关于网络纵深防御的思考

主要分享内容

- 对信息安全攻防模式的思考
- 对体系化攻击模式的理解（防守者视角）
- 对体系化防御需求的理解
- 对纵深防御的理解
- 对多样化纵深以及防御覆盖的理解

复杂攻击与精妙利用层出不穷—绝望的防守者?

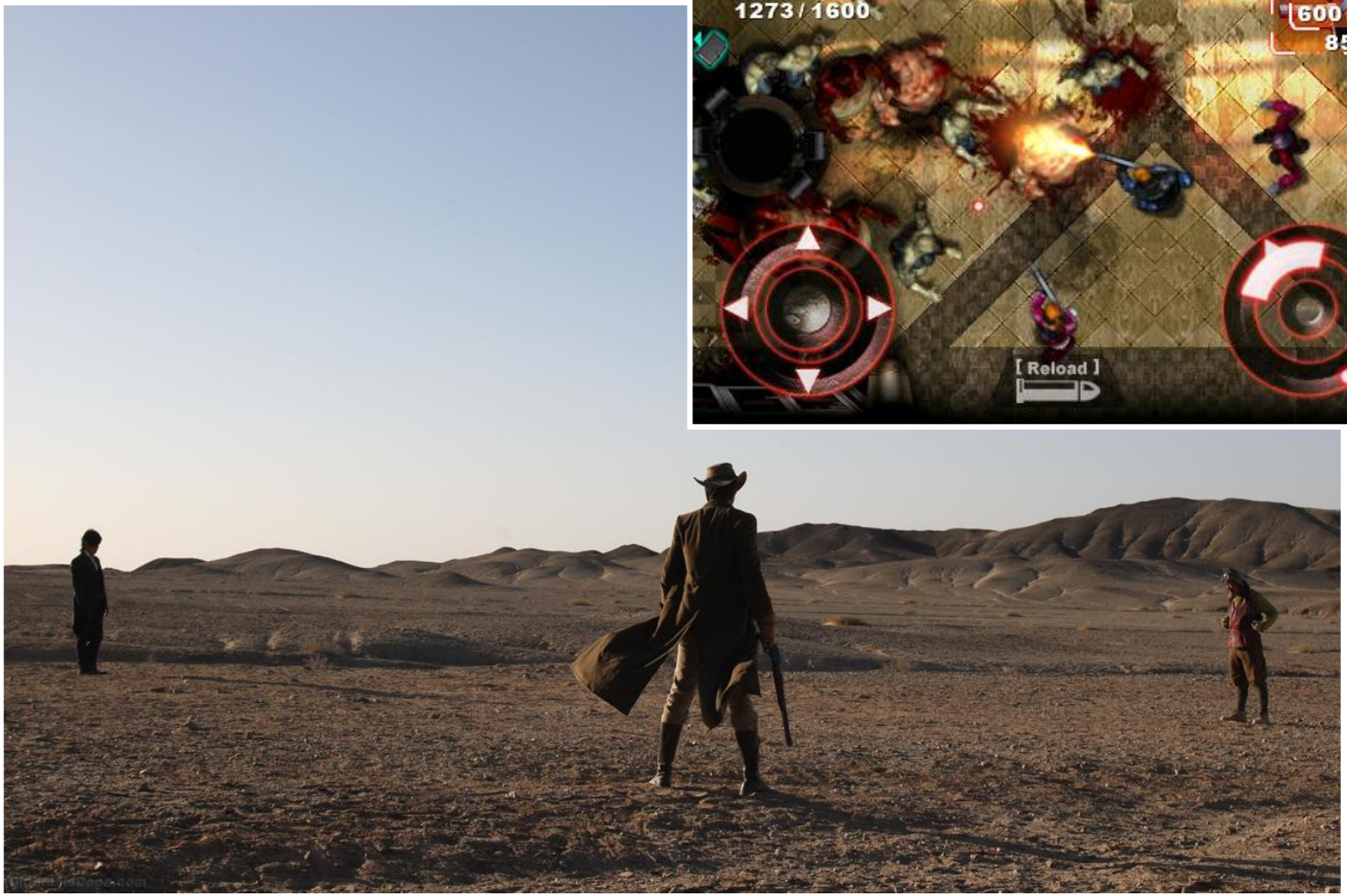
信息安全防御变得越来越困难，感觉不管采用怎么强悍的防御手段，都很快会被一些“精妙”的攻击所击破：

- APT攻击让传统防御手段变得形同虚设；
- 信息交互的刚需使网络隔离难以奏效；
- 各种宣称“解决一起安全问题”的防御技术很快被绕过...

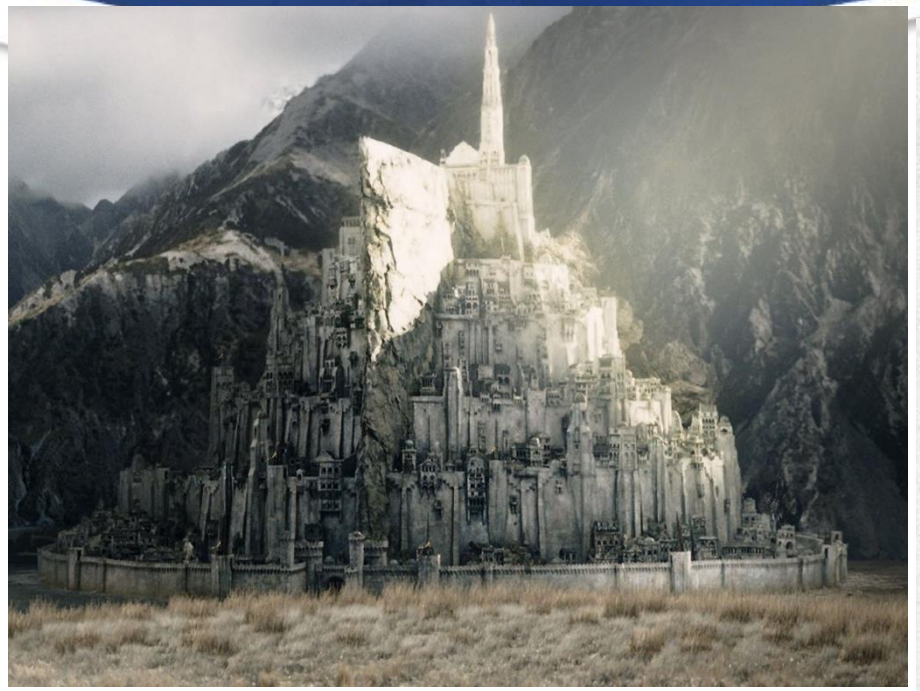
信息安全防守看起来那么让人绝望，重要信息系统就像游戏中的BOSS那样，最终逃不过被打垮的命运...

