Python安全编程

——张龙

方法论

1. 意识
   - ◘安全编程很重要
   - ◘亡羊补牢，为时已晚

2. 认识
   - ■哪些地方需要安全编程

3. 知识
   - ✓如何安全编程

Web相关

后台相关

- 数据库操作
- 反序列化
- eval
- 执行命令
- API接口
- 权限

数据库操作

- 利用现成的库,django etc.
- 封装安全的SQL操作工具类
  - ×拼接sql，cursor.execute(sql)
  - ✓cursor.execute(sql,param)，sql用%s占位符，param用元组

反序列化

! 对于不是自己从已知数据中创建的Python pickle，不要对其进行unpickle操作

×cpickle/pickle, http://drops.wooyun.org/papers/66

pickle.loads("cos\nsystem\n(S'dir'\ntR.")

✓万不得已

通过继承类pickle.Unpickler并重写find_class()方法来实现

eval

将字符串当做有效的表达式进行求值并返回结果

! Use ast.literal_eval whenever you need eval

✗ *错误姿势:*

```
Import eval
Input=__import__('os').system('rm -rf /a-path-you-really-care-about')
eval(input)
```

✓ *正确姿势:*

```
from ast import literal_eval as eval
Input=__import__('os').system('rm -rf /a-path-you-really-care-about')
eval(input)
```

执行命令

! 尽量不要执行有外部输入因子的命令

! 用python自有的库函数(glob, fnmatch, shutil等)

×os.system()、os.popen*()、commands.*、os.spawn*、popen2.*

✓Subprocess，千万不要让shell=True

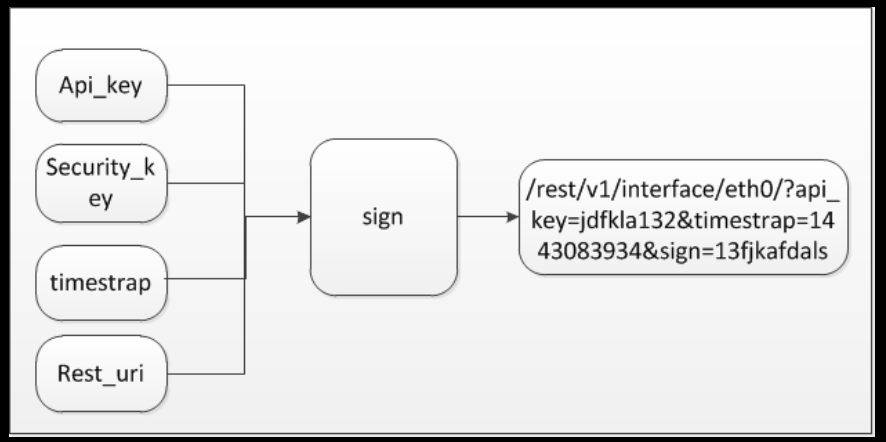
- *.call(args, *, stdin=None, stdout=None, stderr=None, shell=False)*
- *.check_output(args, *, stdin=None, stderr=None, shell=False, universal_newlines=False)*
- *.check_call(args, *, stdin=None, stdout=None, stderr=None, shell=False)*

API接口

! 敏感信息防止窃听

✓POST+SSL

! 避免重放攻击



✓重要ID不透明处理

为了防止字典遍历攻击，可对id进行url62或者uuid处理

权限

✓最小权限

收缩进程所享有的特权，以防进程滥用特权

✓以特定账号运行（apache,jenkins）

进程所有者不能是文件所有者

✓特定运行帐号不可用Shell并锁定

chsh -s /bin/false $xxxuser

usermod -L $xxxuser

✓沙箱隔离

- chroot
- docker

Q&A