Rethinking Security in the Era of Cloud Computing

Jay Aikat | University of North Carolina at Chapel Hill
Aditya Akella | University of Wisconsin at
Jeffrey S. Chase | Duke University
Ari Jue
Micha North Car
Thoma th
Vyas S iversity
Micha consin at Madison

在可预见的未来，云计算已经成为一个占主导地位的计算平台，破坏了我们构建和部署软件的方式。这种中断为集成新的计算机安全方法提供了难得的机会

Cloud computing has emerged as a dominant computing platform for the foreseeable future, disrupting the way we build and deploy software. This disruption offers a rare opportunity to integrate new computer security approaches.

关于计算云的急剧增长及其在现代计算中的变革作用，人们已经做出了很多的贡献。云计算的规模表明，在可预见的未来，它将成为推动力量;现在，互联网上的一个标志和增长部分已经被托管在主要的云计算运营商的数据中心里。列举了他们成功的原因包括节省的成本和灵活性建立和运行自己的数据中心的支出

Much has been made about the dramatic growth of compute clouds and their transformative role in modern computing. Cloud computing's scale suggests that it will be a driving force for the foreseeable future; a significant and growing fraction of the web is now hosted in major cloud operators' datacenters.[1] The often cited reasons for their success include the cost savings and flexibility they enable by freeing tenants from the expenditures of establishing and operating their own datacenters.

在云计算的热潮中，一个复杂的因素是租户安全。出于几个原因，转移到云上可能会加剧安全风险。
随着云计算的普及，云计算成为攻击者的攻击目标。
由于云将租户置于公共计算基础设施中，在基础设施中暴露的一个缺陷可能威胁到所有租户。
云计算在同一个计算平台上可能相互不信任的租户的计算，潜在地为跨租户提供了机会。
这些风险已经催生了大量积极和有价值的研究议程，使租户能够减轻这些风险。

A complicating factor in the rush to cloud computing is tenant security. For several reasons, security risks can be exacerbated by moving to the cloud. As cloud computing grows in popularity, clouds become juicier targets for attackers. Because the cloud puts tenants on a common computing infrastructure, one flaw exposed in that infrastructure can threaten all tenants. Clouds gather computations of potentially mutually distrusting tenants on the same compute platform, potentially introducing opportunities for cross-tenant attacks. These risks have given rise to numerous active and worthwhile research agendas to enable tenants to mitigate them.

We believe there's another research agenda that the move to cloud computing should ignite but that so far has received less attention from the research community. The consolidation of massive resources and myriad activities in the cloud places cloud operators in a unique position to introduce new services to help tenants better manage their security (or to manage it for them outright)—and indeed to solve some of security's "holy grail" problems. Starting in 2013, we were funded by the NSF to explore these opportunities; our goal in this article is to summarize our research agenda and the progress we've made so far. We refer to our effort as the Silver project, connoting a silver lining to the cloud.

我们相信还有另一项研究议程，即云计算的行动应该会点燃，但到目前为止，这一研究还没有得到来自研究界的支持。
在云计算中，大量资源和无数活动的整合使云计算运营商处于一个独特的位置，引入新的服务来帮助租户管理他们的安全(或者直接管理他们的安全)，并且确实解决了一些安全的“圣杯 ” 问题。
从2013年开始，我们得到了国家科学基金会的资助，以探索这些机会;
本文的目标是总结我们的研究议程和迄今为止取得的进展。
我们把我们的努力称为银项目，在云中加入一线希望。

我们相信还有另一项研究议程，即云计算的行动应该会点燃，但到目前为止，这一研究还没有得到来自研究界的支持。在云计算中，大量资源和无数活动的整合使云计算运营商处于一种独特的地位，引入新的服务来帮助他们。

Cloud Security Research Themes

这里我们提供了云安全研究领域的简要概述，大致描绘了两种研究动机的二分法。
第一个问题是迄今为止学术界最关注的问题，是租户对云计算的新威胁，以及租户可以采取什么措施来减轻这些威胁。
然后，我们将这与另一个研究领域的研究进行了对比，这一领域的研究是为了帮助租户实现安全部署，而这一领域的研究还未得到充分的研究。
(对于云的简史，请参阅侧栏。)

Here we provide a brief overview of the cloud security research landscape, roughly drawing a dichotomy between two research motivations. The first, which has been the primary concern of academics thus far, is what new threats to tenants arise due to their move to cloud computing, and what tenants can do to mitigate those threats. We then juxtapose this with another area of research that we find understudied—that of enhancing clouds to help tenants achieve secure deployments. (For a brief history of clouds, see the sidebar.)

Risks from Cloud Computing, and How Tenants Can Mitigate Them

Much work has focused on determining how the shift to cloud computing and public clouds might create new vulnerabilities for tenants. In one type of new risk, a cross-tenant attack arises when one cloud tenant seeks to violate the confidentiality, availability, or integrity of another. Many cross-tenant attacks stem from the fact that multitenancy implies that it's possible that one's

1540-7993/17/$33.00 © 2017 IEEE

A Brief History of Clouds

Modern cloud computing systems are the latest example of shared-infrastructure computing. The idea of "utility computing" on mainframes was a primary motivation for the Multics project, which inspired much early research on computer security. In such systems, users and businesses remotely accessed a central mainframe, amortizing the large cost over many users. With the rise of the World Wide Web in the late 1990s, public hosting centers grew in popularity and offered both bare-metal machines and managed webservers. The free Linux OS and Apache webserver, launched in 1991 and 1995, respectively, and servers with commodity Intel and AMD processors, made it economical to run large server clusters to handle the increasing scale of Internet workloads.

Two innovations really drove the cloud revolution. First, VMware introduced a hypervisor for x86 processors, showing that virtualization could be efficient and useful. This was followed by the open source Xen project from Cambridge University in 2003. Second, Amazon launched the Elastic Compute Cloud that rented virtual machines with a new billing model based on short-term usage. For a few cents, a customer could rent a fraction of a physical machine for an hour. The combination of virtualization, efficient management, and short time-scale billing enabled cloud computing's boom.

Today's public clouds provide access to large-scale computing infrastructure in the form of virtual machines, applications, or software services. The common thread among all these is the dynamic sharing of infrastructure by multiple tenants, which is managed by a third party. In this common structure, clouds today can be divided roughly into three kinds. Infrastructure-as-a-service systems allow customers to launch virtual machines and control the guest OS as well as applications. In
a-service systems, customers can launch application
in a provider-managed OS. Finally, a software-as-a-s
provides hosted application services for customers,
databases, file storage, and the like. A single compar
Google or Microsoft, might offer all three kinds of c
The division into these categories follows the level w
software platform at which there is a management l
the cloud provider to the customer.

云风险的两个问题：1.交叉租户
2.恶意运营商

很多工作都集中在如何确定云计算和公共云如何为租户带来新的漏洞。在一种类型的新风险中，当一个云租户试图破坏另一个租户的可识别性、可用性或完整性时，就会出现交叉租户。许多交叉租户的原因是，多租户意味着，一个人的应用程序可能在与另一个可能具有敌意的租户共享的计算服务器上运行。放置漏洞允许恶意的承租者安排将其应用程序安排在与目标相同的服务器上(放置)，从那里，跨承租者的侧通道可以允许违反隔离边界，例如，窃取机密或计算资源，或进行拒绝服务(DoS)攻击

applications are running on a compute server shared with another tenant who might be hostile. Placement vulnerabilities allow malicious tenants to arrange for their applications to be scheduled (placed) on the same server as a target,[2–4] and from there, cross-tenant side-channel attacks could allow violating isolation boundaries to, for example, steal secrets[5,6] or computing resources,[7] or to mount denial-of-service (DoS) attacks.

