基于协议状态图遍历的 RTSP 协议漏洞挖掘

李佳莉[1] 陈永乐[1] 李 志[2,3] 孙利民[2,3,4]
(太原理工大学计算机科学与技术学院 太原 030600)[1]
(物联网信息安全技术北京市重点实验室 北京 100093)[2] (中国科学院信息工程研究所 北京 100093)[3]
(中国科学院大学 北京 100049)[4]

摘 要 目前,视频监控设备中很多摄像头、DVR、NVR 都支持 RTSP 协议,而且由 RTSP 协议引起的缓冲区溢出漏洞个数较多,危害性大,因此对 RTSP 协议的研究具有理论意义和应用价值。直接利用模糊测试框架中的方法生成的测试用例数量庞大,测试过程耗时长。针对上述问题,以视频监控设备的 RTSP 协议为研究对象,提出对协议基本块的样本集进行去重,利用协议状态间的约束关系和状态转移的关联关系构造协议状态图,并基于协议状态图进行深度遍历的方法。该方法减少了测试用例的生成,并提高了生成的有效性。对 RTSP 协议进行 fuzzy 测试时,利用发送 TCP 探测包的方法,判断测试目标是否异常。去除记录的异常测试用例的冗余部分,以缩短后续重放过程的耗时,从而提高漏洞挖掘的效率。

关键词 视频监控设备,RTSP 协议,模糊测试,漏洞挖掘
中图法分类号 TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.09.028

Mining RTSP Protocol Vulnerabilities Based on Traversal of Protocol State Graph

LI Jia-li[1] CHEN Yong-le[1] LI Zhi[2,3] SUN Li-min[2,3,4]
(College of Computer Science and Technology, Taiyuan University of Technology, Taiyuan 030600, China)[1]
(Beijing Key Laboratory of IOT Information Security, Beijing 100093, China)[2]
(Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China)[3]
(University of Chinese Academy of Sciences, Beijing 100049, China)[4]

Abstract Currently, many video surveillance equipments like cameras, DVRs, and NVRs support RTSP protocol, and the number of buffer overflow vulnerabilities caused by the RTSP protocol is large and harmful. Therefore, the research on the RTSP protocol has both application value and theoretical significance. The number of test cases generated by directly using the fuzzy test framework is huge, and the test process takes a long time. Aiming at the above problems, this paper took the RTSP protocol of video surveillance equipment as the research object, and proposed a method which removes duplicate sample set of the protocol basic block, uses the constraint relationship and state transition between protocol states to construct protocol state diagram, and dose deep traversal based on protocol state diagram. This method reduces the generation of test cases and improves the effectiveness of generation. When the RTSP protocol is tested by fuzzing method, the method of sending a TCP probe packet is used to determine whether the test target is abnormal. The redundant part of the recorded abnormal test case is removed, which facilitates subsequent playback and reduces the time, thereby improving the efficiency of vulnerability mining.

Keywords Video surveillance equipment, RTSP protocol, Fuzzy test, Vulnerability mining

1 引言

当前视频监控系统已被广泛应用于军事、工业、农业、交通、医疗、家庭等领域,ZoomEye 搜索引擎搜索到 2017 年暴露在互联网上的视频监控设备已多达 3581312 个。在普遍使用的视频监控设备中存在一些高危漏洞,这些漏洞的危害性较大,对社会和公共利益安全造成了很大的影响。例如,2008 年 8 月发生在土耳其的石油管道爆炸事件中,正是因黑客利

到稿日期:2017-08-17 返修日期:2017-12-26 本文受国家重点研发计划(2016YFB0800202),国家自然科学基金(61401300),国防基础科研计划-部队纵向(JCKY2016602B001),国家电网公司科学技术项目(52110417001B)资助。

李佳莉(1991—),女,硕士生,主要研究方向为物联网安全,E-mail:1007475906@qq.com;陈永乐(1983—),男,副教授,硕士生导师,CCF 会员,主要研究方向为物联网安全、无线传感器网络等,E-mail:chenyongle@tyut.edu.cn(通信作者);李 志(1985—),男,博士,助理研究员,主要研究方向为移动计算、容忍延迟网络、物联网及安全;孙利民(1968—),男,博士后,研究员,博士生导师,主要研究方向为物联网及安全。

