谷歌 BeyondCorp 系列论文合集

CSA 大中华区 SDP 工作组
奇安信身份安全实验室　　译
二零一九年五月

前言：

随着企业大规模的采用移动互联网和云计算技术，传统的采用防火墙建立的“城堡”安全模式，变得越来越不安全。2014 年 12 月起，Google 先后发表 6 篇 BeyondCorp 相关论文，论文提供了一种新的安全模式，设备和用户只能获得经过验证的资源，构建软件定义安全的雏形。另外，论文也介绍了 BeyondCorp 的架构和实施情况，为传统网络架构迁移至 BeyondCorp 架构提供依据参考。

为推动国内安全技术和理论与国际同步，在国内传播国际优秀实践，中国云安全联盟秘书处组织专家翻译 BeyondCorp 相关论文，供大家学习参考。特别感谢 CSA 大中华区 SDP 工作组与奇安信身份安全实验室对本次翻译工作的贡献及支持！

本文档一共由 BeyondCorp 的六篇论文组合而成：

[1] BeyondCorp：一种新的企业安全方案

[2] 谷歌 BeyondCorp：从设计到部署

[3] BeyondCorp：访问代理

[4] 迁移到 BeyondCorp：提高安全性的同时保持生产力

[5] BeyondCorp ：用户体验

[6] BeyondCorp：构建健康机群

关于 CSA 大中华区 SDP 工作组：

CSA （Cloud Security Alliance）2008 年 12 月在美国发起，以云计算安全为开端，2011 年白宫在 CSA 峰会上宣布了美国联邦政府云计算战略，之后 CSA 演化成为独立、中立、非盈利的世界性产业组织，致力于全球下一代 IT 与新兴技术安全的全面发展。

为提高 Software Defined Perimeter（软件定义边界，即 SDP）在中国企业的应用，在中国云安全联盟的支持下 ，CSA 大中华区成立 SDP 工作组。工作组于 2019 年 3 月成立，首批参与单位有：阿里云、腾讯云、京东云、奇安信、深信服、绿盟科技、Ucloud、顺丰科技、天融信、云深互联、中宇万通、华云数据、三未信安、上元信安、安全狗、易安联、联软科技、上海云盾、缔盟云、缔安科技等三十多家单位。

关于奇安信身份安全实验室：

奇安信身份安全实验室，是奇安信集团下属专注“零信任身份安全架构”研究的专业实验室。该团队以“零信任安全，新身份边界”为技术思想，探索“企业物理边界正在瓦解、传统边界防护措施正在失效”这一时代背景下的新型安全体系架构，推出“以身份为中心、业务安全访问、持续信任评估、动态访问控制”为核心的奇安信天鉴零信任身份安全解决方案。该团队结合行业现状，大力投入对零信任安全架构的研究和产品标准化，积极推动“零信任身份安全架构”在业界的落地实践，其方案已经在部委、金融、央企等进行广泛落地实施，得到市场、业界的高度认可。

参与本文档翻译的专家（按照姓名拼音排序）：

组长：陈本峰（云深互联）

组员：高健凯、刘德林、张泽洲（奇安信）

【第一篇】BeyondCorp：一种新的企业安全方案

如今，几乎所有企业都会采用防火墙来建立安全边界，然而，这种安全模型存在问题：一旦边界被突破，攻击者可以畅通无阻地访问企业的特权内部网络。另一方面，随着企业大规模地采用移动互联网和云计算技术，边界防护变得越来越难。谷歌采用了不同的网络安全方法，逐步摆脱对特权内网的依赖，越来越多地将企业应用程序从内网迁移至公网。

从 IT 基础设施诞生以来，企业一向使用边界防御措施来保护对内部资源的访问。边界安全模型通常被比作中世纪的城堡：有着厚厚的城墙，被护城河环绕，仅有一个守卫森严的入口和出口，任何墙外的东西都被认为是危险的，任何墙内的东西都认为是安全可信的，这也就意味着任何能通过吊桥的人都能获得城堡内的资源。

当所有员工都只在企业办公大楼中工作时，边界安全模型确实很有效；然而，随着移动办公的出现、办公使用的设备种类激增、云计算服务的使用越来越广泛、新的攻击向量也随之增加，如上因素逐渐导致传统安全手段形同虚设。边界安全模型所依赖的关键假设不再成立：边界不再由企业的物理位置决定，边界之内也不再是个人设备和企业应用运行的安全地带。

大部分企业假设内部网络是安全的环境并且企业应用可以放心暴露在内网，但谷歌的经验证明了这种观念是错误的。应该假设企业内网与公网一样充满危险，并基于这种假设构建企业应用。

谷歌 BeyondCorp 的目标是摒弃对企业特权网络（内网）的依赖并开创一种全新安全访问模式，在这种全新的无特权内网访问模式下，访问只依赖于设备和用户身份凭证，而与用户所处的网络位置无关。无论用户是在公司“内网”、家庭网络、酒店还是咖啡店的公共网络，所有对企业资源的访问都要基于设备状态和用户身份凭证进行认证、授权和加密。这种新模式可以针对不同的企业资源进行细粒度的访问控制，所有谷歌员工不再需要通过传统的 VPN 连接进入内网，而是允许从任何网络成功发起访问，除了可能存在的网络延迟差异外，对企业资源的本地和远程访问在用户体验上基本一致。

BeyondCorp 的关键组件

BeyondCorp 由许多相互协作的组件组成，以确保只有通过严格认证的设备和用户才能被授权访问所需的企业应用，各组件描述如下（见图 1）。

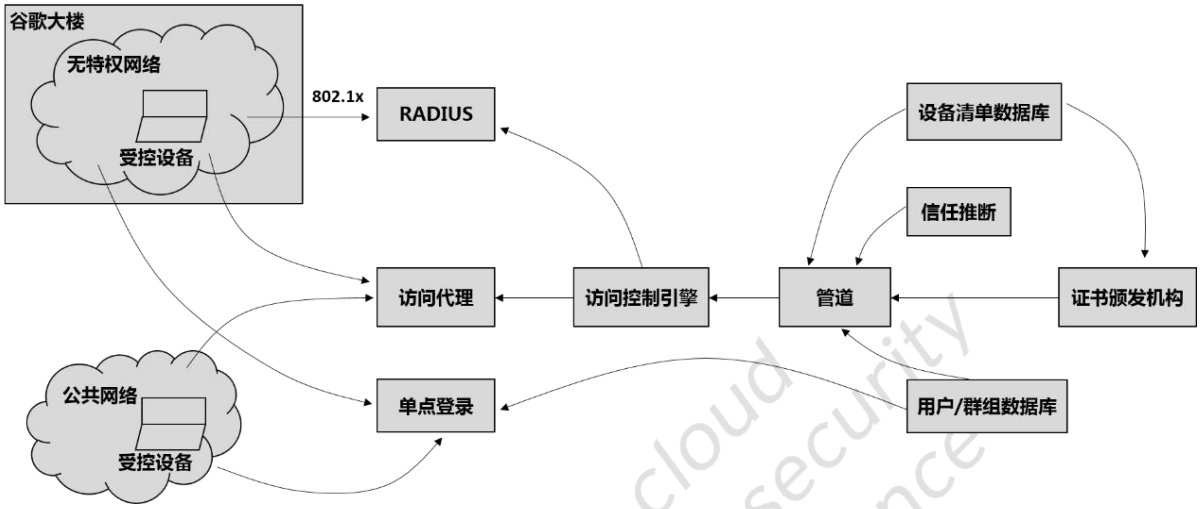

图 1 BeyondCorp 的组件和访问流

安全识别设备

设备清单数据库

BeyondCorp 使用了“受控设备”的概念——由企业采购并管理可控的设备。只有受控设备才能访问企业应用。围绕着设备清单数据库的设备跟踪和采购流程管理是这个模型的基础之一。在设备的全生命周期中，谷歌会追踪设备发生的变化，这些信息会被监控、分析，并提供给 BeyondCorp 的其他组件进行分析和使用。因为谷歌有多个清单数据库，所以需要使用一个元清单数据库对来自多个数据源的设备信息合并和规一化，并将信息提供给 BeyondCorp 的下游组件。通过元清单数据库，我们就掌握了所有需要访问企业应用的设备信息。

设备标识

所有受控设备都需要一个唯一标识，此标识同时可作为设备清单数据库中对应记录的索引值。实现方法之一是为每台设备签发特定的设备证书。只有在设备清单数据库中存在且信息正确的设备才能获得证书。证书存储在硬件或软件形态的可信平台模块（Trusted Platform Module, TPM）或可靠的系统证书库之中。设备认证过程需要验证证书存储区的有效性，只有被认为足够安全的设备才可以被归类为受控设备。当进行证书定期轮换时，这些安全检查也会被执行。一旦安装完毕，证书将用于企业服务的所有通信。虽然证书能够唯一地标识设备，但仅凭证书不能获取访问权限，证书只是用来获取设备的相关信息。

安全识别用户

用户和群组数据库

BeyondCorp 还跟踪和管理用户数据库和用户群组数据库中的所有用户。用户/群组数据库系统与谷歌的 HR 流程紧密集成，管理着所有用户的岗位分类、用户名和群组成员关系，当员工入职、转岗、或离职时，数据库就会相应更新。HR 系统将需要访问企业的用户的所有相关信息都提供给 BeyondCorp。

单点登录系统

外化的单点登录（SSO）系统是一个集中的用户身份认证门户，它对请求访问企业资源的用户进行双因子认证。使用用户数据库和群组数据库对用户进行合法性验证后，SSO 系统会生成短时令牌（short-lived tokens），用来作为对特定资源授权流程的一部分。

消除基于网络的信任

部署无特权网络

为了不再区分内部和远程网络访问，BeyondCorp 定义并部署了一个与外网非常相似的无特权网络，虽然其仍然处于一个内网的地址空间。无特权网络只能连接互联网、有限的基础设施服务（如，DNS、DHCP 和 NTP）、以及诸如 Puppet 之类的配置管理系统。谷歌办公大楼内部的所有客户端设备默认都分配到这个网

络中，这个无特权网络和谷歌网络的其他部分之间由严格管理的 ACL（访问控制列表）进行控制。

有线和无线网络接入的 802.1x 认证

对于有线和无线接入，谷歌使用基于 802.1x 认证的 RADIUS 服务器将设备分配到一个适当的网络，实现动态的、而不是静态的 VLAN 分配。这种方法意味着不再依赖交换机/端口的静态配置，而是使用 RADIUS 服务器来通知交换机，将认证后的设备分配到对应的的 VLAN。受控设备使用设备证书完成 802.1x 握手，并分配到无特权网络，无法识别的设备和非受控设备将被分配到补救网络或访客网络中。

将应用和工作流外化

面向公共互联网的访问代理

谷歌的所有企业应用都通过一个面向公共互联网的访问代理开放给外部和内部客户。通过访问代理，客户端和应用之间的流量被强制加密。一经配置，访问代理对所有应用都进行保护，并提供大量通用特性，如全局可达性、负载平衡、访问控制检查、应用健康检查和拒绝服务防护。在访问控制检查（详述见后文）完成之后，访问代理会将请求转发给后端应用。

公共的 DNS 记录

谷歌的所有企业应用均对外提供服务，并且在公共 DNS 中注册，使用 CNAME 将企业应用指向面向公共互联网的访问代理。

实现基于设备清单的访问控制

对设备和用户的信任推断

每个用户和/或设备的访问级别可能随时改变。通过查询多个数据源，能够动态推断出分配给设备或用户的信任等级，这一信任等级是后续访问控制引擎（详述见后文）进行授权判定的关键参考信息。

例如，一个未安装操作系统最新补丁的设备，其信任等级可能会被降低；某一类特定设备，比如特定型号的手机或者平板电脑，可能会被分配特定的信任等

级；一个从新位置访问应用的用户可能会被分配与以往不同的信任等级。信任等级可以通过静态规则和启发式方法来综合确定。

访问控制引擎

访问代理中的访问控制引擎，基于每个访问请求，为企业应用提供服务级的细粒度授权。

授权判定基于用户、用户所属的群组、设备证书以及设备清单数据库中的设备属性进行综合计算。如果有必要，访问控制引擎也可以执行基于位置的访问控制。另外，授权判定也往往参考用户和设备的信任等级，例如，可以限制只有全职工程师、且使用工程设备才可以登录谷歌的缺陷跟踪系统；限制只有财务部门的全职和兼职员工使用受控的非工程设备才可以访问财务系统。访问控制引擎还可以为应用的不同功能指定不同的访问权限和策略，例如，在缺陷跟踪系统中，与更新和搜索功能相比，查看某一条记录可能不需要那么严格的访问控制策略。

访问控制引擎的消息管道

通过消息管道向访问控制引擎源源不断地推送信息，这个管道动态地提取对访问控制决策有用的信息，包括证书白名单、设备和用户的信任等级，以及设备和用户清单库的详细信息。

一个端到端示例

应用

本例中，我们假设一个应用将被“BeyondCorp 化”，这个应用用于工程师审核、注释、更新源代码，并且经审核者批准后可以提交代码。进一步假设权限设定为：允许全职和兼职工程师从任何受控设备上对这一应用(codereview.corp.google.com)进行访问。

配置面向互联网的访问代理

codereview.corp.google.com 的所有者为这一服务配置访问代理。配置指定了应用后端的网络位置和每个后端可承受的最大流量。codereview.corp.google.com 的域名在公共 DNS 中注册，其 CNAME 指向访问代理。例如：

```
$ dig @8.8.8.8 codereview.corp.google.com

; <<>> DiG 9.8.1-P1 <<>> @8.8.8.8 codereview.corp.google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12976
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0,
ADDITIONAL: 0

;; QUESTION SECTION:
;codereview.corp.google.com. IN  A

;; ANSWER SECTION:
codereview.corp.google.com. 21599 IN  CNAME
accessproxy.l.google.com.
accessproxy.l.google.com.    299    IN  A      74.125.136.129

        ;; Query time: 10 msec
        ;; SERVER: 8.8.8.8#53(8.8.8.8)
        ;; WHEN: Wed Aug 20 19:30:06 2014
        ;; MSG SIZE rcvd: 86
```

配置访问控制引擎

访问控制引擎提供了一个默认规则，限制只有全职员工使用受控设备才能进行访问。codereview.corp.google.com 的所有者可以提供一个更具化的规则进一步限制针对此应用的访问，包括：限制只有最高信任等级的受控设备可以访问、限制只有最高信任等级的全职和兼职工程师可以访问。

当一位工程师访问网络

如果网络位于企业办公大楼外：工程师使用谷歌配发的笔记本电脑，接入任何 Wi-Fi 网络，这个网络可能是一个有登录验证门户的机场 Wi-Fi，也可能是咖啡馆的公共 Wi-Fi。不再需要配置和通过 VPN 来连接到企业网络。

如果网络位于企业办公大楼内：工程师使用谷歌配发的笔记本电脑访问企业网络，这台电脑在与 RADIUS 服务器进行 802.1x 握手过程中提供设备证书。当证书有效时，为这台电脑在无特权网络上分配一个地址；如果电脑不是由公司配发的或者其设备证书过期了，就为这台电脑分配一个补救网络上的地址，而且这个地址的访问权限非常有限。

访问应用，无需区分网络

工程师从公司配发的笔记本电脑上访问 codereview.corp.google.com，读者可以参考图 1 的访问流程。

1、请求指向访问代理，笔记本电脑提供设备证书。

2、访问代理无法识别用户，重定向到单点登录系统。

3、工程师提供双因子认证凭据，由单点登录系统进行身份认证，颁发令牌，并重定向回访问代理。

4、访问代理现在持有标识设备的设备证书，标识用户的单点登录令牌。

5、访问控制引擎为 codereview.corp.google.com 执行对应的授权检查。授权检查基于每个请求进行：

a) 确认用户是工程组成员。

b) 确认用户拥有足够的信任等级。

c) 确认设备是一个良好的受控设备。

d) 确认设备拥有足够的信任等级。

e) 如果所有这些检查通过，请求被转发到应用后端获取服务。

f) 如果上述任何检查失败，请求被拒绝。

基于上述方法和流程实现了丰富的、服务级的认证及针对每个请求的授权检查。

迁移到 BeyondCorp

与世界上几乎所有企业一样，多年来谷歌一直为其用户和应用维护一个特权网络（内网），这种模式使得基础设施对公司的日常运作至关重要。尽管公司的所有组件都应该迁移到 BeyondCorp，但一下子将每个网络用户和每个应用都迁移到 BeyondCorp 环境，对业务连续性来说非常危险。因此，谷歌投入大量资源进行分阶段迁移，在不影响公司生产力的情况下，成功地将大批网络用户逐步迁移到 BeyondCorp，如图 2 所示。下面将详细介绍我们所做的一些工作。

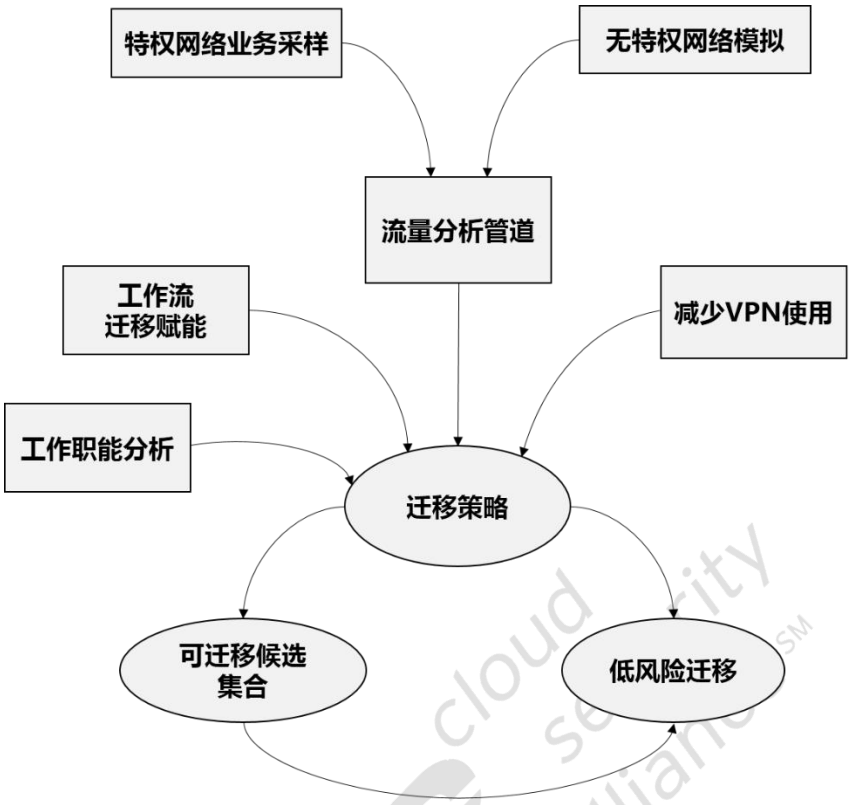

图 2 迁移到 BeyondCorp

工作流迁移评估

需要确保谷歌的所有应用都可以迁移以便最终通过访问代理进行访问。BeyondCorp 会发起对所有应用的检查和迁移评估，确保其平滑迁移。每个应用要实现迁移的难易不一，有的只需要简单的支持 HTTPS 流量，有的比较复杂，如，需要实现单点登录集成。需要对每个应用配置访问代理，在大多数情况下还需要在访问控制引擎中进行特定的策略配置。每个应用都会经历以下阶段实现迁移：