A second type of new risk is from a cloud operator who is itself malicious. This might seem, at first glance, to be a strange perspective: Why would one use a cloud that's adversarial? In fact, this threat model covers many nuanced and realistic threats, such as a rogue employee's insider attack and even a careless provider's accidental errors in the handling of customer data and programs. The widespread occurrence of malicious outsiders intruding into the systems of even large, highly reputed technology companies also encourages such a pessimistic threat model.

另一种类型的新风险来自于一个本身就是恶意的云计算运营商。在frst看来，这似乎是一个奇怪的观点:为什么人们会使用一种对抗的云呢?事实上，这种威胁模型涵盖了许多微妙而现实的威胁，比如一个流氓雇员的内部攻击，甚至一个粗心的提供者在处理客户数据和程序时的偶然错误。恶意外来者侵入甚至是大型、知名的技术公司的系统，也助长了这种悲观的威胁模式。

These new risks have given rise to several influential research threads, notably works on cryptographic techniques such as verifiable outsourced computing and outsourced private computation via fully homomorphic encryption.[8,9] These lines of work strive to remove the cloud provider's software and administrators from the trusted computing base. However, this threat model inherently pushes more responsibility (at the very least, for cryptographic key management) from the security-conscious providers to tenants that are, on the whole, entirely unsophisticated when it comes to security. Another drawback to such approaches is their high performance overhead, which will remain even if innovations bring about orders-of-magnitude improvements.

这些新风险已经催生了几个有影响力的研究思路，尤其是在加密技术方面，如veri寓言的外包计算，以及通过完全同态加密技术外包的私人计算。
这些工作线努力将云提供商的软件和管理员从受信任的计算库中删除。
然而，这种威胁模型从安全意识的提供者中，将更多的责任(至少，加密密钥管理)从安全的方面推给了租户，而这些租户在安全性方面是完全不成熟的。
这种方法的另一个缺点是它们的高性能开销，即使创新带来了数量级的改进，这种开销仍然存在。

Another Viewpoint: Provider as Partner

The above research agendas can be viewed as helping tenants to both understand and preempt new vulnerabilities that arise when moving to a cloud that's oblivious to its tenants' plights at best and openly hostile to them at worst. We believe these agendas are important to pursue further, but they also overlook a largely untapped opportunity: Can security-literate cloud providers help security-illiterate tenants resist threats? This viewpoint is already widely adopted in industry, where companies increasingly offer security-as-a-service solutions. In general, however, these solutions simply represent adoption of traditional security mechanisms in noncloud settings. Although default availability of "best practices" in security would mark important progress and make deployment and management simpler, we fear that the cloud computing industry isn't currently taking full advantage of the security opportunities that the shift to cloud computing offers.

上面的研究议程可以被看作是帮助租户理解和抢占新出现的漏洞，这些漏洞在移动到云的时候会被忽略，而在最坏的情况下，这些漏洞可能会对他们的用户产生不利的影响。
我们认为这些议程对于进一步发展是很重要的，但是他们也忽视了一个很大的未开发的机会:有安全知识的云服务提供商可以帮助不安全的不识字的租户抵御威胁吗?
这种观点已经在业界得到了广泛的应用，企业越来越多地提供安全服务的解决方案。
但是，一般来说，这些解决方案仅仅代表了在非云环境中采用传统安全机制。
尽管在安全方面的“最佳实践 ”的缺省可用性将标志着重要的进展，使部署和管理更加简单，但我们担心云计算行业目前并没有充分利用shif对云计算提供的安全机会。

These opportunities stem from the strategic position occupied by service providers to manage security on their tenants' behalf. In particular, providers can take advantage of deeper specialization and introspection, a broader perspective, and more compute resources.

对租户服务的客户端进行身份验证。
对几乎所有web事务进行身份验证都是对人类用户的验证。绝大多数情况下，最常用的方法是使用密码。然而，密码是一种不完美的保护。他们可以被攻击者猜测，或者，正如今天的行业大流行所显示的那样，他们可以通过破坏应用服务器并破解存储在那里的密码散列而被窃取。
我们描述了两种方法，一种云运算符可以帮助租户更安全地对客户进行身份验证:python和蜜语。这些解决方案旨在最大程度地减少未被发现的密码数据库泄露或用户模拟的风险。
普通租户可以使用它们来保护身份验证系统，但作为增强身份服务提供者基础设施的一种手段，它们尤其具有吸引力。
它们还可以促进对身份和访问管理的安全控制

更深层次的专业化。
在实践中，操作安全的一个反复出现的障碍是缺乏专业知识的专业人员。
基于云的安全服务允许许多组织从一些大型云服务提供商和安全服务提供者的更深入的专门化中获益。

Deeper specialization. A recurring obstacle to operational security in practice is a dearth of professionals with relevant expertise. Cloud-based security services allow many organizations to benefit from the deeper specialization of a few large cloud providers and security service providers.

提供者自省。
云服务提供商拥有分析租户安全性的独特能力，我们认为这有助于促进外包安全管理。
利用内省的一个(众所周知的)挑战是提供者的观察和在租户环境中观察到的行为之间的语义差异，这可能是高度定制的。
另一个挑战是云服务提供商不愿意访问租户数据，原因是租户数据的敏感性和数据共享对数据的要求。
但是即使是轻量级的自省也能在提高安全性方面产生巨大的回报

Provider introspection. Cloud providers have the unique
ability to analyze tenants' s
helps facilitate outsourced se
(well-known) challenge to t
spection is the semantic gap
observations and the significa
in tenant environments, whic
ized. Another challenge is cl
to access tenant data, due to
stemming from the sensitivit
regulatory requirements arou
lightweight introspection co
terms of improved security.

一个广泛的观点。
云服务提供了大量的客户，使云服务提供商能够获得与安全相关的信息的广泛视图。
聚集这些信息可能会带来比那些更小的组织所能获得的更深刻的见解—这是一种对威胁的“群体免疫”。

A broad view. Cloud services
customers, enabling cloud pr
view of security-relevant info
information could lead to de
obtainable by smaller organ
form of "herd immunity" to t

巨大的计算资源。
对云计算运营商所知道的信息的安全分析可能需要大量的计算和存储资源。但是大型云服务提供商可以访问这些资源。
这些战略利益与我们之前提到的其他共同的云研究议程有一定的分歧。例如，假定恶意提供者的许多研究议程试图阻止对租户应用程序的所有自省。我们认为，提供者与租户环境的交互不仅仅是威胁，也是改善租户安全性的一个机会。

Massive compute resources.
information knowable to clou
massive compute and storage
providers have access to these

These strategic benefits ar
the other common cloud re
tioned earlier. For example, m
das that presume malicious pr
introspection on tenant appl
provider interaction with ten
merely a threat but also an op
ant security.

python允许云计算操作员在应用服务器上加强密码，保护他们不受攻击时的妥协。
对于应用程序服务用户来说，这是透明的，只会带来最小的附加延迟，并且只需要对应用服务器进行少量的修改。
python依赖于一个伪随机函数(PRF)服务器，它可能由云计算操作员运行。
Te服务器应用一个PRF-a的确定性密码操作，涉及到应用服务器提交注册(存储)或验证的密码p，从而产生相应的输出x.Te PRF服务器从而“ 硬 ”的密码。
与密码散列不同的是，如果底层密码是弱的，它很容易受到暴力破解的攻击，而prf-硬化值x在计算上是不可行的，它可以让对手破解。
python提供了额外的安全特性。
它的部分未被遗忘的密码子，在PRF服务器上是隐式地隐藏在云计算操作系统中，但是python仍然允许云计算操作员执行对身份验证的帐户级监视。
Pythia还支持有效的关键更新，这意味着PRF服务器可以更新一个PRF键，并向应用服务器发送一个紧凑的更新令牌，允许每个用户的硬密码代表x更新。
这样的关键旋转使应用服务器(或PRF服务器)的破坏的影响无效。

A Silver Lining to the Cloud

在这里，我们提出了一种观点，云是如何能明显地提高租户的安全的。
我们将这一愿景分解为三个截然不同重叠的机会，利用云来帮助租户管理他们不受信任的客户群体、他们自己的外包基础设施，以及他们对云生态系统中的其他租户的依赖。

Here we lay out one vision of how clouds could significantly enhance security for their tenants. We dissect this vision into three distinct but overlapping opportunities for leveraging the cloud to help tenants manage their untrusted client populations, their own outsourced infrastructure, and their dependencies on other tenants in the cloud ecosystem.

