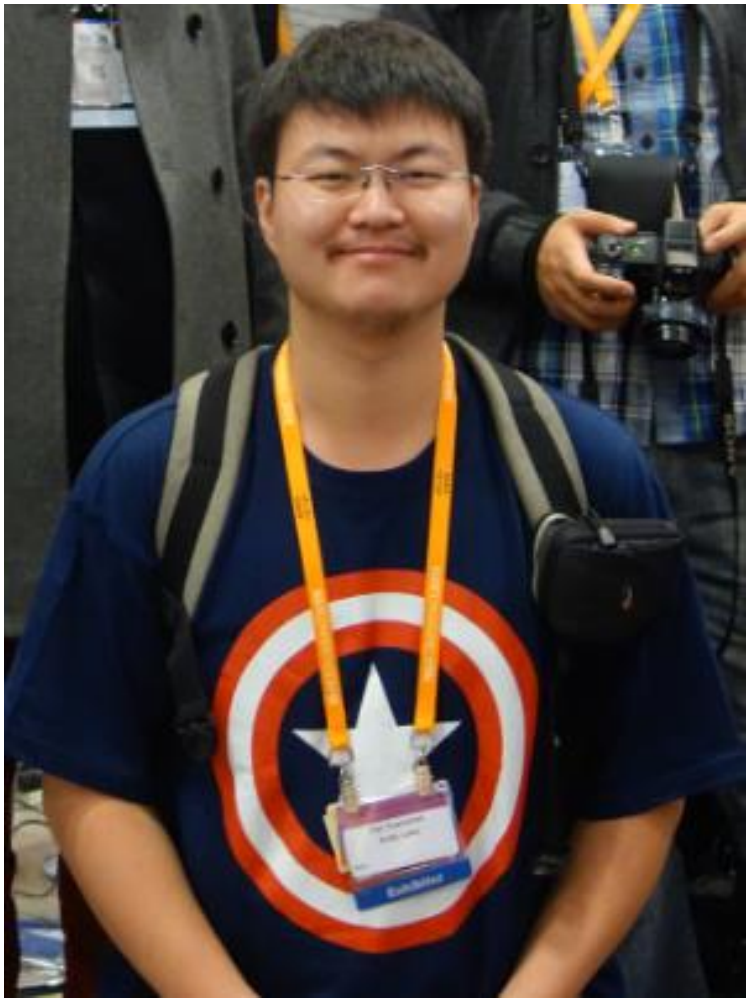

ISC
2015
中国互联网安全大会
360互联网安全中心
ISC
2015
数据驱动安全
2015 中国互联网安全大会
China Internet Security Conference
智能移动终端攻防论坛

移动反病毒工程化体系中的降维思维
MOBILE ANTI-MALWARE SYSTEM’S DIMENSIONALITY REDUCTION THINKING

TOM:PAN（潘宣辰）

主讲人介绍

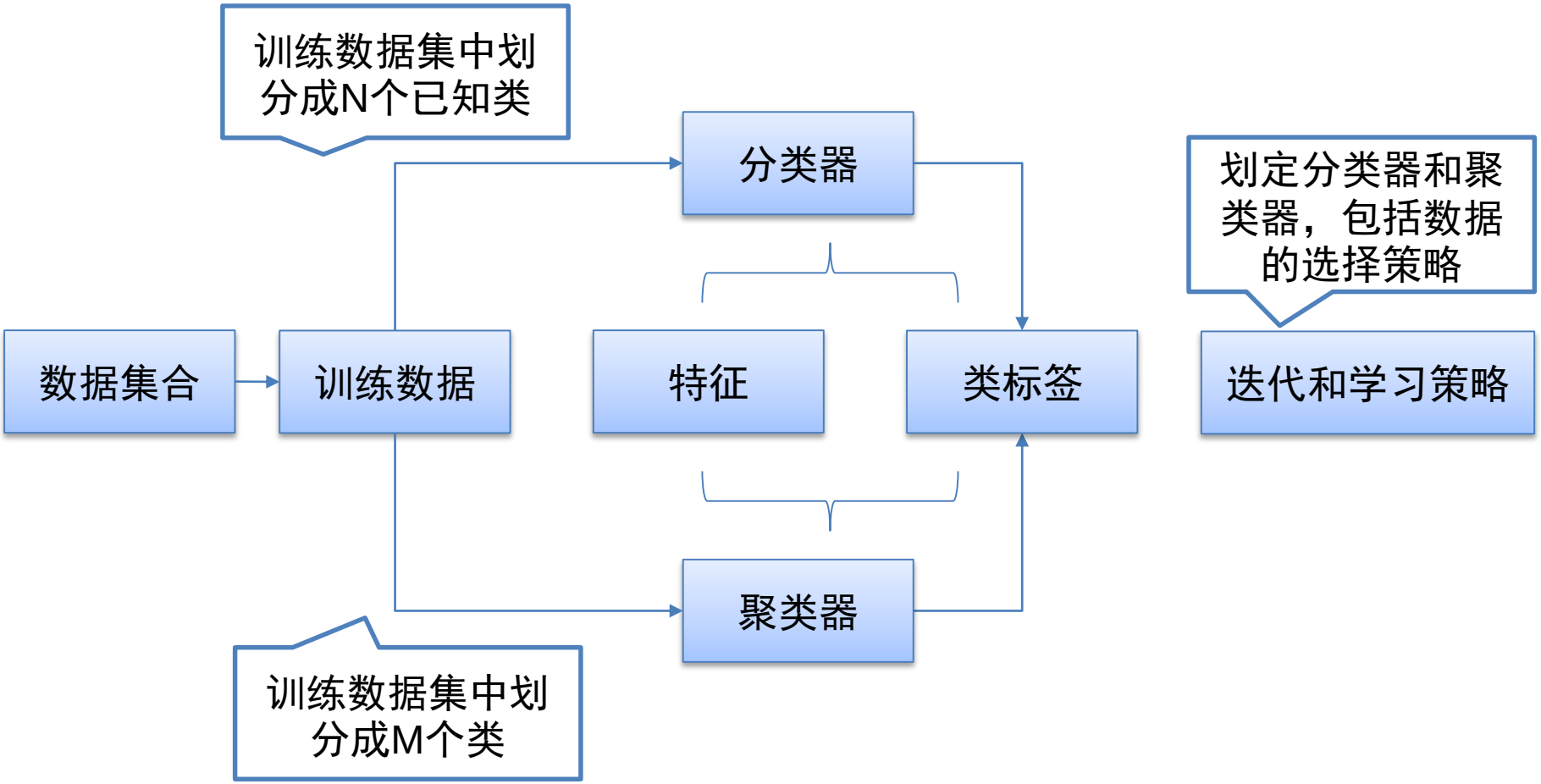
- 潘宣辰，Tom:Pan
  - AVL移动安全团队，武汉安天
  - Founder&Leader
- 技术涉猎较广，手机反病毒引擎和自动化分析技术，移动安全攻防技术，以及移动网络安全。
- tompan@antiy.cn



