测量和建模恶意软件在线分析平台的标签动态

黎锦灏　518021910771

在本课程的学术论文阅读作业中，我选取阅读的论文名为 **Measuring and Modeling the Label Dynamics of Online Anti-Malware Engines**，是 2020 年在四大安全顶会之一 **USENIX Security Symposium** 上发表的一篇关于恶意软件在线分析平台 $VirusTotal$ 的论文。论文链接：

https://www.usenix.org/conference/usenixsecurity20/presentation/zhu

一、内容摘要

$VirusTotal$ 能综合各大反恶意软件引擎的结果，为用户提供上传文件的恶意软件标签，并被研究人员广泛应用于恶意软件注释和系统评估。在这篇论文中，作者采用 **数据驱动（data-driven）** 的策略来对研究者们常用的标签方法进行分类、解释和验证。作者首先调研了 115 篇和 $VirusTotal$ 相关的学术论文，并且在一年内从 65 个 $VirusTotal$ 采用的引擎中收集了超过 14000 个文件的标签动态。在这一分析过程中，作者验证了基于 **阈值（threshold）** 的标签整合分析结果的稳定性，并指出了阈值对结论的影响。通过对数据集的测试，本文发现一些广泛使用且被认为"可信"的反恶意软件引擎的表现并不理想，并且许多引擎的测试结果非常近似，可以认为这些引擎组是紧密相关的，在参考它们的分析结论时应该作为整体综合考虑。最后，作者为将来使用 $VirusTotal$ 进行数据标注、恶意软件注释提供了建设性建议。

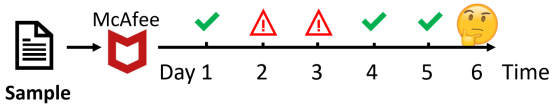
二、研究背景

$VirusTotal$ 是目前最大的在线恶意软件扫描平台，应用了 70 多种反恶意软件引擎，并且能提供详细的分析报告和丰富的数据源，在安全界被研究者们广泛应用于恶意软件注释和系统评估。

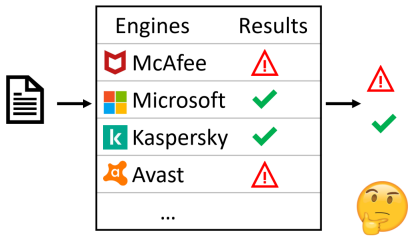
用户在平台提交文件之后，能获得 70 多个引擎提供的恶意软件标签，即对该提交文件是否恶意的判断结果。考虑到不同引擎有时候会得出互相矛盾的分析结果，同一系统也可能对同一文件给出截然不同的结论，$VirusTotal$ 需要设计出一个模型来处理标签数据，整合并给出最终的判定结果。作者在论文答辩中总结了目前 $VirusTotal$ 存在的几大问题：

- $VirusTotal$ 标签动态何时可信赖？
- 如何整合从不同引擎得到的标签？
- 不同的引擎之间参考权值是相等的吗？

Q1: When VirusTotal labels are trustworthy?



Q2: How to aggregate labels from different engines?



Q3: Are different engines equally trustworthy?

过去研究者提出了一些解决检测结果变化的方案，但大多依赖经验和直觉，缺乏可靠证据和合适的评判标准。本文基于数据驱动的方式，完成了以下研究：

- 测量同一个引擎对给定文件的标签动态反转（$flip$）频率
- 建立不同引擎对同一文件的检测结果之间的关联，探索引擎平权假设的合理性
- 鉴定哪些引擎给出的标签动态易受其他系统的影响
- 使用标注好的标准数据（$ground-truth$）检验引擎的检测结果准确率

