Secure Reliable Storage with replication on SGX

Introduction

公司的数据有可能因为黑客的入侵（拖库攻击），管理员的错误配置或者内部员工的倒卖，导致信息的泄露。另一方面，Cloud中用户的敏感数据也不希望能够被Cloud中任何人能够看到。同时，传统的Replication system 只能够保证机器意外的Crash不会导致数据的丢失，并不能提供对攻击者恶意的操作Crash的防护。为了提高系统的可靠性和安全性，我们提出一个practical 的方法：Replication system on SGX，利用基于SGX的Replication来提高系统的可靠性。(Motivation)

（此处加上SGX简单介绍）
系统中的所有的数据和代码在加载进Enclave之前都是明文的。系统的算法也都是公开的。进入Enclave之后，系统的数据对外界是加密的，系统依赖于CPU提供的秘钥和随机生成的Random进行工作，外界无法获得该秘钥或者猜测出芯片提供的Random生成函数。

key idea是通过SGX提供的加密运算功能，在几千台机器中隐藏机器之间的关系，以此来实现更高等级的reliability，以及Rollback attack protection.

在这个系统中，可以防范小范围的Denial of service 攻击，即少数机器Crash无法影响整个系统的availability。如果要让整个系统无法工作，则需要同时将大部分机器上的Enclave 关闭。

机器Crash和Enclave重启之间的概率大小是不一样的。

单机系统上，该系统无法防范单个平台上的side-channel攻击如针对Cache或计算能力的攻击。但是对多个机器之间的交互如网络流量的side-channel channel攻击可以进行简单的防护。

Challenge:

Arrange Nodes：
Hide network traffic：
Delete Operation：
Consist：

Contribution

我们工作的贡献包括：

replication系统中加入SGX， 将系统的配置管理信息隐藏，practical的提高replication 系统的security 和reliability。之前的replication系统无法防止攻击者对信息的盗取和破坏。例如攻击者可以将整个数据库文件下载后解压，现在的replication 系统 中的数据默认是加密的。之前攻击者随意的删除某些文件或者进行Rollback attack/replay attack，现在除非攻击者对系统中绝大多数的机器进行破坏，否则无法破坏系统或者文件。

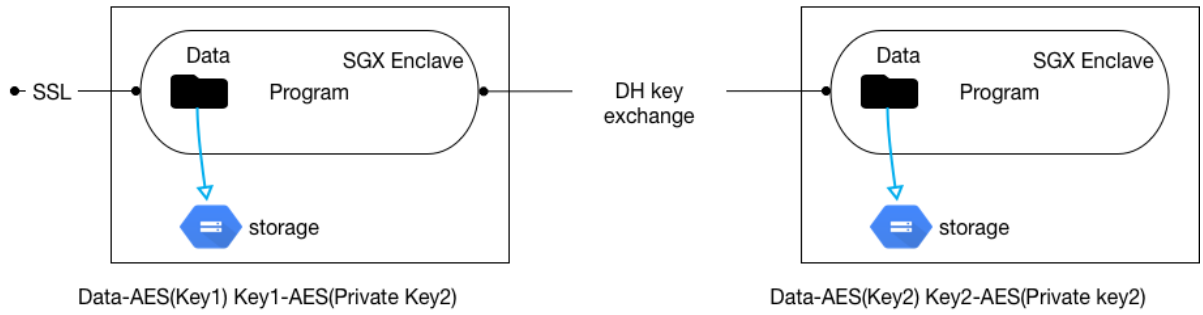
探索SGX在分布式存储领域的应用。单机的SGX只能维护runtime的安全，一旦Enclave重启SGX上的状态都消失了。根据Intel SGX SDK，单机的SGX可以通过Monotonic Counter来维护一个持久的状态，但是这个monotonic counter：1 依赖于ME，安全性没有documented 2 性能较差 3 编程模型复杂（引用ROTE论文）。我们借助于SGX能够隐藏信息的特性，结合分布式的consensus和SGX之间的相互assist, 实现一个更为简便的以object/files 为单位的counter（有多少个操作的对象，就可以维护多少个counter）counter之间互相可以并行化因此可以极大地提高系统的运行效率。

