安全+
SECURITY
2014/07 总第025
技术版 ▶▶ 与安全人士分享技术心得 Share technique experience with security professionals

★ 本期焦点

浅析信息安全方法中的测、评、估、审、查、量

开展信息系统安全事件管理的方法与过程

移动互联网环境下的DDoS攻防

基于Web访问日志的异常行为检测

扫一扫
关注绿盟科技官方微信

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

2014/07 总第 025

安全+ SECURITY+

目录 CONTENTS

积微者速成

前段时间，系列重大网络安全事件的发生，使业界意识到在战略、组织、系统、能力等各方面存在诸多不足，网络安全可谓百事待兴。如果能识别出基础点和杠杆点，优先予以建设，则能最大化资源利用，收到事半功倍的效果。

通常，安全事件的代价有以下几项：应急响应和公共关系费用，律师费，IT 系统体检、取证调查费用，来自司法机构和主管行业 / 协会等的罚款和问责，各种用户通知更新费用，另外，还有其它多种非直接的费用，例如 IT 系统清理、数据丢失带来的知识产权和商誉的损失等。这些加在一起，是企业评估安全投入是否合理合算的重要依据。

简而言之，如果某类安全事件发生后的代价很低，那么就没有必要和理由投入过高的资源去预防或阻止它。相反，如果某类安全事件发生后的代价及其高昂，理性的决策者一定会投入相对应的安全资源而将其降低到可以接受的水平。

另外，网络安全具有外在性（externality）。换句话说，个体在网络安全上的成效和意义超出该个体自身的利益，对个体之外群体的利益产生影响。与此类似的有传染病、消防、吸烟等。对于具有此类特性的社会事务，通常通过立法来进行约束，取消个体的一些自主权，而进行强制要求。

举例来说，当某电商网站被“拖库”时，不仅该电商的业务受到影响，用户信息被泄露，可能被用以“撞库”，还会导致受影响用户的其它信息和业务受到损失。当没有法律强行要求时，受损电商的理性选择通常是将数据泄露事件隐匿，或大事化小，从而降低自身的“代价”。但“捂盖子”的做法却增加了社会整体的潜在风险。相反，如果有相关法律强制要求围绕安全事件的一系列责任，通过大幅增加“隐匿”安全事件的成本，将其理性选择转向客观透明地披露报告安全事件、通知受影响用户、主动或协助第三方进行事件“根源分析”……这样长期实践的综合效果带来的是社会整体风险的降低，以及有数据支撑的、更为科学的公众安全实践。

网络安全“事件”是客观存在的，只不过有检测到的和没检测到的、披露的和没披露的、有根源分析的和不明所以的之分。

安全威胁在前，安全事件在后。

揭示并根源分析安全事件是分析并消减安全威胁进而阻止事件发生的重要基础手段。

浅析信息安全方法中的测、评、估、审、查、量

安全咨询部　白雷

关键词：安全测评 风险评估 信息安全 安全度量

摘要：本文通过对国内、国外流行的信息安全方法进行梳理与分析，浅析从不同角度对信息安全的理解，对信息系统的安全测试、产品与系统的安全性测评、信息安全风险评估、安全审计、安全检查与信息安全度量等概念进行对比，进一步解析信息安全相关的最佳实践类方法、基本要求类方法、通用准则类方法和合规审计类方法，阐述信息安全的观念与发展方向。

引言

就短短几十年信息安全方法发展的历史来研究，对以下几个词汇的理解运用贯穿于信息安全实践之中，对这些方法的深入解析可以使我们更好地把握信息安全方法的变化趋势与发展态势。本文就个人的认识浅析在信息安全实践中安全方法的认知与运用。

- 测：试验、测试（test）
- 评：评价、比选（evaluate）
- 估：估计、估算（assess）
- 审：审核、审计（audit）
- 查：检查、查验（check、inspect）
- 量：测量、度量（Metric）

一、信息安全的测与评

最初阶段面对信息安全问题，凭直观的思路归纳起来总是沿两个方向：“验对”与“识错”。

- 验对：检验与证明信息安全对象是不是正确的，所有方面都对了就是安全了。
- 识错：测试发现信息安全对象中的错误，并通过各种手段修复错误，以达到安全的状态。

测的思想来源于识错的部分，由于运营的信息系统中的错误、问题与漏洞总是难以完全避免的，这归因于技术的局限、能力的缺失、偶然因素。所以尽可能地发现问题并加以改进是一个非常有效的解决安全问题的方法。典型的测的方法包括：

• 渗透测试 (penetration test)：一般认为渗透测试是通过模拟恶意黑客的攻击方法，来评估计算机网络与应用系统安全的一种技术测试方法。

因为没有百分之百安全的系统，所以识错的方法理论上是一种发散的方法，它的结果总是不会收敛的，这类方法并不具有完备性。一个信息系统，即使只是一个操作系统软件也不可能穷尽所有缺陷。以微软为例，它的任何一款 Windows 软件在其整个生命周期中从没有停止过发行软件的安全补丁，即使像 XP 这样已经下市的软件，安全补丁仍然是大问题。而这是一个普遍的规律。总之，安全识错的工作通常会贯穿信息系统的整个生命周期。

引入评的方法可以弥补测的方法的不完全性，所以我国很早就建立起安全测评方法为核心工作的中国信息安全测评中心（china information technology security evaluation center），它被认为是我国第一个以 CC 为核心方法建立的信息安全测评机构，而 CC 的基本思路是验证信息产品与系统自身的安全性的通用准则。CC（Common Criteria）也即国际标准 ISO15408 information technology – security techniques – evaluation criteria for IT security，它的本质是对保护对象（即 TOE）定义一组通用的安全要求与控制全集，从而形成一个广泛认可的通用集合。实际评的时候要针对某一类产品（如防火墙、服务器、操作系统等）生成一个适用性的指标子集也即指标体系，称为保护轮廓（PP：Protection Profile），PP 通常适用于作为产品或系统的安全标准。对一个特定对象如一台防火墙评的时候是要依据安全指标生成一组实际安全目标（即 ST：security target），ST 可以生成实际的安全测试项目的。

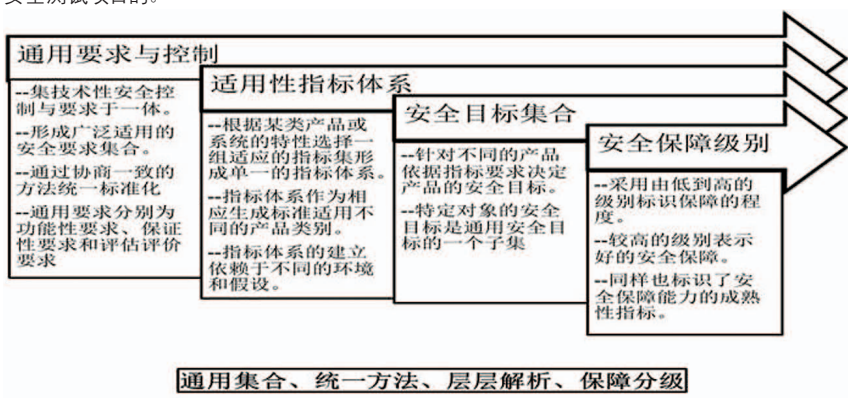


图 1 信息安全的通用准则类方法

评的方法是用于对信息产品或系统进行验对的，而关于评的内容，CC 中分为安全功能要求、安全保证要求和安全评价要求三类，评价的结果分为七个评估保证等级（Evaluation Assurance Level，EAL1-7）。

评的方法不论在国外还是国内都有非常普遍的应用，可以说是一种主流的安全方法论。它解决了通用的集合定义，实现统一评价方法，通过一层层指标要求的解析，对保护对象进行评价保障定级。

但是，CC 主要面向系统的技术方面，至于系统的安全管理，虽有一些安全保证要求，

但很不全面、不系统。而且 CC 在实践中对产品相关的测评的应用多，对系统相关的测评实践一直没有特别好的实践应用，我国最多也就曾出了证券类经营机构信息安全保障指标体系的保护轮廓方面的 PP，而且在 CC 不断修订的过程中也没有进一步加入安全管理对应的指标项目。

二、信息安全风险评估与管理体系

大家都比较认可用 CC 做的技术方面的指标体系，却基本没见有用 CC 做管理体系的，甚至连管理指标都不多，这是什么原因呢?

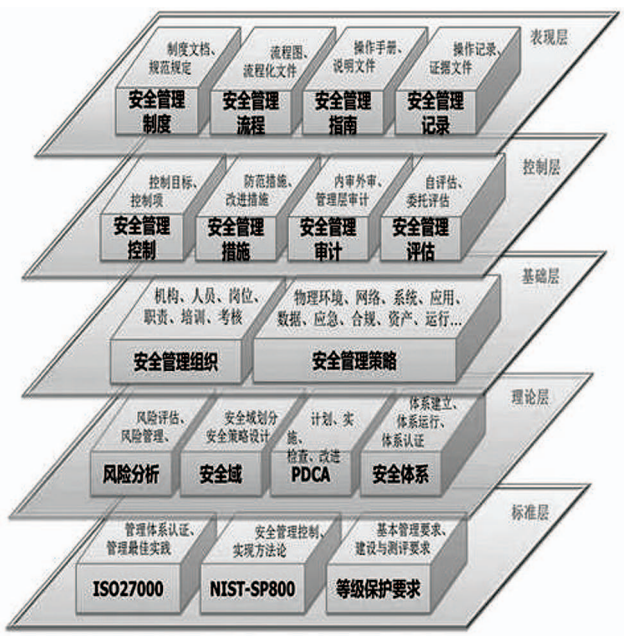


图 2 信息安全管理的范畴

我想既然 CC 是从评价的角度生成指标的思路，在体系形成时单纯地对比指标就变成了缺什么补什么的机械方法，而信息安全管理的范畴是更复杂的方面。如果从纵向分析管理方面的要素，基本可以简单划分为五个层面分析，即标准层面、理论层面、基础层面、控制层面和表现层面。

安全管理体系方面的思想主要是引入了风险的方法形成的，一个很好的讲法是说风险的方法主要是运用于解决现实世界中的不确定性的问题，信息安全中引入风险的概念也是用于不确定性的问题分析的方法。这里的确定性和不确定性可以理解为两种状态：

确定性——系统资产的价值大小、安全漏洞的客观存在、安全威胁与现实的安全攻击行为这些都在现实中有确定性的事与物、域与值。

不确定性——外部的威胁会不会实际发生、漏洞会不会正好被利用了、安全事件即将造成影响的大小程度等。

另一方面，信息安全管理方面的安全措施又可分为两类，一类措施是面向趋于客观性的努力，另一类是面向趋于主观性的努力。客观性努力好理解，通常是客观存在、实实在在的努力，如添加一个防护设备、修补安全漏洞；主观性努力更抽象，如提升安全意识、操作与审计职责分离等，作用于主观层面，说这些措施可以提升安全没有客观直接的证据，只是一些实践经验的总结提炼的结果。

到目前为止，国际上最成功与认可度最高的安全管理的实践应当是最早由英国标准化组织建立的被称为 ISO27001（Information technology-Security techniques-Information security

management systems — Requirements）的一套信息安全管理的体系，它实际上又是一组信息安全管理的最佳实践，实践内容被放在 ISO27002（Information technology-Security techniques -Code of practice for information security management）标准中，包括有 11 大类、39 个控制目标和 133 项控制。但最佳实践的问题是它并不是万能灵药，不能照搬照抄，一旦有了最佳实践，就产生了如何使用最佳实践的问题，就是如何选用最佳实践中的这些控制措施，它适用不适用你的系统？有没有必要用某个措施？这就有必要引入一类方法评估各自状况，选择基于个性化所需的安全控制措施，由此，就产生了估的方法。这种安全管理体系建立的思路方法大致如下：

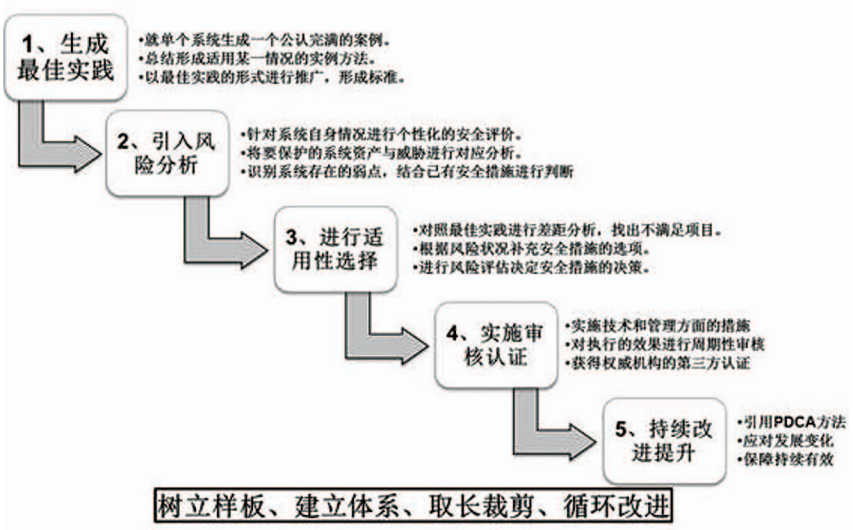


图 3 信息安全的最佳实践类方法

由于信息系统风险是由来自外部的威胁利用系统自身的脆弱性对系统资产造成破坏后产生损失而形成的，所以风险评估（Risk Assessment）基本上是基于对系统资产的现状、面临的安全威胁与系统自身的脆弱性三方面的因素进行分析与计算的风险评估既可以使用定量方法也可以使用定性方法，而且由于有不确定性的因素存在，纯定量的评估方法几乎没有人使用，业界通常在用的都是定性与定量相结合的方法进行评估。

风险评估的结果其实就是一类专家观点，它只有相对的意义而并没有绝对意义，对于系统而言，评估的结果意义在于它表明了针对某个资产的一种风险可能大于另一种风险，或某个资产面临的风险比其它资产高。而风险管理的思路是用于控制相关风险的，手段就是合理选择对抗风险的控制措施，当然也可能考虑转移风险或接受一些较低的风险，从而表明系统可承受一些损失。

估的方法用一种风险的方法推广了一套最佳实践的安全管理体系建设方法，当然另外也引入戴明环 PDCA 这样循环改进的方法论。特别是要解决体系认证的技术性问题，还引入了对体系的认证审核机制，就是审计（Audit）的方法。

三、信息安全审计

估的方法在实质上是识重点的资产、抓紧急的安全威胁和找薄弱环节的脆弱性的过程，理论很容易被接受，但实践起来依据

专业化的知识与能力，特别是一些主观性的结论并没有绝对的说服力的。那么有没有一种方法不是从参照其它成果出发，找出自身安全发展道路的呢？这可能要说说审这种方法了。

审的原理是从系统最终要达到什么结果出发，倒推出需要完成什么样的安全指标和实施什么样的安全控制，这样我们做的每一个努力都是有明确目的的，反而不像风险评估那样要依靠别人的实践经验成果。审的思路最清晰的可以借鉴 IT 治理相关控制目标的形成，其中 COBIT（Control Objectives for Information and related Technology）从业务目标出发定义了 IT 相关控制目标，而信息安全的控制可以看做其中一个目标子集。

图 4 信息安全合规审计类方法

审的方法是查结果、看记录、验证据，信息安全中的审计大致是形成一个安全控制框架，梳理出一组安全控制目标，针对达成安全控制目标的活动进行分析，其内在的逻辑是：如果 IT 过程中所有活动的关键绩效指标都可以很好地执行，则可以保证安全过程目标的达成；如果组织 IT 业务中所有 IT 过程的关键目标指标都很好地完成，则可以保证 IT 目标的达成；如果业务目标中 IT 目标中的关键成功因素可以满足，则对业务目标的最终达成是有保障意义的。

审的方法国内、国外都很流行，所以出现了很多的审核机构、控制手册、审核员 LA（Leader Auditor），看来审核的意义对一个组织的确非常的重要。

四、信息安全检查

应当说解决单个系统的信息安全问题的方法与解决一个群体的信息安全问题的方法是有区别的，单个系统可以测，可以评，也可以估，但如果你拥有成千上万个系统，运用什么方法保护它们的信息安全呢？这正是政府所面对的信息系统安全问题，一旦中国要对十几万在册的重要信息系统进行保护时，方法就变得非常重要了，这就促使产生了一类基线防护的方法，基本上可以总结为分区域、按等级、分层次、沿威胁路径的纵深防御的思想方法。这种方法体现为四个阶段：建立基线、划分等级、实施保护、评估检查。

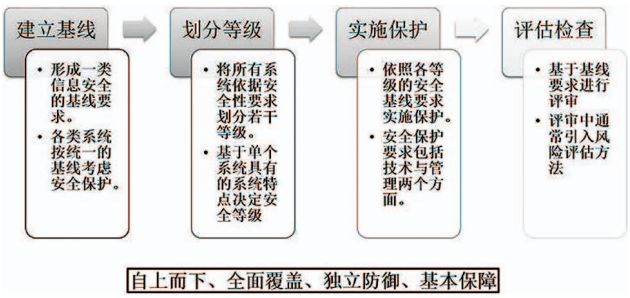


图 5 信息安全的基本要求类方法

检查应当说是一类信息安全工作的方法，现在的信息安全实践中通常是以订立一类信息安全的基线要求开始的，检查的目的是为让安全工作符合相关的要求，并且运用各种手段，依据不同的评价指标进行评判。检查一般由主管单位发起，是一种监督管理工作的体现。国家的等级保护建设类的安全工作就体现了检查的重要思想。

五、信息安全度量

一直以来，信息安全工作的绩效问题也是很受关注的，其中关键还是解决如何度量与评价的问题，这里可以参考一个其它方面的例子。

企业管理中开会如果过多、没有控制是会损失工作效率，可解决的方法却不是很多，而比较有名的韩国三星公司对开会这件事就开始使用度量的原理做事。作为商业公司，所有企业行为都一定是为提升绩效为目标的，但如何提高是有讲究的。西方有句名言，你不能改进你不能测量的东西，所以三星的想法就是量化——准备、主题、纪律、议程、结果、训练、时间、记录、追踪，围绕一系列可以进行量化的维度展开。这已经被认为是三星管理的一个成功实践了。

三星这种想法对企业管理是有意义的，美国有个研究绩效的大牌专家叫詹姆斯·哈林顿，他就曾说：量化管理是第一步，它导致控制，并最终实现改进。如果你不能量化某些事情，你就不能理解它；如果你不能理解它，你就不能控制它；如果你不能控制它，

图 6 msm.mitre.org : “making security measurable”

你就不能改进它。

看来这种方法也可以在其它领域推广使用，比如信息安全。有个美国的工程师叫 Bill Curtis，是个搞软件的，他的思路是：不能理解，就无法管理；不能度量，就无法理解；不能建模，就无法度量；不能设想，就无法建模。

由此看来，信息安全度量是为了更好地评价与改进信息安全，它应该放在信息安全的相对成熟阶段考虑，认清态势、把握尺度、持续改进是它的重要目标。

安全度量方面 MIT 已经有持续很多年的研究项目。

从 MIT 的相关研究网站这几年来的路线看，度量做起来并不简单，它需要定义对系统的各方面的描述语言与格式定义（如 OVAL、MAEC……），需要各方面的知识库（如 CVE、CWE、CCE……），需要用例，还需要标准化的度量过程（如 SP800-51、SP800-53a、CC……）。要做到对系统的软硬件、资产、事件、互联网威胁等等的完整的安全度量，还有很长的路要走。

六、信息安全方法总结

用最朴素的方式总结这几类信息安全方法可以大致有如下的结论与思考：

测——解决有没有的问题，信息安全的方法先测试有没有安全问题，有没有安全漏洞，有没有软件缺陷，这是一类最为直观并且有效的信息安全工作方法。

评——评价好不好的程度，安全措施好不好，安全防护程度高不高，安全管理制度全面不全面，系统不系统。

估——考虑适不适合使用，考虑风险的大小，适合的安全措施与安全防护，选择适当的安全管理控制。

审——查验用没用的结果，安全保护的结果如何，安全防护有没有作用，安全措施有效果还是没有效果，企业整体的安全目标有没有最终达成。

查——检查安全工作做没做，安全要求落实没落实，安全组织、安全技术、安全管理工作开展没开展，是不是按上级要求开展的。

量——测度能不能可持续的提升，能不能不断改进组织的信息安全，使信息安全随需应变，步入良性发展的轨道。

而且，这六类基本方法刻画了信息安全发展的一个简单路线图，发现系统中信息安全漏洞与问题、给当前系统现状以客观评价、通过风险方法选择适用的安全管理与控制、审核各种措施的效果与作用、检查各项安全工作落实情况、最后通过度量方式使当前安全状态可持续地不断提升与发展，这就是我们当前可以看到的信息安全解决之道。

简而言之，测解决技术手段问题，评面向技术控制措施，估对管理控制进行风险分析判断，审实施 IT 安全治理，查体现工作监督管控，量感知把握态势。而且随着信息化的发展，信息安全方法也不是一成不变的，实践中又通常会运用评审、估量等不同的信息安全方法保障信息系统的安全。

后记

读到这里，对信息安全方法感觉有兴趣的可能就会想到，信息安全实际是如何做的？这方面可以期待笔者的另一篇短文《基于标准的控制与基于控制的安全标准建立》，其中讲到信息安全标准的发展与如何建立信息安全相关标准的实用方法，并且会分析到如何围绕“保护对象”、“安全目标”和“控制措施”三者来处理信息安全问题。

开展信息系统安全事件管理的方法与过程

广州分公司　俞琛

关键词：安全事件 监控 响应 审计

摘要：核心信息系统上线运行后一旦发生信息安全事故，对于企业而言很可能是一场灾难。信息系统安全事件管理包含安全事件策略、监控、响应、审计。本文阐述信息系统的安全事件管理工作的开展方法和过程。

引言

随着信息化的深入发展，企业各项业务对信息系统依赖程度逐渐提高，企业自建信息系统功能愈加复杂，系统用户众多，信息系统存储的数据越来越敏感。企业核心信息系统上线运行后一旦发生信息安全事故，对于企业而言很可能是一场灾难。

ITIL Version3 定义信息安全事件(Event) 是指任何可察觉和可识别的、对 IT 基础设施管理或者 IT 服务造成影响和背离的重要现象，侧重于日常监控；信息安全事故（Incident）是指对一项 IT 服务或一项 IT 服务质量减少的非计划中断，侧重于事故发生后的快速响应和恢复的流程。

企业如何做到在海量的安全事件信息中消除过多误报信息，让安全运维人员将注意力集中在真正的威胁和攻击上，避免分析麻痹，尽早识别单一、孤立的安全事件的背后隐忧，防微杜渐，防止信息安全事故的发生，这是企业安全运维人员需要考虑的一个问题。美国政府已制定了如何应对信息安全事件的重要指南。NIST-SP800-61：计算机安全事件处理指南，可以用于应对信息系统安全事件管理的参考材料。

信息系统的安全事件管理包含威胁分析与预警、安全状态和事件的监控、安全事件或事故的响应以及发现不合规、存在异常的操作的日志审计。这些任务主要可通过开展安全事件的监控、响应、审计和制定相应的安全策略共同完成。

本文将讨论如何从宏观的理论层面落实到安全运维人员可执行的策略、步骤，阐述针对信息系统的安全事件策略、监控、响应、审计的开展方法和过程。

一、安全策略管理

目前，大部分企业已经部署了相当多的安全设施，但众多的安全技术与安全设备的应用在相当程度上加重了系统与安全运维人员的负担。要确保信息系统的安全运行，安全管理和安全意识必须贯穿到业务人员的日常业务中。针对信息系统的安全事件同样需要制定相应的安全策略进行管理。

1.1 需求分析

信息安全管理具有宏观、中观、微观三个层次。比如 ISO/IEC27001 标准是从宏观层面定义了整体信息安全管理框架，建立信息安全管理体系（ISMS）。但在中观如何落实、在微观如何实现上需要结合企业自身的实际环境进行解决。

为了实现安全事件管理目标，体现安全事件监控效果，解决安全事件分级排序、合理开展安全事件审计，需要制定安全策略。

1.2 安全策略组成要素

针对信息系统的安全事件管理，相应的安全策略组成要素包括安全事件监控指标、安全事件分级、安全事件响应预案、安全事件审计方案。

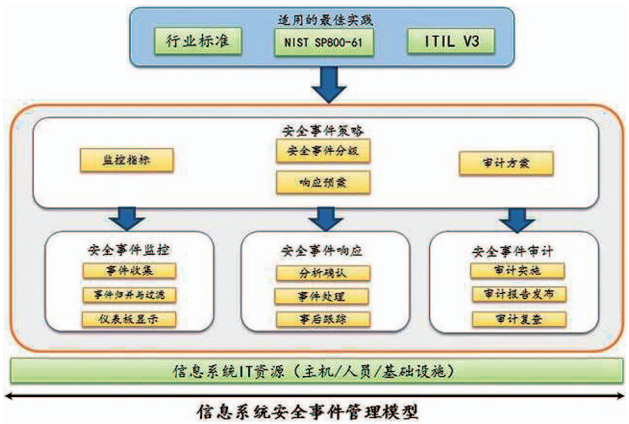


图 1 信息系统安全事件管理模型

1.2.1 监控指标

为了强化监控质量、提升安全防护水平，需要制定可量化的安全事件监控指标。指标内容包括体现内部因素的防护指标和健康指标，考察外部因素的威胁指标，如图 2 所示。

图 2 安全事件监控指标

防护指标是指安全控制手段成功防护的覆盖度（百分比），如防病毒是终端病毒事件处置覆盖度，漏洞扫描、配置核查就是加固完成覆盖度，IPS 是高中风险阻断覆盖度。

健康指标是指保证安全功能的实现，偏向设备健康性监控等日常运维指标，输出数值为健康指标的覆盖度（百分比），如防病毒软件、IPS 版本、漏洞扫描版本是否升级到最新版本。

威胁指标是指安全控制手段检测到具体威胁次数指标，指标输出是具体的数字，与外部威胁有关，如防病毒、IPS 等安全设备的报警次数、漏洞扫描评估到的漏洞数量等。

1.2.2 安全事件分级

根据信息系统重要性以及安全事件对系统可用性、完整性、保密性的影响程度，信息安全事件分极的参考要素包括用户影响、业务影响等，将安全事件分为安全事故（Incident）、严重安全事件（Severe）和一般安全事件（Warning）三个级别。

（1）安全事故（Incident）指：

a) 受影响范围：较多用户无法正常访问。

b) 受影响用户：较多。

c) 业务影响：关键业务非计划中断，无法提供服务。

（2）严重安全事件（Severe）指：

a) 受影响范围：个别信息系统无法正常访问。

b) 受影响用户：较少。

c) 业务影响：短暂业务中断，或出现影响信息系统正常运行的事件，如服务器频繁出现登录错误。

（3）一般安全事件（Warning）指：

a) 受影响范围：无或小部分地区。

b) 受影响用户：个别。

c) 业务影响：无业务中断，存在异常操作行为或违规项，如在敏感时间段进行服务器操作的行为。

1.2.3 响应预案

在任何实际事件中，减轻后果是所有响应机制的核心，系统管理员的主要任务就是让未授权行为停止。那么必须知道，制定了什么安全策略来授权系统管理员这么做。

有些安全事件可以立即发现，例如信息系统服务中断，但有些事件却不是这样，可以隐秘地延续很长时间。对于IT管理者来说，不管是系统管理员、安全运维人员还是首席信息官（CIO），响应措施都应考虑以下问题：

（1）事件正在发生还是已经结束?