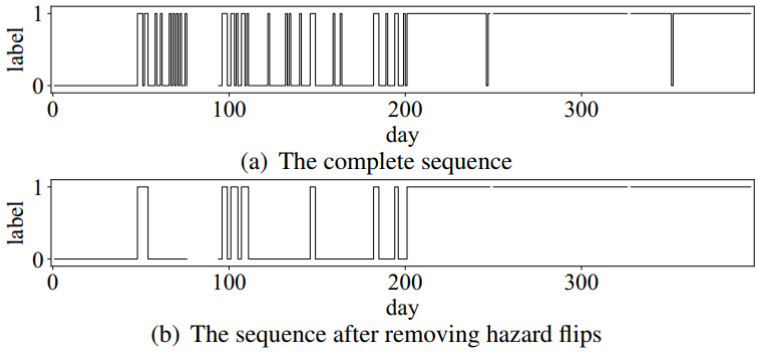
三、标签动态测量分析

本文将标签动态用$0/1$逻辑信号序列表示，特别的，对于文件 f 和引擎 i，有标签序列：

$$S_{i,f} = [l_1, l_2, \ldots, l_t, \ldots, l_N]$$

其中，N为数据集中统计的总天数，且 $l_t = 0(benign)\ or\ 1(malicious)$

对于给定测试文件，本文将标签序列中连续两天的标签发生变化 $0 \to 1$ 或 $1 \to 0$ 定义为 $flip$，$0 \to 1 \to 0$ 或 $1 \to 0 \to 1$ 定义为 $hazardflip$。根据统计，在主数据集中共有 2571809 个 $flip$，其中有 1760484 个 $hazardflip$ 和 811325 个 $non-hazardflip$。以一个特定的文件在 $AegisLab$ 引擎得到的标签序列为例，$hazardflip$ 普遍存在并且持续若干天，对比移除了 $hazardflip$ 后的标签序列与原始序列，容易发现超过一半的 $flips$ 都属于 $hazardflips$。



(a) The complete sequence

(b) The sequence after removing hazard flips

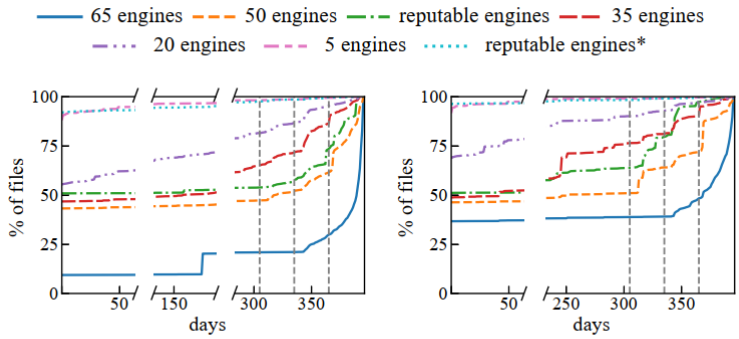
图1.标签序列实例

为分析 $flips$ 的产生原因，作者在 $VirusTotal$ 的API多次提交测试，查询结果表明重复的提交只能减少非常有限的 $flips$ ，并以此推断 flips 源于 $VirusTotal$ 的内部问题。查询引擎的更新日志，容易发现 flips 常常出现在引擎模型的更新前后和版本更迭过程中，这表明模型的更新造成了决策的变化。

考虑到 flips 对测试结果的影响，我们需要分析标签的稳定性并探索何时可以得到稳定的标签动态序列：

- 若统计所有引擎的数据，仅有 9.37% 文件的标签动态序列在 50 天之后能趋于稳定，但在 176 天以后，这一比率跳变到 20.22% ，而 350 天以后大部分序列都不再变化。
- 若只统计高声望（$high-reputation$）引擎数据，序列稳定情况非常接近于 35 个引擎的情况（少数高声望引擎贡献了很大比重的$flips$）。进一步的，只考虑 2 个最具盛名的引擎，我们发现，标签序列很快趋于稳定。
- 若去除$hazard-flip$，并重复上述统计，对比原始数据集可以明显看到趋于稳定的序列比重大幅提高了，但稳定的速率并没有因为去除掉$hazard-flip$而加快，稳定的突变点没有显著变化。