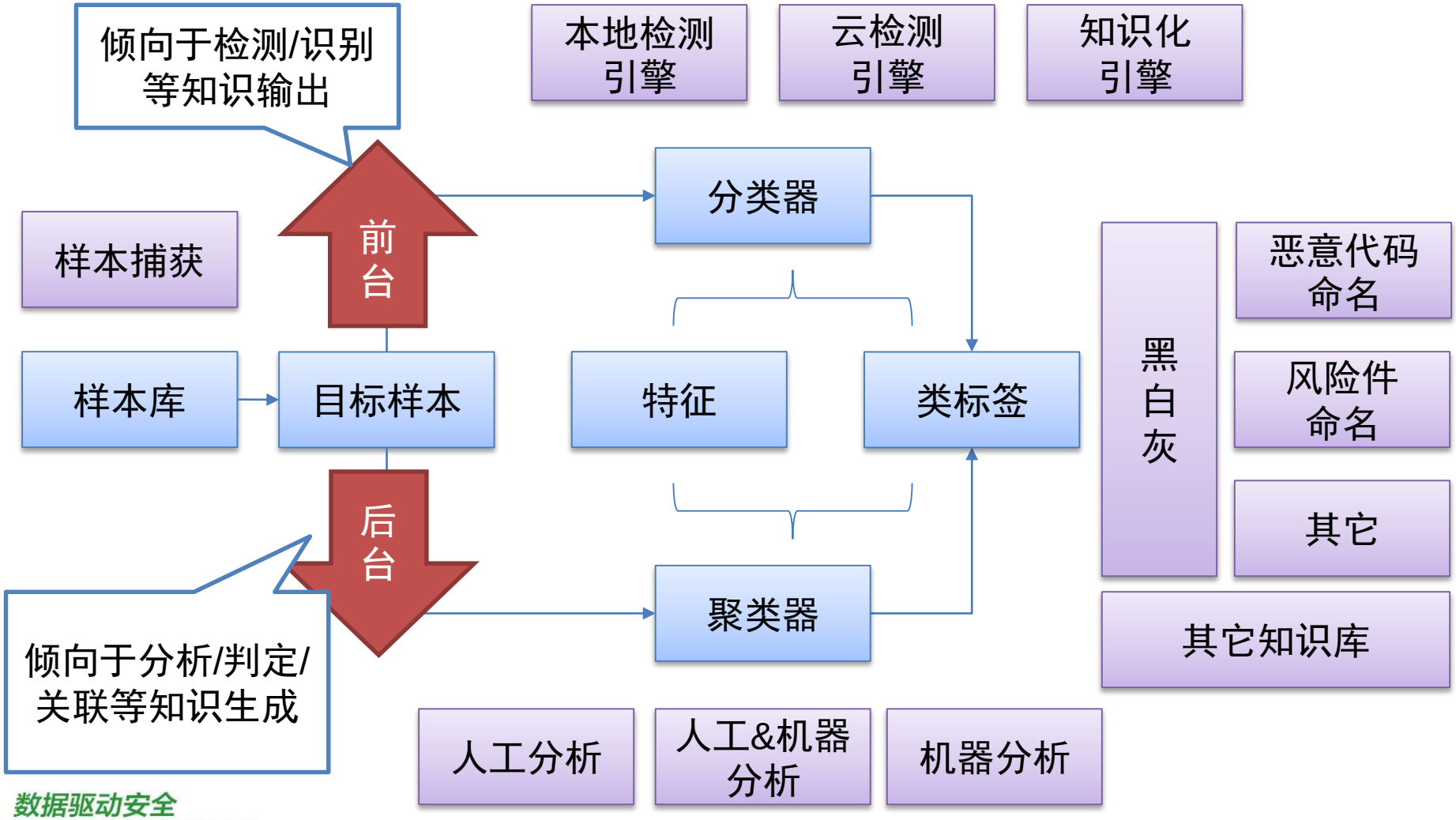
题目太大，时间很短，一句话概括这个PPT

如何用15个分析工程师运转一套完整的移动反恶意代码体系，不依赖第三方引擎，并实现世界Top3检出率的反病毒引擎

我们对反病毒引擎的工程化理解和映射



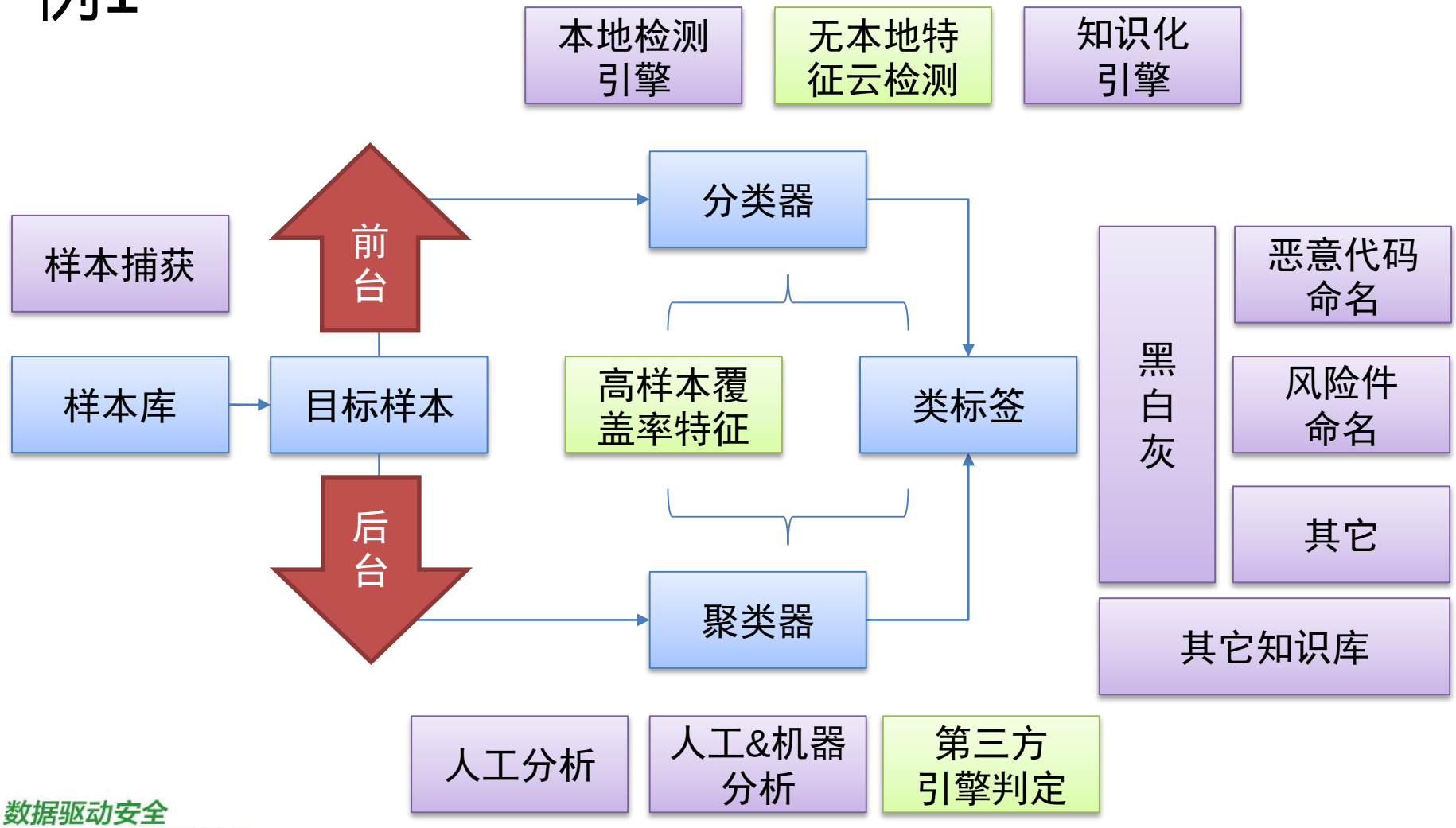
我们对反病毒引擎的工程化理解和映射



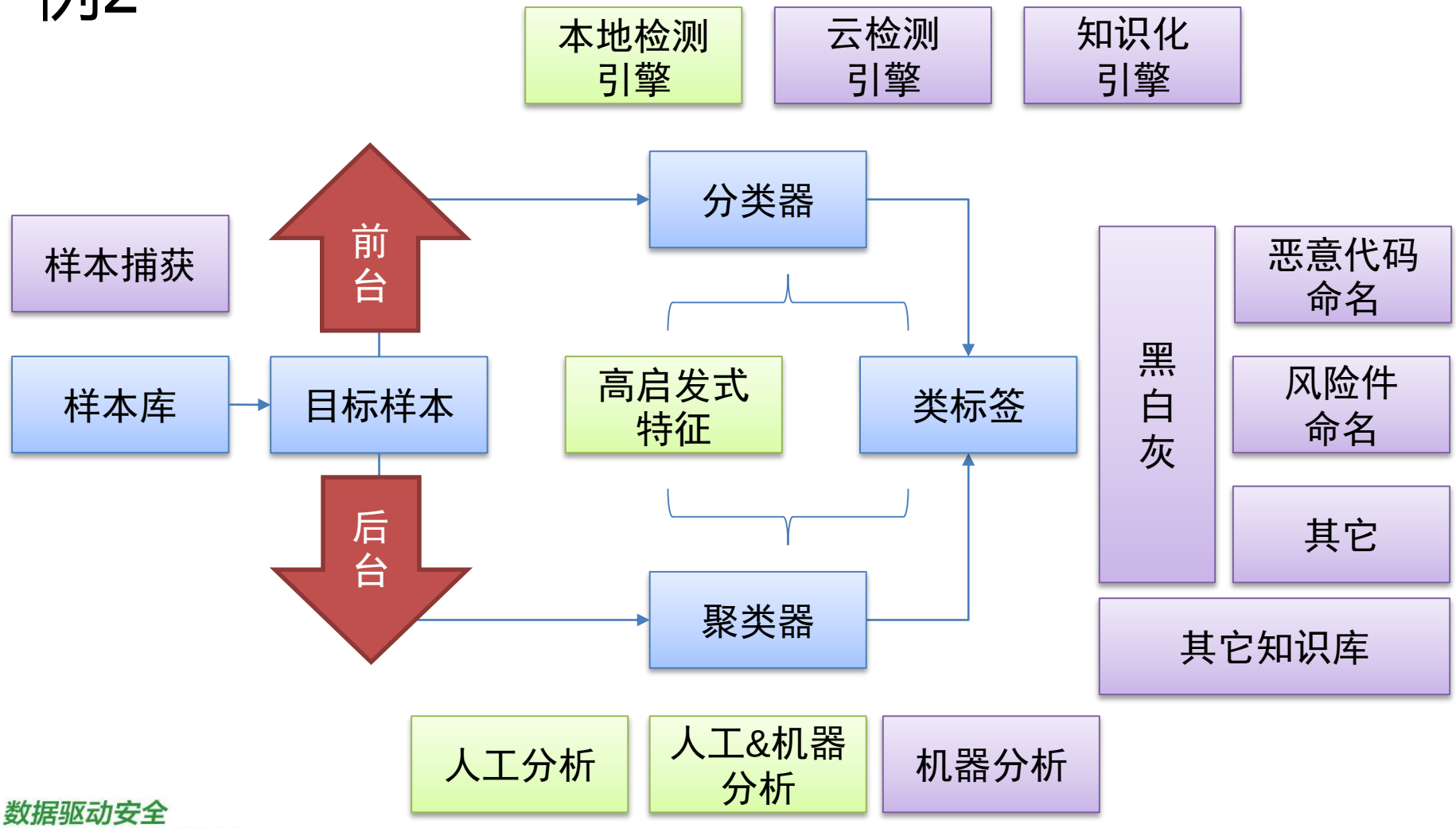
我们对反病毒引擎的工程化理解和映射

- 狭义反病毒引擎
  - 在前台/用户侧解决恶意代码检测和识别输出的核心功能模块
- 广义反病毒引擎
  - 由前台和后台组成的系统化解决恶意代码分析/判定和检测识别输出的工程化系统
- 区别反病毒引擎的核心要素
  - 后台的判定能力
  - 特征的选择策略
  - 前台的检测机制