（2）犯罪者和其使用的技术是否已被定位?

（3）该事件是否是某项更大规模的行动的一部分?

国家标准与技术研究所（NIST）发布的特别报告书（SP）800-61: 计算机安全事件处理指南，提供了一种事件响应机制：（1）准备；（2）探测与分析；（3）控制、排除与恢复；（4）事后行动。

制定响应预案这一步涉及到“准备”步骤，需确定事件响应所需的资源，拟定事件响应预案的演练方式。做好准备，通常意味着保持系统无漏洞，或至少是尽量减少漏洞。准备工作的重点在于为快速行动建立基础，以便能迅速扩大行动规模，将那些与受影响系统联系不紧的员工也纳入进来。关于网络信息、数据格式、系统策略和操作基准的技术说明对于事件响应非常关键。当威胁已经利用了某个漏洞制造了安全事件或安全事故后，应按照响应机制快速响应。

事件响应预案的演练形式可以采取桌面演练、红蓝对抗演练。定期开展演练有利于响应小组成员明确各自角色分工，熟悉响应流程和内容。采取红蓝对抗有利于模拟实战环境，提高参与人员心里素质，提升应变能力。

1.2.4 审计方案

安全审计应该遵循风险为导向的审查方法，在开展审计时充分评估，熟悉审查对象内部风险状况，制定审计计划和审计目标，合理分配审计资源并确定审计业务范围，运用风险控制方法评价各控制是否有效满足业务设计目标，并测试内部控制的执行有效性。

审计方案需要明确定义审计目标、范围、审计任务，说明审计方法和审计项判定标准。审计方法采用符合性测试与实质性测试相结合的方式。符合性测试是指对被审计系统对应审计项的内部控制设计和执行的有效性进行了解，并对该内部控制是否得到一贯遵循加以审计的过程。实质性测试是指在符合性测试的基础上，为取得直接证据而运用检查、观察、查询、计算、分析性复核等方法，对

已收集的凭据的真实性进行审查，以得出审计结论的过程。审计项判定标准通过判定是否符合企业信息安全管理制度要求、是否存在异常操作情况，给出审计结果为“符合”、“不符合”、“存在异常行为”。

安全审计工作应重点关注被审计信息系统账号管理、权限分配是否符合企业信息安全管理制度要求、系统用户对敏感数据的操作行为是否遵照企业相关流程执行（如针对敏感信息模块的操作都应走相关审批流程，提供操作依据），以及系统后台运维人员是否存在异常维护操作情况（如异常时间登录系统维护、内部系统维护人员对业务应用系统的越权访问、违规操作）。

二、安全事件监控

在企业的信息系统中，存在大量的 IT 资源，这些资源在实际运行中每时每刻都在产生各种类型的事件信息。在这些事件信息中，安全事件是需要安全运维人员重点关注的内容。通过安全事件监控，可以帮助企业积极监控整个组织内的 IT 资源，过滤并关联事件，迅速定位安全威胁，并为安全事件的响应提供支持。

2.1 需求分析

安全运维人员面对信息系统产生的繁杂的登录、访问、操作日志以及外部环境的威胁状态等安全事件信息，需要高效、可量化的安全监控指标，事件收集、归并与过滤方法和监控结果的显示及事件预警机制。

安全监控指标已经在“安全策略管理”中制定。

2.2 监控工作执行过程

2.2.1 事件收集

信息系统的日志收集至少应包含日期、时间、事件类型、操作行为、执行结果（成功、失败）、引起此事件的用户标识信息，通过人工或自动工具从信息系统后台、漏洞扫描设备采集。

外部环境的威胁状态收集至少应包含日期、时间、事件类型、引起此事件的 IP 地址信息，通过 IDS、IPS、防火墙设备采集。

2.2.2 事件过滤与归并

安全事件过滤与归并是可选的过程。过滤即丢弃从设备中提取的原始安全事件信息中监控人员认为不重要的信息，从而减少轻微告警安全事件的干扰。归并是一种组合技术，将选定的时间值或安全事件数量，合并为具有匹配字段值的多条安全事件。通常安全事件归并需要技术工具支持。

对单位时间内发生的大量安全事件，建议按照维护要求和管理部门的考评要求及实际管理情况，对指定信息系统的安全事件信息进行事件归并，也可以通过安全事件严重程度级别、安全事件类型等属性进行归并。

2.2.3 仪表板显示

经过安全事件归并后的数据通过 Excel 电子表格软件或 HTML、PDF 等格式存档，以图文并茂的仪表板、报表形式显示。

三、安全事件响应

3.1 需求分析

对于信息系统的安全而言，我们追求的是防患于未然而不是亡羊补牢，只要有可能，就应该尽可能地区主动防止安全事件的发生。然而，我们不可能预防所有的安全事件。一旦有安全事件发生，我们首先要做的，就是及时响应，将安全事件的影响最小化。

安全事件响应需要制定安全事件响应计划、组建安全事件响应小组、确定团队人员及其角色、定义安全事件分级等内容。

安全事件分级已经在安全策略管理中定义。

3.2 事件响应工作步骤

3.2.1 探测与分析

企业根据当前安全事件描述，结合前期进行过的安全检查、安全监控与审计以及网络状况，进行探测与分析。确定系统运行的基准底线对分析工作非常重要。应该制定系统运行规则，当这些规则遭到破坏时，就应拉响警报。

3.2.2 控制、排除与恢复

相关人员需要在现场或者远程依照不同事件等级进行事件处理。这一步通常比较麻烦，要将受损系统修复并恢复到正常运行状态。这既是科学，也是艺术。系统管理员和其他利益相关方可能只想减轻后果，尽快恢复正常运行。事件处理过程中，要对每个处理的动作进行详细的记录。

3.2.3 事后行动

在安全事件处理完毕、所有系统恢复正常以后，应该针对事件进行分析，集中企业所有相关人员来讨论所发生的安全事件以及得到的经验教训，对现有的一些流程进行重新评审，并对不适宜的环节进行修改。

四、安全事件审计

安全审计系统中收集到的各类日志内容都是正常的访问内容，其审计的目的并不是针对漏洞的利用或者蠕虫的攻击等恶意流量的攻击，而是希望通过业务规则、用户行为等为基准，发现不合规、存在异常的操作。

4.1 需求分析

内部威胁研究机构发现：超过 75% 的内部舞弊犯罪事件是来源于授权用户，而且是使用简单、合法的用户命令引起的。

为保证信息系统用户对敏感数据的操作行为都在审批授权、权限控制和操作审计的安全控制下，需要制定审计工作执行过程，对已收集的凭据的真实性进行审查，以得出审计结论。

4.2 审计工作执行过程

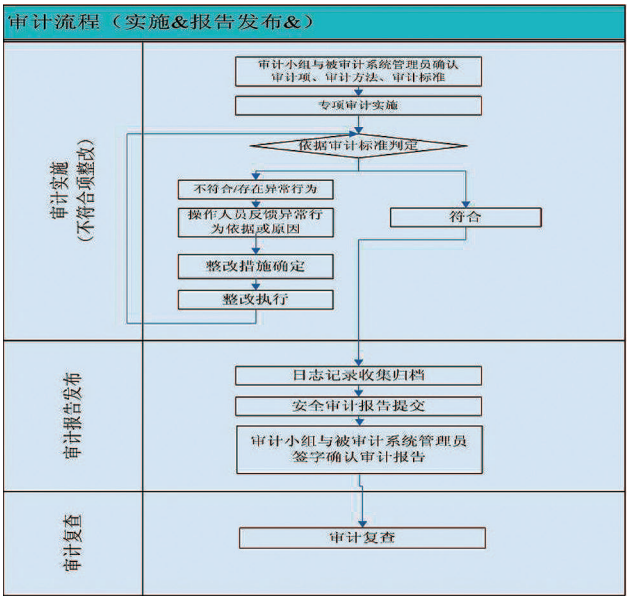


图 3 审计工作执行过程

4.2.1 审计实施

审计实施前，审计小组会与被审计信息系统管理员交流，将依

据每个系统的具体情况，明确审计项的审计方法、审计标准，对于“操作行为”方面的“异常行为”作出定义，确认审计方案及审计计划。审计实施的主要步骤包括：

（1）确定安全审计。申请审计工作主要包括审计原因、内容、范围、重点、必要的升级与纠正、支持数据和审计所需人才物等，并上报审批。

（2）做好审计计划。一个详细完备的审计计划是实施有效审计的关键，包括审计内容的详细描述、关键时间、参与人员和独立机构等。

（3）查阅审计历史。审计中应查阅以前的审计记录，有助于通过对比查找安全漏洞隐患和规程，更好地采取安全防范措施。同时保管好审计相关资源和规章制度等。

（4）确定审计重点和步骤。各类机构都应将主要精力放在审计的重点上，并确定具体的审计步骤和区域，避免审计的延缓或不完全，以免得出令人难以信服的结果。

审计实施后，审计小组应收集被审计信息系统管理员的反馈，督促其对不符合项执行整改。

4.2.2 审计报告发布

审计小组组长负责组织编写审计报告，报告应包括如下内容：

（1）审计的目的和范围
（2）审计依据
（3）审计小组成员、被审计部门及其负责人
（4）审计内容、审计发现、不符合项
（5）审计结论
（6）整改措施建议

审计报告应交由规定的保管责任人存档。应注意后续工作（如对不符合项整改措施执行等内容）产生的相关文件的存档。

4.2.3 审计复查

在审计报告发布并提出整改建议后，审计人员必须获取和评估相关信息，根据与系统管理员协商确定的复查日期，对被审计信息系统是否已完成整改、及时采取恰当的措施做出结论。审计复查工作的关注点如下：

（1）如果被审计信息系统管理员针对审计报告建议而提出的措施已经交换意见，则应该把这些作为“反馈意见”写进最终的审计报告中。

（2）在对审计复查工作做出结论前，如被审计信息系统管理员反馈已经采取措施且提供相关信息，而审计人员对该信息有疑虑时，应该进行恰当的测试或采取其他的程序来确定真实情况和进度。

结论

本文阐述针对信息系统的安全事件策略、监控、响应、审计的开展方法和过程，讨论开展安全事件管理的目的，能够协助 IT 管理者制定符合企业 IT 策略的信息系统安全事件管理策略，让系统管理员、安全运维人员在面对海量安全事件信息时，能够高效、准确地识别真正的威胁和攻击，防止信息安全事故的发生。针对信息系统开展整体安全事件管理，能够帮助企业建久安之势，成长治之业。

参考文献

1. 杨斌《信息安全战——企业信息安全建设之道》

2. 肖岩军《XXX 安全服务项目——安全量化报告》

3.Kim Andreasson《Cybersecurity Public Sector Threats and Responses》

4.NIST Special Publication 800-61 Revision 2 --- Computer Security Incident Handling Guide

基于对抗的智能态势感知预警模型(上)

广州分公司 肖岩军

关键词：早期预警 FISMA CAESARS 框架 态势感知 风险量化 安全大数据

摘要：本文描述了一种基于持续监控保障、态势感知和安全计分技术的早期预警系统的整体模型，并对模型的态势感知、持续监控、风险量化计分部分展开叙述，提出基于对抗的智能态势感知预警模型——“基于攻击行为建模的态势理解方法”和“基于对抗的 APT 攻击推理树态势预警方法”，“基于保障的持续监控模型”和“基于 6sigma 的自动化安全量化模型”，并展示了部分研究成果。本篇为基于对抗的智能态势感知预警模型（上）“基于攻击行为建模的态势理解方法”。

一、引言

随着网络的重要性的逐步加强，网络威胁日益严重，网络安全的地位已经上升到国家层面，网络战逐渐从理论转向实战。2014 年 2 月 27 日，中央网络安全和信息化领导小组成立。该领导小组将着眼国家安全和长远发展，统筹协调涉及经济、政治、文化、社会及军事等各个领域的网络安全和信息化重大问题，研究制定网络安全和信息化发展战略、宏观规划和重大政策，推动国家网络安全和信息化法治建设，不断增强安全保障能力。

作为信息科技高度发达国家的美国，早在 2010 年的《四年防务评估》中就提出：“尽管网络空间是一个人造领域，但它目前已和陆地、海洋、天空和太空等自然领域一样，成为国防部的活动领域。”美国为这第五域起了新名字—Cyberspace。随之而来的就是从战争的角度看待网络安全，态势感知 (Situation Awareness, SA) 等一系列军事名词引用到网络安全领域里，大量的创新围绕在实战或者近乎实战的背景下如何用最小的代价，获得最大的战果。另外，在不对称战争的背景下，美国作为防守方，对于未知对手的攻击情报搜集和对自身的脆弱性的感知能力都是态势感知研究的领域。Endsley 将态势感知定义为“在一定的时空条件下，对环境因素的获取、理解以及对未来状态的预测”，态势感知可简单描述为“始终掌握你周边复杂、动态环境的变化”。 孙子兵法云：“知己知彼，百战不殆”。因此态势感知可以分成两类，一类是基于威胁的态势感知（知彼），另一类是基于脆弱性的态势感知（知己）。而不管是哪种，态势感知强调实战的原则显得越来越重要，2009 年 2 月，美国联邦网络安全专家联盟发布网络

安全基线标准——最重要的二十项关键控制（共识审计指南 CAG，Consensus Audit Guidelines）。其中明确提出“必须让防御了解攻击”。 他们把安全手段分成四类，第一类是快速取胜（Quick Wins），第二类是改善可见性和归因（Improved Visibility and Attribution），第三类是硬化配置和改善信息安全保健（Hardened Configuration and Improved Information Security Hygiene），第四类是提升（Advanced）。

在前期的研究中，我们也发现安全只有实战对抗才有意义，对抗是本文方法的核心。网络安全态势感知分为态势要素的获取、态势理解和态势预测三个部分。因此，本文重点在于阐述一种基于对抗的智能态势感知预警模型，从对抗的角度来进行网络安全态势要素的获取、态势理解和态势预测。由于篇幅限制，本文先重点描述态势理解的部分——“基于专家知识的攻击行为建模态势理解方法”，态势预测部分因为引入 APT 攻击，更加复杂，下期另筹文描述。

二、态势感知遇到安全大数据难题

随着 Web 2.0、云计算、移动互联网的

图 1 基于对抗的智能态势感知预警模型组成

快速发展，随便找一台虚拟化服务器都是双万兆网口上联，动辄几十 G 的出口带宽，形成了安全大数据。根据经验，1 台 IPS 或者 IDS 的 1G 左右的流量 1 天告警量在 20 万以上，而现实中，一个有经验的安全分析师一个小时也只能分析十个告警左右，大量的告警无法处置，往往事件发生后才能追溯。也因此网络态势感知面临更大挑战，各种不同源的告警、各种无用的无关告警造成大量的告警风暴，大量有用的告警被淹没在海量日志告警。美国奥巴马政府去年就已宣布推出“大数据的研究和发展计划”，将大数据上升到国家战略，纳入到 Cyberspace 战略的核心能力。对企业而言，信息安全是为企业信息化服务，而信息化又服务于业务增长，因此只有将安全与业务数据相结合才能为企业带来价值，这一层看似间接却极为必要的关系在大数据时代被无限放大。Gartner 报告指出，最终安全大数据将演化为 IT 商业智能发展趋势的一部分，即结合信息安全情报和 IT 业务数据，以提供更高水平的业务情报。

而现实中多个厂家的告警不统一导致网络安全态势要素的获取都存在问题，更别提态势理解和态势预测。国内外安全厂家的 SOC 失败印证了这一点，每个 SOC 都在不停地接入新的设备，结果接入设备、储存告警花了大量的时间和资源，更重要的态势理解和态势预测却没有人做了。

在态势感知领域里，国外一些 SIEM 系统做的比较好，强调了元语的研究，通过元语的方式将不同厂家的设备告警统一起来，从而实现态势要素的获取和态势理解。但是实际上操作比较复杂，对安全人员要求也非常高。虽然实现了告警的统一，但是现实中的各种误报等因素导致日志量得不到很有效的压缩，而且经常造成告警不可读或者无法呈现。

而采用数学方法也是一种较好的方法。常见的安全大数据分析方法中主要是相关性分析，聚类 / 离群分析等算法（详见我司同事王卫东的《面向安全的大数据分析方法和思路》安全 + 第 24 期），我前期研究也采用过类似的做法。但是鉴于误报、告警混淆等原因造成很大的困扰。如通过相关性分析发现某木马和某 DDoS 工具存在强相关，但是实际分析发现，是扫描触发了这 2 个告警。因此，数学方法虽然能解决问题，但是也遇到较多现实的问题。本文后续举例也证明了这一点。

基于行为方法是我目前认为比较好的方法。在安全大数据方面，IBM 金融大数据分析的方法，在思路有很前的前瞻性和借鉴意义。IBM 团队能够在每秒几千笔的交易信息中找出金融犯罪案件，其核心是一个强大的自然语言规则引擎，风控部门风控专家可以利用规则表、规则树、自然语言、规则流来描述各种风控规则。见图 2，风控部门风控专家通过历史案件进行分析，梳理出规则，然后将规则加载在大数据分析引擎中，从而实现金融犯罪案件的即时监控。可以看出 IBM 的方法也是基于对抗的方法，虽然存在一定的滞后性，但是更加聚焦有效，和我们的方法不谋而合。

而在现实中，公安人员办案也是采用基于行为的分析方法，如发生失窃案件或者银行抢劫案件都是调查案发前几天的周围摄像头，同时封锁周围交通，因为显示犯罪分子都遵循着踩点、下手、逃跑的步骤。虽然下手时可能蒙面，但是踩点时一般会避免别人注意，往往更容易发现不同，也更容易辨认。发现了犯罪分子相貌后，就比较容易让封锁交通的同事从海量人群中把逃跑的犯罪分子找到。在这里，公安使用了规则表（查找 3 天内的行踪可疑的徘徊者。）而踩点 -> 下手 -> 逃跑形成了规则树（或者叫证据链）。

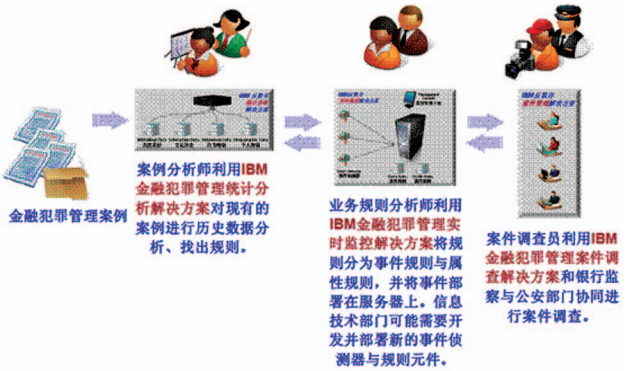


图 2《大数据中的实时精准营销与风险控制》林南晖 IBM

三、基于对抗的智能态势感知预警模型

3.1 整体思路

基于对抗的智能态势感知预警模型的核心思想是通过对抗来获取自然语言行为规则，通过行为规则来解决不同告警源带来的态势要素获取和态势理解难题。从实践来看，这种方法可以较好地跳过告警元语理解的步骤，实现高效的态势理解和精准的态势检测，配合攻击推理树则能更好的实现态势预警。见图 3 整体模型，从下向上，我们能看到，针对我们保护的信息资产的攻击行为（知彼）和评估行为（知己），这些行为会造成各种告警 (warning)，这些告警都是基于特征的，可以用 IDS、IPS、WAF、扫描器等各种设备实现。下一步告警 (warning) 需要形成事件（event），事件是人可读的，可以响应处置的。在这个阶段，我们就用本文的基于对抗的智能态势

感知预警模型 - 基于攻击行为建模的态势理解方法。在本阶段我们能输出可信的事件，给下一步的态势预警做支撑。而态势预警采用证据链或者规则树的方式，进行态势预警。预警可能的结果，同时对当前的态势展开评判，这部分就是下一篇文章的范畴了。

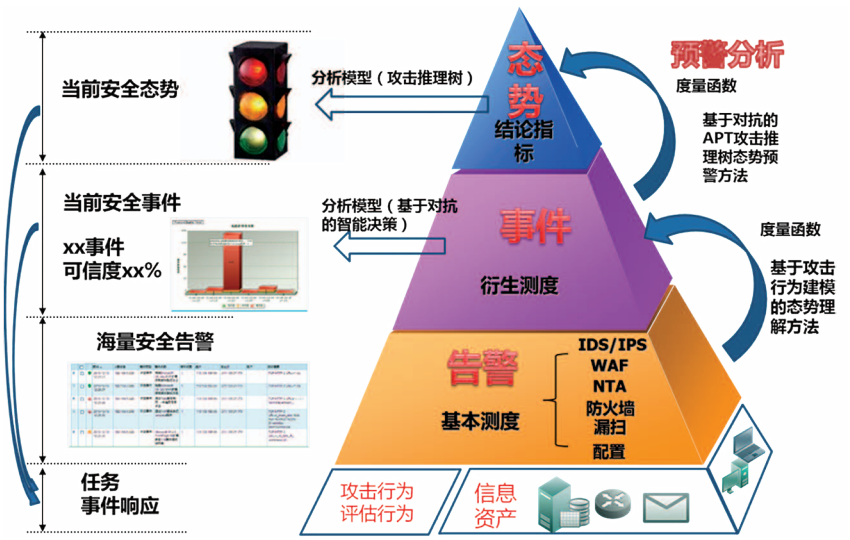


图 3 基于对抗的智能态势感知预警模型全貌

3.2 基于攻击行为建模的态势理解方法

基于攻击行为建模的态势理解方法是一种大数据下分析方法，是在实践中发现的比较好的方法，可以快速进行态势理解，特别是在当前大数据下多源告警泛滥的现实中，这种方法体现的独特性可以跳过安全元语的复杂设定，形成快速的态势理解。实践表明，在海量日志下，通过本方法可以将关注重点关注在我们关注事件上，有效地去除误报和无用告警，把每日的事件告警数量形成人可以处置的水平。当然，这种方法的前提是需要一定经验的安全专家，但是在其他安全大数据分析态势理解方法也同样需要这些专家。具体流程见图 4。

图 4 基于攻击行为建模的态势理解方法步骤

3.2.1 基于专家知识的攻击行为建模

本阶段重点是通过专家知识进行攻击行为建模，来构建相关的攻防场景，核心的核心是基于对抗，安全专家通过其专业知识对常见攻击进行分析，构建常见的攻防场景。

一般而言，我个人不建议做比较复杂的攻击行为建模，一般而言，DDoS 攻击、网络入侵、Web 入侵、暴力破解入侵、僵木蠕和 APT 高级持续性攻击，是主要的攻防场景，可以针对这 6 种事件进行攻击行为建模。至于攻击结果，则可能是信息泄露、系统不可用、代理攻击等不同的结果。针对不同的攻击场景，我们需要采用不同安全设备进行告警采样。当然，如果是成熟客户，有自己的 SIEM 和 SOC，也可以参考这个方法开发自己的安全态势感知平台。

3.2.2 实现攻击告警采样

在完成基于专家知识的攻击行为建模后，安排相应的安全攻防人员进行现场攻击告警采样，在采样的同时，实现不同检测设备的元语关联。相对的，每种检测设备的关注指标是不同的，如在 DDoS 攻击方面，我们关注的流量，1GDDoS 的和 10MDDoS 的显然是不一样的。但是入侵采用入侵检测 / 防护设备来检测 DDoS，由于缺乏流量数据，1G 的 syn flood 攻击和 10M 的 syn flood 攻击，其告警都是 syn 包每秒大于 3K 的告警 xx 多次，因此，我们更建议采用异常流量监测系统来检测 DDoS 攻击。

