1. xxx
2. 为什么选择这个题目，权限系统上至云平台、人工智能、大数据系统，下至文件系统、Unix系统，几乎无所不包，可以说我们在做软件开发这个行当几乎人人都离不开权限系统。
3. Xxx
4. 这些问题的实现都相当的复杂。
5. 鉴权，时间、权限、口令等等

我们今天的主题主要围绕 构建灵活、可信的权限控制体系，也就是说如何建立合理的权限模型，说到建模 （分析模型、关键信息提取、核心要点提取，抽象出简单模型然后分析推演出完备的模型），但这里权限模型它相对开放问题来说比较固定，基本上已经有了一些较为完善的模型方案，下面就对这些模型进行讨论。

提问：大家知道哪些权限模型？

1. Xxx
2. Xxx
3. 提问：ACL这个层次的模型我们在那些系统里面经常会用到？
4. MAC 的优势就是实现资源与主体的双重验证，确保资源的交叉隔离，提高安全性。
5. 与 ACL 模型相比，不需要向每一个用户配置权限，基于角色授权，角色分类：部门/地点/工作岗位等
6. 1 解释模型， 然后看时间样例，充实RBAC

RBAC2 职责分离 （动态、静态（互斥、数量限制、先决条件））

提问：我们在开发软件系统时，通常在权限判断时会遇到越权漏洞，有谁能答出来这种越权漏洞的名字和其特点危害。

提问：watch

1. 提问：第一题
2. 1. 用户组 ，user-group， 同一类型数据权限控制 role-resource

2. （1）通过group组，并不能完全解决，让业务系统无感知的切换权限

（2）超前设计，加入一个domain 表，来限制角色生效范围，role-group

（3）打补丁，加入补充规则曾，拿到所有授权和限制权限后，在对权限进行补充

（4）几乎很难用RBAC实现一个灵活可用的权限系统（通过拼接权限角色SQL实现）

用户集合授权，动态授权、临时授权

1. 用户多时， 角色膨胀 -> 管理人 变成了 管理角色， group组的概念 ，按照组授权

2. 动态授权，按照动态用户集合授权

1. 目前ABAC 的实现方式很多，很多云厂商（如：亚马逊的IAM \ 阿里云的RAM \腾讯的CAM）都有自己的一套实现方式，但是大体的概念都是围绕 主体、客体、策略、环境、属性等建立起来的。
2. Xxx
3. Xxx
4. Xxx
5. 看实践样例......ABAC,

其实上面只是列举了Abac定义过程中通用的一些方式，在具体的实现时是更加灵活的，核心是根据定义的访问规则及策略进行调整。

基于属性，ABAC可以设置很多灵活的策略来进行访问的控制，比如：

**1.当一个文档的所属部门跟用户的部门相同时，用户可以访问这个文档；**

**2.当用户是一个文档的额拥有者并且文档的状态是草稿，用户可以编辑这个文档；**

**3.早上九点前禁止A部门的人访问B系统；**

**4.在除了上海以外的地方禁止以管理员身份访问A系统。**

1. 需要零信任理由一

IT架构越来越无边界化

随着云计算和SaaS化的流行

企业的工作负载可能部署在任何位置传统意义上的「网络边界」，不存在了只要业务所在之处，都需要被保护

1. 零信任最初是Forrester Research分析师John Kindervag于2010年提出，他认为所有的网络流量都必须是不可信的。

2. 大约在同一时间，谷歌开始开发自己的零信任系统，其创建了 BeyondCorp，用于将传统的虚拟专用网络 (VPN) 访问策略迁移到新的基础设施，在该基础设施中没有系统是可信的

3. 360 针对银行、金融等多个行业发布了零信任安全架构产品

4. 阿里、腾讯云等云厂商，针对云服务发布了多种零信任系统模型

1. 传统的网络安全架构是基于信任的内网安全模型，而外部的网络是不受信任的。这种架构依赖防火墙和其他安全措施来保护内网。然而，随着云计算和移动办公技术的进步，内网的边界变得更加模糊，内部的用户和设备也失去了原有的信任基础。此外，传统架构的安全性依赖于边缘设备的可靠性，且仅能提供简单的身份验证，无法有效防范如密码破解、伪造MAC地址等高级攻击手段。同时，传统架构设计复杂，难以全面实施访问控制，难以满足现代企业的需求。

