# 恶意软件图像：可视化与自动分类

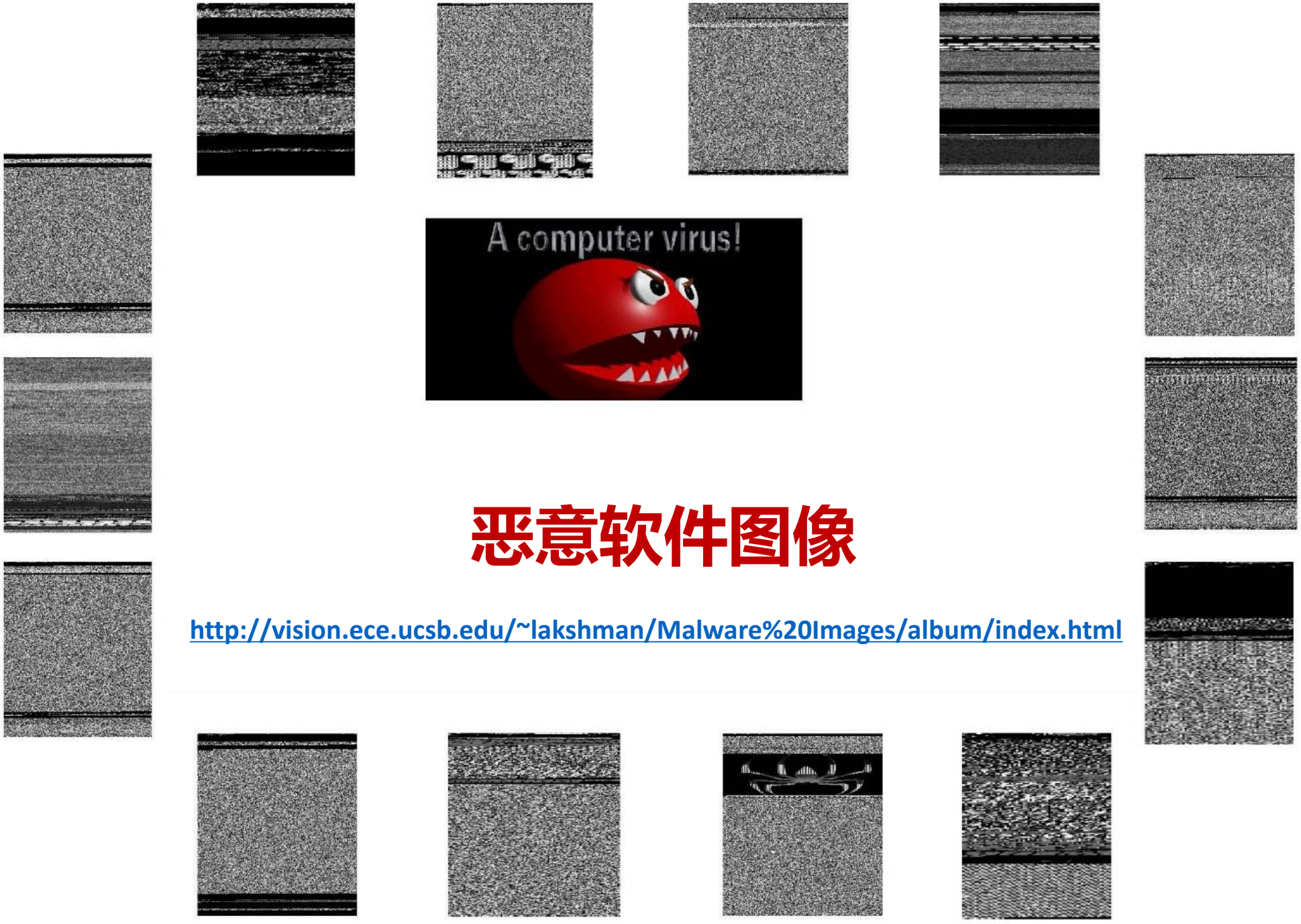
作者：Lakshmanan Nataraj
加州大学，圣芭芭拉分校，视觉研究实验室

# 恶意软件图像：可视化与自动分类

## 非官方中文译本·安天实验室 译注

| 文档信息 | | | |
|---|---|---|---|
| **原文名称** | Malware Images: Visualization and Automatic Classification | | |
| **原文作者** | Lakshmanan Nataraj | 原文发布日期 | 2011年7月20日 |
| **作者简介** | Lakshmanan Nataraj是加州大学圣芭芭拉分校的研究生，对恶意软件分析领域做过多个研究。http://www.linkedin.com/profile/view?id=14912787&authType=NAME_SEARCH&authToken=Tcys&locale | | |
| **原文发布单位** | 加州大学圣芭芭拉分校 | | |
| **原文出处** | http://vision.ece.ucsb.edu/publications/nataraj_vizsec_2011_paper.pdf | | |
| **译者** | 安天技术公益翻译组 | 校对者 | 安天技术公益翻译组 |
| **免责声明** | 本译文译者为安天实验室工程师，本文系出自个人兴趣在业余时间所译，本文原文来自互联网的公共方式，译者力图忠于所获得之电子版本 | | |

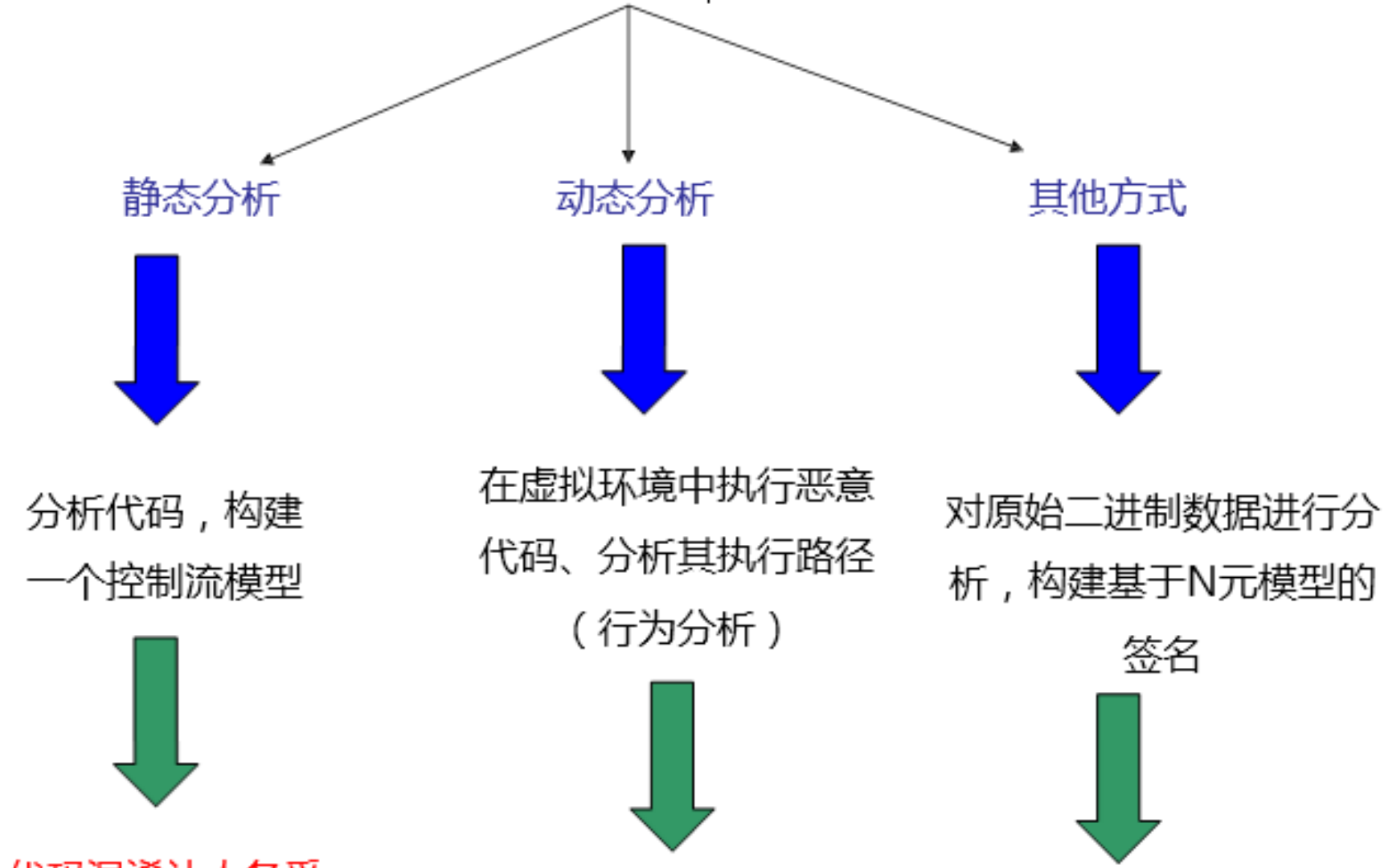
| | |
|---|---|
| **免责声明** | 进行翻译，但受翻译水平和技术水平所限，不能完全保证译文完全与原文含义一致，同时对所获得原文是否存在臆造、或者是否与其原始版本一致未进行可靠性验证和评价。<br>本译文对应原文所有观点亦不受本译文中任何打字、排版、印刷或翻译错误的影响。译者与安天实验室不对译文及原文中包含或引用的信息的真实性、准确性、可靠性、或完整性提供任何明示或暗示的保证。译者与安天实验室亦对原文和译文的任何内容不承担任何责任。翻译本文的行为不代表译者和安天实验室对原文立场持有任何立场和态度。<br>译者与安天实验室均与原作者与原始发布者没有联系，亦未获得相关的版权授权，鉴于译者及安天实验室出于学习参考之目的翻译本文，而无出版、发售译文等任何商业利益意图，因此亦不对任何可能因此导致的版权问题承担责任。<br>本文为安天内部参考文献，主要用于安天实验室内部进行外语和技术学习使用，亦向中国大陆境内的网络安全领域的研究人士进行有限分享。望尊重译者的劳动和意愿，不得以任何方式修改本译文。译者和安天实验室并未授权任何人士和第三方二次分享本译文，因此第三方对本译文的全部或者部分所做的分享、传播、报道、张贴行为，及所带来的后果与译者和安天实验室无关。本译文亦不得用于任何商业目的，基于上述问题产生的法律责任，译者与安天实验室一律不予承担。 |