序号	攻防场景	检测设备	关注告警指标
1	DDoS 攻击	异常流量监测设备 入侵检测 / 防护设备	DDoS 流量 (异常流量监测设备可以直接显示攻击流量。如果是入侵检测设备，需要在采样的同时记录告警次数和实际流量的关系)。
2	网络入侵	入侵检测 / 防护设备	网络、系统、数据库攻击的告警量 (扫描类报警、入侵类告警、远控类告警)
3	Web 入侵	入侵检测 / 防护设备 Web 应用防火墙	Web 攻击的告警量 (扫描类报警、入侵类告警、Webshell 远控类告警)
4	暴力破解入侵	入侵检测 / 防护设备 Web 应用防火墙	暴力破解的告警量 (暴力破解类告警，远控类报警)
5	僵木蠕	入侵检测 / 防护设备 网络防病毒设备 / 模块 威胁分析系统 TAC	入侵 / 检测的蠕虫、僵尸、木马告警 防病毒僵木蠕的传播下载告警 TAC 的未知样本分析告警
6	APT 高级持续性攻击	入侵检测 / 防护设备 网络防病毒设备 / 模块 威胁分析系统 TAC 蜜罐	网络入侵、系统入侵、数据库入侵、Web 入侵告警， 防病毒的相关告警 TAC 的未知样本分析告警 蜜罐的相关告警

3.2.3 安全专家输出行为规则

在本阶段，由经验丰富的安全分析师来进行行为分析，传统的检测设备都是基于特征检测的，如 IDS、WAF 等，但是基于特征的检测很有可能遇到无法检测的情况，如 SSH、HTTPS 的加密协议，或者 Web 表单暴力破解，因为程序都是自行开发的，很难找到通用个规则来进行检测。但是基于行为检测，我们就能发现有趣的误报。图 5 就是一个 Web 表单暴力破解的告警 (注：为了更好的体现效果，我们调整了 NIDS 告警归并时间，调整成 1 分钟。)

我们可以看出，实际上我们用 1.1.1.1 对 2.2.2.2 的 80 端口的网站进行暴力破解，于是 1.1.1.1 开始随机端口连接 2.2.2.2 的 80 端口，于是 IDS 看起来像是 2.2.2.2 对 1.1.1.1 进行端口扫描 (看看 1.1.1.1 的 1-65535 端口哪个开放了)。从 IDS 设备本身基于特征的检测方法来看，这没有错，但是现实确实是一个误报。观察完成后，我们写下了如下的行为规则：

行为规则：

分析（相同源IP，相同目标IP）

IF

(触发2类告警（服务器端口扫描－SYNACK扫描 and 服务器端口扫描－ACK扫描）) and (源端口为1个and 目标端口>1) and ("服务器端口扫描－ACK扫描"告警协议摘要为TCP.HTTP S(注：此功能为IDS/IPS协议识别))

判定结果：

是目标IP正在向源IP进行WEB表单暴力破解。（结果需修正）

危险程度	时间	事件类型	事件名称	事件次数	源IP	源端口	目的IP	目的端口	协议摘要	规则编号	置行程度	攻击手段	服务类型
中风险	2013-05-23 18:32:00	攻击事件	服务器端口扫描－SYNACK扫描	4	2.2.2.2	80	1.1.1.1	55838	IP.TCP SCAN_PORT_SPEED>=20pps SCAN_PORT_COUNT=201 PORT_FIELD=218-65497	30522	低	信息收集类攻击	MISC
中风险	2013-05-23 18:31:59	攻击事件	服务器端口扫描－ACK扫描	5	2.2.2.2	80	1.1.1.1	55830	TCP.HTTP S SCAN_PORT_SPEED>=20pps SCAN_PORT_COUNT=201 PORT_FIELD=218-65497	30520	低	信息收集类攻击	MISC
中风险	2013-05-23 18:30:43	攻击事件	服务器端口扫描－SYNACK扫描	3	2.2.2.2	80	1.1.1.1	52740	IP.TCP SCAN_PORT_SPEED>=20pps SCAN_PORT_COUNT=201 PORT_FIELD=206-65485	30522	低	信息收集类攻击	MISC
中风险	2013-05-23 18:30:42	攻击事件	服务器端口扫描－ACK扫描	4	2.2.2.2	80	1.1.1.1	52731	TCP.HTTP S SCAN_PORT_SPEED>=20pps SCAN_PORT_COUNT=201 PORT_FIELD=12749-64461	30520	低	信息收集类攻击	MISC
中风险	2013-05-23 18:29:36	攻击事件	服务器端口扫描－SYNACK扫描	5	2.2.2.2	80	1.1.1.1	50044	IP.TCP SCAN_PORT_SPEED>=20pps SCAN_PORT_COUNT=201 PORT_FIELD=195-65474	30522	低	信息收集类攻击	MISC
中风险	2013-05-23 18:29:25	攻击事件	服务器端口扫描－ACK扫描	4	2.2.2.2	80	1.1.1.1	49616	TCP.HTTP S SCAN_PORT_SPEED>=20pps SCAN_PORT_COUNT=201 PORT_FIELD=1985-53697	30520	低	信息收集类攻击	MISC

图 5 Web 表单暴力破解攻击告警采样

3.2.4 行为规则加载在海量数据中进行训练修正，形成修正规则

将写好的 Web 表单暴力破解行为规则，通过规则引擎或者其他方式加载在态势感知分析平台进行实时分析，一旦发生此类事件，实时报警。同时安全分析师对分析结果进行训练修正。在本阶段重点是编写修正规则，修正规则的核心是通过关联分析来修正可信度，满足一定的可信度告警才会显示在界面上。

如上例，Web 表单暴力破解行为规则在海量数据中训练时发现仍然存在误报。我们发现，在对网站系统扫描攻击和 Web 扫描攻击时，也会触发这个规则，这个也很容易理解，系统扫描器和 Web 扫描器在发现 Web 表单后，都会进行暴力破解尝试。修正规则通过修正可信度对结果进行修正，因此形成如下修正规则：

修正规则：

分析（相同源 IP，相同目标 IP）

IF

((系统扫描攻击 and WEB 表单暴力破解)

OR （WEB 扫描攻击 and WEB 表单暴力破解))

THEN

WEB 表单暴力破解 事件可信度值为 0

3.2.5 可信可视结果输出

经过修正后的事件修正了可信度和攻击烈度指标后，就可以输出实时告警了。告警的输出应该遵循安全分析师可以尽快研判为主的原则，尽可能提供更详细的信息，在部分情况下，可以隐藏可信度，以免对客户造成困扰。

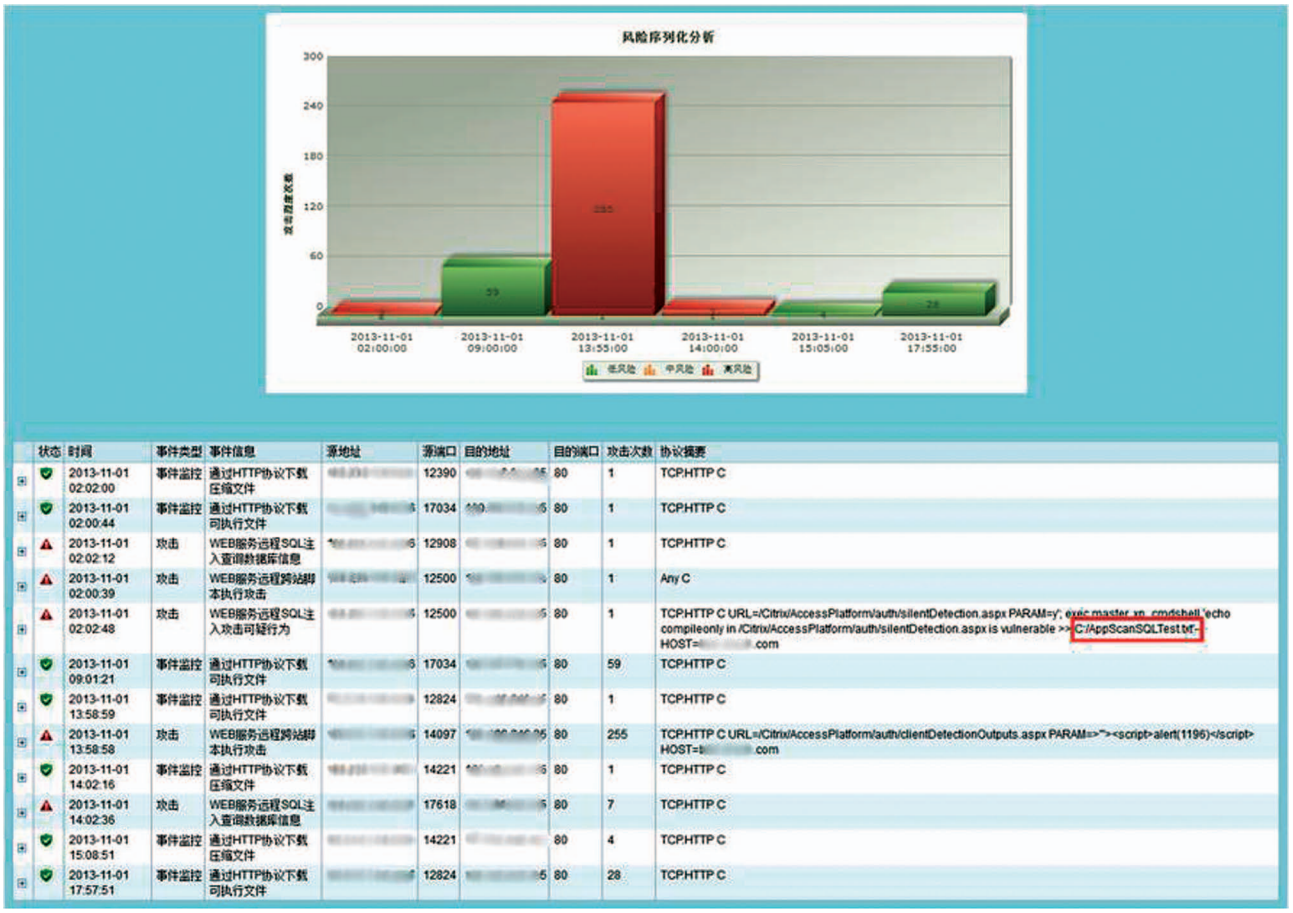
安全事件告警输出：

IP（x.x.x.x） 资产被 xx 事件攻击（时间 xxxx:xx:xx-xxxx:xx:xx，攻击源 IP 为 x.x.x.x）可信度 xx%，攻击烈度为 xx 次 / 分钟

3.3 一些研究成果

根据这种方法，我们形成了有效的大数

据下的态势感知理解，如图 6、图 7 展示的 Web 扫描和系统扫描的告警，原始数据经过基于攻击行为建模的态势理解方法后，仅仅形成了一条事件告警，大大降低了告警量，使得我们能专注于有用的告警。当然研究也在深入，如有人经常攻击一个网站，先进行系统扫描，再进行 Web 扫描，导致智能判断出现一些问题，仅仅判断成系统扫描，后续修正规则会增加端口时间聚类的数学算法



	状态	时间	事件类型	事件信息	源地址	源端口	目的地址	目的端口	攻击次数	协议摘要
⊞		2013-11-01 02:02:00	事件监控	通过HTTP协议下载压缩文件		12390		80	1	TCP;HTTP C
⊞		2013-11-01 02:00:44	事件监控	通过HTTP协议下载可执行文件		17034		80	1	TCP;HTTP C
⊞		2013-11-01 02:02:12	攻击	WEB服务远程SQL注入查询数据库信息		12908		80	1	TCP;HTTP C
⊞		2013-11-01 02:00:39	攻击	WEB服务远程跨站脚本执行攻击		12500		80	1	Any C
⊞		2013-11-01 02:02:48	攻击	WEB服务远程SQL注入攻击可疑行为		12500		80	1	TCP;HTTP C URL=/Citrix/AccessPlatform/auth/silentDetection.aspx PARAM=y'; exec master..xp_cmdshell 'echo compileonly in /Citrix/AccessPlatform/auth/silentDetection.aspx is vulnerable >> C:/AppScanSQLTest.txt'-- HOST= .com
⊞		2013-11-01 09:01:21	事件监控	通过HTTP协议下载可执行文件		17034		80	59	TCP;HTTP C
⊞		2013-11-01 13:58:59	事件监控	通过HTTP协议下载可执行文件		12824		80	1	TCP;HTTP C
⊞		2013-11-01 13:58:58	攻击	WEB服务远程跨站脚本执行攻击		14097		80	255	TCP;HTTP C URL=/Citrix/AccessPlatform/auth/clientDetectionOutputs.aspx PARAM=>"><script>alert(1196)</script> HOST= .com
⊞		2013-11-01 14:02:16	事件监控	通过HTTP协议下载压缩文件		14221		80	1	TCP;HTTP C
⊞		2013-11-01 14:02:36	攻击	WEB服务远程SQL注入查询数据库信息		17618		80	7	TCP;HTTP C
⊞		2013-11-01 15:08:51	事件监控	通过HTTP协议下载压缩文件		14221		80	4	TCP;HTTP C
⊞		2013-11-01 17:57:51	事件监控	通过HTTP协议下载可执行文件		12824		80	28	TCP;HTTP C

图 6 Web 扫描事件发现告警

	状态	时间	事件类型	事件信息	源地址	源端口	目的地址	目的端口	攻击次数	协议摘要
⊞		2013-10-11 10:03:46	攻击	Windows系统下BackOrifice 1.2木马PING操作		23821		31337	1	Any
⊞		2013-10-11 10:04:08	攻击	Windows系统远程管理工具PCAnywhere会话启动请求		12337		5632	1	Any
⊞		2013-10-11 10:01:53	攻击	Cisco IOS ILMI SNMP共同体串访问		57597		161	1	UDP:SNMP
⊞		2013-10-11 10:03:21	事件监控	TFTP服务客户端从服务器端获取文件		53380		69	1	UDP:TFTP
⊞		2013-10-11 10:02:05	攻击	DNS服务获取服务器版本号请求操作		24411		53	1	UDP:DNS NAME=VERSION.BIND:
⊞		2013-10-11 10:02:40	事件监控	SNMP服务试图使用默认public口令访问		31390		161	1	UDP:SNMP
⊞		2013-10-11 11:02:47	事件监控	SNMP服务试图使用默认public口令访问		31390		161	4	UDP:SNMP
⊞		2013-10-11 16:46:47	攻击	DNS服务获取服务器版本号请求操作		18189		53	1	UDP:DNS NAME=VERSION.BIND:
⊞		2013-10-11 16:49:55	事件监控	SNMP服务访问使用默认private口令		30472		161	1	UDP:SNMP
⊞		2013-10-11 16:54:47	事件监控	通过HTTP协议下载可执行文件		16917		80	1	TCP:HTTP C
⊞		2013-10-11 16:52:55	攻击	WEB服务远程SQL注入查询数据库信息		25552		85	1	TCP:HTTP C
⊞		2013-10-11 21:50:09	事件监控	通过HTTP协议下载可执行文件		16917		80	1	TCP:HTTP C

图 7 系统扫描事件发现告警

方式来进行修正。

四、结语

本文所述基于对抗的智能态势感知预警模型——“基于攻击行为建模的态势理解方法”和“基于对抗的 APT 攻击推理树态势预警方法”，都是基于对抗的，看重实效，虽然可能不是很完美，但是确实非常有效。本文的态势感知核心点还是基于威胁的态势感知（知彼），因为威胁的态势感知遭遇到安全大数据后变得非常复杂。对于脆弱性的态势感知（知己）我们会在持续监控的文章来探讨。本文算是抛砖引玉吧。欢迎大家交流。

注：本文方法正在申请相关专利，保留相关所有权利。

参考文献

1. 网络安全态势感知研究新进展，王慧强，哈尔滨工程大学

2. 大规模网络安全态势感知—需求、挑战与技术，贾焰，国防科大计算机学院网络所

3. 大数据中的实时精准营销与风险控制，林南晖，IBM

Openstack网络虚拟化安全分析

战略研究部　刘文懋

关键词：Openstack 网络虚拟化 安全

摘要：越来越多的数据中心和大企业开始采用虚拟化的私有云解决方案，网络虚拟化的安全越来越受到重视，本文以 Openstack 为例探讨了在实现网络虚拟化时采用的安全机制。

引言

随着云计算的落地，越来越多的开源设施虚拟化系统（IaaS，Infrastructure as a Service）被采纳并被部署到生产环境中，如图 1 所示。其中 Openstack 近年来最为流行 [1]，原因在于其具有开放的接口和灵活的架构，特别是互联网企业和新型的云计算中心，通过二次开发定制化模块的方式，部署了满足其业务需求的 Openstack 系统，目前已有一些超过 1k 物理节点的应用案例。因此 Openstack 的安全防护，特别是网络方面的安全机制，对于服务提供商、租户和安全厂商都尤为重要。

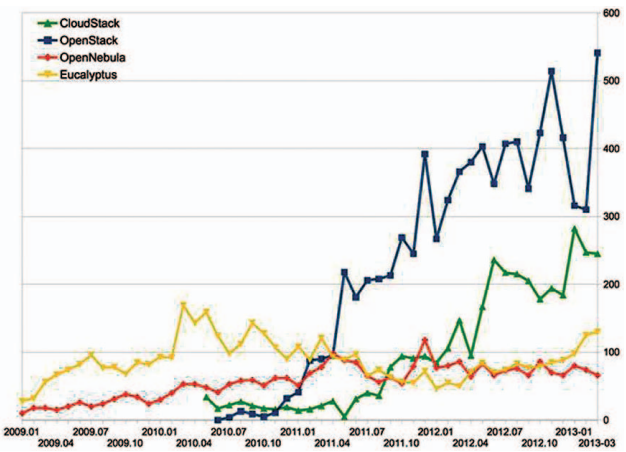


图 1 各类开源 IaaS 系统的社区参与人数比较

一、Openstack 简介

Openstack 包括计算虚拟化（Nova 模块）、存储虚拟化（Swift、Glance 和 Cinder 模块）和网络虚拟化（Neutron 模块）三大部分，

它利用现有开源的计算、网络和存储工具，构建了图 2 中面向服务的设施虚拟化解决方案。通俗来说，Openstack 借助了已有的开源工具，驱动异构的物理资源，并提供集中管理和协同各类资源的能力。如网络虚拟化组件 Neutron 使用 Open vSwitch（OVS）实现虚拟机间的互联，使用 IPTABLES 实现访问控制，使用 Linux kernel namespace 实现网络空间隔离，使用 vlan 实现网络隔离，诸如此类。

在初始几个版本中，Openstack 的网络模块集成在计算模块 Nova 模块中，在 Grizzly 版本后分离为独立的网络虚拟化模块 Quantum，随之在下一个版本 Havana 中被改名为 Neutron。可见网络虚拟化功能越来越受到社区重视，越来越多的网络功能得以以服务的形式交付，如负载均衡服务、防火墙服务和 VPN 服务等，Neutron 自身的安全机制和安全问题因此日益受到人们的重视，本文主要探讨 Neutron 中当前的一些安全机制。

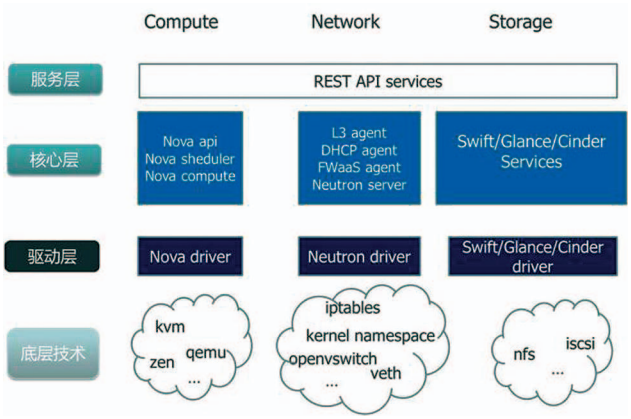


图 2 Openstack 的整体结构图

二、虚拟网络的隔离和 VM 间的访问控制机制

（一）防伪造地址

在公有云中，租户可以租用虚拟机发动 DDoS 攻击或 ARP 欺骗等，这些攻击都是通过伪造源 MAC 或 IP 地址实现的。其特点是攻击者从虚拟化系统内部发起，攻击目标可能是公网主机，或者是内网的主机。对于前者的防御比较容易，只需要在物理网络边界就可以部署安全设施进行检测和清洗；但是后者的攻击流量大部分存在于虚拟网络和虚拟网络设备中，物理的安全设备无法观测或理解物理网络中的流量，无法进行有效的防护，所以对于始于内部的 DoS 攻击，应该在虚拟化环境中检测并抵御。较传统数据中心而言，IaaS 系统的优势在于其知晓 VM 的真实 IP 和 MAC，所以可以直接在 L2 防火墙上对数据包进行过滤，Havana 版已经引入了这部分特性，可抵御虚假源地址的攻击。

以一台 VM（IP 为 100.0.0.16, MAC 为 fa:16:3e:34:c8:f8）为例，Neutron 为其连接 OVS 分配了 ID 为 368d35e1-4db7-405f-af26-1f2b6f224bd8 的端口。那么在其所在的计算节点上查看 Neutron 中基于 IPTABLES 的过滤方案，运行 iptables -L -nvx 可以发现下面项：

Chain neutron-openvswi-s368d35e1-4 (1 references)

pkts bytes target prot opt in out source destination

212 29180 RETURN all — * * 100.0.0.16 0.0.0.0/0 MAC FA:16:3E:34:C8:F8

0 0 DROP all — * * 0.0.0.0/0 0.0.0.0/0

此处允许真实 IP 和 MAC 的数据包经过，对其他数据包抛弃。此外，Nova 也引入了基于 EBTABLES 的过滤方案，运行 ebtables -t nat -L 可以发现下面项：

Bridge chain: libvirt-I-tap368d35e1-4d, entries: 6, policy: ACCEPT

-j I-tap368d35e1-4d-mac

-p IPv4 -j I-tap368d35e1-4d-ipv4-ip

-p IPv4 -j I-tap368d35e1-4d-ipv4

-p ARP -j I-tap368d35e1-4d-arp-mac

-p ARP -j I-tap368d35e1-4d-arp-ip

Bridge chain: I-tap368d35e1-4d-mac, entries: 2, policy: ACCEPT

-s fa:16:3e:34:c8:f8 -j RETURN

-j DROP

Bridge chain: I-tap368d35e1-4d-ipv4-ip, entries: 3, policy: ACCEPT

-p IPv4 –ip-src 0.0.0.0 –ip-proto udp -j RETURN

-p IPv4 –ip-src 100.0.0.16 -j RETURN

-j DROP

Bridge chain: I-tap368d35e1-4d-arp-mac, entries: 2, policy: ACCEPT

-p ARP –arp-mac-src fa:16:3e:34:c8:f8 -j RETURN

-j DROP

Bridge chain: I-tap368d35e1-4d-arp-ip, entries: 2, policy: ACCEPT

-p ARP –arp-ip-src 100.0.0.16 -j RETURN

-j DROP

可见 Openstack 确实将 VM 的 MAC 和 IP 进行了过滤，防止虚假源的数据包经过。需注意的是，虽然这种防护可以抵御伪造源地址的攻击，但是无法抵御使用真实源地址的 DoS 攻击，遇到这种情况，还是采用 ceilometer 或是 SDN 的流检测的方式较好。

（二）虚拟网络隔离

Neutron 中的网络隔离是使用 vlan 实现的，每个租户的 VM 会连接到 OVS 的某个端口，那么该端口被打上一个 Tag，所有流出该端口的数据包都会加上与 Tag 一致的 vlan 值，所以同一个物理节点中，同一个网络的数据包中所含的 vlan 是相同的，但不同网络的数据包则有不同的 vlan 值，从而网络间是彼此隔离的。

当一个网络可能分布在不同物理节点上时，由于 Openstack 的 OVS Tag 不是全局的，即同一个网络在不同物理节点中有不同的 vlan 值，所以当数据在不同物理节点中传输时，需要对这些同属一个网络的不同节点上的 vlan 进行映射。如图 3 展示了使用 GRE 隧道连接的场景，VM1、VM2 和 VM3 同属网络 Net1，VM4 属于网络 Net2，Net1 在两个节点中的 OVS Tag 值分别为 1 和 3。那么当数据到达第一个节点的 GRE 隧道入口时，Neutron 将 vlan 映射为 GRE 隧道的 key，当数据到达另一个节点的隧道出口时，vlan 又被还原为该节点中 Net1 对应的 Tag=3，这样不同网络在多个物理节点