用监控摄像头通信软件的漏洞入侵了报警系统，导致大火发生 40 分钟后才被发现；2016 年 10 月，美国大量互联网视频监控设备被控制，黑客通过攻击北美 DYN 域名解析服务，导致美国上千家公司的 DNS 解析出现问题。由此可见，摄像头漏洞造成的危害范围广，严重威胁了公共安全和国家安全。导致摄像头存在安全漏洞的原因有很多种，特别是网络协议是两个主体之间进行网络通信的独特规则。在网络摄像头中协议是通信的主要支撑，因此由协议引起的漏洞居多，如缓冲区溢出、格式化字符串等都会导致无法预计的后果；并且通过这种方式，攻击者可以获得特权，干扰甚至完全破坏关键系统的网络通信。其中，有些协议的格式是已公布的，有些协议的格式是未公布的，即包括公有协议和私有协议。大多数视频监控设备支持 RTSP 协议，RTSP 协议在网络上为流媒体数据的传输提供控制服务，需要检测的漏洞较多。根据中国科学院信息工程研究所全球视频监控系统实验平台统计，全球可搜索到支持 RTSP 协议的监控设备有 3278931 个，其中由 RTSP 引起的缓冲区溢出漏洞的个数为 71010。因此，在 RTSP 协议使用较广的情况下，安全性和健壮性的研究具有理论意义和应用价值。挖掘出未被发现的视频监控网络协议漏洞很有必要；挖掘网络协议实现的安全漏洞亦是一个迫在眉睫的重要问题。

本文主要针对视频监控设备的 RTSP 协议，提出了一种基于协议状态图遍历的方法进行漏洞挖掘，以检测设备中是否存在由协议中字段异常引起的缓冲区溢出漏洞。首先使用去重算法对 RTSP 协议的基础块进行去重，从而减少测试用例的生成数量。然后利用 RTSP 协议状态间的约束关系和转移关系构造协议状态图，并基于这个协议状态图深度优先遍历，从而使得生成测试用例的遍历路径覆盖率高，且生成的测试用例有效。在测试过程中，利用 TCP 的 3 次连接原理判断所发送的测试用例是否引起目标设备的异常，并利用目标监控机制进行异常测试用例的记录。对记录的异常测试用例去除冗余，减少重放数量，以便在对异常测试用例进行二次重放时节约时间。本文利用所提方法对 Hikvision，Tiandy，Linksys 3 款设备进行了模糊测试，结果证明，所提方法除发现已公布的漏洞外，还发现了 3 个未被公布的缓冲区溢出漏洞，并且测试效率分别提高了19.7%，29.2%和 35.8%。

2 相关工作

漏洞容易被利用和攻击，从而造成泄密等安全事件的发生。目前挖掘漏洞的方法有手工测试、静态分析、动态分析、二进制比对和模糊测试[1]。

手工测试是指由测试者手工向目标对象发送有效和无效的数据，观察目标的响应。这种方法虽然容易实现，但对测试者经验的要求较高[2]。静态分析是指不运行软件，主要通过对目标程序的语法等规则进行分析来发现漏洞。虽然该方法操作起来相对简单，但其误报率较高。二进制比对方法更偏向于漏洞分析，通过比对补丁的前后二进制文件来确定存在的漏洞[3]。但该方法不适用于复杂的文件，且误报率高。模糊测试是指对目标程序输入畸形数据，通过观察目标对象的响应是否出现异常并结合源码，来进一步确认其是否引起安全漏洞。畸形数据是指在网络协议的报文中，一部分字段是正常的，一部分字段是变异后的。模糊测试的对象包括文件格式和协议格式，能够用来检测缓冲区溢出等类型的漏洞。与其他几种漏洞挖掘方式相比，该方法的误报率相对较低。