帮助租户管理客户安全

Helping Tenants Manage Clients Securely

A substantial fraction of the effort to run an application service is consumed by addressing the risks posed by untrusted clients. These risks include client account takeovers, DoS attacks, and exploit attempts on the server. We see numerous opportunities to leverage the cloud's massive resources and elastic scaling capabilities to help tenant services defend against these risks. We outline several examples below.

运行应用程序服务的大部分工作都是通过处理不受信任的客户机带来的风险而消耗的。
这些风险包括客户帐户接管、DoS攻击和对服务器的攻击。
我们看到了大量利用云的巨大资源和弹性伸缩能力来帮助租户服务抵御这些风险的机会。
下面我们列出几个例子。

Authenticating clients of tenant services. Authenticating human users is central to virtually every web transaction. Overwhelmingly, the most common way to do this today is using passwords. Passwords are an imperfect protection, however. They can be guessed by an attacker or, as today's industry pandemic shows, they can be stolen by breaching the application server and cracking password hashes stored there.

We describe two approaches by which a cloud operator can aid tenants in authenticating clients more securely: Pythia and honeywords. These solutions aim to minimize the risk of an undetected password database breach or user impersonation attempt. They can be used by ordinary tenants to protect their authentication systems but are especially attractive as a means to strengthen the identity service providers' infrastructure. They can also facilitate compliance with security controls for identity and access management.

Pythia allows a cloud operator to harden passwords on an application server, protecting them against compromise during a breach.[10] It's transparent to application service users, imposes minimal additional latency, and requires only minor modifications to the application server.

Pythia relies on a pseudorandom function (PRF) server, potentially run by the cloud operator. The server applies a PRF—a deterministic cryptographic operation involving a secret key—to a password p that the application server submits for registration (storage) or verification, yielding a corresponding output x. The PRF server thereby "hardens" passwords. Unlike a password hash, which is vulnerable to brute-force cracking if the underlying password is weak, the PRF-hardened value x is computationally infeasible for an adversary to crack.

Pythia provides additional security features. It's partially oblivious—passwords submitted to the PRF server are cryptographically concealed from the cloud operator, yet Pythia still enables the cloud operator to perform account-level monitoring of authentication attempts. Pythia also supports efficient key updates, meaning that the PRF server can update a PRF key and send a compact update token to the application server that permits every user's hardened password representation x to be updated accordingly. Such key rotation nullifies the effects of a breach of the application server (or the PRF server).

One limitation of Pythia is that, to detect a breach, it relies on anomaly detection—an error-prone approach.

python的一个局限性是，为了检测到一个漏洞，它依赖于异常的检测——一个容易出错的方法。与此形成对比的是，我们开发的一个名为“蜜单词”的计划旨在检测密码系统的漏

In contrast, a scheme we've developed called honey- [illegible] n breaches.[11] [illegible] sible-looking pass- [illegible] ves storing for each [illegible] application server, [illegible] word *p*, but also a [illegible] An adversary that [illegible] e of distinguishing [illegible] honeywords. If the [illegible] attempts to authen- [illegible] is triggered. Given [illegible] ersary will evade [illegible] /*n* [illegible]

为了提供更多的细节，所有的n个密码，真实的和假的，都被存储在应用服务器上，随机的列表中。
当在与给定用户相关联的列表中使用一个密码时，关联的索引将被传递给一个由称为蜂蜜检查器的云操作符维护的系统。
Te蜂蜜检查器存储每个用户唯一的真实密码的索引(位置)，并区分真正的密码和蜜字的提交。
“蜜字系统”(像皮提亚)通过跨越承租者和云计算的环境，达到了突破的阻力。
稍后，我们将讨论蜂蜜对象的其他有趣用途，即虚假资源用于破坏检测和在云中出现敌对的错误方向。

t[illegible] words, real and fake, is stored on the application server in a randomly permuted list. When an authentication attempt occurs using a password [illegible]ed with [illegible]ssociated index is passed to a sys- [illegible] the [illegible] che [illegible] pas [illegible] he [illegible] the [illegible] ista [illegible] env [illegible] ss [illegible] res [illegible] tio [illegible]

抵挡DoS攻击租户的服务。应用程序服务器的第二个威胁,管理员必须解决DoS 攻击反对他们的服务器上。DoS攻击造成的伤害组织的收入损失和客户信任是众所周知的。DoS保护应用程序服务器今天主要以两种形式:专有的解决方案,规模巨大的负荷,但昂贵的(例如,Akamai)或硬件设备,将难题跟上对手能够动态地改变他们的攻击类型,体积,和地点。

作为银项目的一部分,我们设想一个基于云的DoS防御体系结构提供了无缝的地方防御机制的灵活性和弹性发射防御需要它们的地方在必要时根据攻击类型和规模。作为一个概念验证,我们开发了Bohatei系统,利用先进的几个sofware-defned网络虚拟化功能和网络功能。Secifcally,Bohatei利用这些功能弹性适应所需的规模和类型的防御和引导可疑流量通过防御部署在云位置合适。

Fending off DoS attacks against tenant services. A second threat that administrators of application servers must address is DoS attacks against their servers. The damage that DoS attacks cause to organizations in terms of lost revenue and customer trust is well known. DoS defense of application servers today comes primarily in two forms: proprietary solutions that scale to massive load but that are expensive (for instance, Akamai) or hardware appliances that will have difficulty keeping up with adversaries' ability to dynamically change their attacks' type, volume, and locations.

As part of the Silver project, we envision a cloud-based DoS defense architecture that provides the flexibility to seamlessly place defense mechanisms where they're needed and the elasticity to launch defenses as necessary depending on the attack type and scale. As a proof of concept, we developed the Bohatei system, which leverages several of the advanced software-defined networking and network function virtualization capabilities.[12] Specifically, Bohatei leverages these capabilities to elastically adapt the scale and type of defenses needed and to steer suspicious traffic through the defenses deployed at suitable cloud locations.

> **By focusing on the root cause of side channels (that is, coresidency), Nomad is agnostic to the specific side-channel vector used.**

Detecting exploit traffic against tenant services. Another threat type that administrators of application servers need to constantly fend off is exploit attempts against their servers. Sometimes these exploit attempts target logic vulnerabilities in the application servers themselves; in others, they target component protocols that the application servers employ. In many cases, exploits against such vulnerabilities involve clients sending traffic that no legitimate client implementation would send. Examples of such exploits include 10 Common Vulnerabilities and Exposures (CVEs) since 2014 for OpenSSL alone, including the well-known Heartbleed vulnerability (CVE-2014-0160).

We're developing a technique that leverages the cloud's spare compute resources and its visibility across tenant services to detect the emergence of new exploits as soon as they're attempted against any tenant. A cloud resident verifier analyzes the messages between a tenant server and its clients to detect messaging behavior from the client that's inconsistent with the expected client software. For example, our verifier can detect a client's deviation from an OpenSSL implementation of TLS within seconds from when the deviation occurs.[13] Because such deviations are typically characteristic of maliciously crafted packets to exploit server vulnerabilities, this type of verification capability could reduce the delay to detect exploit attempts on zero-day vulnerabilities. For example, this technique could have detected Heartbleed packets within seconds of the first attempted exploit, with no Heartbleed-specific configuration.

