网络安全和信息化

Cybersecurity & Informatization

ISSN 2096-2215
CN 10-1416/TN
2022年
12月刊
第12期
/总第80期/

主办单位：中国电子信息产业发展研究院　赛迪工业和信息化研究院(集团)有限公司

2022安全样板工程

——工业互联网安全、车联网安全篇

ISSN 2096-2215

工控系统组成介绍

工控系统（ICS）由多种自动化控制组件和实时数据采集、监测的过程控制组件共同组成，其组件包括数据采集与监控系统（SCADA）、分布式控制系统（DCS）、可编程逻辑控制器（PLC）、远程终端（RTU）、智能电子设备（IED），以及保证各部件通信的接口技术。工控系统的杀毒软件安装及升级更新，操作系统、操作行为、设备维修时笔记本电脑的随意接入，控制终端、服务器、管理终端、网络设备故障等都存在许许多多潜在的威胁风险。

工控系统层次模型

工控系统层次结构是由工业过程控制部件与计算机设备组成的自动控制系统，从上到下有5个层次，依次为企业资源层、生产管理层、过程监控层、现场控制层和现场设备层。不同层级实时性要求不同。

企业资源层：包括整个企业或区域的企业网站管理系统、财务系统、经营管理系统和维护的基础设施组件，用于为员工及企业决策层提供运行决策手段和企业进行宣传。

生产管理层：包括最终产品的管理工作流程的相关功能、生产进程、生产调度、生产过程中的可靠性保障。

过程监控层：主要包括监控服务器与人机界面系统功能单元，用于对生产过程数据进行采集与监控，并利用人机界面系统实现人机交互。

现场控制层：主要包括各类控制器单元，如何编程集散控制单元、逻辑控制器等，可以用于对各执行设备进行控制。

现场设备层：主要包括各类过程中传感设备与执行设备单元，用于对生产过程进行感知与操作。

工控系统现阶段风险与痛点

近年来工控系统伴随着国家网络安全政策的大力推广而迅速发展，但工控不但要发展也要安全。现阶段工控系统还有很多风险与痛点问题，大体分为以下这些方面：工控系统设计缺乏安全性，工控主机系统版本老旧漏洞多、恶意代码防范能力差，工控系统缺少必要的预警检测手段，工控数据面临被窃听篡改安全风险，工控单位安全管理制度不完善、管理不到位，工控系统信息安全投入不足、人员意识差等，这些都是需要重点解决的问题。

工控系统与等保领域的结合和应用

在现有业务不受影响的前提下，增加工控系统整体安全防御能力，通过等保检查存在的问题，针对问题进行加固整改，提高系统的安全防护能力。下面以远程管理方面为例，对操作方法进行说明。

1. 等保指标要求

当进行远程管理时，应采取必要措施防止鉴别信息在网络传输过程中被窃听。

应采用口令、密码技术、生物技术等两种或两种以上组合的鉴别技术对用户进行身份鉴别，且其中一种鉴别技术至少应使用密码技术来实现。

2. 安全防护要求

应用系统、关键网络设备、关键安全设备、关键主机设备（操作系统）通过可控网络环境进行远程管理。

对于无法在可控网络环境中进行远程管理的，鉴别信息以加密方式在不可控网络环境中传输。

同时，采取多种身份鉴别技术对用户身份信息进行鉴别，例如动态验证码 + 口令以及 U-key+ 口令等。 【下转第111页】

基于大数据的App漏洞分析与挖掘

■ 中国软件评测中心网络空间安全测评工程技术中心 李维娜

编者按：提出了基于大数据架构的 App 漏洞特征分析方法，设计了云环境下基于大数据开发框架的漏洞特征分析系统，给出了基于大数据架构的漏洞特征分析算法的一般流程及分类、聚类、频繁模式等，把漏洞特征分析算法运用到移动互联网 App 产品安全漏洞库中来分析 App 产品的漏洞特征模式，预测 App 的安全问题等。

