹配时, 则认为两条基因独立; 当 *object_1* 不匹配时, 则认为是一个行为中的不同过程。若基因中 *action_ name* 与 *object_1* 完全匹配, 再对 *object_2* 使用 *smith-waterman* 算法进行相似度计算。

若计算得出的相似度低于某一阈值 Θ (根据多组实验对比得出的最佳阈值为 80%), 则说明这两条基因相互独立; 若相似度高于 Θ, 则说明这是两条相似的基因, 并提取公共的基因片段, 生成一条新基因。

2.3 APT 恶意行为基因库

2.3.1 恶意行为基因库构建与可观测链提取

在构建 APT 家族基因库后, 可进一步从恶意基因中凝练出恶意行为基因库。具体来讲, 从恶意软件中提取基因序列并与这个恶意行为基因库进行相似度计算, 选取高相似度的基因序列并按时间顺序构建恶意行为链, 为 HMM 提供可靠数据源。并通过基因相似度检测算法来计算基因库与基因库之间、样本与样本之间、样本与基因库之间的相似度。

算法 2 基因相似度检测

Algorithm 2 Gene Similarity Detection

```
Input: gene a, gene b
1.gene a = [env, object_1,object_2,action_name]
2.gene b = [env, object_1,object_2, action_name]
3. IF (a[0]==b[0] AND a[3]==b[3]):
      score_1 = a[1]&&b[1]
      score_2 = Smithwaterman(a[2],b[2]) * 1.8
        Output score_1 + score_2
4   ELSE:
        Output "no relevancy"
```

为提高效率, 在对比两条基因相似度时, 需先检查二者是否拥有相同的 *env* 和 *action_name*。若不同, 则认为两条基因无关联, 相似度为零。

若两条基因 *object_1* 完全相同, 记 1 分, 否则记 0 分。通过 Smith-waterman 算法对两条基因的 *object_2* 进行相似度判定, 用所得的百分比乘以 1.8(该值是为 *object_1* 和 *object_2* 在基因序列中的所占空间比例), 计算结果即为 *object_2* 的得分。两个得分相加即为两条基因的相似度得分。*Thres_hold* 为基因得分阈值的百分比, 具体的阈值为 *Thres_hold* * 2.8(前两者相加)。

2.3.2 恶意行为基因库构建与可观测链提取

对 APT 基因库中的基因进行相似度检测, 取出与所有 APT 基因库的相似度 *Thres_hold* 为 90%以上的基因序列作为公共基因。将公共基因中所有的基因序列从各个 APT 家族基因库中剔除, 再将基因库中剩下的基因合并, 形成恶意行为基因库, 如图 2 所示。

公共基因库包含恶意软件为躲避检测而进行的无害动作基因、各家族共有动作基因、在沙箱中运行时产生恶意动作基因。在具体使用时, 提取恶意软件基因序列, 再分别与恶意行为基因库进行比较。保留相似度 *Thres_hold* 在 90%以上的基因序列, 并按照该恶意软件在沙箱中运行的时间顺序提取出基因尾作为 HMM 的可观测链。

3 基于 HMM 的攻击路径预测

HMM(Hidden Markov Model, HMM)是一种统计模型, 可描述一个含有未知参数的马尔可夫过程, 并用来表示观测结果的概率分布。在将 HMM 用于 APT 攻击路径还原与预测的工作中, 通过观察分析观测值的转移矩阵和发射矩阵, 便可发现可能的异常情况。基于 HMM 进行 APT 路径还原与预测可分为两个阶段, 一是利用已知数据对 HMM 参数进行估计。二是对攻击进行还原与预测, 即利用 HMM 来确定 APT 攻击路径并预测下一步 APT 活动。

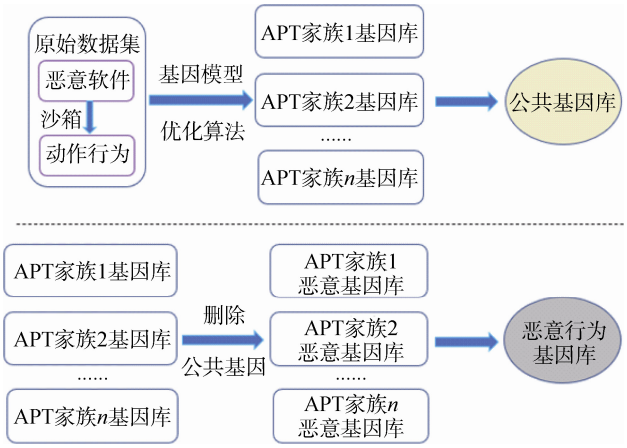


图 2 构建公共基因库与恶意行为基因库

Figure 2 Construction of public gene library and malicious behavior gene library

3.1 APT 攻击路径的状态集

3.1.1 APT 攻击路径的隐藏状态集

恶意软件行为是可观测状态集, 为提取相应的隐藏状态集, 需构建恶意行为与隐藏状态的对应关系, 并考虑使用哪些元素指代这些隐藏状态才能准确描述攻击者的攻击路径。

参考钻石模型, 一个 APT 攻击包含"攻击者"、"受害者"、 "能力"和"基础设施" 4 个核心元素[4]。"攻击者"是攻击事件直接执行者。"受害者"是攻击目标, 在不同场景中, 受害者指代的主体也是不同的。"能力"是攻击者使用的工具或技术, 从探查到最终目的达成, "技术"存在于每一个攻击阶段, 每一个恶意动作对应的技术细节可体现出相应战略意图。"基础设施"是攻击者维持权限控制的通道或者载体。