检测利用交通对租户的服务。另一个威胁类型,应用程序服务器的管理员需要不断的抵挡利用努力反对他们的服务器上。有时,这些利用努力目标逻辑漏洞在应用程序服务器本身;在他人看来,他们目标组件应用服务器使用的协议。在许多情况下,利用这样的漏洞涉及客户发送trafc没有合法的客户端实现将派遣。这样利用的例子包括10个常见的漏洞和风险敞口(cf)自2014年以来仅OpenSSL,包括知名Heartbleed脆弱性(cve - 2014 - 0160)。我们开发一种技术,利用云的空闲计算资源及其跨租户服务能见度检测新出现的利用只要他们努力反对任何租户。云居民校验分析租户服务器和客户端之间的消息来检测从客户机传递行为与预期不一致的客户端软件。例如,我们verifer可以检测客户的偏离TLS的OpenSSL实现秒内发生偏差。13因为这样的偏差通常是利用服务器漏洞恶意crafed数据包的特征,这种类型的验证功能可以减少延迟检测利用零日漏洞上的努力。例如,这一技术可以发现Heartbleed包第一努力利用的几秒内,没有Heartbleed-specifc confguration。

Helping Tenants Manage Their Infrastructure

Numerous organizations outsource portions of their own IT infrastructure to clouds, even if only to serve intraorganization purposes while achieving the cost savings associated with cloud computing. We present several ways in which cloud operators could assist tenants in managing the security of their outsourced infrastructure.

帮助租户管理基础设施大量组织自己的IT基础设施部分外包给云,即使只是intraorganization用途,同时实现与云计算相关的成本节约。我们提出了几条云运营商可以帮助租户管理外包安全的基础设施

Side-channel defense. The basis for the dramatic cost savings enabled by clouds is the sharing of the resources that they enable so effectively. However, this sharing

边信道防御。戏剧性的节约成本的基础上通过云的共享资源,它们使更有效的学习。然而,这种共享并不没有后果。正如我们之前所讨论的,研究表明,共享硬件资源会导致意外泄漏机密信息跨租户边界在云环境中。这些泄漏通道,出现由于租户对电脑的共享使用microarchitectural组件他们一起占领。事实上,这方面的渠道似乎不可避免的在当前的硬件平台上,如果中位数不受其他防御

OpenNF,第二个系统是补充FlowTags和专门设计来支持分布式处理跨多个middlebox实例。
虽然造成的虚拟化允许容易部署和拆卸的实例在云环境中,处理跨middlebox实例必须重新分配与再分配的协调状况的内部状态,造成维护他们处理。
OpenNF允许同步这些国家重新分配内部重新分配决策和安全、一致。
这功能使小说安全应用,例如,安全应用程序的能力可以动态地检测复杂atacks增强。
答是,现场middlebox(例如,一个深层数据包检测引擎)使用简单,本地处理来识别可疑trafc访问的句型。
当这样的句型是观察,深入分析这些句型无缝迁移(连同国家创建了到目前为止的本地实例的处理)能力更强,cloud-resident设备

结果,在银我们领先的发展operator-supported防御通道在云环境中,从专业防御两侧通道处理器缓存更宽范围的整体防御起源于coresidency边信道攻击。
cache-specifc防御的一个例子是虚拟机监控程序调度程序修改之后,以确保一个虚拟机(VM)不能抢占一个非常fne粒度,因为这是一个成分已知的边信道攻击利用每个核缓存。游牧是一个更全面的国防扩展这个想法通过provider-assisted服务,限制跨租户信息泄漏通过仔细协调虚拟机的放置和迁移。通过关注的根源方面渠道(即coresidency),游牧是不可知论者靠近边信道矢量

租户网络。
今天,许多组织严重依赖网络功能(NFs)或造成实施先进的安全功能,包括入侵检测和预防、监测、深层数据包检测和防火墙。
经常,这些功能是“链接”,与交通流量路由沿着这些序列造成执行关键的安全策略。
作为这些组织他们的计算基础设施进入公共云,它成为重要的在这些新的设置实现相同的功能。
为了促进这一点,我们的工作是利用云基础设施的关键属性,即它的虚拟化和软件定义。
我们认为这不仅支持灵活的实现上面的功能还允许新功能在云中实现上下文。

具体地说,我们已经开发了三个框架来说明这些可能性。第一,FlowTags,允许灵活的路由在任意链造成的交通,甚至造成改变数据包路由传统上依赖的头信息。FlowTags达到这样的目标,将标签插入end-toend流动;的逻辑计算标签驻留在一个逻辑上的中央控制器,利用高级策略来确定所需的标签编码端到端路径和任何middlebox-internal沿着路线行动(例如,服务内容的缓存)。标签可以被造成,使他们过程的上下文中处理流到目前为止收到的(例如,它遍历一组具体的造成)。这允许实现端到端traffic-steering政策具体的流动。

doesn't come without consequences. As we discussed, research has shown that sharing hardware resources can cause the unintentional leakage of secret information across tenant boundaries in cloud contexts. These leakages, called side channels, arise due to tenants' shared use of microarchitectural components on the computers they occupy together. Indeed, such side channels seem inevitable on current hardware platforms, if left unchecked by other defenses.

As a result, in Silver we're leading the development of operator-supported defenses against side channels in cloud contexts, ranging from specialized defenses against side channels in processor caches to more holistic defenses for wide ranges of side-channel attacks arising from coresidency. An example of a cache-specific defense is hypervisor scheduler modifications to ensure that one virtual machine (VM) can't preempt another with very fine granularity,[14] as this is an ingredient in known side-channel attacks leveraging per-core caches. Nomad is an example of a more holistic defense that extends this idea via a provider-assisted service that limits cross-tenant information leakage by carefully coordinating the placement and migration of VMs.[15] By focusing on the root cause of side channels (that is, coresidency), Nomad is agnostic to the specific side-channel vector used.

Tenant networking. Today, many organizations rely heavily on network functions (NFs) or middleboxes to implement sophisticated security functionality, including intrusion detection and prevention, monitoring, deep-packet inspection, and firewalls. Often, many of these functions are "chained," with traffic flows routed along a sequence of these middleboxes to enforce key security policies. As these organizations move their compute infrastructures into public clouds, it becomes important to realize equivalent functionality in these new settings. To facilitate this, our work is leveraging key properties of the cloud infrastructure, namely that it's virtualized and software defined. We argue that this not only enables flexible realization of the above functionality but also allows new functionality to be realized in the cloud context.

Specifically, we've developed three frameworks to illustrate these possibilities. The first, FlowTags, allows flexible routing of traffic across arbitrary chains of middleboxes, even as middleboxes alter packet header information on which routing traditionally relies.[16] FlowTags achieves this by inserting tags into end-to-end flows; the logic for computing tags resides at a logically central controller that leverages high-level policy to determine how the tags encode required end-to-end paths and any middlebox-internal actions taken along a route (for instance, content being served out of a cache). The tags can be consumed by middleboxes, enabling them to process the context of processing received by a flow so far (for instance, that it traversed a specific set of middleboxes). This allows end-to-end traffic-steering policies to be implemented for specific sets of flows.

The second system, OpenNF, is complementary to FlowTags and designed specifically to support distributed processing across multiple middlebox instances.[17] Although virtualization of middleboxes allows easy deployment and teardown of instances in a cloud setting, reallocation of processing across middlebox instances must be coordinated with reallocation of the internal state that middleboxes maintain for the traffic they're processing. OpenNF allows such state reallocation to be synchronized with traffic reallocation decisions and to take place safely and consistently. This capability enables novel security applications, for instance, a security application whose capability can be dynamically enhanced to detect sophisticated attacks. That is, an on-site middlebox (for example, a deep-packet inspection engine) employs simple, local processing to identify suspicious traffic access patterns. When such patterns are observed, deeper analysis of those patterns is seamlessly migrated (along with the state created so far by the local instance's processing) to a more capable, cloud-resident appliance.