小故事，员工vpn、木马，造成了google内部大量的服务器瘫痪，

1. 1.不信任任何一台网络设备或用户

2. 管控所有的访问流量，机器、服务、api接口、数据

3. 严格的身份认证，2FA（Two-factor authentication）（双因素认证） 密码+短信 密码+令牌

4. 良好的扩展性，不需要重新设计现有的权限系统能够适应基础设施的变化

基于零信任架构，遵循零信任核心理念。包含环境感知具备以身份为基础、最小权限访问、业务隐藏、终端检测评估、动态授权控制等几大核心能力。

1. 1. 场景模型较为固定，可扩展性不强，对于数据权限基本都不支持

2. 仅有AWS的IAM 身份识别控制管理系统，通过标签方式实现，功能强大，但是管理性和易用性较差

3. ABAC基于属性的权限控制，ACL和RBAC的模型都可以看做ABAC的特殊情况，并且对于权限控制也非常灵活，一些细颗粒的权限也可以通过ABAC进行设置，但在可理解性和易用性上较差

3. 即在RBAC易用\易理解的性上，添加了ABAC强大的控制能力，国内的大多数厂商都使用这种模式，这种设计同单ABAC相似，在设计初期就需要规划清楚每个模型管控的范围和内容，非常容易混淆权限控制的边界，最后变成一地鸡毛。

1. 总结， 分享了权限控制模型，从 列表控制 到 基于角色的权限控制，再到基于属性的权限控制（基于策略的权限控制），权限模型的发展从粗放，控制文件控制系统，到精细 控制数据，从RBAC开始模型的复杂度和不易理解的程度也在逐步的加高，在未来一个复杂的系统也必然会沿着这个方向发展，通过发现用户行为中的风险因子进行权限的控制（授权、警告或者拒绝），所以从未来看AI大模型于权限系统的结合，通过训练出专业的AI安全权限大模型和ABAC结合，将是一个可以看得到的未来。

4.Watch的意思是什么？你在哪里见到过？

[Watch机制](https://www.baidu.com/s?rsv_idx=1&wd=Watch%E6%9C%BA%E5%88%B6&fenlei=256&usm=1&ie=utf-8&rsv_pq=aa0b76ab000fa39d&oq=Watch%20%E6%9C%BA%E5%88%B6&rsv_t=3714xTN0PUohf+yvWDkjgVfn0hEs8wZ4nfZFyVXokEhT5h/MSvA/r+w/KjE&sa=re_dqa_zy&icon=1)是用于[分布式系统](https://www.baidu.com/s?rsv_idx=1&wd=%E5%88%86%E5%B8%83%E5%BC%8F%E7%B3%BB%E7%BB%9F&fenlei=256&usm=1&ie=utf-8&rsv_pq=aa0b76ab000fa39d&oq=Watch%20%E6%9C%BA%E5%88%B6&rsv_t=c843UJGxTEBHNJAf1ZsH1qn3HZsaJAFtfhg/38ASDY0mDVpMibFCM6xc5Ww&sa=re_dqa_zy&icon=1)中的一种机制，它允许应用程序监控特定数据的变化。这种机制通常涉及以下几个方面：

连接层：这是客户端和服务器的通信基础，涉及到TCP连接的建立和管理。

HTTP长轮询：这是一种常见的watch方法，它允许客户端以轮询的方式从服务器接收更新的事件。

长连接模式：这种方式在一个连接上可以处理多个事件，从而减少了许多不必要的开销。

事件流序列化和反序列化：这是在长连接模式下必须解决的问题，包括如何将事件从一个格式转换为另一个格式。

此外，Watch机制还常用于其他场景，如分布式锁。在这种情况下，Watch机制可以被看作是在分布式环境中为共享资源设置的一种监视器，确保只有一个进程在同一时刻访问资源。当资源的控制状态发生改变时，相关客户端会被通知，以便采取相应的行动。

在[ZooKeeper](https://www.baidu.com/s?rsv_idx=1&wd=ZooKeeper&fenlei=256&usm=1&ie=utf-8&rsv_pq=aa0b76ab000fa39d&oq=Watch%20%E6%9C%BA%E5%88%B6&rsv_t=c843UJGxTEBHNJAf1ZsH1qn3HZsaJAFtfhg/38ASDY0mDVpMibFCM6xc5Ww&sa=re_dqa_zy&icon=1" \t "https://www.baidu.com/_self)（一种分布式协调服务）中，Watch机制的具体实现是通过在特定[znode](https://www.baidu.com/s?rsv_idx=1&wd=znode&fenlei=256&usm=1&ie=utf-8&rsv_pq=aa0b76ab000fa39d&oq=Watch%20%E6%9C%BA%E5%88%B6&rsv_t=c843UJGxTEBHNJAf1ZsH1qn3HZsaJAFtfhg/38ASDY0mDVpMibFCM6xc5Ww&sa=re_dqa_zy&icon=1" \t "https://www.baidu.com/_self)上注册触发器来实现的。当znode的状态发生改变时，即调用create、delete或setData方法时，会触发与Watch相关的操作。客户端可以通过设置Watch参数来指示其希望监听此类事件。一旦有事件发生，客户端就会接收到异步通知，从而能够在适当的时间点做出响应。

总结来说，Watch机制是一种在分布式系统中用于监视数据变化的方法，它可以应用于多种场景，包括但不限于分布式锁和状态跟踪。