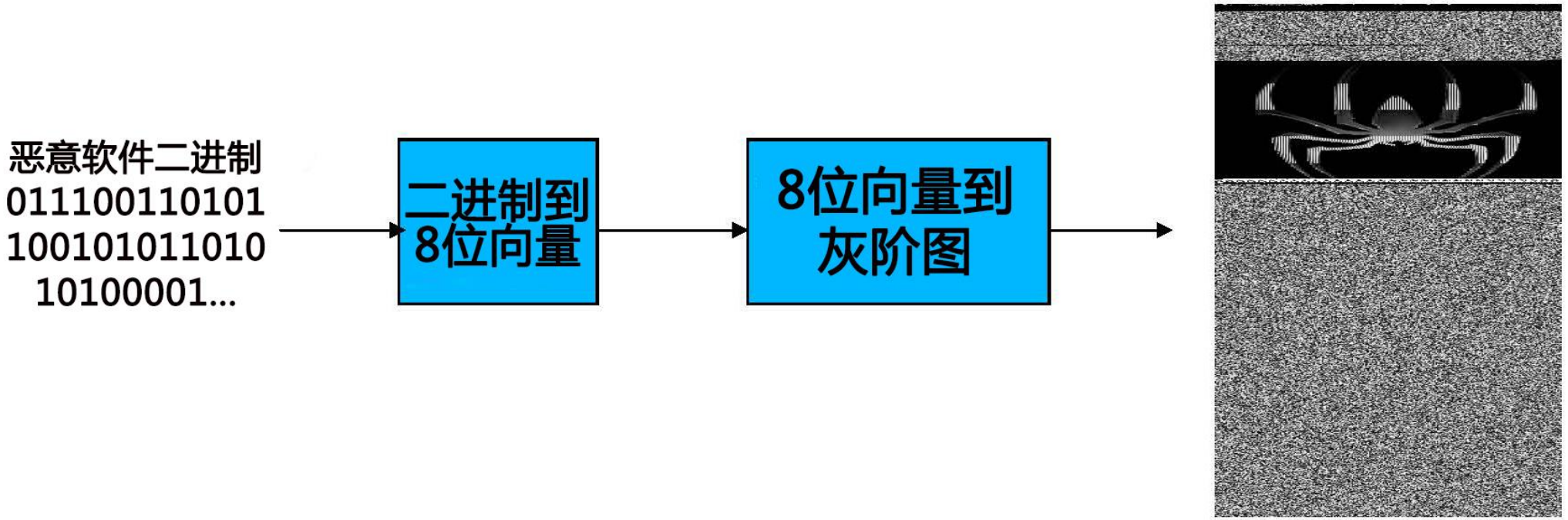

# 恶意软件图像

http://vision.ece.ucsb.edu/~lakshman/Malware%20Images/album/index.html

# 恶意软件分析


静态分析
动态分析
其他方式
分析代码，构建一个控制流模型
在虚拟环境中执行恶意代码、分析其执行路径（行为分析）
对原始二进制数据进行分析，构建基于N元模型的签名
代码混淆让人备受折磨
有希望、但既复杂又耗时（几秒到几分钟）
不要列出太多信息

# 恶意软件图像：下一个选择



# 为什么要用图像呢？

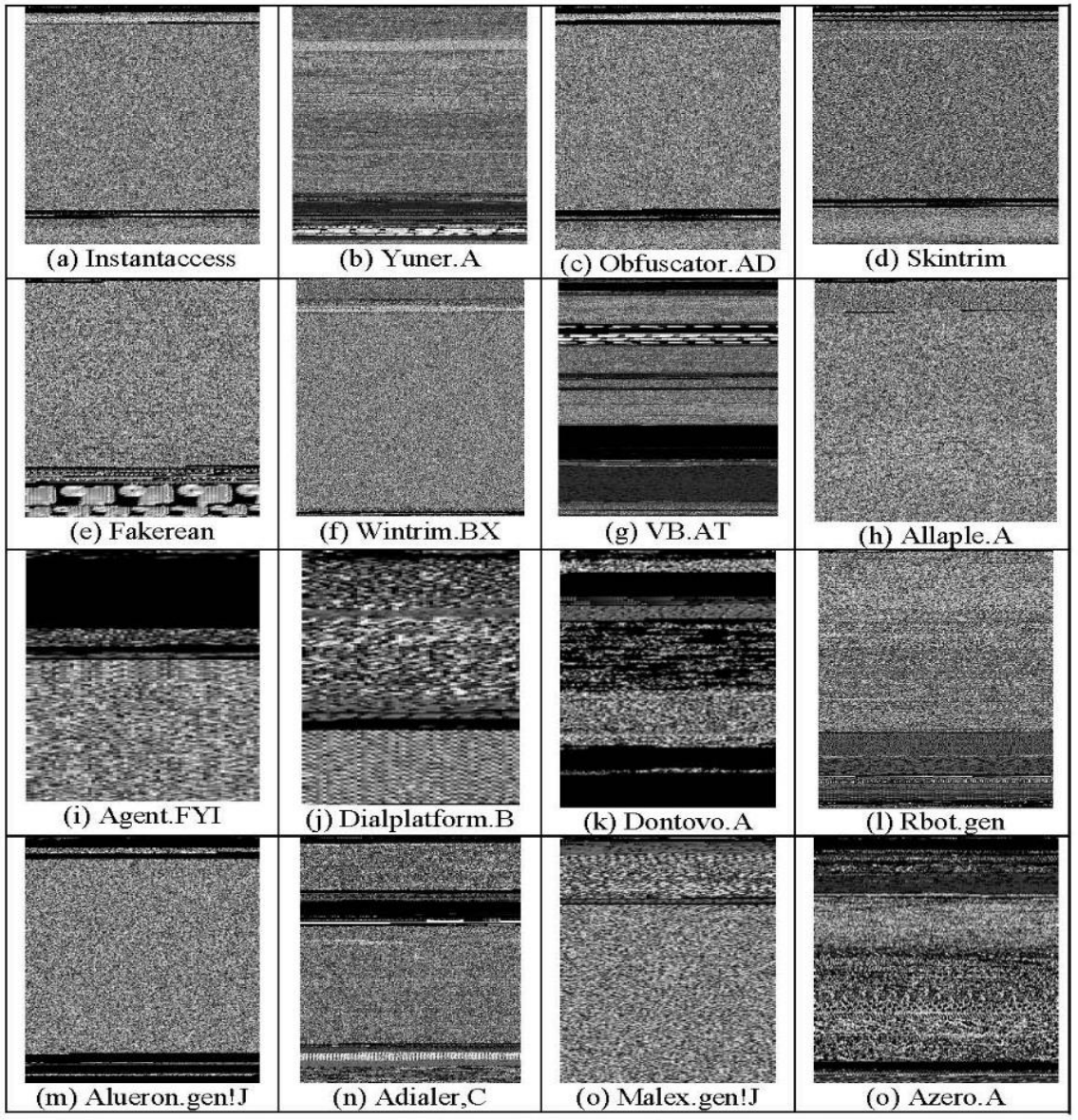
- 不同的二进制字节以图像的形式存在时，可以轻易被看到。

  » 可视化

- 恶意程序的编码人员将原始代码的一小部分进行改动，从而创造出一个新的变种。
- 图像可以捕获微小的变化，而保留整体结构。
- 因此，同一恶意代码家族中不同变种的图像是十分接近的。这些图像与其他恶意代码家族相比，则全完不同。

  »利用图像处理特征进行分类/聚类

# 各种恶意软件家族的图像

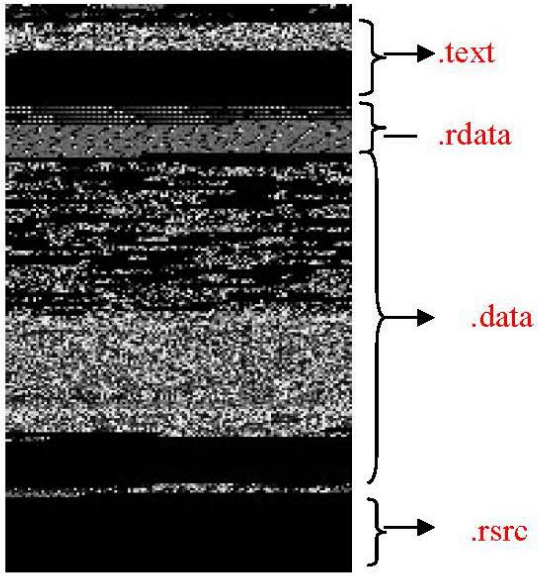



http://vision.ece.ucsb.edu/~lakshman/Malware%20Images/album/index.html

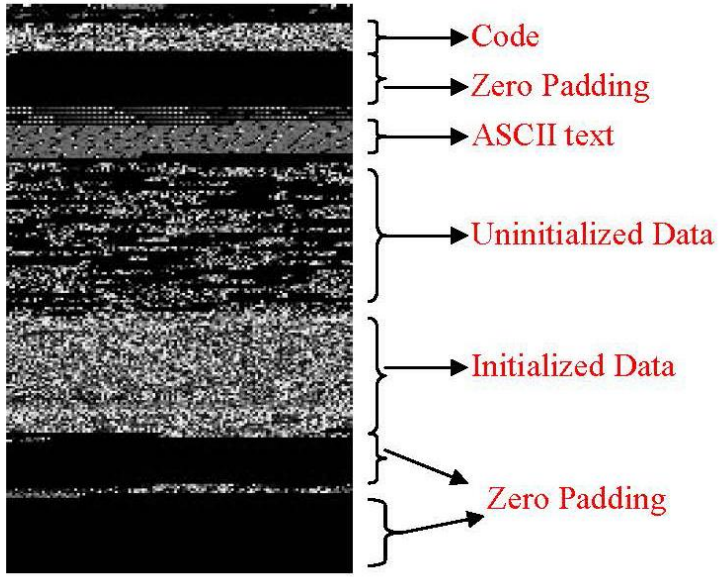
# 图像中的信息

- 图像可以展示恶意代码结构的更多信息。我们可以看到不同的分节有着不同的纹理。整体的结构布局也可以清楚地看到。

***PE文件*中的节***




***从图像中获得的信息***




**code.google.com/p/pefile/*

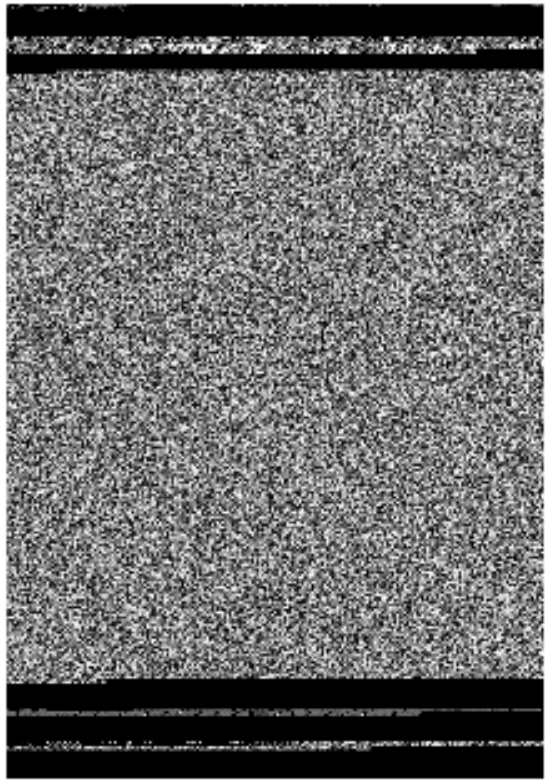
# 如何选择图像的宽度？

- 图像宽度的选择是根据可视化实验中文件的大小而定的。

- 图像的高度因文件大小的不同而有所差异。

| 文件大小范围 | 图像宽度 |
| --- | --- |
| <10 kB | 32 |
| 10 kB – 30 kB | 64 |
| 30 kB – 60 kB | 128 |
| 60 kB – 100 kB | 256 |
| 100 kB – 200 kB | 384 |
| 200 kB – 500 kB | 512 |
| 500 kB – 1000 kB | 768 |
| >1000 kB | 102 |