The third system, Policy Graph Abstraction (PGA), offers the capability to effectively utilize the other two.[18] In many organizations, policies are independently specified by different actors; for instance, department administrators might want to restrict access to servers they own to users with specific credentials, whereas an enterprise-wide policy might impose general constraints on who can access what resources. It's important to ensure that such independently specified policies are composed and implemented consistently in the underlying infrastructure. PGA provides operators a simple graphical interface to specify complex policies among different sets of end points, including policies on middlebox traversal and elastic scaling. Each policy can be supported individually using FlowTags and OpenNF.

Crucially, the PGA runtime analyzes multiple such policies for potential conflicts and, if none exist, quickly computes a routing configuration that ensures consistent policy enforcement.

至关重要的是,PGA运行时分析多个潜在的冲突,并实行这样的政策,如果没有存在,快速计算路由配置确保一致的策略实施。

Strengthening Tenant Ecosystems

Another research direction is to explore how cloud providers can broker trust among tenants. Cloud platforms continue to advance their offerings of foundational services, for example, managed storage, coordination and consensus, and security-enhancing services. We envision that new cloud services can help to mediate secure interactions among tenants and further enable

第三个系统,政策图抽象(PGA),提供了能够有效利用其他两个。在许多组织中,政策规定独立不同的演员;例如,部门管理员可能想要限制对服务器的访问他们自己的用户提供具体的凭证,而企业级策略可能强加限制谁可以访问哪些资源。重要的是要确保这样的独立政策由指定和实现持续在底层的基础设施。PGA运营商提供了一个简单的图形界面来指定不同的端点之间复杂的策略,包括政策middlebox遍历和弹性伸缩。每个策略可以单独使用FlowTags和OpenNF支持

加强租户生态系统的另一个研究方向是探索云提供商如何代理租户之间的信任。云平台基础服务的继续推进他们的产品，例如，存储管理，协调和共识，security-enhancing服务。我们设想，新的云服务可以帮助协调保护租户之间的交互，并进一步使租户提供安全基础服务和应用程序服务，即使没有先天的信任服务的所有者。灵活的信任管理，源于信任云基础设施提供商共享，可以提高云计算服务的开放的市场潜力。一个激动人心的情景是启用安全合作共享的数据和代码分析，在分析软件作为一种服务提供给计算与数据集或主人考虑confdential的算法。共同的目标是使可信计算，结合confdential从多个所有者和产生保护隐私数据输出。安全数据共享可以帮助解锁的潜力大数据在数据隐私是最重要的地区，例如，医疗保健。我们正在开发基于云计算技术将这种合作更安全的基础。相互信任的云提供商可以通过调解这些跨租户加强安全交互。一个显而易见的方法这些目标是扩展云授权模型允许富裕政策控制数据共享和其他租户之间的交互。更雄心勃勃的方向是建立新的基于云的信托服务，使客户获得信任在租户服务没有先验知识或相信它的主人的身份。在我们的方法中，信任的云提供商作为根证明事实对租户安全属性或代码标识。其他租户可能指定信任策略评估对这些断言。例如，数据所有者可能信任第三方服务访问敏感数据集，如果云提供商atest服务里面的限制如何沟通或发布信息

tenants to offer secure foundational services and application services to one another, even without a priori trust in the service owner. Flexible trust management, rooted in shared trust in cloud infrastructure providers, can enhance the potential for an open marketplace of cloud-based services.

One motivating scenario is to enable secure sharing of data and code for cooperative analytics, in which analytics software is offered as a service for computing with datasets or algorithms that their owners consider confidential. A goal is to enable jointly trusted computations that combine confidential datasets from multiple owners and produce a privacy-preserving output. Safe data sharing can help unlock the potential of big data in areas where data privacy is paramount, for instance, healthcare. We're developing cloud-based technologies to place such collaborations on more secure foundations.

> **Flexible trust management, rooted in shared trust in cloud infrastructure providers, can enhance the potential for an open marketplace of cloud-based services.**

A mutually trusted cloud provider can enhance security by mediating these kinds of cross-tenant interactions. One obvious way to approach these goals is to extend cloud authorization models to allow richer policy control over data sharing and other interactions among tenants. A more ambitious direction is to establish new cloud-based trust services that enable clients to derive trust in a tenant service without prior knowledge or trust in the identity of its owner. In our approach, the cloud provider serves as a root of trust by certifying facts about the tenant security properties or code identity. Other tenants might specify trust policies to evaluate against these assertions. For example, a data owner might trust a third-party service to access a sensitive dataset if the cloud provider attests that the service is contained—it's restricted in how it can communicate or release information.

Cloud support for contained execution. We're developing an extended infrastructure-as-a-service (IaaS)-layer framework for managing contained execution, in which a group of tenant instances (VMs) has its network connectivity restricted according to a declared policy as a defense against information leakage. One system prototype, called CQSTR (pronounced "sequester"), implements a new cloud container abstraction as a set of extensions to the OpenStack IaaS platform.[19]

A CQSTR cloud container is a grouping of VM instances comprising an application deployment. A cloud container specifies containment properties that limit network and storage access for computations in the container. CQSTR modifies existing IaaS-level management services to ensure that backups, log monitoring, and other management services can't be abused to extract data from a closed container. In addition, the policy could specify the set of images that's allowable to boot VM instances into the container: the cloud platform provides a simplified form of code attestation using the cloud provider as a root of trust, rather than a hardware root of trust (for example, a Trusted Platform Module). Attesting to a limited set of boot images enables a client to ensure that a service runs on a patched, locked-down OS and a trusted application framework that might implement additional security controls.

We've experimented with several application scenarios that use cloud containers for secure analytics. In these scenarios, CQSTR enables a data owner to enforce control over how its data is used by an analytics service. The owner can demand and verify that data is held securely in a cloud container that's safe from data leakage and misuse. With a cloud container, code interacting with a service can contact CQSTR to verify the service's containment and code properties in advance. A data owner can specify access-control lists (ACLs) with the containment properties needed to access the data. CQSTR extends IaaS storage services to be aware of cloud containers: storage services can base access control on the declared container properties governing the calling VM instance, according to the ACL policies specified by the data owner.

Building trust in tenant services. CQSTR is just one example of a cloud provider service that makes trusted statements about the security properties of a tenant's configuration and allows other client software to check these properties for compliance with a security policy. Other useful security properties available to the cloud platform include a tenant's firewall posture, whether its software is patched adequately, whether it runs defensive (for instance, antivirus) software, whether it encrypts its stored data, whether its password system is protected by Pythia, attestations of software identity, network security services, and so forth. They might also reflect continuous auditing or monitoring checks or incident history. Clients can use this information to make informed decisions about whether a service is trustworthy, rather than relying merely on its reputation, as is common today.

我们开发一个扩展“基础架构即服务”(IaaS)层框架包含执行总经理的一群租户实例(VMs)有其网络连接限制根据宣布政策作为一个防御信息泄漏。
一个系统原型，叫做CQSTR(发音“隔离”)，实现了一个新的云容器抽象为一组扩展OpenStack IaaS平台。
19 CQSTR云的容器是一个分组VM实例组成的应用程序部署。
云容器指定容器属性that.limit计算的网络和存储访问容器。
CQSTR modifes现有IaaS-level管理服务，确保备份、日志监控，和其他管理服务不能被滥用来提取数据从一个封闭的容器。
此外，该策略可能指定的图像集的容许向容器启动VM实例：云平台提供了一种简化的代码使用云提供商atestation作为根的信任，而不是一个硬件根的信任(例如，可信平台模块)。
atest一组有限的启动图像使客户以确保服务运行在修补，锁定的操作系统和受信任的应用程序框架，它可以实现额外的安全控制。
我们已经试验了几种应用场景，使用云安全分析的容器。
在这些场景中，CQSTR使数据所有者执行控制如何使用其数据分析服务。
Te所有者可以安全地举行需求和验证数据在云容器的安全数据泄露和滥用。
与云容器，代码交互与服务可以联系CQSTR验证服务的容器和代码属性。
数据所有者可以指定访问控制列表(acl)容器属性需要访问数据。
CQSTR延伸IaaS云存储服务需要注意的容器：存储服务可以访问控制基于声明的容器属性管理调用VM实例，根据ACL策略specifed数据所有者。

