基于SNORT的工控入侵检测系统设计

廖旭金　王秀英
天津中德应用技术大学，中国·天津　300350

【摘　要】本文对SNORT的体系结构、工作流程进行了简要介绍，并通过改进包解码引擎和设计自定义的预处理器插件来扩展SNORT的功能，使SNORT能解析与识别工业控制系统的常用协议PROFINET，实现对PROFINET IO通信系统的保护。

【关键词】SNORT；工业控制系统；入侵检测；PROFINET

Design of Industrial Control Intrusion Detection System Based on SNORT

Liao Xujin, Wang Xiuying
Tianjin Sino-German University of Applied Sciences 300350

[Abstract]This paper briefly introduces the architecture and workflow of SNORT, and expands the function of SNORT by improving the package decoding engine and designing the custom preprocessor plugin, so that SNORT can analyze and identify the common protocol PROFINET of industrial control system, and realize the protection of PROFINET IO communication system.

[Keyword]SNORT£»industrial control system£»Invasion detection£»PROFINET
【课题】本文课题来源：天津中德应用技术大学科研项目（编号：zdkt2020-006）。

本项目搭建基于PROFINET的工业控制系统网络，以开源的SNORT框架作为入侵检测系统，并利用Wireshark获取PROFINET协议的数据包，通过对其源码的分析与处理，获取关键字段信息，设计包解码插件和预处理器插件，扩展到原有系统中，使其应用到基于PROFINET协议的工业控制系统中,以软件的方式实现入侵检测系统的功能，并模拟工业现场中可能发生的入侵场景，如拒绝服务攻击、ARP欺骗攻击等，验证SNORT系统能否检测攻击行为，并产生报警，利用开源软件Truffle-Hog对结果进行图形化实时显示。

1　SNORT的体系结构

SNORT入侵检测系统由包捕获器、包解码器、预处理器、检测引擎、报警输出、日志文件或数据库系统、规则库等模块组成，各模块分工合作，配合运行，共同组成了功能强大的SNORT系统。其中包解码器、预处理器、检测引擎、报警输出是SNORT系统的基本模块，以插件形式存在，有很强的可扩展性。

（1）包捕获器：SNORT本身没有捕获数据包的模块，只能借助Linux的捕包函数库Libpcap或Windows的Winpcap直接从网卡获得数据包，并将捕获的数据包传递给包解码器。

（2）包解码器：用于将包捕获器传递过来的原始数据包进行解码，并将解码后的数据包存放到SNORT定义的结构体中，方便后续的预处理器和检测引擎处理。

（3）预处理器：用于将包解码器解码后的数据包进行预处理操作，使之规范化，方便检测引擎的检测。另外，预处理器也可以检测数据包是否有明显的错误，如果有则直接报警输出。预处理器是插件形式存在的，SNORT中内置了一些常用的预处理器，用户可以根据需要有选择地使用，也可根据添加自定义的预处理器。

（4）检测引擎：是SNORT系统最核心的模块，通过规则库判断经过解码和预处理之后的数据包是否存在入侵行为。需在SNORT启动前提前配置好规则库。

（5）报警输出：若数据包被检测出错误或非法，则由此模块进行记录并报警。报警信息可以日志文件形式的保存，也可以保存在MYSQL、ORACLE等数据库系统中。用户也可以定制输出插件，将检测后的数据进行直观展示。

（6）规则库：是检测引擎对数据包判断是否非法的依据，在SNORT启动时会对系统中的规则进行初始化，初始化好的规则被加载到一个三维链表中。

（7）日志文件或数据库：用于记录报警信息。

2　SNORT的工作流程

SNORT系统的启动流程如图1所示：

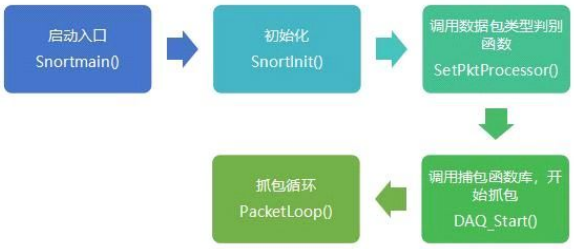

图1 SNORT系统的启动流程

从上述流程可以看出，SNORT是通过循环，依次对捕获到的每个数据包进行解码、预处理和规则匹配的。

SNORT利用捕包库函数能捕获到了数据链路层的所有流量，包解码器按照数据包的类型逐层向上调用相应的解码函数解码。解码完毕后，数据包便会被识别出相应的协议类型，该过程是在数据链路层完成的。接下来，系统又会调用网络层解码函数，对数据链路层解码完毕后的数据包做相似的解码工作。最后，调用传输层解码函数对数据包进行最后的解码，并把解码后的数据包存放到系统定义好的数据结构中供预处理器来处理[1]。

通过对SNORT的架构和工作流程的分析可知，SNORT能够调用不同的解码函数对捕获的数据包进行解析，根据检测规则对异

常协议进行识别与报警。但目前SNORT支持检测的协议类型大多都集中在网络层和传输层，对于工业控制系统的常用协议如PROFINET还无法实现解析与检测。

PROFINET是工业控制系统中的重要协议，其中的PROFINET-RT协议和PROFINET-DCP协议是最重要的、基于以太网层的协议，使用MAC地址寻址，应用层直接在数据链路层上方。ROFINET和其他的工业通信协议一样存在安全性差、缺乏安全认证和授权保护等安全隐患。为了让SNORT能够实现对PROFINET通信系统的保护，本文采用了改进SNORT的包解码引擎和自定义预处理器插件的方法，使SNORT能对PROFINET数据包进行解析与识别。