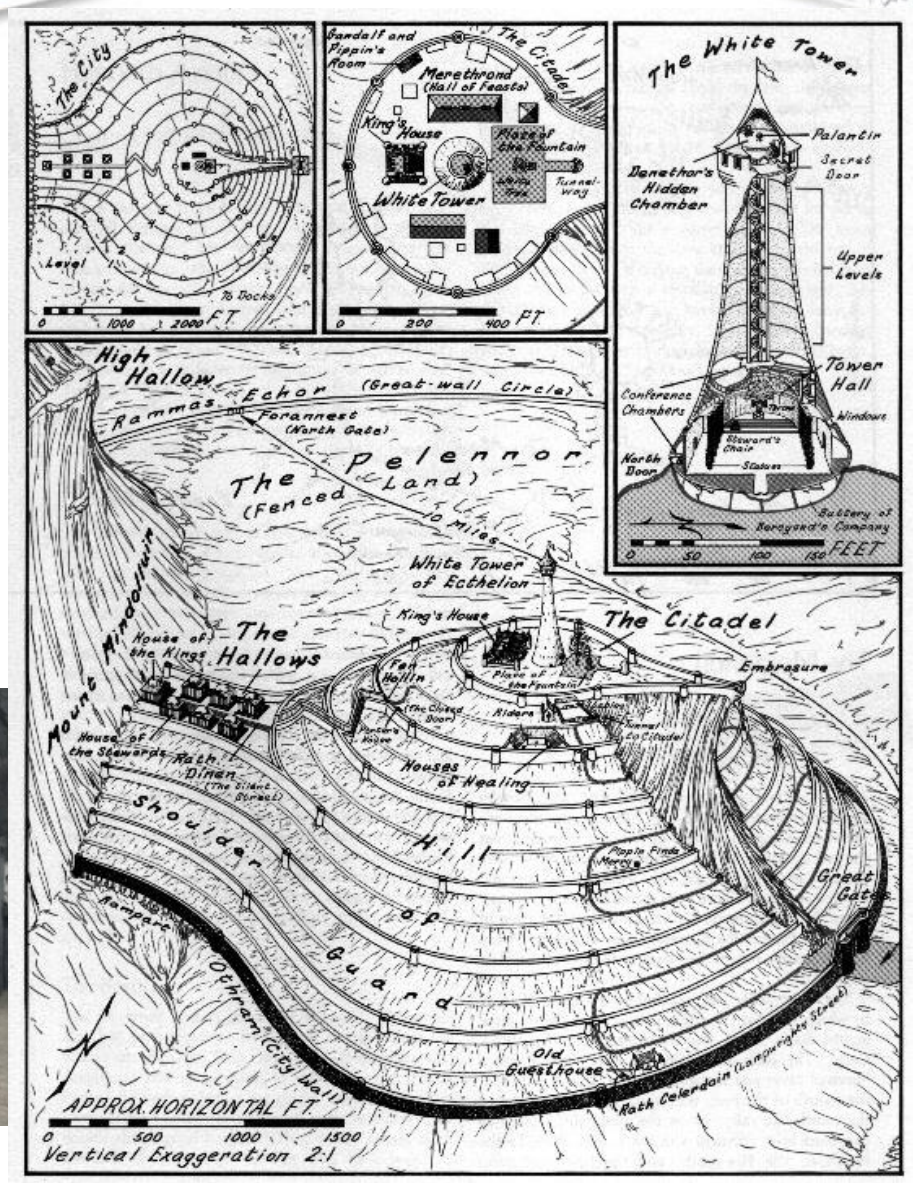
面对“丧尸式”的进攻...

哪一种防御更有效 —— 孤胆英雄?



哪一种防御更有效 —— Minas Tirith?





对攻与防的思考 — 我们是不是“玩错游戏”了?

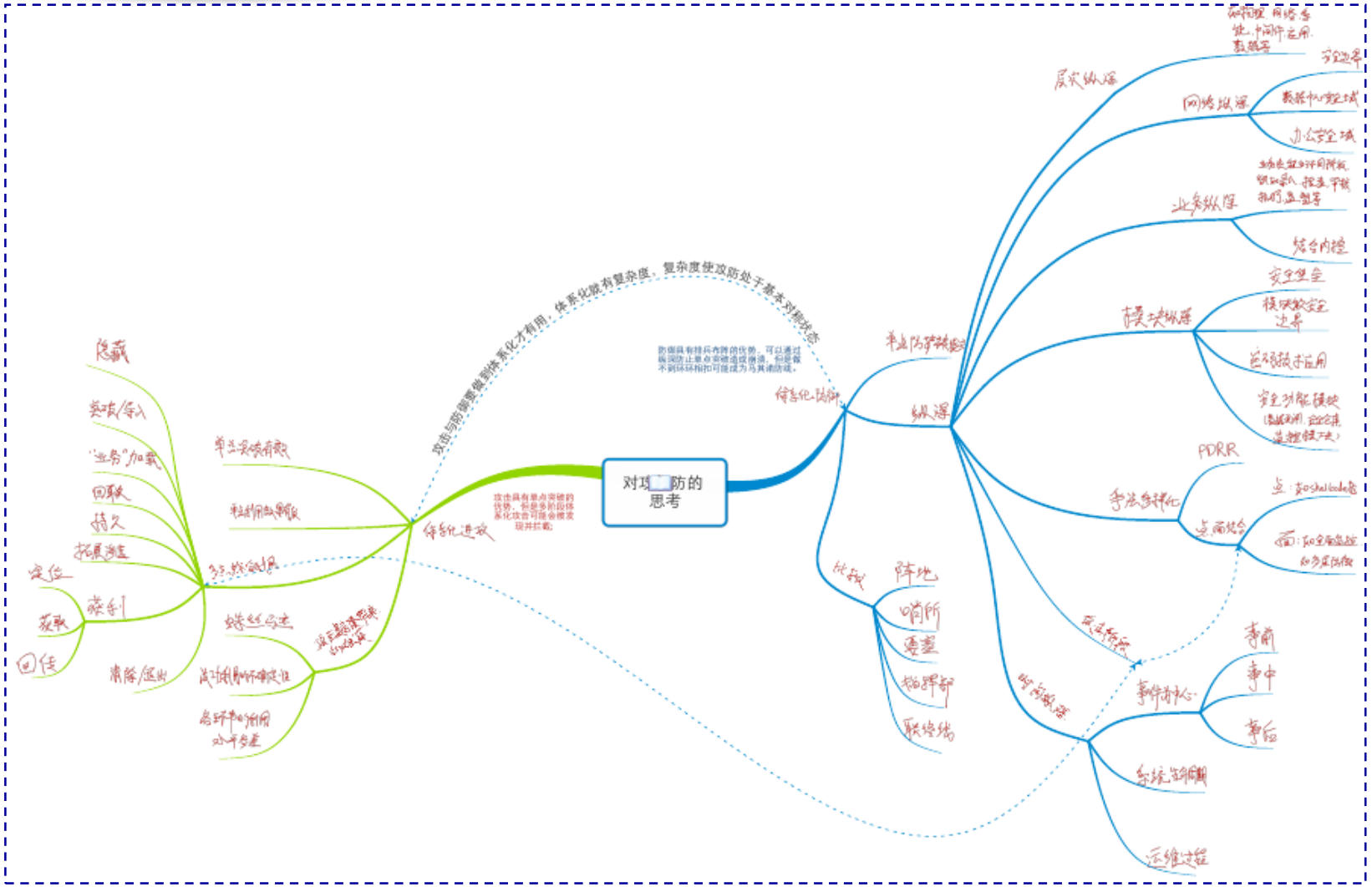
如果按照玩“英雄冒险”游戏的方式来考虑信息安全攻防对抗，可能信息安全防守真是令人绝望的… 要考虑保护资产的防守方英雄面对专注进攻的对手确实没有优势!