为贯彻落实工业和信息化部、国家互联网信息办公室、公安部联合印发的《网络产品安全漏洞管理规定》，2021 年 8 月 26 日，工业和信息化部移动互联网 App 产品安全漏洞库发布会暨安全漏洞管理特设工作组成立仪式在京举办，旨在充分发挥移动互联网领域相关企业的技术优势，联合业界知名科研机构、高校、安全企业、网络产品提供者、网络运营者等，全力做好移动互联网 App 产品漏洞收集、认定、修补等工作，提高威胁应对与风险管理能力，保障国家网络安全。漏洞库平台搜集了多个具有相似特征的 App 产品漏洞，分析 App 漏洞数据库成为分析相似漏洞特征的一种重要手段。

大数据技术框架的出现给大量数据的存储分析提供了比较好的架构，利用大数据手段对漏洞库进行多维度漏洞数据分析成为一个有意义的研究方向。为了适应漏洞规模大、漏洞数据量激增的情况，

【上接第110页】 **3. 补救措施**

对于设备采用非加密管理模式，且其非加密通道无法关闭的情况，需加强日常运维管理要求，着重关注运维管理日志。

对于无法使用多种鉴别技术的，远程管理过程中，多次采用同一种鉴别技术进行身份鉴别，且每次鉴别信息不相同，例如两次口令认证措施（两次口令不同）。

采取限定管理地址、绑定管理终端等技术措施，防止鉴别信息被截获，以及攻击者利用鉴别信息登录设备。

结语

加强和落实网络安全等级保护工作是我国各级政府、机关、企事业单位、团体和个人应尽的责任与义务，网络安全等级保护制度经过 20 余年的不断推广、发展和完善，已经形成了较为完备的安全防御体系，无论应对各级各类系统都具有极大的参考价值。

有效利用网络安全等级保护的相关技术标准和技术参考，能够指导工业企业的运维人员和安全人员快速发现和补足安全短板，增强工控系统整体防护水平，提升工控系统防护工作的效率。

利用大数据技术进行漏洞分析势在必行。

相关工作

大数据技术的发展已经产生了基于 Hadoop 的生态系统，常见的组件有 HDFS、MapReduce、Hive、Hbase、Zookeeper、Spark 等。目前，漏洞分析的研究一般有静态和动态分析方法，并从自动化到智能化方向发展，而且呈现大数据应用趋势。本文在分析了国内外漏洞分析系统算法现状的基础上，结合移动互联网 App 产品安全漏洞库平台，设计了基于大数据框架的漏洞分析系统，而后设计了一种漏洞分析算法，并在移动互联网 App 产品安全漏洞库平台中得到应用，实现了对系统的实时监控与分析。

基于大数据的漏洞分析架构设计

移动互联网 App 漏洞库系统平台可以分析包含各种类型的 App 漏洞，如源码安全漏洞、组件安全漏洞、数据安全漏洞、业务逻辑漏洞、服务端安全漏洞等。漏洞系统架构一般包括漏洞采集模块、漏洞存储模块、漏洞分析模块和分析结果展示模块。从图 1 可以看出，各个客户端通过接口把漏洞信息传送到云端数据采集中心，采集中心通过漏洞数据预处理，将格式化漏洞存储到 Hbase 库中，同时为了快速地查询处理建立基于 ElasticSearch 的漏洞索引库。建立 Hadoop 数据处理中心模块，处理结果通过结果分析模块进行可视化展示，同时把有用模式及规律存入模式库。模式库可以作为 Hadoop 数据处理中心的部分输入，指导数据处理中心算法的优化及算法训练。其中，客户端可以定期通过基本的 HTTP 协议传输漏洞协议；数据采集中心通过漏洞的归一化、去重、去噪等规格化数据，然后传输到 Hbase 存储及 ES 服务器建立索引；Hadoop 数据处理中心可以通过频繁模式算法、聚类算法、分类算法等分析漏洞模式等，然后通过一定人工监控及需求分析把有用的结果加入到模式库，模式库可以作为算法训练的结果反作用于数据处理算法。

基于大数据的漏洞分析算法运行在 Hadoop 或 Spark 数据处理中心，为了达到智能漏洞分析识别漏洞模式的目的，可以实施一些数据挖掘相关的算法，如分类、聚类、频繁模式等。