目前因为模糊测试占有绝对优势，使用模糊测试进行漏洞挖掘的方法已经取得了一些成果。Ma 等[4]于 2014 年提出了一种基于分类树和启发式算子的方法，首先通过建立协议的分类树来获得协议数据的字段属性，然后通过启发式算子删除无用项，以此来减小模糊测试数据的规模，使测试执行的速度提高了 3.61%。2016 年，该团队[5]又提出了采用基于规则的状态机和状态规则树来指导模糊测试数据生成的方法。首先通过建立规则的状态机模型来形式化描述协议的状态，并消除安全路径；然后根据状态规则树描述状态和信息的关系，删除无用项，从而减少测试数据量，找到了使用 Sulley 同样可以发现的已有漏洞，但没有发现新漏洞。Kim 等[6]针对工控协议 MMS，使用字段分类的方法生成测试用例，通过对数据的字段、内容和规则进行分类，生成了 1901743 个测试用例，发现了 MMS 协议中的 3 个未知异常，执行时间较长。Han 等[7]提出了一种基于多领域模糊测试技术的 RATM 模型来对 MAC 层协议进行测试，挖掘出了 peach 工具没有发现的漏洞。Li 等[8]基于 fuzz 网络协议安全测试提出引入遗传算法来生成测试用例，该方法是在完成协议序列对齐和分组格式识别的基础上实施的，并通过设计遗传算法中的适应度函数来提高模糊测试的效率。Wang 等[9]提出 SeededFuzz 模型，通过静态分析、动态监测和符号执行选择更安全的种子来提供定向模糊测试，以此覆盖程序的关键点，提高漏洞检测能力。胡昌振[11]和 Ma 等[10]分别提出了有状态网络协议的半合法化和半有效的模糊测试方法，通过扩展有限状态机[12]的测试序列，使用半合法化和半有效算法，进一步执行变异操作，得出最后的模糊序列，提高测试效率。Ma 等[13]也针对有状态的网络协议模糊测试实现了一个被称为 SulleyEX 的模糊器，但该模糊器是基于 Sulley 框架修改会话管理和数据生成模块实现的，目前 Sulley 框架已不再维护。

综上所述，目前使用模糊测试进行漏洞挖掘的方法虽然通过状态规则树或排列测试序列等方法在一定程度上减少了测试用例的生成，但与原框架相比，在框架使用或挖掘出有效异常方面仍存在缺陷。本文提出的基于协议状态图深度遍历的算法是对输入的基本块去重，在基本块间的约束关联关系和状态转移关联关系的基础上构造协议状态图。本文算法执行协议路径的覆盖率高，生成的测试用例少，并且发现了未被公布的漏洞信息。

3 算法设计

本文提出了基于协议状态图深度遍历的模糊测试方法，其中协议状态图是根据协议数据包内基础块间的约束关联关系以及协议状态转移关联关系构造的。在测试中，如果样本

重叠过多，会造成生成的测试用例冗余，严重影响测试效率。因此，减少重复的样本可以减少测试用例的生成数目。对协议状态图进行深度遍历，生成测试用例，执行模糊测试，提高协议覆盖率，生成高效的测试用例，有效地发现目标设备存在的漏洞。

3.1　基本块去重算法

当存在多个待测文件时，每个文件可能包含一定数量的基本块，这些基本块可能会出现重叠。因此，遍历多个样本集，取出最大基本块，并记录基本块的地址，防止重复比较。最后，样本集被分为两部分：最大基本块和剩下的基本块。剩余的基本块如果属于最大基本块中覆盖的部分，则表示该基本块已存在，可以优化去除，以防止测试用例生成时重复遍历执行路径。具体算法如算法 1 所示。

算法 1　协议样本去重算法 Rm_Duplication

输入：原始样本集 $S=\{s_1,s_2,\cdots,s_n \mid n\in N\}$，已经执行的基本块地址集 BA

输出：去重后的样本集 S'

```
1. S'=[]
2. BA=[]
3. for each s_i in S://从 S 中取得最大长度基本块 s_i
4.    S'=S'+s_i
5.    BA=BA+BA_i
6.    S=S-s_i
7. for each s_j in S:
8.    if S_j∈S & BA_j∩BA≠BA_j:
9.    S=S-s_j
10.       S'=S'+s_j
11.       BA=BA+BA_j
12. if S is not None:
13.       continue
14. end for
15. return S'
```

3.2　基于协议状态图深度遍历的算法

通过分析去重后的基本块，对状态之间转移的认证条件和约束等关系进行关联，并构造协议状态图，设置最大遍历深度，防止遍历复杂协议时效率低下。以协议状态图为输入进行深度遍历，遍历完一个状态后，通过执行节点的状态转移函数进行状态转移，以提高协议覆盖率和生成测试用例的有效性。