但是如果防守者用好先发优势，选择正确的“游戏形式“呢？如果抛弃“英雄对决”而选择和攻击方玩“塔防游戏”呢?

对防御者而言，他会放弃在领土上相对微弱的抵抗，而采取全力压迫攻击方之后勤补给，或是切割敌方在数量上的优势的兵力。一但攻击方失去其动能，或是其在大部份地区被切割后的兵力数量优势不再，防御反攻将在敌人虚弱地带发动，其主旨在于促使敌方资源之消耗，进而带动消耗战，或是迫使攻击方退回原本攻击起始点。

—— 摘自Wikipedia的“纵深防御”词条

体系化进攻 v.s. 体系化防守

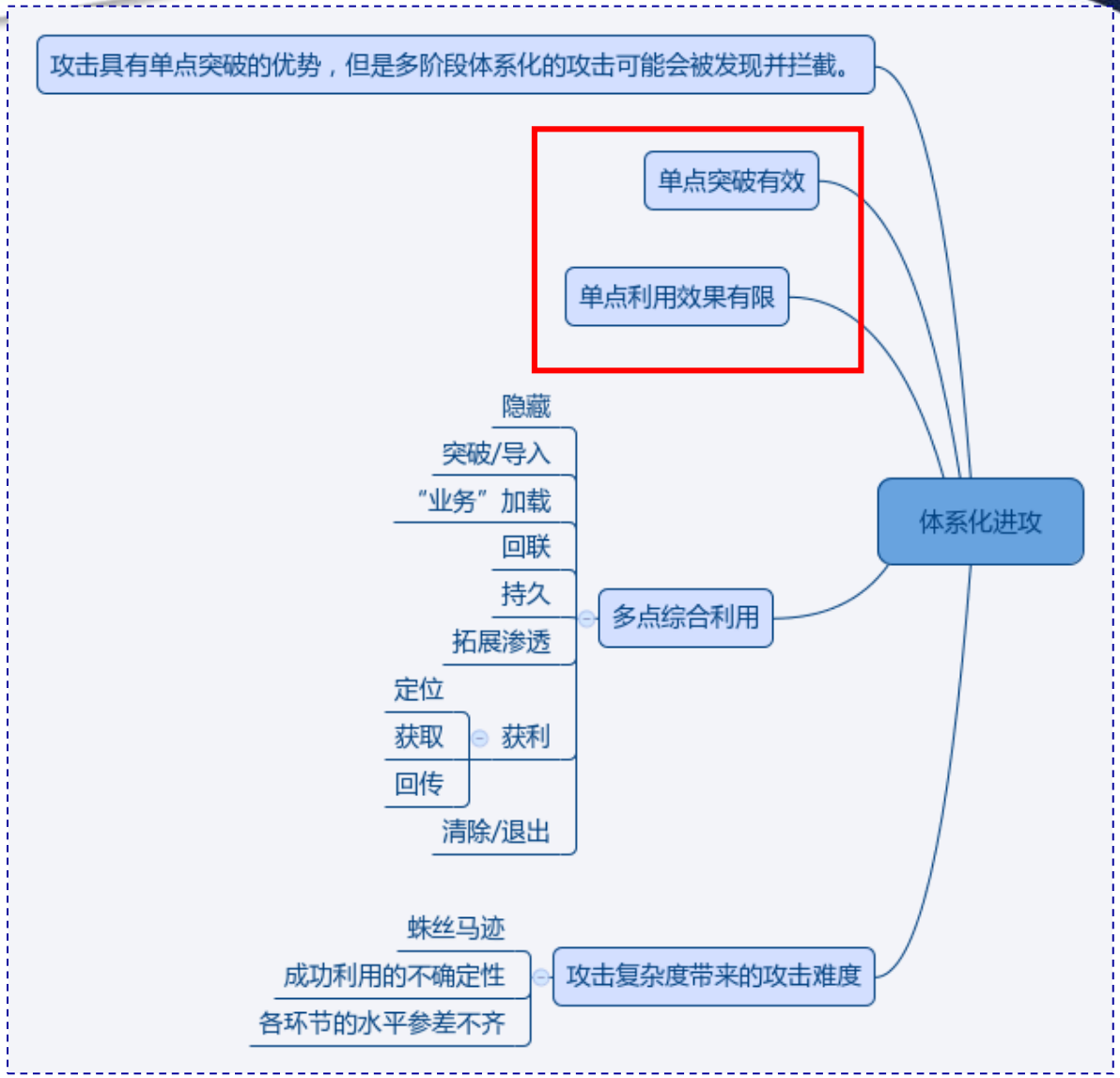


体系化的攻击

信息安全漏洞层出不穷，而且利用手法也日益精妙，但是这些突破大多都是“单点突破”，而绝大多数一个单点突破自身都难以达到攻击目的。

因此目前可以观测到的趋势正是攻击行为的“体系化”，攻击阶段越来越多，甚至攻击任务都出现了“专业外包”的情况。面对这样复杂的进攻，传统的安全边界或网络隔离策略难以奏效...