# 举例：变种1



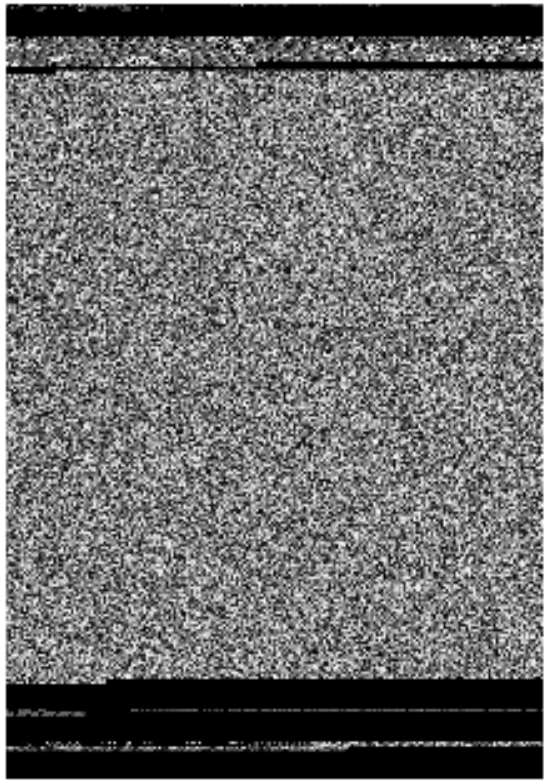
Alueron.gen!J

Dialplatform.B

Agent.FYI

Lolyda.AT

# 变种2



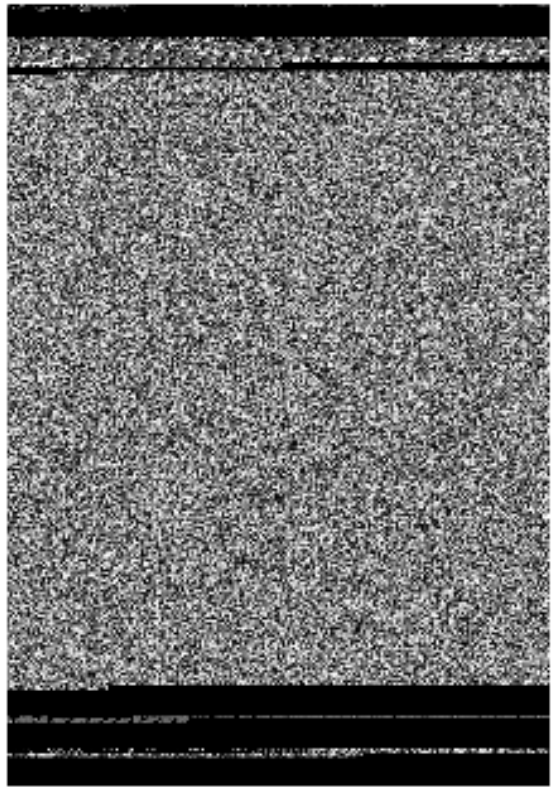
Alueron.gen!J

Dialplatform.B

Agent.FYI

Lolyda.AT

# 变种3



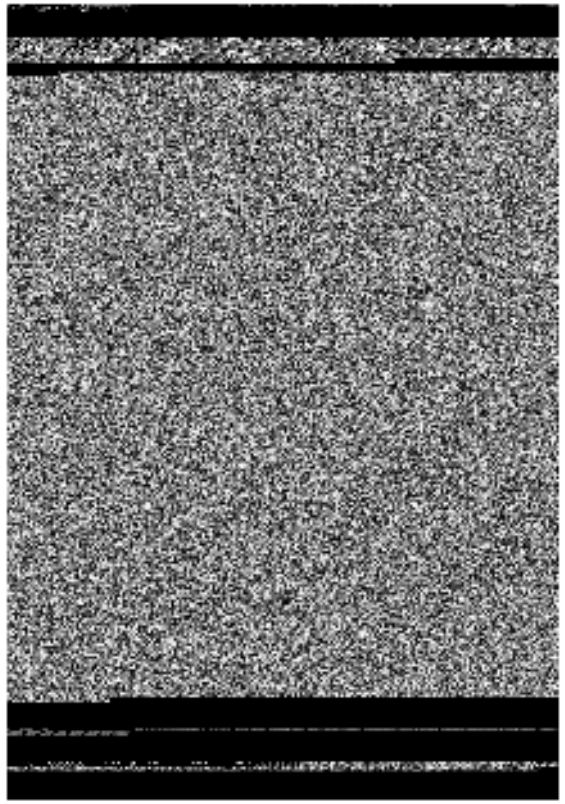
Alueron.gen!J

Dialplatform.B

Agent.FYI

Lolyda.AT

# 变种4



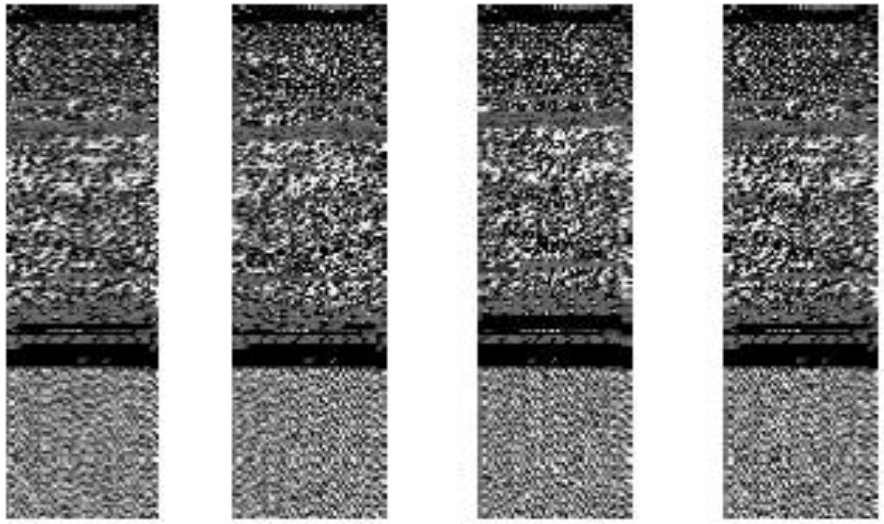
Alueron.gen!J

Dialplatform.B

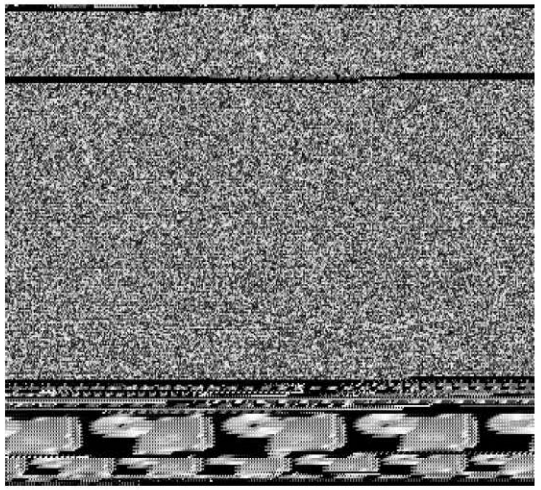
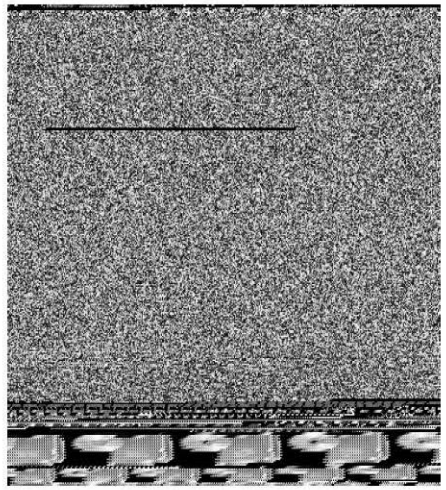
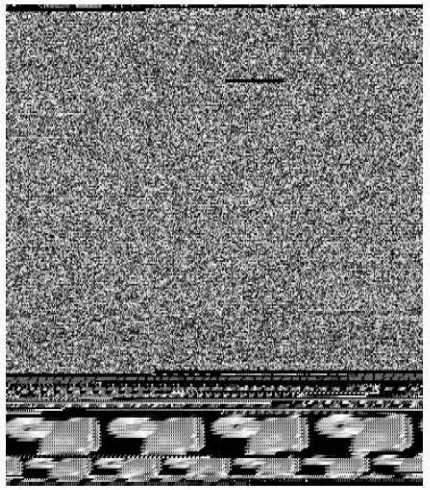
Agent.FYI