进一步参考 TTP(战术 Tactics, 技术 Techniques, and 过程 Procedures)概念, "战术"是攻击者从信息收集开始到痕迹清理为止所采用的攻击策略。攻击目标、攻击目的、信息收集方式、寻找目标系统切入点、载荷投递、数据过滤等等都可以划分在战术指标里面。"技术"是攻击者为达成攻击目的, 在具体事件中使用各种技术, 旨在突破防御、维护 C2(命令与控制)通信、横向移动、获得信息、数据等。"过程"是指当攻击者策划一次行动时, 不能仅靠缜密的战术和精湛的技术, 还需精心策划的战术动作才能达到目的, 而目标就是还原蕴含了战术信息的攻击路径。

攻击者借助基础设施并通过技术能力攻击受害者, 这个行为蕴含了攻击者从战术到制定技术直至实施的全过程, 即“寻找入口点”、“权限提升”、“资产发现”、“数据过滤”、“C2 通信”等战术动作, 而这些攻击阶段构成了攻击者入侵系统的一个战术攻击路径。同时, 承载这些战术的技术是由恶意软件的不同功能模块。由此可在可靠数据的支撑下通过技术反推攻击者战术意图。进而可发现攻击者目前处于哪个战术阶段, 并进一步预测攻击者后续攻击行为。结合 ATT&CK 矩阵的相关样例, 将上述过程和攻击路径用图 3 表示。

通过基因获取可观测链并计算 HMM 参数, 再通过模型还原(求解隐藏状态序列)及预测(计算下一时刻的状态概率)攻击路径。隐藏状态序列发现过程如下。

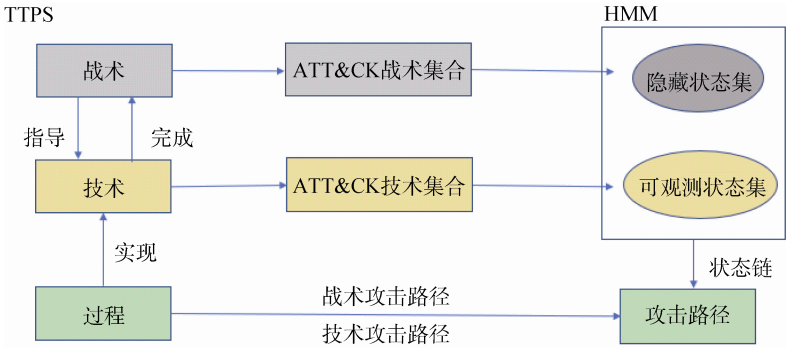


图 3 APT 攻击路径预测的状态集构建

Figure 3 Construction of the states set for APT attack path prediction

第一隐藏状态: 入口点(切入点), 该阶段代表攻击者成功在目标家族内的一个或多个系统上执行了恶意代码, 以此获取目标系统的控制权。

第二隐藏状态: C2 通信, 该阶段是指在家族外围受到破坏之后, 攻击者会保持受感染主机与 C2 服务器之间的连续通信, 以此控制被感染的计算机。

第三隐藏状态: 资产发现, 此阶段旨在识别并找到目标网络中的重要资产。

第四隐藏状态: 数据过滤, 在收获有价值或攻击者感兴趣的数据后, 这些数据会被传输到由攻击者控制的外部服务器。

在整个 APT 攻击阶段中, 实验数据无法体现诸如“情报收集”、“横向移动”等阶段, 因此在实验过程中并未体现。

3.1.2 APT 攻击路径的可见状态集

通过对恶意行为基因库和原始数据分析, 总结出 19 种可观测状态, DRKV(delete-registry-key-value): 删除注册表关键信息的行为; CKO(create-kernel-object): 创建内核对象的行为; CF(create-process): 创建文件的行为; MMP(modify-memory-property): 修改内存权限的行为; MRV(modify-registry-value): 修改注册表的行为; CP(create-process): 创建进程; WTF(write-to-file): 写入文件的行为; SDQ(send-dns-query): 发送 DNS 请求行为; LL(load-library): 加载库; SNP(send-network-packet): 发送网络数据包; CTS(connect-to-socket): 连接 socket; ESI(exec-shellcode-instr): 执行 shellcode 指令; EHI(exec-heap-instr): 执行堆指令; CS(create-service): 创建服务; WTPM(write-to-process-memory): 写入进程内存; ERI(exec-ret-instr): 执行 ret 指令; EEI(exec-esp-instr): 执行 esp 指令; CT(create-thread): 创建线程; LOP(listen-on-port): 监听端口。

上述可观测状态均来自于恶意行为基因的基因尾, 并未包含基因段属性信息, 为保证可观测状态可计算属性, 仅保留基因尾的动作名称来构建可观测序列。

19 种可观测状态构成了 HMM 的可观测状态集, 按照时间顺序相互连接形成可观测状态链, 表征了攻击者在该段时间内完成的一系列动作, 这些动作的集合也构建 HMM 的基础。

3.2 预测算法

通过可观测序列还原 APT 家族模型参数后, 进一步对参数进行评分, 得分最高者作为该家族的 HMM。在使用 HMM 进行预测时, 需要根据实际需求来求解下一阶段最可能的隐藏状态和可观测状态, 获得解码的隐藏状态序列后, 并输入 HMM 参数和可观测序列求解并计算每一个隐藏状态和可观测状态的具体概率值, 以保证结算结果可靠性。具体流程见算法三。

算法 3 计算下一时刻的隐藏状态和可见状态概率

Algorithm 3 calculation probability of the hidden and observed states at the next moment

Input: O_T and HMM parameters A、B、Π

1.FOR each possible next state j = 1,2,⋯ ,N DO:

2. FOR each possible intermediate state i = 1,2, ⋯,N, AND their Respective observation DO:

3. Calculate the probability $\boldsymbol{\alpha_t(i)}$

4. $\boldsymbol{\alpha_t(i) = \sum_{r=1}^{N} \alpha_{t-1}(r) a(r,i) b_i(O_t)}$ where r denotes index of all possible prior states

5. Compute the probability $\boldsymbol{P(q_{t+1} = s_j)}$ by multiplying all $\boldsymbol{\alpha}$ Parameters, at time with their respective transition probabilities

6. $\boldsymbol{P(q_{t+1} = s_j) = \sum_{i=1}^{N} \alpha_t(i) a_{i,j}}$

7. $\boldsymbol{P(o_{t+1} = i_j) = \sum_{q_{t+1}} p(i_j|q_{t+1}) P((q_{t+1}|o_t))}$

Output: the probabilities of next state and observation

4 实验结果与分析

实验数据采用绿盟科技集团股份有限公司广州分公司提供的恶意软件样本, 从中选取了样本无缺失、特征无失真的 6 个 APT 家族(共计 2270 个恶意样本), 进行路径还原及预测实验。这些恶意软件与进行基因库构建的恶意软件仅在检测、对比环节有重合。运行这些恶意软件的沙箱处理器架构为 X86-32, 系统环境包含 WinXP SP3 和 Win7 SP1。总计提取了 255308 条由行为信息构建的基因序列, 并将这些基因序列用于攻击路径的还原与预测。

4.1 APT 攻击路径还原及预测

通过基因模块对恶意软件进行可靠数据提取。并将恶意基因按运行时间顺序抽取基因尾(即恶意软件执行的动作名称)构建可观测链, 部分可观测链如表 2 所示。

表 2 样本中恶意行为分布

Table 2 The distribution of malicious behavior in the sample

Time_1	Time_2	Time_3	Time_4	Time_5
创建文件	加载内存	写入文件	修改内存	创建进程
加载内存	创建服务	发送数据	运行脚本	创建文件
删除文件	修改内存	发送数据	创建进程	删除文件
建立连接	修改内存	运行脚本	写入文件	发送数据

将恶意行为从样本中分离出来并加入时序特征, 可有效地描述该样本在某段特定时间内的动作, 并构成可观测序列, 这些可观测序列所对应的隐藏状态序列也会随时间在 APT 的生命周期内不断前进。

在使用可观测链还原各 APT 家族的 HMM 参数后, 通过 Viterbi 算法对隐藏状态进行还原, 同样以 Anunak 家族中一号样本为例, 其通过 Viterbi 算法进行解码后获取战术攻击路径, 前 4 个阶段的战术攻击路径如表 3 所示:

表 3 Anunak 家族一号样本路径还原结果

Table 3 The results of path restoration for Anunak No.1 sample

	Stage_1	Stage_2	Stage_3	Stage_4
隐藏状态	资产发现	数据过滤	寻找入口	C2 通信
动作数量	8	2	8	29

可以看出一个战术意图会对应多个动作, 这也是在发射矩阵中某个隐藏状态下出现某些特定恶意行为的概率会非常高的原因。

在还原了 APT 攻击路径后, 采用预测算法计算下一时刻攻击路径中每个状态的概率, 并结合 Anunak 家族的模型参数, 同样以 Anunak 一号样本为例, 得出攻击路径在时间长度为 101 时的 4 个隐藏状态的概率和 16 个可观测状态的概率(仅展示 16 个的原因是该家族后三种可观测状态概率极小)。

分析表 4 和表 5 不难看出, 一号样本在时间长度为 101 时处于第三种隐藏状态(即资产发现)的概率最高, 对应的可观测状态为修改内存权限。结合其他概率值, 可以得出该恶意软件在 T=101 时最可能通过修改内存权限获取隐私数据。通过数据比对, 预测结果与 T = 101 时检测到的可观测状态相吻合。在表 4 和表 5 的样本中并没有出现概率相近的情况, 但通过大量实验分析发现, 某些样本存在可观测状态概率峰值非常相近的情况, 这种情况将无法直接判定在下一时刻的状态, 因此在进行最终结果判定时, 将多个观测值作为判定结果, 即选取概率最高的两个可观测状态作为下一时刻的路径状态, 将测试集中的样本放入各自的 APT 家族的 HMM 中计算下一时刻攻击路径的隐藏状态和可观测状态的概率。预测结果的准确率如表 6 所示。

表 4 Anunak 家族一号样本各隐藏状态概率

Table 4 The probability of hidden stases for Anunak No.1 sample

多隐藏状态概率	概率值
The probabilities of state 1	0.0022665
The probabilities of state 2	0.0037617
The probabilities of state 3	0.9875497
The probabilities of state 4	0.0064221

表 5 Anunak 家族一号样本各可见状态概率

Table 5 The probability of observed stases for Anunak No.1 sample

各可见状态概率	概率值
probabilities of observation 1	0.009664
probabilities of observation 2	0.060137
probabilities of observation 3	0.016748
probabilities of observation 4	0.586756
probabilities of observation 5	0.131888
probabilities of observation 6	0.017333
probabilities of observation 7	0.003361
probabilities of observation 8	0.018231
probabilities of observation 9	0.000026
probabilities of observation 10	0.054847
probabilities of observation 11	0.071623
probabilities of observation 12	0.024218
probabilities of observation 13	0.000002
probabilities of observation 14	0.003979
probabilities of observation 15	0.001072
probabilities of observation 16	0.000097

表 6 不同观察值下各家族攻击路径预测准确率

Table 6 The prediction accuracy of attack paths of each family under different observation values

