第一天 - 项目介绍和部署

1. 学习目标

- **理解项目的整体功能, 细节无需明白, 后边会详细讲解**
- **清楚项目整体模块划分和后续要实现的功能**
- **完成项目所需的oracle数据表的导入**
- **Windows/Linux下完成OpenSSL的安装**
- **对加密相关概念有初步理解(后续会逐步深入讲解)**

2. 项目阶段要求

2.1 项目1 介绍

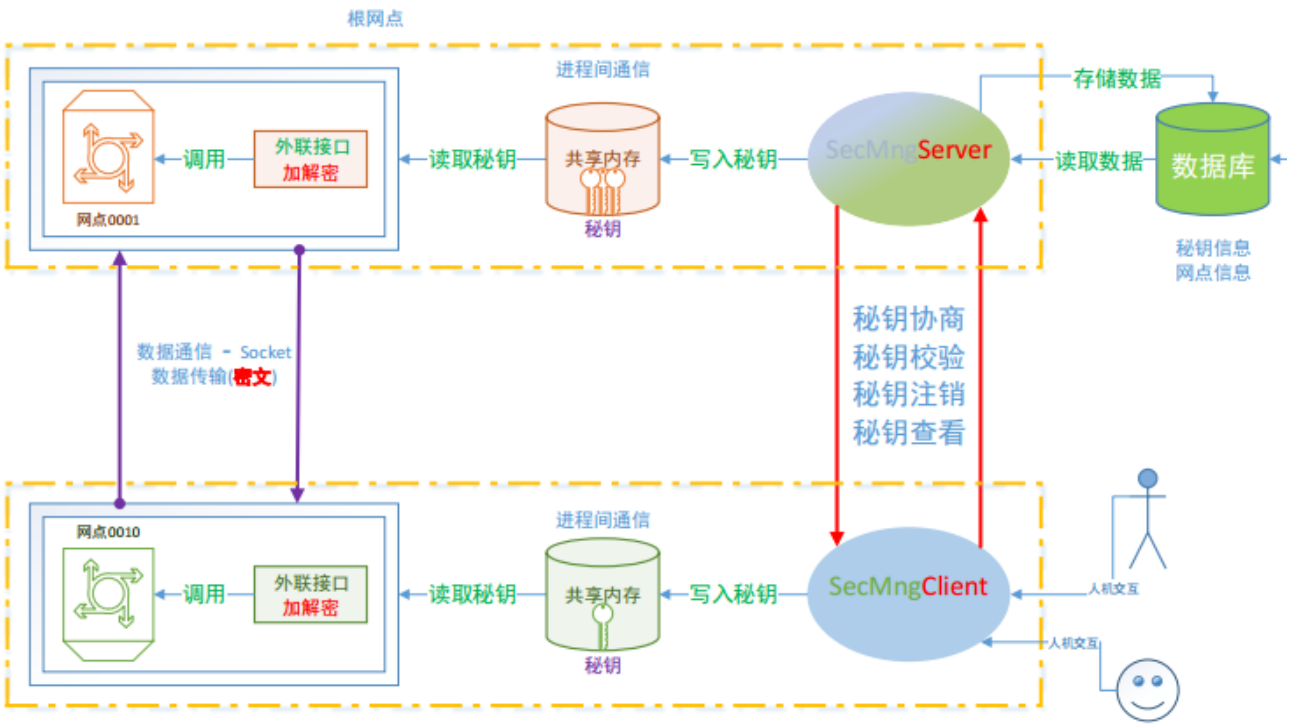
1. 项目名称 - `数据安全传输 基础设施平台`
    - 写简历的时候需要自己起个名字(根据项目的功能)
2. 应用场景:
    - 网络通信
        - socket
        - http
            - 传输层使用的tcp
    - 保证通信时数据的安全
        - 数据加密方式
            - 对称加密

        - 非对称加密
    - 项目1实现的功能
        - 秘钥生成, 校验
        - 数据加密

2.2 项目要求

1. 项目课训练什么?
    - 需求提炼
    - 项目的子系统划分，每个子系统的模块分解
    - 项目的开发经历、经验积累
        - 开发流程
        - 项目调试
    - 第三方框架/开源库的积累
    - 锻炼快读阅读代码的能力
    - 锻炼对封装好的API的快速上手能力
    - 锻炼处理问题的逻辑思维能力
2. 学习过程中注意事项:
    - 重视业务流、软件开发思维的训练；**有些代码是你看不懂的**
    - 用到什么，学什么；不随意扩充。
    - 迭代开发，先出来一个模型， 不要总想一下子把功能做的十分完美
    - 能快速的做出东西的程序员，是企业中的高手
    - 培养独立解决问题的能力

