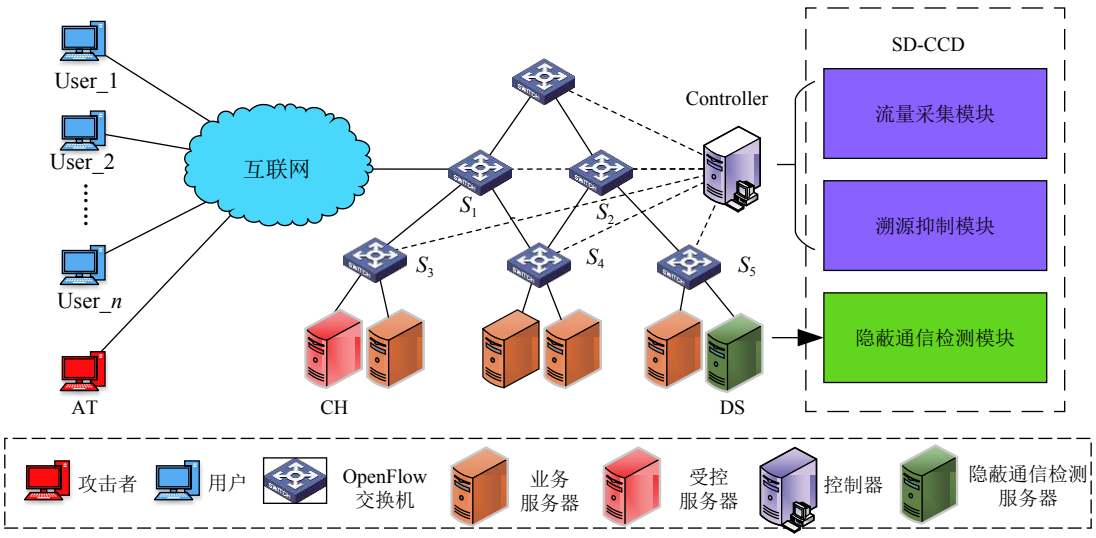

图 2 SDN 隐蔽通信检测架构

如图 2 所示, 本文提出的面向 SDN 的隐蔽通信检测机制 (Software-Defined Convert Communication Detection mechanism, SD-CCD) 由运行在隐蔽通信检测服务器 DS 上的隐蔽通信检测模块及运行在控制器上的流量采集模块及溯源抑制模块组成. 其中, 隐蔽通信检测服务器 DS 上运行隐蔽通信检测模块, 该模块包括证书提取、特征值计算以及隐蔽通信检测算法. 其主要对输入的 SSL 流量进行处理, 判断 SSL 流量是否为非法隐蔽通信. 若为非法隐蔽通信, 隐蔽通信检测服务器发送警报事件到控制器 Controller, 然后由控制器进行处理.

运行在控制器上的流量采集模块和溯源抑制模块负责 SSL 流量的采集及对非法隐蔽通信的溯源和抑制. 流量采集模块利用 OpenFlow 中的流表将 SSL 流量导入隐蔽通信检测模块 DS, 溯源抑制模块对受控服务器发起溯源并抑制其在网络中的行为. 控制器 Controller 中流量采集模块基于 SDN 特性利用 OpenFlow 流表对网络中的 SSL 报文进行采集. 虽然现行 OpenFlow 协议中还没有针对 SSL 协议的匹配项, 但是 SSL 协议封装于 TCP 协议中且通常使用 443 端口, 因此本文将使用 TCP 协议并且目的端口与源端口均为 443 的报文视为 SSL 报文. 当第一个 SSL 报文由交换机上报至控制器, 控制器计算完路由路径并下发流表时, 在流表 MATCH 域中添加运输层协议为 TCP 且源端口和目的端口均为 443 的匹配项, 同时在此流表的 ACTION 域中添加转发至隐蔽通信检测服务器所在的端口, 以此就可抓取所需的 SSL 报文.

在进行 APT 攻击时, 攻击者获取 CH 权限后, 需与 AT 建立 SSL 通信, 但此时 CH 所挂载的交换机 S_3 中没有转发 SSL 报文的流表, 因此不能转发 CH 发出的 ClientHello 报文. 根据 OpenFlow 协议, S_3 将 ClientHello 报文转发至控制器 Controller, 控制器计算出 CH 到 AT 的路由路径并依次下发流表至路径上的交换机, 同时计算出受控服务器 CH 到隐蔽通信检测服务器 DS 的路径并下发流表至路径上的交换机, 由此就可在不影响正常通信的情况下将 SSL 流量导入隐蔽通信检测服务器.