由上述讨论可以看出，提交后等待并观察标签序列并不能保证从系统得到更稳定的检测结果，除非我们只考虑一小部分引擎给出的结果。



(a) The original dataset (b) The dataset without hazards

图2.标签序列稳定趋势

上述过程讨论了单一引擎得到的标签序列的特性，接下来介绍不同引擎标签序列合并、整合的常见方式。目前研究者们普遍采用阈值 t 来作为整合的基准线，即超过 t 个引擎认为提交文件是恶意的则最终给出恶意标签。下面讨论阈值对测试结果的影响：

- $t=1$： 任一引擎给出恶意标签则认为当前文件是恶意的，此时仅有 9.4% 的文件能得到良性的测试结果，49.0% 的文件只有恶意的测试结果，其余的文件（41.6%）标签序列有变化。这意味着如果一个文件重复提交检测，容易得到截然不同的结论。这样的结果显然是难以接受的，所以 $t=1$ 并不是一个合适的阈值。
- $2 \leq t \leq 39$： 在 $t=2$ 时，良性标签的比重显著提升，上升到了 48.4% ，而有变化的序列比例缩小到3.1%。但是在 $t>31$ 后， $flippinglables$ 的比重逐渐增大。由此可以看出这一阈值区间受 flips 影响较小，为减少 $flips$ 对结果的影响，阈值的最佳区间为 $[2,31]$ 。
- $t \geq 40$： 此时更多的文件将会有标签变化，$t=40$ 时，53.3% 的文件只被标注为良性，16.4% 的文件只被标注为恶意的。当 t 增大到 50 ，则我们可以看到受 $flips$ 影响的文件比重明显上升。
- 注意到，如果在$t=1$的情况下，去除$hazards$能显著改善受 $flips$ 影响的程度，有标签波动的文件比例下降到 3.8% 。

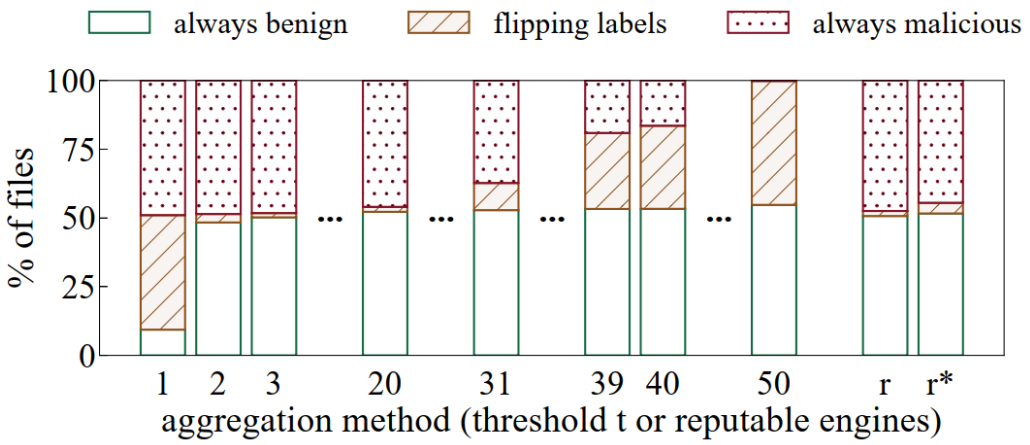


图3.引进阈值后的标签测试结果

由此，我们可以得到结论：引进阈值有助于标签动态整合结果趋于稳定，但这并不意味着这个检测结果就一定正确，即准确性无法通过阈值分析得到保证。

四、不同引擎间标签动态的关联分析

在分析标签序列时，作者提到引入阈值 t 有助于提高测试结果对标签动态变化的适应性与耐受性。然而仅仅分析单个引擎的分析结果是远远不够的，本文对不同引擎间得到标签序列之间的关联进行分析，并得到了每个系统的易受影响程度。