基于协议状态图深度遍历的算法如算法 2 所示。

算法 2　基于协议状态图深度优先遍历的模糊测试算法 Deep_Fuzz

输入：协议状态图 States_graph，目标监控设备 S_Device

输出：触发漏洞的测试用例集合

```
1. DEEP_FUZZ(States_graph,S_Device)
2. Vul_Fuzz_Testcase=[]
3. Max_depth=N//最大遍历深度
4. for each S_i in States_graph.nodes:
5.    Depth=0
6. Testcase_Sequences=Fuzz_Generate(S_i.Blocks_Function)
7.    for Sequence in Testcase_Sequences:
8.    Sending_Testcase (Sequence,S_Device)//发送测试用例到目标监控设备
9.    Device_Alive=Device_Moniter(S_Device)
10.   if Device_Alive==False:
11.      Vul_Fuzz_ Testcase.append(Sequence)
12. With open(os.path.join(crash_path,'boofuzz_crash_log.txt'),'a') as f:f.write(Sequence) //记录异常
13.      else:self._fuzz_data_logger.log_pass("Some data received from target.")
14.   end for
15.   Execute(S_i.edge)
16.   Depth+=1
17.      if Depth<=Max_depth:
18.         continue;
19.   end for
20.   return Vul_Fuzz_Testcase
```

4　框架设计

本文基于 boofuzz 框架进行模糊测试。boofuzz 继承了 Sulley[14-15] 框架中数据生成方便快捷以及目标程序崩溃后能够进行复位的优势，且可扩展、易安装。综合考虑后，本文使用易改进的 boofuzz 框架。本实验设计的整体测试流程图如图 1 所示。

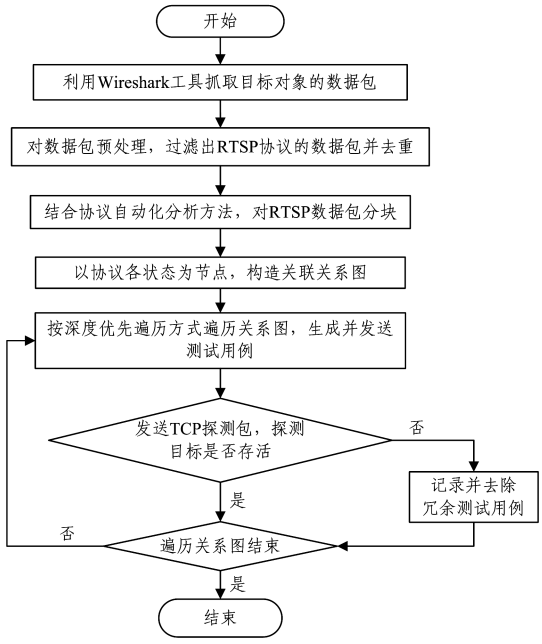

图 1　模糊测试算法的流程图

Fig. 1　Flowchart of fuzzy test algorithm

在模糊测试的过程中，boofuzz 原框架的测试用例生成方法是对定义的原始数据进行随机的强制性变异，生成的测试用例数量较大，尽管覆盖率较高，但执行时间较长，因此直接影响了模糊测试的性能。而本文模糊测试过程中的测试用例是根据协议基础块去重后协议各状态之间的约束关联关系和各状态转移关联关系来构造协议状态图，再基于协议状态图深度遍历生成的，因此协议覆盖率高，生成的测试用例具有针对性。因为基于协议状态图深度遍历是在协议语法规约和协议状态机的基础上进行的，所以通过协议状态机的支持，提高

了协议状态的覆盖率。在考虑协议语法规约的前提下对数据包字段随机变异，从而保证了生成的测试用例具有针对性，且有效性比例较大。利用 TCP 的 3 次握手原理编写脚本，判断向目标程序发送的测试用例是否会引起目标程序崩溃。若 TCP 连接失败，则记录上一个发送的测试用例，用以区分发送的全部测试用例，以便后续进行重放和确认异常包。最后通过编写重放脚本，查看目标的响应，进一步确认引起目标程序异常的测试用例，从而提高模糊测试的效率。

4.1 数据采集

本文模糊测试的目标主要是针对 Hikvision，Linksys 等品牌的 RTSP 协议，检测其中有可能引起缓冲区溢出的字段。获得数据包之前，先对目标监控设备进行访问，访问过程可以通过 VLC 播放工具或 Web 访问进入监控设备的管理页面，并进行视频预览；然后通过 Wireshark 工具捕获到待测监控设备的数据包，并保存为. pcap 格式，编写脚本，进一步过滤出 RTSP 协议的请求数据包；最后利用去重算法对数据包进行去重，并将结果作为后续生成原始数据定义的输入。