隐蔽通信检测服务器 DS 对导入的 SSL 流量进行提取, 得到 SSL 通信中服务器所使用的证书, 然后调用基于 iForest 的隐蔽通信检测算法检测此证书是否异常. 若检测结果为证书异常, 即表示当前 SSL 连接为非法隐蔽通信, 则隐蔽通信检测服务器 DS 立刻发送 SSL 连接异常警报事件至 SDN 控制器 Controller. 控制器收到隐蔽通信检测服务器发送的警报后, 立刻对当前网络中与使用非法证书的 SSL 服务器连接的客户端进行溯源, 并下发流表至异常主机挂载的交换机以阻塞此主机发送的 SSL 报文, 从而达到抑制非法隐蔽通信的目的.

2.2 基于 iForest 的隐蔽通信检测算法 (iFCCD)

基于 iForest 的隐蔽通信检测算法 (iFCCD) 对 SSL 报文中提取的服务器证书进行检测以判断网络中是否存在隐蔽通信. 其检测精度受特征值表征证书的准确

度影响, 因此在调用隐蔽通信检测算法之前需提取能够准确表征证书特性的特征值.

在介绍本文提取的证书特征值之前, 引入两个集合[11], 分别是最常见通用名集合 (the most Frequently Appeared CommonName Set, FAS) 和最常见匿名性通用名集合 (the most Frequently Appeared Anonymous CommonName Set, FAAS). 其中 FAS 是根据统计得到的频繁使用的通用名集合, FAAS 也是相同方法得到, 但目前只包含两个元素, 分别为 localhost 和 localdomain.

攻击者制作隐蔽通信使用的证书时, 为保证隐蔽性, 通常采用自签名证书, 并且为了避免自身信息暴露, 此类证书使用者和颁发者的项通常较少. 由于需保持通信的隐蔽性防止被网络安全设备发现, 攻击者在制作证书时通常会设定较短的证书有效时间. 同时攻击者会采用随机构成的域名作为证书通用名, 以此进一步提高隐蔽性. 本文中 FAS 集合和 FAAS 集合就是针对证书的域名而建立的, 若证书的域名在 FAS 中说明其属于网络中常见的域名, 即表示其可信度较高. 若证书的域名出现在 FAAS 中说明其匿名性较强, 即表示此证书的安全性较低.

基于上述特点, 本文采用了 8 个特征值来表征证书, 如表 1 所示, 分别为: (1) 是否为自签名证书; (2) 证书使用者包含的项数; (3) 证书颁发者包含的项数; (4) 证书是否被可信根验证通过; (5) 通用名是否在 FAS 中; (6) 通用名是否在 FAAS 中; (7) 使用者通用名是否符合域名格式 (CN=xxx.com); (8) 证书的有效年份. 上述八个特征值是从证书中提取的关键信息, 能够表征证书的合法性.

表 1 iFCCD 采取的证书特征值

属性描述	特征值	属性及计算细节
自签名证书/自生成证书	1	采用自签名证书/自生成证书时可疑度高
证书使用者包含的项	0.1*项数	通用名, 机构名, 国家名, 地区名; 每项分值为 0.1
证书颁发者包含的项	0.1*项数	通用名, 机构名, 国家名, 地区名; 每项分值为 0.1
证书被可信根验证通过	1	证书链使用可信根, 可疑度低
通用名在 FAS 中	1	流行性强, 可疑度低
通用名在 FAAS 中	1	匿名性强, 可疑度高
使用者的通用名符合域名格式	1	符合格式, 可疑度低
有效期时间	年份	有效期短, 可疑度高

iFCCD 算法采用孤立森林算法 (isolation Forest algorithm, iForest), iForest 是一种基于决策树的异常检测算法, 其基于数在二叉搜索树中的深度判断数据是否异常, 具有较高的检测精度. 相对于神经网络算法, iForest 的训练时间短, 达到相同精度的计算时间少.