1、可以通过特权网络直接访问；以及在外网通过 VPN 访问。

2、可以通过特权网络直接访问；以及在外网及无特权网络中通过访问代理访问。此阶段需要使用 DNS 分离解析，内部域名服务器直接指向应用，外部域名服务器指向访问代理。

3、在外部、特权和无特权网络中通过访问代理均可访问。

工作职能分析

通过检查整个公司的工作职能，并和工作流迁移评估进行交叉对比，我们能够确定用户群组迁移的优先级。基于当时对用户工作流和 BeyondCorp 组件功能的全面理解，我们从财务、销售、法务或工程师团队中选择网络用户进行迁移。

减少 VPN 的使用

随着越来越多的应用通过访问代理访问，我们开始阻止用户使用 VPN，策略如下：

1、只有经证实确有需要的用户才能使用 VPN 访问。

2、监控 VPN 的使用，删除在一段时期内未使用 VPN 的用户的访问权限。

3、监控 VPN 活跃用户的 VPN 使用情况，如果他们所有的工作流都可以通过访问代理实现，将强烈建议用户放弃使用 VPN。

流量分析管道

只有当我们确信（或者非常接近确信）所有用户的工作流都可以从无特权网络中访问时，才能将用户转移到无特权网络，这一点对迁移的平滑性非常重要。为了建立一个相对的确定性，我们建立了一个流量分析管道。从公司的每个交换机中采样网络流量数据并输入管道，将这些数据与无特权网络和公司其余网络之间的预设访问控制列表对比分析，通过分析，能够识别命中访问控制列表的流量，以及没有命中访问控制列表的流量，并分别形成列表。对于没有命中访问控制列表的“逃逸”流量被关联到特定的工作流和/或特定的用户和/或特定的设备上，并进一步解决这些“逃逸”流量的问题，使其在 BeyondCorp 环境中能够工作。

无特权网络模拟

作为补充，除了通过交换机采样流量并进行流量分析外，我们还通过安装在访问谷歌网络所有用户设备上的流量监视器，对整个公司的无特权网络行为进行模拟。流量监视器检查了每个设备的所有流入和流出的流量，与无特权网络和公司网络其余部分之间的预设访问控制列表对比验证，并记录没有通过验证的非法流量。流量监视器有两个模式：

- 记录模式：捕获非法流量，但仍然允许上述流量流出设备。
- 强制模式：捕获并丢弃非法流量。

迁移策略

通过流量分析管道和无特权网络模拟，可以定义并实施分阶段的迁移策略，包括以下内容：

1、通过工作职能和/或工作流和/或位置来确认潜在的可迁移候选集。

2、模拟器开启记录模式，确认在连续 30 天内合格流量比例大于 99.9%的用户和设备。

3、如果在该时期内，用户和设备的合格流量比例大于 99.99%，则为用户和设备启动强制模式。当然，若有必要，用户可以将模拟器恢复到记录模式。

4、在强制模式下成功运行 30 天之后，将此状态记录在设备清单中。

5、包含在候选集中，且在模拟器执行模式下成功工作 30 天是一个非常强烈的合格信号，下一次 Radius 服务器提供 802.1x 身份验证服务时，设备将被分配到无特权网络。

豁免处理

除了尽可能自动化地将用户和设备从特权网络转移到新的无特权网络外，我们还采用了一个简单的办法允许用户请求临时免除这种迁移。我们维护了一个未获得 BeyondCorp 能力评估、尚未达到迁移标准的工作流列表。用户可以搜索这些工作流，在经过适当的审批后，将自己和设备标记为特定工作流的活跃用户。当工作流完成 BeyondCorp 赋能，达到迁移标准后，相关用户会收到通知，再次进入迁移候选名单并进行迁移。

完成 BeyondCorp

谷歌的 BeyondCorp 迁移正在进行中，所需的大部分工作流已经评估确认完毕。我们的迁移工具和策略允许主动将用户、设备和工作流迁移到 BeyondCorp，而不会影响日常工作生产力。

我们预计 BeyondCorp 迁移的收尾工作还很多，需要花费一段时间。例如，使用专有协议与服务器交互的胖客户端应用将是一个挑战。我们正在研究将此类应用迁移到 BeyondCorp 的方法，也许会为它们配套使用一种特殊的认证服务。

随着我们向 BeyondCorp 迁移工作的推进，我们打算后续发表一系列文章解释谷歌为何以及如何向 BeyondCorp 迁移，同时也希望可以鼓励其他企业实施类似的实践。

【第二篇】谷歌 BeyondCorp：从设计到部署

谷歌 BeyondCorp 项目的目标是为了提高员工和设备访问企业内部应用的安全性。与传统的边界安全模型不同，BeyondCorp 不基于物理位置或发起请求的网络位置来授予用户访问服务和应用的权限；相反，访问策略的制定完全基于设备的信息、状态和当前设备的使用者信息等等。BeyondCorp 模型默认内部网络和外部网络都是不可信的，需要动态评估当前访问请求的安全等级，并确保此等级满足应用的最低安全要求。

本文将介绍谷歌如何从传统的安全基础设施迁移到 BeyondCorp 模式，以及在迁移过程中所面对的挑战和获得的经验教训。关于 BeyondCorp 的架构讨论，请参见第一篇“BeyondCorp：一种新的企业安全方案”[1]。

概述

BeyondCorp 系统的关键组件包括信任引擎（Trust Inferer）、设备清单服务(Device Inventory Service)、访问控制引擎(Access Control Engine)、访问策略(Access Policy)、网关（Gateways）和资源(Resources)，如图 1 所示，BeyondCorp 所使用的各术语定义如下：

- 访问需求被划分为不同的**信任等级（Trust Tiers）**，不同的等级代表着不同的敏感度，等级越高，敏感度越高。
- **资源(Resources)**代表所有访问控制机制将覆盖的应用、服务和基础设施。包括在线知识库、财务数据库、链路层访问、实验室网络等等，需要为每个资源都分配一个访问所需的最小信任等级。
- **信任引擎（Trust Inferer）**是一个持续分析和标注设备状态的系统。该系统可设置设备可访问资源的最大信任等级，并为设备分配对应的 VLAN，这些数据都会记录在设备清单服务中。任何设备状态的更新，或者信任引擎无法接收到设备的状态更新消息，都会触发对其信任等级的重新评估。
- **访问策略(Access Policy)**是描述授权判定必须满足的一系列规则，包含对资源、信任等级和其他影响授权判定的因子的程序式表示。

- **访问控制引擎(Access Control Engine)**是一种集中式策略判定点，它为每个访问网关提供授权决策服务。授权过程一般基于访问策略、信任引擎的输出结果、请求的目标资源和实时的身份凭证信息，并返回成功/失败的二元判定结果。
- **设备清单服务(Device Inventory Service)**是 BeyondCorp 系统的中心，它不断收集、处理和发布所有在列设备状态的变更。
- **网关（Gateways）**是访问资源的唯一通道，如 SSH 服务器、Web 代理或支持 802.1x 认证的网络等。网关负责对授权决策进行强制执行，例如强制最低信任等级或分配 VLAN。

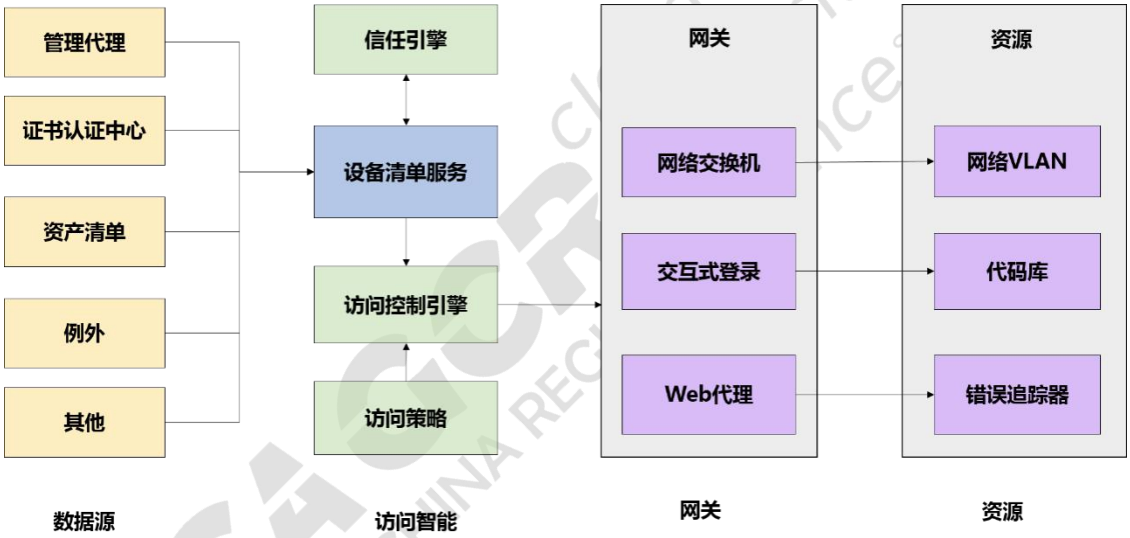

图 1: BeyondCorp 系统的关键组件

BeyondCorp 的组件

通过使用下列组件，BeyondCorp 将各类已有系统组件、新系统组件进行集成，以便实现灵活而细粒度的信任判定。

设备（Device）和主机(Host)

要实现基于清单的访问控制，基本前提就是要建立清单库。基于环境和安全策略，团队需要先针对设备和主机的定义达成一致。**设备（device）**是物理或虚拟“计算机”，而**主机（host）**是指某特定时间点上设备状态的快照。例如，**设备**可能是一台笔记本电脑或一部手机，而**主机**则是运行在该设备上的操作系统和

软件的详细信息。设备清单服务包含设备信息、运行于设备上的主机信息、以及对二者的信任评估。在下面的章节中，根据不同的访问策略配置，“设备”既可能指代物理设备也可能指代主机。建立基础清单库后，其余组件就可以按需部署，以提供安全性更高、覆盖率更广、颗粒度更细、延迟性更低、灵活性更佳的基于清单的访问控制服务。

基于信任等级的访问

信任度可以划分为若干信任等级，并由信任引擎为每个设备分配信任等级，另外，需要为每个资源都事先分配一个访问所需的最低信任等级，简称访问信任等级。设备被分配的信任等级必须大于等于资源的访问信任等级才可访问该资源。简单举一个婚庆餐饮公司的例子：送货员只需要查询婚礼的地址，这种访问请求所需的信任等级较低，事实上，他们也并不需要访问更敏感的服务，比如账单系统，这类系统一般会分配一个较高的访问信任等级。

分配访问信任等级有几个优点：降低了高安全要求的设备相关的运维成本（主要是与技术支持和生产力相关的成本），同时也提高了设备的可用性。如果允许设备访问更多高敏感数据，则需要更频繁地检测以确保设备使用者确实“在场”，因此越是信任某个设备，其身份凭证有效期应该越短。因此，按照潜在访问需求所需的最低信任等级来要求/限定设备所需的信任等级，就意味着设备使用者在访问过程中受到干扰的程度会降到最小。比如，为了维持较高的信任等级，就需要设备在几个工作日内必须完成操作系统最新升级包的安装；而对于信任等级需求较低的设备，安装升级包的时间窗口就可以稍微宽松些。

再举一个例子，一台由公司集中管理的笔记本电脑，由于在一段时间内没有连接到网络，因此没有更新到最新状态。如果操作系统缺少几个*非关键补丁*，其信任等级可能被降为中等，仅允许访问部分业务应用，而被拒绝访问需要更高信任等级的业务应用。但如果它缺少*关键补丁*，或者防病毒软件报告该设备已感染病毒，那就只允许它连接补救服务。在最极端的情况下，一台明确的遗失或被盗设备会被拒绝访问所有企业资源。

除了分配信任等级，信任引擎还通过标注设备可访问的 VLAN 来进行网络分段。网络分段允许我们基于设备状态来限制对诸如实验室和测试环境的特定网络的访问权限。当一个设备变得不可信时，可以将它分配到隔离网络，在设备恢复信任之前仅提供有限的资源访问权限。

设备清单服务

设备清单服务（如图 2 所示）是一个不断更新的数据管道，能够从广泛的数据来源中导入数据。系统管理数据源可能包括活动目录（Active Directory）、Puppet 和 Simian，其他设备代理、配置管理系统和企业资产管理系统也会向该管道导入数据。外部（Out-of-band）数据源包括漏洞扫描系统、证书颁发机构和诸如 ARP 映射表等网络基础设施单元。每个数据源都可以发送设备相关的完整数据或增量数据。

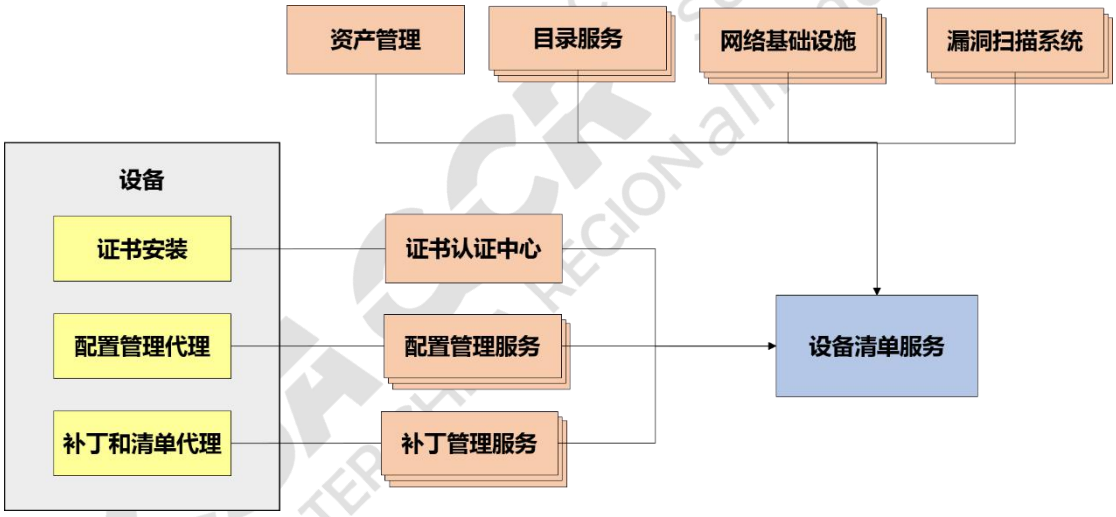

图 2　设备清单服务

BeyondCorp 设备清单服务自实施初期，已经从超过 15 个数据源中吸收了数十亿的增量数据，速度约 300 万条/天，总量超过 80TB。保留历史数据非常重要，因为这样才能更好地了解特定设备的端到端生命周期、跟踪和分析总体趋势、执行安全审计和调查取证。

数据类型

数据有两种主要的类型：观察数据和预设数据。

观察数据由程序产生，包括以下项目：

- 最近一次在设备上执行安全扫描的时间,及扫描结果
- 活动目录的最后同步策略和时间戳
- 操作系统版本和补丁等级
- 已安装的软件

预设数据通过 IT 运维手动维护，包括以下内容：

- 为设备分配的所有者
- 允许访问该设备的用户和组
- 分配的 DNS 和 DHCP
- 对特定 VLAN 的显式访问权限

在数据不足或客户端平台不可定制的情况下，就需要明确分配（比如打印机就属于这种情况）。与观察数据所表现的易变性相比，预设数据通常是静态的。通常需要分析许多来自不同来源的数据，用以识别潜在的数据冲突，而不要盲目地相信单个或少量系统的数据真实性。

数据处理

数据格式的转换与统一

为了使设备清单服务保持最新状态，涉及几个处理阶段。首先，所有数据必须转换成一种通用数据格式。一些数据源，比如内部自研系统或开源解决方案，可以通过系统改造，在提交数据时主动发布给清单服务。而其他来源，特别是那些第三方数据源，可能无法扩展或改造，难以支持主动的变更发布，这种情况需要通过定期轮询来获得更新。

数据关联

当输入数据格式统一后，就进入数据关联阶段。在这个阶段，所有来自不同数据源的数据都被聚合、关联到某一设备，当确定两条记录描述的是同一设备时，就将它们合并为单条记录。数据关联过程看似简单，但在工程实践中却相当复杂，因为许多数据源之间并不具备数据关联所必须的重叠的“标识符”。

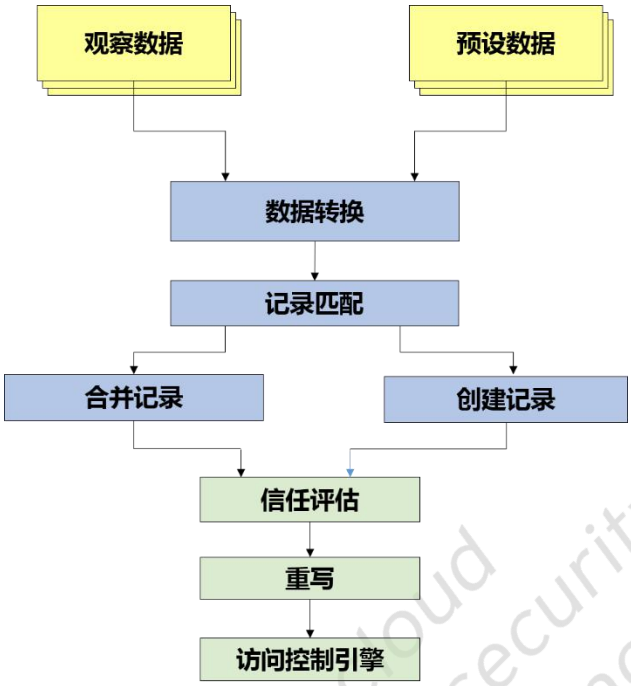

图 3 数据处理管道

例如，资产管理系统可能存储资产 ID 和设备序列号，而磁盘加密托管系统存储硬盘序列号，证书颁发机构存储证书指纹，ARP 数据库存储 MAC 地址。这些数据不具备一个重叠的“标识符”，难以确定来自这些独立系统中的增量是否描述的是同一个设备，只有在清单报告代理同时报告几个或全部这些标识信息之后，这些多源的、没有交集的记录才可能合并为单条记录。