4.2 数据包预处理及协议状态图的构造

对于捕获的 pcap 数据包，结合协议自动化分析和去重算法对协议的各状态进行分块去重。以网络协议 RTSP 为例，其状态包括 options，describe，setup，play 等。之后对数据包进行预处理，首先用分隔符“\r\n\r\n”对 RTSP 协议的请求内容进行各个状态的分块，再根据空格符划分出数据包的协议方法和 url，然后利用“\r\n”对各方法中的内容进行分块，最后对各方法内容的每行以“:”划分，区别出 key-value，具体代码如图 2 和图 3 所示。结合自动化分析和人工分析，对数据包内的基础块进行函数关联，获得基础块之间的取值约束、长度约束等关系（比如 play 方法中的 session 字段的取值依赖于 setup 方法中的 realm，nonce，username 等字段的值），继而构造协议状态图，如图 4 所示。编写自动生成原始数据定义的脚本，并将生成的原始数据保存到 requests 中，设置需要进行模糊测试的字段；再通过编写自动化脚本，绑定待测视频监控设备的 IP、端口以及协议类型，并绑定网络监视代理等，与目标视频监控设备建立通信。

如图 2 所示，该段是 RTSP 协议报文中的状态之一 PLAY 方法。该方法包含字段如图所示，具有的特点是除第一行的地址之外，其余每一行是 Key:Value 的形式，以冒号为分隔符进行分割。

```
PLAY rtsp://192.168.12.143:554/play1.sdp/ RTSP/1.0
CSeq:6
Authorization:Digestusername="admin",realm="8ce748c0316b",nonce=
"10b0807bb13e63e07f81b1fde2866c3f ", uri = "rtsp://192. 168. 12. 143:
554/play1. sdp/",
User-Agent:LibVLC/2.2.4(LIVE555 Streaming Media v2016.02.22)
Range:npt=0.000-
```

图 2 原始请求数据包（play）

Fig. 2 Original request packet(play)

图 3 表示的是针对图 2 的报文以冒号和空格为分隔符作为分割进行数据块处理，并以块进行连接，对 Value 值进行变异，生成测试用例。

```
"data":
{
  "pieces":
    [{"name":"action","value":"PLAY"},
    {"name":"","value":" "},
  {"name":"","value":" rtsp://192.168.12.143:554/play1.sdp/"},
  {"name":"","value":" "},
  {"name":"","value":"RTSP/1.0"},
  {"name":"","value":"\r\n"},
  {"name":"","value":"CSeq"},
  {"name":"","value":":"},
  {"name":"","value":6},
  {"name":"","value":"\r\n"},
  {"name":"","value":"Authorization"},
  {"name":"","value":":"},
  {"name":"","value":"Digestusername='"},
  {"name":"username","value":"admin"},
  {"name":"","value":"',realm='"},
  {"name":"realm","value":"8ce748c0316b "},
  {"name":"","value":"', nonce ='"},{"name":"nonce","value":
  "10b0807bb13e63e07f81b1fde2866c3f ","hanlder":"nonce_handler"},
  {"name":"","value":"',uri='"},
  {"name":"uri","value":" rtsp://192.168.12.143:554/play1.sdp"},
  {"name":"","value":" User-Agent"},
  {"name":"","value":":"},
  {"name":"","value":" LibVLC/2.2.4(LIVE555 Streaming Media
  v2016.02.22)"},
  {"name":"","value":"\r\n"},
  {"name":"","value":" Range"},
  {"name":"","value":":"},
  {"name":"","value":" npt=0.000-"},
  {"name":"","value":"\r\n"},
  {"name":"","value":"\r\n"}],
{
  "name":"response_handler",
  "parameter":
  [
    blocks[2].get('data').get('pieces')[0].get('action'),
    blocks[2].get('data').get('pieces')[13].get('username'),
    blocks[2].get('data').get('pieces')[15].get('realm'),
    blocks[2].get('data').get('pieces')[17].get('nonce'),
    blocks[2].get('data').get('pieces')[19].get('uri')
  ]
}
```

图 3 分块处理（play）

Fig. 3 Block processing(play)