上也是隔离的。

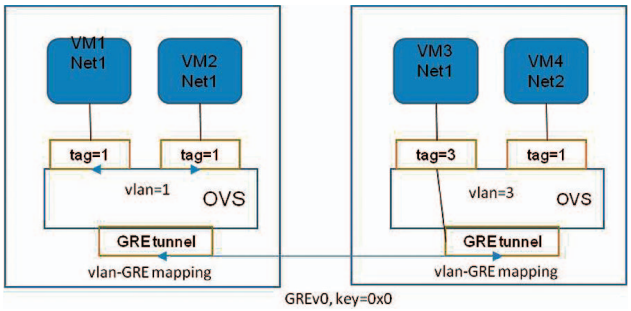


图 3 GRE 隧道连接的不同物理节点的网络隔离

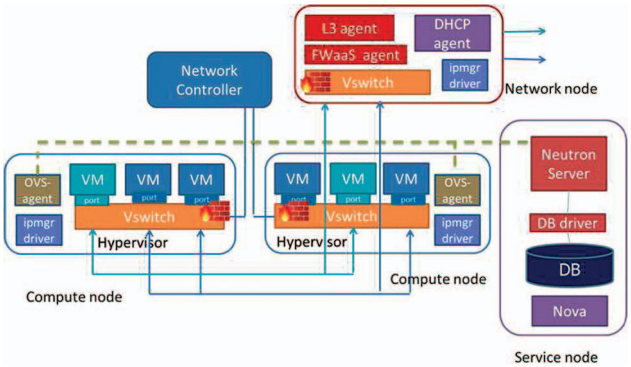


图 4 Neutron 的访问控制机制

需要注意的是，当 Openstack 与 SDN 结合时，通过 vlan 标记不同网络可能无法做到简单的网络隔离。因为首先 OVS 加入 SDN 后发送的 Packet_In 不包含 vlan 值，SDN 控制器不了解 Openstack 中的网络隔离情况，当多租户网络重叠的情况可能会出现路由错误的问题；其次 SDN 控制器控制了全部的二层转发过程，所以 SDN 控制器必须获取 Openstack 中的网络隔离策略，对含不同网络的数据包进行合理路由决策。

（三）VM 访问控制

Neutron 中的 VM 访问控制包括两个部分：二层的安全组（Security Group，SG）和防火墙即服务（Firewall as a Service，FWaaS），如图 4 所示 [2]。

其中，安全组主要位于虚拟机与虚拟交换机之间的连接，用于控制内部网络中虚拟机的访问，特别是同一内网中的虚拟机间的通信，如图 5 所示。安全组的底层驱动通常为 IPTABLES 规则，但在当前的 Linux 系统下，虚拟交换机 OVS 与 IPTABLES 的规则尚不兼容，所以 IPTABLES 策略不能在 VM 到 OVS 的路径上生效。于是 Openstack 在设计时做了一个临时解决办法，即在 VM 与 OVS 之间加了一个 Linux bridge，通过 veth 对将两者相连，所有的 IPTABLES 规则在 Linux bridge 生效，从而完成 VM 出入口的访问控制。

Neutron 的三层访问控制主要是由 FWaaS 实现，底层驱动也是 IPTABLES。那么安全组和 FWaaS 的区别主要在哪里呢？第一，两者的逻辑结构是不同的，如图 6 所示，安全组的主要规则在 security group rules 中，FWaaS 的规则主要在 firewall rules 中；第二，安全组是面向交换机端口的，而 FWaaS 是面向路由器的；最后，安全组不能表达下一代防火墙应有的应用特性，例如安全组不支持“提供商发布一系列规则，租户选择对其启用”的特性，此外，安全组没有涉及很多边界防火墙服务提供的更丰富的管理、应用方面的需

求特性。因而安全厂商可将原有的硬件防火墙进行一定改造，就可以 FWaaS 的方式提供防火墙服务。

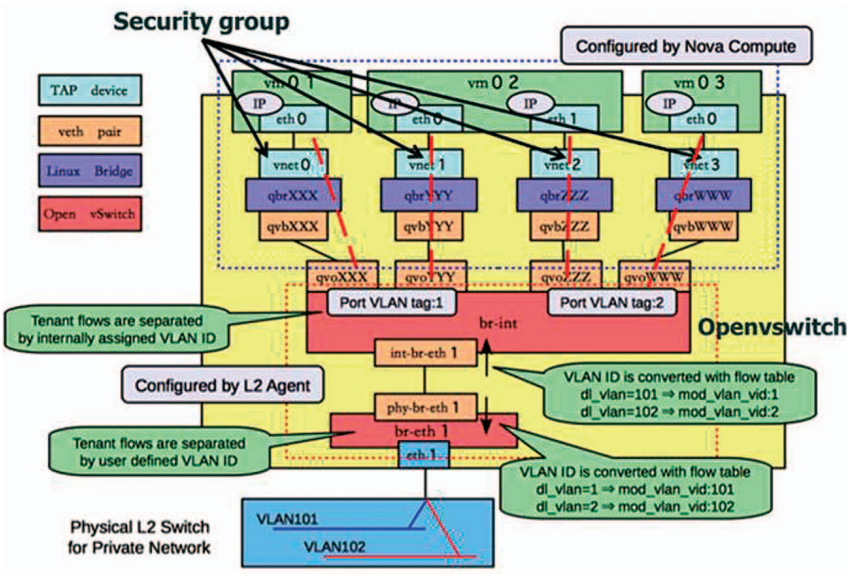
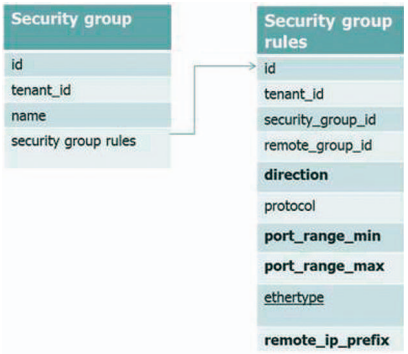
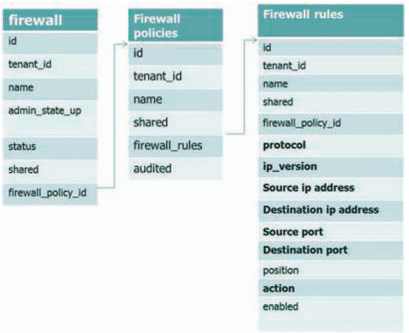


图 5 Neutron 的二层防火墙控制示意图

Neutron 中一种租户和网络隔离的方法是通过命名空间（namespace）实现的，图 7 展现了这种方法下的 FWaaS 部署情况。假定存在租户的两个虚拟内网 A、B 和一个虚拟外网 C，通过路由器 R 相连，内网存在 DHCP 服务。那么在 Neutron 的网络节点上存在三个网络服务，分别是 A 和 B 内部的 DHCP 服务和连接路由器 R，于是对应了三个命名空间，即 A、B 的 DHCP 服务器所在的 qrouter-bbbb 和 qdhcp-cccc 以及 R 所在的 qrouter-bbbb，以 tap、qr 和 qg 开头的均为这些服务器的网卡。一个命名空间内的数据对其他命名空间是隔离的，但可通过网卡连接到网桥 br-int 上进行通信。可见整个三层网络的转发是通过命名空间



(a) 安全组的数据结构



(b) FWaaS 的数据结构

图 6 安全组和 FWaaS 的数据结构

qrouter-bbbb 实现的，所以 FWaaS 应该在网卡 qr-YYY 和 qr-ZZZ 上生效，所有经过路由器的数据包均接受 FWaaS 的访问规则检查。

介绍完 Neutron 访问控制的逻辑结构

后，读者不妨通过 IPTABLES 自行验证一下，安全组的策略可以在相应的计算节点运行 IPTABLES 查看，FWaaS 可在网络节点的相应命名空间中用 IPTABLES 查看。其中 neutron-openvswi- 开头的规则为 Neutron 下发的，而 nova-compute 开头的则是 Nova 下发的。

三、Openstack 网络安全的产品

传统的安全厂商在 IaaS 系统中面临巨大的挑战，特别是如何将自有的产品部署在虚拟化环境中。Neutron 提供的 FWaaS 就是一种很好的解决办法，即厂商可以先将自己的产品形态虚拟化，然后将流量牵引到虚拟安全设备，实现安全防护。

山石网科和 vArmour 都发布了支持 Neutron 的 FwaaS：前者的虚拟防火墙当前支持源地址转换、目的地址转换、服务器负载均衡、基于应用的访问控制、拒绝服务攻击防护、会话限制等网络安全特性；vArmour 替换了 Neutron L3 Agent，支持 NAT，使用 REST 调用 vArmour 的防火墙，并扩展了原有的 neutron FWaaS 驱动，底层支持 vArmour 的防火墙，提供了全方位的访问控制防护。

四、结论

网络虚拟化安全还有很多工作需要完成，例如 Neutron 与 SDN 控制器整合后的访问集中管理；此外现有的工作还有大量需要改进之处，如 FWaaS 当前还有很多限制（一租户一 FW）。越来越多的安全企业开始将安全产品兼容网络虚拟化和 SDN 场景，Garnter 预测截止 2014 年，TOP 10 网络安全厂商中的 9 家会支持 OpenFlow；截止 2015 年，40% 的企业安全数据中心网络和安全服务会通过虚拟设备交付 [3]，所以需要在 Openstack 和 SDN 结合的企业私有云环境中研究快速交付、用户可编程的软件定义安全解决方案。

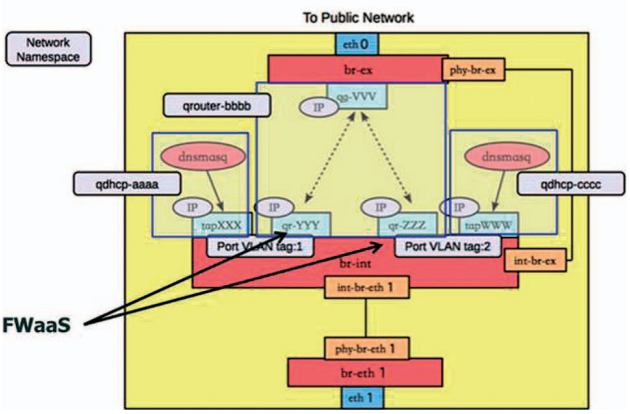


图 7 Neutron 三层防火墙的实现示意图

参考文献

[1] CY13-Q4 OpenStack, OpenNebula, Eucalyptus, CloudStack 社区活跃度比较 http://www.qyjohn.net/?p=3431,2013

[2] OpenStack Configuration Reference - Havana, http://docs.openstack.org/havana/config-reference/content/under_the_hood_openvswitch.html ,2013

[3] Garnter, The Impact of Software-Defined Data Centers on Information Security, https://www.gartner.com/doc/2200415/impact-softwaredefined-data-centers-information, 2013

RSA会议回顾与应用安全

战略研究部　忽朝俭

关键词：应用安全 安全评估 漏洞管理 云安全 大数据

摘要：从去年的“知识中的安全”到今年的“利用集体智慧”，“智能驱动的安全”作为RSA的主旋律并没有发生太大的变化，因此本文不打算再重复这些话题。在对本届RSA会议的热点和独特之处进行简单点评之后，本文重点对第三方组件（二进制商业软件或开源代码）的安全评估以及以数据分析为核心的下一代漏洞管理等话题进行介绍。

引言

继Snowden披露了NSA的PRISM计划[1]之后，路透社又披露了RSA公司收取NSA 1000万美金并同意后者在其安全软件Bsafe中植入后门[2]，这一系列事件让本届RSA会议的气氛非常敏感。而在“分享•学习•加固：利用集体智慧”主题的衬托下，本届RSA会议自然就带给人一种异常微妙的感觉——我们是应该维护自己的个人隐私还是通过分享与学习，进而保障集体的安全。

随着APT继续被全球信息安全行业热炒，Fireeye也在股市上创造了一系列的神话，而收购Mandiant更是带来APT产品的高度关注。虽然未能摘得本届RSA创新沙箱竞赛的最高荣誉——最佳创新公司的桂冠，但是Cyphort公司[3]凭借其在检测并分析下一代威胁和高级恶意软件，提供可追溯的、有关联的智能信息，从而使得安全团队能够更快更高效地响应安全事件方面的努力，仍旧进入了最终的10支决赛队伍名单。

大数据是今年RSA会议上最热门的话题。这从RedOwl Analytics公司[4]摘得本届RSA最佳创新公司的桂冠可以得到佐证。RedOwl Analytics公司基于大数据分析的主动安全分析服务可有效减少企业运作风险，并为金融机构、研究公司、财富500强公司和政府机构等提供尖端的前瞻性安全解决方案以及大数据分析和调查的新思路。

除此之外，越来越多的展商开始讲述自己在大数据至少数据分析上的创新性产品和服务。例如，Intel Security（原 MaCaff 公司）使用大数据分析做 APT 对抗和异常行为检测，而 Tenable 公司更是将数据分析作为其下一代漏洞管理（NGVM）解决方案 [5] 的核心内容。

对于绝大多数展商来说，基于云的安全服务已经成为标配，并且 XX as a Service 和 Software Defined XX 概念更多地被提出。例如，CheckPoint 推出了“Software Defined Protection”的概念，但是更像是市场营销，并没有从技术架构上遵从主流的 SDN/SDS 体系。

随着 DDoS 市场的逐步成熟，DDoS 公司之间整合的节奏也逐步加快。2013 年 11 月，Akamai 收购 Prolexic; 2014 年 2 月，Imperva 收购 Incapsula。各相关公司都在重新为自己定位。

随着移动互联网的飞速发展和日益成熟，越来越多的公司和个人开始关注移动安全。其中，移动设备中企业和个人数据以及隐私安全、移动恶意软件检测等话题得到较多的关注。

除了数据和隐私安全、APT、大数据、云计算安全以及移动安全等热门话题之外，根据笔者自己的研究兴趣和专长，本届 RSA 会议期间更多地关注了应用安全厂商的展台和相关演讲。下文笔者将重点与大家分享针对本届 RSA 会议中应用安全专题的整理和总结。

一、什么是应用安全

在深入介绍应用安全之前，非常有必要弄清楚信息安全、网络安全、软件安全以及应用安全等概念间的差异和相互关系。但是笔者在这里并不打算给出这些概念的严格定义，而只是简单地使用一个示意图来帮助大家理解这些概念。

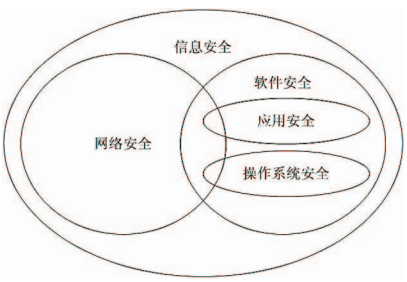


图 1 应用安全的内涵和外延

与网络相比，应用庞大而复杂，而应用安全也可以说是信息安全最后的处女地。大约十年前，应用安全还只是一个冷僻的话题，大家提到的信息安全基本上都是网络安全。最近几年，随着风险投资者和大公司的关注，应用安全不仅成为 RSA 会议中一个独立的专题，而且地位日益巩固。

二、如何确保应用安全

与保障网络安全相比，确保应用安全很困难。应用安全涉及到应用程序的整个生命周期（设计、开发、部署、升级和维护）中可能出现的安全策略缺失以及与应用软件漏洞相关的所有问题，因此不能简单地依靠在网络上放置一个或者多个盒子来保证应用安全。

总体来说，本次 RSA 会议上与应用安全相关的演讲和展示主要包括对软件开发中使用的第三方组件（二进制商业软件或开源代码）的安全评估及漏洞发现以及以数据分析为核心的下一代漏洞管理等话题。下面，笔者将分别就上述几个方面和大家分享针对本届 RSA 会议的整理和总结情况。

（一）安全评估与漏洞发现

1. Sonatype 公司的开源组件分析产品

软件开发越来越像堆积木——通过将不同来源的代码或组件进行组装，一个软件就可以构建出来了。根据 Sonatype 公司 [6]2012-2013 年间对 1000+ 企业应用的分析：当前自写代码在应用开发中所占的比例大约在 10%。

图 2 软件开发中 90% 的代码来自外部

但是任何事物都有两面性，这种敏捷开发技术同样也是一柄双刃剑。正如世界上没有两片完全相同的树叶，来自不同程序员的代码在质量上也难免参差不齐。如何保证第三方组件或代码足够安全，一直以来都是一个让人头疼的棘手问题。Sonatype 公司的组件生命周期管理（CLM）工具通过联合使用动态、静态程序分析技术，可帮助开发人员避免使用粗糙的开源组件。

最后再顺便提一下，本届 RSA 会议期间，HP 宣布将在自家的基于云的安全解决方案 Fortify on Demand 中集成 Sonatype 的产品。对于 Fortify on Demand 的用户来说，通过创建一个使用到的组件列表，就可以识别出这些组件中存在的漏洞，并得到修复建议。

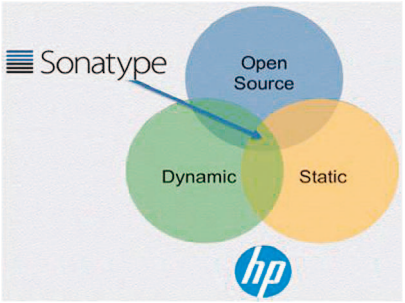


图 3 HP Fortify on Demand
集成 Sonatype 的开源分析产品

2. Veracode 集成了二进制静态分析引擎等组件的云漏洞分析平台

内部部署的传统的应用安全方法通常强制一些不必要措施从而增加复杂性，并采用非中心模型使得一致性的策略、报表和度量难以在不同的团队实施，因此难以解决全球化公司普遍存在的应用层安全问题。Veracode 公司 [7] 基于云的漏洞分析平台提供一种更简单和可规模化的方法解决这一问题。

Veracode 基于云的漏洞分析平台：1）可向内部及第三方应用与组件开发人员提供开发插件；2）向 WAF、MDM 等产品或运维操作提供基于云的服务；3）向开发者工作流集成、中心策略管理等提供 API 接口。

在核心技术组件上，Veracode 专利的二进制静态分析引擎（SAST）除了可以识别 Web 应用中常见的 SQL 注入、XSS 等漏洞类型，还可识别诸如缓冲区溢出、未处理的错误条件和潜在后门等漏洞类型。其工作原理为：首先创建应用程序的详细的数据和控制流（路径）模型，然后搜索所有的流（路径）查找潜在的缺陷。

除此之外，该平台还集成了 Veracode 的动态分析引擎（DAST）、移动行为分析引

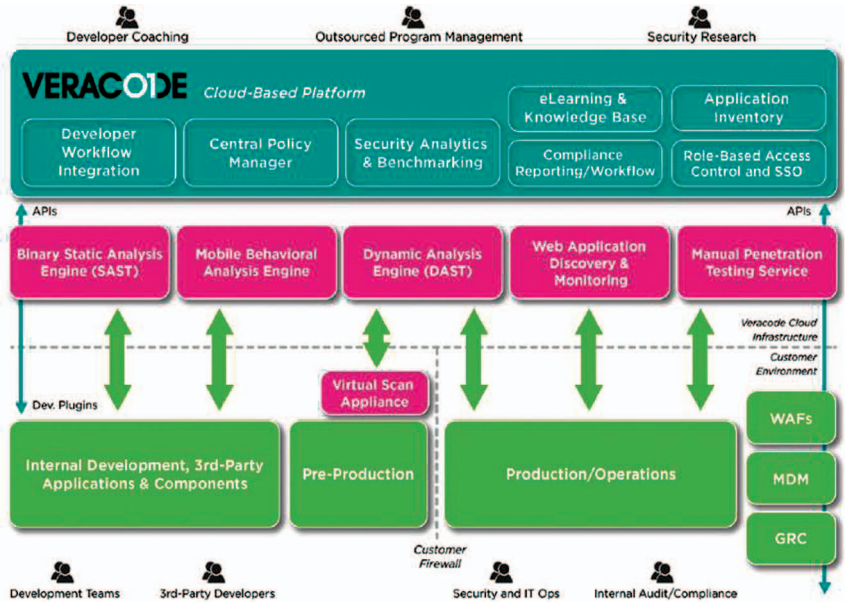


图 4 Verocode 基于云的软件漏洞分析平台

擎、Web 应用发现和监视、手工渗透测试服务等组件。

（二）以数据分析为核心的下一代漏洞管理

1. Tenable 的带连续监视的安全中心

本次 RSA 会议期间，Tenable 公司总裁 Ron Gula 组织了对其新书 “Definitive Guide to Next-Generation Vulnerability Management” [5] 的签名送书活动。这本 60 多页的小册子提出了下一代漏洞管理的概念，并详细介绍了 Tenable 的下一代漏洞管理战略，如表 1 所示。

除此之外，Tenable 公司带来了名为 “Security Center Continuous View” 的组合活动扫描、被动监视以及日志分析的集成漏洞和威胁管理解决方案。

除了传统的活动扫描外，其中的被动监视组件 “Tenable Passive Vulnerability Scanner” 和日志分析组件 “Tenable Log Correlation Engine” 是其最新成果。被动监视组件可在网络层监视 IPv4、IPv6 以及混合的网络流量，以决定网络拓扑情况、开放的服务以及漏洞存在情况，而日志分析组

关键能力	VM	NGVM
中心管理	Y	Y
仪表盘和报表	Y	Y
活动漏洞扫描	Y	Y
被动漏洞扫描	N	Y
日志聚集和关联	N	Y
移动设备扫描	N	Y
虚拟平台扫描	N	Y
智能扫描负载均衡	N	Y
补丁审计	N	Y
恶意软件检测	N	Y

表 1 Tenable 提出的下一代漏洞管理

件可使用活动扫描和被动扫描产生的所有数据并进行关联和分析。

简单来说，该方案可聚集、规范化、关联和分析来自原始网络流量、入侵检测数据、系统和应用日志以及用户活动的事件日志数据。

2. Secunia 的 2013 年漏洞趋势分析

本次 RSA 会议期间，Secunia 公司 [8] CTO Morten Stengaard 带来了针对 2013 年全球漏洞趋势的数据分析（通过 Secunia Personal Software Inspector 收集），主要就漏洞数量和检测情况、漏洞较多的程序、供应商修复漏洞所用的时间、最常用浏览器和 PDF 阅读器的安全情况等方面的数据进行了分析。要点如下：

1）在 539 家供应商的 2289 个产品中共检测到 13073 个漏洞，同比增加 32%，有漏洞的供应商增加 13%，有漏洞的产品减少 6%。

2）较高 Critical 级别漏洞比率升高，较低 Critical 级别漏洞比率降低：16.3% 漏洞评为 Highly Critical，0.4% 评为 Extremely Critical，Moderately Critical 级别从 29.2% 减少到 23.3%，Less Critical 从 46.6% 增加到 52.%。

3）最主要攻击向量是远程网络，占 73.5%，本地网络和本地系统都有一定增长，分别从 15% 增加到 19.9%，从 5% 增加到 6.6%。

4）漏洞最多的程序依次是 firefox、chrome、jre、IE、Windows 7……

RANK	TYPE	PROD	SHARE	ADVS	CVES	VULNS
1	MS	MICROSOFT XML CORE SERVICES (MSXML)	99.9%	1	2	2
2	MS	MICROSOFT WINDOWS MEDIA PLAYER	99.4%	1	1	1
3	MS	MICROSOFT INTERNET EXPLORER	99.1%	14	123	126
4	MS	MICROSOFT .NET FRAMEWORK	99.1%	6	18	18
5	TP	ADOBE FLASH PLAYER	97.5%	12	56	56
6	MS	MICROSOFT VISUAL C++ REDISTRIBUTABLE	95.4%	0	0	0
7	TP	ADOBE READER	85.6%	5	67	67
8	MS	MICROSOFT SILVERLIGHT	84.3%	3	9	9
9	MS	MICROSOFT POWERSHELL	82.1%	0	0	0
10	TP	ORACLE JAVA JRE	82.1%	7	181	181
11	MS	MICROSOFT WINDOWS DEFENDER	77.1%	1	1	1
12	MS	MICROSOFT WORD	74.9%	4	17	17
13	MS	MICROSOFT EXCEL	73.8%	3	6	6
14	MS	MICROSOFT POWERPOINT	71.7%	1	1	1
15	MS	WINDOWS DVD MAKER	70.8%	0	0	0
16	TP	MOZILLA FIREFOX	63.8%	20	140	270
17	TP	GOOGLE CHROME	61.0%	20	209	245
18	MS	WINDOWS MEDIA CENTER	59.8%	0	0	0
19	MS	MICROSOFT VISIO VIEWER	56.5%	1	1	1
OS	MS	MICROSOFT WINDOWS 7	N/A	37	100	102

图 5 2013 年漏洞最多的程序

5）在披露的当天有补丁可用的漏洞比率从 70.1% 增长到 78.6%。

6）浏览器中市场占有率最高的是 IE（99%），未补丁率最高的是 Opera（39%），漏洞最多的是 firefox（270 个）；PDF 阅读器中占有率最高的是 Adobe（91%），未补丁率最高的是 Foxit（44%），漏洞最多的是 Adobe（67 个）。

三、结束语

在 Snowden 和 RSA 后门等一系列事件后，个人隐私问题变得极度敏感，而“利用集体智慧”的会议主题又使得本届 RSA 会议的气氛相当微妙。基于云的安全服务已经成为标配，