CQSTR只是云提供者服务的一个例子使信任的语句对租户的安全属性的配置，并允许其他客户端软件来检查这些属性符合安全策略。
云平台可用其他有用的安全属性包括租户的防火墙的姿势，是否充分其软件打补丁，无论是防守(例如，杀毒)软件运行，无论是其存储的数据进行加密，密码系统是皮提亚的保护，是否atestations软件的身份，网络安全服务，等等。他们也可能反映了连续审计监督检查或历史事件。客户端可以使用这些信息来做出明智的决定服务是否值得信赖，而不是仅仅依靠它的声誉，因为今天是很常见的。

灵活性提供分层的平台是一个开放的云生态系统的基础。我们正在探索如何启用第三方PaaS服务的部署,从底层继承信托IaaS云系统通过认证,审核和介导IaaS-layer服务访问安全凭据。PaaS服务,反过来,可以利用其高层次的编程模型来执行基于语言安全检查或介于高级监视或控制。此外,我们相信,PaaS平台是值得信赖的承诺目标计算通过软件认证和基于代码的访问控制。我们的前提是更高级别的PaaS项目实践来验证和证明比二进制可执行文件,因为它们紧凑:他们建立在强大的语言和库的标准原语实现可以信任。作为一个例子,我们正在开发最小信任扩展标准的火花分析堆栈(spark.apache.org)提供一个PaaS服务安全合作分析。它提供了丰富的访问控制,允许数据所有者控制数据共享与其他租户按照自己的方式。PaaS服务证明代码标识的阶段分析工作流:政党可以指定相互信任的火花可以输入敏感数据的程序,可能来自多个业主,并生成输出“ 解密 ”是安全的。通过工作流系统跟踪流程,以确保安全标签为每个对象反映其内容的潜在的敏感性

我们的研究旨在提供一个通用和实用的基础管理信任云生态系统身份验证断言和政策的基础上,建立在大量的先前工作授权逻辑。我们的方法解决一些潜在的问题。首先,政策应该富有表现力的语言,高效、使用方便、可扩展的词汇增长的安全属性。第二,应该保护敏感数据的保密的方法,包括安全配置和政策本身。我们开发了一个基于逻辑的声明式语言(信任逻辑)和翻译软件(称为安全),使参与entities-including服务云提供商的租户,外部服务,和客户端软件对最终用户问题验证断言,从别人的断言和原因。Te语言也表示逻辑策略规则,verifable。声明性政策使促成合规检查,客户端提交声明式安全策略信任策略引擎(翻译)的控制或atested-by云平台:策略引擎检查遵守政策没有揭示政策或任何人的安全属性。保护隐私的合规中介的一个例子是一个安全的基础服务协调租户交互的云。安全适用于更一般的信任管理在联邦环境中,包括系统跨越多个网络云提供商(例如,ExoGENI)。在这种情况下,参与者可以交换安全安全断言和政策规则certifcates签署并运行一个当地的现成的翻译生成策略遵从性的证明。安全也是一个更一般的网络云系统中访问控制的基础。例如,我们实现了一个可重用的规则包嵌套组和角色在一个安全层次名称空间,威力相当于的命名和授权结构亚马逊网络服务(AWS)身份和访问管理但适用于多畴的系统而不是依靠一个信任锚。安全也可以支持丰富的政策授权租户之间的互连和与外部网络的连接

越来越多的商业云运营商提供更高级的平台抽象(平台或PaaS)租户,例如,Google AppEngine和AWS弹性MapReduce。PaaS云系统简化编程与更强大的模型,加强客户对云服务生产率和增加价值。PaaS系统也由租户提供其他租户(例如,Heroku和CloudFoundry)

Our research seeks to provide a general and practical foundation to manage trust in cloud ecosystems based on authenticated assertions and policies, building on a wealth of prior work in authorization logics. Our approach addresses several potential concerns. First, the policy language should be expressive, efficient, easy to use, and extensible to a growing vocabulary of security properties. Second, the approach should protect the secrecy of sensitive data, including security configurations and the policies themselves.

We developed a declarative logic-based language (a trust logic) and interpreter software (called SAFE) to enable participating entities—including services of the cloud providers, tenants, external services, and client software for end users—to issue authenticated assertions about one another, and reason from the assertions of others. The language also expresses logical policy rules, which are verifiable. Declarative policy enables brokered compliance checks, in which a client submits a declarative security policy to a trusted policy engine (interpreter) that's operated—or attested—by the cloud platform: the policy engine checks compliance with the policy without revealing the policy or the security properties to anyone. A privacy-preserving compliance intermediary is an example of a secure foundational service for mediating tenant interactions in the cloud.

SAFE is suitable for more general trust management in federated environments, including systems spanning multiple networked cloud providers (for instance, ExoGENI). In this case, the participants can exchange SAFE security assertions and policy rules as signed certificates and run a local off-the-shelf interpreter to generate proofs of policy compliance end to end. SAFE also serves as a basis for more general access control in networked cloud systems. For example, we've implemented a reusable package of rules for nested groups and roles in a secure hierarchical name space, equivalent in power to the naming and authorization structure of Amazon Web Services (AWS) Identity and Access Management but applicable to multidomain systems rather than relying on a single trust anchor. SAFE can also support rich policies to authorize interconnection among tenants and connectivity with external networks.

Securing the PaaS layer. Increasingly, commercial cloud operators offer higher-level platform abstractions (platform-as-a-service, or PaaS) for tenants, for instance, Google's AppEngine and AWS Elastic MapReduce. PaaS systems simplify cloud programming with more powerful models that enhance customer productivity and add value to cloud services. PaaS systems are also offered by tenants to other tenants (for instance, Heroku and CloudFoundry).

Flexibility to offer layered platforms is fundamental to an open cloud ecosystem. We're exploring how to enable deployment of third-party PaaS services that inherit trust from the underlying IaaS cloud system via attestation, auditing, and mediated access to security credentials for IaaS-layer services. The PaaS service, in turn, could leverage its higher-level programming model to enforce language-based safety checks or interpose higher-level monitoring or containment. In addition, we believe that PaaS platforms are promising targets for trustworthy computing via software attestation and code-based access control. Our premise is that higher-level PaaS programs are more practical to verify and attest than binary executables because they're compact: they build on powerful languages and a library of standard primitives whose implementations can be trusted.

As one example, we're developing minimal trust extensions to a standard Spark analytics stack (spark.apache.org) to provide a PaaS service for secure cooperative analytics. It offers rich access control that allows data owners to regulate data sharing with other tenants on their own terms. The PaaS service attests to code identity for the stages of the analytics workflow: parties can designate mutually trusted Spark programs that can input sensitive data, possibly from multiple owners, and generate "declassified" outputs that are safe to share. The system tracks flow through the workflow to ensure that the security label for each object reflects its contents' potential sensitivity.

Industry-Inspired Challenges

我们问同事在工业列举他们认为 云安全问题是最紧迫的业务和业务需求。他们反应不仅存在一系列开放的挑战和机遇,也帮助照亮的方式“ 提供者作为受信任伙伴 ” 的角色可以在云安全带来进一步发展。我们提供几个例子。

We asked colleague [illegible] the cloud security problems t [illegible] for their operational and bus [illegible] s not only present a range of o [illegible] nities but also help illuminate [illegible] "provider as trusted partner" [illegible] vances in cloud security. We [illegible]

Securing Security Logs

Security logs are important for forensic investigation of security events and increasingly also in security analytics systems, which aim to detect anomalous events indicative of security compromises. Industry practitioners have stressed the importance of ensuring the integrity of security logs; the ability to extract meaningful intelligence from them remains a perennial technical challenge.