Lolyda.AT

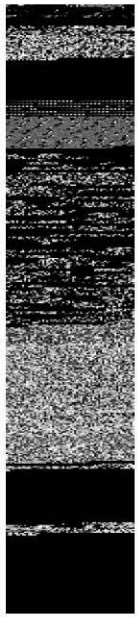
# Dialplatform.B的所有变种



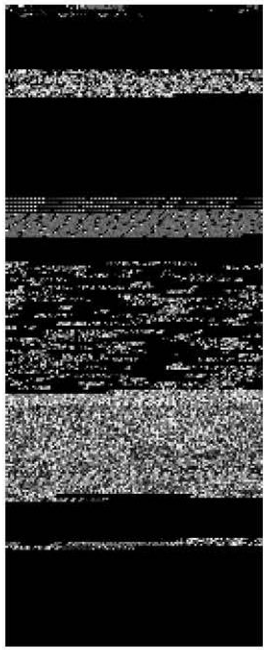
# 恶意软件图像的更多例子

Rogue: FakeRean



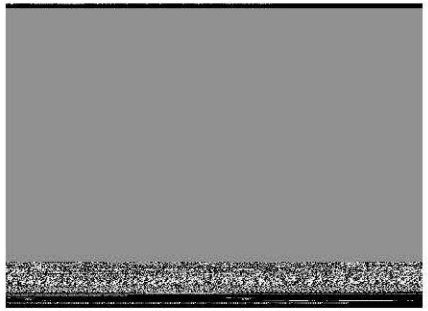
尽管文件大小各不相同，但从图像中可以看出它们的整体结构是相似的。

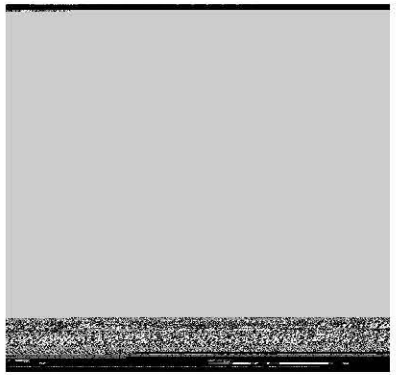



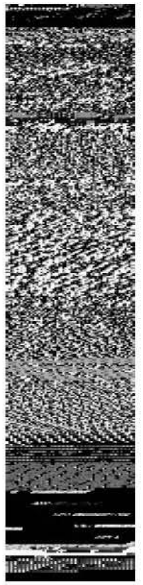
TrojanDownloader: Dontovo.A

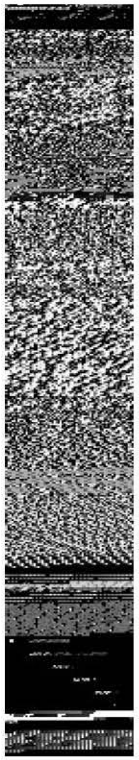
# 新的命名方案

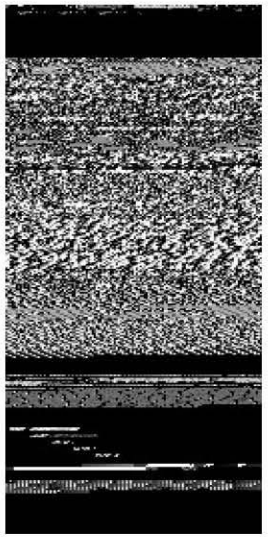
下面的恶意代码样本被微软杀毒软件（Microsoft Security Essential）命名为Lolyda.AA。但从图像中可以清楚地看到，他们可以再分为3类。

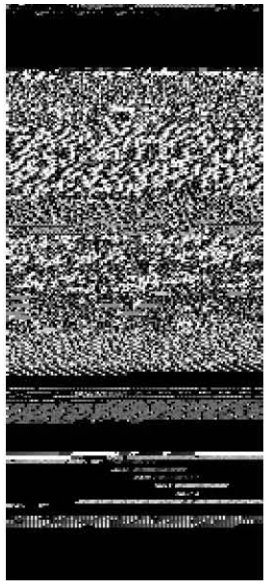


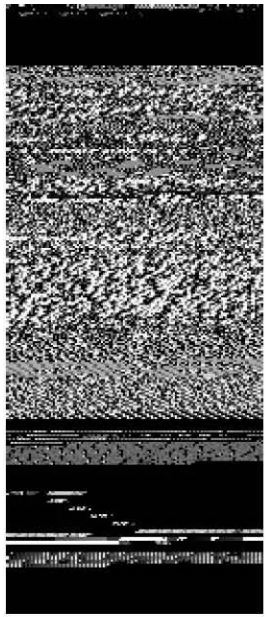












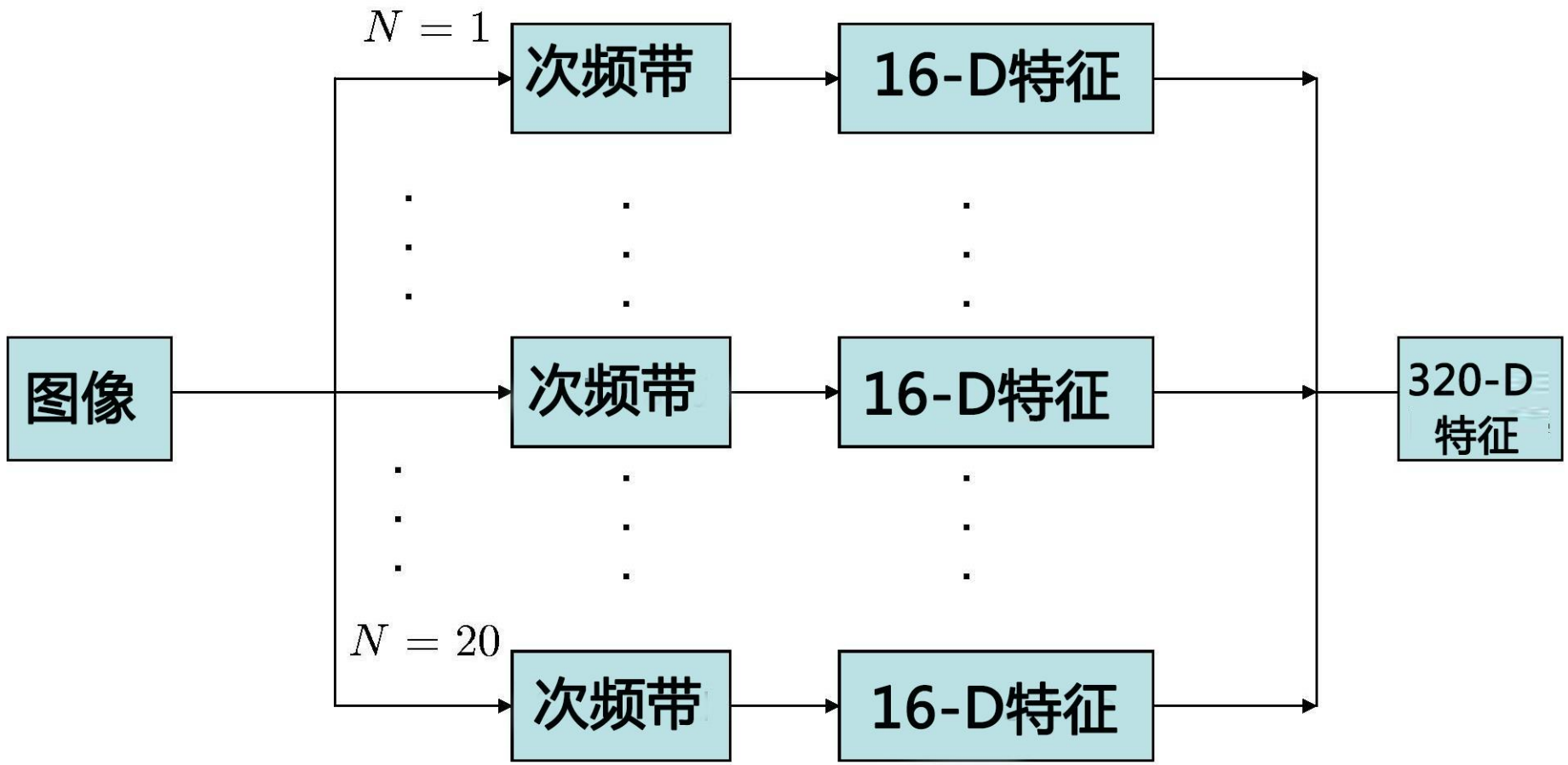
# 图像相似性分析

- 一旦恶意软件被转换为图像表示，基于图像的特征就可用来描述一类恶意软件。
- 我们利用基于图像的一种纹理特征，该特征常被应用于例如海岸、山峦、森林、街道等场景类别分类中。
- 而这里，我们将之应用于恶意软件家族分类。

# 结构特征

- 每个图像的位置都由过滤器调谐到不同方向及尺度的输出结果所表示。
- 此处利用了具有4个尺度和8个方向的可操纵式金字塔。
- 图像的局部表示由下式给出: $v^L(x) = \{v_k(x)\}_{k=1,N}$
- 这里N代表次频带的数量。
- 全局特征则平均表示为: $m(x) = \sum_{x'} |v(x')| w(x'-x)$
- 然后它们被降低到4×4的分辨率。

# 通用搜索树（GIST）特征计算



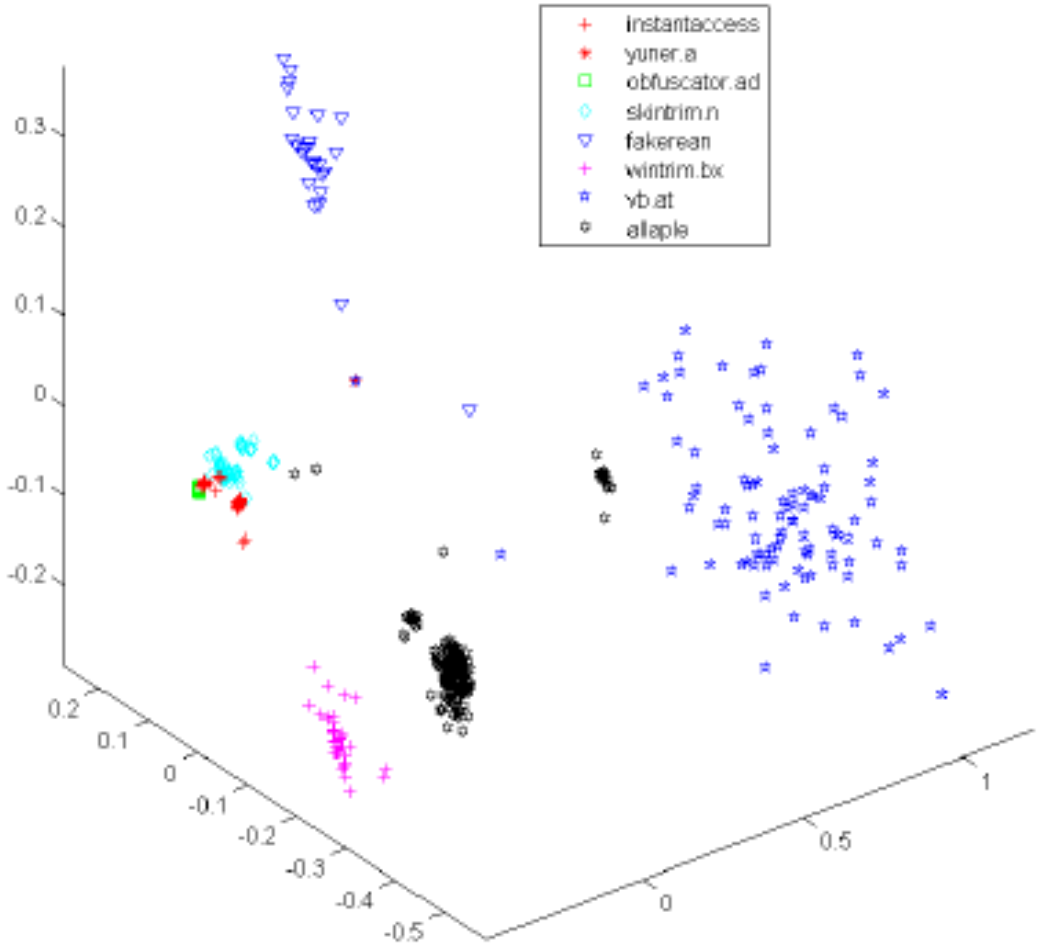
# 分类器

- 分类：k-近邻（k-nn）
  - – 如果某一测试样本在i家族的特征空间中有k近邻，则该样本属于i家族。
- 距离测量：欧几里德距离
  - – 为测量特征空间的距离，我们利用欧几里德距离（Euclidean Distance）作为距离测量标准。
- 10倍交叉验证。