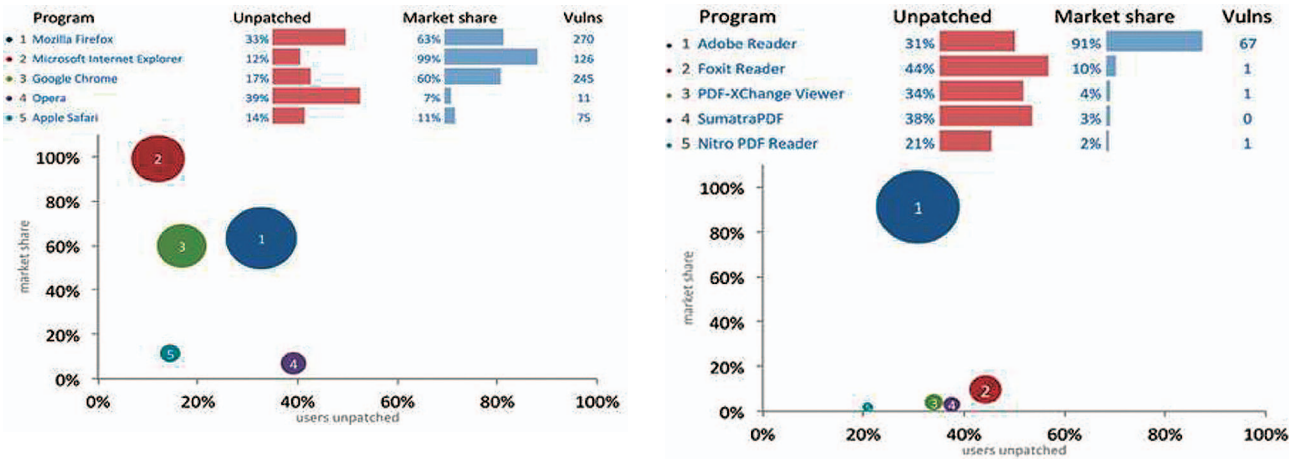


图 6 2013 年主流浏览器和 PDF 阅读器漏洞情况

完全不涉及云的厂商会因为无人问津而略显尴尬，大数据以及基于数据分析的智能安全到处可见。具体到应用安全领域，基于云的软件安全评估和漏洞分析将成为未来的潮流，面向漏洞的数据或大数据分析的广度和深度都将进一步加强。

参考文献

[1] http://en.wikipedia.org/wiki/PRISM_(surveillance_program)

[2] Menn Joseph, "Exclusive: Secret contract tied NSA and security industry pioneer"

http://www.reuters.com/article/2013/12/20/us-usa-security-rsa-idUSBRE9BJ1C220131220

[3] http://www.cyphort.com/products/

[4] http://redowlanalytics.com/product/

[5] Ron Gula, "Definitive Guide to Next-Generation Vulnerability Management"

http://static.tenable.com/marketing/definitive-guide-to-ngvm.pdf

[6] Ryan Berg, "The Game of Hide and Seek, Hidden Risks in Modern Software Development"

http://www.rsaconference.com/writable/presentations/file_upload/asec-r02-the-game-of-hide-and-seek-hidden-risks-in-modern-software-development_final.pdf

[7] http://www.veracode.com/products/cloud-based-platform

[8] Morten Stengaard, "Secunia Vulnerability Review 2014"

http://secunia.com/resources/reports/vr2014/

无文件系统嵌入式固件后门检测

研究院　忽朝俭 李鸿培 赵粮

关键词：嵌入式系统 固件 文件系统 库函数识别 后门检测

摘要：在无文件系统嵌入式固件中，系统代码和应用代码集成在单个文件中，无法看到熟悉的系统调用名字，故针对此类固件的分析将更为困难。本文以此类固件为研究对象，分析了其中的库函数识别问题，并提出了一种针对网络套接字和字符串 / 内存操作函数的基于启发式规则的识别方法。在此基础上，讨论了多种典型的后门类型的检测问题，包括未授权侦听者、非预期功能、隐藏功能和向外的连接请求等，并在一款实际系统上成功检测出多个后门（其中有一个严重级别的）。实验结果表明，本文提出的针对无文件系统嵌入式固件的库函数识别方法对于此类固件的安全分析具有重要的参考价值。

引言

嵌入式系统是一种完全嵌入受控设备内部、为特定应用而设计的专用计算机系统。典型的嵌入式系统一般由四部分组成，即处理器、存储器、输入输出（I/O）和软件。由于历史原因，嵌入式系统的软件又称为固件。获取嵌入式固件的方法很多，可以直接从提供商的更新发布中获取，也可以从嵌入式设备导出（Dump）。其中，前一种方法比较简单，而后一种需要较高的技术和实践能力，并可能需要借助于特定的工具。

根据嵌入式固件是否使用文件系统，本文将其分为带文件系统的（例如 VxWorks、Win CE 和 uClinux 等）和无文件系统的（例如 ThreadX 和 uCOS-II 等）两类。对于规模和复杂度较大的嵌入式固件，通常需要使用不同的文件来实现不同的功能模块，并使用某种特定的文件系统（例如 romfs、ext2 和 fat 等）负责存取和管理其内部的文件信息。由于带文件系统的嵌入式固件通常可以根据其文件系统将其分解为不同的软件模块，然后使用针对一般计算机程序的常用分析方法对单个模块进行分析，本文不再赘述这种情况。

目前，针对无文件系统嵌入式固件的安全性分析还较零散，也不够深入。无文件系统的固件规模上通常不及带文件系统的固件庞大，容易让人误认为出现安全问题的概率不大，故其安全性分析目前尚未得到足够的重视。另一方面，无文件系统的嵌入式固件通常没有系统软件和应用软件的区分（例如，ThreadX[1] 操作系统是一个函数库，应用程序可使用该库提供的“创建线程”函数创建一定数目的线程，每个线程完成一个任务，应用程序经编译后和该库链接，生成单一的二进制固件文件），两者集成在单个文件中，故无法看到熟悉的系统调用（库函数）名，从而使得实际的分析难度更高，目前尚未看到对其安全性的深入分析。

本文关注无文件系统的嵌入式固件（如无特别说明，下文提到的固件均指此类固件）的安全问题，重点对其中的后门检测问题进行深入分析，并对面临的主要挑战和可能的应对措施进行讨论。此类固件由于规模较小，因此出现漏洞的概率确实要小一些，但是并不代表其中不会被故意植入后门。一般的计算机程序运行于通用的硬件平台之上，因此不管是使用物理环境还是采用虚拟化技术模拟执行环境都相对比较容易。而固件通常专用于特定的硬件设备，因此不管是采用物理环境还是模拟环境，最终将固件运行起来都要困难很多。更严重的是，针对一般的计算机程序运行过程的监控（用于获取程序的输出或者内部状态等信息）很容易实现，而监控不同提供商的嵌入式设备通常需要使用其配套的调试设备，因此实现通用的监控机制难度较大。基于上述观察，本文对无文件系统固件后门检测问题采用静态分析为主、动态分析为辅的方法。

一、相关工作

针对后门检测，通常采用动态监控或者静态扫描的方法。例如，Y. Zhang 和 V. Paxson[2] 提出并实现了一种通过被动地监视站点的 Internet 访问链接，并基于包的大小和时间特性检测交互流量的通用算法，可检测运行在非标准端口的标准服务和在标准端口运行的非标准服务后门。而静态扫描的方法主要用于反病毒软件，通过预先提取后门签名，然后采用模式匹配技术确认程序中是否隐藏有后门代码。C. Wysopal 和 C. Eng[3] 讨论了几种典型的后门类型，并介绍了几种基于启发式规则的静态检测方法，本文对固件后门的检测方法主要受该工作的启发。

不管是通过人工逆向还是自动代码分析来剖析和理解二进制代码的行为，有意义的系统调用名字通常可大大降低分析的难度。但在无文件系统嵌入式固件中，无法看到熟悉的系统调用名字，故如何识别重要的库函数将是此类固件后门检测所亟待解决的问题。

对于 Windows PE 文件，通过解析导入地址表（Import Address Table，IAT），IDA Pro[4] 和 Ollydbg[5] 等工具均实现了对系统调用函数的识别（动态链接的运行时库中函数与之类似，不再赘述），可使用有意义的字符串表示每个系统调用函数的名字。除此之外，IDA Pro 中还实现了一种称为快速库辨认与识别的技术（Fast Library Identification and Recognition Technology，简称 FLIRT）[6] 来识别对静态链接的运行时库中的函数调用。FLIRT 的基本思想是对主要编译器（例如 Microsoft 或 Borland 的各种编译器，以及 GNU 的 GCC 等）的静态链接的运行时库中的函数，取其指令序列的前 32 个字节作函数签名（带有简单的冲突处理机制），并保存为不同的签名库文件。当 IDA Pro 反汇编二进制代码时，通过签名匹配技术就可识别已知的库函数，简化反汇编工作，进而简化控制流图和调用图的构造。值得注意的是，IDA Pro 的 FLIRT 签名的制作过程并不是一个完全自动化的过程，当多个函数在预定的签名生成算法下产生签名碰撞的情况下，通常需要手工干预。

通常需要利用反汇编技术 [7][8] 将固件的二进制代码转换为人工容易理解的汇编指令。二进制反汇编主要包括线性扫描（Linear Sweep）和递归遍历（Recursive Traversal）两种。Ollydbg[5] 和 Objdump[9] 工具是使用线性扫描反汇编方法的典型代表。Hex-

Rays 公司的 IDA Pro[4] 是一款基于递归遍历算法且准确率相当高的商业反汇编工具。但由于这两种方法均存在一定的局限性，故对于典型的二进制代码，到目前为止，还没有一款反汇编工具能达到 100% 的准确率和覆盖率。

二、库函数识别

根据上文的分析，库函数识别对于后门检测的成败具有重要的决定作用。实际上，获取准确、完整的反汇编代码不仅对库函数识别具有重要意义，而且在后门检测中同样扮演重要角色。线性扫描反汇编可能取得极高的覆盖率，但其会对任何嵌入在代码中的数据也进行反汇编，而识别所有嵌入在代码中的数据又相当困难，故其准确率通常相当低。基于上述考虑，通过对准确率较高的递归遍历反汇编工具进行优化，提高其覆盖率将是一种可行的方法。

（一）反汇编的覆盖率问题

本文以递归遍历反汇编工具 IDA Pro 为基础，考虑使用静态方法提高反汇编的覆盖率。可能的应对措施包括基于数据流分析的反汇编优化、基于序言 / 结束语的函数识别方法。

• 基于数据流分析的反汇编优化

递归遍历的主要局限性在于遇到间接跳转 / 调用时，难以确定最终的目的地址 / 函数，从而造成大量代码不能识别并反汇编。通过使用常量传播等数据流分析算法，可以确定部分间接跳转 / 调用的目标地址 / 函数，从而提高反汇编的覆盖率。

• 基于序言 / 结束语的函数识别

传统递归遍历难以识别被间接调用的函数。对于源码中的函数，主流编译器生成的二进制通常会具有一些显著的特点，例如函数序言和函数结束语。函数序言是出现在函数代码开始处的指令，函数结束语是出现在函数结束处（返回到调用者之前）的指令。搜索特定的函数序言和与之匹配的函数结束语，是发现被间接调用的函数的一种简单有效的措施。

（二）库函数识别问题

本质上，无文件系统嵌入式固件中也存在库函数。其系统调用通常以静态链接的库函数形式出现，两者指代的对象相同（如无特殊说明，本文提到的库函数即系统调用）。但由于这种类型的固件通常就是单个文件，故通常无法简单地区分库函数与普通函数。这种情况不仅给本文的后门识别研究带来直接的挑战，而且对更广义的漏洞检测也有巨大的影响。

基于上述观察，本文提出了一种针对网络套接字和字符串 / 内存操作函数的基于启发式规则的检测方法，其基本工作流程如图1所示。

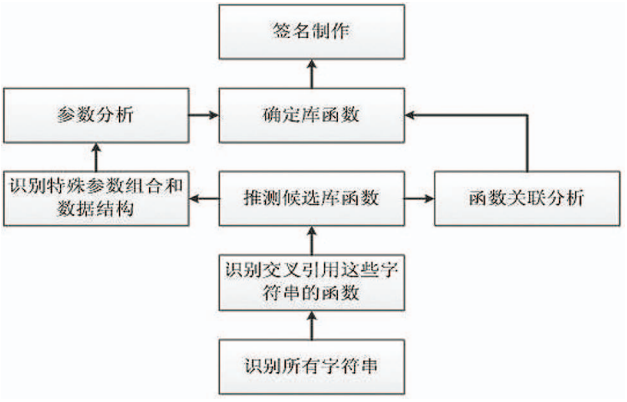


图 1 库函数识别的工作流程

其中，本文使用的主要的库函数识别启发式规则包括基于字符串交叉引用的候选库函数推测和基于参数和函数关联分析的库函数确认。

• 基于字符串交叉引用的候选库函数推测

程序员使用字符串操作函数时，有可能会使用到常量字符串。而为了增加代码的可读性或者单纯的调试目的，调用库函数后也可能对返回值（或者输出参数）进行判断，并且在失败的情况下使用打印函数输出特定的信息，这些信息通常也是硬编码的字符串。通过对感兴趣的字符串的交叉引用进行分析，不难初步推测出部分候选库函数。例如：某固件中调用 recv 函数之后，判断失败的情况下会打印“recv fail:……”形式的字符串；而通过对交叉引用“QUIT”的几个函数的分析很容易确认 strcmp。

• 基于参数和函数关联分析的库函数确认

某些重要函数可能带有特定数目的参数，且通常情况下这些参数的值或者来源是固定的。基于该观察，可以很容易识别出这些库函数。例如，调用 socket 函数时，常用的参数为（ ， ， ）和（ ， ， ），分别表示创建 TCP 和 UDP 套接字；创建 socket 之后，通常随后会以 socket 函数的返回值作参数调用 bind 或者 connect 函数，这时候通常还会填充一个 sockaddr 结构，该结构中的第二个成员表示端口号（例如，Modbus 协议 [10] 的 502 端口号是我们感兴趣的）。

三、后门检测

在计算机科学中，后门特指对计算机系统的未授权访问，是一种严重的安全隐患。需要注意的是，后门既可能是硬件上的，也可能是软件上的。软件形式的后门是指隐藏在程序中（由程序员所创建）的、可在用户未知或者未同意的情况下提供对系统非法访问的代码。本文重点关注未授权侦听者、非预期功能、隐藏功能和向外的连接请求四种常见的固件后门（软件形式）的检测问题。未授权的侦听者是指规范中未规定要使用的端口，非预期的功能是指具体实现与规范描述不符的功能，隐藏功能是指规范中未指定的功能，而向外的连接请求是指主动连接外部结点的网络操作。

由于嵌入式系统对实时性的要求比较高，针对固件后门的检测和防护不应过多依赖于运行时的监控信息，因此本文采用动静结合的方式，如图 2 所示。

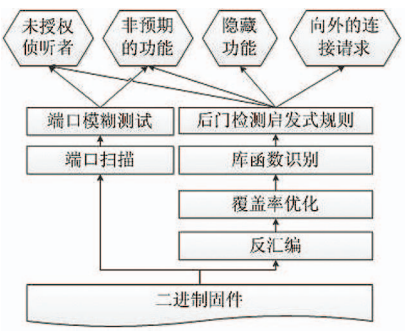


图 2 动静结合的固件后门检测整体工作流程

其中，静态方式主要通过在对 IDA Pro 生成的反汇编代码执行“覆盖率优化”和“库函数识别”的基础上，采用“启发式规则”+“逆向分析”的方法；而动态分析主要依赖“端口扫描”和“端口模糊测试”技术。

（一）后门检测启发式规则

针对上述不同的后门类型，本文分别提出了有针对性的后门检测启发式规则，包括：

• 未授权的侦听者

对 bind 函数做交叉引用和调用参数分析，确定固件中使用的所有未授权端口；并对 recv、recvfrom、send 和 sendto 函数

做交叉引用分析，确定这些未授权端口的网络交互行为。

• 非预期的功能

对 recv 和 recvfrom 函数做交叉引用分析，确定授权端口的消息处理循环。针对规范指定的每种功能，在消息处理循环中确定其处理逻辑，并对处理逻辑进行逆向分析，找出实际功能与其规范的差异。

• 隐藏功能

对 recv 和 recvfrom 函数做交叉引用分析，确定各端口各自的消息处理循环。针对出现在消息处理循环中的多余处理逻辑（而规范中并未描述）进行逆向分析，理解其具体功能。

• 向外的连接请求

对 connect、gethostbyname 和 gethostbyaddr 函数做交叉引用分析和调用参数分析，确定固件中所有向外的连接请求使用的目的地址和端口；并对 recv、recvfrom、send 和 sendto 函数做交叉引用分析，确定这些未授权端口的网络交互行为。

（二）端口模糊测试

端口模糊测试是指对特定端口执行的模糊测试。一般可以先采用端口扫描工具（例如 Nmap[11]）确定系统打开的端口（侦听端口）。若存在打开的未授权端口，则检测到未授权的侦听者。再针对打开的授权端口（例如 Modbus 的 TCP 502）进行模糊测试 [12][13]。端口模糊测试可以检测非预期功能类型的后门，其具体检测流程如下：

1. 根据使用不同端口的协议（例如 Modbus 的 TCP 502）自身的特点，选用特定的模糊测试数据生成器和异常检测规则。

2. 将生成器生成的数据发送到协议授权使用端口，并记录每个请求和相应的响应数据。

3. 根据异常检测规则对请求 / 响应交互序列进行过滤，找出实际响应与协议规范描述存在偏差的请求以及未收到任何响应的请求。

4. 人工对过滤后的请求 / 响应交互序列进行细致分析。

对于打开的未授权端口，也可以执行模糊测试，并记录每个请求和相应的响应数据做进一步分析。但由于缺乏对端口的先验知识，效果通常不容乐观。

四、实例分析

本节以一个完整的例子验证和演示上述后门检测方法的实际应用过程和检测效果。其中，嵌入式设备选用霍尼韦尔的 HC 900 远程终端系统 [14]，固件为原始文件（无文件头，指令从首字节开始），基于 ARM 架构指令集，采用大头序，版本为 4.4。

（一）反汇编覆盖率

目前有多种反汇编工具可以对其进行反汇编。本文采用 IDA Pro 作为基础反汇编工具。通过观察发现，HC 900 固件中的绝大多数函数均有规则的序言和结束语（“STMFD” 和 “LDMFD”）。基于该观察，使用基于序言 / 结束语的启发式函数识别方法，对其覆盖率进行的二次优化实验，结果如表 1 所示。基于数据流分析的反汇编优化方法目前仍在编码阶段，暂无实验数据。

表 1 提高反汇编覆盖率的启发式规则

启发式规则	IDA Pro	序言 / 结束语	优化效果
# 识别的函数	2804	3627	823
覆盖率（%）	75.1%	97.2%	29.4%

覆盖率 = # 识别的函数 / # 确认的总函数

其中，总函数数量为 3731，系通过人工逆向分析结果估计。具体实施方法为：1）

由人工确认未反汇编的字节片段是否是数据；2）然后在所有非数据的字节片段执行递归遍历反汇编；3）通过几次迭代，当无法继续时，停止反汇编，并假定最终识别的函数数量即为程序中总的函数数量。该方法仍可能将某些数据识别为代码，但是由于是人工参与，故引入错误的概率极小，将其作为估算固件中总函数数量的依据是合理的。

（二）库函数识别

熟悉的系统调用和库函数名字对于增加程序的可读性、降低分析难度具有积极的作用。本文重点关注网络套接字、字符串 / 内存操纵函数的识别，采用的方法主要有字符串交叉引用、参数分析和函数关联分析三种。识别的部分主要函数和识别的方法如表 2 所示。

识别上述函数的过程中，我们发现了明显的静态链接库函数的特征：网络套接字函数位于一段连续的代码区域，而字符串和内存操作函数位于另一段代码区域（来自两个不同的静态链接库）。基于该发现，我们对这些函数制作了 FLIRT 签名。在对 HC 900 之前版本（4.3、4.2、4.1）固件分析过程中，

表 2 识别的库函数

函数	类型	字符串交叉引用	参数分析	函数关联分析
accept, bind, connect, getsockname, getsockopt, listen, select, setsockopt, socket	网络套接字		√	√
recv, recvfrom, send, sendto	网络套接字	√		
malloc, memset, memcmp, memmove, memcpy	内存		√	√
strcat, strchr, strcmp, strcpy, strcspn, strlen, strncat, strncmp, strncpy, strspn, strstr	字符串	√		

这些函数签名均得到了匹配，再次验证了其库函数的特性。

（三）后门检测

针对 HC 900 的 4.4 版本固件，根据上述检测方法，对未授权侦听者、非预期功能、隐藏功能和向外的连接请求四种典型的后门类型进行了检测和分析。实验结果显示，该固件中未发现向外的连接请求，但存在未授权侦听者、非预期功能、隐藏功能三种类型的后门。具体情况如下：

1. 使用 Nmap 扫描的结果显示，除 Modbus 协议规定使用的 TCP 502 端口外，HC 900 还打开了 TCP 和 UDP 的 7777 端口。利用静态启发式规则的分析同样验证了该结果。

2. 采用“端口模糊测试”对 TCP 502 端口进行测试的结果显示，0x14 和 0x15 功能与协议规范描述存在较大偏差。通过对固件 TCP 502 端口命令处理循环的逆向分析显示，该固件仅实现 Modbus 协议规定的部分功能码 0x01-0x06、0x08、0x10、0x11、0x14 和 0X15。

对该命令处理循环中每个功能码对应的处理逻辑进一步分析显示：其 0x14 和 0x15 两个功能码，并没有实现规定的功能；尤其是 0x15 的实现存在严重后门，通过发送精心构造的数据，可以造成 HC 900 系统停止采集数据，并向用户发送攻击者设定的数据，如图 3 所示。

3. 对 TCP/UDP 7777 端口的命令处理循环的逆向分析显示，TCP 7777 端口接受“R”（写

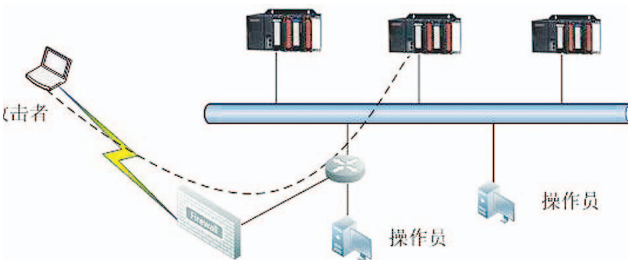


图 3 HC 900 系统 0x15 实现后门

入）、“T”（读取）和 “X”，UDP 7777 端口接受 “B” 和 “T” 命令。

五、结束语

嵌入式固件与计算机软件运行环境的差异，使得针对固件的分析通常采用静态的方法。而无文件系统嵌入式固件与带文件系统的嵌入式固件在库函数上的显著差异，又使得对其进行静态分析难度更大。

本文以二进制无文件系统嵌入式固件为研究对象，讨论了其中的库函数识别和后门检测问题，成功识别了一款真实固件中主要的网络套接字、字符串 / 内存操纵函数，并检测到多个后门（包括一个严重级别的后门）。实验表明，本文提出的针对无文件系统嵌入式固件的库函数识别方法对于该类固件的安全性分析具有重要的参考价值。

参考文献

[1]ThreadX. http://rtos.com/products/threadx

[2]Y. Zhang, V. Paxson. Detecting Backdoors [C]. Proceedings of the 9th USENIX Security Symposium, 2000.

[3]C. Wysopal, C. Eng. Static Detection of Application Backdoors [C]. Blackhat USA, 2007.

[4]IDA Pro. https://www.hex-rays.com/products/ida/index.shtml

[5]OllyDbg. http://www.ollydbg.de

[6]I.Guilfanov. http://www.hex-rays.com/idapro/flirt.htm

[7]B. Schwarz, S. Debray, G. Andrews. Disassembly of Executable Code Revisited [C]. IEEE Working Conference on Reverse Engineering, USA:IEEE Computer Society, 2002:45-54.