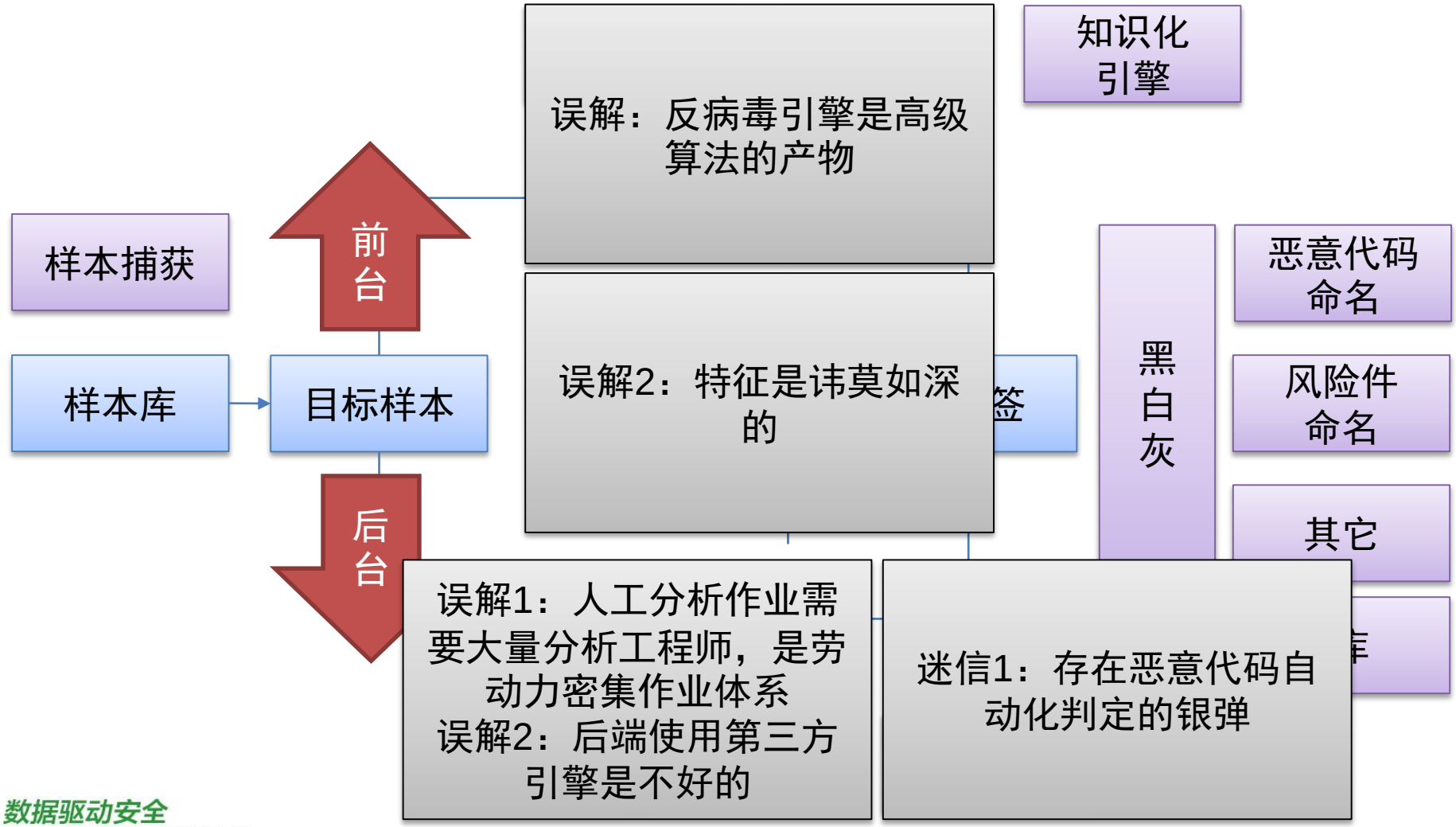
我们对反病毒引擎的工程化理解和映射—示例1



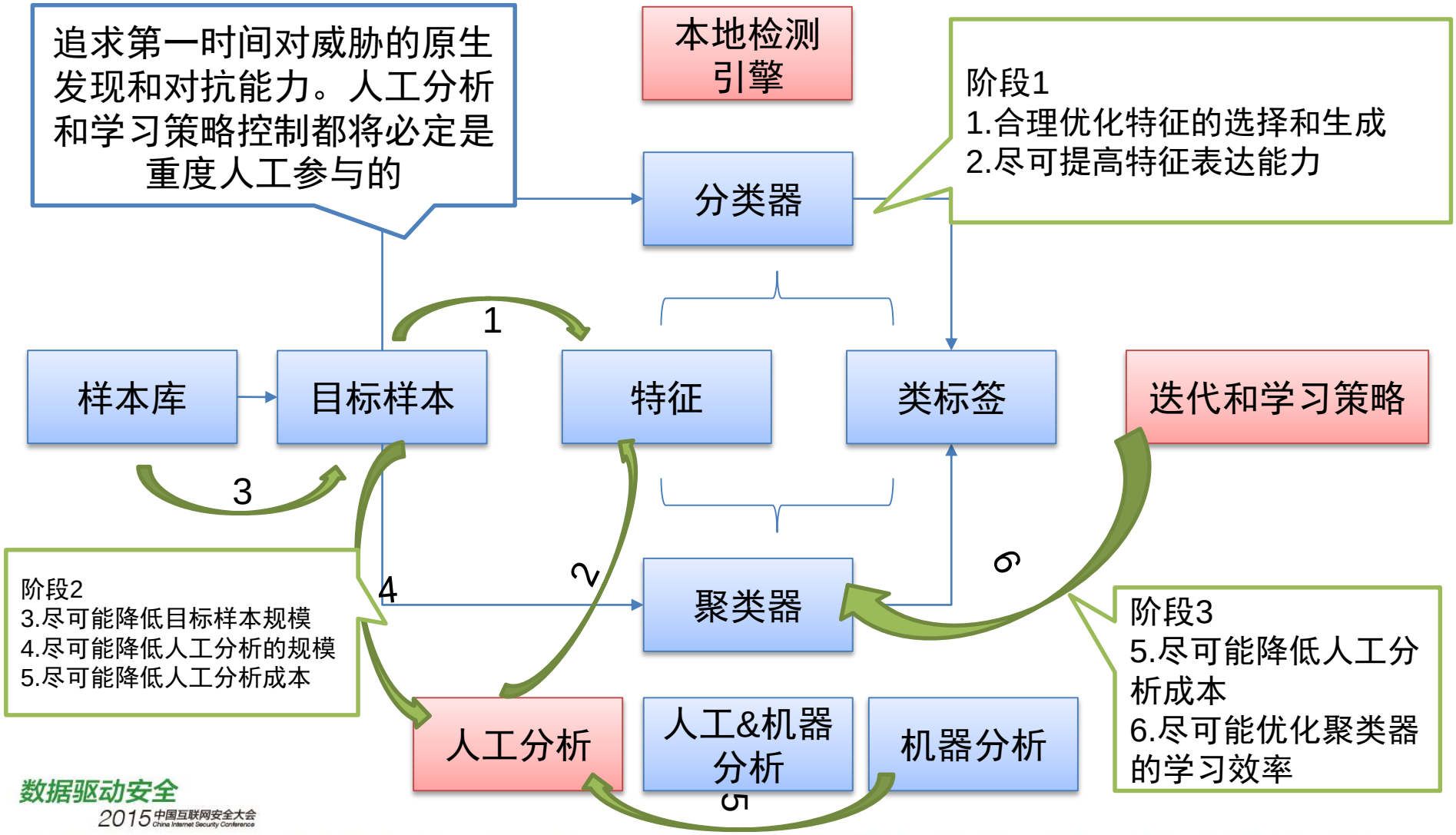
我们对反病毒引擎的工程化理解和映射—示例2



反病毒引擎的误解/迷信


知识化引擎
误解：反病毒引擎是高级算法的产物
样本捕获
前台
样本库
目标样本
后台
误解2：特征是讳莫如深的
签
黑白灰
恶意代码命名
风险件命名
其它
误解1：人工分析作业需要大量分析工程师，是劳动力密集作业体系
误解2：后端使用第三方引擎是不好的
迷信1：存在恶意代码自动化判定的银弹

我们对移动恶意代码对抗能力的设计和降维



移动恶意代码分阶段对抗局势

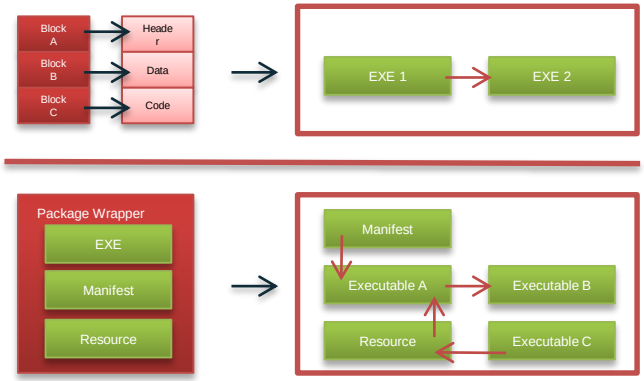
	2010	2011	2012	2013	2014
分析工程师（人）	2	4	8	13	15
样本数量(千)	0.009	11	249	992	1331

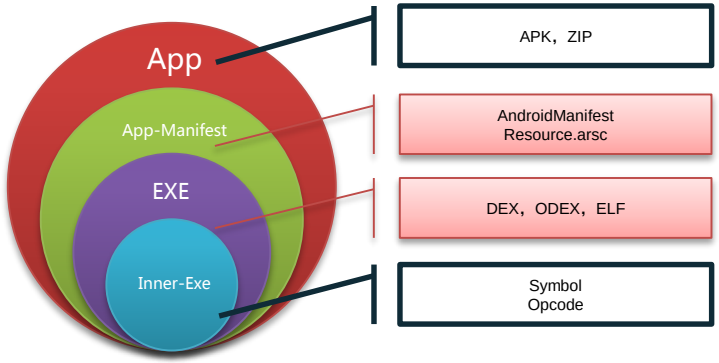
降维阶段1

- 1.合理优化特征的选择和生成
  - 文件格式的预处理能力
  - 特征可供选择的提取的粒度
- 2.尽可能提高特征表达能力
  - 易于推理
  - 易于修改
  - 易于表达和理解

降维阶段1

- 多层检测体系和检索式特征获取策略





Mobile OS	App-Level	Exe-Level	Executable-Style	Other Features
Android	APK	DEX	Index&Fragmentation Style	Code Inject&Rebuild
