







安全审计系统设计与实现

tuhao

提纲

-  为什么要安全审计
-  设计怎样的安全审计系统
-  如何实现安全审计系统
-  如何应用安全审计系统
-  延伸和扩展
-  FAQ

为什么要服务器安全审计

服务器信息收集

- ✎ 什么操作系统内核、跑什么进程、开什么端口、有什么用户...

日志收集

- ✎ 通过分析能及时发现系统入侵痕迹或者检查到用户su/sudo记录是否合法
- ✎ 安全应急响应（机器被黑，故障发生前做了什么操作）

访问控制检查

- ✎ iptables和常见服务的访问控制是否安全合理







漏洞本地检测

- ✎ 普通用户提权、包存在漏洞未及时升级...

异常流量发现

- ✎ 出向大流量：被脱库or ddos肉鸡？

画什么样的饼

-  c/s架构
-  client
-  collector
-  storager
-  analyzer
-  scheduler

饼要怎么画1


client

应用	工作模式	开发语言	资源消耗	功能	扩展性	社区活跃度	操作系统支持
Lynis	crontab	shell	均值很低，基本不吃cpu	根据测试用例和插件自定义	看测试用例和插件实现程度	强，也有商业版本	*nix
Ossec	c/s	C	吃cpu/memory不到5%	支持实时入侵检测和系统审计	强，更新活跃	强	支持*nix系统和windows

collector-storager

 ELK

scheduler

 自研

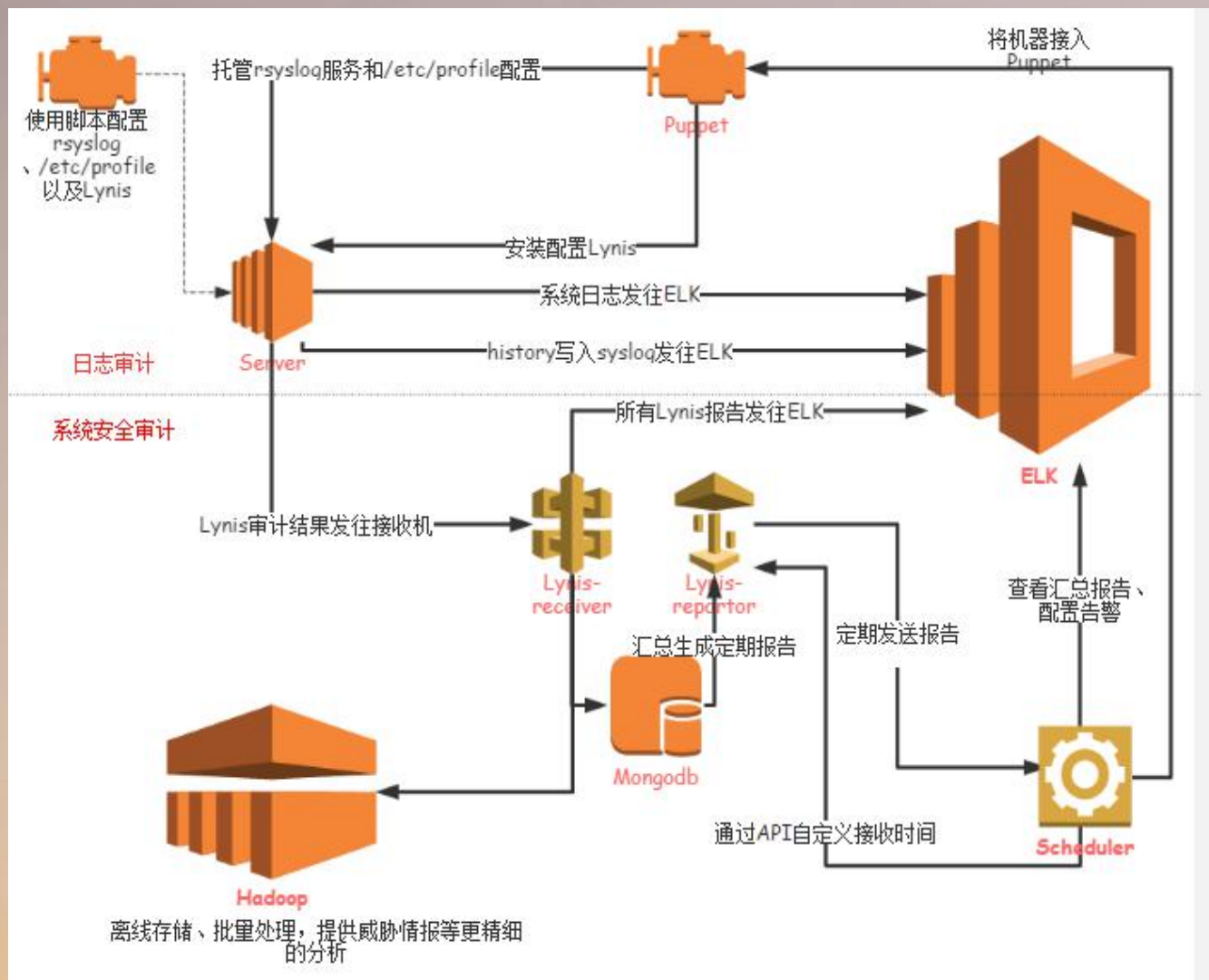
analyzer

hadoop

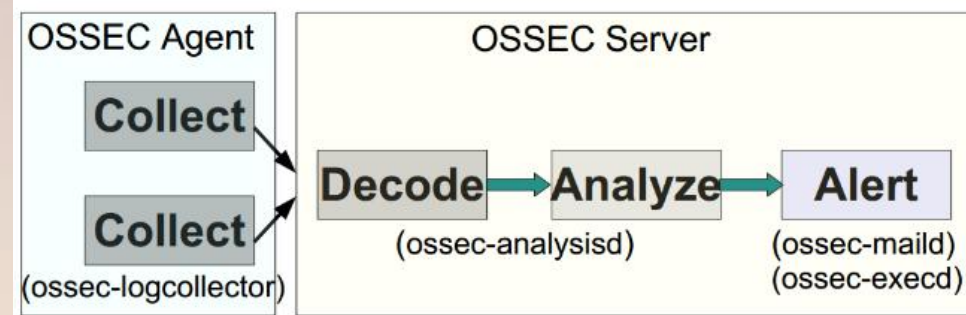
部署

puppet/ansible/saltstack

饼要怎么画2




Log flow (agent/server)



饼要怎么吃

- ✎ 海量机器 如何汇总分发检测结果
- ✎ 检测什么
- ✎ 如何跟现有CMDB结合
- ✎ 如何与端口扫描、漏洞扫描结合
- ✎ 如何分析

够好吃了吗






 用户体验

 迭代

 是否推送修复补丁或者只提供修复方案

 如何结合威胁情报快速检测和响应最新漏洞

参考资料

-  OSSEC: <http://ossec.github.io/>
-  FreeBuf: <http://www.freebuf.com/articles/system/21383.html>
-  Lynis: <https://cisofy.com/lynis/>
-  ELK: <https://www.elastic.co/webinars/introduction-elk-stack>
-  PUPPET: <https://puppet.com/>