# 基于特征的图像初步分类结果

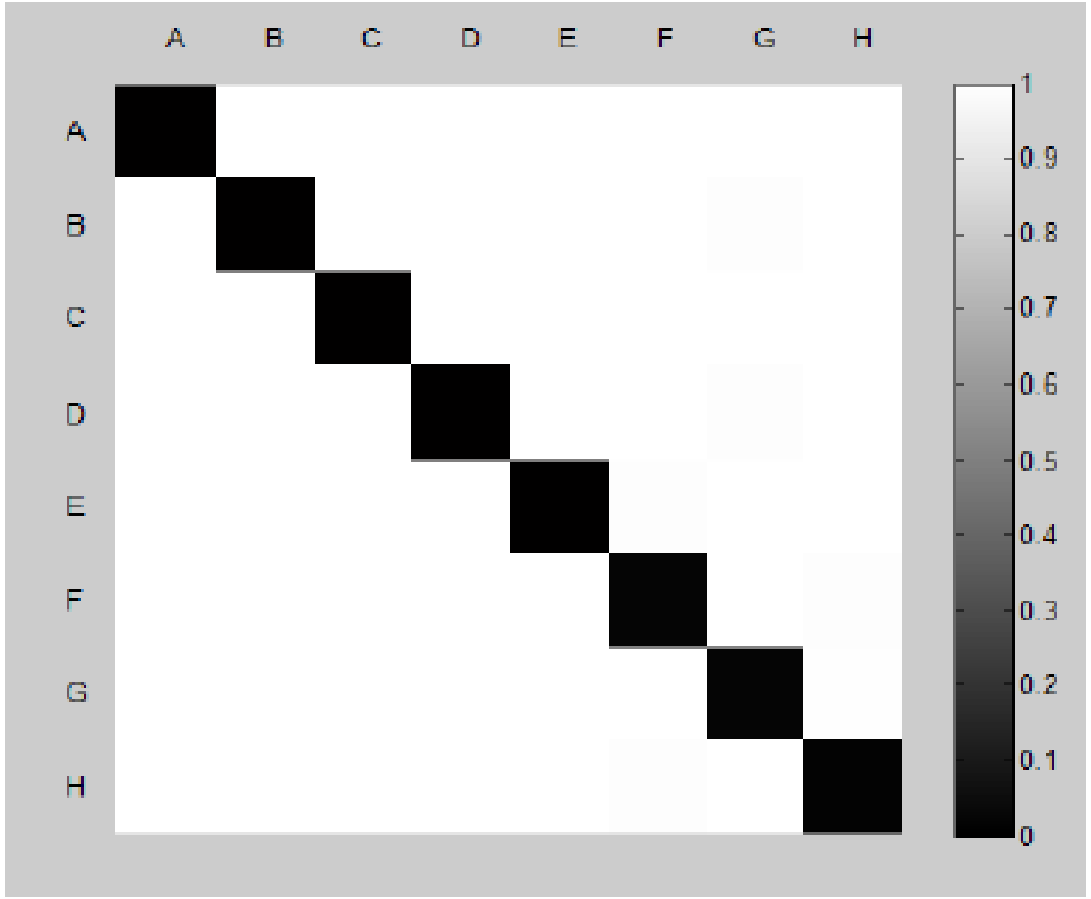
- 类属8个恶意代码家族的2000个恶意代码被转换为数字图像。[1]
- 在图像中推断基于特征的图像纹理（320维度）。
- k-nn分类器（k=3）生成准确率达98% 的分类。

**8大恶意软件家族特征图像的低维度映射**

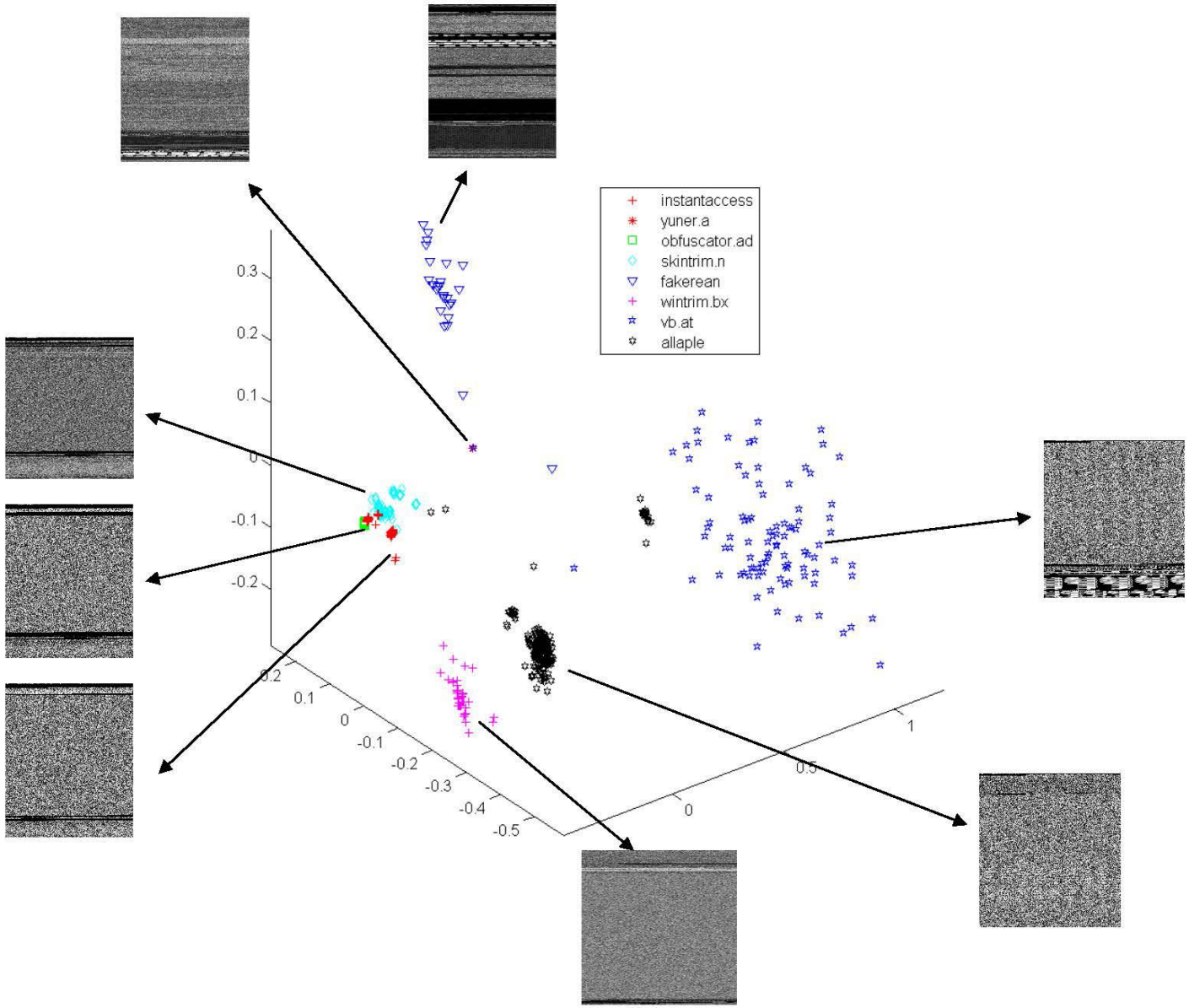



*[1]从Anubis（anubis.iseclab.or）获取并且用Microsoft Security Essentials命名的恶意代码。*

# 混淆矩阵 – 无混淆（几乎）



# 进一步观察


instantaccess
yuner.a
obfuscator.ad
skintrim.n
fakerean
wintrim.bx
vb.at
allaple
0.3
0.2
0.1
0
-0.1
-0.2
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
1
0.5
0

# 压缩会如何？

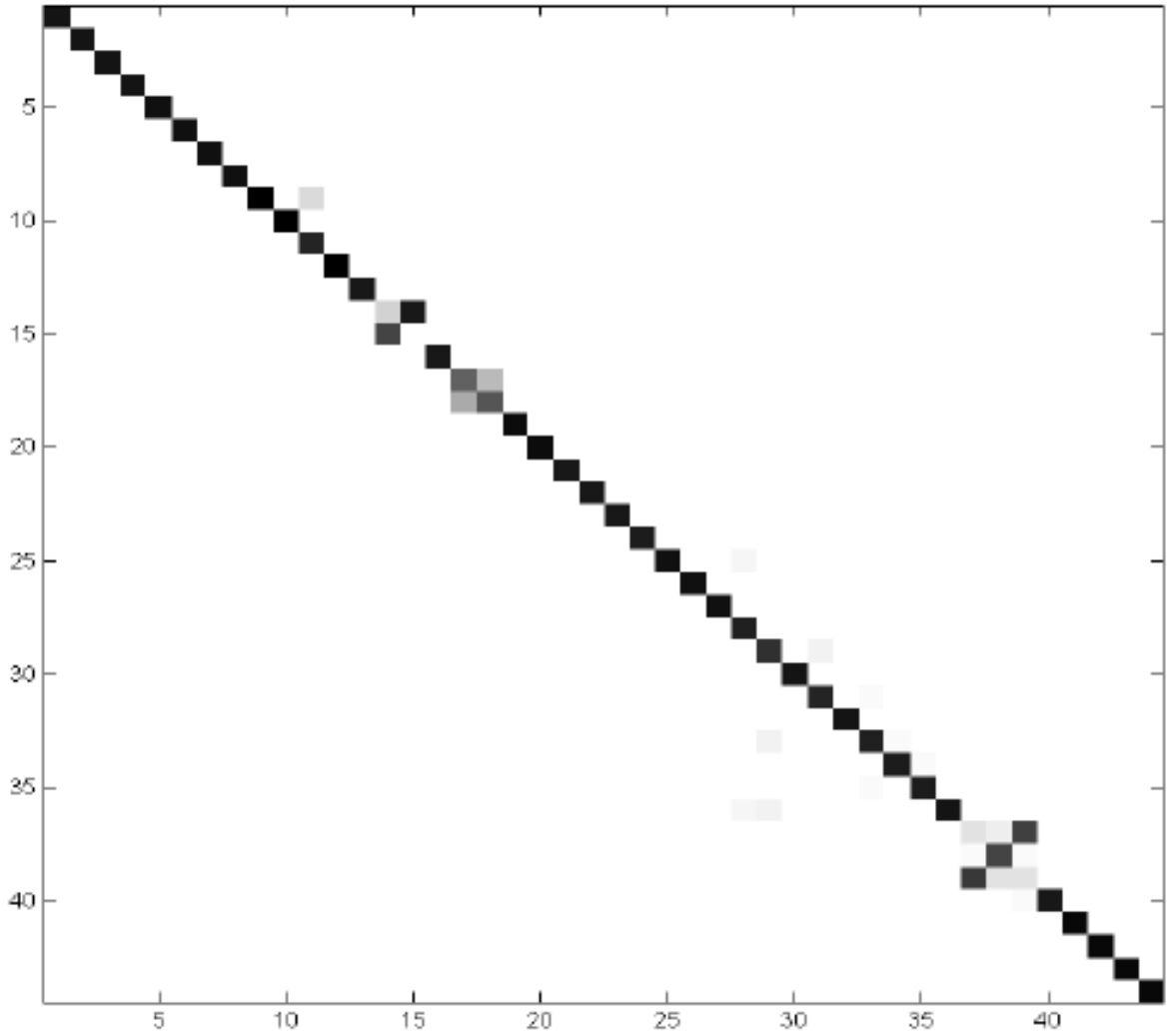
- 压缩可将二进制转换成一种完全不同的格式。
- 因此，经过压缩的图像“通常”会与之前截然不同。
- 一种常见的误解是，如果用相同的压缩器对从属不同家族的两种二进制文件进行压缩，则它们会呈现相同的形式。
- 但是，这并不是本例所说。为了对此验证，我们做了一个测试。