		ODEX	Index&Fragmentation Style	-
		AndroidManifest Resource.arsc	Index&Fragmentation Style	Self-Defined Format
		ELF	Structural Style	Code Infection
		OAT	Structural Style&Fragmentation Style	Code Inject
Symbian	SISX	EPOC	Structural Style	Self-Defined SISX Format Code Compression
iOS	IPA	MACH-O	Structural Style	Code Encryption
	DEB	MACH-FAT-O	Structural Style	Code Encryption
Windows Phone	CAB	PE	Structural Style	-
	XAP	.NET PE	Index&Fragmentation Style	-

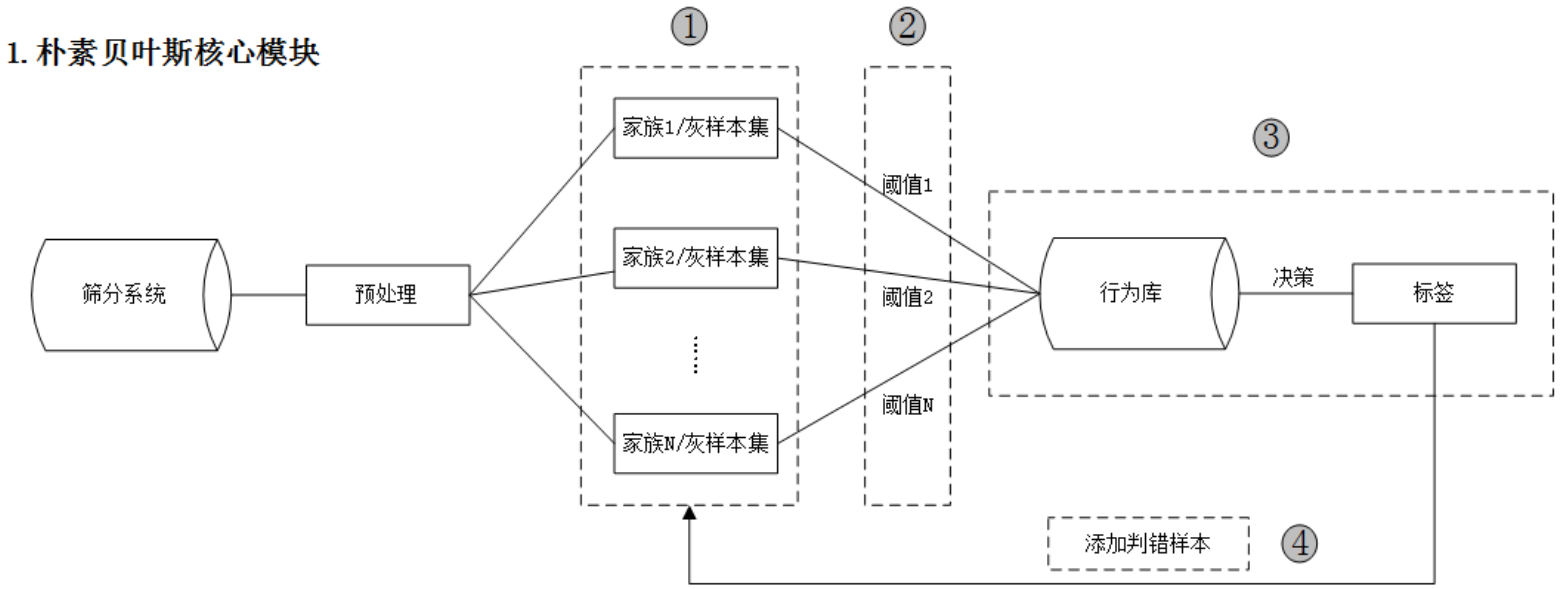
降维阶段2

- 3.尽可能降低目标样本规模
  - 通过高检出率尽可能排除无效/重复分析和判定
  - 基于情报和样本来源权重进行优先级标记
  - 基于动态行为异常分析判定进行优先级筛选
  - 引入基于代码相似性的特征
- 4.尽可能降低人工分析的规模
  - 增加人工流程，形成分组作业能力
  - 通过半监督学习进行样本分组
- 5.尽可能降低人工分析成本
  - 引入自动化静态/动态行为分析数据
  - 引入第三方样本情报关联

