文章编号：1007-5321(2017)增-0098-05　　DOI:10.13190/j.jbupt.2017.s.022

面向大规模嵌入式设备固件的自动化分析方法

王猛涛[1,2]，刘中金[3]，常　青[1,2]，陈　昱[1,2]，石志强[1,2]，孙利民[1,2]

(1. 中国科学院 信息工程研究所，北京 100093；2. 中国科学院大学 网络空间安全学院，北京 100049；
3. 国家计算机网络应急技术处理协调中心，北京 100029)

摘要：设计了一种面向大规模嵌入式设备固件的自动化分析方法，该方法能够对固件进行自动化分析，提取其文件系统、操作系统、中央处理器指令架构等关键信息.针对固件解码成功的自动化判定难题，提出了一种基于分类回归树的固件解码状态检测算法，并选取收集的6 160 个固件和固件自动化解码后得到的1 823 个可反汇编二进制文件作为样本进行实验.实验结果表明，该算法相对其他分类器具有更好的分类效果，其分类准确率、召回率均在96% 以上.

关　键　词：嵌入式设备固件；分类回归树；状态检测
中图分类号：TN911.22　　**文献标志码**：A

An Automated Analysis Method for Large-Scale Embedded Device Firmware

WANG Meng-tao[1,2]，LIU Zhong-jin[3]，CHANG Qing[1,2]，CHEN Yu[1,2]，
SHI Zhi-qiang[1,2]，SUN Li-min[1,2]

(1. Institute of Information Engineering，Chinese Academy of Sciences，Beijing 100093，China；
2. School of Cyber Secwrity，University of Chinese Academy of Sciences，Beijing 100049，China；
3. National Computer Network Emergency Response Technical Team/Coordination Center of China，Beijing 100029，China)

Abstract：An automated analysis method for large-scale embedded firmware was designed to get device information，such as file system type，operating system type，or CPU instruction set. But it was difficult to know whether it was decoded successfully during automated firmware analysis. To solve this problem，a firmware decoding status detection method was proposed based on classification and regression tree algorithm. The dataset contained 6 160 firmware samples and 1 823 disassembled binary files that were collected from firmware decoding. The experiments conducted on the dataset demonstrated that the proposed method had a considerable performance comparing with other classifiers，whose precision and recall rate are both above 96%.

Key words：embedded device firmware；classification and regression tree；status detection

随着物联网的兴起，工业 4.0 的稳步推进，嵌入式设备的网络化程度越来越高.然而，由于传统嵌入式厂商普遍缺乏安全意识，设备生产以可用性为前提，导致的安全问题越来越多[1].因此，对嵌入式设备的研究与分析也就变得越来越重要.

收稿日期：2016-05-29
基金项目：国家自然科学基金项目(U1636120)；国家重点研发计划项目(2016YFB0800202)；中国科学院国防科技创新基金项目面上基金项目(CXJJ-16M118)；工信部重点科研项目(JCKY2016602B001)；北京市科委重点课题(Z161100002616032)
作者简介：王猛涛(1989—)，男，硕士生，E-mail:wangmengtao@iie.ac.cn；石志强(1970—)，男，博士，正研级高级工程师.

由于固件包含维持设备正常运行所必须的文件系统和通信环境，因此，现阶段国际上对嵌入式设备的研究与分析[2,3]多转换为对设备固件进行研究与分析. 设计了一种面向大规模嵌入式设备固件的自动化分析方法，该方法能够对固件进行自动化逆向解码分析，提取其文件系统、操作系统、CPU 指令架构等信息.

由于不同厂商、类别和型号的固件结构、设计模式不一，导致适用于某设备固件的解码方案并不一定会适用于其他固件，这就给固件的大规模、批量自动化分析带来了困难. 针对此问题，提出一种基于分类回归树(CART，classification and regression tree)决策树[4]的固件解码状态检测算法. 该算法通过对固件熵谱信号指纹特征进行分析，实现对固件解码状态的检测，一定程度上提升了大规模固件自动化分析的效率，并通过具体的实验对该算法的性能和效果进行了验证.