System Design：

Thread Model: 攻击者或者内部人员可以获得整个软件栈的权限，甚至 OS 和硬件。攻击者可以在任意时间终止 Enclave 的执行，但是无法获得 Enclave 内部的信息。但是我们仍然假设设施的所有者是好的，他希望他提供的服务可以继续使用。因此，攻击者不能 crash 整个系统引起注意。另外就算他 Crash 所有的 Enclave，也不能让他获得任何的好处。因此我们的系统可以称为 practical 的，理论上无法达到最优，但是实际工作中效果不错。

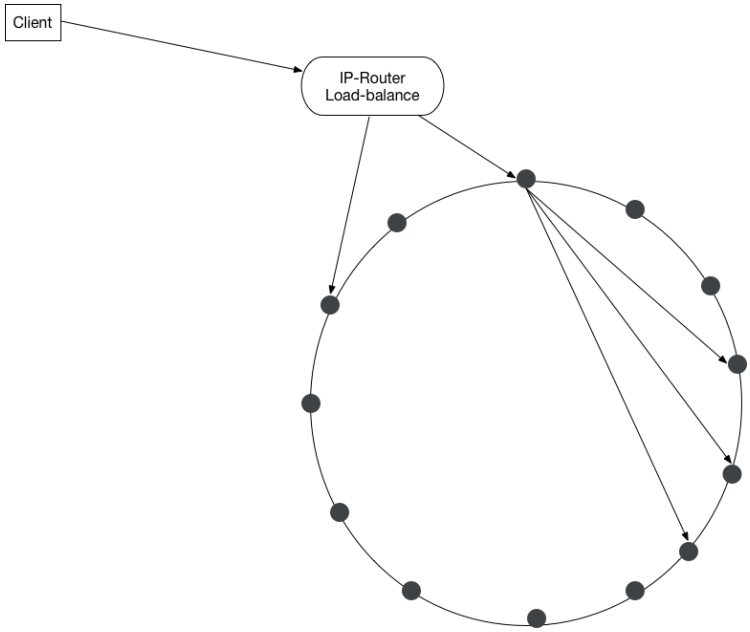
System Model:

The system consists of a group of N nodes. Each Node has SGX feature enabled and Bios configured.



Picture：Data Flow in system.

节点之间的数据流动如上图所示，由远端的数据通过 SSL 加密信道后传入 SGX 中，由 SGX 内部的 Enclave 将 SSL 解密。Encalve 之间通过 remote attestation 建立 Diffie–Hellman 密钥交换协议的安全通信，进行数据交换。当数据传入 Enclave 并且决定要存在本地时，Enclave 会随机生成一个 key，并且用这个 key 加密 Data，并将加密后的 Data 拷贝出 enclave 并由 host 存入 disk，而后用 SGX 提供的 sealing key 加密随机生成的 key 并存入本地。



Picture：系统拓扑图

系统之间的节点采用 Distributed Hash table（DHT）来管理，在 SGX 内部维护一个 Zookeeper 协议，记录当前节点的状态。每个物理节点对应一些虚拟节点。每个虚拟节点拥有一个 key，标志着这个虚拟节点在 DHT 中的位置。当请求通过 IP 层的 Router 发送到这个 DHT 中时，接受的节点判断这个请求的文件对应的 key 由哪三个 replication 管理，然后转发到这三个机器上。

System operation：

这里描述系统的各个操作的过程。

系统初始化：

写：

读：

delete：

crash and recovery：

当 Crash 之后

如果crash系统之后就丢弃掉所有的数据，那么可能会导致系统的可靠性降低。但是在我们的攻击模型中，如果攻击者恶意破坏既可以crash系统，也可以人为的删除掉所有的数据。因此

crash系统之后就丢弃掉整个系统的文件在这个threat model中不会破坏reliability（因为同时crash 3个replica的概率很小）

How to hide relationship between nodes?

1. Data

   Replica Node 上存储同一个数据，但是他们使用的是不同的密钥，加密之后的 bytes 也不一样，因此通过 Data 无法找到对应的 3 个 replica。
2. Network traffic

Security of this system：（这里描述攻击方式和为什么攻击方式不起作用）

Rollback attack:（从 storage 角度的攻击）

因为攻击者无法知道

Replay attack：（从重复发送请求的角度的攻击）

Network traffic:

1. Attack with precise network traffic on one machine。
2. Attack on statistics

Future work：

1. 磁盘满了
2. 用户管理，文件分享等对象存储的功能
3. 合并两个 Nodes 作为持久对象（区分冷对象和热对象）参考 facebook f5 和 haystack。