3. 项目整体架构图



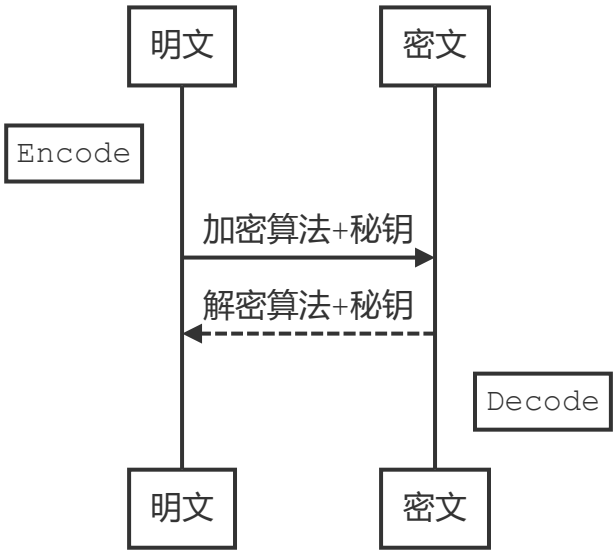
1. 网点A和网点B进行数据通信
    - 对数据加密, 需要秘钥(一个固定长度的字符串)
2. 秘钥协商系统
    - 秘钥协商服务器
    - 秘钥协商客户端
3. 套接字通信
    - 服务器
        - 多线程
        - 多进程
        - 多路IO + 多线程
    - 客户端
        - 多线程
        - 连接池
4. 进程间通信
    - 共享内存 - shm
    - 也可以使用redis
5. 数据的加解密
    - 对称加密
        - des/3des/aes
6. 数据库操作
    - oracle官方提供的C++接口
        - OCCI
7. QT相关
8. 守护进程
    - 信号捕捉
    - 编写shell脚本

4. 项目整体模块划分

1. 基础组件
    - 报文编解码组件
    - 通信组件
    - 进程间通信组件（共享内存）
    - 数据库访问组件
2. 秘钥协商服务器 && 客户端
3. 图形界面
    - 配置管理终端
    - 秘钥协商客户端（可选）
4. 加解密接口（外联接口）的封装
    - openssl中 AES 、DES、3DES 的使用

5. 数据的加密和解密

5.1 加密三要素



1. 三要素
    - 明文/密文
    - 算法
        - 加密算法
        - 解密算法
    - 秘钥
        - 字符串
        - 不同的加密算法对秘钥的长度不同

5.2 常用的加密方式

- 对称加密
    - 加密和解密的时候使用的是同一个秘钥
    - 特点:
        - 秘钥分发困难
        - 加密效率高
        - 安全级别低(相对于非对称加密)
- 非对称加密
    - 加密和解密的时候使用的秘钥不同 - 是一个秘钥对
        - 公钥 - 可以公开的秘钥
        - 私钥 - 不能公开
            - 传输的数据对谁更重要, 谁就那私钥
    - 应用场景
        - 开通网银, 会得到一个U盾, 私钥
        - 银行拿的是公钥
    - 加密的过程

        - 使用公钥加密, 必须使用私钥解密
        - 使用私钥加密, 必须使用公钥解密
    - 特点:
        - 秘钥分发简单
        - 加密效率低
        - 安全级别高

5.3 常用的加密算法

- 对称加密
    - **DES/3DES**
        - DES - 已经被破解了
            - 要求秘钥长度8字节
        - 3DES - 效率低
            - 秘钥长度24字节, 内部会将其分成3份
    - TDEA
    - Blowfish
    - RC2/RC4/RC5
    - IDEA
    - SKIPJACK
    - **AES**
        - 使用最广泛的对称加密算法
        - 秘钥要求:
            - 16字节, 24字节, 32字节
- 非对称加密
    - **RSA(数字签名和密钥交换)**
    - ECC（椭圆曲线加密算法）
    - Diffie-Hellman(DH, 密钥交换)
    - El Gamal(数字签名)
    - DSA（数字签名）
- Hash算法 -> 单向散列函数
    - 将任意长度的数据, 生成一个固定长度的字符串
    - MD4/MD5
        - 散列值长度16字节
    - SHA-1
        - 散列值长度20字节
    - SHA-2
        - SHA224/SHA256/SHA384/SHA512
        - sha224
            - 散列值长度: 28字节
    - SHA3-224/SHA3-256/SHA3-384/SHA3-512