本文采用的八个参数特征值对应的当前 iForest 中的评价值计算公式如下[15]:

$$S_i = \frac{1}{2} - \frac{1}{2}^{\frac{E(h(x))}{C}} \tag{1}$$

其中, $E(h(x))$ 为特征值 x 在树中的平均深度, C 为当前森林中数的平均深度, 计算公式如下[15]:

$$c(n) = 2H(n-1) - \left(\frac{2(n-1)}{n}\right) \tag{2}$$

$$H(i) = \ln(i) + 0.577\,215\,6649 \tag{3}$$

隐蔽通信检测启动时, 从配置文件中读取已准备好的训练数据, 训练数据均为从正常证书中提取的相关特征值, 孤立森林算法使用这些特征值训练算法模型. 假设某一特征值训练数据有 L 组, 训练阶段孤立森林有放回的取出 G 组数据, 然后构建 K 棵决策树, 得到 K 棵决策树组成的森林, 并根据孤立森林算法计算出当前森林的阈值 $threshold_i$. 8 个孤立森林分类器均采用上述的训练流程. 孤立森林算法训练完成之后, 隐蔽通信检测正式进入检测阶段.

算法. iFCCD

Input: SSL 流量
Output: 恶意证书检测结果

1. **Begin**
2. Get Hello packet in SSL
3. Extract SSL certificate
4. Calculate features of SSL certificate
5. **if** this certificate is Self-signed certificate **then**
6. append '1' to eigenvalue matrix
7. **else**
8. append '0' to eigenvalue matrix
9. **end if**

```
10.  Append the length of subject to eigenvalue matrix
11.  Append the length of Issuer to eigenvalue matrix
12.  if the root in certificate is trusty then
13.     append '1' to eigenvalue matrix
14.  else
15.     append '0' to eigenvalue matrix
16.  end if
17.  if domain name in FAS then
18.     append '1' to eigenvalue matrix
19.  else
20.     append '0' to eigenvalue matrix
21.  end if
22.  if domain name in FAAS then
23.     append '1' to eigenvalue matrix
24.  else
25.     append '0' to eigenvalue matrix
26.  end if
27.  if subject's domain name is belong to normal format
28.     then
29.     append '1' to eigenvalue matrix
30.  else
31.     append '0' to eigenvalue matrix
32.  end if
33.  Append valid time of certificate to eigenvalue matrix
34.  Send matrix to iForest
35.  if the result of detection is true then
36.     send result to controller
37.  else
38.     goto end
39.  end if
40. End
```

进行检测时, 隐蔽通信检测算法接收新的证书特征值矩阵, 然后调用训练完成的算法模型判断此证书特征值是否为异常. 当待检测的特征值矩阵输入时, 各特征值分别送入相应的分类器. 在各自的分类器中, 每个决策树返回待测特征值在树中的层数, 然后得到当前特征值在该森林中的评价值 S_i. 在一个分类器中, 若特征值评价值小于该分类器阈值, 则表示当前数据为异常数据.

在 iFCCD 中, 最后计算分类器的评价值, 若该评价值小于阈值则判定当前特征值矩阵为异常, 即当前证书为非法证书; 否则判定为正常证书, 如公式 (4) 所示:

$$\sum_{8}^{i=1} S_i < \sum_{8}^{i=1} threshold_i \tag{4}$$

综上所述, 为准确表征证书特征, iFCCD 算法采用八个特征值表征证书信息, 同时使用孤立森林作为异常检测算法, 提高非法证书的检测精度. 在训练阶段中, iForest 的训练数据均为正常证书的特征值. 在检测阶段中, iFCCD 利用已训练的 iForest 模型分别对 8 个特征进行判定, 最后综合判定表征证书信息的八元组矩阵是否异常, 以此可判定网络中是否存在隐蔽通信.

3 实验验证与分析

3.1 实验环境

本文采用的数据集有三个: ITOC2009[16]、Contagio Malware Dump (CMD)[17]以及从本地抓取的正常 HTTPS 证书. 训练数据中有 100 个证书, 其中 50 个为 ITOC 数据集中的正常证书, 50 个为 CMD 数据集中的正常证书. 测试数据有 271 个证书, 其中包括 74 个 CMD 数据集中正常的证书, 26 个本地抓取的正常证书以及 171 个 CMD 数据集中的非法证书. 首先提取训练数据集中证书的特征值, 然后将训练证书特征值作为输入训练孤立森林算法. 在训练 iForest 算法模型时, 从 100 组训练数据中有放回地取出 75 组数据, 构建 100 棵决策树.

本文采用文献[11]中提到的证书可信度计算算法 (文献[11]中将其简称为 CCD 算法) 作为对比实验, 其中计算结果小于 0 时视为证书非常可疑, 计算结果大于 0 小于 0.85 时视为证书较可疑, 大于 0.85 则认为证书正常. 本文使用受试者工作特征曲线 (Receiver Operating Characteristic curve, ROC 曲线) 衡量 iFCCD 与 CCD 两种算法的检测精度.