为了比较不同引擎对同一文件的标签序列，本文引进了两个序列间距离 $distance$ 的概念，对于序列 A 和序列 B，有

$$distance = \sum_{i=1}^{days} |A_i - B_j|$$

例如，序列 A = “01000000000000” 和序列 ” B = “00000000010000” 的距离为 2 。

接着对标签序列进行编码，将原始序列分为一个个固定大小的块，在每个块内分别统计：

- $0 \rightarrow 1$ 的个数
- $1 \rightarrow 0$ 的个数
- 连续为 1 的子序列最大长度
- 连续为 0 的子序列最大长度

从而组成一个四维特征向量，若 L 定义为序列长度，S 定义为块的长度，则一个标签序列的总特征向量维度数为：

$$D = 4 * \lceil \frac{L}{S} \rceil$$

还是以上述序列 A、B 为例，令块长 $S = 7$，则恰好可分为两块。对于序列 A= “0100000”+“00 00000” 而言，两块的特征向量分别为 $[1,1,1,5]$ 和 $[0,0,0,7]$ ，故整个序列的特征向量为 $[1,1,1,5,0,0,0,7]$ 。同理可得序列 B=“0000000”+“0010000” 的特征序列为 $[0,0,0,7,1,1,1,4]$ 。对于得到的两个向量求**余弦相似度**为0.871，再对 A、B 两个引擎的所有序列生成特征向量、计算余弦相似度并求均值，即可得到两个引擎间的相似指数 **ss** 。接着定义 **距离** $d = 1 - ss$ 为分析不同引擎间的关联，我们可以使用聚类算法将距离较小的引擎归为一个大类里。定义 t_d 为划分量度，距离小于 t_d 的引擎分为同一个群组，则随着我们增加 t_d ，群组的个数将逐渐增加（注意到我们只统计满足组内元素大于1的群组个数）。当 $t_d = 0.01$ 时，我们能得到最大的群组个数（5个），这5个群组包含了16个引擎，表明这16个引擎之间高度相似。当t_d增大到0.2时，所有元素个数大于1的群组合并为同一个大的集合，共含有28个引擎，即这28个引擎的相似度较高。

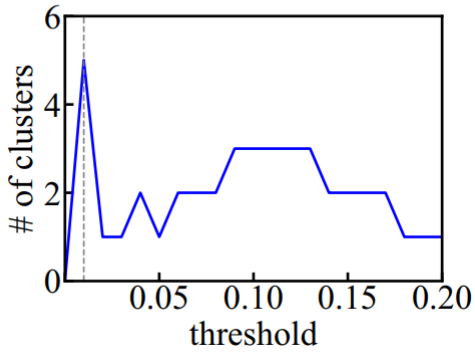


图4.距离阈值与群组个数的关系

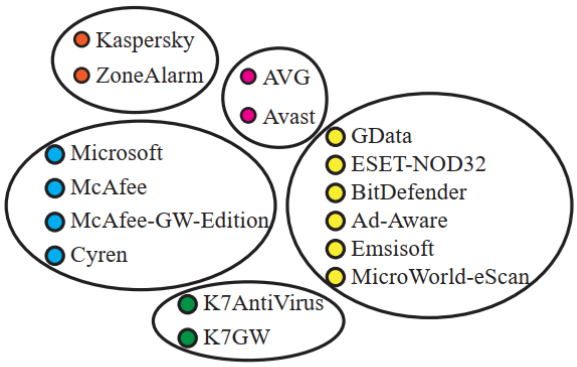


图5. td=0.01时群组分类结果

通过上述分析结果，本文提出了以下两种引擎间关联模型

- 主动影响模型（$Active\ Influence\ Model$）

  主动模型用于讨论短时间内不同引擎的$flips$之间的因果关系，作者定义一个影响事件为在，在时间窗口w下，引擎 i 的 $flips$ 发生在引擎 j 的 $flips$ 发生时刻 t 之前 w 时间内，即$[t-w,t)$.