如果再考虑到设备的全生命周期，相关的信息及其关联过程将更加一团糟，因为硬盘、网卡、机箱和主板都有可能被替换，甚至会在设备之间交换。另外，如果还考虑人为的数据录入错误，情况会更加复杂。

信任评估

一组输入记录一旦完成关联合并，就会触发引擎进行重新评估。为了分配信任等级，评估分析过程引用多种字段并聚合产生最终结果。信任引擎目前从不同的数据源引用了数十个字段，包括针对特定平台的和平台无关的；随着系统的不断演化，还有数百万个额外字段可供分析。例如，为了获得较高信任等级，可能需要一个设备满足以下所有（或更多）需求：

- 加密
- 成功执行所有的管理和配置客户端程序（agents）
- 安装最新的操作系统安全补丁
- 从所有输入源中获得的数据状态一致

这种信任等级预计算减少了必须被推送到网关的数据量，以及在为访问请求做授权判定时所需的计算量。这一步也确保所有的执行网关都使用了一致的数据集合。在这个阶段，我们甚至可以对非活跃设备修改信任等级。比如在以前，我们会预先拒绝所有可能受到 Stagefright[2]漏洞影响的设备的访问权限，即便它们还没发起实质性的访问请求。预计算同样为我们提供了一个实验框架，在此框架中可以对变更进行预验证，以及在不影响整个公司的情况下，对策略或信任引擎进行小幅度的局部调整和验证。

当然，预计算也有它的局限性，还不能完全依赖它。比如，访问策略可能要求进行实时的双因子认证，或者限制来自某已知恶意网段的访问请求。策略或设备状态变更与网关真正执行这个变更之间的延迟并不是什么大问题，因为更新延迟通常在 1 秒以内。事实上更本质的问题是，并不是所有的信息在预计算阶段都能够获取。

特殊处理

信任引擎对于设备信任等级的分配有最终决定权。信任评估还需要考虑设备清单服务中已存在的特殊处理。通过特殊处理，允许对通用访问策略进行覆盖和重写。特殊处理主要为了降低策略变更或新策略的生效延迟。比如，由于安全扫描设备可能尚未升级，检测不出某种零日攻击，但可以通过特殊处理立即阻止某台可能遭受零日攻击的设备；同样，可以采用特殊处理，允许将某台不可信设备连接到实验室网络。物联网设备安装和维护设备证书可能并不可行，同样可以通过特殊处理，直接为其分配适合的信任等级以确保正常访问。

部署

首次上线

BeyondCorp 上线的第一阶段包含一部分网关和初步的元清单服务，这些服务仅由少数几个数据源构成，主要是一些预设数据。最初实现的访问策略模拟了谷歌已有的基于 IP 的边界安全模型，并将这个策略集应用到不可信设备上，为来自特权网络的设备保留不变的访问权限。这种策略能够确保在系统完善之前，能安全地部署系统的一些组件，而不会影响用户的平滑使用。

与此同时，BeyondCorp 团队也在设计、开发并持续迭代一个规模更大、延迟更低的元清单解决方案。这个设备清单服务从超过 15 个数据源收集数据，根据正在主动生成数据的设备数量，每秒钟可能有 30 至 100 个不等的数据变更。设备清单服务主要提供的是企业设备的信任资格标注和强制授权。随着元清单解决方案的成熟，可以获得更多的设备信息，能够逐步地依靠信任等级分配，逐步替代基于 IP 的策略。在验证了低信任等级设备的工作流后，对更高信任等级的访问进行细粒度限制，并逐步迈向最终目标：随着时间的推移，有序扩大设备和企业资源的信任等级分配范围，并基于信任等级进行访问控制。

考虑到前文提到的从不同来源关联数据的复杂性，BeyondCorp 采用 x.509 证书作为固定的设备标识符。x.509 证书提供了两个核心功能：

- 如果证书发生变化，即使所有其他标识符都保持相同，设备也被标记为不同设备。
- 如果证书安装在不同的设备上，关联逻辑会发现证书冲突以及与辅助标识不匹配，随即做出反馈，降低设备信任等级。

证书并未降低数据关联的必要性，其本身也不足以获得访问权限。但它确实能提供一个基于密码学的 GUID，访问网关还可将其用于流量加密，并持续、唯一地标识设备。

移动设备

谷歌一直力图使移动设备成为主流平台，移动设备必须能够完成与其他平台相同的任务，因此也需要相同的访问等级。与其他平台相比，在移动平台上部署

信任等级访问模型更容易。移动设备的特点是没有太多传统遗留通信协议和访问方法，因为几乎所有通信都是基于 HTTP 的。安卓设备使用加密的安全通信，允许在设备清单中识别设备。值得一提的是，由于 API 也位于与访问控制引擎集成的访问代理之后，因此原生应用程序与通过 Web 浏览器访问的资源都能通过相同的授权机制进行保护。

遗留（Legacy）平台和第三方平台

为了支持遗留平台和第三方平台，我们需要采用比移动设备更广泛的访问方法。为此任意 TCP 和 UDP 流量，我们通过 SSH 隧道和客户端 SSL/TLS 代理技术提供的隧道通信。 而网关只允许符合访问控制引擎中策略的隧道业务通过。RADIUS[3]是一个特例：它与设备清单服务集成，但它从信任引擎接收的是 VLAN 的分配结果，而不是信任等级的分配。在网络连接时，RADIUS 使用 802.1x 认证的证书来作为设备标识符，通过信任引擎分配的结果，动态设置 VLAN。

避免干扰用户

在部署 BeyondCorp 的过程中，面临的最大挑战之一是如何在不干扰用户的情况下完成如此大规模的任务。为了制定策略，需要先确认现有的工作流。从现有的工作流中，可以确定：

- 哪些工作流，可以与无特权网络兼容
- 哪些工作流允许进行超出预设的访问或哪些工作流允许用户绕过已经存在的限制

为了确认工作流，我们采用双管齐下的模式。一方面，开发了一个模拟管道，它可以检查 IP 级元数据，将流量划分到服务，并在模拟环境中应用了我们预期的网络安全策略；另一方面，将安全策略转换为每个平台本地防火墙配置语言。在企业网络上，这种手段可以很好的记录流量元数据，这些流量是访问谷歌企业服务所必须的，稍有差池，迁移到无特权网络后，这些服务很可能无法访问。在此过程中，我们还有一些令人惊讶的意外发现，比如那些早就应该下线的服务，却不明就里的仍在运行。

收集了这些数据之后，通过与服务所有者合作，将他们的服务迁移到支持 BeyondCorp 的网关。有些服务很容易迁移，但还有些服务则比较困难，需要一

些特殊处理机制。不过，这种情况都明确指定了责任人，确保服务所有者能在限定期限内消除例外。随着越来越多的服务进行了更新和改造，越来越多的用户在不执行任何例外处理的情况下也可以正常工作很长一段时间，此时，就可以将用户的设备分配到一个无特权的 VLAN。通过这种方法进行过渡，用户使用不兼容 BeyondCorp 的应用不会感到不太方便；迁移压力基本都在服务提供者和应用程序开发人员身上，这可以促使他们正确地配置相关服务。

特殊处理增加了 BeyondCorp 生态系统的复杂性，随着时间的推移，“为什么我的访问被拒绝了？”这个问题的答案已经不那么明了。基于清单数据和实时请求数据，需要非常明确地判断特定请求在特定时间点失败或成功的原因。回答上述问题的第一步是与终端用户建立沟通（警告其潜在的问题，以及如何进行自我修复或联系支持），并培训 IT 运维人员。此外，还开发了一种服务，它可以分析信任引擎的决策树和影响设备信任等级分配的事件的时间顺序，从而提出补救措施。有些问题用户可以自己解决，不需要权限更高的支持人员。拥有额外访问路径的用户通常能够自我修复，例如，如果用户认为他的笔记本电脑信任评估不当,但手里还有一只信任等级足够的手机，我们可以将诊断请求转发给这个手机进行评估。

挑战和经验教训

数据质量及相关性

资产管理的数据质量问题可能导致设备无意中失去对企业资源的访问权限。拼写错误、标识错误和信息丢失都是常见问题。此类问题可能由于采购团队收到资产并将其添加至系统时的人为失误，也可能是由于制造商工作流程的失误导致。数据质量问题也经常发生在设备维修过程中，主要原因在于替换设备的零部件或在设备之间交换某个部件。这些问题可能会破坏设备记录，除非人工检查这些设备，否则很难修复这些记录上的差错。例如，单条设备记录可能实际上包括两个不同设备的数据，要自动修复和分离数据甚至需要调整设备硬件的资产标签甚至主板序列号。

这时最有效的解决方案是通过本地工作流程改进并增加自动输入验证，以便在输入时发现并减少人为错误。复式记账法有一定帮助但是并不能发现所有错误。做出准确的信任评估需要设备清单库提供高精度的数据，所以这又迫使人们不得不重新关注设备清单库中的数据质量。这种数据的精确性要求是前所未见的，也带来了前所未有的价值。比如，我们能精确地知道终端信息，安装最新补丁的情况，进而提高整个系统安装最新补丁的百分比。

稀疏数据集

如前所述，上游数据源未必有重叠的设备标识符。以下列举一些潜在的场景：新设备可能有资产标签，但没有主机名；在设备生命周期的不同阶段，硬盘序列号可能与不同的主板序列号相关联，又或者 MAC 地址可能会发生冲突。一组简单的启发式算法可以将大部分增量与数据源某个子集相关联，但为了将精度提高到接近 100%，需要一组非常复杂的启发式算法来处理看似无穷无尽的边缘情况。一小部分数据不匹配的设备，可能会使数百甚至数千名员工无法使用他们工作中的必需应用。为了减少这种情况的发生，监控并验证各种综合数据可能的情况，精细设计和验证信任评估路径，最终确保符合预期的信任等级评估结果。

管道延迟

由于设备清单服务从几个不同的数据源中获取数据，所以每个源可能都需要一个特定的实施方案。自研系统或基于开源系统的数据源很容易扩展，以便异步地向我们现有管道发布增量。对于其他来数据源必须定期轮询，这需要在轮询频率和由此产生的服务器负载之间取得平衡。尽管将变更信息传递到网关通常不到一秒，但是对于轮询的场景，一些变更可能需要几分钟才能获悉。此外，串行处理本身也会增加时延。因此，需要采用流式处理。

沟通

对安全基础设施的根本性改变可能会对整个公司的生产力产生负面影响。与用户沟通改变的潜在影响、会出现的问题和可能的补救措施十分重要，但是很难找到过度沟通和沟通不足之间的平衡点。沟通不足会让用户感到惊讶和困惑，造成补救措施效果差，IT 支持人员的工作也会超负荷。过度沟通也有问题：不愿改变的用户会倾向于高估变化带来的影响并企图寻求不必要的豁免。过于频繁的

沟通也会让用户对潜在的影响判断出现偏差，由于谷歌的企业基础设施在许多互不关联的方面同时开展工作，用户很容易将访问相关的问题与其他正在进行的项目问题混淆，这也会降低补救措施的效率，增加支持人员的操作负荷。

灾难恢复

正因 BeyondCorp 基础设施的组成是非常复杂的，而灾难性的失败甚至会导致支持人员无法访问恢复所需的工具和系统，因此 BeyondCorp 系统中构建了各种故障保护系统。除了监测信任等级分配的潜在或明显的变化，我们已经利用了现有的一些灾难恢复实践，以确保在发生灾难性紧急情况时，BeyondCorp 仍能发挥作用。BeyondCorp 的灾难恢复协议基于最小依赖关系，并允许极少的一部分特权维护人员重放清单变更的日志记录，以便恢复到设备清单和信任评估工作以前的良好状态。我们也有能力在紧急情况下细粒度地变更访问策略以确保维护人员启动恢复流程。

下一步

与任何大项目一样，我们在部署 BeyondCorp 时面临的挑战，有些是预期内的，而有些在意料之外。在谷歌，越来越多的团队正在寻找新的、有趣的方式来整合系统，并提供更详细、更有层次的防护以对抗恶意攻击者。在没有牺牲易用性的前提下，BeyondCorp 已经大幅改善了谷歌的安全态势，同时还提供了一个灵活的基础设施，能够基于策略进行授权决策而不受具体技术限制。BeyondCorp 在谷歌自身的系统和规模内已取得了相当大的成功，也欢迎其他组织基于这些原则和流程进行部署和完善。

参考文献：

[1] Architectural discussion of BeyondCorp: http://research .google.com/pubs/pub 43231.html.
[2] Stagefright: https://en.wikipedia.org/wiki/Stagefright _(bug).
[3] RADIUS: https://en.wikipedia.org/wiki/RADIUS.

【第三篇】BeyondCorp：访问代理

本文将详细介绍 BeyondCorp 前端基础设施——访问代理（Access Proxy, AP）的实现，关注其实施过程中遇到的挑战,以及设计和上线中学到的经验教训。此外，对于我们正在开展的，旨在提高员工访问内部应用时使用体验的项目，本文也有所涉及。

向 BeyondCorp 模型迁移过程中（之前在“BeyondCorp：一种新的企业安全方案”[1]和“谷歌 BeyondCorp：从设计到部署”[2]中有讨论），有许多难题需要解决，比如，如何将公司策略应用到所有内部服务就是一个重大挑战。传统方法通常要把每个业务后端与设备信任引擎集成，以便进行应用级策略的评估，然而，这种方法会明显降低产品发布和迭代的速度。

为了解决这个问题，谷歌通过前端访问代理（AP）作为中心化的策略强制执行点，实现粗粒度的公司安全策略。访问代理的设计具有足够的通用性，基于同一套代码我们实现了不同逻辑的网关。目前， 访问代理已支持 Web 代理和 SSH 网关组件[2]。由于 AP 是员工访问内部 HTTP 服务的唯一机制，所有内部服务都需要迁移到 AP 的后面。

事实证明，我们一开始只打算支持 HTTP 协议是完全不够的，随着项目的推进，不得不为更多的协议（其中多数都需要端到端加密,如 SSH）提供解决方案。支持这些协议通常需要对客户端进行改造，以确保 AP 准确识别设备。

结合访问代理（AP）和集中的访问控制引擎（Access Control Engine, ACE）（共享的 ACL 评估系统）主要有两个好处：一是所有请求都途经同一个日志记录点，便于更有效地进行流量分析；二是能够更迅速、更统一地改变执行策略。

BeyondCorp 的前端基础设施

任何大规模部署的现代 Web 应用程序都会采用前端基础设施——通常是负载均衡和/或 HTTP 反向代理的组合，企业 Web 应用也不例外，前端基础设施为

策略执行点的部署提供了理想位置。因此，谷歌的前端基础设施对于 BeyondCorp 访问策略的强制执行至关重要。

谷歌前端基础设施的主要组件是 HTTP/HTTPS 反向代理集群，即谷歌前端服务（Google Front Ends, GFEs）[3]。GFE 有很多优点，例如负载均衡和 TLS 卸载服务。这样 Web 应用的后端可以专注于服务请求的具体内容，而几乎不必考虑请求的路由细节。BeyondCorp 将 GFE 作为访问策略强制执行的逻辑中心。通过逻辑上的集中，带来请求的汇集，在此基础上就可以自然而然地扩展 GFE 的功能，比如自助服务开通、认证、授权和集中式日志记录。扩展后的 GFE 即访问代理（AP）。下文将详细阐述访问代理提供的具体服务。

扩展后的 GFE 特性：产品需求

GFE 有一些内置功能，并不是专门为 BeyondCorp 设计的但可以为 BeyondCorp 所用：如，为后端提供负载均衡服务、通过 GFE 实现 TLS 卸载。AP 通过引入**认证**和**授权**策略扩展了 GFE。

认证

为了正确处理一个授权请求，AP 需要识别发出请求的用户和设备。在多平台环境中，设备认证面临许多挑战，将在后文的“多平台身份认证的挑战”中进行详细讨论，本节重点介绍用户认证。

AP 通过集成谷歌的身份提供服务（Identity Provider, IdP）完成用户身份认证。如果要求后端服务必须修改它们自身的身份认证机制才能迁移到 AP,不具备伸缩性，所以 AP 需要支持一系列的认证机制,包括：OpenID Connect、OAuth 和一些定制化协议。

AP 还需要处理不能提供用户凭证的请求场景，例如，一个软件管理系统试图下载最新的安全补丁，这种情况下，AP 可以禁用用户认证。

当AP认证用户后，将用户凭证相关信息从请求中去除后再转发至后端服务，这样做至关重要，有两点原因：

- 确保后端不能通过访问代理重放请求（或凭证），进行重放攻击。
- 代理对后端服务透明。这样做的好处在于后端业务可以独立于访问代理的数据流叠加自身的认证逻辑，并且也避免了将 cookie 和用户凭证不必要的暴露给后端业务。

授权

以下两个设计推动 BeyondCorp 中授权机制的实施：

- 一个可通过远程过程调用（Remote Procedure Calls, RPC）查询的集中访问控制列表（Access Control List, ACL）
- 采用领域特定语言（domain-specific language, DSL）表达访问控制列表（ACL），使其同时兼顾可读性和可扩展性

以服务形式提供 ACL 评估能够保证多种前端网关的一致性（如 RADIUS 网络访问控制基础设施、AP 和 SSH 代理）。

集中式授权有好有坏。好处是，通过集中策略执行点，由前端访问代理负责授权可以将后端开发者从处理授权的细枝末节中解放出来，并且一致性更强。坏处是，代理可能无法执行细粒度策略，细粒度授权还是要交由后端处理更好（例如，“用户 A 被授权去修改资源 B”）。

从过去的实践经验来说，将 AP 提供的粗粒度、集中式授权与后端实现的细粒度授权结合对于前后端来说都是最佳选择。这种方法不会导致重复工作，因为针对特定应用的细粒度策略通常与前端基础设施所执行的企业级策略相互独立。