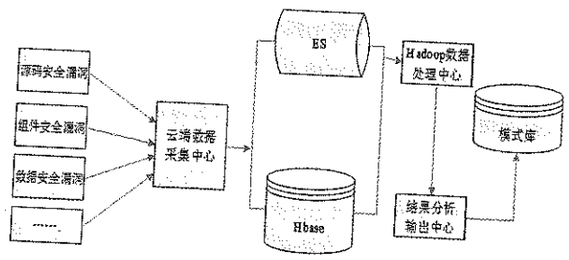

图 1 一种漏洞库中大数据漏洞分析系统架构

基于大数据的漏洞分析算法及实施步骤

1. 基于分布式的漏洞分析算法一般过程

基于分布式的漏洞分析算法一般由单机算法改进而成，在经过漏洞数据采集后，数据存储于漏洞数据库，然后经过数据分片，进入对每个数据分片的数据分析过程。对各个部分模式结果通过一定的规则整合成为模式库。基本过程如图 2 所示。

2. 基于分类算法的漏洞分析算法

常见的分类算法有决策树、贝叶斯分类、支持向量机、基于关联规则的分类，分类的目的是把一些目标数据划分到已知的类别中去。

决策树是一种常用的有监督的分类算法。监督分类即首先给定一定数量的样本，这些样本都包含相关的属性和类别，通过识别这些属性制作一个分类器，当新的包含相关属性的对象到来时，可以利用该分类器把对象划分到对应的类中。

贝叶斯分类是基于统计学的分类算法。该算法是计算待分类的对象出现的条件下各个类别出现的概率，并且把出现最大概率的类别作为待分类对象的类别。支持向量机也是一个基于统计学的分类算法，是一种二分类模型，是求定义在特征空间上间隔最大的线性分类器。

关联规则的分类是基于频繁模式及支持度与置信度的算法。这种算法求的是支持度及置信度大于给定阈值的对象。关联规则反映了给定数据集上属性和值对之间的强关联关系。

分类算法可以用于漏洞分析。首先，可以采用有监督的分类，通过标记漏洞属性，比如漏洞产生时间、漏洞来源客户端、漏洞接口信息、漏洞关联的 App 模块等，标记一定量的样本，把漏洞对应的安全问题级别标记为分类类别，制作分类器。然后，对后面采集的漏洞输入分类器进行分类，基本结构如图 3 所示。

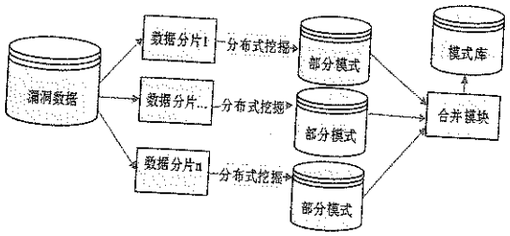

图 2 分布式漏洞分析算法

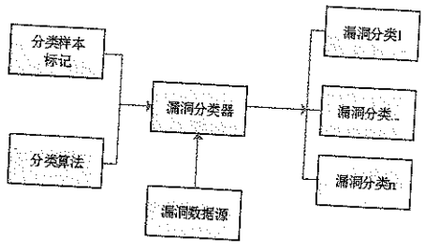

图 3 漏洞数据分类算法模型

3. 基于频繁模式的漏洞分析算法

频繁模式算法也就是求解支持度大于给定阈值的项目的过程。常见的频繁模式算法有基于序列的频繁模式算法，有基于项集的频繁模式算法，也有基于数据流的频繁模式算法。

基于项集的算法的数据来源于事务数据库，每条事务包含的数据也就是完成一次交互所包含的项的集合。通过对事务数据的扫描计算，获取出现频率达到要求的项及项集。常见的有 Apriori、FP-Growth 算法。

基于序列的算法的数据也是来源于事务型数据库，把执行每项事务所包含的项目进行有序排序组成序列，构成序列数据库，然后扫描序列数据库，获取出现频率达到支持度阈值的序列及其子序列。常见的算法有 GSP、Prefix-Span 算法等。