1　嵌入式设备固件自动化分析方法

设计了一种面向大规模嵌入式设备固件的自动化分析方法，方法框架如图 1 所示，包括固件收集、固件解码及固件信息提取 3 个模块.

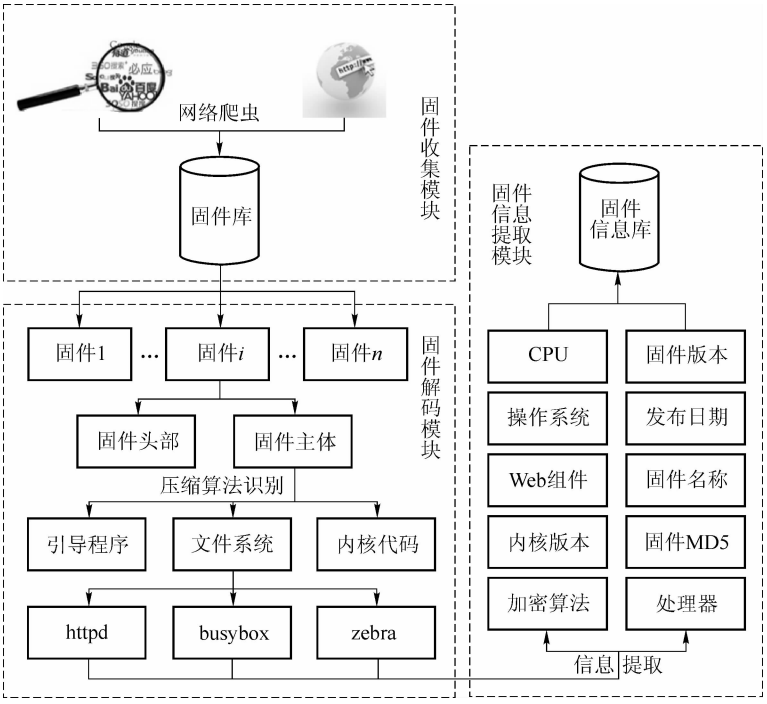


图 1　面向大规模固件的自动化分析框架结构示意图

固件收集模块是采用网络爬虫的方法(具体是采用 Python 的 Scrapy 爬虫框架)爬取、收集公网上可下载的嵌入式设备厂商固件.

固件解码模块是利用 Binwalk 对固件收集模块中所收集的固件进行检索、识别，分离其头部和主体部分；然后针对主体部分识别其压缩算法，得到其引导程序、文件系统和内核代码；再对文件系统进行过滤，获取可反汇编的二进制文件.

固件解码成功(也称该固件可解码)是指在对固件实施解码操作后，得到的文件系统中含有可反汇编二进制文件(含无文件系统固件解码后得到可反汇编二进制文件的情况). 固件解码失败是指该固件不能执行解码操作(含未执行解码操作)或执行解码操作后没有得到可反汇编二进制文件.

固件信息提取模块是对固件解码模块处理后得到的可反汇编二进制文件进行信息识别、提取，获得固件文件系统、操作系统、CPU 指令架构等信息，并将得到的信息存储到固件信息库.

数据存储使用 MongoDB 非关系型数据库进行分布式存储. 作为面向文档的数据库，MongoDB 与关系型数据库之间有着显著的区别，它采用类似

json 的 bjson 格式,因此可以存储比较复杂的数据类型;模式自由,意味着对于存储在 MongoDB 数据库中的文件,不需要知道它的任何结构定义,可以把不同结构的文件存储在同一个数据库中.