代理和后端之间的双向身份认证

因为后端业务将访问控制逻辑完全交由前端的 AP 进行，迫切需要适当的机制确保后端业务能信任 AP 转发的业务流量已经通过了认证和授权。这种机制尤

其重要，因为 TLS 握手和传输在前端代理就终结了，前端代理是通过另外的加密通道传输 HTTP 请求给后端业务。

为满足上述要求，需要一个能够建立加密通道的双向认证机制----举个例子：一种可能的实现是基于 TLS 双向证书认证和企业公钥基础设施。BeyondCorp 采用了内部开发的认证和加密框架 LOAS(Low Overhead Authentication System，低开销认证系统)，它可以对代理和后端之间的所有通信进行双向认证和加密。

前端和后端之间进行双向认证和加密的一个好处是，后端可以信任 AP 插入的任何元数据（通常以 HTTP 消息头的形式）。在反向代理和后端之间额外插入元数据、使用自定义协议（比如， Apache JServe 协议）并不是什么新方法，但 AP 的双向认证机制，确保了元数据的完整性。

采用此方法的另一个好处是，当 AP 逐渐部署了更多新功能时，后端可以通过简单地解析相应的消息头，获取 AP 插入的新功能数据，并选择所需信息。使用这个功能可以将设备的安全等级传递到后端，后端可据此调整服务内容。

ACL 语言

将领域特定语言（domain-specific language, DSL）用于表述 ACL 是解决集中式授权挑战的关键。这种语言支持静态编制 ACL（有助于提高性能和可测试性），同时减少了策略表述和具体实现之间的逻辑鸿沟。这一策略提高了了以下各方的职责分离：

- **安全策略团队：**负责对访问策略进行抽象和静态编制
- **清单管道团队：**根据发起访问请求的用户和特定设备，负责提供对资源的访问决策的具体实例化（请参阅“谷歌 BeyondCorp：从设计到部署”[2]了解关于清单管道的更多细节）
- **访问控制引擎团队：**负责评价和执行安全策略
- ACL 语言语义上采用首次匹配（first-match）模型，和传统防火墙规则比较类似。虽然这种模型存在一些极端情况（例如，规则之间会相互覆

盖），但好在这些情况已经众所周知，安全团队理解起来还是相对容易。当前采用的 ACL 结构包括两大部分：

- **全局规则：**通常是粗粒度的，影响所有服务和资源。例如，“安全等级低的设备不允许提交源代码”。
- **针对特定服务的规则：**专属于某个服务或主机，通常包括和用户有关的断言。例如，“群组 G 中的所有厂商允许访问 Web 应用 A”。

以上结构基于一个假设，即服务所有者可以识别应用策略的 URL 地址范围，除非请求对象不在 URL 中指定而在报文主体中指定（可以通过修改 AP 来处理这种情况）。不可避免地，针对特定服务的规则规模会越来越大，因为访问代理会对越来越多的服务负责，而这些服务都需要特定的 ACL 规则。

全局规则在处理一些特殊的安全状况（例如，员工离职）和应急响应（例如，浏览器漏洞利用或设备被盗）时具有极大的便利性。比如，这种机制曾帮助我们成功处置 Chrome 浏览器某个第三方插件的 0Day 漏洞风险,通过创建一条全新的高优先级规则，使用老版本的 Chrome 浏览器时将会被重定向到一个带有更新指南的页面，该规则 30 分钟内就在整个公司完成部署和强制执行，最终，存在漏洞的浏览器的数量急剧减少。

集中式日志记录

为了进行必要的事件响应和取证分析，所有请求日志必须进行持久化存储。AP 提供了一个理想的日志记录点。日志记录主要包括部分请求头、HTTP 响应码、调试或重构访问决策和 ACL 评估过程所需的元数据，一般包括访问请求的设备标识和用户标识。

访问代理的特性：运维弹性

自助服务开通

一旦访问代理准备就绪，企业应用的开发人员和所有者就可以着手配置通过代理的服务访问模式。

当谷歌逐渐从网络层开始限制用户对公司资源的访问，访问代理就成为了能在迁移过程中保持服务正常运行的最快方案。显然，单个团队无法支撑对 AP 配置的全部更改，因此将 AP 配置过程结构化，使用户可以更为便利地使用自助服务。用户保留了他们自己的配置片段的所有权，而 AP 团队保留构建配置系统的所有权，可以校对、测试、灰度发布（金丝雀发布）和更新配置。

这种设置有几个主要好处：

- 解放了 AP 团队，让他们不再需要根据每个用户请求持续修改配置
- 鼓励服务所有者拥有他们的配置片段（并为其编写测试）
- 确保开发速度和系统稳定性之间的平衡

仅仅只需几分钟就可以在 AP 后设置一个服务，用户也可以在不请求 AP 团队支持的情况下迭代自己的配置片段。

多平台身份认证的挑战

目前，已经理清了 BeyondCorp 前端在服务侧的情况，包括了其实现及由此带来的困难和挑战。现在将采用类似方法来梳理 BeyondCorp 中客户端方面的情况。

准确的设备识别至少需要以下两个组件：

- 某种形式的设备标识
- 能追踪任何指定设备最新状态的清单数据库

BeyondCorp 的目标之一是以适当的设备信任替代基于网络的信任。每个设备都必须有一个一致的、不可克隆的标识，设备的软件、用户和位置的相关信息必须集成到清单数据库中。正如在前两篇 BeyondCorp 论文中所说，构建和维护设备清单库可能面临着诸多挑战。下面几个小节将更详细地描述与设备认证相关的挑战和解决方案。

台式机和笔记本电脑

台式机和笔记本电脑使用 x.509 证书，以及系统证书库中对应的私钥。密钥存储是现代操作系统的标准功能，它确保了通过 AP 与服务器通信的命令行工具（和守护进程）可以与正确的设备标识匹配。由于 TLS 要求客户端提供拥有私钥的加密证明，而且设备标识存储在类似可信平台模块（Trusted Platform Module, TPM）的安全硬件中，这能确保标识的不可欺骗性且不可克隆性。

但这种实现方式有一个主要缺点：证书验证提示通常会影响用户体验。幸好大多数浏览器都支持通过策略配置或插件扩展自动提交证书。但如果客户端提供了无效证书，服务器拒绝 TLS 握手,此时也会对用户有所影响。TLS 握手失败，浏览器会显示特定的错误消息，且大多不可定制。为了提升用户体验，AP 可以接受没有有效客户端证书的 TLS 会话，但必要时会按需弹出一个 HTML 拒绝页面。

移动设备

上述解决证书提示问题的策略，在几个主流移动平台中都无需考虑。移动设备的认证可以不依赖证书，因为移动操作系统本身就可以提供安全性高的设备标识。比如，ios 设备可以使用苹果的 Vendor 标识符（Identifier For Vendor, IDFV），安卓设备使用企业移动管理（EMM）应用提供的设备 ID。

一些特殊情况和例外

虽然在过去的几年中已经将绝大多数 Web 应用程序迁移到访问代理，但是仍然有些特殊的用例，要么自身无法与访问代理模式兼容，要么需要经过特殊处理才能兼容。

非 HTTP 协议

有些谷歌企业应用程序使用了非 HTTP 协议，这些协议需要端到端加密。为了通过 AP 为这些协议提供服务，需要将它们封装在 HTTP 请求中。

幸好有现成的 ProxyCommand 工具，因此在 TLS 上将 SSH 业务封装成 HTTP 流量并不难。我们开发了一个类似 Corkscrew 的本地代理，不同之处在于我们使用了 WebSockets 进行封装。虽然 WebSockets 和 HTTP CONNECT 请求都能兼容 AP 的 ACL 评估，但 WebSockets 本身能从浏览器继承用户和设备的身份凭据，这一点比 CONNECT 机制更占优势。

对于 gRPC 和 TLS 流量，最终选择使用 HTTP CONNECT 请求进行封装。封装有个很明显的缺点，它会给传输带来性能损失（虽然可以忽略不计）。但封装有一个重要优势，它能够将设备标识和用户标识分离，在协议栈的不同层来实现。这种方案缘于基于清单库的访问控制是一个相对新的概念，虽然通常现有协议支持用户认证（例如，LOAS 和 SSH 都支持），但要扩展到支持设备认证并不容易。

在封装 CONNECT 请求的 TLS 层执行设备认证，就不需要重写应用来识别设备证书。以 SSH 为例：客户端和服务器之间能够使用 SSH 证书来进行用户认证，但是 SSH 原本并不支持设备认证。此外，不能通过修改 SSH 证书来传递设备身份，因为 SSH 客户端证书默认是可移植的：一个 SSH 证书可以用在多个设备上。类似于 HTTP 的处理方式，CONNECT 封装确保了用户认证和设备认证的良好分离。使用 TLS 客户端证书来认证设备的时候，也可以使用用户名和密码的方式来认证用户。

远程桌面

在 Chrome 代码库中公开可用的 Chrome 远程桌面[5]，是谷歌 BeyondCorp 主要使用的远程桌面解决方案。虽然 HTTP 的封装协议可以满足很多使用场景，

但还有些专门用于远程桌面的协议，它们对通过 AP 后可能产生的额外延迟格外敏感，需要单独考虑。

为了确保请求得以授权，Chrome 远程桌面在连接建立的交互流程中引入了基于 HTTP 的授权服务器。这个服务器位于 Chromoting 客户端和 Chromoting 主机之间充当第三方授权服务器，同时也帮助两个实体共享密钥，与 Kerberos 协议工作方式类似。

我们将授权服务器作为 AP 的一个简化的后端服务来实现，并为其配置特殊的 ACL。这种实现效果还不错：通过 AP 带来的额外延迟仅在每个远程桌面会话发起时发生一次，并且也确保了访问代理能对每个会话创建请求都实施 ACL。

第三方软件

第三方软件通常比较麻烦，因为它可能无法提供 TLS 证书，也可能其实现逻辑假设网络总是直连的。为了适配这些软件，我们设计了一种可以自动建立点到点加密隧道（使用 TUN 设备）的方案。软件对隧道无感知，就像是直连到服务器一样。理论上来看，隧道建立机制与远程桌面方案类似：

- 客户端运行辅助程序来建立隧道
- 服务端同样运行辅助程序作为 AP 的后端
- AP 执行访问控制策略并且协助会话信息和加密密钥在客户端和服务端的辅助程序之间交换

经验教训

ACL 很复杂

推荐下面的最佳实践来减少 ACL 相关的困难：

- **确保语言的通用性。**AP 的 ACL 改变了无数次，而且还持续不断地增加新信息（如，用户和组）。因此需要定期更新可用功能，并且确保语言自身不会妨碍这些更新。
- **尽早启动 ACL。**原因有两个方面：
  - 确保用户尽快了解 ACL 以及访问被拒绝的可能原因。
  - 确保开发者尽快开始调整代码来满足 AP 的要求。例如，为了处理用户和设备认证，我们甚至重新开发了软件来替换 cURL。
- **完善自助服务。**正如前面提到的，单个服务配置团队无法支撑多个团队。
- **建立能将数据从 AP 传递给后端的机制。**正如前面提到的，AP 能够安全地将额外数据传递给后端，允许其能够进行细粒度的访问控制。尽早规划所需要的功能。

紧急情况

事先充分测试，充分准备，以应对意外紧急情况。尤其注意以下两类紧急事件：

- **产品类紧急事件：**由于服务访问的逻辑链路上关键部件的中断或失灵造成的紧急事件。
- **安全类紧急事件：**由于迫切需要授权/撤回特定用户和/或资源的访问造成的紧急事件。

产品类紧急事件

为了确保 AP 在大多数宕机期间还能存活,请根据 SRE 最佳实践进行设计和运维[3]。为了避免可能出现的数据源中断，需要定期对所有数据进行快照以便能本地访问。此外，还需要设计不依赖于 AP 本身的 AP 修复路径。

安全类紧急事件

安全紧急事件比产品紧急事件更为敏感，因为在设计时往往容易被忽略。在用户撤销/设备撤销/会话撤销时均需考虑到 ACL 推送频率和 TLS 问题。

用户撤销相对简单：作为撤销过程的一部分，已撤销的用户将自动添加到特殊组，通过一条靠前的 ACL 全局规则（请参阅上面的“ACL 语言”）确保这些用户访问任何资源的权限都被禁止。会话令牌（例如，OAuth 和 OpenID Connect 令牌）和证书有时候会泄露或丢失，同理也需要撤销。

正如第一篇 BeyondCorp 论文中所说[1]，除非收到设备清单管道的状态上报，否则设备标识不可信。这意味着即使丢失 CA 密钥（意味着不能撤销证书）也不会失控，因为直到被列入清单管道的目录中，新的证书才可信。

由于上述特性，我们决定彻底忽略证书撤销过程：如果怀疑证书相应的私钥丢失或者泄露，不再发布证书撤销列表（certificate revocation list, CRL），而是降低证书的清单信任等级。清单本质上就是可信设备标识的白名单，并且不依赖于 CRL。这种方法的主要缺点是它可能会带来额外延迟。不过通过在清单和访问代理服务器之间设计快速传播通道，可以相对容易地解决这种延迟。

为了保证执行策略的及时可达，需要一个 ACL 的标准快速推送机制。ACL 超出一定规模后,必须要将部分 ACL 定义过程委托给服务所有者,这就会导致一些不可避免的错误。虽然单元测试和冒烟测试通常可以发现明显错误，但逻辑错误会通过安全措施渗透，并进入生产阶段。工程师必须具备快速回滚 ACL 变更的能力，才能恢复丢失的访问权限、锁定意外的访问权限。引用之前 Chrome 插件的 0Day 漏洞为例，快速推送 ACL 是应急响应团队的关键能力，通过快速推送自定义 ACL 可以强制用户进行更新。

工程师需要支持

迁移到 BeyondCorp 不可能一蹴而就，需要多个团队之间的协调和沟通。在大型企业中，将整个迁移任务委托给单个团队是不可能的。迁移很可能涉及一些不能向后兼容的变更，这需要得到管理层的强大支持。

迁移的成功很大程度上取决于团队在访问代理背后配置服务的难易程度。以减轻开发人员的开发负担为目标，要把异常情况的出现维持在最低限度。提供合理的默认设置，为常见用例撰写指南和文档。使用沙箱应对更高级和更复杂的变化，比如可以创建一个访问代理的单独实例，负载均衡器会忽略这个实例，但开发人员还可以访问（如临时覆盖其 DNS 配置）。沙箱在大部分情况都非常有用，比如在对 x.509 证书或底层 TLS 库进行重大变更之后，需要确保客户端 TLS 连接能成功进行。

展望未来

虽然 BeyondCorp 的前端实现在很大程度上是相当成功的，但仍然有一些问题尚未解决。首当其冲的，就是台式机和笔记本使用证书进行身份认证，而移动设备则使用设备标识。证书的轮换仍然很痛苦，因为出示一个新的证书需要重启浏览器才能确保现有的套接字已经关闭。

为了解决上述问题，计划将台式机和笔记本电脑同样采用移动设备的方式，以消除对证书的需求。构建一个桌面设备管理器来处理这种迁移，该桌面管理器看起来与移动设备管理器非常相似。它将提供一个通用的标识，以设备-用户-会话-ID（DUSI）的形式出现，DUSI 会在所有浏览器和工具间共享，也许会使用一个通用 OAuth 令牌授予守护进程来实现。一旦迁移完成，不再需要通过证书验证台式机和笔记本电脑，并且在各类操作系统中的所有的控制都可以持续使用 DUSI。

结论

作为 BeyondCorp 的核心组件，访问代理的部署实施考虑了谷歌特有基础设施架构和用例。访问代理的最终设计实践与常见**网站可靠性工程（Site Reliability Engineering, SRE）**最佳实践一致，并且已证明其具有较高的稳定性及伸缩性——在部署过程中，访问代理已经完成了几个量级的增长。

任何想要实现类似 BeyondCorp 安全模型的组织都可以采用类似访问代理设计和部署的解决方案。希望本文分享的谷歌如何解决多平台认证、特殊案例和例外等挑战的解决方案，以及在这个项目中所学到的经验，可以帮助其他组织能以最小的代价实现类似项目。

参考文献：

[1] R. Ward and B. Beyer, “BeyondCorp: A New Approach to Enterprise Security,” login:, vol. 39, no. 6 (December 2014): https://www.usenix.org/system/files/login/articles/login_dec14_02_ward.pdf.

[2] B. Osborn, J. McWilliams, B. Beyer, and M. Saltonstall, “BeyondCorp: Design to Deployment at Google,” login:, vol. 41,no. 1 (Spring 2016): https://www.usenix.org/system/files/login/articles/login_spring16_06_osborn.pdf.

[3] B. Beyer, C. Jones, J. Petoff, and N. Murphy, eds., Site Reliability Engineering (O’Reilly Media, 2016).

[4] Apache JServer Protocol: https://tomcat.apache.org/connectors-doc/ajp/ajpv13ext.html.

[5] https://src.chromium.org/viewvc/chrome/trunk/src/remoting/.