	2 个(%)	3 个(%)	4 个(%)	5 个(%)
Anunak	62.77	77.66	86.17	92.55
APT 29	62.16	74.32	78.85	86.54
Lazarus Group	69.12	77.94	85.29	89.71
Molearats	66.13	75.81	85.48	96.77
Turla Group	69.23	69.23	76.92	84.61
Dropping Elephant	60.00	66.67	86.67	93.33

在采用两个观测值进行预测时各个 APT 家族均能达到 60%以上的准确率，部分家族甚至可以接近 70%。

训练集的样本数量会在一定程度上影响最终的准确率，样本数量过少可能无法包含 APT 家族中的全部有效特征从而导致效果下降。同时，HMM 的预测结果会受训练时的可观测序列长度和 APT 本身的一些相似性的影响。在训练时可观测序列长度较低时，HMM 无法有效的学习出该 APT 家族的各个参数，导致实验结果失真。但过长的观测序列会导致无法对 HMM 进行评定，即无法通过设置阈值评价模型的好坏。另一方面，经过多次实验验证后发现，在预测长时间序列的下一状态和预测短时间序列的下一状态时，前者和后者在 4 个和 5 个观测值下的概率差距不大，但在 2 个和 3 个观测值下的概率有较大的差别，说明可以在攻击前期快速且准确的发现并预测攻击。具体原因是在 APT 攻击早期，恶意软件仍未取得更高的权限，其执行的动作单一，且目的性较强，易被系统察觉，在预测时通过累加所有可能状态的后验概率与转移矩阵、发射矩阵对应位置的乘积，在较短的时间序列中每个可观测状态在下一时刻的概率之间的差值会比长时间序列的差值更为明显，由此实现在 2 个或是 3 个观测值下的准确率提升。

各家族的攻击路径预测准确率如图 4、图 5、图 6 所示。

接下来对比不同方法下的预测结果。文献[5]中是采用 K-mean 聚类的方法定义网络中的状态，之后通过构建 HMM 模型来计算网络中事件的得分(概率取对数)来判断是否被攻击。这里将使用相同的数据集(Anunak 家族下的样本)进行试验对比，实验结果如图 7 所示。