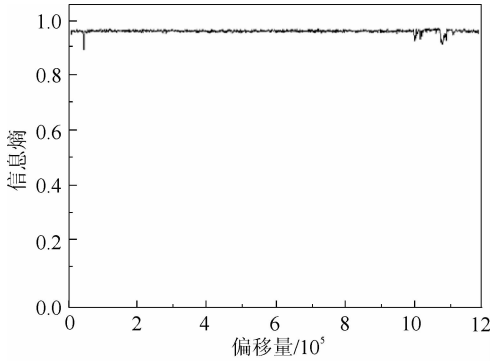
2 固件解码状态检测算法

2.1 问题描述

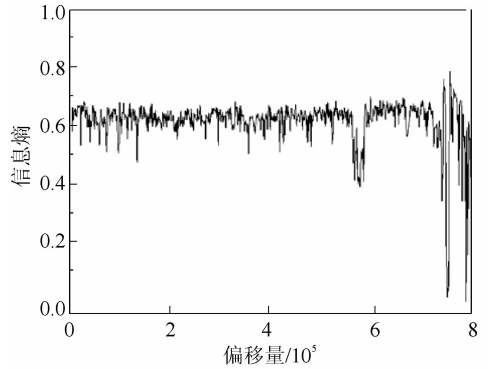
设备固件解码前后其熵谱信号(具体信号图谱可通过 binwalk-E 指令来获取)的变化趋势如图 2 所示. 若假定固件解码前的状态为 S,固件解码后的状态为 S_d,则有

$$\left.\begin{aligned} S = S_d, &\quad \text{固件解码失败} \\ S \neq S_d, &\quad \text{固件解码成功} \end{aligned}\right\} \tag{1}$$

由图 2 可见,固件解码前后其熵谱信号发生了"显著的"变化,在固件解码成功后,其熵谱信号出现了"明显的"波动;而固件解码前其熵谱信号的"波动趋势"则较为稳定. 因此,可通过提取固件解码前后的熵谱信号特征来表征固件的解码状态,并以此区分解码前后的文件,即可将对固件解码的状态检测问题转换为固件解码前后文件的分类问题.



(a) 固件解码前熵谱信号



(b) 固件解码成功后熵谱信号

图 2 嵌入式设备固件成功解码前后其熵谱信号变化

2.2 算法步骤

1) 样本数据采集

如图 2 所示,横坐标代表某文件的偏移量 ϕ,若文件大小为 M(单位:KB),则 $\phi_{\max} = M$.

在进行具体分析时,仅需要关注熵谱信号"波动"情况,提取能够表征固件熵谱信号"波动"趋势的特征;显然,这里只需要采集固件解码前后文件的熵谱信号曲线上的数据点,将其作为分类器的输入,即可通过分类器完成对应解码状态的预测.

为进一步提升分类器训练的效率,在对熵谱信号进行数据采样时,仅需采集"定量"的数据点,即可完成对熵谱信号的描述.

定义采集偏移间隔 δ 为

$$\delta = |M/N| \tag{2}$$

其中:M 为对应的文件大小,N 为采集的数据点数量,对于不同大小的文件,其采样偏移间隔 δ 是不断变化的. 这里采集相同数量的数据点是为保证所采集样本数据集的维度一致.

2) 特征选择

对于样本数据集 $X = \{x_1, x_2, \cdots, x_i, \cdots, x_N\}$,其中 $1 \leqslant i \leqslant N$,$N$ 为采集数据点数量. 若 N 值过大,会导致生成的决策树分支较大,规模较大,造成其分类效果不高. 因此,这里选择计算样本数据集 X 的"波动性"特征期望、方差和信息熵,并将之作为分类器的指纹特征. 则有

$$\text{期望 } E(X) = \sum_{i=1}^{N} x_i/N (1 \leqslant i \leqslant N) \tag{3}$$

$$\text{方差 } D(X) = E(X^2) - E(X)^2 \tag{4}$$

对于信息熵 Entropy(X)的计算,采用一维离散序列信息熵的算法.

```
Algorithm EntropyCalculate(X, block)
% X: 待求信息熵的离散序列
% block: 待求信息熵的序列要被分块的块数
% Hx: 输入离散序列 X 的信息熵,初始值为 0
1   Begin
2       首先对离散序列 X 作归一化处理
3       根据 block 划分序列 X(X_1, X_2, ..., X_block)
4       for each_block in block do
5           count = len(each_block) %统计各块数据量
6           p = count/len(X)
7           if p = = 0 do
```

```
8              Hi = 0
9          else
10             Hi = -p * log2(p)
11         End if
12         Hx = Hx + Hi
13     End for
14     return Hx % 返回输出序列信息熵 Hx
15  End
```