  定义 A_{i2j} 为引擎 i 到引擎 j 通过所有文件的影响事件数，而 A_i 定义为引擎 i 的 $flips$ 总数,则

$$p_{i,j}=\frac{|A_{i2j}|}{|A_i|}$$

  定义为从 i 到 j 的主动影响分数($active\ influence\ score$)。对主动影响分数作出热点图，可以发现图中有一些亮线，即这些列所表示的引擎极易受到其他引擎检测结果的影响。热点图如下：

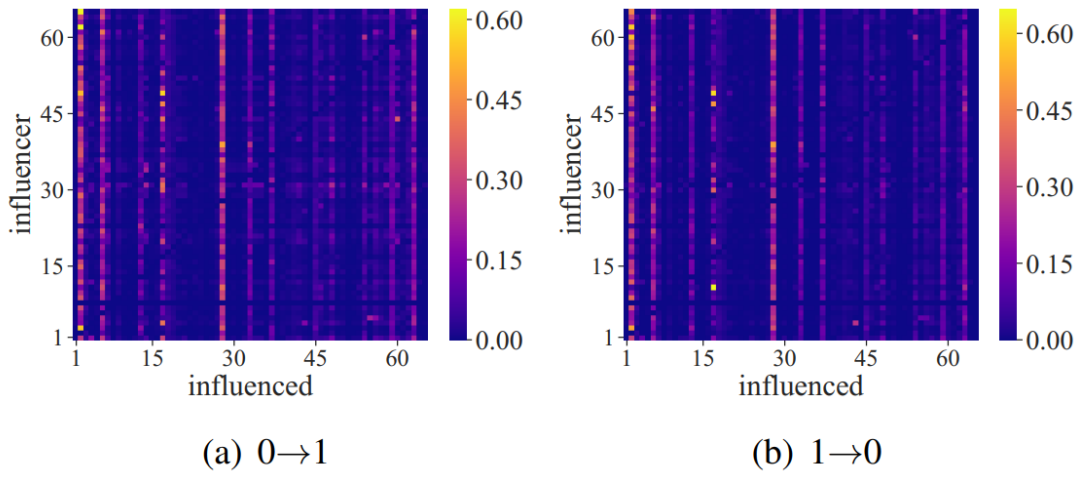
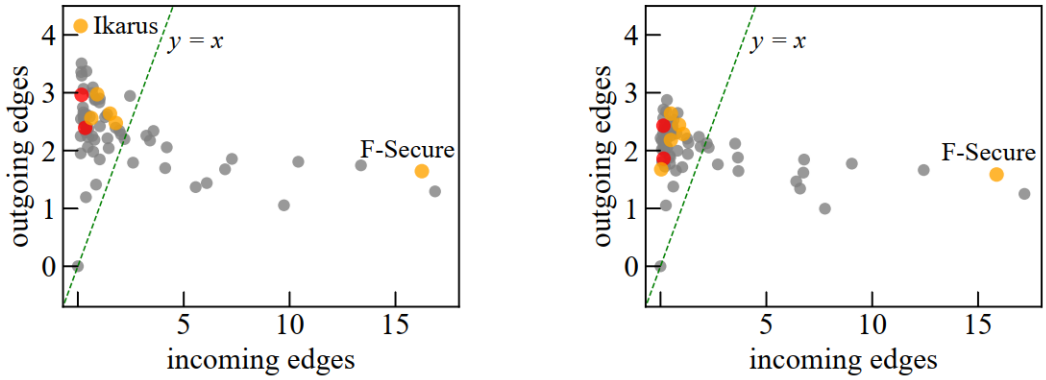