有些数据产生的形态不再是静态型的，而是随着时间无限产生，达到了数据流的形态，此时基于数据流的频繁模式算法也就产生了。一般基于数据流的算法可以采用基于时间的滑动窗口，把数据流分成不同的连续的数据片段，然后分别对这些片段进行序列的频繁率统计，获取支持度大于阈值的序列及子序列。同时，为了达到挖掘结果的连续性，也需要用后续数据片段的挖掘结果更新或合并先前的挖掘结果。

基于频繁模式的漏洞分析算法是很有意义的。

首先，可以按照时序关系，将漏洞数据库中的漏洞转换成漏洞序列或漏洞项集，然后统计项或序列的支持度，获取达到阈值要求的序列、项和子序列。其次，通过分析频繁出现的漏洞信息，可以获取系统中频繁使用的模块或时常出现异常问题的软件模块，有助于调整系统维护的重点及优化系统结构，模型结构如图 4 所示。

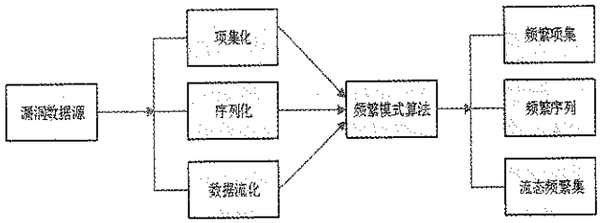

图 4 漏洞数据频繁模式算法模型

4. 基于聚类的漏洞分析算法

聚类算法是对目标数据划分类别的过程，与分类算法相区别的是，聚类算法事先不知道数据可以具有的类别，而是通过计算目标对象与聚类中心的相似度，达到类别之内的对象相似度很大，而类别之间的对象相似度很小的目的。聚类分析一般可以分为基于划分的方法、基于层次的方法、基于密度的方法、基于网格的方法、基于模型的方法等。

基于划分方法的思路是构建目标数据的 K 个分类，把目标数据分配到给定的分类中去。首先随机的给定 K 个划分，把数据随机地分到 K 个类中，然后根据迭代的方法循环计算数据和类中心的距离，最后根据该距离，把数据重新调整到距离小的类中。为了计算类的中心，一般采用基于中心点的方法和基于K均值的方法。基于中心点的算法是把距离中心点最近的对象作为新的类中心；基于 K 均值的方法是把类中的平均值作为类的中心。

基于层次的方法可以分为自底向上和自顶向下的算法。自底向上的算法也就是首先每一个待分类对象作为一类，然后通过合并最近的组或对象，直到满足结束条件，如满足组数目的限定。自顶向下的算法是初始化所有的对象为一类，然后通过迭代进行类的分裂。

基于密度的方法的目的是在给定半径的邻域中必须至少包含一定数据的对象，这样可以发现任意形状的分类。基于网格的方法是把目标对象量化成网格结构。基于模型的方法为每个类定义一个数据模型，把具有最佳模型匹配的对象划分到对应的类中。

利用聚类方法可以在不假定类别各个属性的情况下对日志数据完成分类，然后分析类别内具有相似特征的数据达到识别各个系统模块所具有的相似特征或行为，实现系统各个模块问题的统一识别、定位和优化。分析模型如图 5 所示。

互联网 App 产品安全漏洞分析构想

App 漏洞又可以分为源码安全漏洞、组件安全漏洞、数据安全漏洞以及业务逻辑漏洞等。组件安全漏洞主要记录 Android 四大组件或 Webview 等组件在被调用时未做验证或在调用其他组件时未做验证。漏洞模式挖掘的一种实现方法可以根据 Android 四大组件或 Webview 等组件之间的接口调用行为等进行处理。