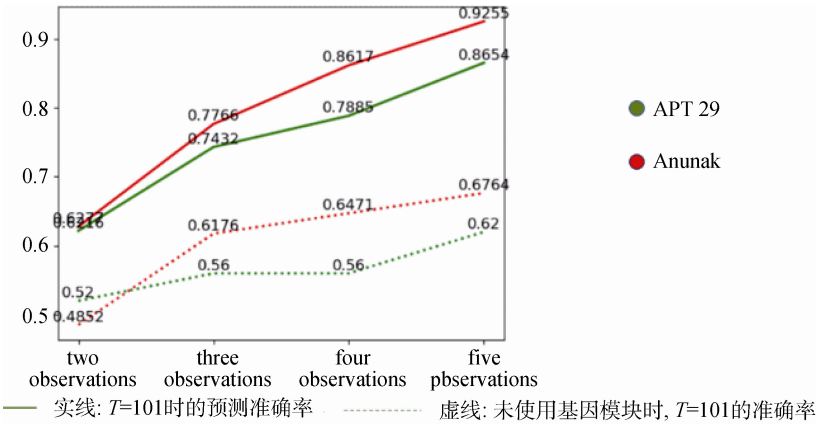


图 4 Anunak 家族和 APT 29 家族的准确率对比

Figure 4 Accuracy comparison between Anunak and APT 29

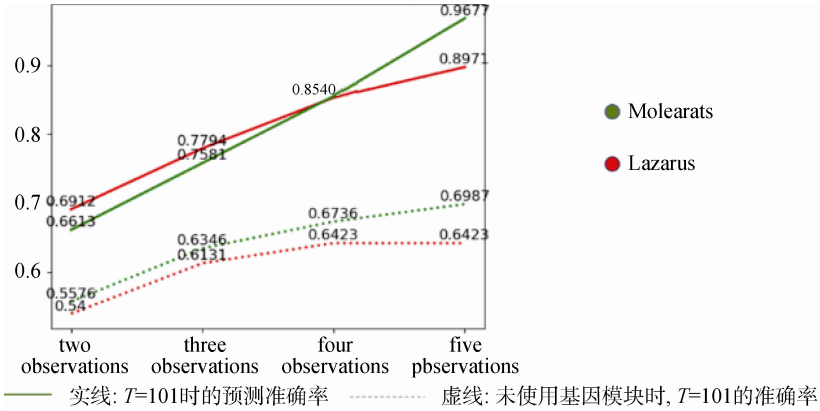


图 5　Lazarus 家族和 Molearats 家族的准确率对比

Figure 5　Accuracy comparison between Lazarus and Molearats

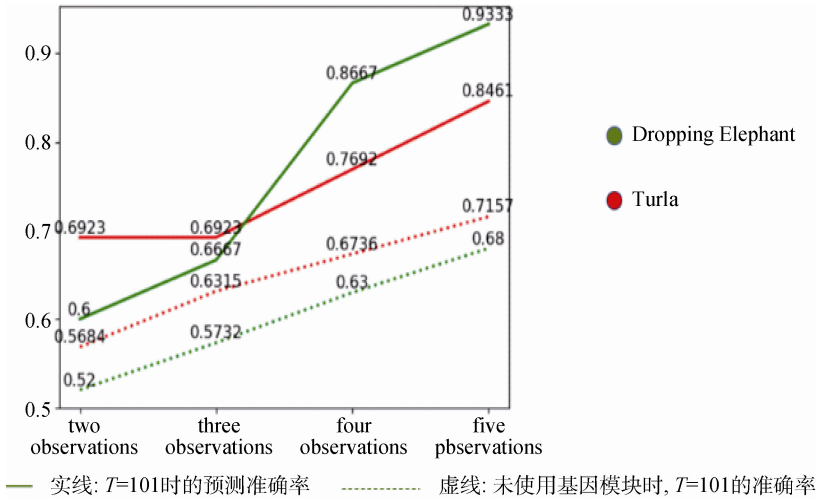


图 6　Turla 家族和 Dropping Elephant 家族的准确率对比

Figure 6　Accuracy comparison between Turla and Dropping Elephant

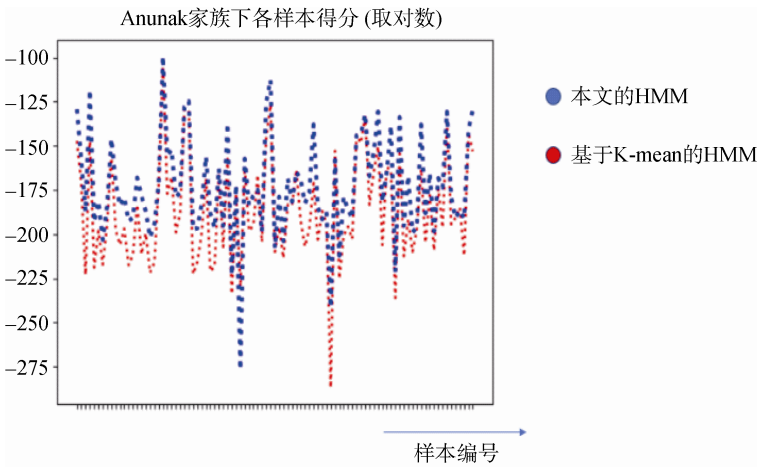


图 7　Anunak 家族样本在两个 HMM 下的得分

Figure 7　The scores comparison of Anunak sample using two HMMs

基于 K-mean 的 HMM 是用于预测系统中是否存在攻击, 其通过检测可观测序列的得分是否处于临界值来判断接下来是否会受到攻击。从图中可以看到, 同一个恶意样本在不同的模型下的得分差距不是特别大, 且部分样本点会高于红点的得分。

图 8 表明本文提出的方法预测准确率更高, 也

同时说明了能在一定程度上检测系统中是否存在攻击。同时，本文中的 HMM 的隐状态是 APT 阶段，能够还原出攻击者在战术层面上的攻击路径。

对比了不同方法构建 HMM 状态集之后，对同样使用 APT 阶段作为隐藏状态集的工作进行了横向比较，文献[6]将 APT 各攻击阶段作为隐藏状态集，使用安全设备的告警作为可观测状态集，并设计相应的规则和时间窗口来构建同一个 APT 活动下的可观测链。使用其公开的数据集进行实验，结果如图 9 所示。

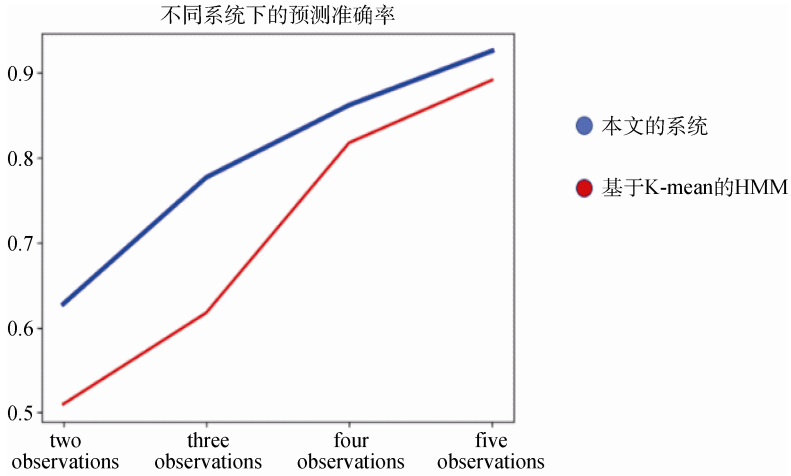


图 8　Anunak 家族样本在两个 HMM 下的预测准确率

Figure 8　The prediction accuracy of Anunak sample using two HMMs

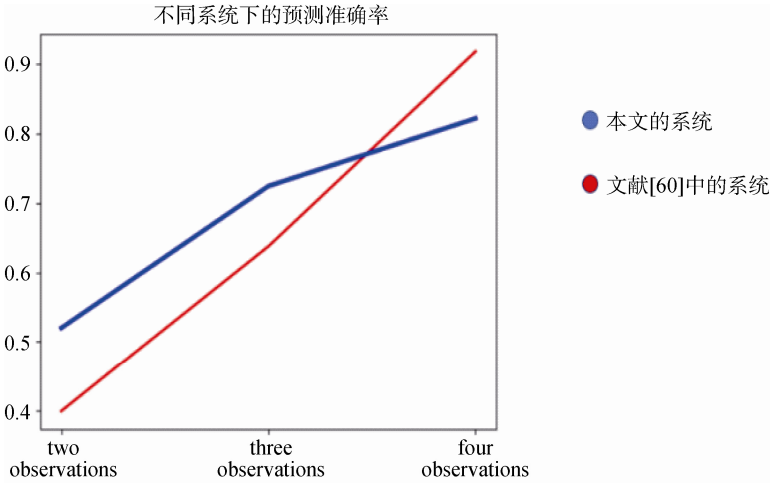


图 9　不同系统下的预测准确率

Figure 9　The prediction accuracy of different systems

可以看出，在 2 个和 3 个观测值下，本文所提方法有更高的准确率，但在 4 个观测值下的准确率比文献中略有下降，其原因是通过恶意行为基因库进行可观测链提取，而这些基因序列来自于软件行为。在实验中使用的是告警信息，因此只能使用基因中包含 C2 通信的部分进行关联和提取，基因库的使用率不足 30%，导致了准确率下降。但通过前两个观测值的比对，可以看出本文系统对攻击路径的还原和预测具有一定的可靠性和科学性。