图6. 主动影响模型的热点图

再考虑建立主动影响有向图，(i,j)的边权定义为 i 到 j 的主动影响分数，建好图后统计每个节点的入边($incoming\ edges$)权重和与出边($outgoing\ edges$)权重和，作出分布图。图中用黄色点表示之前提到过的高声望引擎，用红色表示之前提到过的2个最高声望引擎，并作出$y=x$分界线，可以清晰看出分界线以上的模型的检测结果会影响其他模型的结论，而分界线以下的模型容易受到其他模型的影响。而且，具有高声望的引擎一般对其他模型有着更大的影响力（$F-Secure$引擎除外）。分布图如下：



(a) 0→1　　(b) 1→0

图7. 主动影响模型的入边出边分布图

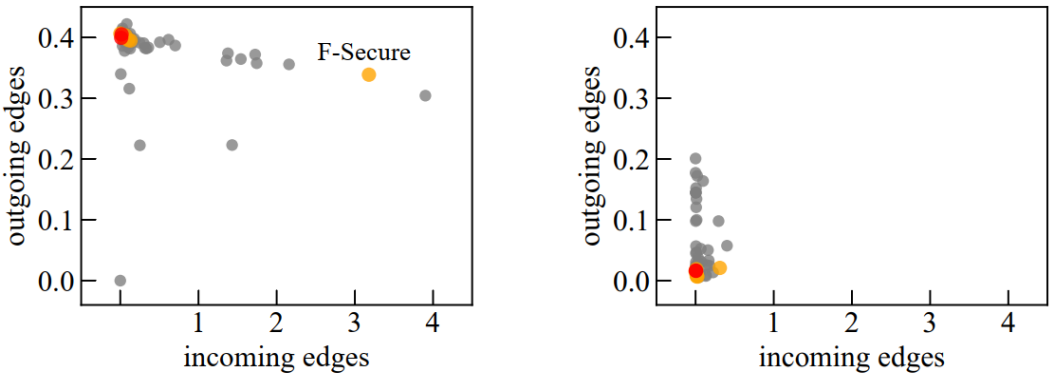
- 被动模型（$Passive\ Model$）

  作者定义一个被动事件为，引擎 j 因为引擎 i 保持在某一标签状态 l 长达 w 时间而发生 $flips$ 变为与引擎 i 一样的状态 l 。在被动模型中，定义 A_{i2j} 为引擎 i 到引擎 j 被动影响事件数，而 A_i 定义为引擎 i 的连续 7 个标签 l 的子序列总数,则

$$p_{i,j}=\frac{|A_{i2j}|}{|A_i|}$$

  例如，序列“11111111100011111” 和标签$l=1$，则$|A_i|=4$.

与主动模型同理，作出被动影响图，容易发现此时$0\rightarrow 1$的情形与$1\rightarrow 0$的情形有了显著差异：系统检测结果由恶意到良性的$flips$更容易受到最近发生相同$flips$的系统影响，而不是保持良性标签的系统。