Although forward-secure cryptography has been advocated for securing logs in a device, it can only detect tampering, not prevent it. However, cloud operators could provide several services for securing logs against tampering despite compromise of a tenant's operational environment. It could, for example, provide a real-time

安全日志对法医调查很重要的安全事件也越来越安全分析系统,旨在检测异常事件表明安全妥协。行业从业人员强调的重要性,确保安全日志的完整性;提取有意义的情报从他们的能力仍然是一个常年技术挑战虽然forward-secure密码学一直主张保护登录设备,它只能检测出篡改,而不是阻止它。然而,云计算运营商可以提供一些服务来保护日志对篡改尽管妥协的租户的操作环境。例如,它可以提供一个实时管道日志,迅速消除从租户的日志存储在一个环境operator-secured环境。Te运营商可以另外丰富日志来提高他们在安全分析工具。例如,提供者可以提供一个统一的时间戳服务,这将解决一些基本的同步问题确定在今天的日志系统。此外,利用其广泛的观点,一个云运营商可能执行安全分析在多租户的日志数据,使识别通过比较广泛的攻击和差异化的、有针对性的攻击。这个机会的主要障碍是扩展的挑战,更重要的是,需要保护租户的机密性:尽管租户可以执行安全分析的数据,安全分析包括多个潜在相互日后组织需要计算在数据相结合。云计算支持包含执行提供了一个有前途的方法

pipe for logs, quickly removing logs from a tenant's environment for storage in an operator-secured environment. The operator could additionally enrich logs to improve their utility in security analytics. For example, the provider could offer a uniform timestamping service, which would address some of the fundamental synchronization issues identified in today's logging systems.

> **There's an opportunity to deploy honey objects in a tenant's environment with the cloud operator's support.**

Furthermore, capitalizing on its broad view, a cloud operator could perform security analytics over the log data of multiple tenants, enabling identification of broad attacks and differentiation of targeted attacks through comparison. The major impediments to this opportunity are the challenges of scaling and, more fundamentally, the need to protect tenants' confidentiality: although a tenant can perform security analytics on its own data, security analytics encompassing multiple potentially mutually mistrusting organizations would require computation over combined data. Cloud support for contained execution offers a promising approach.

Security Control and Vulnerability Mapping

Another opportunity in cloud systems that industry practitioners have identified is the need to map security controls, as defined in information security standards, such as ISO/IEC 27001:2013 and NIST Special Publication SP 800-53, to policies and technical enforcement mechanisms. In most organizations, this process is a time-consuming and labor-intensive business requirement; it involves inventorying systems and assets in conjunction with a systematic review of information security policies. However, a cloud operator—particularly one offering security services to its tenants—can in principle leverage economies of scale and homogeneous elements of its tenants' environments to streamline such mappings.

As an example, the Cloud Security Alliance Cloud Controls Matrix v. 3.0.1 control IAM-01, Identity & Access Management: Audit Tools Access, specifies that "Access to, and use of, audit tools that interact with the organization's information systems shall be appropriately segregated and access restricted to prevent inappropriate disclosure and tampering of log data." In implementing protections for security logs such as those we described earlier, a cloud operator can facilitate compliance with controls of this type by providing management tools for tenants and/or incorporating compliant controls into applications that the operator itself provides. A further opportunity exists when software vulnerabilities are identified. An organization must then identify affected systems and formulate and prioritize remediation plans.

Tools such as Amazon CloudTrail (for logging) and Amazon Inspector go some way toward realizing this vision of automated control/vulnerability mapping but are in their infancy and limited in scope. For example, CloudTrail handles only AWS API calls. Significant opportunities exist in extending the reach and sophistication of these tools.

Thwarting Adversarial Reconnaissance

A hallmark of recent advanced persistent threats is their apparent reliance on extensive preliminary reconnaissance. Industry practitioners have thus highlighted the need for cloud-based resources to mitigate reconnaissance, particularly given the risks that aggregation of organizations in the cloud presents.

One opportunity here lies in counterintelligence using decoy or honey objects. Honeywords is one example of such objects. But there's a much more general opportunity to deploy honey objects in a tenant's environment with the cloud operator's support. The cloud operator could create and monitor honey files or documents, honeypots, honeytokens, or honeynets.[20]

Cloud operator support of honey objects offers several attractive features. First, a cloud operator can maintain state outside a tenant's environment that distinguishes between real and honey objects; such state can thus be protected from compromises that occur inside the tenant's environment. Second, the cloud operator can observe adversarial interaction with honey objects via introspection, enabling well-concealed breach detection and monitoring of adversarial behavior. Finally, the cloud operator is well positioned to take an active role in honey object deployment. Thanks to the cloud's elasticity, servers and entire networks can quickly be spun up upon suspicion of compromise or in a way tailored to the apparent behavior of an adversary, enabling real-time deployment of strategies for misdirection or provision of misinformation to the adversary.

Cloud computing has unquestionably transformed the computing landscape. We believe, however, that opportunities for further transformation made possible by the move to clouds remain underexplored,

另一个机会在云系统行业从业者已经确定是需要地图的安全控制,defned在信息安全标准,如ISO / IEC 27001:2013和NIST的特殊出版SP 800 - 53年,政策和技术实施机制。在大多数组织中,这个过程是一个耗时和劳动密集型业务需求;它包括盘点系统和资产与信息安全政策的系统回顾。然而,云operator-particularly原则上它提供安全服务一个租户可以利用规模经济和齐次元素租户的环境来简化等映射的一个例子,云安全联盟云控制矩阵诉3.0.1控制IAM-01、身份和访问管理:审计工具访问指定“访问和使用,审计工具,与组织的信息系统应当适当地隔离和访问限制,以防止不适当的信息披露和日志数据的篡改。”在实施保护安全日志如我们前面所述,云运营商可以促进符合这种类型的控制通过为租户提供管理工具和/或将兼容的控制纳入运营商本身提供的应用程序。软件漏洞识别时存在进一步的机会。一个组织必须识别受影响的系统和制定和优化修复计划。工具,如亚马逊CloudTrail(日志)和亚马逊检查员一定程度上对实现这一愿景的自动控制/脆弱性映射但还在起步阶段和范围有限。例如,CloudTrail处理只AWS API调用。重要机遇存在于扩展这些工具的范围和复杂性

最近的先进的持续威胁的一个特点是他们明显依赖广泛初步侦查。行业从业者就强调了基于云计算的资源需要减轻侦察,特别是考虑到风险聚集的组织在云中礼物。一个机会在于反间谍使用诱饵或蜂蜜对象。Honeywords是这样一个例子。但是有一个更一般的机会来部署蜂蜜租户的环境中对象与云运营商的支持。云计算运营商可以创建和监视蜂蜜fl或文档,“粘蜜罐”、honeytokens或蜜网。20云运营商支持的蜂蜜对象提供了几种atractive特性。首先,云运营商维护状态租户的外环境,区分真实和蜂蜜对象;这种状态可以从内部发生的妥协从而保护房客的环境。第二,云运营商通过内省,与蜂蜜可以观察到敌对的互动对象启用和违反敌对行为的检测和监控。最后,云计算运营商也将积极参与蜂蜜对象部署。坦克云的弹性、服务器和整个网络可以迅速旋转上涉嫌妥协或根据敌人的明显的行为方式,使实时部署策略误导或提供错误信息的对

云计算无疑改变了计算景观。然而，我们相信，机会进一步转换由移动云仍未开发，特别是利用云来改善他们的租户的安全。我们已经对比研究视觉占主导地位的趋势在今天的云计算安全研究的动机。我们也总结了一些研究方向的上下文中，我们正在探索这一愿景以及一些额外的机会识别通过与云运营商和租户社区的成员

particularly leveraging clouds to improve their tenants' security. We have contrasted this research vision to the dominant trends in security research motivated by cloud computing today. We also summarized several of the research directions we are exploring within the context of this vision as well as several additional opportunities identified through conversations with members of the cloud operator and tenant communities. ■