# 对压缩的可执行文件进行测试

- 利用UPX、Winupack和PeCompact对11个家族中未压缩的恶意代码进行压缩
- 经过压缩的恶意软件被当作新的家族。
- 现在家族总数为44个（包括未压缩的家族）。
- 分类实验再次运行。

| Adialer.C |
| --- |
| Adpclient |
| Agent.dz |
| Browsermodifier.cnnicc |
| Dontovo.A |
| Lolyda.AA |
| Lowsones.gen!B |
| Rbot.gen |
| Rootkit.gen!C |
| Vb.at |
| Yuner.A |

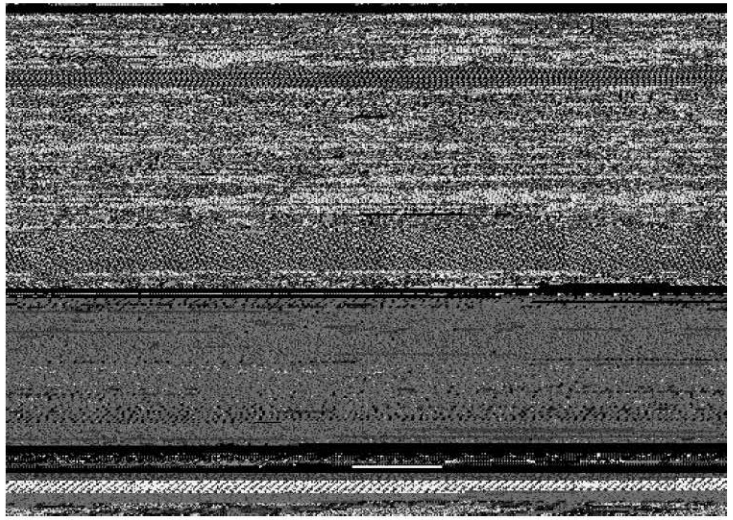
# 压缩测试的混淆矩阵



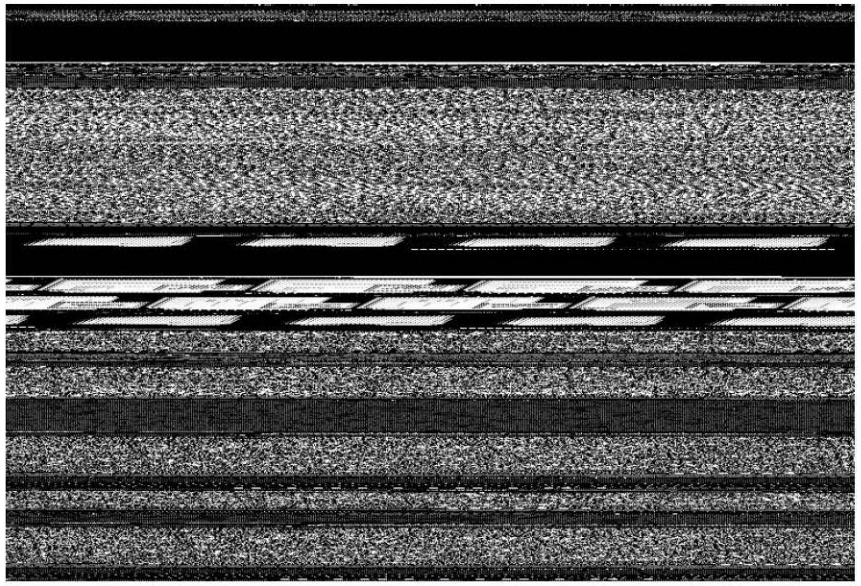

仅在家族内存在混淆，这也适用于压缩比例较小的恶意软件。

# 压缩的效果

压缩前

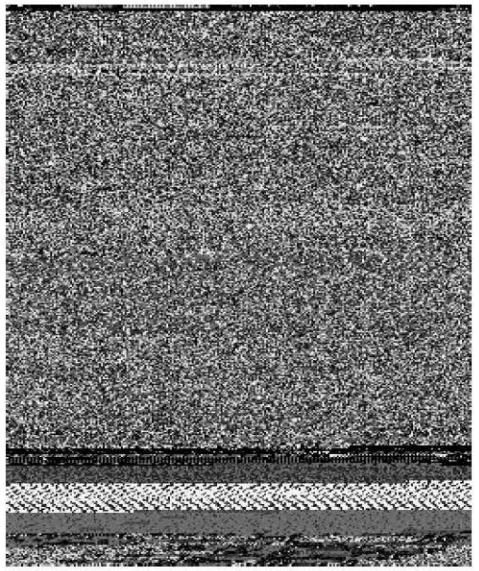


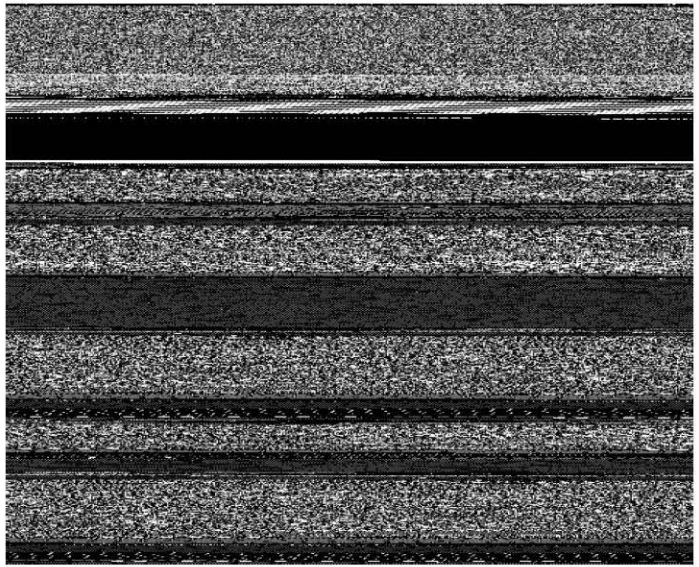
Adialer.C



由此，可分析压缩恶意软件与未压缩恶意软件之间的关系。

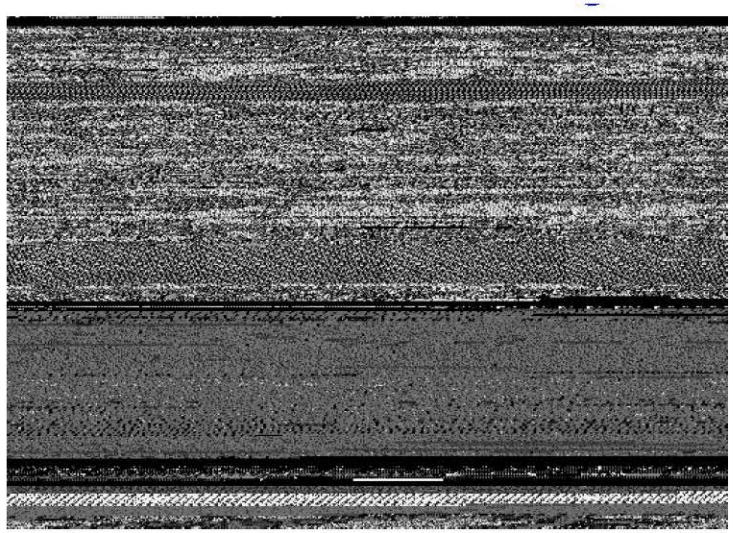
压缩后





# 压缩的效果

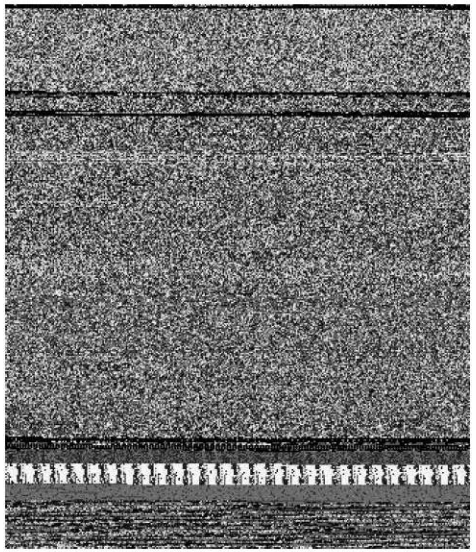
压缩前

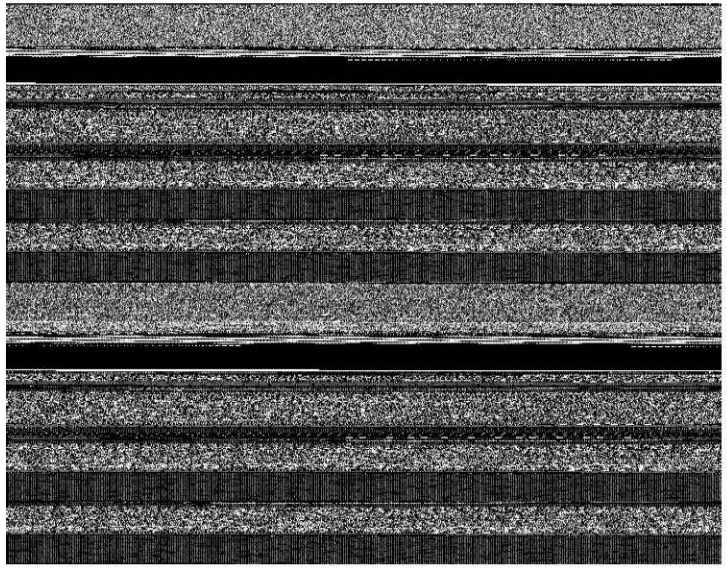


Adialer.C

由此，可分析压缩恶意软件与未压缩恶意软件之间的关系。

压缩后





# 经UPX压缩的 Dontovo.A

# 经UPX压缩的Agent.DZ

# 经UPX压缩的Lolyda.AA

# 对压缩的可执行文件的分析

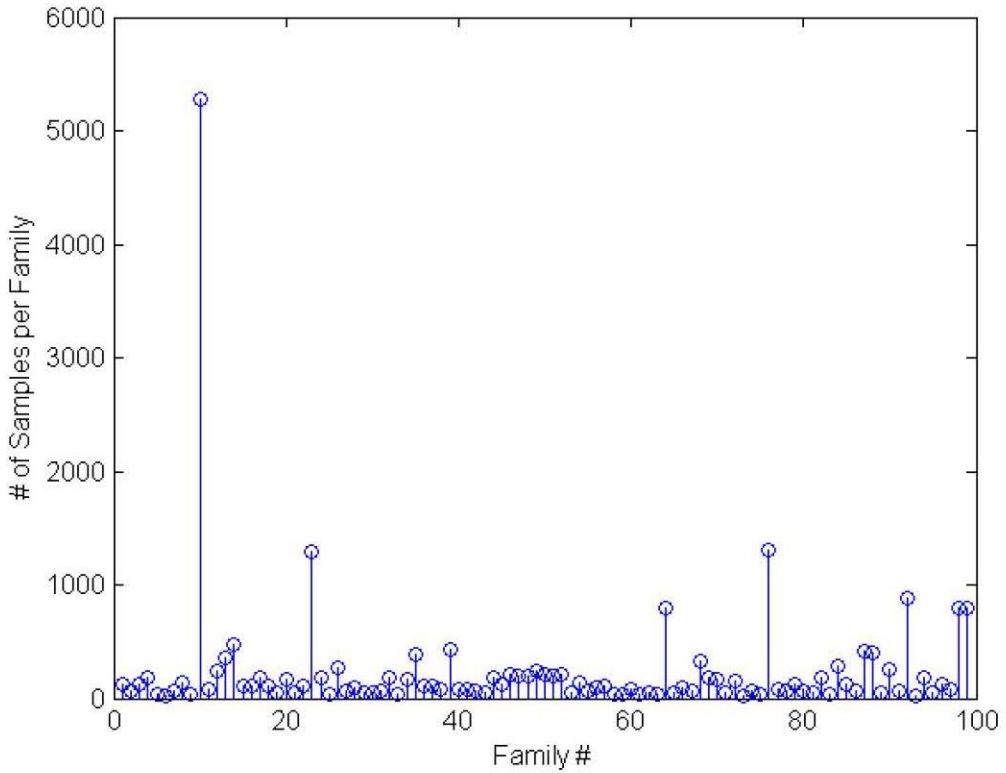