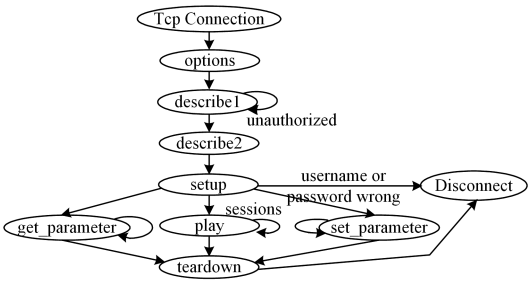

图 4 RTSP 协议状态

Fig. 4 RTSP protocol status

4.3　遍历协议状态图及启动模糊测试

根据 RTSP 协议数据包中各状态之间的关联关系构造协议状态图，结合算法 2，以深度优先遍历的方式对其进行遍历，生成并发送相应的测试用例，启动 session.fuzz()函数开始测试。测试过程中本文对框架中的 session.py 作了修改，根据发送 TCP 探测包来判断监控设备是否异常。若发送测试用例后没有接收到目标监控设备的响应，则利用 TCP 3 次握手原理发送 TCP 探测包，若收到返回的确认包，则目标设备正常；否则，说明目标设备崩溃，记录之前所发送的测试用例。测试用例是由字段尽可能出现的情况组成的，所以具有重复性，在记录时将根据变异字段长度、目标重启的时间等属性去掉冗余部分，以减少重放量而又不失准确性。每次 TCP 探测包发送结束之后，若整个协议状态图遍历达到最大深度，则回溯，继续遍历协议状态图，直到协议状态图遍历完毕，测试结束。

4.4　查看执行进度

通过执行 parse_session.py 文件或者 http://127.0.0.1:26000 来观察测试执行的进度，测试即使因为网络中断，重新执行时仍可以继续，不需要重新开始，如图 5 所示。

图 5　查看执行进度

Fig. 5　Viewing progress

5　实验分析

本文对 Hikvision 品牌的 DS-2CD7153-E 型号、Tiandy 品牌的 TC-NC9200S3E-2MP-E-IR30 型号以及 Linksys 品牌的 WVC54GCA 型号的视频监控设备执行模糊测试。编写重放脚本，通过重放异常的测试用例来观察测试目标的反应，再对返回的结果信息进行分析，发现异常，并在 RTSP 的 RFC[16] 和 CVE、CNNVD 等官方安全漏洞库中查找该型号设备已发现的漏洞中是否存在此异常。通过对比和对摄像头的观察，来判断记录的异常测试用例是否引起了摄像头的崩溃，从而确认漏洞信息。

5.1　模糊效率

针对上述 3 个品牌，将 boofuzz 原框架中的强制性变异方法和本文提出的基于协议状态图深度遍历的方法进行测试对比，其中后者产生的测试用例个数分别比前者产生的测试用例个数减少了 16.5%，15.1%，23.4%，如图 6 所示。后者对记录的异常测试用例中同一字段的变异去除冗余，与前者监控机制记录的异常测试用例相比减少了 63.4%，67.3%，61.3%，提高了重放效率，如图 7 所示。强制性变异方法虽然也可以检测出引起缓冲区溢出漏洞的字段，但生成的测试用例数目较多，导致测试时间较长，效率不高。本文提出的方法不仅检测出了引起缓冲区溢出漏洞的字段，而且生成的测试用例数量较少，缩短了测试的执行时间，相比原框架，测试效率分别提高了 19.7%，29.2%和 35.8%，如表 1 所列。

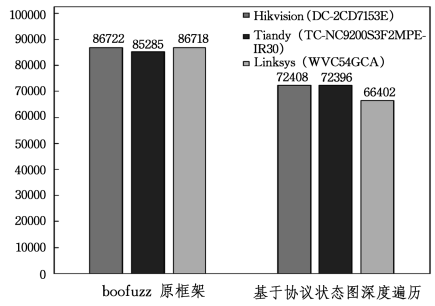

图 6　测试用例生成总数

Fig. 6　Total test case

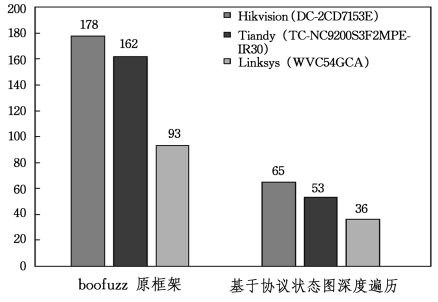

图 7　异常测试用例记录

Fig. 7　Exception test case record

表 1　模糊效率

Table 1　Fuzzy efficiency

测试方法	设备	测试用例数量	测试执行时间
boofuzz	Hikvision	86722	62 h19 min
基于协议状态图深度遍历	(DS-2CD7153-E)	72406	50h2min
boofuzz	Tiandy	85285	60 h38 min
基于协议状态图深度遍历	(TC-NC9200S3E-2MP-E-IR30)	72403	42 h57 min
boofuzz	Linksys	85271	65 h13 min
基于协议状态图深度遍历	(WVC54GCA)	66402	41 h51 min