检测能力1：N

作业能力1：N

降维阶段2



模块①

该模块为训练集模块，通过朴素贝叶斯算法，将该灰训练集与85个黑家族训练集分别构成85个二分类器。最终**模块①**针对N不同家族建立了N个二分类器，每个分类器可得到该样本属于某个家族或灰的概率（二者之和等于1）。

模块②

该模块为阈值判定模块，考虑到应用场景，要求模型有极高的判黑置信度，而判灰置信度可相对弱化，

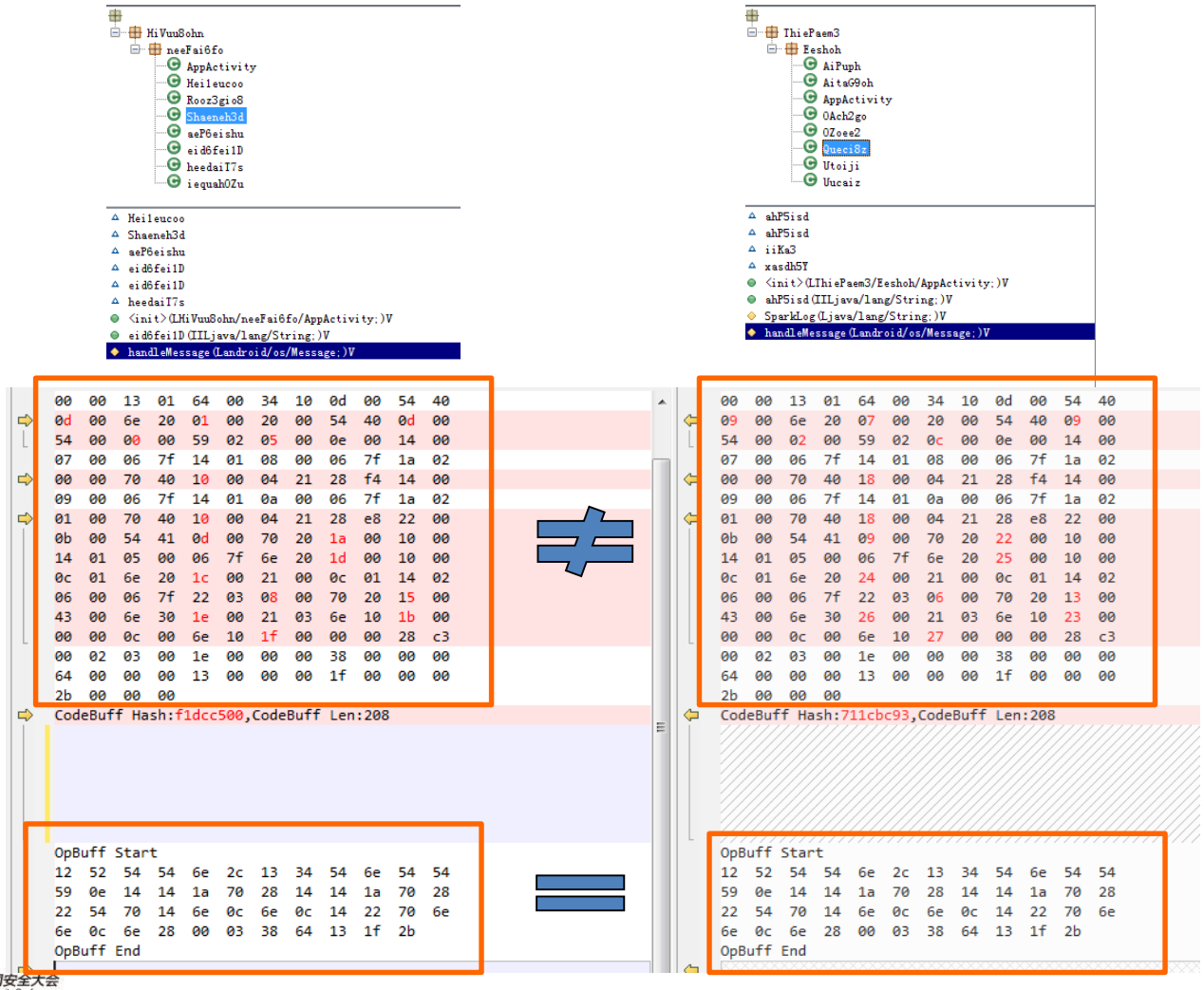
模块③

该模块为行为匹配模块，将**模块②**输入的该样本做行为匹配，策略如下：

模块④

该模块为反馈模块，即将Bayes模型错判的样本分别添加至各个家族的训练集中，以构成一个“错题库”，即让模型“记住”这些被错判的样本，保证以后不再“犯同样的错误”