- HMAC

6. 安装OpenSSL

6.1 openssl介绍

> OpenSSL 是一个安全套接字层密码库，囊括主要的密码算法、常用的密钥和证书封装管理功能及SSL协议，并提供丰富的应用程序供测试或其它目的使用。
>
> SSL是Secure Sockets Layer（安全套接层协议）的缩写，可以在Internet上提供秘密性传输。Netscape公司在推出第一个Web浏览器的同时，提出了SSL协议标准。其目标是保证两个应用间通信的保密性和可靠性,可在服务器端和用户端同时实现支持。已经成为Internet上保密通讯的工业标准。

6.2 安装

> 参考提供的文档

7. Oracle数据库表的导入

7.1 连接数据库前的准备工作

centos oralce数据管理员密码:

- System11g

如果想要通过外部客户端连接oracle服务器, 必须先关闭防火墙。

关闭防火墙的两种方式:

1. 通过linux命令
   - 切换到root用户

```
su - root
加 - 的意思是用户切换，对用的环境变量也切换
```

   - 执行命令

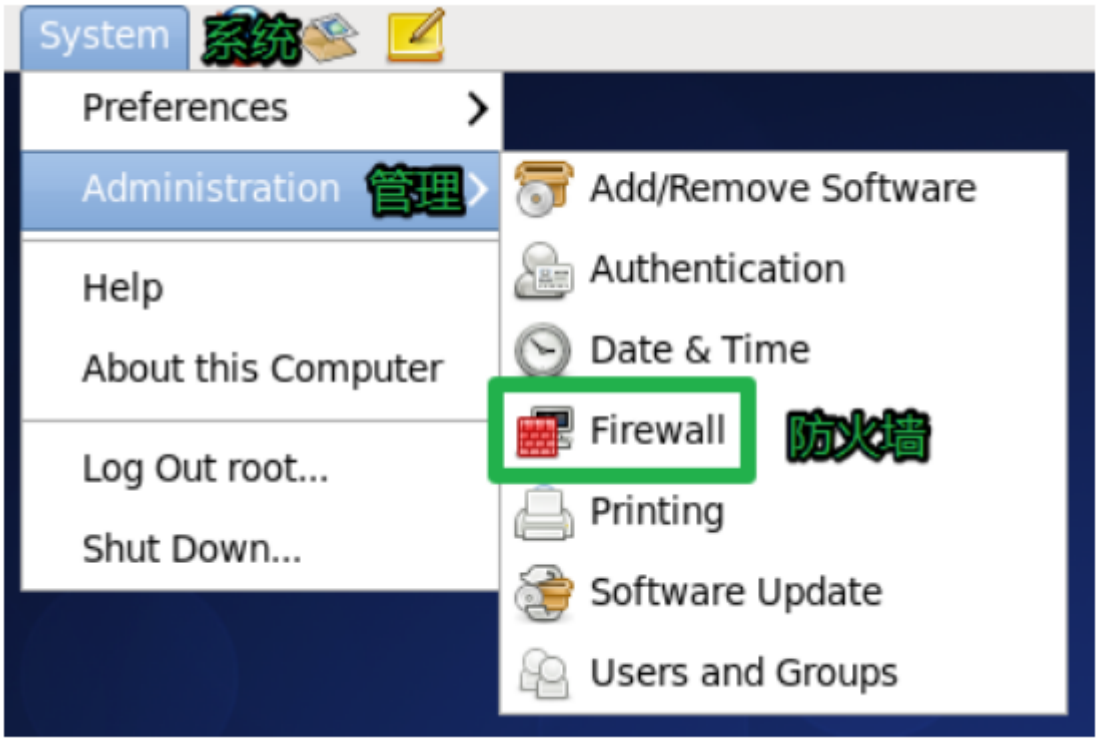
```
# iptables
# 关闭防火墙 - 不是永久设置
service iptables stop
# 启动防火墙
service iptables start
# 查看防火墙状态
service iptables status

# systemctl - centos7
# 关闭防火墙 - 不是永久设置
systemctl stop firewalld
#启动防火墙
systemctl start firewalld
# 查看状态
```