【第四篇】迁移到 BeyondCorp： 提高安全性的同时保持生产力

如果你熟悉过去两年发表在《;login:》[1-3]上的谷歌 BeyondCorp 网络安全模型文章，可能会想：“这一切听起来不错，但我的组织如何从现有模式转变到 BeyondCorp 类似模式？需要做些什么？这种转变对公司和员工会有什么影响？”本文讨论了从现有网络迁移到 BeyondCorp 模型所采用的方法，探讨如何实现在根本上改变网络访问模式的同时，却不影响公司生产力。

向 BeyondCorp 及其类似模型迁移面临诸多挑战，其中有几个尤其值得注意:

- 这个过程会影响整个公司。要让每个人都参与进来并保持一致和确保知情，就需要得到各级管理层的承诺和支持，这就需要与各方广泛沟通，从拥有个性化服务的团队，到管理层，再到支持团队，最后到用户。
- 迁移不可能一蹴而就。这个过程是多层次和渐进式的，包含信息收集、实验部署、流程和技术修正等各阶段，以及必要地点与时间的例外和补救。
- 这个过程涉及到在业务/技术栈中的多层、甚至所有层上进行更改：网络、安全网关、客户端平台和后端服务。为了保证不同层面工作进展相互独立，需要各层分治，确保多线程并行且易于管理和实现。

下面将讨论我们如何将 BeyondCorp 迁移工作进行分解，介绍各个层面平滑、一致的迁移所需的技术和工具，当然，必须确保整个过程对用户导致的负面影响最小化。

先决条件：认同和沟通

着手迁移到一个类似 BeyondCorp 的模型之前，需要来自公司高层及其他干系人的支持。第一步是理解和沟通迁移的动机：减少成功网络攻击所造成的威胁，同时保持生产力。需要将迁移背后的基本原理、威胁模型以及维持“业务照常运行”所需的成本形成文档。然后，准备好向每一个业务部门解释迁移过程的价值

和必要性。与所有的安全项目一样，部署新模型需要付出代价：新工具、额外流程和使用习惯的改变。高层管理者需要积极支持这种改变，并将这种改变的动机和认同理念在所有干系人中推广。

有了管理者的准许和认同，接下来确定并争取到关键领域负责人的支持：安全、身份、网络、访问控制、客户端和服务器平台软件、关键业务应用程序服务，以及任何第三方合作伙伴或 IT 外包。负责人应该梳理和确定各领域专家，获得其承诺，并确保他们投入时间和精力。谷歌 BeyondCorp 团队是一个分布到全球各地的虚拟团队，有负责决策的总监，有项目技术经理负责协调落地执行。随着时间的推移，团队参与成员虽然会有所变化，但是高层领导、团队负责人和其他参与者会通过在线文档、邮件组和定期会议（面对面的和远程的）联系，始终保持对当前进展和项目状态的了解。

随着迁移工作的推进，通用的变更管理规则同样适用，因为每个工作组都有自己的关注点和优先级。要倾听反馈，调整兼顾每个参与者或受影响群体的特殊情况和要求。及时公开计划和资讯很有必要，但仅仅这样还不够，还需要互动沟通（最好是当面沟通，至少也要通过视频或音频会议进行）才能加深团队间的协同、更易获取帮助和得到认可。

分步推进

BeyondCorp 的总体目标是，从允许客户端直接访问服务器的网络，过渡到无特权网络：取消客户端直接访问后端服务器的特权。详情请参见 BeyondCorp 系列文章的第一篇《BeyondCorp，一种新的企业安全方案》[1]。为此，谷歌曾考虑依次阻断每个应用或服务器，以便逐步移除遗留 VLAN 的访问特权。但这一策略并不理想，有两个原因：一是在网络层部署和协调很困难；二是在应用层增加了影响生产力的风险。因此，决定在最终的 BeyondCorp 配置中部署一个新的 VLAN。这个 VLAN 只允许通过访问控制网关访问服务器网络，确保所有流量都经过身份认证、授权和加密。这一策略不是逐步限制遗留 VLAN 的特权，而是逐步将设备最终都转移到这个新的 VLAN 上。

VLAN 迁移项目实现了一个复杂但至关重要的目标，迁移遗留“特权”网络的用户设备，并将它们分配给新的受控无特权客户端（Managed Non-Privileged Client, MNP）VLAN。这次迁移有一关键约束：对于运行在新 VLAN 工作站上的任何遗留应用，无论是预期还是必需，直接访问服务器网络都将失败。因此，近期目标是在不破坏业务关键操作的情况下实现迁移工作。为此，采用了三管齐下的策略来实现这一目标：

1、广泛分析网络流量日志

2、识别和修复不符合迁移要求的应用程序

3、在确定设备可以在新网络上成功运行之后，迁移设备

这种策略允许网络层面相对稳定地应用新的配置，且能够独立于 BeyondCorp 的其他部分进行。BeyondCorp 的设计包括基于 802.1x 认证进行网络准入以及 VLAN 分配，这种方式能够将网络层与迁移策略的细节隔离开。更高层的软件和数据分析决定了设备的 VLAN 分配，并由 RADIUS 服务器将其返回给网络层。

实现这一系列目标任务艰巨，需要对技术/业务栈的每一层进行修改。但迁移团队并没有试图在一次过渡中修改所有层（毫无疑问这会引起灾难性的崩溃），而是分步实现：

- 解耦网络层：新的 VLAN、802.1x、RADIUS 策略服务器
- 解耦客户端平台升级：证书生成和安装，用户认证工具
- 完成不符合迁移要求的服务和工作流的修复，逐步地迁移设备
- 持续修正流程和程序

第一步：802.1x 网络

在 BeyondCorp 的第一阶段，为每个用户设备安装证书并基于 802.1x 认证实现所有的网络访问准入。这个看似简单的步骤暗含了几个新的开发项目：证书颁发机构(CA)，为公司受控设备（针对所有操作系统）安装证书的工具，在网络交换机上启用 802.1x，集成一个策略驱动的 RADIUS 服务。以上开发项目并行开展。

安全团队设计了一个新的证书颁发机构，通过提供 API 接口的方式，使每个操作系统平台管理团队能够在对应的平台上获取并安装证书。每个平台团队独立部署软件、工具和监测系统，执行和监测每个设备的证书安装。在与接入交换机集成的同时，我们还创建了批量分发和维护证书的流程。

同步开展的还有对接入交换机的重新配置工作，为接入交换机配置新的 VLAN 定义，开启 802.1x 认证，支持基于 RADIUS 的 VLAN 分配。自动脚本通过审计交换机的升级，来识别尚未配置新 VLAN 的交换机。这样，RADIUS 服务器就不会为这些交换机分配其尚未开通的 VLAN。

采用 802.1x 认证，就可以将 VLAN 分配的控制权从网络层转移到 VLAN 策略服务器。为了减少新 RADIUS 服务器可能引发的故障，初始策略仅匹配现有 VLAN 分配（包括复杂的黑名单和白名单）。一开始，配置策略服务器在审计模式工作，比对新的 VLAN 分配与既有的 VLAN 分配。当两者差异足够小，就启用新策略。此后，就可以使用软件和数据驱动的策略，接近实时地管理设备的 VLAN 分配。这个简单初始策略的使用，使得最终状态（和过渡）策略仍在开发中时，在网络层面率先启用动态 VLAN 分配。

以成功为导向的迁移

全面部署 802.1x 认证花费了数年时间，随后又花了更长的时间来实现基于清单、按信任等级动态分配 VLAN，并将其作为 RADIUS 策略服务器的输入[2]。在这些开发工作进行时，需要识别出两类主要用户群和应用服务：那些准备好采用 BeyondCorp 的，和那些需要升级网络和安全能力才能兼容 BeyondCorp 的。

首先，捕获和分析网络路由器的流量。通过日记记录和分析经过公司路由器的全部流量的部分采样，发现使用模式不兼容的情况。此外，这种分析还可以协助发现网络上的异常、意外和未经授权的流量。识别出这些不兼容 BeyondCorp 的应用，就可以尽早对这些应用进行兼容性改造，并避免对这些应用的使用者造成干扰。

有些网络用例，比如使用 NFS/CIFS 文件服务器的工作站，显然是不兼容 BeyondCorp 的。虽然 NFS / CIFS 文件服务器是实现文档共享和协同的最简单方法，但其底层协议不支持我们所需的安全属性（强加密和身份认证）。为了消除对 NFS / CIFS 的依赖，我们很早就启动了一个项目，来实现两个目标：一是将 NFS 主目录移动到本地磁盘，并通过自动备份同步至安全的云存储；二是使用 Google Drive 或其他安全的文件共享技术取代其它 NFS 的使用。即便如此，还是有些应用程序非常依赖 NFS，如 CAD（计算机辅助设计）编辑器，对于这种情况，我们在将其用户和工作站移动到受限的 MNP VLAN 之前，就需要定制解决方案。在下文的“修复困难用例”一节中将讨论如何处理这些特殊需求的框架细节。

还有些不兼容的工作流不那么容易判断出来，但一旦受到 MNP 网络的 ACL 限制时，这些业务就会运行失败。让其失败是必要的，因为我们无法假设 NFS、RDP、SQL 等具有足够的身份认证、授权和加密能力。当不得不在网络层面进行修复时，检测出这些工作流后通过改变设备的网络分配来恢复其生产力，费时费力。为了避免这种情况对生产力产生巨大影响（更不用提影响用户的情绪），需要一个分析驱动的策略，在将用户分配到 MNP VLAN 之前，预先检测并修正可能失败的工作流。

为了方便在无特权网络上进行简单的分析和用户工作流测试，我们创建了一个基于 C/S 架构的网络 ACL 仿真器，仿真器能识别被 MNP ACL 阻塞的网络数据包。底层技术采用 Capirca（参见源代码[4]），并依据真实的 MNP ACL，创建本地 iptable 规则或其他的包过滤规则。在分析和迁移阶段，用户设备继续在特权网络上运行，而 MNP 仿真器监视网络流量，并将所有非 MNP 兼容的流量的源和目的地址记录到中心数据库。IP 源地址标识潜在故障用户，IP 目的地址

标识潜在故障服务。通过分析日志（必须考虑适当的隐私限制），可以识别出已经兼容 MNP 的设备，从而将它们分配到 MNP VLAN。同样，可以识别出暂不兼容流量的设备、用户和服务，并启动项目将这些服务转移到为其需求其他解决方案。随着时间的推移，更多的设备变为兼容设备并被自动分配到 MNP VLAN。

在第二种模式下，MNP 仿真器实际上也可以阻止/丢弃非 MNP 流量，从而在不依赖 MNP VLAN 和 802.1x 管道网络层部署的情况下强制执行 MNP ACL。尽管 ACL 的最终执行是在网络设备中完成，设备中将 ACL 与用户（或黑客）的滥用隔离，但在试用和过渡阶段，在客户端工作站上启用和禁用这种“强制”模式要更容易、更迅速。客户端强制执行模式既是迁移过程中的重要步骤，也是用于测试验证的自助服务工具。如果当初没有这种工具，BeyondCorp 迁移团队恐怕难以实现最终快速、成功的设备迁移工作。

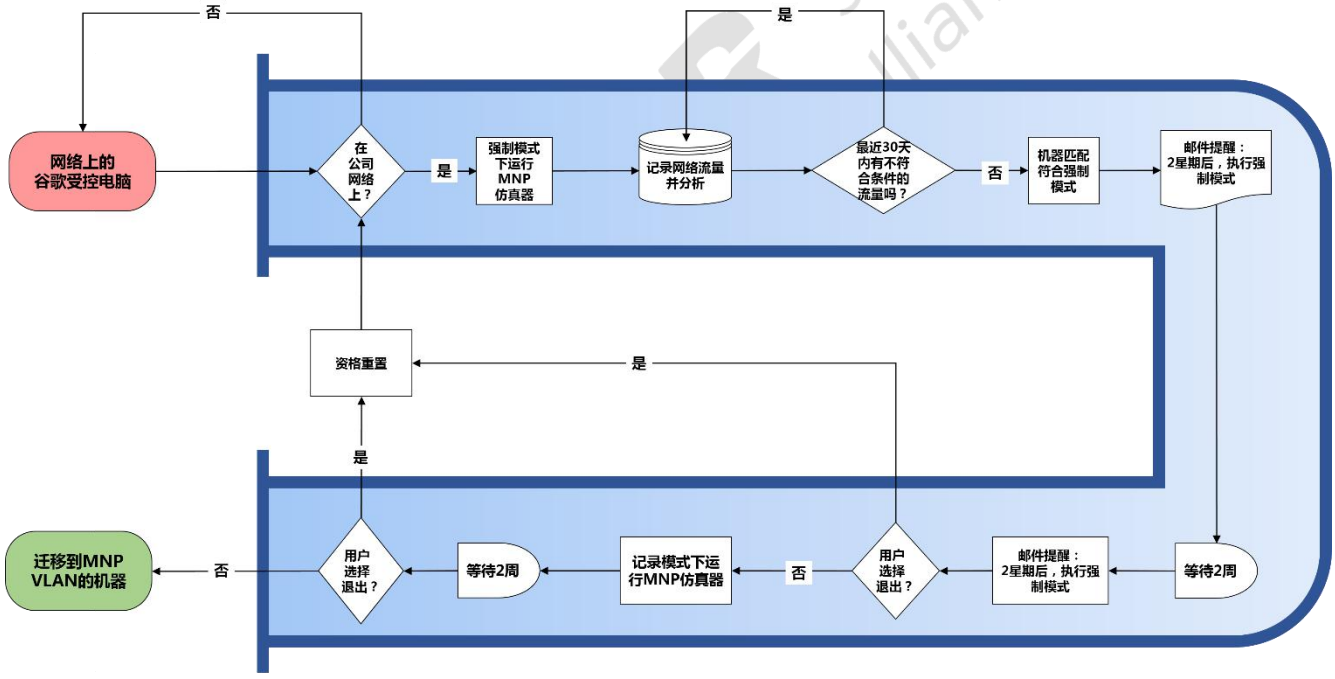

图 1 将谷歌电脑迁移到受控的无特权（MNP）网络的管道

使用访问代理处理简单用例

谷歌的基本安全策略要求所有从工作站流向服务器的业务流量都需要：

- 已认证（识别发出请求的设备和用户）

- 已授权（验证用户和设备是否被允许访问后端资源）
- 已加密（防止窃听）
- 单独记录日志（为了协助取证分析）

对于 HTTP/S 流量和 HTTP 封装的 SSH 流量，访问代理[3]可以满足以上所有要求。

幸运的是，在谷歌内大多数高频使用的应用程序都是基于 B/S 架构的 Web 应用。这种“幸运”并非巧合，因为谷歌有一独特的核心理念在业界“闻名”：尽可能的使用基于 B/S 架构的应用。谷歌为每个 Web 应用提供者准备了工具和文档，让每个应用提供者都可以配置自己的应用运行在访问代理之后。

当一个应用运行在访问代理后端时，企业和公共 DNS 包含一个可以解析到访问代理的 CNAME，这样此类应用的 URL 在企业和公共网络中都具有同样的易用性和安全性。能从公共网络访问企业应用就意味着，经过身份认证的远程用户可以直接访问企业 Web 应用，而不需要再拨通 VPN 进行访问。因此，使用和支持 VPN 连接用于远程办公所需的费用会立即大幅减少。据粗略估算，由此产生的生产力提升轻松超过了 BeyondCorp 的实施成本。

一旦基于 B/S 架构的应用在访问代理后受到安全保护，我们就可以大刀阔斧地推进了。通过启动一个自动化流程，分析、验证并将设备迁移到无特权网络；在不到一年的时间里，超过 50%的设备迁移到无特权网络访问模式。

修复疑难用例

虽然可以通过访问代理来处理大多数的应用，但还有些应用难以通过此方法处理。整个迁移的时间安排还必须考虑到非 Web 案例的长尾问题，因为这些问题用例需要更多的时间和资源。为保证这些用例能够兼容 BeyondCorp，需要新工具、新技术和工作流改造。

特别是，一些工作小组使用基于非 HTTP 协议的第三方桌面或“胖客户端”应用程序，这就涉及到一系列特殊的问题。例如：

- 有些工具原本就是依赖网络文件共享。
- Java 应用程序可能使用远程方法调用（Remote Method Invocation，RMI）或其他直接套接字连接。
- 许多工具可能需要使用非 HTTP 套接字和协议连接许可服务器。

即使是基于 HTTP 的应用程序，也可能遇到一些莫名奇妙的、出乎意料的问题。例如，有些应用无法支持客户端证书或适当的用户凭证，而有些应用则内置了一些负载均衡逻辑，导致不易和访问代理整合。对于其中一些案例，通过调整访问代理，允许来自 MNP VLAN 的流量在没有证书的情况下通过。这种临时策略效果还不错，因为设备必须出示证书才能访问 MNP。每个有问题的案例都需要一个诊断和补救项目。

为了解决这类疑难杂症，开发了一个解决方案，使用多端口加密通道来传输客户端和服务器之间的流量：

- 当客户端向服务器发起连接时，访问代理使用常规的用户和设备身份认证及授权。
- 客户端上的路由表将数据包发送到 TUN 设备，该设备可以捕获和加密到特定后端服务器的流量。
- 加密后的数据包采用基于 UDP 的封装协议直接在客户端和加密服务器之间传输。
- 加密服务器只允许应用程序必须的服务和端口流量通过。

用例	**解决方案**
B/S 架构的 HTTP/S 连接	访问代理
HTTP 命令行的原生应用： 提供了一个客户端代理服务器程序，该服务器提供平台证书，以建立与访问代理的认证与加密的连接。然后，将简单应	本地认证代理程序

用定向到本地主机代理。	
单个 TCP 连接： 对于需要 TCP 套接字连接到服务器的应用，一般通过与后端堡垒机建立 SSH 连接来解决，并为简单 TCP 应用端口建立隧道	SSH 隧道和端口转发
多端口或无法预测的端口号	加密服务隧道
对延迟敏感、实时，UDP 流	加密服务隧道

表 1：解决问题工作流的方法

这种方法可以让第三方传统应用更安全地从任何网络连接到它们的服务器，同时也满足了 BeyondCorp 要求的身份认证、授权和加密。

表 1 描述了解决问题工作流的常规方法。详细论述请查阅《BeyondCorp，访问代理》[3]。在有些场景下，表 1 中的解决方案还要求用户通过运行脚本或在访问后端资源之前提供必要的身份认证来修正工作流。

有些基本框架服务也不具备兼容性。当然，这些关键服务的兼容问题并未阻止迁移的整体推进，而是通过开通从 MNP 到特定端口或服务器的临时访问权限进行解决。为了防止这些临时例外变成常态甚至颠覆 BeyondCorp 的基本目标，只有服务所有者给出实现和部署兼容解决方案的明确计划时，我们才允许进行临时的例外放行。

随着一个一个的应用或用例完成整改或调整，借助自动化的分析、验证和迁移工具，越来越多的用户和设备转移到无特权 VLAN 上。随着工作推进，网络日志记录和分析可以用于度量已成功迁移到 MNP 的用户和设备数量。

逐步上线并不断完善迁移方法

MNP 仿真器，分析管道，以及将设备自动分配到 MNP VLAN，组成了一个重要的软件开发和流程再造项目。所以整个项目的开发和部署也是逐步完成的：

首先在针对各个阶段进行小规模测试，持续修复软件，合适的用户调整通告，培训技术支持团队，然后逐步推进到全面部署。

当识别出那些不兼容工作流的用户，仿真和预分析的方法有助于规避对这些用户的负面影响。然而，这种方法将所有新配置的、尚未分析的设备分配给特权网络，并且没有阻止未迁移的用户使用或创建新的不兼容应用，因此它不能作为长期策略。通过纠正大量用例来减少异常案例后，实施方法变为“默认采用 MNP”策略。随着工作逐项推进，全部设备被默认分配到 MNP，同时对那些由于工作职责需要使用未修复应用的用户设备，予以例外处理。这个基于策略的分配完成了从“证明用户会成功，然后迁移设备”到“假设用户会成功，直接迁移设备”的演变。

扩大支持，尽量减少对员工的影响

使用上述工具和流程，能够自动识别、联系和迁移整组用户。但无论是在迁移开始前，还是出现问题时，都需要一些办法来帮助用户，与用户沟通。技术支持的专业培训和增加与用户的沟通和互动，这两点对将工作流迁移到新模型至关重要。