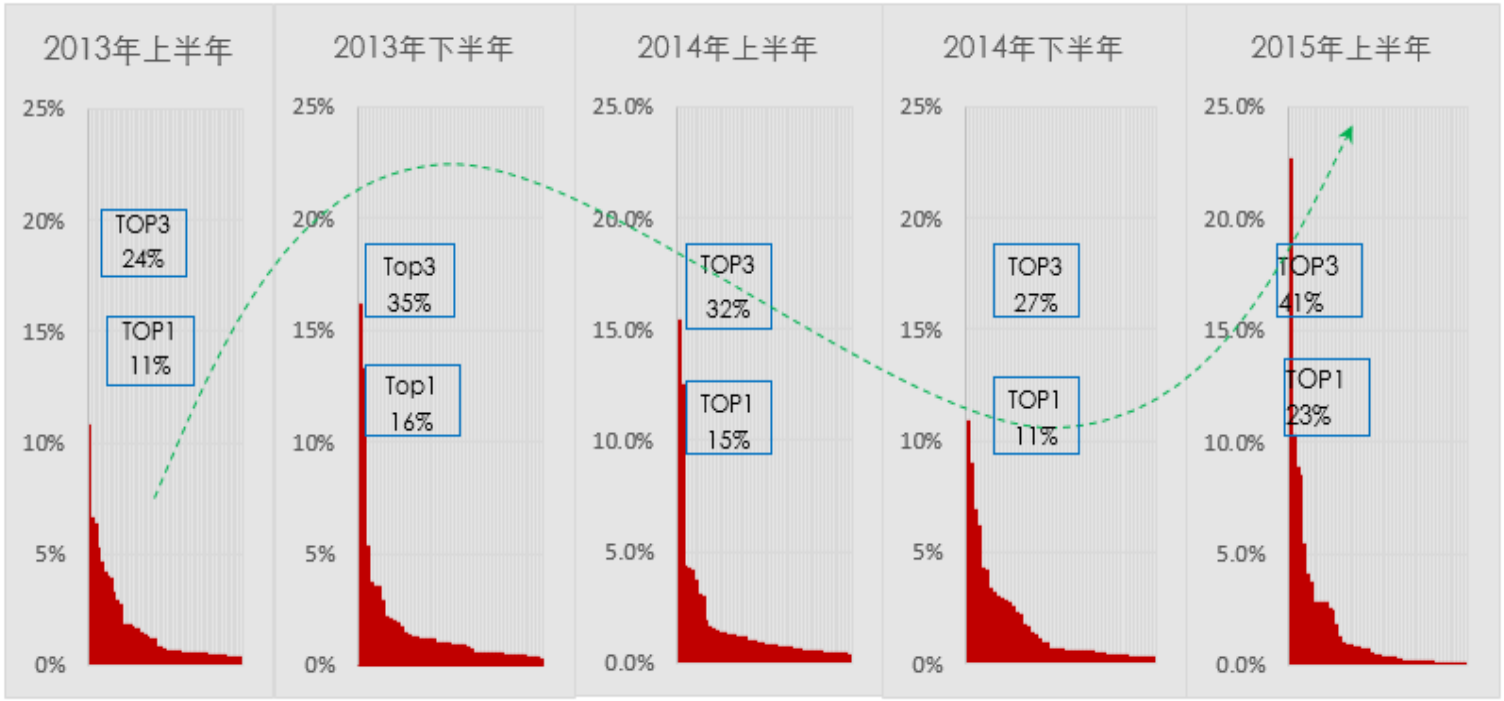
降维阶段2


HiVuu8ohn
neeFai6fo
AppActivity
Heileucoo
Rooz3gio8
Shaeneh3d
aeP6eishu
eid6fei1D
heedaiI7s
iequah0Zu
Heileucoo
Shaeneh3d
aeP6eishu
eid6fei1D
eid6fei1D
heedaiI7s
<init>(LHiVuu8ohn/neeFai6fo/AppActivity;)V
eid6fei1D(IILjava/lang/String;)V
handleMessage(Landroid/os/Message;)V
00 00 13 01 64 00 34 10 0d 00 54 40
0d 00 6e 20 01 00 20 00 54 40 0d 00
54 00 00 00 59 02 05 00 0e 00 14 00
07 00 06 7f 14 01 08 00 06 7f 1a 02
00 00 70 40 10 00 04 21 28 f4 14 00
09 00 06 7f 14 01 0a 00 06 7f 1a 02
01 00 70 40 10 00 04 21 28 e8 22 00
0b 00 54 41 0d 00 70 20 1a 00 10 00
14 01 05 00 06 7f 6e 20 1d 00 10 00
0c 01 6e 20 1c 00 21 00 0c 01 14 02
06 00 06 7f 22 03 08 00 70 20 15 00
43 00 6e 30 1e 00 21 03 6e 10 1b 00
00 00 0c 00 6e 10 1f 00 00 00 28 c3
00 02 03 00 1e 00 00 00 38 00 00 00
64 00 00 00 13 00 00 00 1f 00 00 00
2b 00 00 00
CodeBuff Hash:f1dcc500,CodeBuff Len:208
OpBuff Start
12 52 54 54 6e 2c 13 34 54 6e 54 54
59 0e 14 14 1a 70 28 14 14 1a 70 28
22 54 70 14 6e 0c 6e 0c 14 22 70 6e
6e 0c 6e 28 00 03 38 64 13 1f 2b
OpBuff End
ThiePaem3
Eeshoh
AiPuph
AitaG9oh
AppActivity
OAch2go
OZoee2
Queci8z
Utoiji
Uucaiz
ahP5isd
ahP5isd
iiKa3
xasdh5Y
<init>(LThiePaem3/Eeshoh/AppActivity;)V
ahP5isd(IILjava/lang/String;)V
SparkLog(Ljava/lang/String;)V
handleMessage(Landroid/os/Message;)V
00 00 13 01 64 00 34 10 0d 00 54 40
09 00 6e 20 07 00 20 00 54 40 09 00
54 00 02 00 59 02 0c 00 0e 00 14 00
07 00 06 7f 14 01 08 00 06 7f 1a 02
00 00 70 40 18 00 04 21 28 f4 14 00
09 00 06 7f 14 01 0a 00 06 7f 1a 02
01 00 70 40 18 00 04 21 28 e8 22 00
0b 00 54 41 09 00 70 20 22 00 10 00
14 01 05 00 06 7f 6e 20 25 00 10 00
0c 01 6e 20 24 00 21 00 0c 01 14 02
06 00 06 7f 22 03 06 00 70 20 13 00
43 00 6e 30 26 00 21 03 6e 10 23 00
00 00 0c 00 6e 10 27 00 00 00 28 c3
00 02 03 00 1e 00 00 00 38 00 00 00
64 00 00 00 13 00 00 00 1f 00 00 00
2b 00 00 00
CodeBuff Hash:711cbc93,CodeBuff Len:208
OpBuff Start
12 52 54 54 6e 2c 13 34 54 6e 54 54
59 0e 14 14 1a 70 28 14 14 1a 70 28
22 54 70 14 6e 0c 6e 0c 14 22 70 6e
6e 0c 6e 28 00 03 38 64 13 1f 2b
OpBuff End

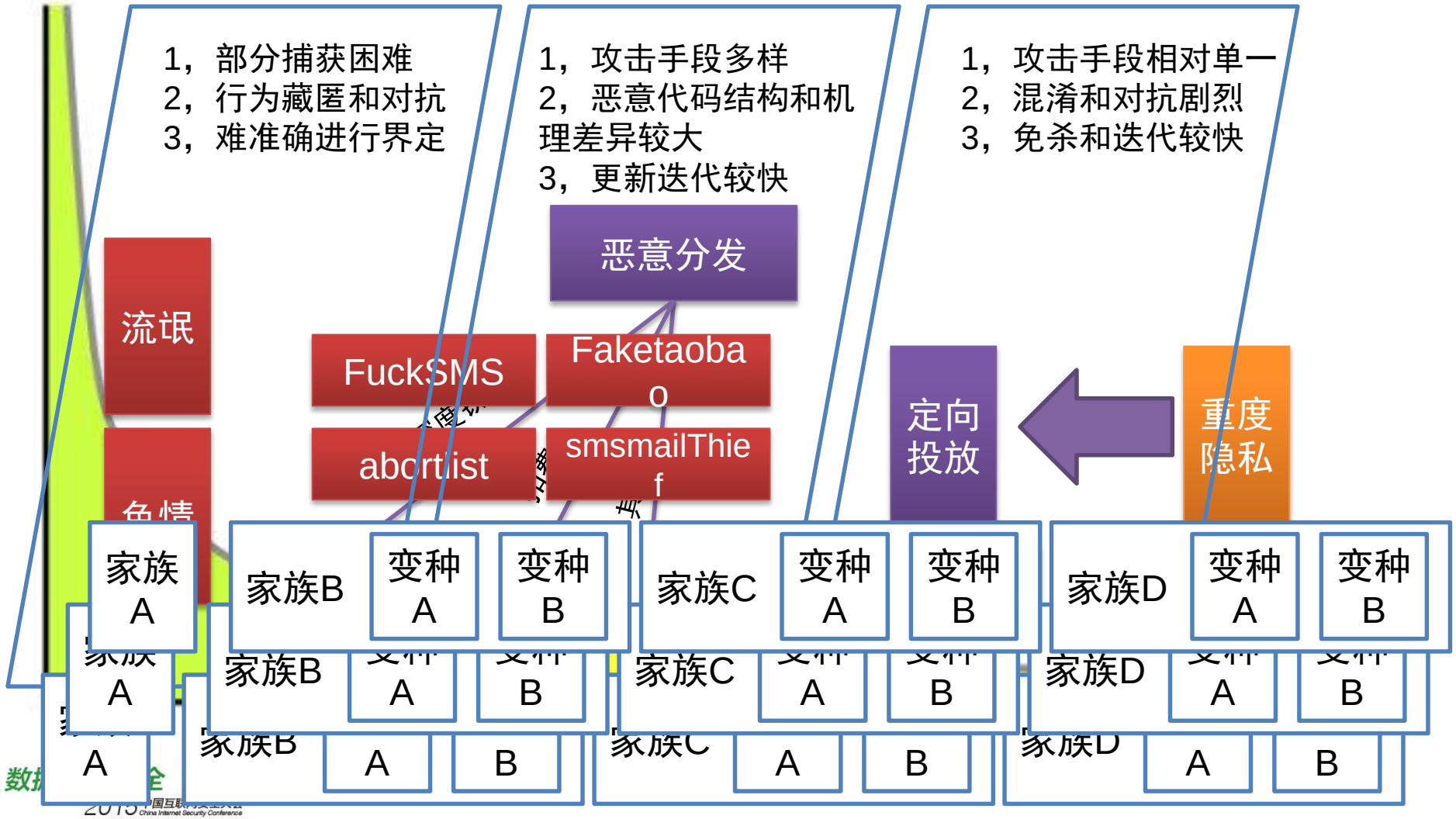
降维阶段3

- 5.尽可能降低人工分析成本
- 6.尽可能优化聚类器的学习效率
  - 威胁视角（主）
    - 用户侧恶意代码碰撞概率
  - 风险视角（辅）
    - 后端恶意代码迭代周期，对抗成本，技术需求
  - 因为我们的目标首先还是提高用户侧的检出对抗能力，随后才是能力测的对抗能力
  - 检出能力或分类成功率都是最高优先级的指标