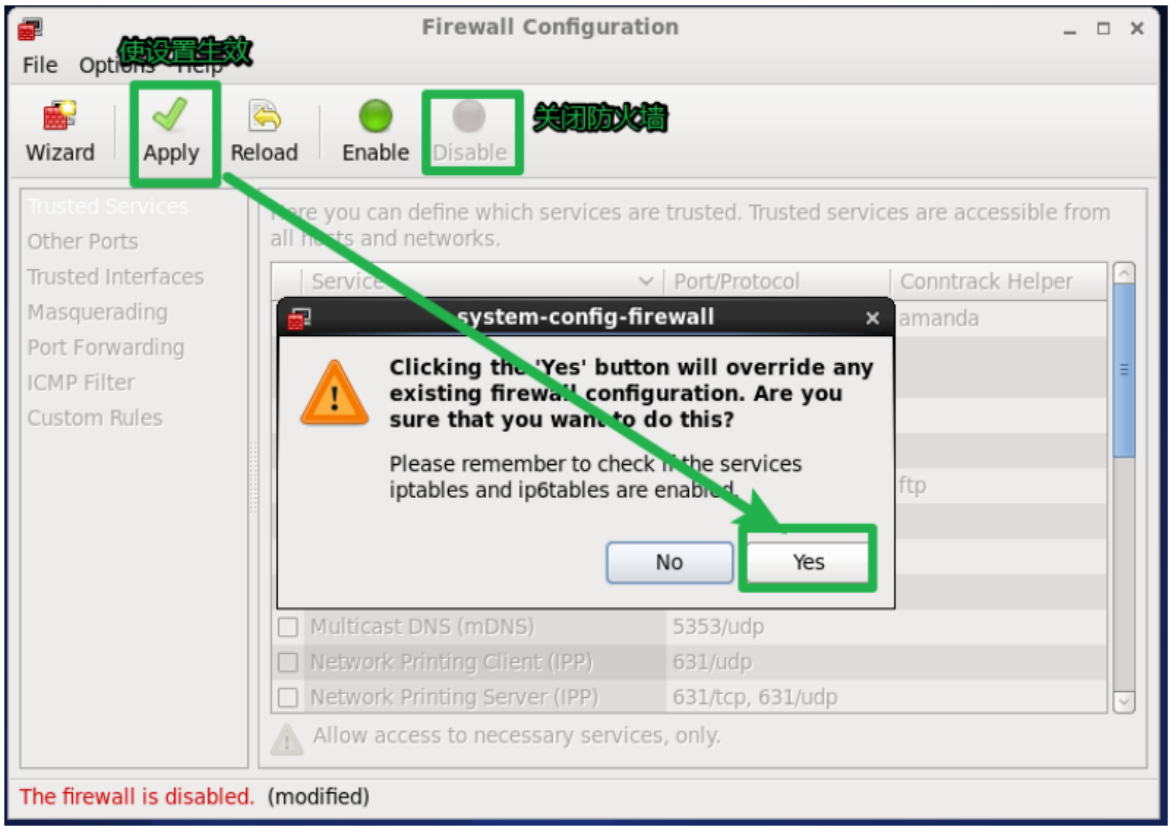
```
systemctl status firewalld
# 设置防火墙永久关闭
systemctl disable firewalld
# 设置开机启动防火墙
systemctl enable firewalld
```

- ■

2. 通过操作系统提供的对应的窗口 - RedHat, centos 6.x
   - 使用root用户登录Linux



   - 关闭防火墙



7.2 启动oracle数据库

1. 启动oracle数据库流程
   - 使用oracle用户登录到linux操作系统
   - 启动数据库 使用sqlplus

```
sqlplus
system/System11g as sysdba
startup - 启动
shutdown immediate - 关闭
exit  - 退出sqlplus
```

2. 启动TNS监听服务

```
lsnrctl start/stop
```

启动tns服务器失败

1. 切换到root用户
2. 执行一个命令

```
hostname oracle
```

3. 切换到oracle用户

4. 重写启动tns服务: lsnrctl start

3. 客户端连接

8 源码安装

安装流程:

1. 以下文件, 里边有安装步骤
    - readme
    - readme.md
    - INSTALL
2. 找 可执行文件 **configure**
    - 执行这个可执行文件
        - 检测安装环境
        - 生成 makefile
3. 执行**make**命令
    - 编译源代码
        - 生成了动态库
        - 静态库
        - 可执行程序
4. 安装 **make install** (需要管理员权限)
    - 将第三步生成的动态库/动态库/可执行程序拷贝到对应的系统目录