- 从初步分析中, 我们观察得出:
  - – 当使用特定压缩器对带有多个相似变种的未压缩的恶意软件家族进行压缩时, 那么新压缩的恶意软件的图像 (相同的家族) 也同样相似。
  - – 如果压缩率高，它们之间是相似的.
  - – 如果压缩率低, 那么它们与原始的未压缩的恶意软件家族相似。
- 目前我们正在进行更彻底的分析以支持我们的想法。

# 大规模的实验

- 来自Anubis 和 VxHeavens 数据集的25,000 个恶意软件。
- 以Microsoft Security Essentials标记的家族。
- 选择排名前100的家族。

# 一些数据集整理

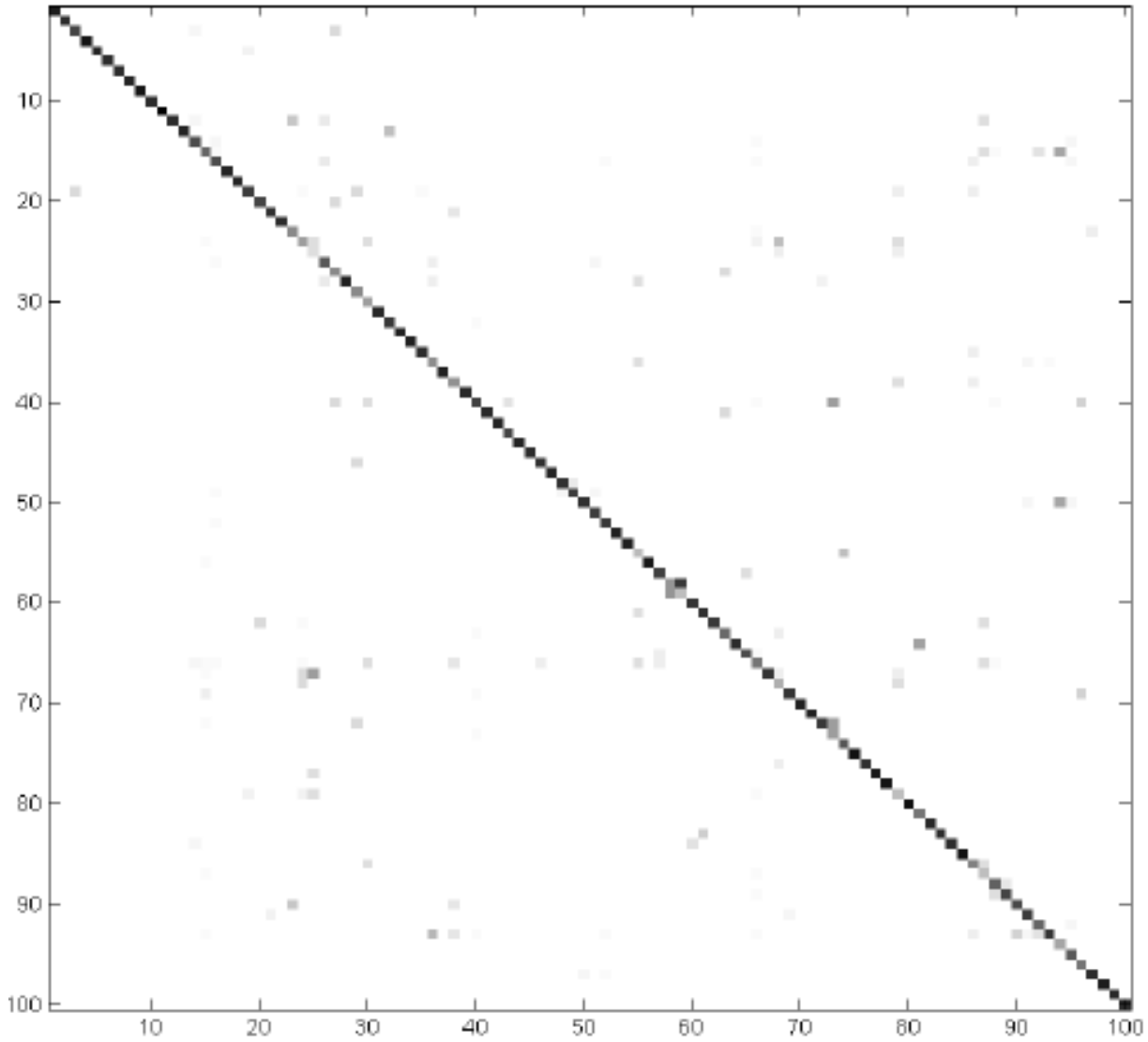
每个家族的样本




排名前11的家族

| |
|---|
| Allaple.A |
| Alueron.gen!j |
| Browsermodifier.cnnic |
| Instantaccess |
| Pcclient.bx |
| Seimon.D |
| VB.AT |
| VB.AT UPX |
| Vundo.gen!r |
| Yuner.A |
| Yuner.A UPX |

# 100个家族分类的混淆矩阵




k-nn = 3, 100 家族

# 高精确度的家族

| *家族名称* | *样本数量* |
| --- | --- |
| Instantaccess | 431 |
| Adialer.C | 63 |
| Adialer.G | 40 |
| Adpclient | 29 |
| Agent.Dz | 63 |
| Agent.Fyi | 140 |
| Agent.Wx (FSG) | 41 |
| Cnnic | 1287 |
| Dontovo.A | 162 |
| Hupigon.gen!A | 114 |