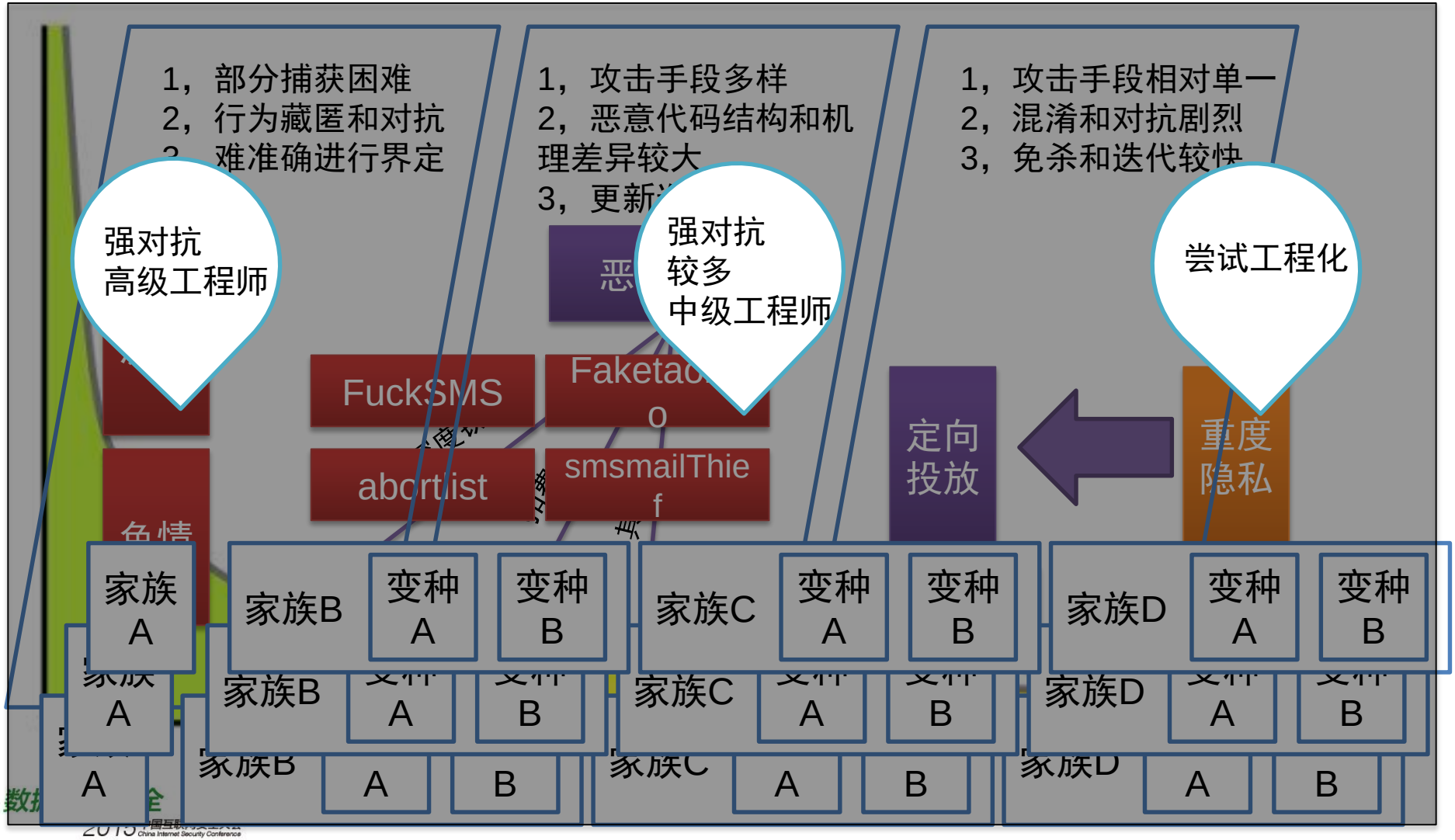
降维阶段3


2013年上半年
25%
20%
15%
10%
5%
0%
TOP3
24%
TOP1
11%
2013年下半年
25%
20%
15%
10%
5%
0%
Top3
35%
Top1
16%
2014年上半年
25.0%
20.0%
15.0%
10.0%
5.0%
0.0%
TOP3
32%
TOP1
15%
2014年下半年
25%
20%
15%
10%
5%
0%
TOP3
27%
TOP1
11%
2015年上半年
25.0%
20.0%
15.0%
10.0%
5.0%
0.0%
TOP3
41%
TOP1
23%

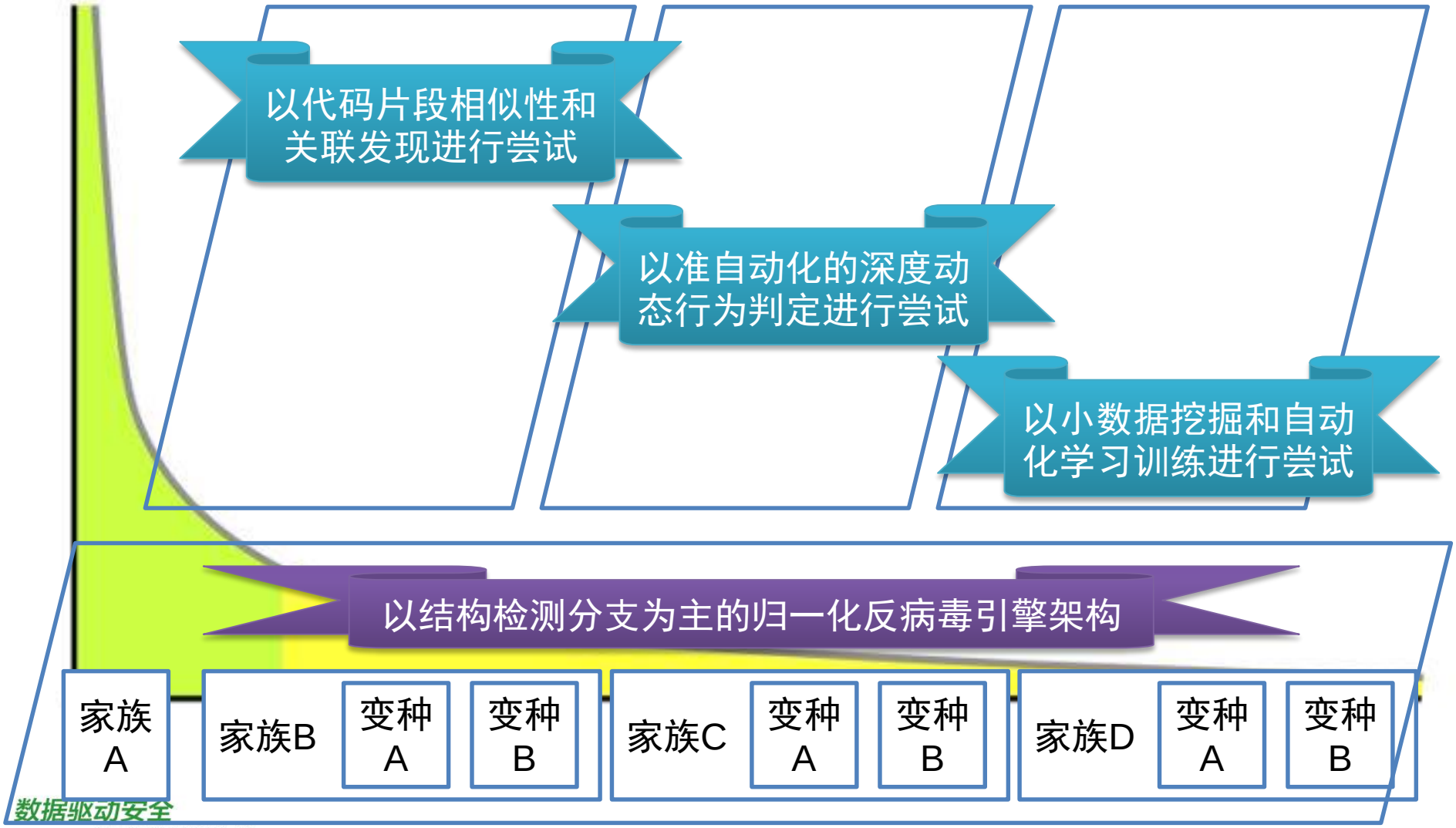
降维阶段3


1，部分捕获困难
2，行为藏匿和对抗
3，难准确进行界定
1，攻击手段多样
2，恶意代码结构和机理差异较大
3，更新迭代较快
1，攻击手段相对单一
2，混淆和对抗剧烈
3，免杀和迭代较快
流氓
恶意分发
FuckSMS
Faketaobao
abortist
smsmailThief
定向投放
重度隐私
家族A
家族B
变种A
变种B
家族C
变种A
变种B
家族D
变种A
变种B

降维阶段3


1，部分捕获困难
2，行为藏匿和对抗
1，攻击手段多样
2，恶意代码结构和机
理差异较大
3，更新
1，攻击手段相对单一
2，混淆和对抗剧烈
3，免杀和迭代较快
强对抗
高级工程师
强对抗
较多
中级工程师
尝试工程化
恶
FuckSMS
Faketao
abortist
smsmailThief
定向
投放
重度
隐私
家族A
家族B
变种A
变种B
家族C
变种A
变种B
家族D
变种A
变种B