的确，单一的防护机制在体系化进攻面前无能为力！但是如果假设我们拥有多样化的纵深防御能力，进攻行为的体系化形式，恰恰带来了更多的防御点——“进攻的薄弱环节”。

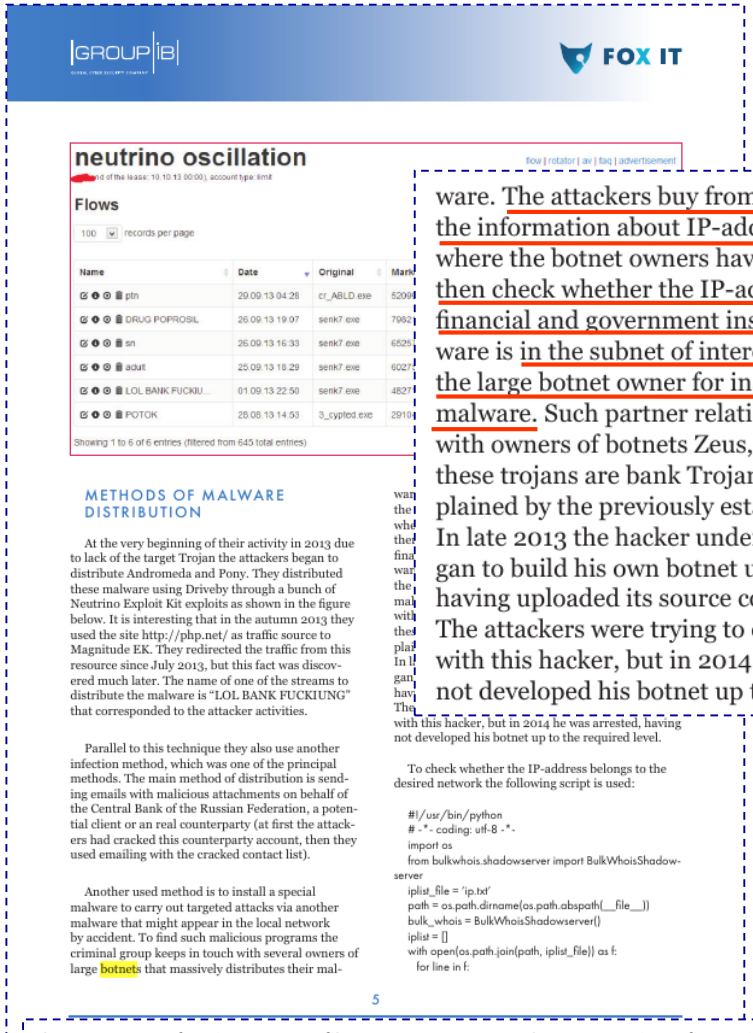


体系化的攻击——“攻击供应链”的“业务化”

俄罗斯犯罪团伙恶意代码Anunak开展对金融机构开展APT攻击，在Group-IB与Fox-IT 的联合分析报告中，可以发现一个很有意思的细节，那就是对“攻击供应链”的“业务化”利用。

攻击者与大型僵尸网络“运营商”形成了紧密的合作关系，攻击者不是简单地将恶意代码交给“运营商”大规模散布，而是在“运营商”提供的肉鸡名单中进行IP地址比对挖掘，找出那些可能属于金融机构或政府部门的肉鸡，并指示“运营商”向这些特定目标投放恶意代码。

看看咱们内网的僵尸网络肆虐情况，“细思极控”啊！



GROUP-IB FOX IT

neutrino oscillation

Flows

Name	Date	Original
ptn	29.09.13 04:28	cr_ABLD.exe
DRUG POPROSIL	26.09.13 19:07	senk7.exe
sn	26.09.13 16:33	senk7.exe
adult	25.09.13 18:29	senk7.exe
LOL BANK FUCKIU...	01.09.13 22:50	senk7.exe
POTOK	28.08.13 14:53	3_cypted.exe

Showing 1 to 6 of 6 entries (filtered from 645 total entries)

METHODS OF MALWARE DISTRIBUTION

At the very beginning of their activity in 2013 due to lack of the target Trojan the attackers began to distribute Andromeda and Pony. They distributed these malware using Driveby through a bunch of Neutrino Exploit Kit exploits as shown in the figure below. It is interesting that in the autumn 2013 they used the site http://php.net/ as traffic source to Magnitude EK. They redirected the traffic from this resource since July 2013, but this fact was discovered much later. The name of one of the streams to distribute the malware is “LOL BANK FUCKIUNG” that corresponded to the attacker activities.

Parallel to this technique they also use another infection method, which was one of the principal methods. The main method of distribution is sending emails with malicious attachments on behalf of the Central Bank of the Russian Federation, a potential client or an real counterparty (at first the attackers had cracked this counterparty account, then they used emailing with the cracked contact list).

Another used method is to install a special malware to carry out targeted attacks via another malware that might appear in the local network by accident. To find such malicious programs the criminal group keeps in touch with several owners of large botnets that massively distributes their malware. The attackers buy from these botnet owners the information about IP-addresses of computers where the botnet owners have installed malware and then check whether the IP-address belongs to the financial and government institutions. If the malware is in the subnet of interest, the attackers pay the large botnet owner for installation of their target malware. Such partner relations were established with owners of botnets Zeus, Shiz Ranbyus. All of these trojans are bank Trojans, their usage is explained by the previously established relationships. In late 2013 the hacker under the alias Dinhold began to build his own botnet using modified Carberp, having uploaded its source code for public access. The attackers were trying to create similar relations with this hacker, but in 2014 he was arrested, having not developed his botnet up to the required level.