4.2　恶意软件家族识别

前述实验中探讨了 HMM 的预测工作，但这是基于已知家族的前提下，在实际场景中应用场景可能会受到很大的限制。因此需要更深层次探索是否可以使用路径特征进行家族识别。

在验证过程中，首先根据隐藏状态集构建状态转移矩阵，并使用所有的可观测行为和隐藏状态序列构建 4*19 的发射矩阵，将各个家族的发射矩阵和概率转移矩阵填充。在发现一个新样本后，首先进行基因筛选，通过基因比对来判定该样本是否为恶意样本，若是则提取出疑似恶意行为的可观测链。将该状态链放入各个家族的 HMM 中进行概率计算，将最高概率值作为家族识别依据。

在进行基因识别时, 从新的恶意软件中提取基因序列, 并使用基因相似度算法将这个恶意软件中的基因组与各个 APT 家族基因库进行基因检测。在使用基因检测进行家族识别时, 提取出恶意软件中与恶意基因库相似度在 90%以上的恶意基因与各个 APT 家族的恶意基因库进行相似度比对。将样本$A=(a_1,a_2,\dots,a_n)$中的一条基因a_1与某个 APT 家族基因库中的所有基因分别计算相似度, 取其中的最大值作为a_1与某个 APT 家族的相似度, 计算完样本 A 中所有的基因之后, 设置阈值为 0.8(多组对比试验后得出的最佳阈值), 求阈值之上(2.8*0.8)的得分之和, 作为样本 A 与该 APT 家族基因库的相似度。在计算完所有 APT 家族的相似度之后, 输出这个软件与每个 APT 家族的相似度, 相似度得分最高的家族将作为这个恶意软件所属的家族。

在使用 HMM 进行家族识别时, 由于训练集的内容会导致 HMM 的参数发生较大的变化, 所以进行了多次实验, 最终选出六个家族中得分最高的 HMM 进行综合判定, 而在讨论 HMM 对恶意样本的识别能力时, 则分别选取 3 个模型进行准确率展示。家族识别的结果如表 7 和表 8 所示。

表 7　三种模型的识别准确率

Table 7　The recognition accuracy of the three models

	基因检测	HMM 检测(平均)	神经网络
准确率	82%	90.4%	73%

表 8　HMM 下各家族的识别准确率

Table 8　The recognition accuracy of different familys using HMM

APT 家族	最低准确率(%)	最高准确率(%)
Anuank	92	100
APT 29	73.9	89
Lzarus Group	80.2	92
Molearats	82	92.3
Turla Group	78	96
Dropping Elephant	83	100

由表 7 可知, 基于 APT 家族基因库的相似度检测相较于一般的机器学习方法优势明显, 并且基因的可读性和可解释性都优于单纯的代码特征提取。从这些实验中可以证明: APT 家族基因库可以在一定程度上包含这个家族的特征, 且这部分特征可以用于恶意软件的分类工作。使用恶意行为链构建的 HMM 对同类型的观测序列具有很高的家族识别性, 比使用基因检测进行前二判定的识别准确率高 8%。主要原因是, 在特定家族下, 攻击者实现某个战术意图时使用某个恶意行为的概率非常高, 在实现一连串战术意图时会对应特定的恶意行为链。这些行为在 HMM 中就会体现在 HMM 的发射矩阵中, 当攻击者使用同样的方法发起攻击时, 对应的行为链在该家族下的 HMM 中便会有较高的得分, 由此证明了 HMM 能有效对恶意软件进行家族识别。

表 8 表明了部分家族可以通过模型 100%识别自身的恶意软件。通过分析最差和最好的 HMM 后发现: 准确率和训练样本的数量和质量有直接关系。在数据质量方面, 若一条观测链在多个模型下均展现出极少或是单一的 APT 阶段, 就会严重影响 HMM 的构建。这种数据训练得到的状态转移矩阵会出现大量零值, 在发射矩阵中会得到大量概率为 1 的值, 从而严重影响最终结果。在减少了这类影响后, 训练所得矩阵的参数分布具有明显的家族特征。另一方面, 虽然基因检测的识别准确率不如 HMM 检测法, 但是能精准定位到具体基因, 能在一定程度上应对恶意软件变种。相反, HMM 检测法虽然准确率高, 但是十分依赖数据的质量和数量, 且对恶意软件的追踪能力远不如基因检测。

综上, 在控制样本数量、质量及训练集时间长度的情况下, 可以使用 HMM 对 APT 家族攻击路径进行还原, 并且可以预测其下一步的战术意图和恶意动作。同时, 包含了家族特征的 HMM 有优于基因检测的 APT 家族识别能力。

5　相关研究工作

APT 攻击模式通常是根据目标设计, 具有强针对性。同时, 由于 APT 攻击是多阶段的, 受害者很难发现它们正在被攻击[7-8]。近年来, 研究人员在构建 APT 入侵检测系统时主要采用两种技术手段获取恶意特征: 大数据分析和软件分析。基于大数据分析的入侵检测系统一般会从网络流量和系统日志中获取特征; 基于软件分析主要包括动态分析[9]和静态分析[10]等技术。