[8]C. Kruegel, W. Robertson, F. Valeur, G. Vigna. Static Disassembly of Obfuscated Binaries [C]. USENIX Security Symposium, USA: USENIX, 2004(13)

[9]GNU Binutils. http://sourceware.org/binutils/docs/binutils/objdump.html

[10]Modbus IDA. http://www.modbus.org/specs.php

[11]Nmap. http://nmap.org

[12]A. Takanen, J. DeMott, C. Miller. Fuzzing for Software Security Testing and Quality Assurance [M]. USA:Artech House Inc., 2008: 22-32

[13]J. Duran, S. Ntafos. An Evaluation of Random Testing [J]. IEEE Transactions on Software Engineering, 1984, SE-10(4): 438-444

[14]Honeywell.https://www.honeywellprocess.com/en-US/explore/products/control-monitoring-and-safety-systems/modular-control-systems/hc900-control-system/Pages/hc900-controller.aspx

移动互联网环境下的DDoS攻防

安全研究部　洪海

关键词：DDoS 攻防 移动互联网

摘要：随着移动互联网的快速发展，DDoS 攻防在新环境下也面临着一些新的变化和挑战。本文从三个方面对移动互联网环境下的 DDoS 攻防问题进行探讨。

一、概述

在移动互联网出现以前，传统的网络访问过程是由普通的 PC 主机通过运营商提供的有线宽带网络访问 Web 等服务器，而 DDoS 攻击者也是通过同样的路径进行攻击。对于传统网络环境下的 DDoS 攻击，我们已经十分熟悉，现有的方案也已经能够进行比较有效的防护。

在移动互联网出现之后，一种新的网络访问过程产生了。在移动互联网环境下，智能设备通过运营商提供的移动网络来访问 App 服务器成为一种常见的网络访问方式。然而，移动互联网并不仅仅增加了这一种信息传递路径。例如，智能设备可以通过 Wi-Fi 接入传统的宽带网络，访问 App 服务器或者传统 Web 服务器；而传统的 PC 主机则可以通过 3G 无线网卡等设备接入移动网络，访问传统的 Web 服务器或通过特定方式访问 App 服务器。同样，DDoS 攻击者也可以使用这些路径当中的任一路径进行 DDoS 攻击。在这种情况下，DDoS 攻击产生了一些新的变化，现有的抗 DDoS 方案也面临新的挑战。

为便于讨论，我们将整个网络访问 / DDoS 攻击的过程分成图 1 所示的客户端、数据网络和服务器三个部分，分别探讨移动互联网中每个部分给 DDoS 攻防带来的新变化与新挑战。

二、客户端

随着移动互联网的快速发展，智能设备的计算能力和设备数量都得到了极大的提高。主流的手机和平板等智能设备，不论是 Android 设备还是 iOS 设备，其 CPU 都已经达到了多核 1.5GHz 以上的运算速度，RAM 容量也都在 1GB 以上，其计算能力已经与 PC 机不相上下。智能设备的数量也在飞速增长，根据官方给出的数据，Android 设备的数量已经超过 9 亿台，iOS 设备的数量也已经超过了 6 亿台，作为对比，2013 年全球 PC 数量为 17.8 亿台，可见，仅 Android 设备和 iOS 设备的总和就已经接近

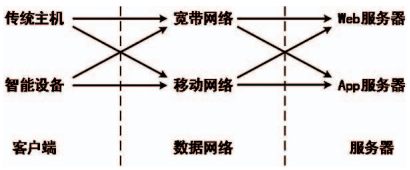


图 1 网络访问 /DDoS 攻击的三个组成部分

全球 PC 的数量，而移动设备的数量依然在高速增长。快速增长的计算能力和设备数量，使智能设备具有了发动 DDoS 攻击的可能性。

同时，与 PC 相比，智能设备的安全防范较差，更容易成为僵尸设备。现有的智能设备主要是 Android 和 iOS 两大平台。Android 设备由于操作系统和生态环境的开放性等原因，很容易被植入恶意代码成为僵尸设备；iOS 设备相对比较封闭，但进行越狱后依然能够从第三方源安装程序，存在成为僵尸设备的可能。不过，即使是从官方应用市场上安装的应用，也依然有可能是逃避过检查的恶意应用。更重要的是，智能设备的用户通常没有较强的安全意识，不能够意识到这些智能设备与 PC 并没有本质的区别，导致恶意软件趁虚而入并控制设备发动攻击。

由于智能设备的这些特性，DDoS 攻击工具和僵尸程序已经逐渐出现在了智能设备上。

一些 DDoS 攻击工具以压力测试工具的名义出现。例如，著名的 LOIC（Low Orbit Ion Cannon，低轨道离子炮）就以压力测试工具的名义出现在 Google Play 上，可以进行 TCP、UDP 和 HTTP 洪水攻击，另一款名为 AnDOSid 的应用则可以发动 HTTP POST 洪水攻击。这些工具都非常简单易用，只需要输入目标就可以开始进行攻击。

而 Android.DDoS.1.origin 等恶意软件也瞄准了智能设备，通过伪装、诱骗等方式吸引用户点击运行，在后台连接 C&C 服务器，并等待命令发动攻击。

可以说，目前的智能设备计算能力强、设备数量多、安全性差，

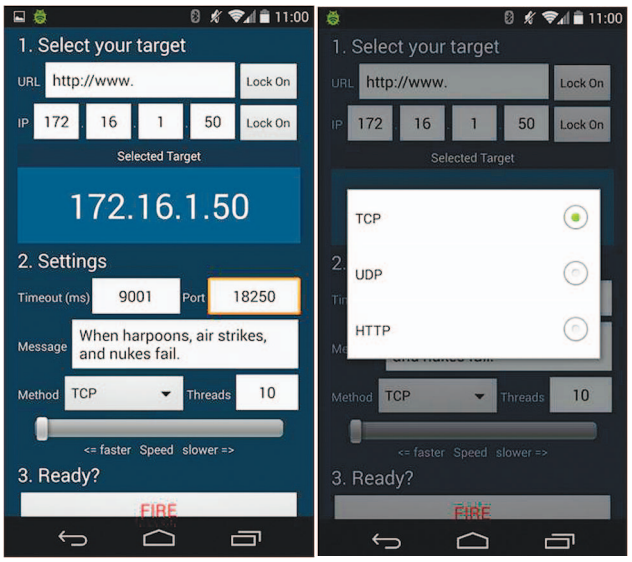


图 2 Android 上的 LOIC 工具

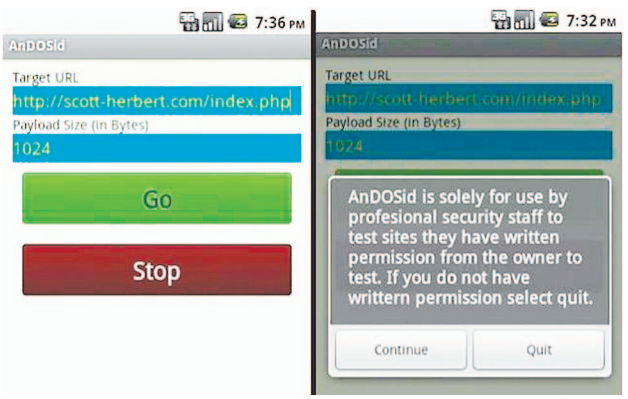


图 3 Android 上的 AnDOSid 工具

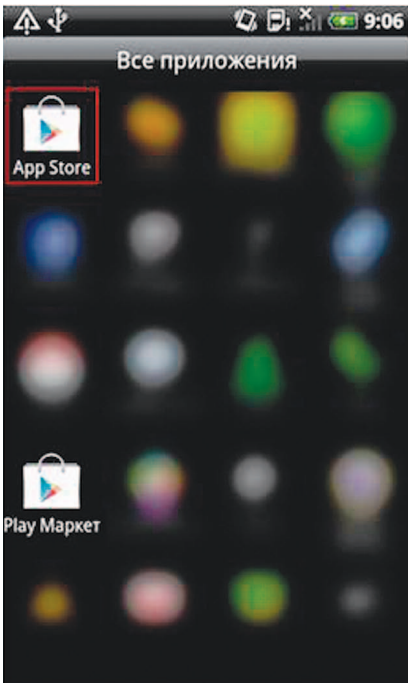


图 4 Android.DDoS.1.origin 恶意应用伪装成应用商店

已经具备了形成僵尸网络发动 DDoS 攻击的可能性。随着移动互联网的不断发展，由智能设备组成僵尸网络的风险会越来越大。不过，目前这种类型的僵尸网络还不多见，其恶意行为多以信息窃取为主，即使发动 DDoS 攻击也与传统 PC 上的攻击方式没有太大的区别，尚未对 DDoS 攻防的现状造成太大影响。

三、数据网络

移动数据网络传输速度的提升与移动互联网的快速发展具有密不可分的关系，同时网络带宽的提升也为通过移动网络发动 DDoS 攻击提供了可能性。

与 2G 网络相比，3G 网络在数据传输速度上有了很大的提升，其下行速度可以达到 2Mbps 以上，上行速度可达 384kbps 以上。此外，部分国家和地区的运营商已经部署了实用的 3.5G（下行速度 14Mbps，上行速度 5.8Mbps）和 4G（下行速度 1Gbps，上行速度 500Mbps）通信网络，甚至超过了一般的传统宽带网络的传输速度，通过这些高速网络进行 DDoS 攻击是完全可能的。

目前，还没有通过移动网络发起大规模 DDoS 攻击的案例，这是因为移动数据网络的流量资费还相对较高，用户对于流量的消耗也比较敏感。不过，随着数据流量资费的不断下调以及部分运营商推出的不限流量等套餐，通过移动网络发起的 DDoS 攻击流量将会不断增长。

虽然移动网络上的数据传输同样是基于 TCP/IP 协议进行的，但与传统的宽带网络相比，依然存在一定的差异性，这些差异会对当前的 DDoS 防护方案产生一定的影响。

例如，移动网络的一个特点是对于接入的终端大量使用 NAT 方式进行地址分配。由于智能设备数量的不断增长和 IPv4 地址数量耗尽，国内的三家运营商目前都会给接入其网络的设备分配一个私有的 IP 地址，设备在进行通信时通过 NAT 方式与互联网上的其他设备进行连接。在这种情况下，大量的设备会使用同一个网关 IP 地址与服务器进行通信，而如果依然以 IP 地址作为 DDoS 攻击防护和清洗机制的话，就会产生大量的误伤或漏报，影响 DDoS 防护的效果。

不过，随着 IPv6 的逐渐应用和推广，上面的问题会逐步得到解决。IPv6 网络的地址空间非常大，能够从根本上解决目前 IP 地址短缺的问题。同时，IPv6 网络环境下也不再有 NAT 机制，网络变得更加扁平化，每一台设备进行网络访问的时候都使用不同的公网 IP 地址，这样 IP 地址依然可以作为 DDoS 攻击的防护和清洗依据，而现有的防护方案不受到影响。

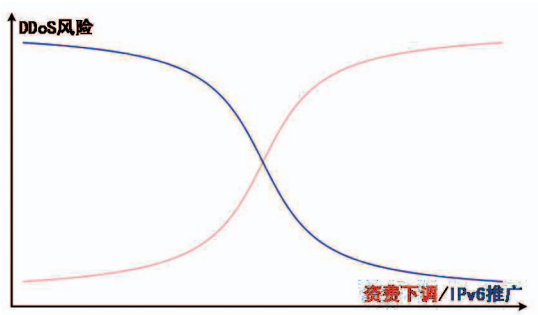


图 5 通过移动网络发起 DDoS 攻击的风险变化

除大规模使用 NAT 带来的问题外，移动数据网络还有一些其他的特性可能导致现有防护方案失效或产生大量误杀（例如由于传输媒介不同导致网络延时增加等问题），这需要对现有的防护方案做进一步的测试和改进。

四、服务器

移动互联网环境下，DDoS 攻防变化的另一个主要方面体现在对于服务器防护难度的增加。

对于传统的 Web 服务器，防护应用层 DDoS 攻击原理如图 6 所示。

当用户使用浏览器访问 www.example.com 时，访问请求会被发送到防护设备，防护设备会将请求重定向到验证页面。在验证页面，防护设备会使用 JavaScript 等脚本语言发送一条简单的运算请求，或者在验证页面生成一个验证码，以检查访问请求是否是由真实的浏览器 / 用户发出的。

如果请求是由真实的浏览器发出的，那么浏览器会正确地计算出运算结果并返回；如果请求是由真实的用户发出的，那么用户就能够填写正确的验证码。此时防护设备再将浏览器的访问请求重定向到 Web 服务器上的 www.example.com/thisurl 上。经过这个过程，正常用户即可进行有效的访问。

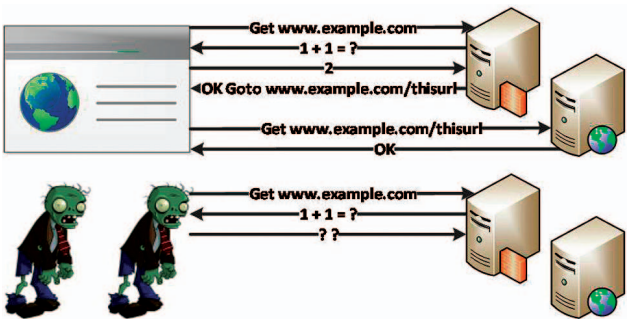


图 6 传统 Web 服务器应用层 DDoS 攻击的防护原理示意图

而如果请求是由发起 DDoS 攻击的僵尸主机发送的，由于僵尸程序通常不能进行 JavaScript 等脚本的执行，也无法自动识别出验证码的内容，因而不能返回正确的结果，这时防护设备会丢弃访问请求，而不会给出跳转到 Web 服务器的链接。因此，当发生应用层 DDoS 攻击时，大量僵尸程序发出的访问请求会被防护设备直接丢弃，不会对真正的服务器造成影响。

然而，对于移动互联网环境下的 App 服务器，使用传统防护方法则有可能造成对合法应用访问的拒绝服务。

大部分移动 App 的 Get 请求取回的数据不是一个页面，而是一个 XML 或者 JSON 结构，App 在得到数据后进行解析并展示即可。因此，大量的移动 App 并不是基于浏览器页面进行交互的，在开发

的过程中也就不需要加入对 JavaScript 脚本执行的支持。当防护设备将访问重定向到验证页面时，App 能够得到该页面的内容，但没有办法执行并展示其中的 JavaScript 脚本，无法给出正确的验证结果。这时，访问请求就会被当成僵尸程序的访问请求直接丢弃，导致无法正常获取服务内容。

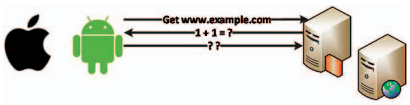


图 7 App 服务器无法使用传统方案进行 DDoS 防护

可以看到，当 App 服务器遭到应用层 DDoS 攻击时，如果不开启防护设备，则大量的僵尸程序请求会耗尽服务器资源，导致正常应用请求访问缓慢或无法访问；而如果开启防护设备的现有防护方法，则会直接将正常应用的访问请求当作僵尸程序发起的请求丢弃，正常用户依然无法访问。现有的 DDoS 防护方案面临着巨大的挑战。

虽然目前还没有方法能够完全解决这个问题，但在发生了针对 App 服务器的应用层 DDoS 攻击时，可以通过一些技术手段进行一定程度的缓解。

通常可以认为，对于 App 服务器的合法访问绝大多数来源于移动设备上的 App。这是因为 App 的 Get 请求取回的数据通常不是一个页面，而是一个 XML 或者 JSON 结构，App 能够对这个结构进行解析和展示，而使用浏览器进行访问通常却没有意义。因此，在发生应用层 DDoS 时，可以通过检查请求来源的方法，丢弃明显不合理的请求包。

例如，可以通过检查 App 的指纹信息来过滤掉明显非法的访问请求。通常，App 的请求格式比较单一固定，不具有浏览器访问的格式多样性，也不需要为非官方的 App 提供兼容。最简单的 App 指纹是 HTTP 请求中的 User-Agent 字段，大多数的移动 App 都会定义自己的 UA，可以将其作为最明显的指纹特征。除了 UA 以外，一部分应用还可能在 HTTP 请求的 Header 中加入其他的头部字段，这些字段的存在与否以及这些字段的顺序关系都可以作为 App 的指纹信息。建立指纹信息后，当发生应用层 DDoS 攻击时，就可以将不符合 App 指纹信息的请求直接丢弃，以对 DDoS 攻击产生一定的缓解作用。

但是，HTTP 请求中的 Header 字段是能够被伪造的，攻击者只要知道应用的请求格式，就能够伪造出完全符合的 HTTP 请求头部。因此该方法只能对一部分通用的攻击工具进行防护。

对于 App 服务器的 DDoS 攻击，最根本有效的解决办法是在 App 应用中添加对脚本或验证码的支持，以便使防护设备能够分辨出僵尸程序和合法用户的区别，从而有效地屏蔽和丢弃僵尸程序发起的访问请求。

五、总结

随着移动互联网的快速发展，DDoS 攻防在新环境下面临着一些新的变化：从智能设备上组建僵尸网络、发起 DDoS 攻击的风险正在不断增大；移动网络与传统网络特性上的一些差异对 DDoS 防护提出了新的挑战；同时，针对 App 服务器的应用层 DDoS 攻击防护难度大大增加。面对这些问题，需要运营商、App 开发者、安全厂商加强沟通与合作，才能够形成一个好的解决方案，减少 DDoS 攻击带来的损失。

基于Web访问日志的异常行为检测

安全技术部　姚伟

关键词：异常行为 Web 日志 特征匹配 统计分析

摘要：本文介绍了在海量 Web 访问日志中，使用特征字符匹配、访问频率统计分析、机器自学习等方法去挖掘攻击行为，提高安全威胁的检测能力。

一、前言

随着互联网的快速发展，在经济利益的趋势下，发生了越来越多的网络安全事件，使得互联网安全正在面临前所未有的挑战。攻击者也从单一的攻击行为，发展成有组织、有目标、持续时间长的攻击，新名词 APT 攻击也越来越多的被人们熟悉。攻击者为了获得某个组织甚至国家的重要信息，会利用多种攻击手段，甚至很多攻击利用的漏洞还未公开，在攻击过程中也会使用各种技巧，长期潜伏在系统中，不断收集各种信息，最终达到目的。

对于重要机构和企业而言，虽然已经部署了大量针对某类威胁的安全防护设备，但很多安全设备还是主要依赖于提取漏洞特征的方式检测攻击，如果碰到未公开的漏洞被黑客利用、厂商未及时发布升级包或者管理员未及时更新程序等几种情况，都会导致设备无法检测和防范攻击。

为了解决传统安全设备的不足，经过安全技术的积累，考虑通过 Web 访问日志分析的手段，对 Web 用户的行为进行异常监控，发现可疑行为，从而进一步追查原因，及时采取措施应对问题。

二、日志分析流程

1. 日志采集

Web 访问日志有两种采集方式，一种是直接导入 Web 服务器的访问日志，另一种是从 Web 应用防火墙的访问日志接口导入，导入的是经过 Web 应用防火墙清洗后的 HTTP 数据。

2. 日志入库

采集到的 Web 访问日志将根据字段进行切割，保存到数据库中等待分析。

3. 数据分析

主要通过前面介绍的几种分析方法，编写算法和规则，进行海量数据分析，然后报告可疑行为。

4. 结果展现

将分析的可疑行为，以图表、地图定位等可视化的方式展现给安全管理员。

三、访问日志结构

访问日志是 Web 服务器 (例如 Apache、IIS) 记录用户访问行为产生的文件，标准的 Web 日志是纯文本格式，每行一条记录，对应客户端浏览器对服务器资源的一次访问。典型的日志包括来源地址、访问日期、访问时间、访问 URL 等丰富的信息，对日志数据进行分析，不仅可以检测到可疑的漏洞攻击行为，还可以提取特定时间段、特定 IP 对应用的访问行为。

由于不同 Web 服务器的日志格式不同，以 Apache 访问日志为例，介绍日志组成字段及字段的意义。

上述日志记录了来自 134.147.23.42 的地址，在北京时间 2012 年 3 月 13 日 20 点 58 分 25 秒，对服务器的 /index?page=news 页面进行了一次访问，访问使用 GET 方式，使用 HTTP 1.1 协议，返回状态码是 200(页面存在，请求成功)，返回的页面大小是 36312 字节。

Apache 日志格式可以通过编辑 httpd.conf 文件自定义，默认有简单和复杂两种格式 :

Common Log Format

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access_log common
```

Combined Log Format

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-agent}i\"" combined
CustomLog log/access_log combined
```

combined 比 common 格式多两个字段 referer 和 user-agent，也可以自定义格式，各字段含义如下 :

表 1 Apache 日志配置字段含义

字段	描述
%h	客户端地址
%l	identd 登录名
%u	HTTP 认证用户名
%t	访问时间
%r	访问方法、访问 URL、协议版本
%>s	返回状态
%b	返回大小
%{Referer}i	来源站点
%{User-Agent}i	浏览器版本

四、常见分析方法

1. 特征字符分析 (Signature-based)

此方法是在日志中查找已知的漏洞特征，去发现黑客攻击行为，是最简单的方法。

例如，可以通过匹配扫描器请求 URL 中的特征，检测漏洞扫描行为 :

2013-05-06 08:53:54 192.168.16.193 GET /include/functions.php path=http%3A%2F%2Fdownload.boulder.ibm.com%2Fibmdl%2Fpub%2Fsoftware%2Fdw%2Fwatchfire%2Fphpinfotest.txt%23 80 -

Mozilla/4.0+(compatible;+MSIE+7.0;+Windows+NT+5.1;+Trident/4.0;+InfoPath.2;+.NET+CLR+2.0.50727;+NET+CLR+3.0.4506.2152;+.NET+CLR+3.5.30729;+.NET4.0C;+.NET4.0E) 404 0 3

通过匹配 URL 中的 SQL 关键词，检测 SQL 注入攻击：

2013-04-25 09:05:05 GET /cart.aspx

act=buy&id=1%20and%20(Select%20Top%201%20char(124)%2BisNull(cast([Name]%20as%20varchar(8000)),char(32))%2Bchar(124)%2BisNull(cast([Pass]%20as%20varchar(8000)),char(32))%2Bchar(124)%20From%20(Select%20Top%204%20[Name],[Pass]%20From%20[Web_Admin]%20Where%201=1%20Order%20by%20[Name],[Pass])%20T%20Order%20by%20[Name]%20desc,[Pass]%20desc)>0%20-- 192.168.16.12 200

Mozilla/4.0+(compatible;+MSIE+8.0;+Windows+NT+5.1;+Trident/4.0;+InfoPath.2;+.NET+CLR+2.0.50727;+.NET+CLR+3.0.4506.2152;+.NET+CLR+3.5.30729;+.NET4.0C;+.NET4.0E) guid=6f3f8975-78ce-423a-9e93-a36bd73996a0;+V5ShopLoginID=dYGYgYN/udIoOMSzwIWWmvJPZDWQ80D/3oUfdZw28RM= 192.168.16.191 - 1105 973 46

2. **访问频率分析**（Frequency analysis）

在黑客攻击过程中，需要对系统进行各种特定的访问，这些访问与正常用户访问有很大差别，每种攻击行为都有不同的特征。通过对大量用户访问数据的挖掘，可以发现这些异常访问行为。

	访问时间	来源地址	目标 URL	来源 URL	返回状态
漏洞扫描检测	高频率	固定	离散		404
暴力破解检测	高频率	固定	登录页		200
webshell 检测	低频率	固定	固定		200
脱数据库检测	高频率	固定	参数变化较为规律		200
账号盗用检测		非常用 IP	登录页		200

表 2 几种常见的异常访问特征

（1）漏洞扫描检测

黑客使用漏洞扫描器对 Web 应用进行扫描，可以用匹配 User-Agent 特征的方式进行检测。如果自定义扫描器的 User-Agent，这个方法的效果可能会不好。但可匹配扫描器扫描的行为，如访问目标离散、来源地址相对固定，访问结果大多数失败。根据这些特征对 Web 访问日志进行分析，即可提取出来可疑的扫描行为。

（2）暴力破解检测

暴力破解密码的特征是相对固定的来源地址，对登录 URL 短时间内高频率发起请求，与漏洞扫描的区别主要是目标 URL 固定。

（3）webshell 检测

如果黑客发现系统漏洞，并且利用漏洞获得上传权限，会向系统上传 webshell。webshell 是一种后门程序，此程序由脚本语言编写，可以在 Web 服务器上运行，攻击者可以通过网页执行系统命令，读写系统文件。从访问行为的角度看，webshell 通常只有攻击者访问，来源地址相对固定，访问时间相对集中，无内嵌其他页面。通过这些特征即可提取出可疑文件，再通过人工确认的方式，检测出 webshell。

（4）脱数据库检测

假设某Web应用，用户通过访问http://xxx.com/accountprofile?userid=1100101可以查看到当前用户信息，如果应用存在未授权对象引用漏洞，攻击者可以遍历userid来访问其他账户信息，提交如下一系列请求：

http://xxx.com/accountprofile?userid=1100101

http://xxx.com/accountprofile?userid=1100102

http://xxx.com/accountprofile?userid=1100103

黑客可以利用此漏洞编写程序，通过批量请求获得大量账户信息，这种脱库行为从访问URL上看，userid参数值的变化有明显的特征，可以通过此特征进行日志分析检测。

（5）账号盗用

如果Web应用管理员登录后台的源地址较为固定，可以通过日志分析，提取出不在经常登录范围内的地址，进行账号盗用的判断。例如，管理员平时登录Web应用后台，都是在单位和家里，这两个地方都有相对固定的IP地址范围，如果某一天的登录不在此范围内，例如分析时出现了一个外地的或者国外的IP地址来源，则可能是账号被盗用后进行的登录行为。

3. 机器自学习（Machine Learning Based）

机器自学习是一种比较高级的检测方法，分为两个阶段：

第一阶段是样本学习阶段，检测系统通过对大量真实样本的学习，提取出特征，构建合法访问知识库。

第二阶段是实时检测阶段，检测过程依赖于学习阶段建立的知识库，一旦特征超出某个范围，则判定为可疑行为，然后进行告警。

例如，用户访问网站新闻、博客等栏目首页时，返回的数据大小在几万字节范围内，当某个访问企图通过安全漏洞读取系统密码文件时，返回的数据大小可能会不在几万字节范围内，系统则进行告警。

```
134.147.23.42 - - [13/Mar/2012:20:58:25 +0100] "GET
/webapp.php?page=news HTTP/1.1" 200 36312
134.147.61.15 - - [13/Mar/2012:21:02:13 +0100] "GET
/webapp.php?page=blog HTTP/1.1" 200 27140
134.147.12.77 - - [13/Mar/2012:20:58:25 +0100] "GET
/webapp.php?page=index HTTP/1.1" 200 30745
134.147.12.77 - - [13/Mar/2012:20:58:29 +0100] "GET
/webapp.php?page=news HTTP/1.1" 200 36312
212.32.45.167 - - [13/Mar/2012:21:05:42 +0100] "GET
/webapp.php?page=../../etc/passwd HTTP/1.1" 200 2219
134.147.12.131 - - [13/Mar/2012:20:58:29 +0100] "GET
/webapp.php?page=wiki HTTP/1.1" 200 73141
```

此方法使用过程中，需要不断根据结果进行修正，更新知识库来提高检测的精度。

五、应用场景展现

1. 数据泄露事件调查

2013年某天接到客户电话，描述在某论坛发现有网友发布了大量该客户的用户数据，初步判断可能是某业务系统被黑客非法登录，然后批量下载了用户数据。

工程师到达现场后，查看了防火墙、IDS、Web应用防火墙日志，均未发现攻击记录，又对Web应用进行了安全检测，也未发现明显的Web漏洞。跟业务人员沟通后得知未使用123456等弱口令。