技术支持赋能

在支持团队中培训一批技术人员，将他们培养成为 BeyondCorp 模型的专家和本地的主要接口人。项目上线的初期，这些技术人员帮助受影响用户能够在不影响迁移策略的情况下迅速恢复工作，还能有效地将问题准确地反馈给实施和策略专家。一开始，这些受过专业训练的技术人员比其他部门同事获得了更高的访问修复系统的权限。作为 BeyondCorp 上线的第一批“观察员”，他们可以提前参与思考，接下来的技术支持会需要哪些方式、工具和流程。此外，他们还通过全球科技论坛、讨论列表、午餐时间和办公时间来给其他支持团队做培训。随着信息的不断传播，将系统访问权限赋予全部支持团队人员。

成立本地专家组，使 BeyondCorp 团队能够直接与工作流不兼容的部门进行沟通。在本地专家组中确定一个资深对接人，问题部门就可以与 BeyondCorp 团

队项目经理直接沟通，一起找到解决方案。与此同时，允许并鼓励技术人员，让他们在发现问题后立即在内部文档中添加新的临时变通办法或修复手段，以便将解决问题的能力尽可能遍布全网，更有效地实现信息共享并获得规模化支持。

自助服务

为了避免出现海量问询，需要尽量减少员工疑问，并在无需技术人员人工干预的情况下能够对常见问题进行回答。当用户被选中进行迁移时，系统会自动给他们发送一封启动邮件，内含明确时间安排、迁移将如何影响他们的工作、以及项目信息链接、常见问题答疑链接、自助链接和加急服务点。

此外，还提供一个自助服务门户网站，允许受业务关键时间节点约束的用户延迟迁移。为了回答问题，并进一步扩大信息传播范围，创建了一个内部讨论列表，征集员工答案。通过对常见问题的分析，能够快速迭代启动邮件内容和项目文档。

在整个上线过程中，通过专门的 Web 应用，我们还能快速迭代并改进故障处理指南。这个 Web 应用清楚地识别了常见问题（例如，解释为什么用户被拒绝访问某个资源），提供了解决问题的步骤，并链接到知识库文章。用户可以解决诸如组成员关系和证书问题等常见问题，从而减少对技术支持的请求。Web 应用还通过将来自许多不同层面和系统的信息合并成一系列操作，来帮助技术人员解决错误。

内部宣传活动

团队还组织内部宣传活动来提高大家对 BeyondCorp 的认识，比如推出了电脑贴纸、标识和口号，还在办公室张贴随处可见的文章。这些材料都标明了自助服务和办公时间，任何人有任何问题都可以寻求帮助。BeyondCorp 团队坚持宣传、指导、提供帮助，这使其直接与用户建立信任、取得信誉、得到用户的理解支持。在整个过程中，企业内沟通和技术专家参与是至关重要的，尤其是在早期阶段，那时亟需为项目的愿景和潜在影响给定一个清晰的蓝图。

分阶段上线

BeyondCorp 最初是一个小规模试点，试点位置与项目团队很近。随着时间的推移，逐步延伸至具有本地技术专家的试点位置，最终扩展到风险高的工作流和距离项目团队远的地点。直到有了成功经验，用户支持，以及对策略的信心，我们才开始实现关键业务流的迁移。在此过程中，即便上线规模和受影响工作流在增加，但技术支持负载却在减少。分阶段上线实施是迁移能成功的关键。

最终结果

通过持续分析和改进上面提到的所有方法，BeyondCorp 团队还建立了一个系统，确保 BeyondCorp 能够在全球范围内扩展，而不会对业务、支持或用户体验造成负面影响。并不是简单地通过人海战术，而是通过构建系统和流程来有效地处理问题、进行升级和培训。此外，基于良好的沟通、开放和高度一致的目标，我们确信用户会帮助迁移团队一起实现变革。

随着公司越来越多的人采用 BeyondCorp 模式，我们也仔细追踪了由于 BeyondCorp 上线所产生的支持案例。近几个月来，BeyondCorp 相关问题只占技术支持团队全部处理问题的 0.3%。一开始这个百分比有 0.8%，但随着文档、培训、消息传播和上线方法的不断完善，升级问题已经稳步减少。与谷歌内部其他大规模 IT 变革相比，BeyondCorp 的支持问题少了 30%。

结论

在提高安全的急迫性与改变终端用户的使用习惯之间总是存在矛盾。当基础设施和工作流的改变威胁到生产力的时候，这种矛盾只会升级。在发展和稳定之间取得平衡，与其说是科学，不如说是艺术。BeyondCorp 能够取得成功、为员工所接受的关键原因是分析、可行的规划和主动沟通。

通过将 BeyondCorp 迁移工作划分为独立任务，可以确保各项任务并行向前推进，确保每个阶段的用户影响最低。尽管部署 BeyondCorp 到各个层级花费了

数年时间，但每一个里程碑都实至名归。这个过程逐渐使远程访问变得更容易，更快捷，网络管理变得更简单，安全性得以增强。

开发出能实现 BeyondCorp 安全模型的技术很具挑战性。上线规划和管理用户迁移同样具有挑战性。注意一定要确保每次迁移对用户的影响最小，并且不会中断正在开展的业务。每一次成功迁移都带来了对这个项目价值的全新认识，并为用户和管理人员带来了持续的热情和对项目目标的接纳。通过赋能一个跨职能团队，其中包括每个技术和实施团队的代表、安全策略的责任人还有终端用户支持和通信方面的专家，BeyondCorp 项目最终取得成功。

在谷歌内部，已经能够将在 BeyondCorp 工作中所学到的东西应用到其他项目和服务中。其中最显著的就是最近为谷歌云平台(Google Cloud Platform,即 GCP)增加的新服务（比如基于身份识别的访问代理 IAP）。BeyondCorp 所获得的最大经验教训之一，就是当遇到其它用例时，一定要分步完成项目，并持续完善优化策略。尽管这篇文章关注的是谷歌自己的经验，但它所分享的经验可以在任何组织中采用，无论规模大小，当然，获得相关干系人的坚定支持至关重要。

致谢

感谢以下的人员，有了他们的帮助才完成这篇文章：希瑟•阿德金斯（Heather Adkins）、杰夫•贝尔德（Jeff Baird）、达伦•比比（Darren Bilby）、约翰•布莱迪（John Brady），维克多•埃斯科贝多（Victor Escobedo），辛西娅•霍伊奇（Cynthia Horiguchi），迈克尔•简诺斯科（Michael Janosko），罗伯•皮斯古德（Rob Peasegood），丹•波尔斯比（Dan Polsby），瓦尔•斯蒂里斯（Val Stiris）和罗里•沃德（Rory Ward）。

参考文献：

[1] R. Ward and B. Beyer, “BeyondCorp, A New Approach to Enterprise Security,” ;login:, vol. 39, no. 6 (December 2014), pp. 6–11: https://www.usenix.org/system/files/login/articles/login_dec14_02_ward.pdf.

[2] B. Osborn, J. McWilliams, B. Beyer, and M. Saltonstall, “BeyondCorp: Des

ign to Deployment at Google," ;login:, vol.41, no. 1 (Spring 2016), pp.28–35:https://www.usenix.org/publications/login/spring2016/Osborn.

[3] L. Cittadini, B. Spear, B. Beyer, and M. Saltonstall," BeyondCorp Part III: The Access Proxy," ;login:, vol. 41,no. 4 (Winter 2016), pp.28–33: https://www.usenix.org/publications/login/winter2016/cittadini.

[4] Capirca is a tool designed to utilize common definitions of networks, services, and high-level policy files to facilitate the development and manipulation of network access control lists: github.com/google/capirca.

【第五篇】BeyondCorp ：用户体验

BeyondCorp 系列论文的前几篇讨论了在实践过程中我们如何解决各个方面的技术挑战[1-3]。在迁移过程中，除了技术因素，还要考虑人的因素：在整个迁移过程中，必须始终将用户牢记于心，为最终用户尽可能地提供无缝的体验。当出现问题时,我们希望用户知道如何解决,去哪里寻求帮助。本文将讨论谷歌员工在 BeyondCorp 模型中的工作体验,从新员工入职，新设备配置,到遇到问题时如何处理。

创造无缝的新员工体验

对于许多新员工来说,BeyondCorp 模型这个概念是相当陌生的:他们习惯了通过 VPN、公司专属 WiFi、和其他特权环境来访问他们日常工作所需的资源。BeyondCorp 上线之初, 许多新员工继续向我们的 IT 服务台团队（内部称为技术站 Techstop）请求 VPN 访问。用户过去习惯性地认为如果不在办公室的时候需要工作，就要经过复杂的 IT 设置，需要 VPN。BeyondCorp 架构师原本以为用户不在办公室，有远程访问需求时，会尝试直接访问内网资源，并发现可以成功访问。这样看上去非常完美：无需用户申请访问配置，无需技术站的支持负担，简直就是双赢。然而事与愿违，（远程访问需要申请 VPN 权限的）用户习惯根深蒂固。

新员工入职培训

显然，在用户开始谷歌的 IT 之旅时，就应该让其尽早了解这种新的访问模式，因此我们在新员工入职培训时就开始介绍 BeyondCorp。在培训中，我们有意避免去讲解模型的技术细节，而是关注最终的用户体验。我们强调用户不需要 VPN，可以“自动”获得远程访问权限；用户无需改变他们的工作流就可以在办公室、家里、飞机上，或咖啡馆工作。通过培训，我们向用户展示了 BeyondCorp 的谷歌浏览器（Chrome）扩展程序，作为 BeyondCorp 访问模型中最常见的面向用户的方式（有关扩展的更多细节，请参见下面章节“BeyondCorp 扩展”）；我们还展示了在 BeyondCorp 中代表连接“正确”的图标（参见图 2）。只要有“正

确”连接标识，用户就可以通过任何网络连接访问他们需要的绝大多数工具和资源。

新设备安装配置

当用户初次使用公司账号密码登录其公司设备时，其访问设置将被自动配置。为了实现这种无缝的入职体验，清单进程和平台管理工具在后台工作，以配置新的租用设备并进行初始化。如“谷歌 BeyondCorp：从设计到部署”[1]中所述，我们根据大量的数据来判定设备的信任等级，包括观察数据（最近安全扫描时间，补丁级别，安装软件等）和预设数据（分配的所有者，VLAN 等）。为了解决这种判定的复杂性，我们的清单团队遵循自动配置流程，以确保首次登录时正确信任新租用设备。验证必要的用户账密后，我们会自动将自定义 Chrome 扩展程序推送到用户设备。从用户的角度看，只要能够看到扩展中的绿色图标，他们就可以访问企业资源。通过在新员工培训中讲解 BeyondCorp 的 Chrome 扩展，基本消除了新员工困惑，并且可以支持新员工的远程访问请求。

减少 VPN 使用

尽管新员工在培训中了解了 BeyondCorp，但毕竟他们在入职谷歌的头几天中可是接受了大量的信息冲击，让每个人都能回忆起培训中的每个细节不太现实。于是我们修改了 VPN 申请流程和工具来强调在培训中讲解的 BeyondCorp 概念。

默认情况新员工没有访问 VPN 网关的权限，他们必须通过在线申请门户来申请 VPN 访问权限。在此门户上，我们明确提醒用户 BeyondCorp 是自动化配置的，他们在请求 VPN 访问之前应尝试直接访问他们需要的资源。

如图 1 中的流程图所示，如果用户跳过这个警告，我们还会对用户通过 VPN 隧道访问的服务进行自动分析。如果用户在过去 45 天内没有访问过任何一个 BeyondCorp 模式不支持的企业服务，我们就会向他们发送电子邮件，邮件中会解释，由于他们访问的所有公司资源都是通过 BeyondCorp 支持的，因此他们的 VPN 访问权限将在 30 天后到期，除非他们访问 BeyondCorp 不支持的服务。我

们在 VPN 访问权限失效前 7 天会再发送一个通知，然后在第 7 天结束后取消用户对 VPN 网关的访问许可。这种自动化流程使我们主动剔除对传统访问基础架构的不必要使用，并最终完全拒用 VPN 基础设施。

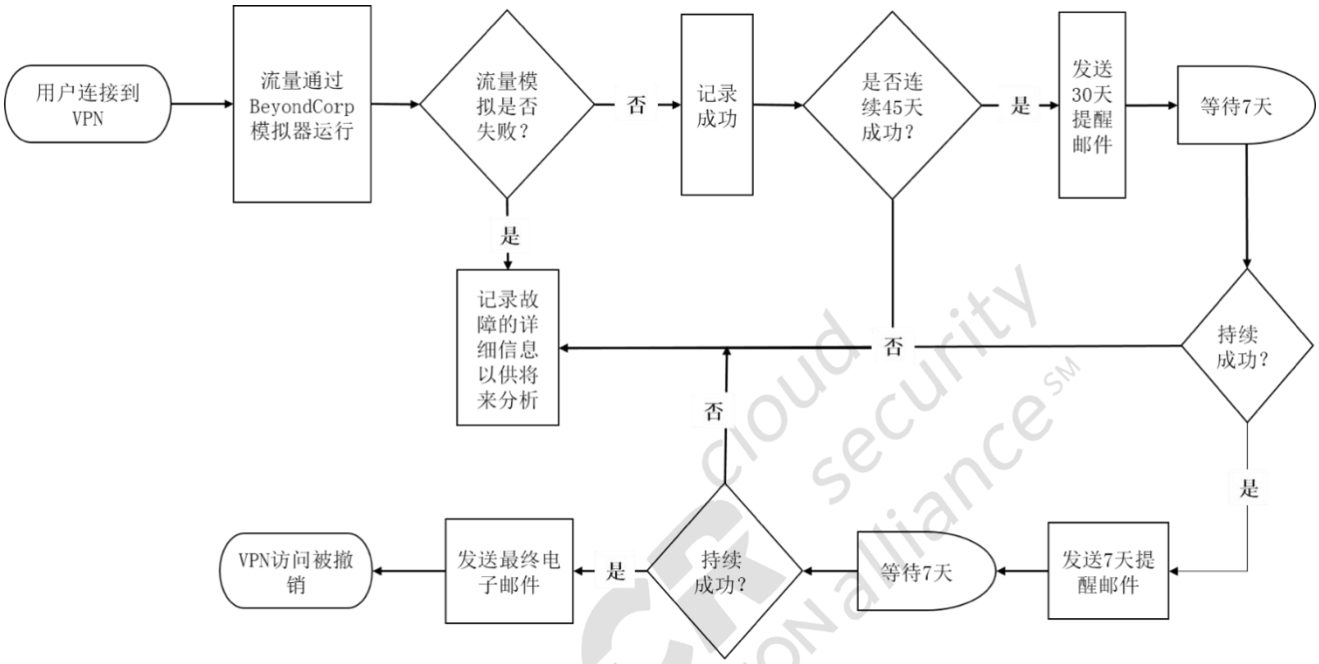

图 1 自动分析和取消员工的 VPN 使用

借用项目

实现 BeyondCorp 自动配置还带来了一个意外好处，为用户改进了其他方面的技术体验。其中一项最明显的改进是我们的借用笔记本电脑项目。像许多现代公司一样，我们员工的工作方式非常灵活，可以在办公桌，会议室，休息室或家中自由工作。移动设备 - 特别是笔记本电脑 - 对其生产力至关重要。为了处理忘带、遗失或被窃的情况，我们提供了一种自助式借用笔记本电脑程序，可以让用户尽快恢复正常工作。

使用遍布全球的自助式谷歌 Chromebook 笔记本电脑借用站，任何用户都可以将借用的笔记本电脑临时注册为自己的工作电脑，最长可达 5 天。从拿到笔记本到开启工作状态可能就几分钟时间，这样简单的流程让用户受益良多。借用设备开通足够简单，所需支持服务也随之减少，技术站的资源就可以释放出来处理

其他问题。当用户归还设备或借用时间到期时，系统会自动撤销其证书，并降低其信任等级，为下一个用户重新借用做好准备。

BeyondCorp 的 Chrome 浏览器扩展程序

通过或多或少地消除对 VPN 客户端的需求，我们可以通过 Chrome 扩展程序这个单一入口来封装几乎所有的访问需求——无论是远程访问还是本地访问。Chrome 扩展程序会自动管理用户的代理自动配置（Proxy Auto-Config，PAC）文件，明确将一些特定访问场景路由到访问代理[2]。当用户连接到网络时，该扩展程序会自动下载最新的 PAC 文件并显示“正确”连接的绿色图标。浏览器根据 PAC 文件中的规则自动将企业服务的访问请求路由到访问代理。这使得内部开发人员可以不用明确配置客户端访问入口参数的情况下部署企业内部 Web 服务：客户端访问入口配置要求开发人员在公网 DNS 中配置 CNAME 指向访问代理，访问代理就会自动处理用户身份认证和授权。

由于 BeyondCorp 扩展程序将所有流量路由到访问代理，用户将无法访问那些访问代理不可达的设备。另外，扩展必须下载正确的 PAC 文件，以便准确路由业务流量。这种设置可能在某些场景下可能会出现问题，比如有强制验证门户的网络连接的场景，或用户需要访问本地网络上与设备而不希望通过访问代理进行路由。我们需要对用户解释这些问题并提供补救方法，最好不要增加技术站的支持负担。Chrome 扩展程序的认证状态图标（如图 2 所示）提示了进一步排除故障的方法信息。

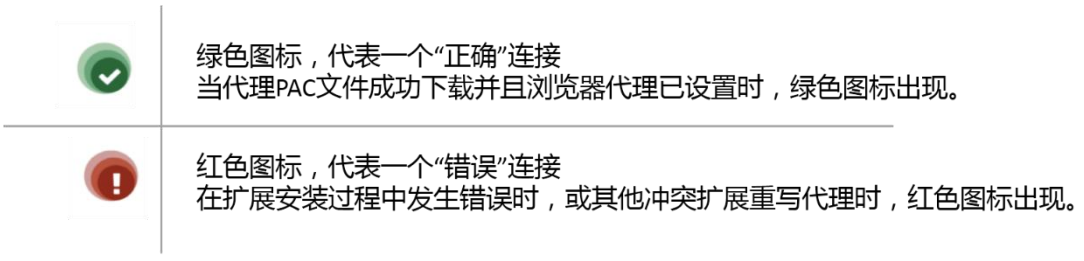

图 2 Chrome 扩展中表示身份认证状态的图标

当出现问题的时候

当出现故障或用户遇到复杂的边界情况时，会发生什么？通过假设用户会遇到的问题，确定常见场景，制定计划尽可能顺利地解决这些问题。让用户能够理解问题，并在可能的情况下自我修复，这是我们一贯的首要目标。

