API接口安全checklist

身份认证

- [x] 不要使用Basic Auth，请使用标准的认证协议（如JWT，OAuth）。
- [x] 不要重新实现Authentication、token generating和password storing，请使用标准库。
- [x] 限制密码错误尝试次数，并且增加账号冻结功能。
- [x] 加密所有的敏感数据。

JWT（JSON Web Token）

- [x] 使用随机复杂的密钥（JWT Secret）以增加暴力破解的难度。
- [x] 不要在请求体中直接提取数据，要对数据进行加密（HS256或RS256）。
- [x] 使 token 的过期时间尽量的短（TTL，RTTL）。
- [x] 不要在 JWT 的请求体中存放敏感数据，因为它是可解码的。

OAuth 授权或认证协议

- [x] 始终在后台验证redirect_uri，只允许白名单的 URL。
- [x] 始终在授权时使用有效期较短的授权码（code）而不是令牌（access_token）（不允许response_type=token）。
- [x] 使用随机哈希数的state参数来防止跨站请求伪造（CSRF）。
- [x] 对不同的应用分别定义默认的作用域和各自有效的作用域参数。

访问

- [x] 限制流量来防止 DDoS 攻击和暴力攻击。
- [x] 在服务端使用 HTTPS 协议来防止 MITM （中间人攻击）。
- [x] 使用HSTS协议防止 SSL Strip 攻击。

输入

- [x] 使用与操作相符的 HTTP 操作函数，GET)，POST，PUT/以及DELETE，如果请求的方法不适用于请求的资源则返回405 Method Not Allowed。
- [x] 在请求头中的content-type字段使用内容验证来只允许支持的格式（如application/xml，application/json等等）并在不满足条件的时候返回406 Not Acceptable。
- [x] 验证content-type中申明的编码和你收到正文编码一致（如application/x-www-form-urlencoded，multipart/form-data，application/json等等）。
- [x] 验证用户输入来避免一些普通的易受攻击缺陷（如XSS，SQL-，等等）。
- [x] 不要在 URL 中使用任何敏感的数据（credentials，Passwords，security tokens，orAPI keys），而是使用标准的认证请求头。
- [x] 使用一个 API Gateway 服务来启用缓存、限制访问速率（如Quota，Spike Arrest，Concurrent Rate Limit）以及动态地部署 APIs resources。

处理

- [x] 检查是否所有的接口都包含必要都身份认证，以避免被破坏了的认证体系。
- [x] 避免使用特有的资源 id。使用/me/orders替代/user/654321/orders。
- [x] 使用UUID代替自增长的 id。
- [x] 如果需要解析 XML 文件，确保实体解析（entity parsing）是关闭的以避免XXE攻击。
- [x] 如果需要解析 XML 文件，确保实体扩展（entity expansion）是关闭的以避免通过指数实体扩展攻击实现的Billion Laughs/XML bomb。
- [x] 在文件上传中使用 CDN。
- [x] 如果数据处理量很大，尽可能使用队列或者 Workers 在后台处理来避免阻塞请求，从而快速响应客户端。
- [x] 不要忘了把 DEBUG 模式关掉。

输出

- [x] 增加请求返回头X-Content-Type-Options: nosniff。
- [x] 增加请求返回头X-Frame-Options: deny。
- [x] 增加请求返回头Content-Security-Policy: default-src 'none'。
- [x] 删除请求返回中的指纹头 –X-Powered-By, Server, X-AspNet-Version等等。
- [x] 在响应中遵循请求的content-type，如果你的请求类型是application/json那么你返回的content-type就是application/json。
- [x] 不要返回敏感的数据，如credentials，Passwords，security tokens。
- [x] 给请求返回使用合理的 HTTP 响应代码。（如200 OK，400 Bad Request，401 Unauthorized，405 Method Not Allowed等等）。

持续集成和持续部署

- [x] 使用单元测试以及集成测试的覆盖率来保障你的设计和实现。
- [x] 引入代码审查流程，禁止私自合并代码。
- [x] 在推送到生产环境之前确保服务的所有组件都用杀毒软件静态地扫描过，包括第三方库和其它依赖。
- [x] 为部署设计一个回滚方案。