初步检查未果后，工程师将客户Web服务器的访问日志导入数

据分析平台，进行分析后发现，大约在 1 周前，周三至周五每天晚上都有大量的请求后台登录 URL 的访问记录，系统对请求的源 IP 进行归并、提取分析，最终定位了 1 个成功登录过系统的最可疑的 IP，并且归属地是韩国。

将此 IP 的访问记录筛选出来，分析梳理出攻击流程：

2. 新漏洞引发的攻击潮

利用 Web 访问日志，除了能用于异常行为特征检测出黑客攻击，还能对历史攻击进行追溯。以下是在一次应急响应中，在客户某台 Web 服务器上提取到的，struts 命令执行漏洞产生的 Web 访问日志。

X.X.X.X - - [10/Mar/20XX:16:24:18 +0800] "GET
/struts2/login.action?redirect:$%7B%23a%3d(new%20
java.lang.ProcessBuilder(new%20java.lang.
String%5B%5D%20%7B'whoami'%7D)).
start(),%23b%3d%23a.getInputStream(),%23c%3dnew%20
java.io.InputStreamReader%20(%23b),%23d%3dnew%20
java.io.BufferedReader(%23c),%23e%3dnew%20
char%5B50000%5D,%23d.read(%23e),%23matt%3d%20
%23context.get('com.opensymphony.xwork2.dispatcher.
HttpServletResponse'),%23matt.getWriter().println%20
(%23e),%23matt.getWriter().flush(),%23matt.getWriter().
close()%7D HTTP/1.0" 200 50002

由于客户的业务系统数量很多，漏洞刚刚被爆出，客户很希望尽快确认其他 Web 应用是否也被黑客攻击过。通过提取特征，创建规则，在最近一个月的日志中进行排查，最终确定了十几台被攻陷的服务器。经过分析发现，这些服务器都使用了低版本的 struts 框架，框架漏洞被自动化工具所利用，导致了大量服务器被攻击。发现问题后，协助客户及时进行了系统恢复，避免了对业务的影响。

六、结论

本文中提出了使用 Web 访问日志分析的方法检测异常行为，可以利用此方法实现一个 Web 访问日志分析系统，配合传统 Web 应用防火墙进行安全防范。实际使用中，外部用户的访问首先到达 Web 应用防火墙，由 Web 应用防火墙进行检测、过滤，干净的流量进入 Web 服务器。通过对 Web 访问日志分析，不仅能补充检测 Web 应用防火墙没有检测到的攻击，也可以在出现新漏洞后，及时统计哪些 Web 应用受到影响，从而切实提高网络威胁检测和防御能力。

参考文献：

1.Apache Documentation，http://httpd.apache.org/docs/2.4/logs.html

2. 钱秀槟，李锦川，方星，信息安全事件定位中的 Web 日志分析方法

3.jens muller,web application forensics httpd logfile security analysis

Web安全之验证码小结

核心技术部　周大　刘亚

关键词：验证码 Web 安全 CAPTCHA

摘要：为防止服务器端的资源被客户端的计算机程序滥用或攻击，服务器需要区分当前用户是计算机还是人类，一般在网站的关键操作位置都会采用验证码技术来区分。围绕验证码展开的攻防技术在 Web 安全中有着重要地位，本文将从验证码的工作原理出发，介绍验证码实现上容易产生的问题，并对其攻防技术现状和未来做一个简要介绍。

1. 前言

全自动区分计算机和人类的图灵测试（Completely Automated Public Turing test to tell Computers and Humans Apart，简称 CAPTCHA），俗称验证码，是一种区分用户是计算机或人的全自动化程序。在 CAPTCHA 测试中，作为服务器端的计算机会自动生成一个问题由用户来解答。这个问题可以由计算机生成并评判，但是必须只有人类才能解答。由于计算机无法解答 CAPTCHA 的问题，所以回答出问题的用户就可以被认为是人类。

在以前的网络访问中，还不存在验证码的用法，但由于网络更加深入地融合到人们的生产生活中，暴力猜测登录、垃圾广告贴等在网络中泛滥，消耗了大量的服务器资源，同时也可能威胁到服务器的安全，验证码作为一种实用高效技术被大量使用起来。

Web 安全的攻击与防护技术一直是互相促进的，验证码识别技术的不断发展推动着验证码生成技术的提高。本文将从验证码技术的原理出发，介绍验证码容易出现的问题以及相关识别技术，以使读者对验证码攻防技术有一个了解。

2. 工作原理

常见的图形验证码是与 Web 中的会话相关联的，在一个会话开始时，在需要使用验证码的地方会生成一个与当前会话相关的验证码，用户识别出验证码后通过填写表单将数据提交给服务器，服务器端会验证此

次会话中的验证码是否正确。具体来说，其工作流程如图 1 所示。

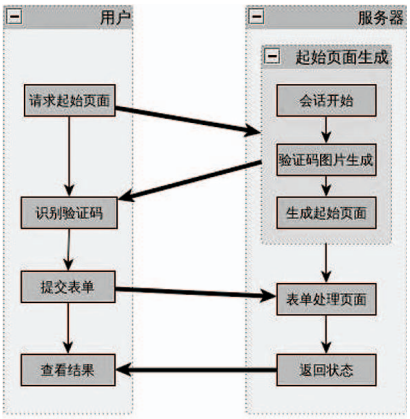


图 1 验证码技术工作原理

对用户方来说，用户访问起始页面，识别返回页面中的验证码，在输入验证码和其他信息后提交表单，在服务器处理后可查看到当前操作是否成功。

服务器在接收到用户对初始页面的请求后，会自动创建一个新的会话，同时生成验证码来关联这个会话，并且生成用户可见到的验证码图片，最后这些页面返回到用户的浏览器上，用户此时可看到完整的页面。在接到用户提交的表单请求时，服务器会比较用户提交的验证码值，并与之前存储在此会话下的验证码值做比较，如果一致判断验证码是正确，否则认为提交的验证码是错误的，可能客户端是计算机或者用户识别错误。服务器端进行这些处理后将处理结果反馈给用户。如果提交的验证码是正确的，则按照预定流程进行下一步骤，否则回到需要用户输入的那个界面上。

3. 存在的问题

验证码作为区分人与机器的一道重要屏障，与之相关的对抗技术一直在 Web 安全研究上有着重要意义。图片验证码生成算法以及程序实现流程上都有可能带来问题，容易被攻击者突破。

图片验证码的生成可能存在如下问题：

1）图片验证码的字符空间小

如果选取的字符空间较小，则让验证码识别变得相对简单。字母数字组合的字符集比单纯为数字的字符集效果要好。

2）图片中的字符规则

字符进行变形、扭曲不利于程序的识别，而对人眼识别是无障碍的，但此方法对生成程序来说有一定的难度。

3）图片中缺少干扰图案

干扰图案能有效增加验证码的识别难度，并且对生成程序来说代价小。

在验证码的程序实现流程方面可能存在如下问题：

1）验证码固定

一般地，用户在开始访问初始页面时，浏览器会向服务器发起页面请求，服务器此时创建会话，同时返回的页面里会嵌入验证码图片地址，浏览器在加载响应页面后，会自动加载验证码图片地址。服务器在接收到验证码图片地址的请求时，会对当前会话生成一个对应的验证码，并且返回验证码图片。用户此时就可根据验证码图片信息填写表单数据来进行后面的操作，这时，服务器如果检验出提交的验证码是错误的，就会返回页面提示错误然后跳转到初始页面，因初始页面内嵌了验证码图片地址，浏览器再次自动刷新验证码页面。

如果验证码输入错误，会在上面的流程里自动刷新验证码，正常情况下，这个过程看起来是没有问题的。但是，由于 HTTP 请求响应式的工作原理，使得攻击者有可能控制验证码生成页面请求不被触发，使得在

服务器上同一会话内，服务器端保存的此会话下的验证码一直有效。

基于此，服务器端的正确处理应该是在验证码检验失败时，就需要设置此验证码失效，同时对生成的每一个验证码也需要设置有效期。

2) 验证码的字符串值出现在返回的响应中

这是程序编码考虑不当导致的，比如忘记注释掉调试信息导致。验证码可能出现在响应包中的 Cookie、URL、页面注释，甚至验证码在展示的时候直接就是文本方式，这样就完全失去了使用验证码的价值了。

3) 验证码长度可指定

在页面上存在参数可指定验证码的位数，这可简化识别工作。此问题的出现也可能是调试的需要，发布时忘记注释掉相关代码而导致。

4. 对抗现状

自验证码技术问世以来，其成为自动化程序运行的第一大敌人。在利益的驱动下，突破封锁技术自然也会产生。一般来说，验证码的对抗技术有如下几个方面：

1) 避免触发验证码

验证码的引入会带来用户友好度的下降，增加验证码输入以及人眼对验证码识别的可能错误等都会带来不好的使用体验。这对追求用户体验的网站来说，在没遇到可疑行为时，其采取的策略是默认不开启验证码功能，当触发可疑行为识别规则时，才会出现验证码。基于此原理，对攻击方来说，就是尽量避免触发可能出现验证码的规则。一般可疑行为识别是通过检查频繁尝试并且出错这个行为来进行的，基于此的对抗措施则是使用不同的 IP 来进行尝试，或者等待足够的时间再次尝试，这样让目标网站程序认为这些尝试都是正常的访问请求，从而在自动化程序端连续作业时不会出现验证码而达到绕过的目的。

2) 验证码固定

从前面的验证码固定问题描述可看出，攻击者可以在同一个会话下，在获得第一个验证码后，后面不再主动触发验证码生成页面，并且一直使用第一个验证码就可循环进行后面的表单操作，从而绕过了验证码的屏障作用。

3) 验证码机器自动识别

计算机自动识别验证码，主要原理是通过一定的算法预先建立验证码范围内的字体特征库，再将要识别的验证码通过同样的算法生成特征，与之前保存的特征库进行比较，进而得到图片验证码的值。

一般地，其识别过程有如图 2 所示的处理流程。

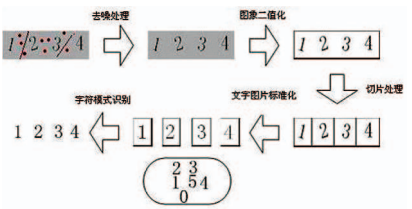


图 2 验证码计算机识别过程

1) 去噪处理

主要是去掉图像里的所有干扰信息，比如背景的点、线等。

2) 图像二值化

图像二值化就是将图像上的像素点的灰度值设置为 0 或 255，也就是将整个图像呈现出黑白效果。图像二值化主要是为了将图像与空白区分开来，方便后续步骤的进行。

3) 切片处理

图片的切片处理是将每个字符所在区域

分离成一个个的图片，确保一个图片内只有一个文字。

4) 文字图片标准化

对切片后的每个字符图片，还需要进行变形修复处理，比如旋转、缩放等操作尽量将每一个字符图片都调整到标准格式，减小随机度。

5) 字符模板识别

在最后的字符识别阶段，常用的是通过模板对比的方法。模板的生成就是通过上述处理过程后，对给定图片处理后把生成的特征保存下来，并且人工识别输入其对应的字符。在用作识别阶段时，就与保存的每个字符特征模板进行比较，从中找出最相近的一个字符来。

4) 人工分布式识别

机器自动识别图片验证码，对简单的情况能有较高的准确率，但对干扰多、变形复杂的图片验证码，其准确率会很差。由于图片验证码重要度增加，复杂的图片验证码被大量使用，导致近年来出现了利用众包力量实现的人工验证码识别平台。

其工作原理图下所示：

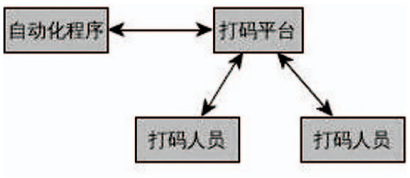


图 3 人工分布式识别

自动化程序将要识别的验证码发送到打码平台，打码平台再将验证码发放给从事验证码识别工作的人员（俗称打码工作），在人工识别后再将值依次返回到自动化程序。由于是人直接参与了验证码的识别，此方法就让验证码完全失去了屏障功能。

5. 验证码未来可能的主要形式

随着验证码攻防技术的对抗升级，验证码技术也出现了一些新的发展动向。

从图片验证码自身的发展来看，主要是围绕增强人脑交互性来展开的。具体来说有如下几个方向：

1）增强干扰和字符变形

从验证码的机器识别可以看出来，增强干扰和字符变形能极大地提高识别难度。像下面的生成的图片验证码，对人眼识别来说都存在着一些难度。

2）拓展字符空间

常见的图片验证码都是数字或者字母，近年来出现了中文字符作为验证码，这样字符空间就增大了很多。

如果字符空间足够大，试图通过制作字符模板库方式来实现识别的难度就变得很大了。

3）增强与用户的互动性

通过增加与用户的互动也可增加难度，

但因为这类互动都是通过问答题方式来实现的，其题库数量是否足够以及是否具有足够的随机性则成为了关键所在。

如下图所示的问答型验证码。

从验证码的信息传递途径和方式来看，有如下几个方式：

1）增加信息传递途径

在原来只通过网页来传递信息的基础上增加了其他传递方式，比如手机短信等。对于使用手机短信验证码，需要确保其关联的手机号不为攻击者所控制，否则也将失去保护意义。因为通过在手机里安装特定 App 软件就可以获得短信内容，这对于手机号码攻击者可控的情况下，验证码可轻松绕过。但如果其攻击场景是账号登录类的，由于手机号码不可控，则使得暴力猜测变得不可能实施。扩展验证码的传递途径需要结合应用场景来使用，确保此途径不容易被攻击者获得。

2）利用动态令牌进行一次一密

通过动态令牌实现的一次一密方式，就必须通过人工交互才能实现，但此方式也增加了使用成本，只适用于密级要求很高的场合。

3）语音方式

语音方式在当前网络也有出现，主要只是作为方便盲人使用的一个备用方案，但由于智能手机终端的广泛普及，以后也有可能成为一个主要的验证码发展场景。

随着技术的不断发展，以后还会出现其他方式的验证码，但可以肯定的是，基于人脑的思维特性来区别于程序的预设特性，至于以后会如何发展，还是让我们拭目以待吧。

6. 小结

本文从验证码概念、工作原理入手，介绍了当前验证码的攻防情况，并且简要介绍了验证码的发展趋势。可以肯定的是，不管使用何种方式的验证码，都会影响用户体验，机器难以识别的验证码最后也会影响人眼的识别，验证码的设计者们需要在用户体验和安全性方面做一个折中选择。寄希望于通过验证码来作为对抗机器自动化行为的唯一途径是不可取的，在识别自动化程序发起的远程 Web 扫描、登录猜测等方面，推荐架设 WAF 类的专业防护产品，可以有效阻挡即将发生的安全事件。

参考文献

http://drops.wooyun.org/tips/141 常见验证码的弱点与验证码识别

http://zh.wikipedia.org/zh/%E9%AA%8C%E8%AF%81%E-7%A0%81 验证码

Web应用安全开发参考

北京分公司 封炎 赵波

关键词：安全开发 Web 应用系统 漏洞修复

摘要：Web 应用安全漏洞时刻威胁着我们的信息系统和网站。如何帮助开发人员编写安全的程序，让他们能够更从容地应对安全威胁呢？Web 应用安全开发参考，也许能有所帮助。

一、引言

提到如何提高 Web 应用开发安全水平，首先会想到“全生命周期安全开发”流程，该流程从需求阶段的安全需求分析，经过系统设计的威胁建模、代码开发的规范指导和上线前的安全评估，到运维阶段的安全巡检和监控，将安全作为软件开发的一个属性，实现最终的用户安全。谈到流程，就不得不考虑流程的落地。既然是应用开发，核心角色便是开发人员。开发人员在实现应用功能方面具备较强实现能力，但在安全开发方面普遍积累较少，并且在开发进度和需求变更等方面的重压下，不得不以牺牲安全为代价优先实现应用功能。

下面描述两个与开发人员接触的场景。

某安全服务团队对系统A进行安全渗透测试，测试完成后提交了漏洞描述和修复建议，开发部门遵从修复建议进行了程序修正，在进行回归测试时，测试人员稍微变换了一下攻击特征即可绕过安全防护程序。

一个安全测试场景

这个场景并不能说明安全测试人员的技术有多好，更不能证明开发人员的开发技术水平不足，普遍情况是部分开发人员仅知晓基本的修复思路，缺少足够健壮的安全代码编写能力。作为专业的安全

服务团队需要提供更为详实的安全开发建议，而开发团队需要注重安全开发经验的积累。

> *某安全服务人员与开发部门针对系统B发现的安全漏洞进行讨论，由于数字证书校验类的易用性较差，经常出现异常导致程序无法正常运行，开发人员简化了数字证书校验类，导致了安全漏洞的出现。漏洞修复的阻碍依然是开发人员无法有效利用API的安全属性及相关配置。*
>
> ***一个测试汇报场景***

针对上述两个场景，我们需要思索：到底开发人员需要什么样帮助？一眼就能看完的安全建议，背后是开发人员大量的知识搜索与消化和转化成代码的时间消耗。何解？这就是本文编制目的：提出一个实用的“Web 应用程序安全开发参考”思路。

二、安全开发参考缘由

站在安全服务团队角度，大多数情况下，安全评估及测试工作提供的漏洞解决方案并不能满足用户的需求，另外在设计安全功能、指导 Web 应用系统安全开发的过程中，安全服务团队提供的《安全编码规范》更多地是针对程序语法，对于业务逻辑及常见的功能点安全性的帮助有限，而《Web 应用程序安全开发参考》则可为开发人员提供更多更实用的帮助。

站在开发人员角度其实需要更多的指导，尤其是在安全方面。国内的软件开发行业经历了几十年的发展，但无论是开发人员、项目还是产品，总摆脱不了“速成”的标签。经验丰富的开发人员更多的做了项目经理、需求设计专家和框架开发人员，而真正还在写功能代码的往往不具备丰富的安全开发经验。

2.1 宽泛的漏洞修复建议

一些缺乏安全经验的开发人员见到宽泛的漏洞修复建议总是被搞得一头雾水，比如 SQL 注入漏洞，安全修复建议一般都是“使用参数化查询、存储过程、对特殊字符进行过滤”，但哪种代码修复方式更有效？需要对哪些特殊字符做过滤？这些才是开发人员更为关心的，宽泛的漏洞修复建议所得到的结果往往是开发人员没能有效地对漏洞进行正确修复。

例如某网站存在存储型跨站脚本攻击漏洞，渗透测试人员进行了测试，找到威胁点并提供了特殊字符过滤的漏洞修复方案。开发人员仿佛明白了漏洞原理，他在前台 JavaScript 脚本增加了过滤程序，对部分特殊字符及标签进行过滤处理，结果漏洞并没有有效的修复。

其实开发人员更需要一个有效的、详实的、合理的漏洞修复代码段或过滤函数，也许并不只是漏洞修复建议这么简单。

2.2 安全编码参考需积累

好的代码应该像宝石一样被妥善保存，有经验的开发人员都是宝石的收集者，一旦有需要，他们能够直接复用之前保存的代码段甚至整个功能模块。安全功能类参考代码也是如此，一个成熟的过滤程序或是安全处理机制都能够极大地提高产品的安全性，有效地避免安全威胁，并在类似功能开发及功能扩展时帮助缺乏安全经验的开发人员提高安全防护水平。

一些企业已经着手制定自己的安全开发框架，随着安全开发框架的不断更新和完善，产品也会有更为坚固的防护壁垒。

三、安全开发参考介绍

Web 应用程序安全开发参考是以攻防视角提供代码级别安全开

发参考，使开发人员了解什么样的代码才能够进行有效的漏洞防护。

3.1 与其他安全工作关系

Web 应用程序安全开发参考可以理解为一个工具，它对于 Web 应用系统开发、Web 应用程序加固、安全测试、源代码审计及漏洞修复都有指导和参考的作用。而通过不断的积累和更新，也能使 Web 应用程序安全开发参考更加完善，能够应对更多的安全漏洞，覆盖更多的功能模块。

下图展示了 Web 应用程序安全开发参考与其他安全工作的关系。

3.2 安全开发参考结构

Web 应用程序安全开发参考分为两部分，Web 应用程序安全开发实例和常见 Web 应用漏洞加固两部分。Web 应用程序安全开发实例部分为开发人员提供了与安全相关的源代码参考，而常见 Web 应用漏洞加固部分则提供了一些常见安全漏洞的加固方案，这些漏洞都是与开发框架相关的。

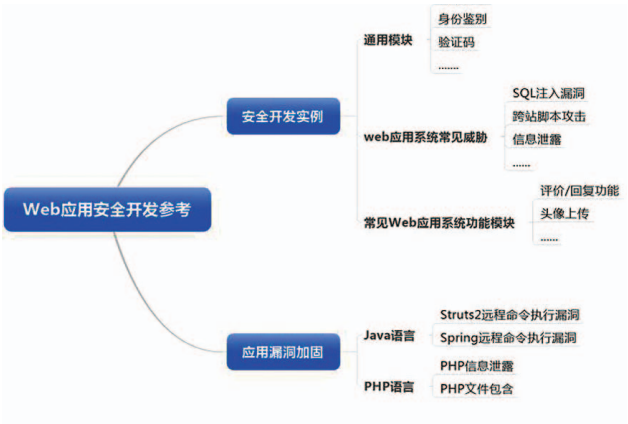
Web 应用程序开发实例分为通用模块、Web 应用系统常见威胁和常见 Web 应用系统功能模块三部分：

• 通用模块部分对登录、权限控制、验证码、注销等常见功能进行安全分析，描述可能存在的安全威胁，并提供代码参考。

•Web 应用系统常见威胁部分分析了常见 Web 应用安全漏洞可能涉及的功能点，并通过代码样例详尽地描述漏洞防护及修复方案。

• 常见 Web 应用系统功能模块部分收集了 Web 应用系统常见功能的最佳编码实践，能够帮助开发人员优化代码结构，提高功能安全性。

常见 Web 应用漏洞加固部分提供了部分开发框架安全配置的代码描述及常见框架类安全漏洞的正确配置及修复方案。



四、参考代码实例展示

安全开发参考代码的描述包括应用场景、代码说明和代码参考三部分：应用场景描述此部分代码建议使用在哪些功能模块或控制机制上，便于使用者快速定位；代码说明将对核心代码进行说明，帮助使用者理解代码精髓；代码参考为具体的代码实现。本章节将展示部分可参考代码。

4.1 通用模块之会话管理

4.1.1 会话固定避免

在登录验证成功后，通过重置 SessionID，失效处理匿名登录 SessionID，这样可以避免使用预置的 SessionID 进行会话固定攻击。

应用场景：

网站登录模块

代码说明：

使用 request.getSession().invalidate() 方法使匿名 session 失效，重新生成 SessionID 避免会话固定攻击。

代码参考：

```
protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                String username=request.getParameter("username");
                String password=request.getParameter("password");
                if("admin".equals(username) &&
"pass".equals(password)){
            //使之前的匿名session失效
                    request.getSession().invalidate();
                    request.getSession().setAttribute("login", true);
                    response.sendRedirect("hello.jsp");
                }else{
                    response.sendRedirect("login.jsp");
                }
            }
```

4.1.2 会话失效时间

会话失效时间过长会增加服务器资源消耗和会话劫持的风险，设置合理的会话失效时间可提高服务器性能和安全水平。

应用场景：

会话使用场景

代码说明：

ession.setMaxInactiveInterval() 方法用户设置会话失效时间，单位为“秒”；tomcat 的 web.xml 配置方式时间单位为：“分钟”。

代码参考：

```
// 设置 Session 对象的最长不活动间隔
session.setMaxInactiveInterval(30);
```

```
//tomcat 通过 web.xml 配置文件实现
<session-config>
<session-timeout>30</session-timeout>
</session-config>
```

4.1.3 越权访问与操作

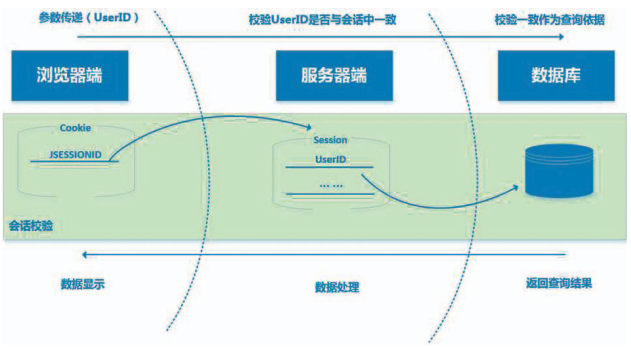
越权访问与操作漏洞可通过会话机制进行防护，存在权限差异的网站功能在返回数据到客户端用户时，应基于会话进行数据返回。客户端登录成功后，应将客户端身份标识 UserID（假设参数）保存在服务器端的会话中，在后续的交互中，依据客户端 cookie 中的会话 ID（即 JSESSIONID）在服