网络流量检测是将网络流量作为系统输入, 再利用机器学习和统计分析等技术手段对输入进行处理, 以此发现 APT 攻击痕迹或特征的检测方法[11]。在进行网络流量检查时, 需要特定工具来捕获网络上的流量信息。有两种主流数据采集方法, 一是收集两台主机之间的字节数或流量包数等信息; 二是收集流量包中某些会变化的特征信息, 如 IP 地址或协议类型[12]。在获得这些数据之后, 可以使用无监督学习、监督学习和半监督学习等方法作为异常检测和

APT 溯源分析的基础。

基于软件分析中的恶意代码分析可以有效地提取恶意软件特征, 这些特征通常包含某些恶意软件家族的关键代码段, 也可能成为某个 APT 家族的家族特征, 这些特征可以有效的检测 APT 攻击以及对恶意软件进行跟踪[13]。恶意代码分析包括静态分析和动态分析[14]。静态分析是一种在不运行软件的情况下分析可执行代码的方法[15]。通过静态分析可以得到软件文件结构、数据流、指令流、API 调用等信息。利用这类信息构造恶意样本静态特征, 再利用这些特征进行异常检测和软件跟踪[16]。机器学习技术可以在静态分析的基础上检测异常代码的静态特性, 以此提高自身检测效率。静态分析和网络流量分析相比有更高的覆盖率且花费的时间更少[17]。然而, 当恶意软件打包、变形或使用代码混淆和加密技术时, 静态分析很难分离出有效特征信息 [18]。

动态分析一般使用 Anubis、Norman、Joebox 等分析工具在沙箱或虚拟环境中动态运行程序, 捕捉程序运行过程中产生的函数访问、系统调用、API 操作、注册表修改和访问等一系列动态特征[19]。动态分析的优势在于它能够克服各种反分析技术[20], 但无法保证代码覆盖率和执行效率, 也无法在软件崩溃时提取动态特征[21]。在动态分析的基础上有基于沙箱的攻击路径还原技术, 其监控沙箱中运行的恶意软件, 记录其交互接口和创建线程等信息, 实时构建出一个恶意软件的攻击链路。但是这种方法局限于该恶意软件运行期间的动作, 延展性较差[22]。

入侵检测系统不仅需要通过传统手段检测 APT 攻击, 也要致力于关联 APT 生命周期内的数个阶段, 以此提高检测攻击的能力。为检测并还原多阶段攻击, 国内外学者提出了基于 HMM(Hidden Markov Model, HMM)等技术手段。HMM 是一种统计模型, 该模型已经被用于多个领域, 例如, 语音识别[23]、文本理解[24]和图像识别[25]。在面对诸如 APT 攻击这种多阶段威胁模式时, HMM 便可以通过可靠数据集求解出状态转移矩阵和发射矩阵。通过分析这些概率矩阵, 防御人员便能发现网络中的异常。当 IDS 未能检测到攻击的某些阶段时, HMM 就能用来还原这些细节。因此, HMM 为解决还原攻击活动的完整性提供了一种可行的思路。

6 结论

本文提出了一种基于 HMM 和恶意行为基因库的攻击路径研究方法, 旨在还原并预测 APT 攻击者的攻击路径。首先通过分析恶意软件的攻击流程设计了一个基因模型, 并使用基因序列优化算法和基因相似度算法构建了 APT 家族基因库, 通过这些家族基因库生成了一个恶意行为基因库为后续实验提供可靠数据集。在对 HMM 进行构建时, 设计了 APT 攻击路径下的状态集并提出了相应的专家规则。之后进行了大量的研究和对比实验, 最终找出得分最高的 HMM。进行还原和预测时, 通过对多种情况进行分析以保证结果的准确性, 并设计了对比实验以保证可靠性。最后, 结合基因检测和 HMM 进行家族识别, 完整的还原了一个样本的家族归属、攻击路径以及对其下一步进行预测, 并由此达到实时防御的目的。

本文虽然在 APT 攻击路径的研究方面取得了一定的研究成果, 但由于时间以及实验数据的局限性, 还有一部分问题没有解决。例如, 由于实验数据限制, 隐藏状态只有 4 个。后续可通过收集更多维度的样本来补充这部分研究。此外, 恶意动作的粒度并不相同, 后续可以将某些粗粒度的恶意行为划分, 再进行对比试验并分析相关结果。

参考文献

[1] Lv K, Chen Y, Hu C Z. Dynamic Defense Strategy Against Advanced Persistent Threat under Heterogeneous Networks[J]. *Information Fusion*, 2019, 49: 216-226.

[2] Bedi P, Gupta N, Jindal V. I-SiamIDS: An Improved Siam-IDS for Handling Class Imbalance in Network-Based Intrusion Detection Systems[J].*Applied Intelligence*, 2021, 51(2): 1133-1151.

[3] Duque S, Omar M N B. Using Data Mining Algorithms for Developing a Model for Intrusion Detection System (IDS)[J]. *Procedia Computer Science*, 2015, 61: 46-51.

[4] Caltagirone, S，Pendergast, A，Betz, C. The Diamond Model of Intrusion Analysis [R]. Diamond Model of Intrusion Analysis, 2013.

[5] Shin S, Lee S, Kim H, et al. Advanced Probabilistic Approach for Network Intrusion Forecasting and Detection[J]. *Expert Systems With Applications*, 2013, 40(1): 315-322.

[6] Ghafir I, Kyriakopoulos K G, Lambotharan S, et al. Hidden Markov Models and Alert Correlations for the Prediction of Advanced Persistent Threats[J]. *IEEE Access*, 7: 99508-99520.

[7] Patnaik S, Sidhardh S, Semperlotti F. Fractional-Order Models for the Static and Dynamic Analysis of Nonlocal Plates[J]. *Communications in Nonlinear Science and Numerical Simulation*, 2021, 95: 105601.