To check whether the IP-address belongs to the desired network the following script is used:

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import os
from bulkwhois.shadowserver import BulkWhoisShadowserver
iplist_file = 'ip.txt'
path = os.path.dirname(os.path.abspath(__file__))
bulk_whois = BulkWhoisShadowserver()
iplist = []
with open(os.path.join(path, iplist_file)) as f:
  for line in f:
```

5

https://www.fox-it.com/en/files/2014/12/Anunak_APT-against-financial-institutions2.pdf

体系化的攻击——“追求极致”的NSA

TOP SECRET//COMINT//REL USA, AUS, CAN, GBR, NZL

Success 1: IPSec

Follow-the-Money and TAO Targets

- TOPI (S2C22) has had a close relationship with TAO for quite some time
- FTM Target 1
  - Not susceptible to any of NSP's implants
  - TAO got the configuration files which provided us the PSKs to enable passive exploitation

TOP SECRET//COMINT//REL TO USA, AUS, CAN, GBR, NZL

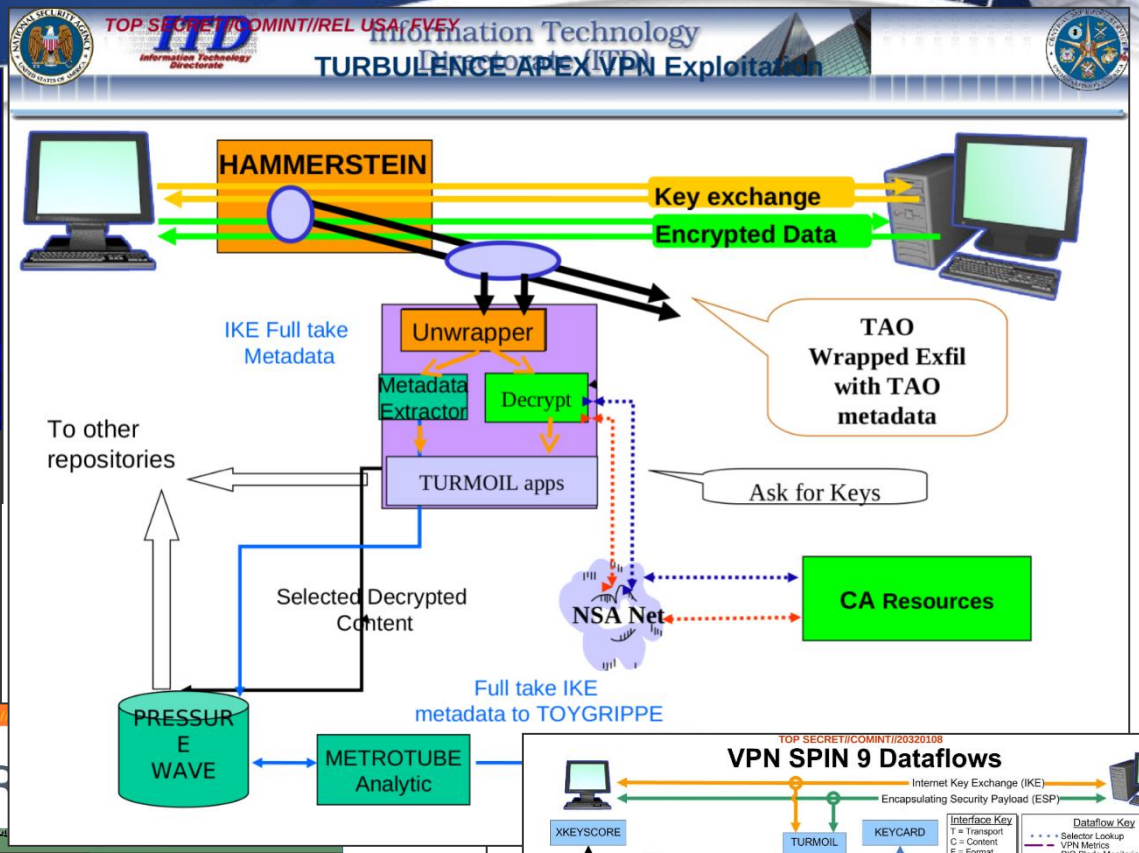
(S//SI//REL) DISCOROUTE Manifest Tag

- (TS//SI//REL) H - TAO has a presence on the router
- (S//SI//REL) M - multihop router. The admin telnetted into a router and then telnetted again to another device. Potential goldmine of information about your network, but be careful when looking through them to make sure you are associating an IP with the correct device.