可以自我修复的问题

强制验证门户

因为我们是一家拥有许多差旅员工的全球性公司，当他们在机场、酒店和咖啡馆办公时经常会遇到强制验证门户。这些门户网页通常在私有网络的默认网关上，当用户连接到这样的网络时，BeyondCorp Chrome 扩展程序会尝试下载 PAC 文件，但是强制门户网页会阻拦 PAC 文件的下载。

要解决此问题，当扩展程序检测到网络状态变化时，我们都会确定设备是否位于强制门户之后：通过尝试访问 http://clients3.google.com/generate_204，正常情况下应返回 HTTP 204 的空页面。如果我们收到 HTTP 204 以外的任何内容（最可能出现的是 HTTP 302），就认为该设备需要先通过强制验证门户的认证。这种情况下，Chrome 扩展程序会直接使用内置的预定义 PAC 文件，并警示用户。

当用户碰到强制验证门户的场景，可以点击 Chrome 扩展程序图标，我们会告知他们在连接机场或酒店的网络的时候碰到强制验证门户的这个问题很常见。BeyondCorp 的工作不会受到影响，只需要将 BeyondCorp 的设置更改为“Off:Direct”即可，当用户完成强制门户验证后，浏览器扩展即可成功下载最新 PAC 文件。这个简单的流程允许用户在最短时间内完成自我修复，没有增加技术站的负担。

本地网络设备

用户还经常尝试访问私有网络中的设备，比如许多谷歌员工就会使用公司笔记本电脑来执行配置连接家庭打印机或其他网络设备等任务。但由于

BeyondCorp 配置通过访问代理来路由所有连接，所以启用 BeyondCorp 扩展后，连接就会失败。与强制验证门户的情况类似，解决方案是将 BeyondCorp 设置更改为 Off：Direct。但不同的是，我们无法轻松检测到此故障状态。因为通常情况下，这种场景下的用户有一个激活的并且功能正常的互联网连接，因此从 BeyondCorp 扩展程序的角度来看，一切正常，用户可以通过 BeyondCorp 访问所有的企业资源，没有理由发出警报。

为了弄清楚在这种情况下如何有效地与用户交互，我们进行了一次典型的用户体验测试：工程师把公司笔记本电脑带回家，想用它来更改家里打印机的设置，两台设备通过 IP 地址连接。用户连接到家庭网络，BeyondCorp 扩展成功连接，下载最新的 PAC 文件，并配置浏览器代理。当用户在新建的浏览器 Tab 标签页中输入打印机的 IP 地址时，对私有网络的访问流量一起重定向到了访问代理。网络请求失败，用户得到错误提示。

我们将解决问题的关键点放到最终的错误页面上，并提出了一个解决方案：通过访问代理展示错误页面。我们创建了一个自定义 HTTP 502 错误提示页面，以便在某些场景下将警示信息插入到错误页面中。具体来说，特别是针对用户试图访问 RFC1918 或 RFC6598 约定的地址时，我们返回的 HTTP 502 错误提示页面可以明确给出提示，用户就会知道如果他们在访问本地网络设备如家用路由器或打印机时（两个最常见情况），需要将 BeyondCorp 扩展修改为“Off:Direct”。通过这种方式，我们能够基于现有的基础设施和流程，让用户自行修复问题。

自定义代理设置

我们的海外员工有时需要配置一些自定义代理来测试广告。如果用户安装了多个扩展，每个扩展都试图配置代理，那么这些扩展就会相互冲突，这可能会使用户感到困惑，并影响他们访问企业资源。

我们用两种解决方案来处理这种情况。首先，我们将海外的代理配置直接集成到 BeyondCorp 扩展程序中。当用户有业务需求要从特定位置对外访问时，他们可以从支持国家的下拉菜单中选择该位置。这为用户提供了一个单独扩展来满足他们管理最常见的业务代理服务器的需求。

此外，当用户有合理需求运行额外的代理管理扩展时，他们的 BeyondCorp 图标将从绿色变为红色。然后我们给他们一个选项，将状态更改为 Off：System Alternative 并告知他们何时应该使用此设置。同样，这个过程允许用户进行自我修复，提高他们的工作效率并减少对我们支持团队的咨询的工作量。

复杂问题解决：门户

对于上述简单问题，可以通过自定义错误页面或 Chrome 扩展程序让用户可以快速地进行自我修复。然而对于一些看似正常的访问失败场景，用户和支持团队都会迫切需要知道被拒绝的原因。后端基础设施中的 ACL 逻辑复杂、层级多，无论对用户还是支持团队而言，想要理解这特定决策背后的逻辑都有困难。即使是一个经验丰富的 SRE 工程师，也可能需要花费很多时间查询许多内部服务，来找出一个 403 错误页面的原因。考虑到访问代理每天可能产生的 403 错误页面的数量级（仅 HTTP/S 就有约 12M/天），人工参与故障排除是不可规模化的，也是不切实际的。

为了方便诊断和解决更复杂的 BeyondCorp 访问问题，我们设计了一个门户网站来帮助用户和支持团队。我们不只是用一串通用错误代码来告诉用户他们的访问被拒绝，而是解释他们被拒绝的原因以及如何解决这个问题。门户是独立的，而不是直接集成到访问代理服务器中，因为它使用的是更细粒度的 ACL，取决于最终用户当前信任级别。由于访问代理默认是公开的，所以我们需要限制攻击者从 403 错误页面中获得的信息量。

架构

门户大致分为前端和后端，两者之间采用 API 进行通信。

- 前端是一个交互式 Web 服务。它根据用户的输入内容向后端 API 发出请求。
- 后端可以查询参与访问决策的多个基础设施服务。这个过程会绕开各种缓存层，这样用户就可以接收到最新信息。
- 前端和后端之间的 API 也可以用于其他用途，比如批处理、分析，或者将输出能力嵌入到其他工具中。

解释引擎

除了查询和表示 ACL 外，门户还需要有效地向用户展示这些信息。针对这些被拒请求的响应报文细节，我们构建了一个解释引擎（Explanation Engine）来进行错误诊断。它通过递归遍历负责提供授权决策的子系统来完成操作。

例如，访问代理的 ACL 可能要求设备完全可信才能访问一个特定的 URL。在查询这个 ACL 后，解释引擎会和设备推断管道交互，并获取访问此资源的必要条件，然后将这个信息发送至前端，并翻译成通俗语言，用户就可以通过访问门户来找出他们当前状态存在何种问题以及如何解决这些问题。

为 ACL 定义 ACL

虽然解释引擎可以提供有效信息，但它可能会暴露敏感数据。它会暴露受保护系统存在问题的 ACL，泄露用户账号和设备状态信息，而这些信息都会为潜在攻击者所用。为这些数据定义 ACL 非常棘手，因为我们需要在工具易用性和保护敏感信息之间实现平衡。

根据用户和设备请求故障诊断信息，我们可以使用不太具体的信息替换输出中的敏感信息。在极端的情况下，我们可以将敏感信息替换为联系技术站的提示信息。这样技术站和 SRE 工程师就可以通过验证用户的身份并以用户的名义查看相关信息，在帮助用户的同时不泄露敏感信息。

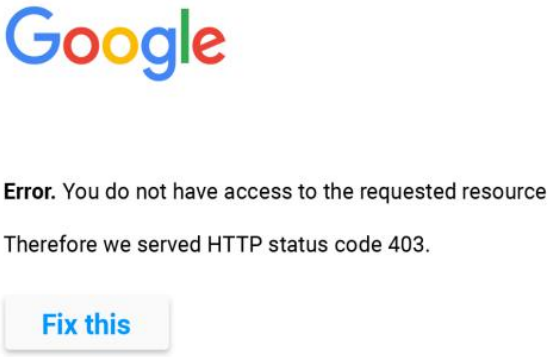

图 3 当 BeyondCorp 阻止请求后展现的错误页面

访问拒绝登录页

一旦门户开发完成，即可将门户集成到访问代理，向用户展示错误消息。当用户遇到 HTTP 403 错误时，他们可以一键返回到门户，查看所有相关错误细节（参见图 3）。然后，门户会向后端重新发送访问请求，并解释导致问题的原因。

例如，如果一个资源要求特定群组成员才可访问，门户会提供群组名和到群组管理系统的超链接，这样用户就可以申请访问权限。门户在后台查询后端的访问控制列表服务来判断该资源的授权要求，与用户当前的归属组信息进行比较，门户前端将比较结果转换为通俗语言（参见图 4）。这一切都发生在几秒钟之内，远远快于用户猜测什么是“组成员问题”或寻求帮助。

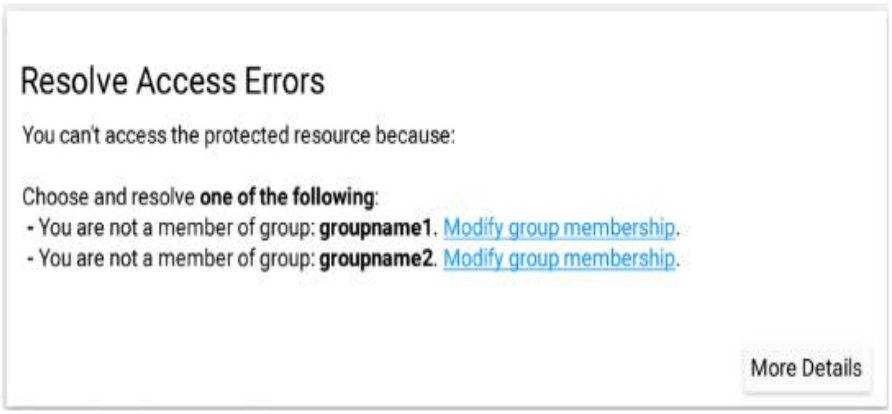

图 4 面向员工的访问拒绝错误故障排除指引

将这个流程直接集成到我们的错误消息中，允许用户可以无缝地完成这个过程——并且，完全通过自助服务。

临时的故障排除

尽管我们期望大多数用户通过错误页面访问到门户，但是我们还提供了一个独立页面来支持更多临时的故障排除。前端门户的登录页面是根据用户身份和访问设备自定义的，它会显示用户及其名下设备的信息，并突出可能导致拒绝访问的问题。我们允许最终用户主动访问这个工具来了解其名下设备的全局视图和潜在访问问题，用户就能一步到位地解决他们任何设备上的问题。去外地出差或者演示之前，使用这个能力进行设备信任度自查非常方便。

支持赋能

门户前端也使技术站能够快速地执行详细的故障诊断，提供立即可执行的方案，大大缩短了解决问题的时间。例如，为了解释一个 403 错误页面，技术人员就可以使用门户登录页面查询特定的用户名或设备标识，锁定到某个特定设备，确认它是否是一个完全可信的公司设备。如果不是，系统会给出设备不可信的具体原因，以及技术人员该如何解决这个问题（参见图 5）。

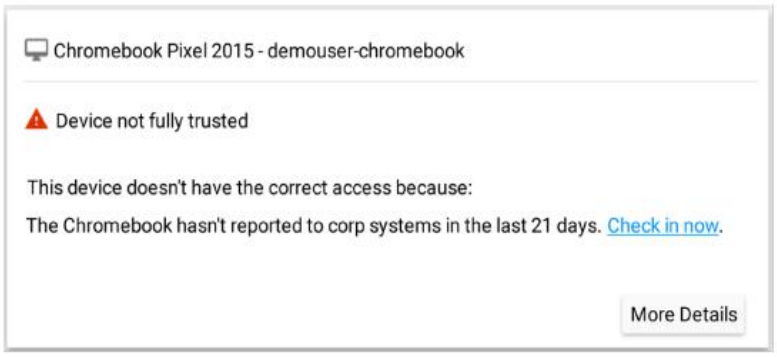

图 5 面向服务台的访问被拒绝错误故障排除指引

未来目标

除了当前的功能外，门户还提供了进一步实现自动化的可能。我们计划在将来持续检查潜在的拒绝访问问题，并会在这些问题真正出现前，告知用户如何自助解决问题的方法。同时，对于不能自我修复的重大问题，会启动自动通知到技术站采取补救措施。我们还希望扩大我们可以自动解决的问题范围，而无需人为干预。

聚焦经验

尽管 BeyondCorp 迁移在多个技术领域困难重重，但它也给予了我们足够的空间可以重新评估用户支持体验。通过关注迁移期间和迁移后的用户体验，我们可以使用户能够轻松地使用复杂的网络模型。专门设计的工具使出现在用户侧的组件变得清晰易用。这些支持界面的意义是为了允许用户尽可能地自我修复，从而节省用户时间，释放出支持团队的资源。当用户需要其他帮助时，我们提供有效工具和信息，使技术站发挥最优价值。

对于绝大多数用户来说，BeyondCorp 是完全透明的。当谷歌员工担心他们自己的业务流程时，BeyondCorp 模型负责处理全部的访问逻辑问题。当用户的确有问题时，我们会快速有效地介入，在合适的时间给他们提供正确的信息协助他们恢复正常。然后我们退回二线，让他们专注于他们最擅长的事情。

参考文献：

[1] B. Osborn, J. McWilliams, B. Beyer, and M. Saltonstall,“BeyondCorp: Design to Deployment at Google,” ;login:, vol.41, no. 1 (Spring 2016), pp. 28–35: https://www.usenix.org/publications/login/spring2016/osborn.

[2] L. Cittadini, B. Spear, B. Beyer, and M. Saltonstall,“BeyondCorp Part III: The Access Proxy,” ;login:, vol. 41,no. 4 (Winter 2016), pp. 28–33: https://www.usenix.org/publications/login/winter2016/cittadini.

[3] J. Peck, B. Beyer, C. Beske, and M. Saltonstall, “Migrating to BeyondCorp: Maintaining Productivity While Improving Security,” ;login:, vol. 42, no.

2 (Summer 2017), pp. 49–55:https://www.usenix.org/publications/login/summer2017/peck.

【第六篇】BeyondCorp：构建健康机群

任何系统的安全等级不可能超过其信任的所有其他系统的安全性。BeyondCorp 项目帮助谷歌基于其信任的计算平台，明确定义并施行访问决策，将安全策略从保护服务转变为保护可信平台。此前发表的 BeyondCorp 系列论文中，详细讨论了谷歌对设备“正本清源”所使用的各类工具，但尚未深入探讨建立设备信任的机制。

我们如此关注计算平台的安全性是有据可依的，业内大量证据已经表明[1]，终端用户已经成为各类攻击的首要目标，并且其攻击手段层出不穷。攻击者通过设计巧妙的社会工程学攻击，将恶意代码安装到设备上，随后即可利用现代操作系统的大量攻击面发起攻击。高级攻击者目的在于利用设备固有的信任、设备的身份凭证、或者已授予用户的信任来进一步利用系统。

为了成功防止这种持续的在可信内容（企业 Web 应用程序、企业凭证）和不可信内容（外部软件库、社交媒体、个人电子邮件等）的混合环境中可能遭受的破坏，平台自身必须具有层次化的、一致的若干控制措施，最终，作为设备机群（fleet）的构成要素，计算平台就成为了新边界。

基于前期工作展开

本文描述的工作内容主要基于白皮书“大规模设备机群管理”[2]和之前发表的 5 篇 BeyondCorp 论文[3]。在这些前期工作的基础上，我们的目标是通过以下方式进一步增强 BeyondCorp 模型：

1. 从通用的控制视角来定义健康设备机群
2. 确保始终如一地全面应用和衡量这些控制措施，并强制执行
3. 基于持续度量推动控制措施的闭环改进

定义待保护环境面临的威胁

与其他安全防护工作类似，我们首先必须定义待保护环境所面临的威胁。在创建威胁列表时，需要考虑一类攻击而不是各种可能的单一攻击向量。攻击者会持续发现新的攻击向量，这意味着很难穷举环境可能面临的所有威胁。但是，如果成功减少了某一类攻击，即可减少对这一类威胁中的多数攻击向量[4]。

从一个较高的层面来看，平台应该考虑的威胁类别包括：

1. 未知设备：由未知的或非受控设备对敏感系统发起的访问；
2. 平台失陷：利用计算平台上操作系统或软件的错误配置；
3. 安全控制旁路：未生效或错误配置的安全策略导致的系统失陷；
4. 非法提权：通过代码的执行导致系统特权被接管，且持续控制；
5. 软件失陷：恶意软件安装和持续存在；
6. 持续攻击：由于缺乏检测而导致攻击者长期攻击；
7. 认证旁路：通过被盗密码或绕过认证机制，造成平台失陷；
8. 数据泄漏：对磁盘、内存或传输中的敏感数据发起未授权访问；
9. 攻击隐蔽：由于缺乏日志和监控手段导致攻击者长期持续存在；
10. 攻击抵赖：攻击者通过掩盖其攻击痕迹而妨碍取证分析；

通过改善设备机群健康来解决环境威胁

定义威胁后，就可以更好地识别出缓解这些威胁所需的控制措施。并且进一步在服务访问时对这些控制措施的状态进行度量（有效性、是否开启等）。表 1 列出了上述各种威胁类别对应的控制措施，对一个理想的可信平台来说，这些措施是必不可少的。

#	威胁	控制
1	未知设备	设备机群清单库和资产管理
2	平台失陷	操作系统&基础软件配置管理
3	安全控制旁路	安全策略管理&强制执行
4	非法提权	防止系统接管
5	软件失陷	软件控制和反恶意软件
6	持续攻击	可远程验证平台状态
7	认证旁路	对平台和用户的可靠认证
8	数据泄漏	数据保护
9	攻击隐蔽	基于日志的检测能力
10	攻击抵赖	平台响应能力/检测&响应

表 1 威胁种类和潜在缓解机制

健康设备的特征

健康的设备机群由健康的设备组成，这些设备的健康由工具、流程和设备健康维护团队提供保障。设备如满足以下条件，即可认为是健康的：

- 可以承受大部分攻击；
- 提供了足够的检测和度量能力，在出现问题时能及时止损；

下面我们将深入研究，为什么这些控制手段至关重要。

设备机群清单库和资产管理

硬件是操作系统和应用程序运行的基础。通过限制硬件配置的差异性，可以更有效地判断出设备机群中设备的能力和不足。设备清单系统通过设备访问开通机制，为可访问敏感系统的设备范围进行了明确界定。