3.2 实验结果

本文对上述提出的检测算法的实验验证过程如下: 首先使用训练数据训练 iForest 模型, 检测时将数据集里的证书依次提取出来, 获得其中的证书特征值, 然后将证书特征值依次送入训练好的 iForest 模型中. 在衡量 iFCCD 与 CCD 两种算法精度时, 通过动态调整 iForest 的阈值得到的 ROC 曲线如图 3 所示.

图 3 中, 横坐标为误检率, 纵坐标为检测精度, 其中实线为本文提出的 iFCCD 算法得到的检测结果, 虚线为对比算法 CCD 的检测结果. 从图中可以看到, iFCCD 算法的误检率在 16% 时可以达到 100% 的检测精度, 而 CCD 算法要达到 100% 的检测精度其误检率需达到 24%. 显然本文提出的证书检测算法在提高证书检测精度的同时能够降低误检率. 图 3 中, 两种算法的 ROC 曲线均在误检率达到一定值时快速上升, 出现上述现象的原因是数据集里的非法证书的特征值基本

相似.

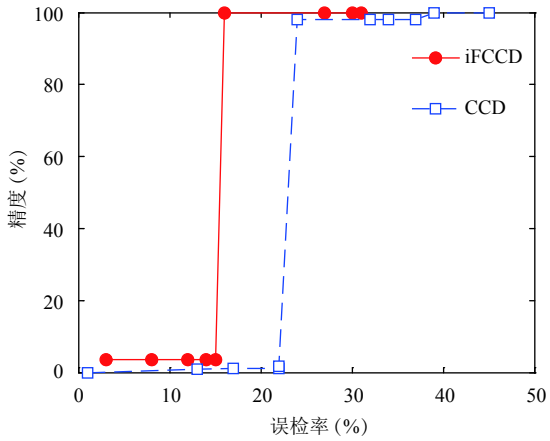

图 3 iFCCD 及 CCD 算法 ROC 曲线图

3.3 实验结果分析

为进一步说明 iFCCD 及 CCD 算法的检测精度, 本节对 iFCCD 及 CCD 算法的检测结果进行了详细分析.

数据集 ITOC2009 中出现了大量类似如下格式的非法证书: “Version=3; Issure:C=--; ST=SomeState; Validity:one year; OU=SomeOriganizationalUnit; EMAIL=root@localhost.localdomain; O=SomeOrganization; L=SomeCity;”, 对于上述非法证书, CCD 算法计算出的可信度为–0.1, CCD 算法认定其为非常可疑的证书, 既 CCD 算法能够成功检测上述非法证书.

但是对于图 4 所示的非法证书, 使用 CCD 算法计算的可信度为 0.35(属于较可疑级别), 其不能精确检测上述非法证书. 通过进一步分析可知此证书属于非法证书, 因为其使用者项中都是匿名信息符合非法证书的特征, 而且证书有效时间较短. 此证书相对于 ITOC2009 数据集的一般非法证书仅仅将使用者 CN 换了一个匿名, 就使得 CCD 产生漏检. 虽然可以将匿名域名加入到 FAAS 中提高 CCD 的检测精度, 但匿名性域名范围太广无法将全部新匿名域名包含在集合中, 因此采用 CCD 方法容易导致漏检. 而在采用 iFCCD 方法检测上述非法证书时, 由于 iForest 所具有的检测精度高的优点, 所以 iFCCD 方法能够将其判定为非法证书.

与 CCD 算法相比, iFCCD 算法的误检率更低. 图 5 所示为 ITOC2009 的合法证书, CCD 算法计算该证书可信度为-0.1, 判定其为非法证书, 但是经查询得知此证书为根级证书属于可信任证书, CCD 算法明显导致了误检测. 而 iFCCD 算法能够正确判定其为合法证书.

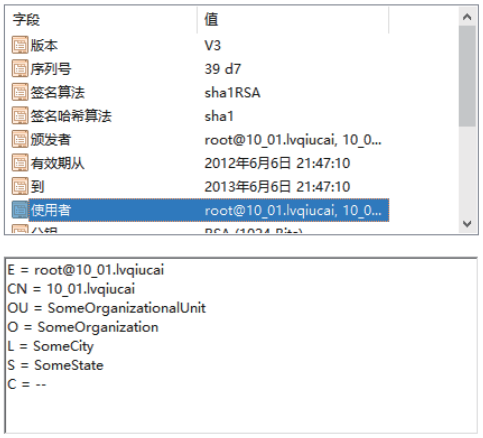

图 4 非法证书使用者信息示例

图 5 正常证书使用者信息示例

从上述分析得知, 本文提出的基于 iForest 的隐蔽通信检测机制能够在降低误检率的同时提高检测精度.