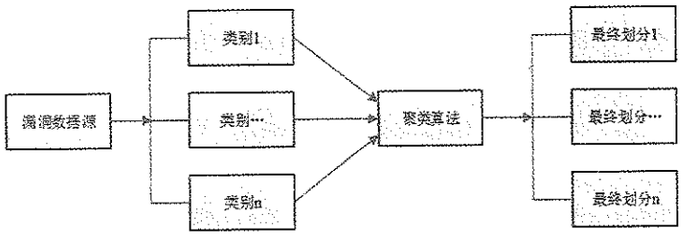

图 5 漏洞数据聚类算法模型

1. 基于 App 组件调用的漏洞模式分析

App 软件漏洞库中收集记录了较大量的组件调用数据，通过组件间的调用行为可以定位和分析 App 漏洞的来源，分析 App 组件之间的调用模式可以挖掘 App 漏洞模式。

组件的调用行为可以用 $S=<S_1,S_2,S_3\ldots,S_n>$ 来表示。其中，S_n 表示组件调用了软件中的某个操作，S 表示了 App 组件调用的操作序列。例如，通过一定时间内的数据采集，可以得到类似表 1 的序列数据库。

表 1 一定时间段内的组件调用序列

序号	调用序列
1	a, f,e, c,d
2	a,c,m,d,n
3	a,q,c,d
4	a,u,c,d

对于表 1，可以通过序列模式算法进行分析，假定最小支持度阈值 min_sup=50%, 项 S_n 的支持度用 Support(S_n) 来表示。可以得到 Support(a)=Support(a)=Support(c)=Supported(<a,c,d>)=Support(<a,c>)=Support(<a,d>)=Support(<c,d>)=100%>50%。从这个例子可以得知 App 经常调用的是 a，c，d 三个操作，这三个操作是具有漏洞 App 的共同特征，表明这三个操作很可能造造成了软件漏洞。因此，可以通过这种方式查找漏洞模式，对现有的其他 App 进行软件漏洞预测。

2. 基于 App 接口调用的安全漏洞问题预测

App 系统内的接口调用是监控 App 产品是否稳定的有效手段，系统的访问量、访问频繁度、系统内部接口的响应时间等都是重要的数据采集点。系统的访问频率一般可以通过记录客户机的 IP 的方式来实现。内部接口的响应时间可以由记录调用的等待时间来确定。

表 2 访问量的漏洞信息示例

序号	客户 IP	访问时间
1	211.155.94.143	2022/2/27 10:10:20.120
2	211.155.94.144	2022/2/27 10:10:20.120
3	211.155.94.145	2022/2/27 10:10:20.120
4	211.155.94.143	2022/2/27 10:10:21.121
5	211.155.94.143	2022/2/27 10:10:21.122
6	211.155.94.143	2022/2/27 10:10:21.123
7	211.155.94.143	2022/2/27 10:10:21.124
8	211.155.94.143	2022/2/27 10:10:21.125

App 漏洞库中采集存储了具有漏洞软件的关键接口调用信息，可以通过预处理得到如表 2 和表 3 中的漏洞示例数据。其中，客户 IP 及接口名称为假定数据。

表 2 表示了客户对系统的访问示例，经过简单的统计可以看出 IP 为 211.155.94.143 的客户每 1 毫秒就访问一次系统，而且连续不停地访问。这样就可以断定系统有被恶意攻击的可能性，即可对访问方式这样的 IP 进行拦截。

表 3 表示了接口的访问及返回时间。可以看出，api/login/userset 接口发生过两次超时等待，api/login/reset 接口的最长等待时间达到了 10 s。而其他接口的响应时间在 1 s 以内。这样就需要重点调整这些接口，进行代码优化、服务器带宽、系统集群等方面的优化。

3. 基于大数据的漏洞分析与预测

系统漏洞是监控 App 产品是否稳定的有效手段。以 App 漏洞库为漏洞信息的来源，按图 1 中的大数据架构的漏洞分析模型为依据，阐述一种基于源代码的漏洞应用实例。