操作系统&软件配置管理

软件管理是保证设备机群健康的关键组件。一个集中式的软件管理基础设施有利于保证平台配置的一致性，以确保可信平台满足以下条件：

- 默认是安全的，并且随着时间的推移仍能保持最小偏离
- 能持续地进行安全的升级

为正在运行的操作系统、敏感的软件栈和安全代理打补丁的能力对于健康的安全态势至关重要，软件配置的管理也同样不可或缺（如软件自动更新策略）。

安全策略强制执行

可信平台应始终如一执行安全策略，并且报告和记录与预期策略之间的偏差。安全策略通常与上面提到的操作系统管理和配置策略交织在一起，但安全策略是独一无二的，是用户无法规避的强制访问控制策略。例如，通过同时登录限制策略可以减少横向移动的威胁；默认禁用 root 权限，有助于缓解恶意进程可能造成的危害。

防止系统接管

通过叠加防护层，确保恶意软件无法破坏系统的安全性。在高级恶意软件屏蔽掉主机日志子系统之前，确保主机可以报告异常行为。

软件完整性及控制

可以限制未授权代码在平台上执行。常用策略包括，仅允许已知的“好”软件运行，明确阻止可疑 “坏”软件运行。我们通常会选择白名单策略，因为明确定义出工作所需的所有应用程序是可行的，但需要阻止的所有潜在坏人或恶意软件却无穷无尽，无法枚举出来。

可远程验证平台的状态

平台应具备基于密码学的完整性验证机制，在底层平台上提供从固件到运行的操作系统的完整性保证。可行的机制包括：第一命令执行控制[5]、安全启动和远程验证。

平台和用户的可靠认证

在尽可能的情况下，用户账密信息存储应使用系统上的硬件支持或硬件隔离。Windows Defender Credential Guard[6]就是一个很好的例子。

数据保护

我们假设每位用户的系统都有一些敏感数据，因此敏感数据在存储和传输过程中都应该被加密。为了应对设备丢失或被盗的情况，设备应支持远程数据擦除功能，可以销毁一切存储在系统上的数据和长期凭据。

基于日志记录和日志收集进行威胁检测

为了提供纵深防御，平台威胁模型应该假设攻击者能够绕过预设控制措施并且设备存在攻破的可能性。为了缓解此类风险，平台需记录这类事件。日志应该记录发起操作的用户和设备的相关属性，对所有敏感数据的访问或修改，包括对平台安全控制机制、状态和行为的更改都应该详细记录。这些信息应该输出到一个集中式日志记录工具，理想的日志记录策略必须能够阻止未授权进程对其篡改。

平台响应能力/检测和响应

如果检测到威胁，平台应该提供有效手段，让得到授权的入侵分析师可以进行远程事件响应。诸如 GRR 之类的工具可以提供远程访问能力来执行这类分析[7]。人工检测工作应尽可能保持在最低限度，因为它无法规模化地应对广泛的攻击行为。理想情况下，授权分析师应该能够创建用于调查取证的有效时间线，可以从受影响系统中一次性的拉取数据来进一步调查，通过复现事件，检测和响应团队可以全面了解发生的情况并做出相应的响应。

维护健康的设备机群

具有上述控制特征的一组客户端设备构成了一般意义上健康安全的设备机群。为了达到这个状态，首先要弄清楚如何一步一步地建立平台信任。

建立信任

敏感的服务只能通过可信设备访问，我们为系统信任划分等级，基于特征和行为，设备能够获得不同的信任等级 [8]。

然而，这种方法带来了“鸡生蛋蛋生鸡”的问题：将设备转化为可信状态，需要首先访问客户端软件库，而客户端软件库本身就是一个敏感系统。为了解决这个问题，在设备从不可信到可信的迁移过程中引入了“已识别”状态。状态为“已识别”的设备是指那些清单库认为信誉良好但由于某种原因还未取得信任的设备。可以允许这些设备访问客户端软件库的某个子集，以便安装补救软件。补救软件使得机器能够报告设备状态、下载和安装所需补丁，并采取所有必要步骤来满足对于可信平台的要求。

随着健康设备机群构建工作的开展，就会更加了解自身环境，就会更有信心地开通访问权限。另一个挑战就是需要确保技术和业务的不断演化过程中，信任能够持续得到保证。下文将讨论随着技术和业务的演进如何保持设备机群良好的健康状态，以及如何在健康状况恶化时迅速纠正。

对抗设备熵

一旦设备发放到用户手中，其安全性会逐步减弱，因为随着时间推移，安全性难以避免的会衰减。在对抗设备熵的过程中，我们发现了一些有用的策略。

第一条策略，也是最强大的策略，即将访问决策与清单系统集成。在获得授权内部资源的访问权限前，所有的设备都应在列且被信任。对机群的每一台设备，在接收和镜像的过程中，确保其信息都添加到公司清单库。对于任何上报为失踪、

被盗或丢失的设备,应立即删除其访问权限。为了保证用户能及时报告设备丢失或被盗，要求用户必须在收到新的替换设备之前进行主动报告。

对访问环境的任何设备状态变化采用严密的监测和度量手段非常重要。Facebook 的 OSQuery[9]是一个优秀的开源监测工具，适用于 Linux、OS X 和 Windows 系统：它可以监测设备属性，如设备的操作系统版本、关键软件的补丁级别和加密状态。

另外，补丁和配置管理工具[10]能够改变设备的状态，将不可信的设备转换为可信的设备。BeyondCorp 通过限制用户访问的方式，强制用户进行某些必要操作，例如重新启动或接受更新。

检测不健康的主机

在主机的整个生命周期中，某些操作或不作为都可能会导致设备转变到不健康状态。信任推断引擎[11]通过持续信任评估来检测设备状态变化。当设备不能满足信任标准，则将设备的信任状态降低为“已识别”，通知机器的所有者并为其提供设备的修复指导。

检测与响应团队可以为信任决策提供额外的信息，这个团队有权删除对任何恶意设备的信任。

提供灵活的策略

乍看起来，设备机群健康状况的定义是一项简单的任务。但是，和大多数 IT 环境一样，魔鬼总是隐藏于细节（和例外）之中。在处理大量不同的操作系统和各种用例时，会遇到许多这样的细节问题。

为设备机群上线控制措施时，我们会尝试为策略的合规性设置一些阈值，而不是一上来就提出绝对严格的要求。这种策略允许用户在良好的状态下更灵活地运作，并能避免使用会让用户崩溃的严格规则集（这会导致用户寻求规避手段）。例如，如果用户需要安装非关键补丁，我们会在降低其访问权限前给予一个宽限期。

同时，通过一些防范控制措施为事件监测和响应功能提供信号也非常重要。为此，我们努力将这些控制措施集成到安全信息和事件管理管道中，以便可以报告和记录策略相关数据。捕获访问有关的数据可以帮助后续的调查取证和事件检测， 这些数据可能包括什么时候允许访问，以及根据策略，什么时候对访问进行了拒绝。

试点并推广这些原则

这是一个典型的由安全团队及其合作伙伴主导的项目，从开发到上线，始于设计和原型阶段，小规模测试，从设备机群和用户那儿搜集反馈并逐步完善。我们逐渐达成一项策略，即首先在监控模式下推出控制措施并建立内部试用团队（Dogfood）[12]以方便调试。例如，我们可能会推送一个新的 USB 审计代理到部分硬件工程师组织，因为这部分人员经常与自定义 USB 组件交互。通过这种方式，发现在大样本量的情况下，边缘情况并不会集中涌现。还有种方案，按照地理位置来划分内部测试，当然必须在变更实施前准备好本地支持人员。

在新控制措施上线时，清晰的沟通有助于了解新政策及其存在的原因。将每个控制措施映射到它所解决的威胁可以帮助每个人理解安全团队选择特定操作的原因。高度透明和对标准的清晰解释帮助我们加深了与用户间的理解，建立了与干系人的共识。当他们看到我们并没有任何隐藏的目标或动机时，就会充分参与到这一愿景及目标的规划中。一般来说，负责这种安全驱动的变革团队可以从清晰的全局目标阐述中受益，使得每项行动师出有名，更有利于争取到合作伙伴团队的支持，这种支持会形成一个良性循环，为“如何使设备机群更安全”带来更好的闭环。

平台度量和一致性控制

一旦定义了预期效果的基线，会发现某些控制措施无法普遍应用，因为无论是设备本身还是管理/策略层面，平台的能力都参差不齐（有时甚至能力差别很大）。例如，Chrome 操作系统的 Secure Access 提供了可靠的软件控制，但是

Linux 却没有开箱即用的防恶意软件的能力。为了确保整个设备机群获得一致的安全性，需要对安全评估进行规范和统一。虽然希望在不同平台上 100%实现完全相同的控制效果是不可能的（因为平台能力和威胁模型不同），但也并非绝对，对于所有需要实施的控制，可以将评估标准统一为“对安全风险是否有效”。

为了完成统一评估，分析了所有相关平台目前在满足控制效果方面的现状。然后，评估了现状和理想差距的总体情况。为谷歌管理的每个平台都创建了总体设备机群健康报告——这不是“成绩单”，而是对其能力的分析材料。针对每个平台，都需评估以下方面：

- 平台是否能够支持控制措施？
- 控制措施是否默认开启？
- 控制措施的状态是否可以度量？
- 设备机群是否合规？

要推动各项目标的可度量行和可比较性，可能需要考虑：

- 将这些策略统一到标准的度量单位中：如，自补丁发布以来的时间，地理位置，数量等
- 从相对量的角度来推动度量的标准化：如，和当前版本的偏离量、功能特性的实现比例等

难点在于如何设置这些评估标准。一旦拥有了各平台的可比较的度量标准，探讨设备机群健康状况的能力将大大提高。

当防护控制措施无法生效或者仅部分生效时，可以寻找其他的方式来消除风险，例如，更全面的监控/检测在某些平台下可以作为防护控制措施的补偿。这种评估方式可能看起来有点主观，有点依赖于对平台抵御攻击能力的整体感觉。

现代操作系统有非常复杂的攻击面、能力和威胁模型；我们发现聚合所有这些信息的最佳方式，仍然要采用手动比较设备所需特性与实际具备的特性。这种比较允许我们能够围绕项目提出顶层建议，以填补缺失并提高项目的优先级。无论是基于什么数据得出的结论，重点是必须记录下这些结论背后的缘由，或至少记录下产生结论的过程，这能让应急安全工程师之外的人了解设备机群的状态。

与理想情况的偏差

尽管在定义、上线、测量和强制执行控制上，我们都做出了最大努力，但仍不可避免地需要面对这样一个严峻的现实：要部署 100%统一的控制措施过于理想，这种理想也许仅存在于独角兽自由玩耍的上古神秘国度，那里没有恶意软件，没有国家级的网络攻击者。现实是残酷的，我们需要针对偏差制定计划、进行根因分析、妥善处理例外情况。

许多偏差的发生是在所难免的，包括流程中断、管理工具故障、不稳定的版本发布或其他各类原因，都可能造成偏差。例如，为系统打上最新补丁总是会有延迟的。重要的是要了解什么时候需要对设备机群全范围内的例外情况进行处理，需要防止异常规模的增长，但不要总是通过控制措施来强硬纠偏。如果懂得如何在威胁模型和用户影响之间权衡，就不难做出正确的决策。

例外情况应该可以被度量并且有明确的时间窗口限制。我们建议在整个设备机群范围采用一致的方式对根因进行分类，以便了解当前差距，并明确出哪些控制措施不适用于某些设备机群或某类用户。如果例外在不断更新（又或者永不过期），控制手段就会失效。这时应重新设计控制措施甚至重新审视这项控制措施在设备机群中的角色。

启动

那么，如何将本文讨论的 BeyondCorp 原则应用到你的设备机群呢？一般包含以下 4 个步骤：

1. 定义需要重点考虑的安全控制措施；

2. 找到度量这些控制措施的方法；

3. 判断设备机群的不合规项；

4. 修改工作流，让其符合预定安全策略，或将其定义为例外。

第一步十分关键，目的是为了确定想要实现的目标。我们不应毫无根据地开始创建安全控制，而是应该明确这些控制是针对某项威胁的具体应对机制。明确列举出威胁列表，不仅能够为度量控制的有效性提供启发，还能为梳理每个特性的优先级提供一个框架。当为这些特性进行定义和排序时，需要咨询合作伙伴的意见（详见下文“经验教训”）。当已经明确威胁及缓解这些威胁的控制措施时，就可以通过测试来评估这些控制措施的有效性，包括单元测试、端到端的红队渗透测试等。基于这些举措，可以进一步明确这些控制是否有助于在实践中达到安全目标。

为了持续确定设备的安全态势，必须能够对其当前状态与理想状态进行监测和度量。如果尚不能实现度量，需要在设备机群安装度量监测软件来收集相关数据。不过，即便获得了原始监测数据也才只完成了一半，我们还需要定义待测量设备的理想状态。由于设备机群所包含的设备多种多样，需要定义多个理想状态，尽可能覆盖所有潜在的用例。

一旦可以度量设备机群的安全态势，就可以开始检查设备实际状态与理想状态的偏差。一些偏差可能不会带来安全风险（因为它们可以通过补偿控制来缓解），但仍有许多偏差将会暴露风险。一开始，我们就需要确保新设备从员工使用它们的第一时间起就符合控制要求。确保所有新设备都能从一个已知的良好状态开始加入机群，这样我们就可以将注意力放在设备机群中的其他设备上来提高设备机群的整体健康状态。

建立一个例外框架，这样在执行一个重要的新控制措施时，就可以先为现有的设备机群创建例外。此时整个设备机群的偏差将保持静止不变，于是就可以保

持新设备符合要求的同时逐步修复现有设备。一旦将问题明确隔离到设备机群的例外集合中，就可以将故障原因进行分类，就能够发现整个设备机群或工作流所共有的一些问题。首先解决其中规模最大、风险最大的问题，以最小的代价获得最大的安全性。重复这样的分类及修复过程，直到已经解决了设备机群中的所有问题。如果用户的工作流与所需安全特性明显不兼容，这种情况可以作为例外特殊处理。

虽然这个系统需要许多不同团队的大量协作和努力工作，但完成这项工作可以提升我们在面对持续攻击时的灵活性。

经验教训

制定一个连贯的计划来衡量和评估信任和设备机群的健康状况并不是一个短期项目。完全达成本文所描述的目标（以及 BeyondCorp 的其他目标）需要许多重要资源。因此，希望谷歌在过去几年中学到的一些经验教训可以为读者们节省一些时间和麻烦。

尽早设置目标里程碑

尽早设定关键目标里程碑。确定重要资产并对它们进行（至少粗略的）排名。这样有助于有效地分配资源，并为实施大型项目提供合理的动机。将设备机群管理系统的数据整合到授权决策过程中，是一个很好的初始里程碑。仅此一项就可以防止未知设备访问关键服务，同时还能生成一份已知良好设备清单。

确定如何处理例外

在项目中尽早定义例外处理框架。每个设备机群中都包含那些无法完全符合理想安全状态的设备。确定例外管理的流程和技术实现是成功部署的关键。定义允许创建例外的各种场景及其理由，记录这些场景，确定允许该例外的最大时间窗口，梳理已有例外的审核过程。

与合作伙伴互动并且尽早影响其他团队

BeyondCorp 的成功实施需要整个 IT 部门的配合。尽早与合作伙伴和可能受影响的团队展开合作，这样会大大提升后续实施上线的顺畅性 。

例如：

- 设备采购和上线团队需要确保在设备加入或退出设备机群时，设备机群管理系统进行更新，保持最新状态。

- 其他安全团队在定义设备的安全属性时可以提供有价值的输入，来自安全团队的各种输入对整个项目的价值重大。

- 传统的 IT 支持团队将负责绝大多数用户升级。他们必须了解项目的目标，并能够帮助解决用户问题。

同时，我们也需要知道如何与那些会受到影响到的用户进行沟通，确保普通用户能够真正遵循并完成自我修复过程，减少故障排除时间，减少 IT 负担。

结论

保护员工设备安全是保护企业关键信息资产安全的基石。为此，我们彻底评估并定期检查所有企业设备来确保其健康状况，只有已知安全设备可以访问关键的内部系统和信息。

员工及其设备已经引起了坏人的注意，因此我们有义务在保证其安全性的同时维持生产力。为了达成这一目标，需要强烈的设备机群健康意识，明确的策略和衡量标准，以及处理目标偏差的流程。通过一致的控制和强制执行，每个企业都可以提升设备机群的健康和安全性，提高对不断增加的各类攻击和威胁的防御能力。

致谢

BeyondCorp 仍然是谷歌当前一项大型的跨团队项目，该项目有很多贡献者，在此我们想要特别感谢赛勒斯·威苏那（Cyrus Vesuna）在协助完成定义平台通用可信控制方面的工作。

参考文献：

[1] Executive Summary": https://www.verizonenterprise.com/resources/reports/rp_DBIR_2018_Report_execsummary_en_xg.pdf; Mandiant, M-Trends 2018: https://www.fireeye.com/content/dam/collateral/en/mtrends-2018.pdf.

[2] Google, "Fleet Management at Scale," November 2017:https://services.google.com/fh/files/misc/fleet_management_at_scale_white_paper.pdf.

[3] https://cloud.google.com/beyondcorp/#researchPapers.

[4] New variants often stretch the common understanding of classes of attacks, so you can't ignore variants completely. For instance, the industry thought we had a good grasp on microarchitecture security up until 2018—see Jann Horn, Project Zero (Google), "Reading Privileged Memory with a Side-Channel," January 3, 2018: https://googleprojectzero.blogspot.com/2018/01/reading-privileged-memory-with-side.html.

[5] Such as Intel's Boot Guard: https://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/4th-gen-core-family-mobile-brief.pdf.

[6] Microsoft's Defender Credential Guard: https://docs.microsoft.com/enus/windows/security/identity-protection/credential-guard/credential-guard.

[7] https://github.com/google/grr.

[8] For a description of trust levels and calculation, see B. Osborn, J. McWilliams, B. Beyer, M. Saltonstall, "BeyondCorp: Design to Deployment at Google": https://ai.google/research/pubs/pub44860.

[9] https://osquery.io/.

[10] For more on the tools we use at Google, see “Fleet Management at Scale: How Google Manages a Quarter Million Computers Securely and Efficiently”: https://ai.google/research/pubs/pub46587.

[11] For more on the trust inference system and the other moving parts of our BeyondCorp model, see B. Osborn, J. McWilliams, B. Beyer, M. Saltonstall, “BeyondCorp: Design to Deployment at Google”: https://ai.google/research/pubs/pub44860.

[12] Dogfood: early release of products to employees to get feedback and catch bugs before a wider release.