(a) 0→1　　(b) 1→0

图8. 被动影响模型的入边出边分布图

五、模型结论的验证分析

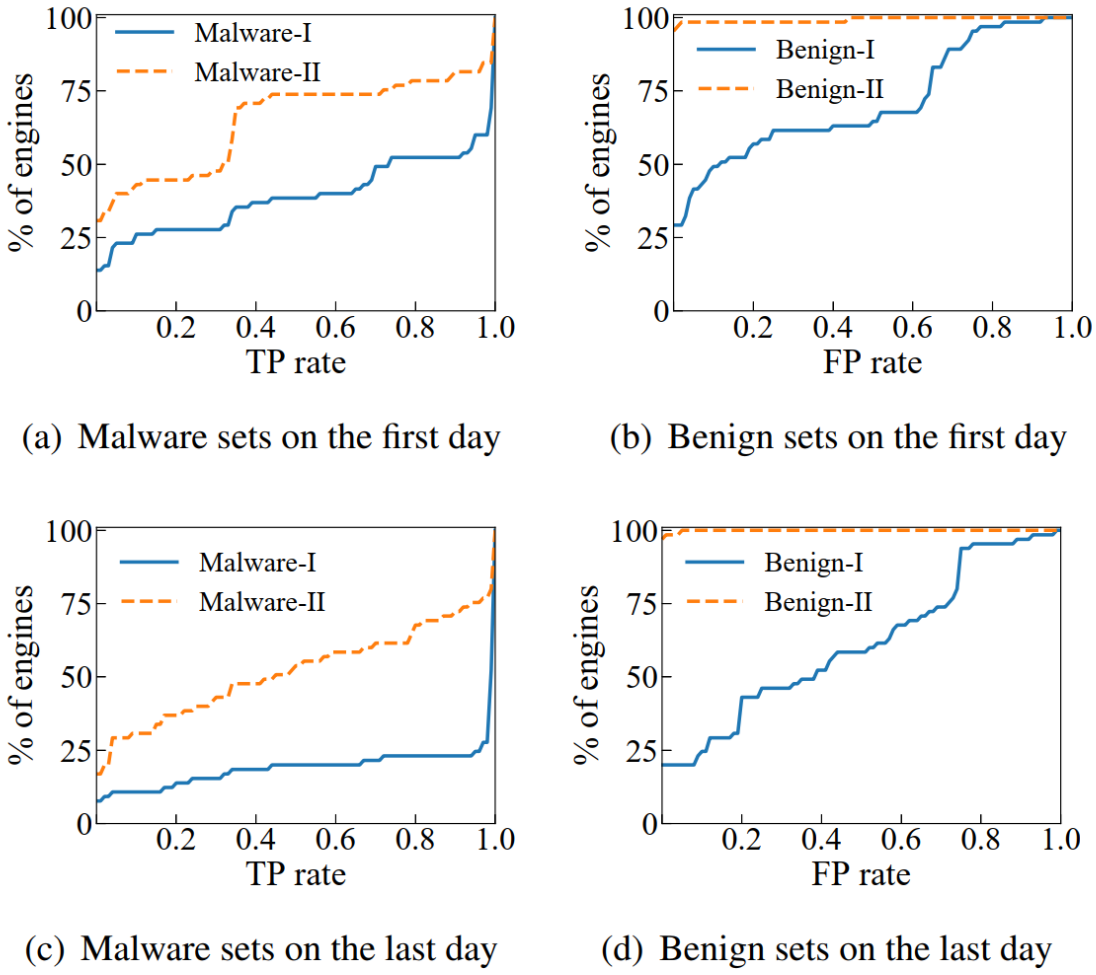
作者使用已经正确标注的数据($ground-truth$)对之前的结论进行统计验证，对于检测结果计算正确率，即

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

参考数据集包括：

Dataset	Files	Type	Days
Main	14423	U	396
Malware-Ⅰ	60	M	105
Malware-Ⅱ	60	M	97
Benign-Ⅰ	80	B	93
Benign-Ⅱ	156	B	70