5.2　漏洞发现

分别在对这 3 个品牌的异常测试用例记录去除冗余后进行重放观察，共发现了 3 个原框架方法未发现的缓冲区溢出漏洞。其中，发现导致 DS-2CD7153-E 型号监控设备崩溃的异常字段有 2 个：CSeq 和 Range。实验证明，Hikvision 品牌的 DS-2CD7153-E 型号的 RTSP 协议中 play 方法的 Range 字段会引起缓冲区溢出，导致摄像头崩溃；而 CSeq 字段是由特殊字符“;”引起的摄像头崩溃。导致 TC-NC9200S3E-2MP-E-IR30 型号监控设备崩溃的异常字段是 Describe 方法中的 URL，URL 字段值超长时会造成缓冲区溢出，导致摄像头崩溃。除此之外，导致 WVC54GCA 型号监控设备崩溃异常的字段有 1 个：Content。在该型号的 RTSP 协议 Setup 方法中的 Content 字段超长时，也会引起缓冲区溢出漏洞，导致摄像头崩溃。重放异常模糊测试用例后的结果如表 2 所列。以上所记录的异常测试用例中目前只有 Hikvision 品牌 DS-2CD7153-E 型号的 play 方法中的 Range 字段在 2013 年时被 CNNVD 收录，其他字段未被检测出异常。因此，还需专业测试人员进行进一步分析评估，以便尽快对问题进行修复。

表 2 导致摄像头崩溃的字段记录

Table 2 Field records resulting in crash of camera

设备	字段	描述	个数
Hikvision (DS-2CD7153-E)	CSeq Range	CSeq 中含特殊字符';', Range 字段值字符超长	2 (Range 为已公布)
Tiandy (TC-NC9200S3E-2MP-E-IR30)	URL	Describe 方法中 url = rtsp://192.168.12.2dos:dos5545454 * 18//a * 1024 RTSP/1.0	1
Linksys (WVC54GCA)	Content	Setup 方法中 Content='A' * 120009	1

结束语 本文以 Hikvision,Tiandy 和 Linksys 的某些型号设备为研究对象进行测试,对捕获的样本集去重,并针对 boofuzz 框架的测试用例生成方法进行改进,通过构造 RTSP 协议状态图,使用基于深度优先遍历的方式遍历协议状态图,使在生成测试用例时遍历的路径更有效,生成的有效测试用例占比更高,执行时间更少。在执行测试用例过程中使用 TCP 3 次握手连接的原理来判断测试目标是否崩溃,以使监控模块在对 RTSP 进行模糊测试的过程中记录异常包时的精确性更高,减少了假死情况的记录。另外,在监控过程中实现远程监控,使记录的异常测试用例中没有过多的冗余,以便后续重放时减少因冗余产生的耗时。本文利用提出的模糊测试方法对上述 3 个品牌的监控设备进行测试,其不仅发现了已有漏洞,而且还发现了 3 个之前未被发现的导致摄像头直接崩溃的异常字段,并在得到结果后重放,进行了进一步确认。本文的不足之处在于针对复杂协议,该方法覆盖执行路径的效率较低,下一步需针对构造的协议状态图进行优化,使其适应面更广,效率更高。

参 考 文 献

[1] SHI F Y,FU D S. A survey on analysis and utilization of buffer overflow vulnerability[J]. Journal of Computer Science,2013,40(11):143-146. (in Chinese)
史飞悦,傅德胜. 缓冲区溢出漏洞挖掘分析及利用的研究[J]. 计算机科学,2013,40(11):143-146.

[2] MEI H,WANG Q X,ZHANG L,et al. Analysis of the progress of software technology[J]. Chinese Journal of Computers,2009,32(9):1697-1710.