- (TS//SI//REL) K - crypto keys

TOP SECRET//COMINT//REL TO USA,

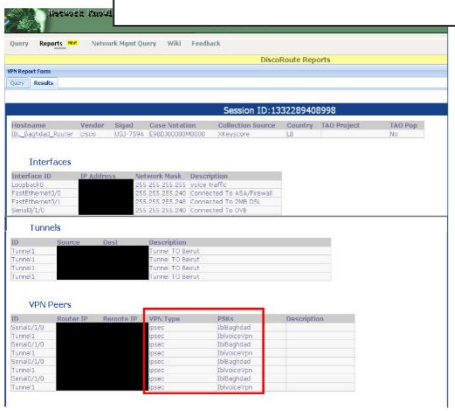
TOP SECRET//COMINT//REL USA, FVEY

Information Technology Directorate (ITD) TURBULENCE APEX VPN Exploitation



TOP SECRET//

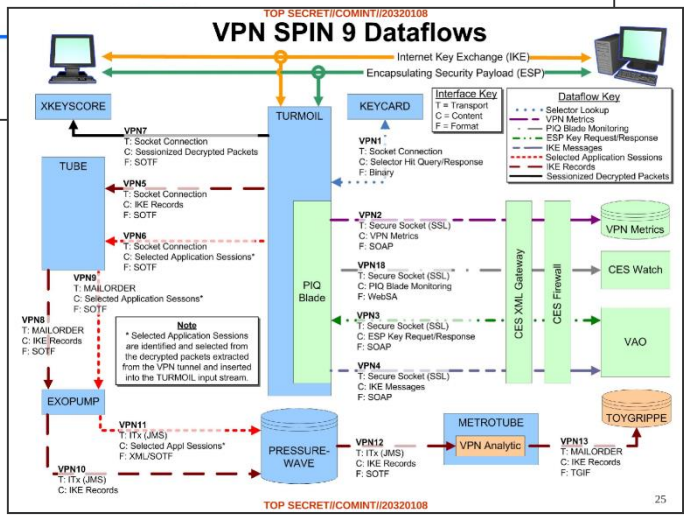
(S//SI//R



TOP SECRET//COMINT//REL TO USA, AUS, CAN, GBR, NZL

TOP SECRET//COMINT//20320108

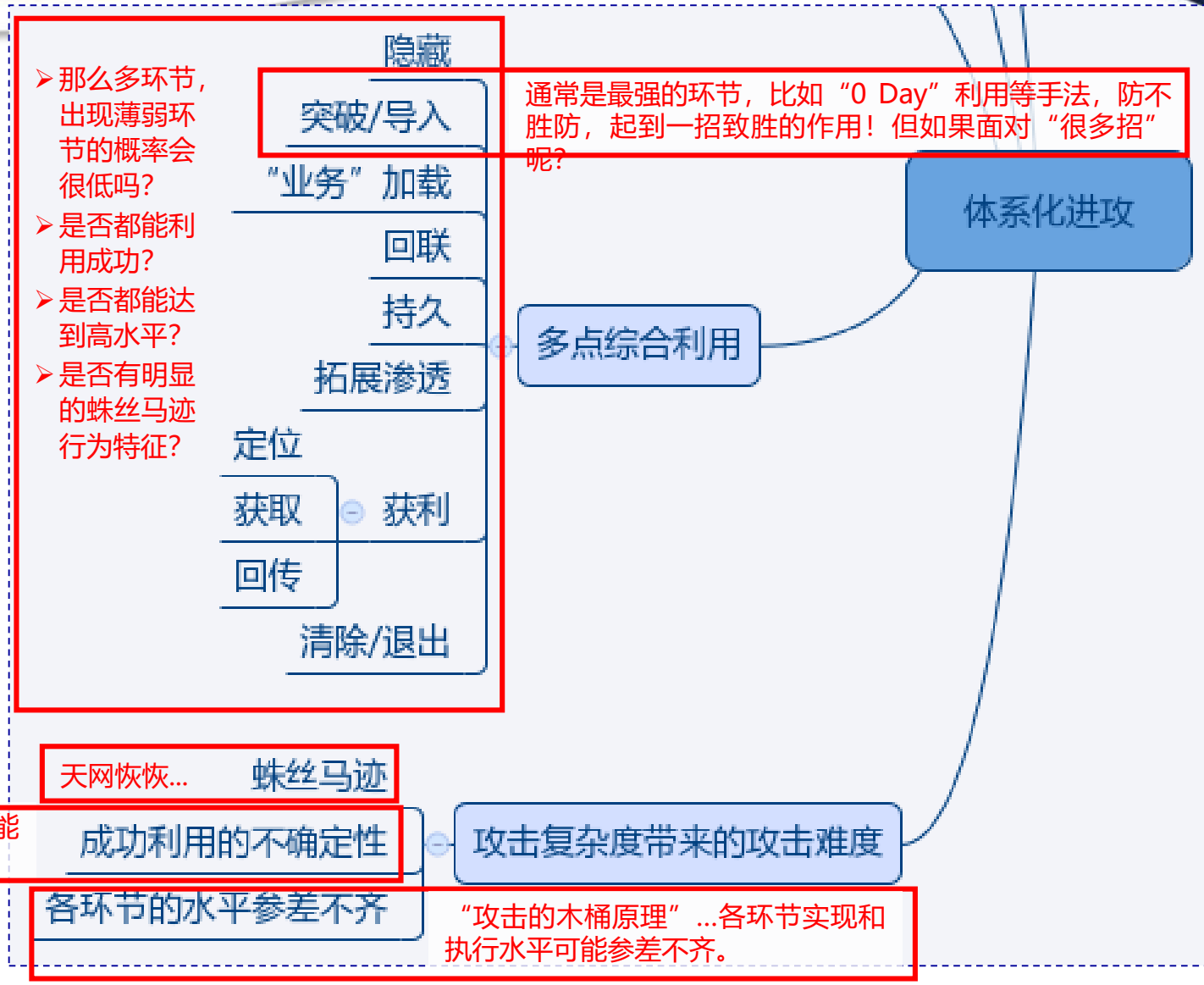
VPN SPIN 9 Dataflows



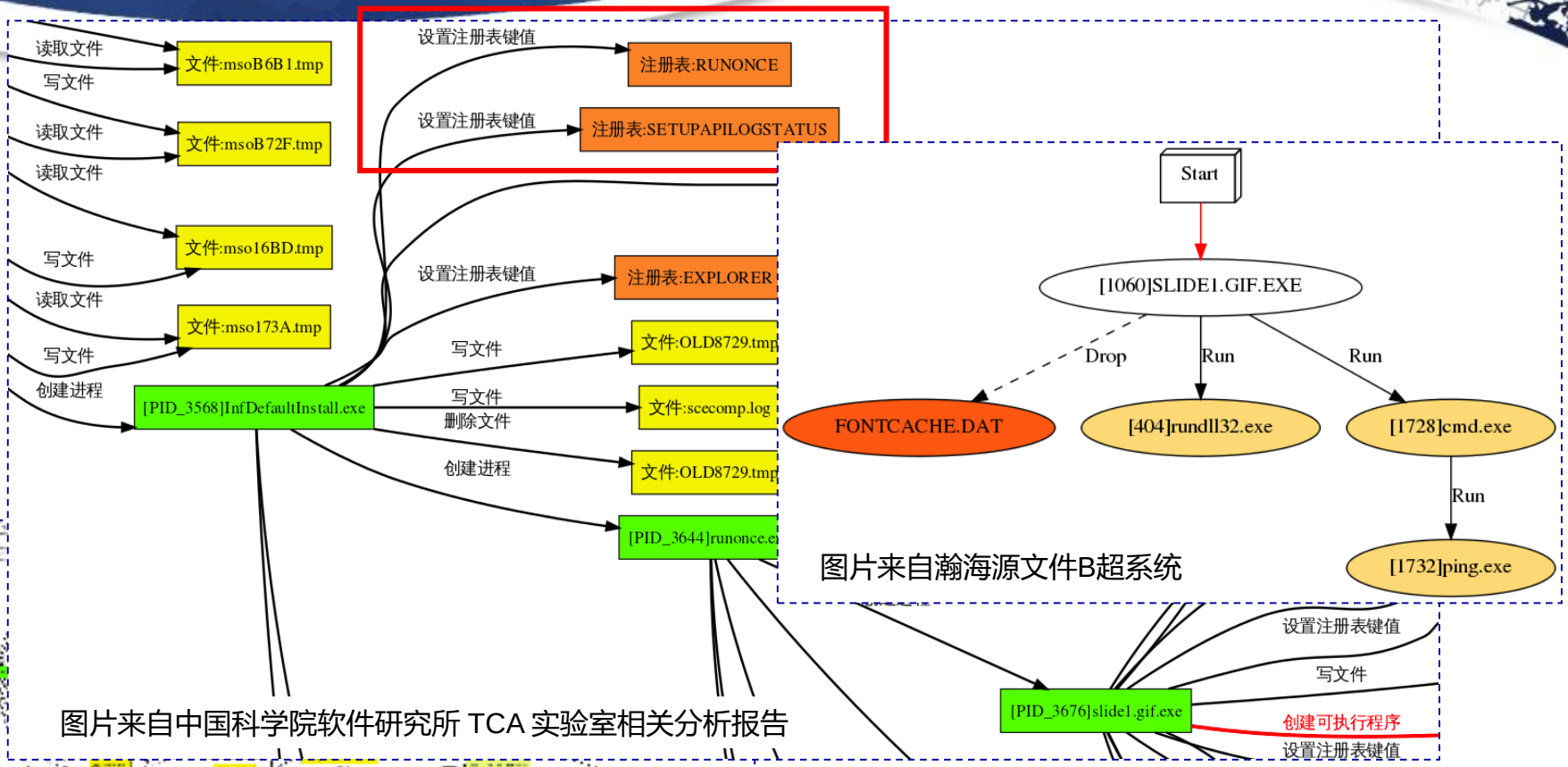
TOP SECRET//COMINT//20320108

25

体系化攻击中的薄弱环节

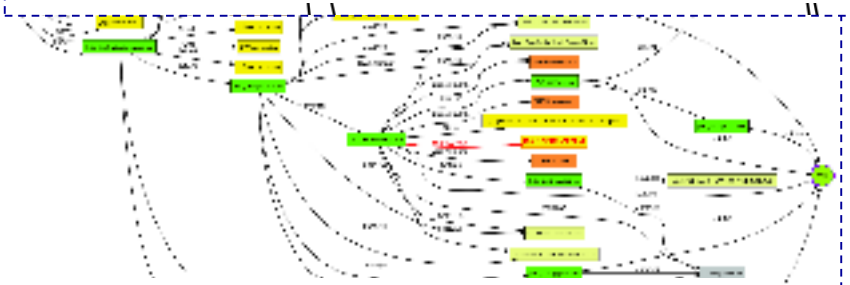


体系化攻击中的薄弱环节



图片来自瀚海源文件B超系统

图片来自中国科学院软件研究所 TCA 实验室相关分析报告



图片来自中国科学院软件研究所 TCA 实验室相关分析报告