3）分类器选择

如图 2 所示，某固件解码后所得可反汇编二进制文件的熵谱信号呈“非线性”分布，相对其他分类算法来说，CART 决策树在处理非线性分类问题具有明显的优势，且能够很容易地处理特征间的相互作用，没有复杂的参数调整和设置，在面对存在异常点、变量数多等问题时非常稳健，故选择采用 CART 决策树作为固件解码状态检测依据.

4）分类器评估

分类器性能的评估与分类器本身同样重要. 评估分类器性能优劣可信任的一个基本工具是混淆矩阵（CM, confusion matrix），如表 1 所示.

表 1　用于分类性能评估系统的混淆矩阵

实际	分类为正例	分类为负例
正例	TP	FN
负例	FP	TN

一般来说，分类器分类性能优劣的评价指标以其分类准确率（P, precision）、召回率（R, recall）、F 值（F, f-measure）和接收者操作特征（ROC, receiver operating curve）曲线作为衡量标准. 则有

$$\text{准确率：} P = \mathrm{TP} / (\mathrm{FP} + \mathrm{TP}) \tag{5}$$

$$\text{召回率：} R = \mathrm{TP} / (\mathrm{TP} + \mathrm{FN}) \tag{6}$$

$$F\text{ 值：} F_1 = 2PR/(P + R) \tag{7}$$

$$\text{真阳率：} \mathrm{TPR} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN}) \tag{8}$$

$$\text{假阳率：} \mathrm{FPR} = \mathrm{FP}/(\mathrm{FP} + \mathrm{TN}) \tag{9}$$

其中：F 值是准确率 P 和召回率 R 的调和平均值，真阳率（TPR, true postive rate）代表分类器预测的正类中实际正实例占所有正实例的比例，也称之为敏感度，假阳率（FPR, false postive rate）代表分类器预测的正类中实际负实例占所有负实例的比例.

因此，可以通过计算 CART 决策树的分类准确率 P、召回率 R、F_1 值及其 ROC 曲线来判定其分类性能的优劣.

3　仿真与结果分析

3.1　仿真数据

实验选择应用上文提到的固件自动化分析框架中固件收集模块所收集的来自多个不同厂商的固件作为实验负样本集，选择解码后得到的可反汇编二进制文件作为实验正样本集. 其中，实验正样本用“+1”标记，实验负样本用“-1”标记.

选取实验正样本时，应遵循如下规则：

1）多个具备相同 MD5 值的可反汇编二进制文件仅选取一个作为实验分析正样本；

2）多个同类文件（如 Tplink 和 Dlink 的可反汇编二进制文件中均包括 busybox）也仅选取其中一个作为实验正样本.

需要说明的是，实验负样本（固件）在进行爬虫下载时候已经过滤具备相同 MD5 值的文件.

实验选择的来自各设备厂商的实验仿真数据集如表 2 所示.

表 2　选取来自各设备厂商的正、负样本数量

厂商	负样本数	正样本数
Schneider	456	102
Tplink	210	287
Dlink	451	366
Linksys	43	104
Tenda	194	131
Tomato	3 285	236
Openwrt	1 215	320
Buffalo	75	56
Hikvision	231	221
总计	6 160	1 823

3.2　实验设计

采用 Python 的 scikit-learn 机器学习库进行仿真实验，选取数据样本集的 80% 作为训练集，其余 20% 作为测试集，并进行交叉重复验证；为更进一步说明采用的 CART 决策树的分类效果，将之与 SVM、LogisticRegression、RandomForest 及 KNeighbors 进行对比. 各分类器参数选择如表 3 所示.