4 总结

为解决 SDN 网络中存在的隐蔽通信检测问题, 本文提出了面向 SDN 的隐蔽通信检测机制, 该机制利用 SDN 的特性准确获取网络中可能存在的隐蔽通信流量. 针对 CCD 方法较为模糊的判定结果问题, 本文提出了基于 iForest 算法的隐蔽通信检测算法 iFCCD, 该算法可以降低误检率、提高检测精度, 同时避免了使用经验值作为阈值可能导致的误检率增高或精度降低的问题. 本文提出的 iFCCD 方法具有较好的可扩展性,

只需改变提取证书的特征值或训练数据集就可运用于不同场景的 SDN 网络.

参考文献

1 Nunes BAA, Mendonca M, Nguyen XN, *et al*. A survey of software-defined networking: Past, present, and future of programmable networks. IEEE Communications Surveys & Tutorials, 2014, 16(3): 1617–1634. [doi: 10.1109/SURV.2014.012214.00180]

2 McKeown N, Anderson T, Balakrishnan H, *et al*. OpenFlow: Enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69–74. [doi: 10.1145/1355734]

3 Casado M, Freedman MJ, Pettit J, *et al*. Ethane: Taking control of the enterprise. ACM SIGCOMM Computer Communication Review. Kyoto, Japan. 2007. 1–12. [doi: 10.1145/1282380.1282382]

4 Yoon MS, Kamal AE. Power minimization in fat-tree SDN datacenter operation. Proceeding of 2015 IEEE Global Communications Conference. San Diego, CA, USA. 2015. 1–7. [doi: 10.1109/GLOCOM.2014.7417135]

5 Daly MK. Advanced persistent threat. USENIX 23rd Large Installation System Administration Conference. Baltimore, MD, USA. 2009, 4(4): 2013–2016.

6 Tankard C. Advanced persistent threats and how to monitor and deter them. Network Security, 2011, 2011(8): 16–19. [doi: 10.1016/S1353-4858(11)70086-1]

7 Li MC, Huang W, Wang YB, *et al*. The study of APT attack stage model. Proceedings of 2016 IEEE/ACIS 15th International Conference on Computer and Information Science. Okayama, Japan. 2016. 1–5. [doi: 10.1109/ICIS.2016.7550947]

8 Chandra JV, Challa N, Pasupuletti SK. A defense approach from advanced persistent threat through defense in depth mechanism for cloud security. Advanced Science and Technology Letters, 2017, 147: 268–275.

9 Rass S, König S, Schauer S. Defending against advanced persistent threats using game-theory. PLoS One, 2017, 12(1): e0168675. [doi: 10.1371/journal.pone.0168675]

10 Sood AK, Enbody RJ. Targeted cyberattacks: A superset of advanced persistent threats. IEEE Security & Privacy, 2013, 11(1): 54–61.

11 Cao ZG, Xiong G, Zhao Y, *et al*. Two-phased method for detecting evasive network attack channels. China Communications, 2014, 11(8): 47–58. [doi: 10.1109/CC.2014.6911087]

12 Fu PP, Guo L, Xiong G, *et al*. Classification research on SSL encrypted application. In: Yuan Y, Wu X, Lu Y, eds. Trustworthy Computing and Services. ISCTCS 2012. Communications in Computer and Information Science, vol 320. Springer, Berlin, Heidelberg. 2012. 404–411. [doi: 10.1007/978-3-642-35795-4_51]

13 Bro-Project. 2017. Intelligence Framework. https://www.bro.org/sphinx/frameworks/intel.html. [2018-05-21].

14 Ghafir I, Přenosil V, Hammoudeh M, *et al*. Malicious SSL certificate detection: A step towards advanced persistent threat defence. Proceedings of International Conference on Future Networks and Distributed Systems. Cambridge, Britain. 2017. 1–6. [doi: 10.1145/3102304.3102331]

15 Liu FT, Ting KM, Zhou ZH. Isolation-based anomaly detection. ACM Transactions on Knowledge Discovery from Data (TKDD), 2012, 6(1): 1–39. [doi: 10.13140/RG.2.1.2591.8165]

16 ITOC research: CDX datasets. http://www.oalib.com/references/13245491.

17 Contagio Malware Dump: Collection of PCAP files categorized as APT, Crime or Metasplot. http://contagiodump.blogspot.com/.