OWASP
Open Web Application
Security Project

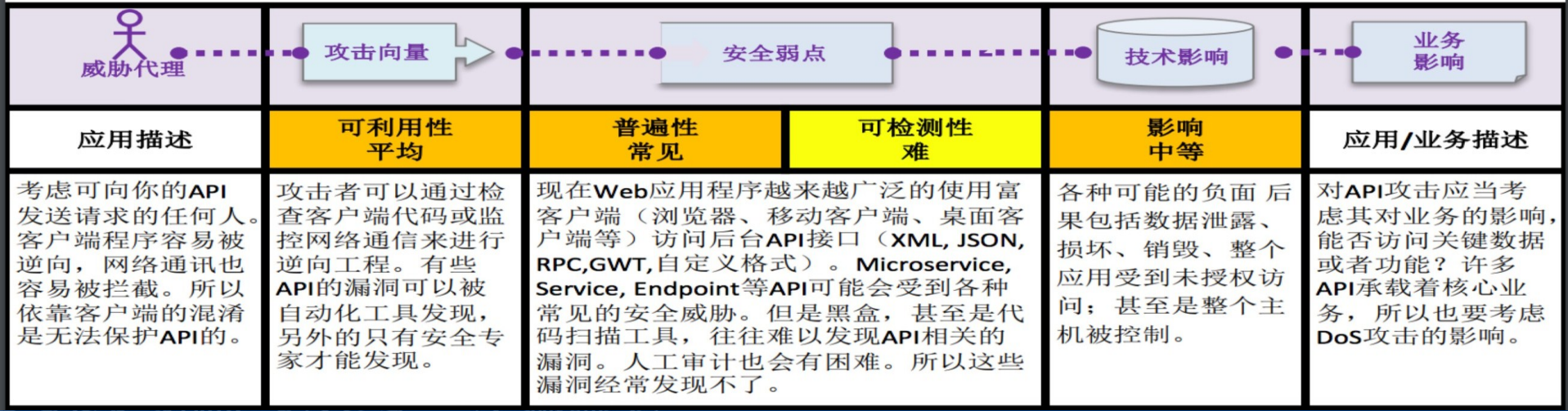
从S-SDLC视角看API安全

【API威胁检测防御】专题研讨会

Silver Zhang

为何将API安全拧出来说

A10 未受有效保护的API



应用描述	可利用性 平均	普遍性 常见	可检测性 难	影响 中等	应用/业务描述
考虑可向你的API发送请求的任何人。客户端程序容易被逆向，网络通讯也容易被拦截。所以依靠客户端的混淆是无法保护API的。	攻击者可以通过检查客户端代码或监控网络通信来进行逆向工程。有些API的漏洞可以被自动化工具发现，另外的只有安全专家才能发现。	现在Web应用程序越来越广泛的使用富客户端（浏览器、移动客户端、桌面客户端等）访问后台API接口（XML, JSON, RPC,GWT,自定义格式）。Microservice, Service, Endpoint等API可能会受到各种常见的安全威胁。但是黑盒，甚至是代码扫描工具，往往难以发现API相关的漏洞。人工审计也会有困难。所以这些漏洞经常发现不了。		各种可能的负面 后果包括数据泄露、损坏、销毁、整个应用受到未授权访问；甚至是整个主机被控制。	对API攻击应当考虑其对业务的影响，能否访问关键数据或者功能？许多API承载着核心业务，所以也要考虑DoS攻击的影响。

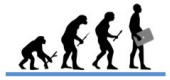
为何将API安全拧出来说

感冒 VS. 禽流感

新业态

应用安全 VS. 移动安全

API历史沿革及风险评估



早期API

低风险，风险主要源于传统的安全开发过程

内部API微服务

中高风险，风险主要来源于团队间对业务安全需求理解不一致

标准API

高风险，风险主要源由于过于广泛的使用暴露API提供房安全实现上的弱点

API与业务系统100%捆绑

客户需求驱动API分离-

内部API打包服务

对外提供的公共API

标准的API如：Restful

新业态API

客户集成需求

中等风险，风险主要来源于API与公司产品业务实现不一致

公共API

较高风险，风险主要源于API提供方无法预知最终用户的使用场景及使用方法

万物互联

极其复杂的使用环境、不理解安全的使用方以及无安全意识的最终用户

API面临的安全挑战

传输	存储	认证	授权	输入	输出
客户端不支持加密传输协议怎么办 客户端需要走非加密支持代理 客户端未做服务端证书合法有效性校验怎么办	服务端加密数据如何让客户端解密 客户的服务端与客户端都在使用同一API密钥泄露了怎么办	客户的客户端与服务端都支持登录过程处理 无人参与的设备登陆认证问题 设备更换主人	预设权限与客户业务映射 不充分的授权业务架构 不正确的授权ID的使用	API服务提供方输入验证 API提供方输入验证 调用方输入验证匹配性	输出编码 不同的输出编码-CSV/JASON 仅输出必要的内容

IoT攻击面分析

攻击面	是否涉及API	攻击面	是否涉及API
生态系统	√	第三方后端API	√
设备内存	X	更新机制	√
设备物理接口	〆	移动应用	√
设备Web界面	√	供应商后端API	√
设备固件	〆	生态系统交互	√
设备网络服务	〆	网络流量	〆
管理界面	√	认证/授权	√
本地数据存储	〆	隐私	√
云Web界面	〆	硬件（传感器）	〆

S-SDLC & API安全

API安全	需求	设计	实现	确认	DevOps	运维	场景
传输	✗	✗	✗	✗	✗	✗	✓
存储	✓	✓	✓	✓	✗	✓	✓
认证	✓	✓	✓	✓	✗	✓	✓
授权	✓	✓	✓	✓	✗	✓	✓
输入	✓	✓	✓	✓	✗	✓	✓
输出	✓	✓	✓	✓	✗	✓	✓

API安全检测技术

- 人为构造所有可能的应用场景的Demo,通过正常应用安全测试完成...
- 与功能性自动化脚本融合，同步完成安全自动化测试...
- 借助现存的典型用户客户端完成全功能安全测试...
- 借助于Fuzz Test技术完成安全性测试...
- 借助于IAST技术完成安全性测试...
- 借助于RASP技术完成与业务强相关的恶意行为监测...

问题讨论-1

如何避免过度实现尚未开放的功能？

1. 无人知道对外提供的API竟然可以以服务提供方的超级管理员身份登录进去
2. 现实是：开发为了方便放开了口子
3. 挑战：如何从根本上避免这样的问题出现?

API特殊性决定了它一定会被深度使用

问题讨论-2

如何确保API只输出当前功能所必需要的信息？

1. 开发：我知道这些信息未来用得上，所以干脆直接将更多的信息一并传回客户端，反正前端也不会有人使用它们
2. 现实：批量、敏感信息泄漏
3. 挑战：如何从根本上避免这样的问题出现?

API特殊性决定了它一定会被深度使用

问题讨论-3

最佳实践与现实之间的矛盾

1. Web端提供多因子认证
2. 手机端API则不提供
   1）易用性考虑
   2） 客户的客户端不支持

API特殊性决定了它一定会被深度使用