表3 各分类器的参数选择

分类器	参数选择
CART	criterion = "gini", max_depth = 5, splitter = "best
SVM	kernel = "rbf", gamma = 0.7, c = 1.0, max_iter = -1
LogisticRegression	multi_class = "ovr", solver = "liblinear", c = 1.0
RandomForest	criterion = "gini", max_features = "auto", n_jobs = 1
KNeighbors	algorithm = "kd_tree", metric = "minkowski", weights = "distance", n_neighbors = 5, leaf_size = 30

3.3 结果分析

图3为经测试集测试后绘制的各分类器的ROC曲线示意图. 其中,AUC(area under the curve)表示ROC曲线与横坐标围成的区域的面积,AUC越大,表明分类器的分类性能越好. 可以看出,相对其他分类器来说,CART决策树的分类性能最优.

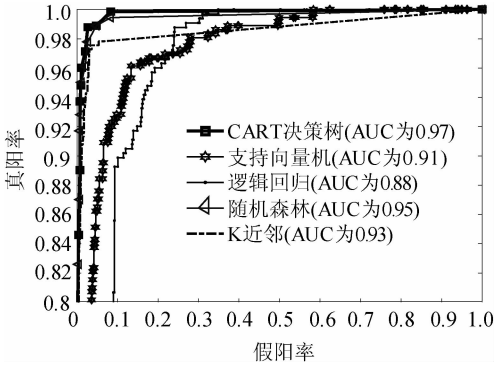


图3 各分类器分类效果示意图(ROC曲线)

同样,在绘制ROC曲线的同时,也可以得到各分类器的分类准确率P、召回率R和F值,如表4所示.

表4 各分类器的分类准确率、召回率和F值

分类器	准确率P	召回率R	F值
CART	0.96	0.98	0.97
SVM	0.81	0.89	0.85
LogisticRegression	0.70	0.86	0.77
RandomForest	0.97	0.93	0.95
KNeighbors	0.95	0.91	0.93

由表4不难看出,相比其他分类器,CART决策树的分类准确率、召回率和F值均在96%以上. 由此可见,CART可以很好地用来检测固件解码的状态,判定固件是否解码成功.

3.4 算法应用效果

将基于CART决策树的固件解码状态检测算法应用于设计的大规模嵌入式设备固件自动化分析框架,对收集的固件进行自动化解码分析(解码效率得到了很大程度的提升),成功识别出SquashFS、CramFS及JFFS2文件系统,Linux与VxWorks操作系统,boa、thttpd、lighttpd等Web组件,MIPS、PowerPC与ARM架构,DES、AES、Blowfish、Towfish等加密算法.

4 结束语

为了对固件进行深入研究与分析,设计了一种面向大规模嵌入式设备固件的自动化分析方法,该方法能够对固件进行自动化分析,提取其文件系统、操作系统、CPU指令架构等信息;并针对固件解码成功的判定难题,提出了一种基于CART决策树的固件解码状态检测算法. 实验结果表明,CART决策树对于固件解码状态具有非常好的检测效果,其检测准确率高达96%,召回率达到98%.

参考文献:

[1] Sophia. 乌克兰电网遭黑客入侵 工控网络安全敲响警钟[J]. 信息安全与通信保密, 2016(2):66-67.

[2] Costin A, Zaddach J, Francillon A, et al. A large-scale analysis of the security of embedded firmwares[C]// Proceedings of the 23rd USENIX Conference on Security Symposium. San Diego:ACM, 2014:95-110.

[3] Zaddach J, Bruno L, Francillon A, et al. Avatar: a framework to support dynamic security analysis of embedded systems' firmwares[C]// Network and Distributed System Security Symposium. San Diego:[s. n.], 2014.

[4] Rutkowski L, Jaworski M, Pietruczuk L, et al. The CART decision tree for mining data streams[J]. Information Sciences, 2014, 266(5): 1-15.