3 改进包解码引擎

在SNORT中，用于以太网数据包解码的函数是DecodeEthPkt()，为了能解析PROFINET信号，需在DecodeEthPkt()函数中加入对它的识别。具体流程如下：

(1)修改src目录中的detect.h文件，添加一个常量，用于保存PROFINET的包头信息，数据包的EtherType字段等于0x8892，说明为PROFINET协议，同时在此文件中添加PROFINET协议的解码函数DecodeProfinet()的声明。

(2)修改src目录中的detect.c文件，添加PROFINET的解码函数DecodeProfinet()的定义，将捕获的数据包保存到SNORT的Packet数据结构中，包括包头信息、数据信息、数据包长度等，同时记录协议信息和当前解码的层次信息。Packet数据结构中保存了SNORT正常工作需要用到的关键信息，包括原始数据包中各字段的信息、数据包经过协议解析后所得的信息字段和标识字段等。

(3)在detect.c文件的DecodeEthPkt()函数的EtherType协议类型判断部分加入对PROFINET协议的判断，如果EtherType字段的值为0x8892，则调用解码函数DecodeProfinet()。

这样就完成了对PROFINET协议的数据捕获。接下来还需要再设计针对PROFINET协议的检测模版和预处理器用来识别入侵行为。SNORT的预处理器可在检测引擎之前对数据报文进行预处理，实现检测直接报警输出，也可以进行重组、转码等操作。这样就可以实现SNORT对PROFINET协议的解析与识别。

4 自定义PROFINET的预处理器插件

SNORT使用了大量的插件，这是SNORT的一个主要特征，也是SNORT模块化结构设计的标志。用户可以灵活选择要使用哪些功能，开发人员可以开发符合自己需求的第三方插件，扩展SNORT的功能。

SNORT所有的预处理器插件都保存在src\preprocessors目录中，要自定义SNORT预处理器插件，需要用到SNORT源码templates目录下的spp_template.c和spp_template.h这两个模板文件。将spp_template.c拷贝到src\preprocessors目录中，并重命名为spp_profinet.c。

SNORT通过链表来实现插件，ProprocessKeywordlist是预处理器插件链表的节点，封装了每一个预处理器插件的相关信息。每个插件都需要通过注册、初始化和调用三个步骤来完成，每个步骤对应一个功能函数[2]。

(1)注册：SNORT启动时，主函数调用plugbase.c中的RegisterPreprocessors()函数进行预处理器插件的初始化，将每个预处理器对应的ProprocessKeywordlist节点通过next链接起来，生成一个链表。此时需要修改plugbase.c文件，包含插件的头文件spp_profinet.h，并将插件安装函数SetupProfinet()插入到RegisterPreprocessors()函数中；修改spp_profinet.h文件，定义插件安装函数原型void SetupProfinet()；修改spp_profinet.c文件，根据模板中的SetupTemplate()函数定义SetupProfinet()，将自定义的预处理器名Profinet与对应的初始化函数ProfinetInit()进行简单的封装。

(2)初始化：读取配置文件SNORT.conf中关于预处理器插件的配置，对需要使用的预处理器进行初始化，分析配置参数，填充相应的数据结构，将实际功能函数填充至实际执行的预处理器链表。在这里需要在spp_profinet.c文件中，根据模板中的TemplateInit()函数定义插件初始化函数ProfinetInit()。

(3)调用：在数据包进入检测引擎之前，会遍历预处理器插件链表，对数据包进行预处理。预处理器的功能函数ProfinetFuction()是自定义的PROFINET预处理器的核心，在该函数中需要对PROFINET协议进行识别、解析和存储。流程如下：

(1)SNORT捕获数据包经过包解码器解码，如果数据包的EtherType字段等于0x8892，说明为PROFINET协议[3]。

(2)进入预处理器模块，对数据包各字段进行解析，保存在树形结构中。

(3)如果数据包的源和目的MAC地址不是指定的MAC地址，说明产生非法连接。

(4)如果数据包的长度、Service Type、Service ID、Block的值不符合规范，说明是非标准协议数据包。

(5)如果数据包的目的MAC地址为01:0e:cf:00:00:00，Block值为0xff，说明存在扫描威胁。

(6)如果数据包的Service Type值为0x01，Service ID值为0x05且源MAC地址不在指定的地址列表中，b_op值为0x01或0x02，b_s_op值为0x02，说明存在潜在的拒绝服务攻击。

5 系统测试

为了验证SNORT入侵检测系统的有效性，本项目搭建相应的实验环境。其中SNORT入侵检测系统是安装了SNORT2.9.10软件和Truffle-Hog软件（用于将结果进行图形化显示）的PC机，模拟攻击设备是安装了模拟攻击软件ProfinetExplorer的PC机，工业交换机是西门子SCALANCE XM408-8C，IO控制器是西门子S7-1200，IO设备是西门子ET200SP。

实验证明，SNORT入侵检测系统可以对网络流量进行实时监控，同时在Truffle-Hog软件显示监控到的流量的实时状态，当有非法流量时，SNORT可以产生报警信息，同时在Truffle-Hog软件的实时界面也能进行显示。

结语

入侵检测系统是工业控制系统网络安全防护体系中不可或缺的重要设备，目前用于工业控制系统的入侵检测系统大多是硬件和软件的结合，价格昂贵，本文通过对开源的SNORT框架的分析与改进，实现SNORT能够对PROFINET系统进行一定的防护，以软件的方式实现入侵检测系统的功能。

参考文献：

[1]刘红阳.基于SNORT的工业控制系统入侵检测系统设计[D].北方工业大学,2019.

[2]冯凯.工业控制网络入侵检测系统的设计与实现[D].郑州大学,2018.

[3]王小龙.基于SNORT的网络入侵检测系统的设计与实现[D].西安电子科技大学,2017.