References

1. K. He et al., "Next Stop, the Cloud: Understanding Modern Web Service Deployment in EC2 and Azure," *Proc. Conf. Internet Measurement Conference* (IMC 13), 2013, pp. 177–190.
2. T. Ristenpart et al., "Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds," *Proc. 16th ACM Conf. Computer and Communications Security* (CCS 09), 2009, pp. 199–212.
3. V. Varadarajan et al., "A Placement Vulnerability Study in Multi-tenant Public Clouds," *Proc. 24th USENIX Security Symp.* (SEC 15), 2015, pp. 913–928.
4. Z. Xu, H. Wang, and Z. Wu, "A Measurement Study on Co-residence Threat inside the Cloud," *Proc. 24th USENIX Security Symp.* (SEC 15), 2015, pp. 929–944.
5. Y. Zhang et al., "Cross-VM Side Channels and Their Use to Extract Private Keys," *Proc. 19th ACM Conf. Computer and Communications Security* (CCS 12), 2012, pp. 305–316.
6. Y. Zhang et al., "Cross-Tenant Side-Channel Attacks in PaaS Clouds," *Proc. 21st ACM Conf. Computer and Communications Security* (CCS 14), 2014, pp. 990–1003.
7. V. Varadarajan et al., "Resource-Freeing Attacks: Improve Your Cloud Performance (at Your Neighbor's Expense)," *Proc. ACM Conf. Computer and Communications Security* (CCS 12), 2012, pp. 281–292.
8. M. Walfish and A.J. Blumberg, "Verifying Computations without Reexecuting Them," *Comm. ACM*, vol. 58, no. 2, 2015, pp. 74–84.
9. F. Armknecht et al., "A Guide to Fully Homomorphic Encryption," Cryptology ePrint Archive, report 2015/1192, 2015; eprint.iacr.org/2015/1192.pdf.
10. A. Everspaugh et al., "The Pythia PRF Service," *Proc. 24th USENIX Security Symp.* (SEC 15), 2015, pp. 547–562.
11. A. Juels and R.L. Rivest, "Honeywords: Making Password-Cracking Detectable," *Proc. ACM Conf. Computer and Communications Security* (CCS 13), 2013, pp. 145–160.
12. S.K. Fayaz et al., "Bohatei: Flexible and Elastic DDoS Defense," *Proc. 24th USENIX Security Symp.* (SEC 15), 2015, pp. 817–823.
13. A. Chi et al., "Server-Side Verification of Client Behavior in Cryptographic Protocols," *Proc. 14th USENIX Symp. Networked Systems Design and Implementation* (USENIX NSDI 17), 2017.
14. V. Varadarajan, T. Ristenpart, and M. Swift, "Scheduler-Based Defenses against Cross-VM Side-Channels," *Proc. 23rd USENIX Security Symp.* (USENIX Security 14), 2014.
15. S.-J. Moon, V. Sekar, and M.K. Reiter, "Nomad: Mitigating Arbitrary Cloud Side Channels via Provider-Assisted Migration," *Proc. 22nd ACM Conf. Computer and Communications Security* (CCS 15), 2015, pp. 1595–1606.
16. S.K. Fayazbakhsh et al., "Enforcing Network-Side Policies in the Presence of Dynamic Middlebox Actions Using Flowtags," *Proc. 11th USENIX Symp. Networked Systems Design and Implementation* (NSDI 14), 2014, pp. 533–546.
17. A. Gember et al., "OpenNF: Enabling Innovation in Network Function Control," *Proc. ACM Conf. Special Interest Group on Data Communication* (SIGCOMM 14), 2014, pp. 163–174.
18. C. Prakash et al., "PGA: Using Graphs to Express and Automatically Reconcile Network Policies," *Proc. ACM Conf. Special Interest Group on Data Communication* (SIGCOMM 15), 2015, pp. 29–42.
19. Y. Zhai et al., "CQSTR: Securing Cross-Tenant Applications with Cloud Containers," *Proc. ACM Symp. Cloud Computing* (SoCC 16), 2016, pp. 223–236.
20. F. Pouget, M. Dacier, and H. Debar, *Honeypot, Honeynet, Honeytoken: Terminological Issues*, white paper RR-03-081, Institut Eurecom, Sept. 2003.

Jay Aikat is a research associate professor in the Department of Computer Science at the University of North Carolina at Chapel Hill (UNC-CH). She's also the chief operating officer (COO) at the Renaissance Computing Institute (RENCI). Her research interests are in computer networking and cloud security. Aikat received a PhD in computer science from UNC-CH. She's a member of ACM. Contact her at aikat@cs.unc.edu.

Aditya Akella is an associate professor in the Department of Computer Sciences at the University of Wisconsin—Madison. He received a PhD from Carnegie Mellon University. Contact him at akella@cs.wisc.edu.

Jeffrey S. Chase is a professor of computer science at Duke University. His research interests include infrastructure control and trust management for networked services. Chase received a PhD in computer science from the University of Washington. He's a member of ACM and USENIX. Contact him at chase@cs.duke.edu.

Ari Juels is a professor at the Jacobs Institute at Cornell Tech. His current research interests include applied cryptography, blockchains and smart contracts, cloud security, and the use of deception in computer security. Juels received a PhD in computer science from the

University of California, Berkeley. He's a member of ACM. Contact him at juels@cornell.edu.

Michael K. Reiter is the Lawrence M. Slifkin Distinguished Professor in the Department of Computer Science at UNC-CH. His research interests include computer security, distributed computing, and networking. Reiter received a PhD in computer Science from Cornell University. He's a Fellow of ACM and IEEE. Contact him at reiter@cs.unc.edu.

Thomas Ristenpart is an associate professor at Cornell Tech. His research interests include security, privacy, and cryptography. Ristenpart received a PhD in computer science from the University of California, San Diego. Contact him at ristenpart@cornell.edu.

Vyas Sekar is an assistant professor in the Electrical and Computer Engineering Department at Carnegie Mellon University. His research interests lie at the intersection of networking, security, and systems. Sekar received a PhD in computer science from Carnegie Mellon University. Contact him at vsekar@andrew.cmu.edu.

Michael Swift is an associate professor in the Department of Computer Sciences at the University of Wisconsin—Madison. Swift received a PhD from the University of Washington. Contact him at swift@cs.wisc.edu.

现代云计算共享基础设施计算的系统是最新的例子。“效用计算”的想法大型机Multics项目的一个主要动机，激发了更早期的计算机安全研究。在这样的系统中，用户和企业远程访问中央大型机，掩盖了大量成本在许多用户。随着万维网的兴起在1990年代末，公共托管中心越来越受欢迎，提供裸机的机器和网络服务器管理。免费的Linux操作系统和Apache web服务器，启动了1991年和1995年，分别和服务器与商品Intel和AMD处理器，使其经济运行大型服务器集群来处理网络规模的增加的工作负载。两个创新是推动云计算革命。首先，VMware介绍x86处理器的管理程序，显示，虚拟化可以有效的和有用的。这是紧随其后的是开源Xen项目从2003年的剑桥大学。其次，亚马逊推出弹性计算云，租来的虚拟机与一个新的计费模式基于短期使用。几美分，一个客户可以租物理机器一个小时的一小部分。虚拟化的结合，efcient管理和短时间尺度计费使云计算的热潮。今天的公共云提供大规模计算基础设施以虚拟机的形式，应用程序或软件服务。之间的共同之处都是由多个租户的动态共享基础设施，这是由第三方管理。在这个共同的结构，云今天可以大致分为三类。“基础架构即服务”系统允许用户启动虚拟机和控制来宾操作系统以及应用程序。在平台系统中，顾客可以启动provider-managed OS应用程序执行。最后，saas云为客户提供托管应用程序服务，例如电子邮件、数据库、文件存储等。一个公司，如谷歌或微软，可以提供所有三种计算。部门为这些类别遵循水平在一个典型的软件平台，有一个从云提供商管理切换到客户。