务器会话中查找身份标识 UserID，进行数据库查询后返回 UserID 对应数据。

采用会话机制的关键有两点：

1. 创建会话时，要将客户端用户的关键信息保存在服务器端会话中；

2. 返回数据时，不以依赖客户端传递的变量为身份标识。



适用场景：

存在权限差异的功能都有越权查看和操作的风险，列举典型的攻击场景如下：

1. 未登录状态下越权访问其他 URL 路径。

2. 网上银行用户 A 可越权查看用户 B 的账单或信用卡消费记录。

3. 网上银行用户 A 可使用用户 B 的借记卡来还用户 A 的信用卡。

4. 电子商城用户 A 可越权查看用户 B 的订单详情和收货地址信息。

5. 审批系统中低审批权限审批员 A 可越权审批总经理权限的申请单。

代码说明：

代码采用 Java 过滤器技术，对 /pages 下所有 URL 进行登录状态检查，通过 session.getAttribute() 方法从 session 中获取登录成功时存入 session 中的身份标识，判断客户端传递过来的身份标识是否与 session 中保存的一致，不一致则跳转到登录页面。关键代码如下：

```
// 从 session 里取的用户名信息
String username = (String) session.getAttribute("userID");
//getAttribute 中变量根据实际变量传入。
// 判断如果没有取到用户信息，就跳转到登录页面
if ((username == null) || "".equals(username)) {
// 跳转到登录页面
res.sendRedirect("http://" + req.getHeader("Host") +"/login_oa.jsp");}
else {
// 已经登录，继续此次请求
chain.doFilter(req, res); }}
```

代码参考

```
<filter>
<filter-name>SessionFilter</filter-name>
<filter-class>com.nsfocus.frame.filter.SessionFilter</filter-class>
</filter>
<filter-mapping>
<filter-name>SessionFilter</filter-name>
<url-pattern>/pages/*</url-pattern>
</filter-mapping>
<filter>
--------------------------------------------------------------------------------
```

```
package com.nsfocus.frame.filter;
import java.io.IOException;
import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
public class SessionFilter implements Filter {public void
 init(FilterConfig filterConfig) throws ServletException
{}
    public void doFilter(ServletRequest request, ServletResponse
 response,FilterChain chain) throws IOException, ServletException {
    HttpServletRequest req = (HttpServletRequest) request;
    HttpServletResponse res = (HttpServletResponse) response;
    HttpSession session = req.getSession(true);
       // 从 session 里取的用户名信息
      String username = (String) session.getAttribute("userID");
      //getAttribute 中变量根据实际变量传入。
       // 判断如果没有取到用户信息 , 就跳转到登陆页面
      if ((username == null) || "".equals(username)) {
      // 跳转到登陆页面
      res.sendRedirect("http:///" + req.getHeader("Host")
 +"/login_oa.jsp");}
      else {
      // 已经登陆 , 继续此次请求
      chain.doFilter(req, res);       }   }
    public void destroy() {    }
    }
```

4.2 常见威胁之 SQL 注入

4.2.1 SQL 注入攻击

SQL 注入攻击就是通过把 SQL 命令插入到 Web 表单递交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的 SQL 命令。如果 SQL 注入成功，攻击者可以获取系统数据库的信息，可以修改删除数据库，还可能获取执行命令的权限，进而完全控制服务器。

应用场景：

任何涉及数据库连接的功能点，如用户登录、查询、数据修改、删除等功能。

代码参考（代码说明分节体现）：

(1) 通过参数化查询方式进行 SQL 注入攻击防护。

```
using (SqlConnection conn = new SqlConnection(connectionString))
{
       conn.Open();
       SqlCommand comm = new SqlCommand();
       comm.Connection = conn;
       // 为每一条数据添加一个参数
      comm.CommandText = "select COUNT(*) from Users where
Password = @Password and UserName = @UserName";
       comm.Parameters.AddRange(
       new SqlParameter[]{
                new SqlParameter("@Password", SqlDbType.VarChar) {
Value = password},
                      new SqlParameter("@UserName", SqlDbType.
VarChar) { Value = userName},});
       comm.ExecuteNonQuery();
      }

      // 以 “?” 等位符进行参数化查询
     PreparedStatement pstmt = con.prepareStatement("select * from t
able where name = ?");
pstmt.setString(1, para);
ResultSet rs = pstmt.executeQuery();
```

(2) 使用 MyBatis 技术，通过 Mapper.xml 文件定义 SQL 语句进行 SQL 注入攻击防护。

```
<mapper namespace="TestUser">// 命名空间
<select id="getById" parameterType="java.lang.String"
        resultMap="TestFlowResult">
        select
        <include refid="TestFlowColumns" />
        <![CDATA[
            from TEST_TABLE
            where
                INSPECT_ID = #{id}
        ]]>
</select>
</mapper>
// 在编写 mybatis 的映射语句时，尽量采用 “#{xxx}” 这样的格式。若不得不使用 “${xxx}” 这样的参数，要手工地做好过滤工作，来防止 sql 注入攻击。
```

(3) 在 SQL 语句提交数据库查询前使用特殊字符过滤程序防护 SQL 注入攻击。

```
public static bool SqlCheck(string OldString)
{
    bool Checkvalue = false;
    string NewString = OldString.ToLower();
    string Replace =
"'|and|exec|insert|select|delete|update|count|*|;|%|union|chr|mid|master|truncate|char|dec
lare|asc|cast|set|fetch|varchar|sysobjects|drop|alert|script|<|>";
    string[] arrReplace = Replace.Split('|');
    for (int i = 0; i < arrReplace.Length; i++)
    {
        if (NewString.IndexOf(arrReplace[i].ToString()) >= 0)
        {
            bolvalue = true;
            break;
        }
    }
    return Checkvalue;
}
```

4.3 漏洞加固之 PHP 基础安全

4.3.1 allow_url_fopen 设置

allow_url_fopen 是 PHP 的一个特性，它可以打开远程 http 或 ftp 的文件。这个特性在发生文件包含漏洞的时候容易带来一些安全问题，所以如果不是必要关闭这个选项。

应用场景：

PHP 开发网站

代码说明：

当应用程序需要远程读取某些资源时，如果设置 allow_url_fopen=OFF，则应用程序就没有办法实现远程读取。在这种情况下，可以使用 curl_init 模块实现远程读取。

代码参考：

```
<?php
// create a new curl resource
$ch = curl_init();
// set URL and other appropriate options
curl_setopt($ch, CURLOPT_URL, “http://www.google.nl/” );
// grab URL and pass it to the browser
curl_exec($ch);
// close curl resource, and free up system resources
curl_close($ch);
?>
```

五、总结

Web 应用程序安全开发参考能够为开发及安全测试人员提供更直接的帮助，而安全开发参考也需要开发人员不断动态完善。其实无论是安全公司还是企业都可以借鉴这个思路，将优秀的安全代码积累起来，合理地应用到安全开发和其他安全工作上，提升企业的安全防护水平，提升最终用户的满意度。

glibc TLS变量初始化问题分析

核心技术部　张云海

关键词：glibc　TLS 初始化

摘要：本文介绍了glibc TLS 变量初始化问题的分析过程。

1. 引言

同事在调试一个复杂程序时，发现了glibc 中一个 TLS 变量的问题：当 TLS 变量的类型是 char 或者 unsigned char 时，在主程序中与动态链接库中该变量的值不一样，甚至取变量的地址得到的结果都不一样。

本文记录了对此问题进行分析，逐步确定其本质问题的过程。

2. 分析过程

2.1 缩小问题范围

原始的示例程序比较复杂，涉及动态加载动态链接库与多线程操作，可能导致问题的位置比较多，因此，想通过简化示例程序来排除一些可能，缩小问题的范围。

用 gdb 调试示例程序，在尝试打印该 TLS 变量时会得到以下提示：

```
(gdb) p magic_c
The inferior has not yet allocated storage for thread-local variables in
the executable `/home/test/test/tls_demo_2'
for Thread 0xf7df0700 (LWP 56071)
```

由此可以推测该 TLS 变量没有正确的初始化，而这个初始化应该是与多线程的操作无关的。去掉示例程序中多线程相关的部分后再进行测试，确实还可以稳定地重现问题，基本证实了这一猜测。

更进一步，虽然该 TLS 变量在动态链接库中有引用，但是其初始化是在加载动态链接库之前完成，因此也应该与加载动态链接库无关。将示例程序中动态链接库相关部分去掉后再进行测试的，问题表现为另外一个形式——TLS 变量没有正确赋值。

最终，我们得到一个最小的问题代码，如下：

```
~/test$ cat demo1.c
#include <stdio.h>
;__thread char magic_c = 1;
;int main(int argc, char** argv)
{
        printf("magic_c @ %p = %d\n", &magic_c, magic_
c);
}
~/test$ ./demo1
magic_c @ 0xf757393f = 0
```

作为对照的正常代码如下：

```
~/test$ cat demo2.c
#include <stdio.h>
__thread short magic_c = 1;
int main(int argc, char** argv)
{
        printf("magic_c @ %p = %d\n", &magic_c, magic_
c);
}
~/test$ ./demo2
magic_c @ 0xf757493e = 1
```

2.2 定位 TLS 变量的初始化

由于问题发生在 TLS 变量的初始化，因此想对其初始化的过程进行调试，这就需要确定初始化相关代码的位置。

首先，对正常代码进行调试。

在 _start 处设置断点，发现 TLS 变量的初始化已完成。

```
(gdb) b _start
Breakpoint 1 at 0x8048380
(gdb) r
Starting program: /home/zhangyunhai/test/demo2
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/
libthread_db.so.1".

Breakpoint 1, 0x08048380 in _start ()
(gdb) p magic_c
$1 = 1
```

此时，一个可行的方案是在 dl_main 处设置断点，然后跟踪程序的执行，从而找到 TLS 变量初始化的位置。另一个思路是尝试固定 TLS 变量的地址，进而通过设置数据断点来确定 TLS 变量初始化的位置。

通过禁用系统的 ASLR，可以固定 magic_c 的地址：

```
~/test$ echo 0 > /proc/sys/kernel/randomize_va_space
~/test$ ./demo2
magic_c @ 0xf7df56fe = 1
~/test$ ./demo2
magic_c @ 0xf7df56fe = 1
~/test$ ./demo2
magic_c @ 0xf7df56fe = 1
~/test$ ./demo2
magic_c @ 0xf7df56fe = 1
```

另外，gdb 也提供了禁用 ASLR 的选项 disable-randomization，并且其缺省值是启用。

在 magic_c 的地址处设置数据断点，中断后查看调用栈，可以

确定 TLS 变量是在 __GI__dl_allocate_tls_init 中初始化的。

```
(gdb) watch *(short*)0xf7df56fe
Hardware watchpoint 1: *(short*)0xf7df56fe
(gdb) r
Starting program: /home/zhangyunhai/test/demo2
Hardware watchpoint 1: *(short*)0xf7df56fe
Old value = <unreadable>
New value = 1
mempcpy () at ../sysdeps/i386/i686/multiarch/../mempcpy.S:60
60    ../sysdeps/i386/i686/multiarch/../mempcpy.S: No such file or directory.
(gdb) bt
#0  mempcpy () at ../sysdeps/i386/i686/multiarch/../mempcpy.S:60
#1  0xf7fed939 in __GI__dl_allocate_tls_init (result=0xf7df5700) at dl-tls.c:437
#2  0xf7fdf2ed in dl_main (phdr=0x8048034, phnum=10, user_entry=0xffffd6dc, auxv=0xffffd7d0) at rtld.c:2316
#3  0xf7ff0987 in _dl_sysdep_start (start_argptr=0xffffd770, dl_main=0xf7fddfe0 <dl_main>)
    at ../elf/dl-sysdep.c:244
#4  0xf7fe0efb in _dl_start_final (arg=0xffffd770) at rtld.c:338
#5  _dl_start (arg=0xffffd770) at rtld.c:564
#6  0xf7fdd1d7 in _start () from /lib/ld-linux.so.2
```

2.3 问题成因

__GI__dl_allocate_tls_init 的实现如下：

```
374 void *
375 internal_function
376 _dl_allocate_tls_init (void *result)
377 {
378   if (result == NULL)
379     /* The memory allocation failed.  */
380     return NULL;
381
382   dtv_t *dtv = GET_DTV (result);
383   struct dtv_slotinfo_list *listp;
384   size_t total = 0;
385   size_t maxgen = 0;
386
387   /* We have to prepare the dtv for all currently loaded modules using
388      TLS.  For those which are dynamically loaded we add the values
389      indicating deferred allocation.  */
390   listp = GL(dl_tls_dtv_slotinfo_list);
391   while (1)
392     {
393       size_t cnt;
394
395       for (cnt = total == 0 ? 1 : 0; cnt < listp->len; ++cnt)
396     {
397       struct link_map *map;
398       void *dest;
399
400       /* Check for the total number of used slots.  */
401       if (total + cnt > GL(dl_tls_max_dtv_idx))
402         break;
403
404       map = listp->slotinfo[cnt].map;
405       if (map == NULL)
406         /* Unused entry.  */
407         continue;
```

```
408
409     /* Keep track of the maximum generation number.
    This might
410        not be the generation counter.  */
411     maxgen = MAX (maxgen, listp->slotinfo[cnt].gen);
412
413     if (map->l_tls_offset == NO_TLS_OFFSET
414         || map->l_tls_offset == FORCED_DYNAMIC_
TLS_OFFSET)
415       {
416         /* For dynamically loaded modules we simply
store
417        the value indicating deferred allocation.  */
418         dtv[map->l_tls_modid].pointer.val = TLS_DTV_
UNALLOCATED;
419         dtv[map->l_tls_modid].pointer.is_static = false;
420         continue;
421       }
422
423     assert (map->l_tls_modid == cnt);
424       assert (map->l_tls_blocksize >= map->l_tls_
initimage_size);
425 #if TLS_TCB_AT_TP
426       assert ((size_t) map->l_tls_offset >= map->l_tls_
blocksize);
427     dest = (char *) result - map->l_tls_offset;
428 #elif TLS_DTV_AT_TP
429     dest = (char *) result + map->l_tls_offset;
430 #else
431 # error "Either TLS_TCB_AT_TP or TLS_DTV_AT_TP
must be defined"
432 #endif
433
```

```
434     /* Copy the initialization image and clear the BSS
part.  */
435     dtv[map->l_tls_modid].pointer.val = dest;
436     dtv[map->l_tls_modid].pointer.is_static = true;
437     memset (__mempcpy (dest, map->l_tls_initimage,
438             map->l_tls_initimage_size), '\0',
439         map->l_tls_blocksize - map->l_tls_initimage_
size);
440   }
441
442   total += cnt;
443   if (total >= GL(dl_tls_max_dtv_idx))
444  break;
445
446   listp = listp->next;
447   assert (listp != NULL);
448  }
449
450 /* The DTV version is up-to-date now.  */
451 dtv[0].counter = maxgen;
452
453 return result;
454 }
455 rtld_hidden_def (_dl_allocate_tls_init)
```

比较问题代码与正常代码在 __GI__dl_allocate_tls_init 中的执行路径，发现问题代码进入了 413 行处的 if 分支，因而没有分配 TLS 变量的存储空间。之所以会进入这一分支，是因为条件 map->l_tls_offset == FORCED_DYNAMIC_TLS_OFFSET 成立。

map->l_tls_offset 是 .tdata 段的大小，由于问题代码中只有一个 TLS 变量，且其类型是 char，因此 map->l_tls_offset 是 1。

FORCED_DYNAMIC_TLS_OFFSET 是在 include/link.h 中定义的：

```
288 #ifndef NO_TLS_OFFSET
289 # define NO_TLS_OFFSET  0
290 #endif
291 #ifndef FORCED_DYNAMIC_TLS_OFFSET
292 # if NO_TLS_OFFSET == 0
293 #  define FORCED_DYNAMIC_TLS_OFFSET 1
294 # elif NO_TLS_OFFSET == -1
295 #  define FORCED_DYNAMIC_TLS_OFFSET -2
296 # else
297 #  error "FORCED_DYNAMIC_TLS_OFFSET is not
      defined"
298 # endif
299 #endif
```

显然，这里将 map->l_tls_offset 复用为一个标志字段，又没有正确地选择标志的值，使得偏移的值被当作标志处理了，从而导致了未正确初始化的问题。

确定问题之后，在 glibc 的 Bugzilla 中检索了一下，发现这是一个已修复的 Bug（https://sourceware.org/bugzilla/show_bug.cgi?id=14898）：


First Last Prev Next This bug is not in your last search results.

Bug 14898 - Unable to create small static TLS block in shared library

Status: RESOLVED FIXED

Product: glibc
Component: libc
Version: 2.15

Importance: P2 normal
Target Milestone: 2.17
Assigned To: Not yet assigned to anyone

URL:
Keywords:

Depends on:
Blocks:

Show dependency tree / graph

Reported: 2012-11-30 17:12 UTC by Bharath Ramesh
Modified: 2013-01-14 14:12 UTC (History)
CC List: 4 users (show)

See Also:
Host:
Target:
Build:
Last reconfirmed:

修复的版本是 2.17，修复的方法将 FORCED_DYNAMIC_TLS_OFFSET 的定义从 1 改为 -1。看起来这应该是一个笔误导致的 Bug。

```
287 #ifndef NO_TLS_OFFSET
288 # define NO_TLS_OFFSET  0
289 #endif
290 #ifndef FORCED_DYNAMIC_TLS_OFFSET
291 # if NO_TLS_OFFSET == 0
292 #  define FORCED_DYNAMIC_TLS_OFFSET -1
293 # elif NO_TLS_OFFSET == -1
294 #  define FORCED_DYNAMIC_TLS_OFFSET -2
295 # else
296 #  error "FORCED_DYNAMIC_TLS_OFFSET is not defined"
297 # endif
298 #endif
```

2.4 回到原始问题

最后，我们回到原始问题看一下，为什么在主程序中 TLS 变量的地址会与动态链接库中的不一致。

TLS 变量有 4 种访问模型：

•General Dynamic (GD) - dynamic TLS

可以访问所有的 TLS 变量，支持 TLS 的推迟分配，通过调用 __tls_get_addr() 来获取 TLS 变量的地址。

•Local Dynamic (LD) - dynamic TLS of local symbols

对 GD 模型的优化，用于访问本地的 TLS 变量。只调用一次 __tls_get_addr() 获取 TLSBlock 的基址，在此基础上加上偏移计算出 TLS 变量的地址。

•Initial Executable (IE) - static TLS with assigned offsets

只能访问 initial static TLS template 中的 TLS 变量，用 GOT 中存储的偏移来计算 TLS 变量的地址。

•Local Executable (LE)- static TLS

只能访问主程序本身定义的 TLS 变量，用 gs:0 加上链接时计算好的偏移来获取 TLS 变量的地址。

原始的示例程序中主程序是使用 Local Executable (LE) 模型来访问 TLS 变量的。

```
0x08048a22 <+213>:    mov    %gs:0x0,%eax
0x08048a28 <+219>:    lea    -0x1(%eax),%edi
```

此时得到的地址是 TLS 变量预期加载的地址，实际上因为初始化的 Bug，这个地址并没有被分配。

动态链接库中是使用 General Dynamic (GD) 模型来访问 TLS 变量的。

```
0xf7fd72bd <+45>:     lea    -0x8(,%ebx,1),%eax
0xf7fd72c4 <+52>:     call   0xf7fd7280 <___tls_get_
addr@plt>
0xf7fd72c9 <+57>:     mov    %eax,%esi
```

__tls_get_addr 的实现如下：

```
748 void *
```

```
749 __tls_get_addr (GET_ADDR_ARGS)
750 {
751   dtv_t *dtv = THREAD_DTV ();
752   struct link_map *the_map = NULL;
753   void *p;
754
755   if (__builtin_expect (dtv[0].counter != GL(dl_tls_
generation), 0))
756   {
757       the_map = _dl_update_slotinfo (GET_ADDR_
MODULE);
758     dtv = THREAD_DTV ();
759   }
760
761   p = dtv[GET_ADDR_MODULE].pointer.val;
762
763   if (__builtin_expect (p == TLS_DTV_UNALLOCATED, 0))
764     p = tls_get_addr_tail (dtv, the_map, GET_ADDR
    MODULE);
765
766   return (char *) p + GET_ADDR_OFFSET;
767 }_
```

由于前面 __GI__dl_allocate_tls_init 中将 dtv 的 pointer.val 设置成了 TLS_DTV_UNALLOCATED，因此这里会调用 tls_get_addr_tail 重新分配并初始化 TLS 变量。此时得到的地址就是新分配的 TLS 变量的地址了。

3. 总结

glibc 2.17 之前的版本存在一个 Bug，在特定情况下没有正确分配并初始化 TLS 变量。

该 Bug 触发的条件是：.tdata 段的大小为 1，即主程序中有且仅有一个 char 或 unsigned char 类型的 TLS 变量，同时该变量的对齐为 1。

该 Bug 触发后，以 Local Executable (LE) 模型访问 TLS 变量时，不能得到正确的变量；以 General Dynamic (GD) 模型访问 TLS 变量时，会重新分配并初始化一个 TLS 变量。

参考文献

1.THREAD-LOCAL STORAGE DESCRIPTORS FOR IA32 AND AMD64/EM64T - ALEXANDRE OLIVA <AOLIVA@REDHAT.COM, OLIVA@LSD.IC.UNICAMP.BR>

HTTP://WWW.FSFLA.ORG/~LXOLIVA/WRITEUPS/TLS/RFC-TLSDESC-X86.TXT

2.ELF HANDLING FOR THREAD-LOCAL STORAGE - ULRICH DREPPER <DREPPER@GMAIL.COM>

HTTP://WWW.AKKADIA.ORG/DREPPER/TLS.PDF

3.RUNTIME ALLOCATION OF THREAD-LOCAL STORAGE

HTTP://DOCS.ORACLE.COM/CD/E19683-01/817-3677/CHAPTER8-10/INDEX.HTML

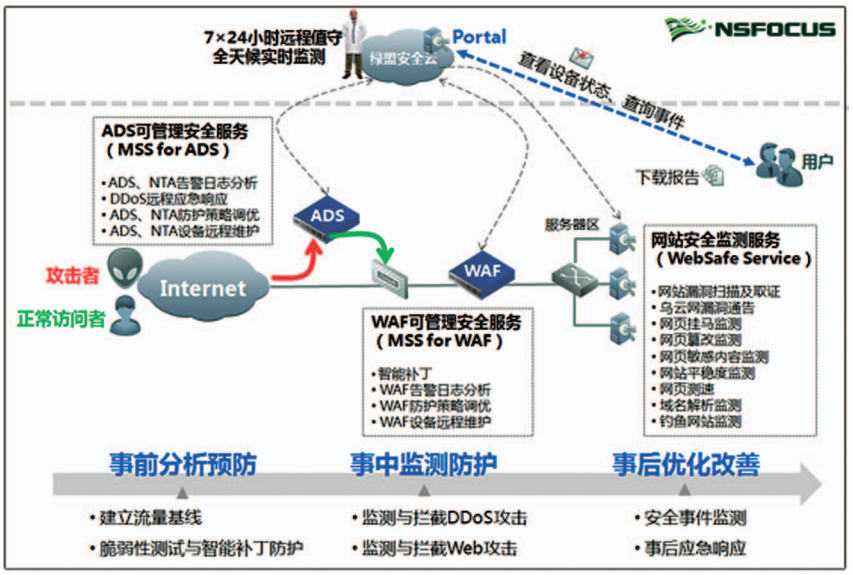
4.THREAD-LOCAL STORAGE ACCESS MODELS

HTTP://DOCS.ORACLE.COM/CD/E19683-01/817-3677/CHAPTER8-20/INDEX.HTML

绿盟科技通过ISO27001认证 树立云安全服务行业标杆

近日，绿盟科技宣布绿盟科技云安全运营服务业务获得 ISO27001 管理体系的认证，这标志着绿盟科技成为国内首家通过ISO27001认证的可管理安全服务(简称 MSS）和安全即服务（简称 SECaaS）提供商。ISO27001 是一项信息安全管理国际标准，它是目前国际上最权威、最严格、也是最被广泛接受和应用的信息安全标准。通过 ISO27001 的认证可以表明组织在信息安全管理已建立了一套科学有效的管理体系作为保障。

绿盟科技云安全运营中心负责人李钠表示：“此次绿盟科技云安全运营服务业务获得 ISO27001 管理体系的认证，说明绿盟科技有能力为客户提供有安全保障的可管理安全服务和安全即服务。由于国内的这两项服务还在起步阶段，客户对这两项服务引入的安全风险存在疑虑，因此服务商需要通过中立的第三方认证向客户展示自身的安全保障能力。作为国内最先开展可管理安全服务和安全即服务的领跑者，绿盟科技正通过努力和持续创新缩短中国与海外的差距。”



当前，绿盟科技提供的云安全运营服务以网站安全为核心，对网站面临的威胁和安全事件提供 7X24 小时全天候的监测与防护（参见上图）。云安全运营服务可以在安全事件发生前对网站提供 Web 漏洞智能补丁，预防针对 Web 漏洞的攻击；在安全事件发生中，云安全运营服务可对 DDoS 攻击和 Web 攻击提供 7X24 小时监测与防护，对攻击进行有效拦截；在安全事件发生后，云安全运营服务可及时监测到网站篡改、网站挂马等安全事件并进行响应和处置，快速消除安全事件带来的影响。

关于可管理安全服务 MSS

MSS 是一种通过云交付的安全服务，主要通过安全云实现对企业安全威胁和安全事件的远程监测和响应。Gartner 认为，MSS 是通过远程安全运维中心对 IT 安全功能的远程管理和监视服务，MSS 不包含安全值守、咨询、开发以及集成服务。

THE EXPERT
BEHIND GIANTS

巨人背后的专家

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

“美丽的互联时代给用户带来了前所未有的便利，与之俱来的信息安全风险亦令人们防不胜防。”

周文宝

绿盟科技成都分公司 高级安全顾问

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

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