[3] CHI Q,LUO H,QIAO X D. A survey of vulnerability mining and analysis technology[J]. Computer and Information Technology,2009(Z2):90-92. (in Chinese)
迟强,罗红,乔向东. 漏洞挖掘分析技术综述[J]. 计算机与信息技术,2009(Z2):90-92.

[4] MA R,JI W,HU C,et al. Fuzz testing data generation for network protocol using classification tree[C]// Communications Security Conference. IET,2014:1-5.

[5] MA R,WANG D,HU C,et al. Test data generation for Stateful network protocol fuzzing using a rule-based state machine[J]. Tsinghua Science and Technology,2016,21(3):352-360.

[6] KIM S J,JO W Y,SHON T. A novel vulnerability analysis approach to generate fuzzing test case in industrial control systems [C]// IEEE Information Technology, Networking, Electronic and Automation Control Conference. IEEE,2016:566-570.

[7] HAN X,WEN Q,ZHANG Z. A mutation-based fuzz testing approach for network protocol vulnerability detection[C]// International Conference on Computer Science and Network Technology. IEEE,2013:1018-1022.

[8] LI H,WANG S,ZHANG B,et al. Network protocol security testing based on fuzz[C]// International Conference on Computer Science and Network Technology. IEEE,2015:955-958.

[9] WANG W,SUN H,ZENG Q. SeededFuzz:Selecting and Generating Seeds for Directed Fuzzing[C]// International Symposium on Theoretical Aspects of Software Engineering. IEEE,2016:49-56.

[10] MA R,REN S,MA K,et al. Semi-valid fuzz testing case generation for stateful network protocol[J]. Tsinghua Science & Technology,2017,22(5):458-468.

[11] 胡昌振,马锐,纪文东,et al. Case generation method for semi-legalized fuzz test of network protocol based on finite-state machine:CN 105095075 A[P]. 2015.

[12] NARAYAN J,SHUKLA S K,CLANCY T C. A Survey of Automatic Protocol Reverse Engineering Tools[J]. Acm Computing Surveys,2015,48(3):1-26.

[13] MA R,ZHU T,HU C,et al. SulleyEX:A Fuzzer for Stateful Network Protocol[M]// Network and Systems Security. 2017:359-372.

[14] 龚波,冯军. 模糊测试——强制性安全漏洞发掘[M]. 北京:机械工业出版社,2009.

[15] Sulley[EB/OL]. (2013-06-11)[2016-10-18]. http://github.com/OpenRCE/sulley.

[16] RFC2326. RTSP Protocol[Z/OL]. (2009-08-10). https://tools.ietf.org/html/rfc2326.

(上接第 155 页)

[8] ZHU B,WANG J,ZENG G. A non-integral-Q Algorithm for RFID system in anti-collision[C]// International Conference on Control,Automation and Robotics. IEEE,2016:374-377.

[9] FU Y,QIAN Z G,MENG J,et al. FSA Anti-collision Algorithm Based on Continuous Slot Prediction[J]. ACTA Electronica Sinica,2016,44(9):2081-2086. (in Chinese)
付钰,钱志鸿,孟婕,等. 基于连续时隙预测的帧时隙 Aloha 防碰撞算法[J]. 电子学报,2016,44(9):2081-2086.

[10] ZHANG X J,MA J F,CHEN Y J,et al. An Enhanced Q Parameter Hybrid Anticollision Algorithm[J]. Computer Technology and Development,2013(8):47-51. (in Chinese)
张学军,马军飞,陈彦君. 增强型 Q 参数混合防碰撞算法[J]. 计算机技术与发展,2013(8):47-51.

[11] PAN S C,WANG H Q,ZHANG X H,et al. Research in anti-collision algorithm of block ALOHA in static environment[J]. Computer Engineering and Applications,2016,52(20):114-117. (in Chinese)
潘思丞,王慧琴,张小红,等. 静态环境中分组 ALOHA 防碰撞算法研究[J]. 计算机工程与应用,2016,52(20):114-117.

[12] LI J X,FENG X,SHI W G,et al. RFID anti-collision algorithm based on dynamic Q method[J]. Journal of Tianjin Polytechnic University,2015,34(6):55-60. (in Chinese)
李建雄,冯鑫,史伟光,等. 基于动态 Q 值的 RFID 防碰撞算法[J]. 天津工业大学学报,2015,34(6):55-60.

[13] SCHOUTE F C. Dynamic Frame Length ALOHA[J]. Mobile Communications,1983,31(4):565-568.