$Malware$数据集即常见的勒索软件等，$Benign$数据集采用常规的排序程序等。
重复前面叙述的实验统计，作图如下：



(a) Malware sets on the first day (b) Benign sets on the first day

(c) Malware sets on the last day (d) Benign sets on the last day

图9. TP+FP

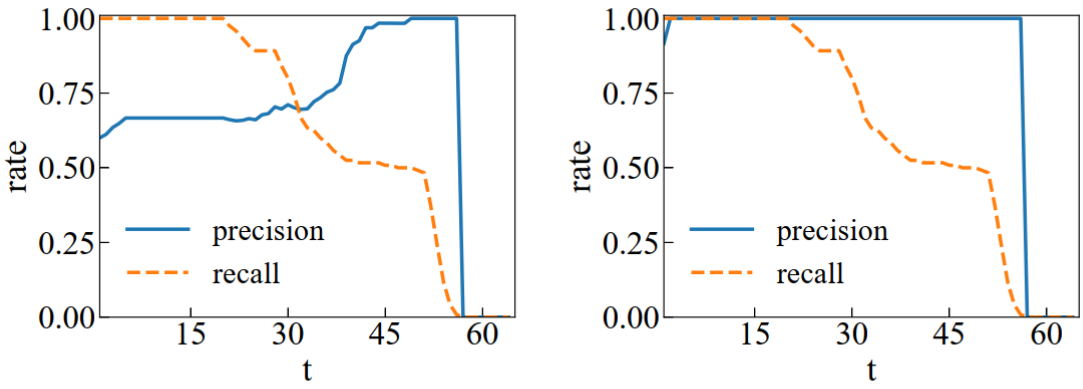


图10. 不同阈值t下，所有引擎的准确率与召回率

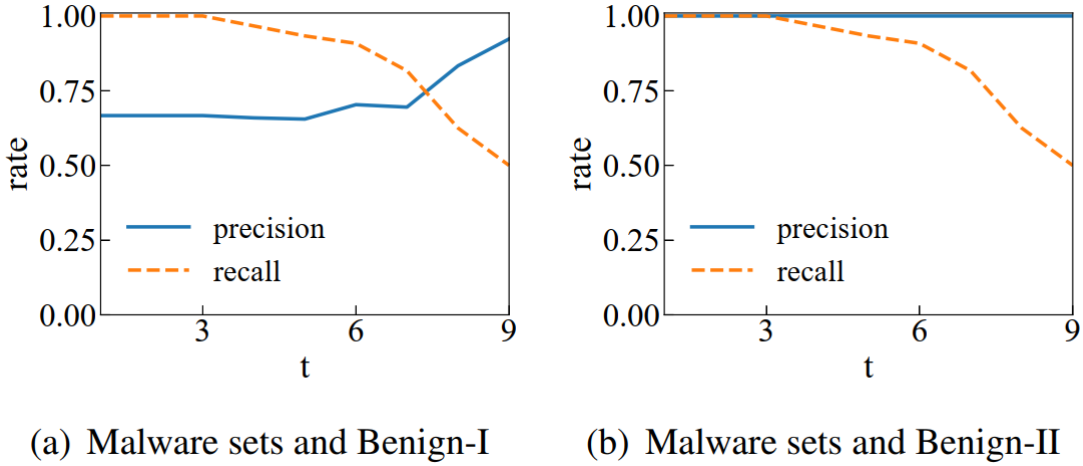


图11. 不同阈值t下，只考虑高声望引擎的准确率与召回率

六、总结与评价

本文提出了 $VirusTotal$ 恶意软件在线分析平台当前存在的问题，并基于数据驱动的模型，指出采信结果时应去除频繁反转的部分，探索选取合适的阈值应用于标签动态整合最终测试结果，通过建立不同引擎间的关联证明了其中一些引擎存在紧密联系，部分系统易受其他系统影响，推翻了所有引擎平权综合的假设，最后用正确标注好的参考数据来检验结论的正确性。

作者也提到了本文的局限性与不足。比如，没有采用多种多样的数据集，主要数据来源于可移植的可执行文件($PortableExecutable$)，恶意软件参考数据集则选取了常规的勒索软件。最后，作者提到未来将会拓展本文的实验到各种类型的文件，使用更加丰富的数据集，并且探索新的恶意软件标签整合规则与策略。

七、参考文献

[1] S. Zhu, J. Shi, L. Yang, B. Qin, Z. Zhang, L. Song, and G. Wang. Measuring and modeling the label dynamics of online anti-malware engines. In Proc. of USENIX Security Symposium, 2020
[2] Zhu's presentation slides
[3] Presentation Video (https://www.bilibili.com/video/BV1Rk4y127cQ/)