首先，从软件漏洞库中采集源代码漏洞信息，特别是厂商提交的漏洞特征，并保存到云端存储于数据处理中心。

其次，对软件漏洞信息进行预处理，规范源代码中的函数成函数调用序列。

再次，利用基于 Hadoop 或 Spark 的大数据序列模式算法，挖掘频繁函数调用序列或子序列，得到函数序列模式。

最后，把得到的序列模式存储到模式库，辅助后来的软件开发和软件漏洞信息审核与预测。

结语

漏洞特征的分析算法研究、漏洞关联模式挖掘，成为预测分析 App 是否存在漏洞的重要方法。本文从大数据漏洞算法分析的角度，提出了移动 App 漏洞分析、预测和应用的解决方案。首先，分析了基于移动互联网 App 漏洞库、大数据及漏洞分析系统的研究现状及系统构建形式；其次，给出了一个基于 Hadoop、HBase、Spark 以及 ElasticSearch 等技术框架的通用漏洞系统架构模型；再次，给出了分布式漏洞分析算法一般流程及分类、频繁模式、聚类等算法在漏洞系统中的应用；最后，从移动互联网 App 漏洞库中算法应用的角度给出了 App 漏洞分析的应用流程及实例，展现了漏洞库建设的作用与优势。N

表 3 接口调用及响应示例

序号	接口	请求时间	返回时间
1	api/login/set	2022/2/27 10:10:20.120	2022/2/27 10:10:21.120
2	api/login/reset	2022/2/27 10:10:20.120	2022/2/27 10:10:25.120
3	api/login/userset	2022/2/27 10:10:20.120	超时等待
4	api/login/userset	2022/2/27 10:10:21.121	2022/2/27 10:10:22.121
5	api/login/userset	2022/2/27 10:10:21.122	超时等待
6	api/login/set	2022/2/27 10:10:21.123	2022/2/27 10:10:21.223
7	api/login/reset	2022/2/27 10:10:21.124	2022/2/27 10:10:21.224
8	api/login/reset	2022/2/27 10:10:21.125	2022/2/27 10:10:31.125

勒索软件攻击冲击下的软件供应链安全风险分析

■ 广西壮族自治区信息安全测评中心 零家勇

编者按：探讨了软件供应链安全的常见风险及特征，并从新的角度阐述了在勒索软件攻击与软件供应链安全双重风险下，给企业带来的危害以及应对之策。

在数字化时代，软件几乎无处不在，不管是人们日常消费生活，还是工业企业生产中，作为信息化和数字化重要的载体和表现形式，软件已经渗透进各行各业领域。2021 年，工信部印发的《“十四五”软件和信息技术服务业发展规划》指出，软件是新一代信息技术的灵魂，是数字经济发展的基础，是制造强国、网络强国、数字中国建设的关键支撑。

然而，由软件引发的安全问题，特别是软件供应链安全问题已经愈发突出，并成为制约软件产业发展的重要因素。

软件供应链安全现状

软件供应链涉及从设计、开发、运行、维护等软件全生命周期内一系列环节，其复杂多样性导致其安全问题成为挑战。根据云计算开源产业联盟发布的《软件供应链安全发展洞察报告（2021 年）》对软件供应链安全的定义，软件供应链安全指软件供应链上软件设计与开发的各个阶段中来自本身的编码过程、工具、设备或供应链上游的代码、模块和服务的安全，以及软件交付渠道和使用安全的总和。悬镜安全发布的《软件供应链安全白皮书（2021 年）》关注于软件生命周期包括设计、编码、发布、运营阶段的安全问题。

软件供应链安全问题早在确立软件开发流程时代就已存在。2004 年，微软针对软件开发流程提出著名的软件安全开发生命周期流程，在不同的阶段引入不同的安全措施，以此有针对性的应对软件供应链中的安全风险。此外，微软还从技术和管理两方面提出规范性措施，并发布多种安全测试工具及相关运营管理规范，其前瞻性影响至今。

一般来说，软件供应链面临的主要风险包括：在软件开发环节，组件框架存在漏洞、后门，相关的开发工具和环境缺乏安全管控，开发人员安全意识不足等；在软件供应环节，存在外包开发或外采交付缺乏安全管控，发布环境安全风险等；在软件使用环节，软件升级代码漏洞，软件使用过程中出现框架漏洞，运行环境存在安全风险等。

时至今日，随着云计算、人工智能等技术的发展，软件的种类愈加繁多，且应用场景也更加多样。其中，开源和云原生是当前软件供应链发展的两大特征。

开源作为软件快速开发的新方式，已成为当前