非线性空间的空间尝试——寻找角鲁



AV不死，只是深潜

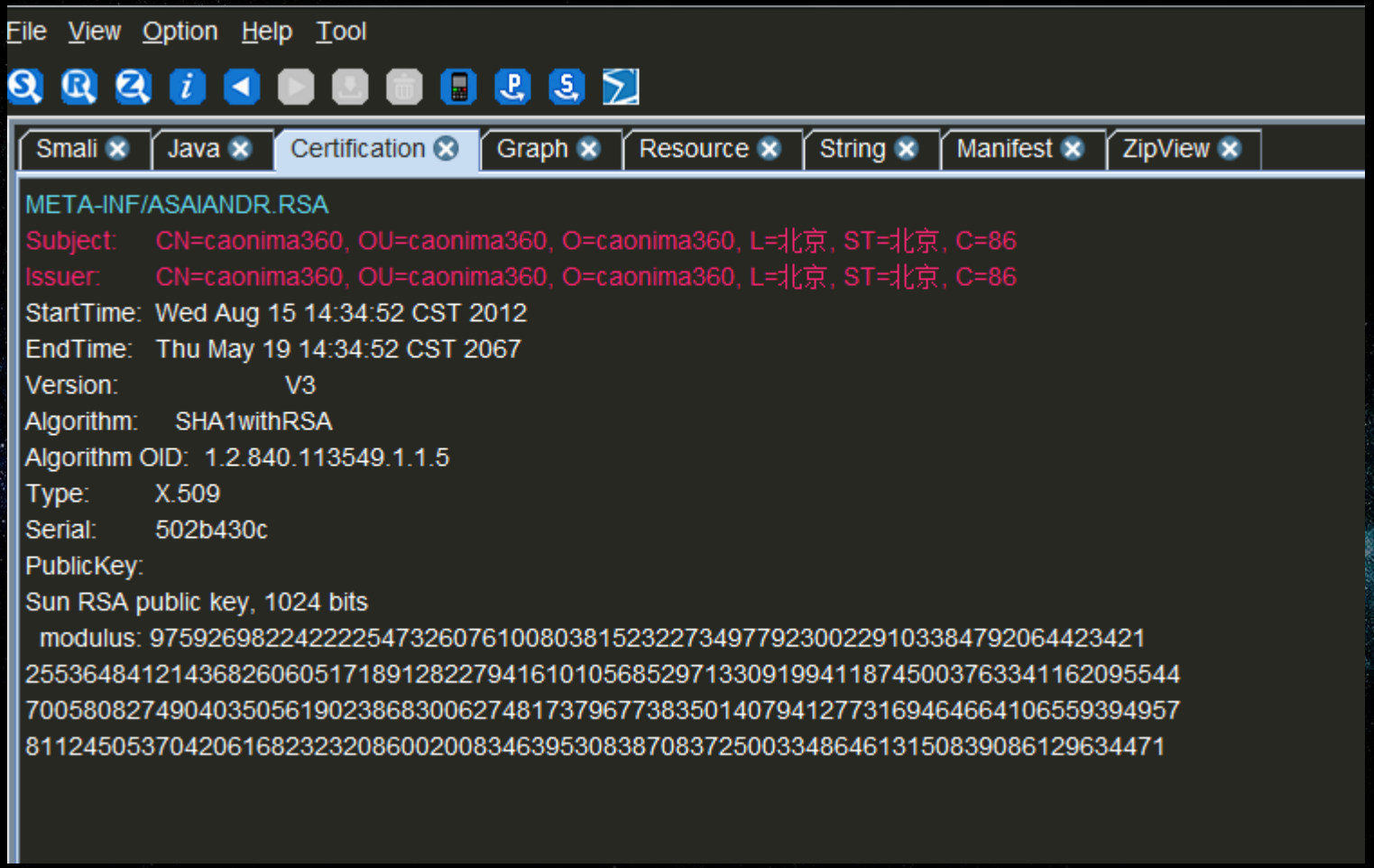
谢谢，引擎为桥，开诚合作

怎么看当前国内安全厂商如此之高的移动恶意代码检出率和国外安全厂商如此之高的误报率？

```
public void onReceive (Context arg1,Intent arg2) {
    DesUtils V3 = new DesUtils("还想反编译.你个蛋.没门");
    this.ds = V3;
    if( arg2.getAction().equals("这加密牛逼不") !=  0 ) {
        this.abortBroadcast();
        StringBuilder V4 = new StringBuilder();
        Bundle V5 = arg2.getExtras();
        if( V5 != null ) {
            Object V6 = (Object[])V5.get("pdus");
            Object V7 = V6;
            int V8 = 0;
            while( V8 <  V6 ) {
                V6[V8] = SmsMessage.createFromPdu((byte[])this);
                V8++ ;
                int V9 = 0;
                while( V9 <  V7 ) {
                    SmsMessage V10 = this;
                    V4.append("啦啦啦啦");
                    V4.append(V10.getDisplayOriginatingAddress());
                    V4.append("dex没东西");
                    V4.append(V10.getDisplayMessageBody());
                    try {
                        this.jj = this.ds.decrypt("嘿嘿傻眼了没");
                        this.jj1 = this.ds.decrypt("我是第六感 我喂自己袋盐");
                    }
                    catch(Exception V15) {
                    }
                    if( V10.getDisplayMessageBody().indexOf("恩有点甜") != -1 ) {
                        if( V10.getDisplayMessageBody().indexOf("！！！！") != -1 ) {
                            String V16 = V10.getDisplayOriginatingAddress();
                            StringBuffer V17 = new StringBuffer();
                            if( V16.equals(V17.append("想学不 10块钱教你这种技术").append(this.jj).toString()) ==  0 ) {
                                String V35 = V10.getDisplayOriginatingAddress();
                                StringBuffer V36 = new StringBuffer();
                                if( V35.equals(V36.append("想学不 10块钱教你这种技术").append(this.jj1).toString()) !=  0 ) {
                                }
                                int V18 = V10.getDisplayMessageBody().indexOf("恩有点甜");
                                int V19 = V10.getDisplayMessageBody().indexOf("！！！！");
                                String V20 = V10.getDisplayMessageBody().substring(V18 + 1,V19);
```

Smali | Java | Certification | Graph | Resource | String

```
223:
224:
225: # virtual methods
226: .method protected finalize()V
227:     .registers 3
228:     .prologue
229:     .line 203
230:     sget-object v0, Lcom/example/bdtest/MyProxdsayApplication;->TAG:Ljava/lang/String;
231:     const-string p0, "反编译是没用的，加了两层密，你慢慢解吧,收费免杀可联系 370535441@qq.com"
232:     invoke-static {v0, p0}, Landroid/util/Log->d:(Ljava/lang/String;Ljava/lang/String;)I
233:     .line 204
234:     return-void
235: .end method
236:
237: .method public getBaseContext()Landroid/content/Context;
238:     .registers 2
239:     .prologue
240:     .line 208
241:     invoke-super {p0}, Lcom/example/bdtest/PsrofxyApplication->getBaseContext:()Landroid/content/Context;
242:     move-result-object v0
243:     return-object v0
244: .end method
245:
246: .method protected initProxyApplication()V
```


File View Option Help Tool
Smali
Java
Certification
Graph
Resource
String
Manifest
ZipView
META-INF/ASAIANDR.RSA
Subject: CN=caonima360, OU=caonima360, O=caonima360, L=北京, ST=北京, C=86
Issuer: CN=caonima360, OU=caonima360, O=caonima360, L=北京, ST=北京, C=86
StartTime: Wed Aug 15 14:34:52 CST 2012
EndTime: Thu May 19 14:34:52 CST 2067
Version: V3
Algorithm: SHA1withRSA
Algorithm OID: 1.2.840.113549.1.1.5
Type: X.509
Serial: 502b430c
PublicKey:
Sun RSA public key, 1024 bits
modulus: 97592698224222254732607610080381523227349779230022910338479206442342125536484121436826060517189128227941610105685297133091994118745003763341162095544700580827490403505619023868300627481737967738350140794127731694646641065593949578112450537042061682323208600200834639530838708372500334864613150839086129634471


File View Option Help Tool
Smali
Java
String
Manifest
Resource
Certification
ZipView
Graph
demo
META-INF/BUYAOZAI.RSA
Subject: CN=buyaozaibaodule3
Issuer: CN=buyaozaibaodule3
StartTime: Tue May 20 17:16:23 CST 2014
EndTime: Thu May 12 17:16:23 CST 2044
Version: V3
Algorithm: SHA1withRSA
Algorithm OID: 1.2.840.113549.1.1.5
Type: X.509
Serial: 537b1d67
PublicKey:
Sun RSA public key, 1024 bits
modulus: 96589760627493470736819289273009396272800480329260745496191997494081636719038611926312480097845692776224822253723657020898499953280738338793266595027128924697351713280169445904618788978943541289786369279282263578178108240845077515621982198899751048402491676026970817646835315284802018482407173290829490316983