[8] Chai Y H, Du L, Qiu J, et al. Dynamic Prototype Network Based on Sample Adaptation for Few-Shot Malware Detection[J]. *IEEE Transactions on Knowledge and Data Engineering*, 2820, PP(99): 1.

[9] Caballero J, Yin H, Liang Z K, et al. Polyglot: Automatic Extrac-

tion of Protocol Message Format Using Dynamic Binary Analysis[C]. *The 14th ACM conference on Computer and communications security*, 2007: 317-329.

[10] Moser A, Kruegel C, Kirda E, et al. Limits of static analysis for malware detection[C]. *Twenty-Third Annual Computer Security Applications Conference*, 2008: 421-430.

[11] Nari S, Ghorbani A A, Processing C A, et al. Automated malware classification based on network behavior[C]. *2013 International Conference on Computing, Networking and Communications*, 2013: 642-647.

[12] J. Vries, J. van den Berg, M. Warnier, H. Hoogstraaten. An analysis framework to aid indesigning advanced persistent threat detection systems[D]. *Delft university of technology*, 2012.

[13] Kruegel C, Toth T. Using Decision Trees to Improve Signature-Based Intrusion Detection[M]. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003: 173-191.

[14] Mihai Herda. Combining Static and Dynamic Program Analysis Techniques for Checking Relational Properties[D]. *Karlsruhe Institute of Technology*, Germany, 2020.

[15] Ahmed B H, Lee S P, Su M T. The Effects of Static Analysis for Dynamic Software Updating: An Exploratory Study[J]. *IEEE Access*, 8: 35161-35171.

[16] Do L N Q, Kruger S, Hill P, et al. Debugging Static Analysis[J]. *IEEE Transactions on Software Engineering*, 2020, 46(7): 697-709.

[17] Manjunath K G, Jaisankar N, Shreedevi K G. An Intelligent Intrusion Detection for Detecting Unauthorized Malware over the Network[J]. *International Journal of Engineering and Technology*, 2013, 5(2): 1373-1380.

[18] Imtiaz N, Williams L. A Synopsis of Static Analysis Alerts on Open Source Software[J]. *HotSoS*, 2019, 12:1-12:3

[19] Eduardo Rosales, Daniele Bonetta, Isabella Defilippis, Sergio Oporto, Walter Binder. Automated Large-scale Multi-language Dynamic Program Analysis in the Wild[J]. *Software Engineering,* 2021, 111.

[20] Lu T, Liu C, Duan H, et al. Mining Component-Based Software Behavioral Models Using Dynamic Analysis[J]. *IEEE Access*, 8: 68883-68894.

[21] Kim Y H, Park W H. A Study on Cyber Threat Prediction Based on Intrusion Detection Event for APT Attack Detection[J]. *Multimedia Tools and Applications*, 2014, 71(2): 685-698.

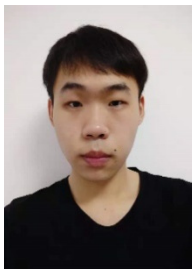
[22] Matias T, Correia F F, Fritzsch J, et al. Determining Microservice Boundaries: A Case Study Using Static and Dynamic Software Analysis[M]. Software Architecture. Cham: Springer International Publishing, 2020: 315-332.

[23] Wang Z Y, Xiao X. Duration Distribution Based HMM Speech Recognition Models[J]. *Acta Electronica Sinica*, 2004, 32(1): 46-49.
(王作英, 肖熙. 基于段长分布的 HMM 语音识别模型[J]. *电子学报*, 2004, 32(1): 46-49.)

[24] Xu H H. Recognition Study of Text Dependent Speakers Based on Optimized HMM Algorithm[J]. *Microcomputer & Its Applications*, 2010, 29(2): 69-70, 74.
(徐惠红. 优化的 HMM 算法在文本相关的说话人识别中的研究[J]. *微型机与应用*, 2010, 29(2): 69-70, 74.)

[25] Jin H, Gao W. Analysis and Recognition of Facial Expression Image Sequences Based on Hmm[J]. *Acta Automatica Sinica*, 2002, 28(4): 646-650.

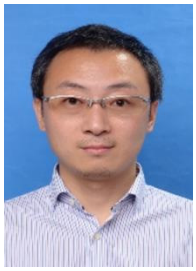
[26] (金辉, 高文. 基于 HMM 的面部表情图像序列的分析与识别[J]. *自动化学报*, 2002, 28(4): 646-650.)



陈伟翔 于 2021 年在广州大学计算机技术专业获得硕士学位。现任美团算法工程师。研究领域为流量安全、反爬虫。Email: chen825457110@qq.com

侯锐 于2007年在中科院计算所获得博士学位。现为中国科学院信息工程研究所研究员。主要研究领域为 CPU 架构与安全，数据中心服务器设计。Email: hourui@iie.ac.cn

任怡彤 于 2020 年在中国民航大学计算机技术专业获得工程硕士学位。现在广州大学网络空间安全专业攻读博士学位。研究兴趣包括: APT 检测与溯源，知识图谱，图神经网络。Email：renyitong@e.gzhu.edu.cn



田志宏 于 2006 年在哈尔滨工业大学计算机科学与技术专业获得博士学位。现任广州大学网络空间安全学院院长，教授，博士生导师，主要研究领域为网络攻防对抗。Email: tianzhihong@gzhu.edu.cn

肖岩军 于 2004 年郑州大学信息管理与信息系统专业获得管理学学士学位。现任绿盟科技平行实验室主任研究员。研究领域为态势感知、知识图谱、APT 追踪、人工智能决策指挥、网络靶场。Email：xiaoyanjun@nsfocus.com