**精确度并不以每个家族的样本数量为依据**

# 一个高精确度家族的截图

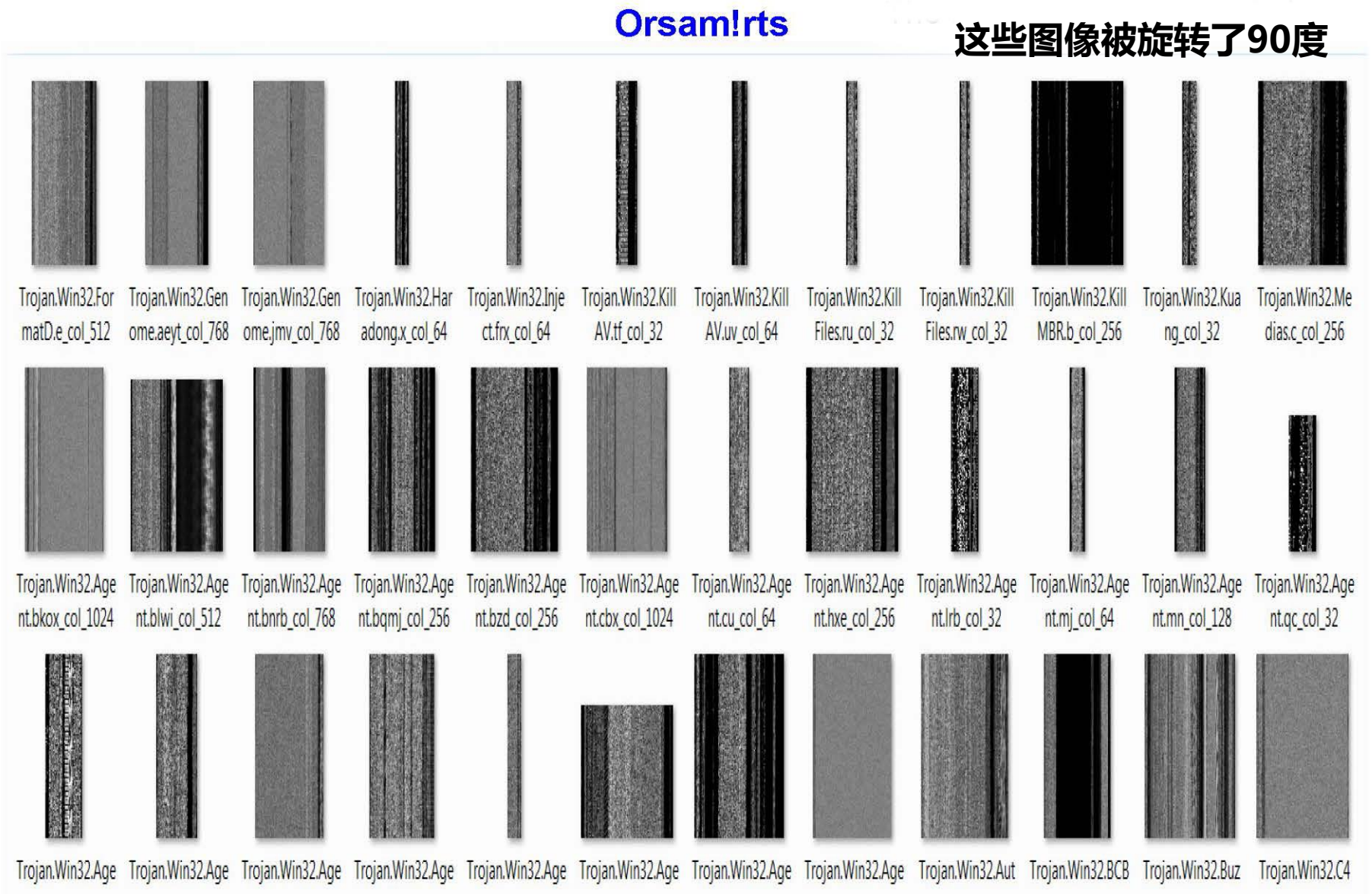
Browsermodifier.cnnicc

这些图像被旋转了90度

# 低精确度的家族

| *家族名称* | *样本数量* |
|---|---|
| Orsam!rts | 56 |
| Malex.gen!j | 215 |
| Bumat!rts | 188 |
| Backdoor.Agent | 189 |
| Pakes | 37 |
| Swizzor.gen!k | 127 |
| Poison.G | 59 |
| C2lop.O | 64 |
| Ceeinject.gen!j | 54 |
| Trufip!rts | 117 |

# 一个低精确度家族的截图

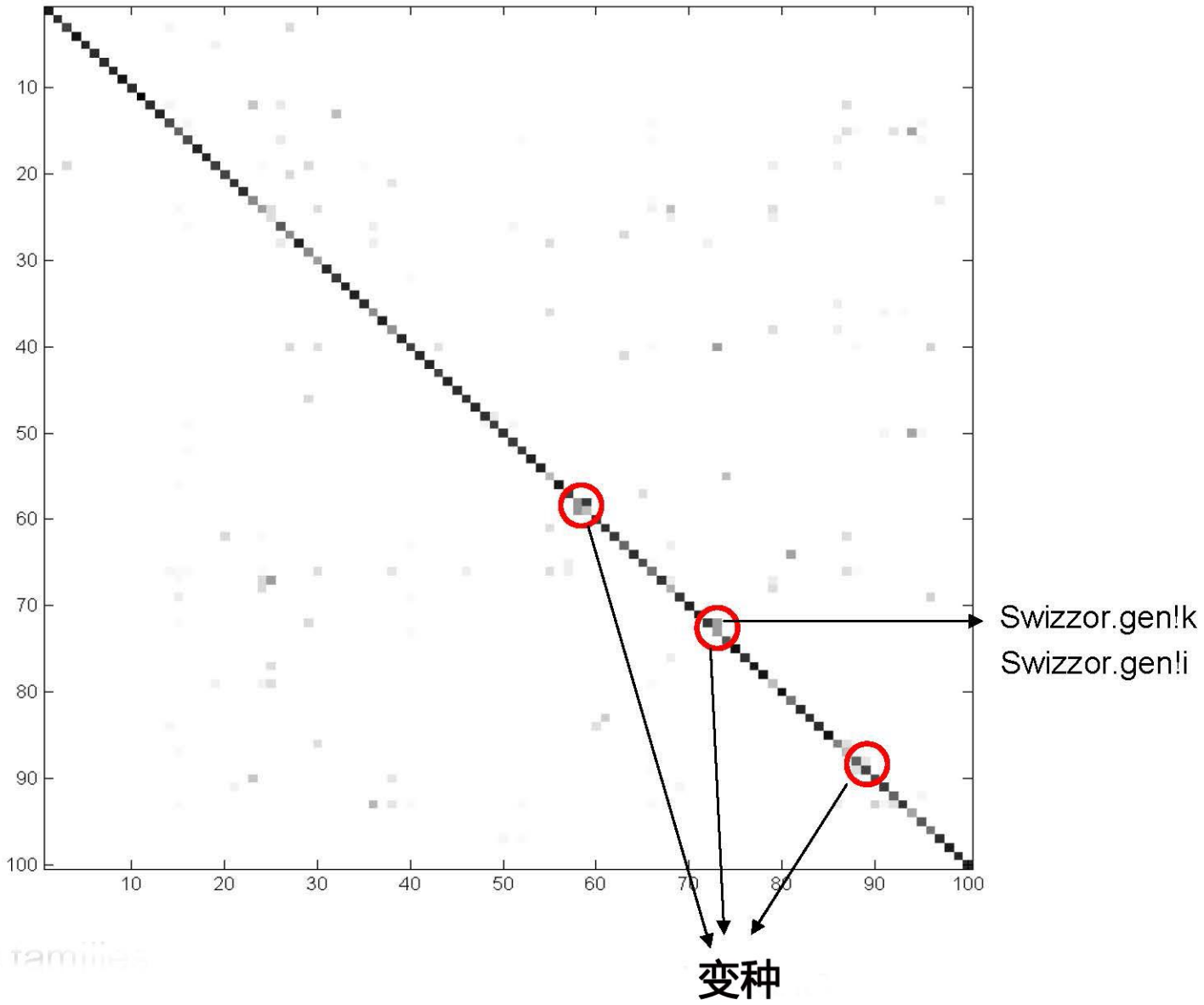



**恶意软件图像之间的差异可能是由于视频软件造成的。**

# 对 Orsam!rts的统计- 混合

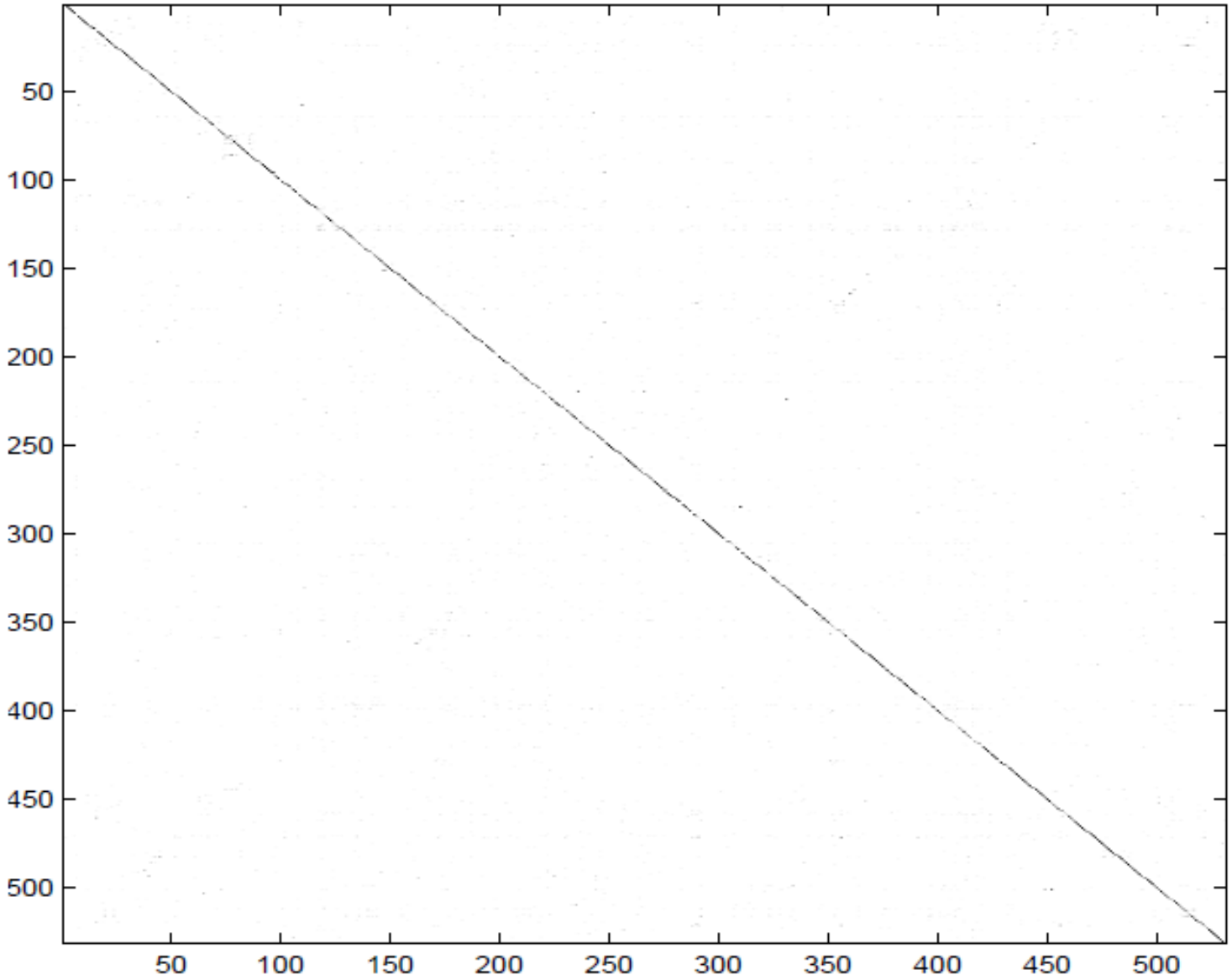
| | |
|---|---|
| 没发现 | 15 |
| Microsoft VC ++ | 13 |
| Microsoft Visual Basic | 2 |
| Borland Delphi | 8 |
| UPX | 7 |
| Themida, Aspack | 1 |
| Nspack | 2 |
| PeCompact, LCC | 1 |

# 进一步观察




k-nn = 3, 100 家族

# 64,000个 恶意软件, 531 个家族



# 基于图像分析恶意软件的优势

- 快速 (特征计算时间约等于50毫秒)
- 不用执行或者反汇编样本。
- 图像可以针对恶意软件结构提供更多信息。
- 视觉吸引力：在相似的恶意软件图像的基础上开发新的命名方案。
- 新颖：利自图像处理和机器视觉进行恶意软件分析。

# 基于图像分析恶意软件的局限性

- 数据驱动：基于现有的恶意软件进行分析。因此, 难以防御零日攻击。
- 表征: 目前，除了反病毒软件给出的标记外，恶意软件图像表征并未给出关于恶意软件实际行为的过多信息。我们也没有查找恶意软件的实际特征。

# 谢谢！