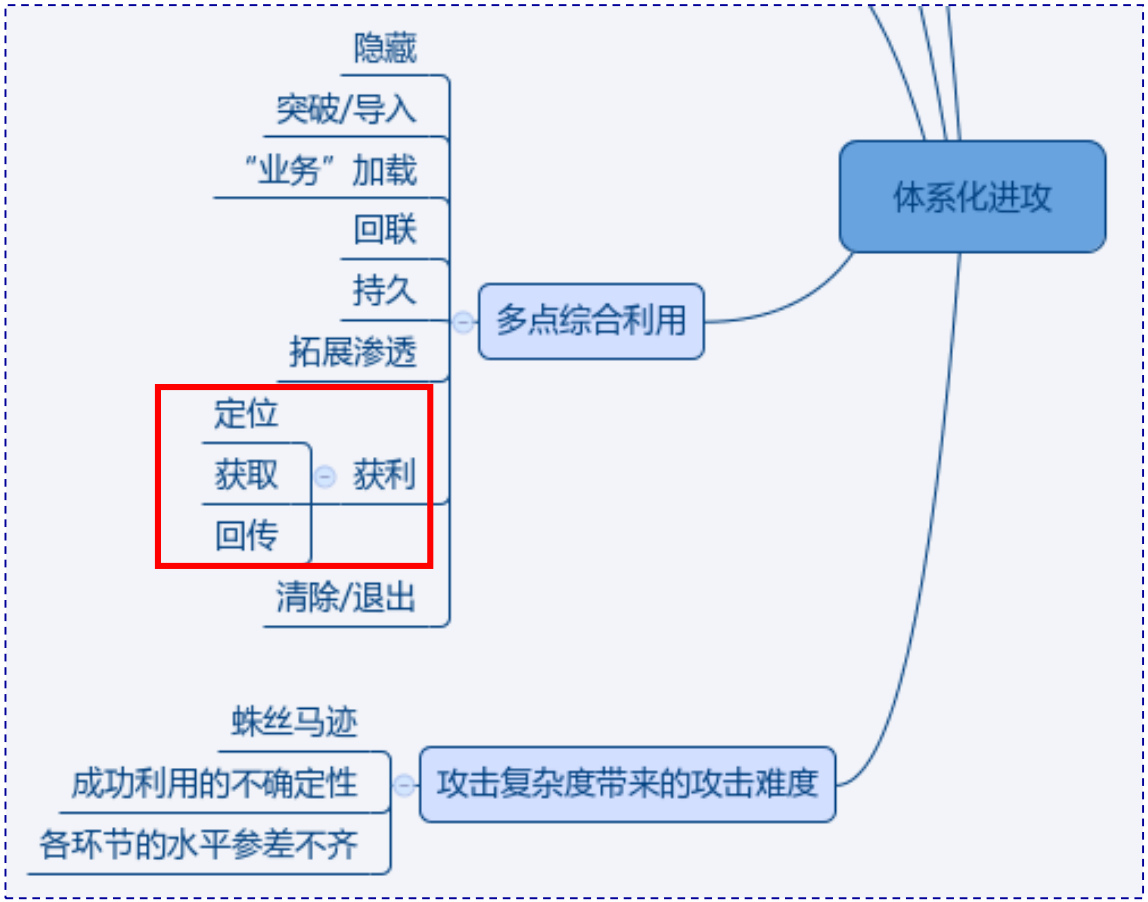
与利用CVE-2014-4114漏洞并在当时多个沙箱检测引擎中未触发告警的“突破/导入”环节相比，SandWorm样本中第二阶段的“加载”环节显得相当薄弱，特别是出现了明显会触发报警的行为。

窃取信息攻击的关键 — 获利环节

在分析窃取信息为目的的攻击并设计防御措施时，特别需要关注“窃取”这个获利环节。

定位寻找有价值的信息，读取访问获得目标信息，以及各种渠道回传窃取的信息，从而实现攻击的目的。

如果没有获利环节，一次针对信息系统的攻击可能是没有效益的。另一方面，在整个攻击过程中，获利环节的隐匿性可能是最低的，而且由于攻击产业链的信任关系问题，其执行水平可能也是最差的！

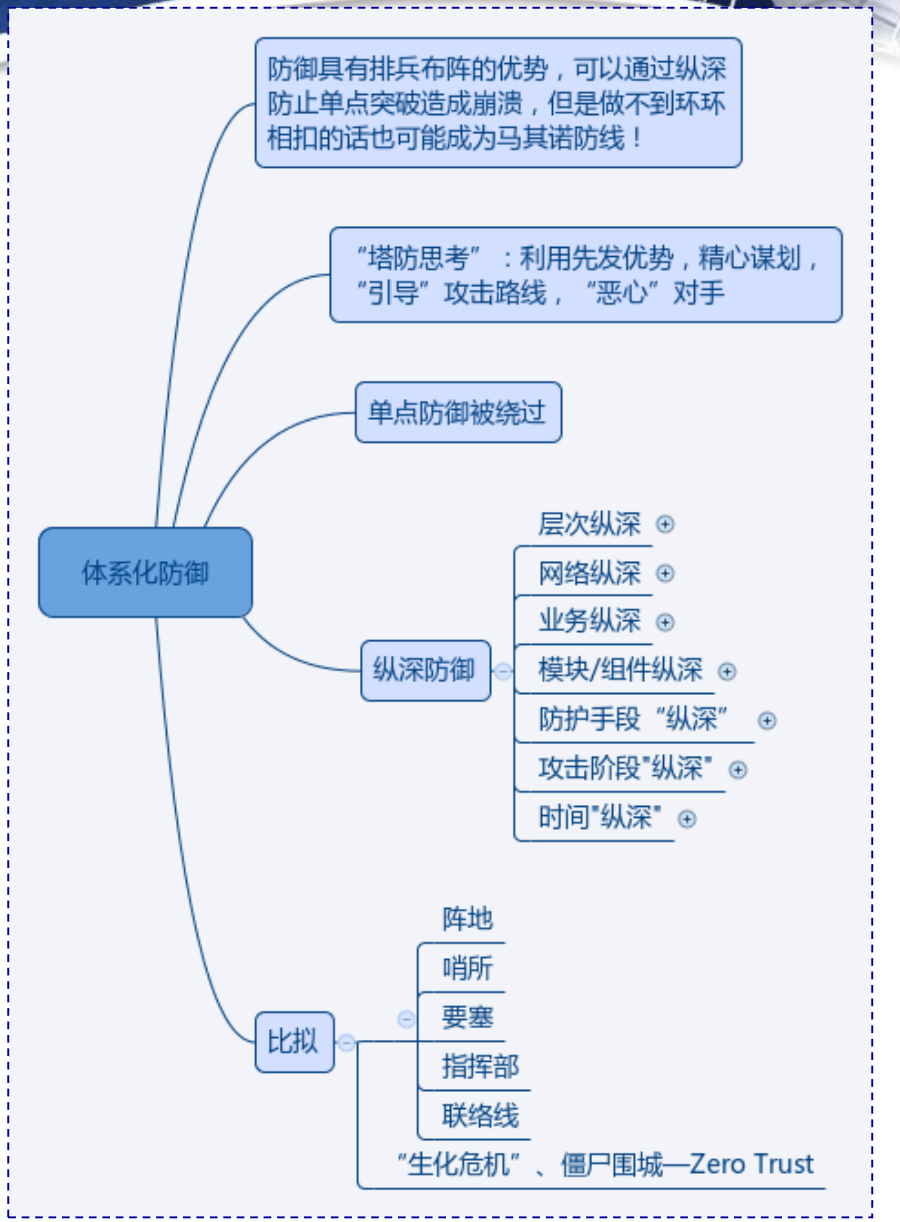


面对体系化的进攻，防守方如何应对?

攻击者已经联合起来了，已经形成分工合作的生态圈了，此时如果防守还是孤立、静态而且不成体系的，毋庸置疑成功者会是攻击者!

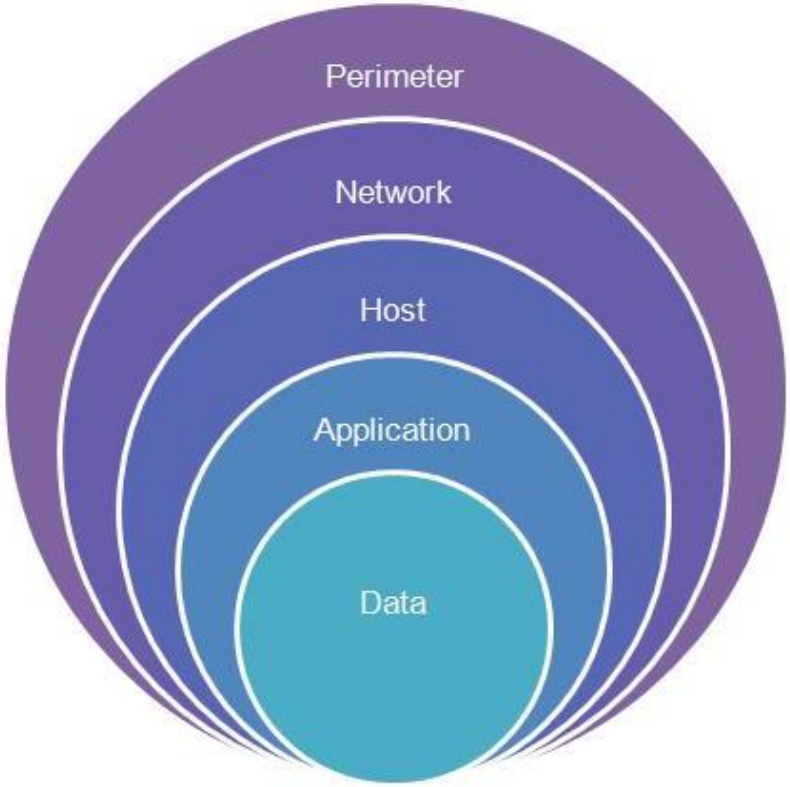
防守者只能利用好先发优势，布置好层层防线，综合利用多样化的手段，让进攻者在防守者布局的环境中“挣扎”！

传统的“纵深防御”原则，以及近些年火爆的“Moving Target Defence”，都强调体系化的防护!作为重要信息系统的防守方，我们**只能**抛弃“一招制敌”的幻想，“排兵布阵”构造**纵深防御**战线，体系化地与进攻者对抗。

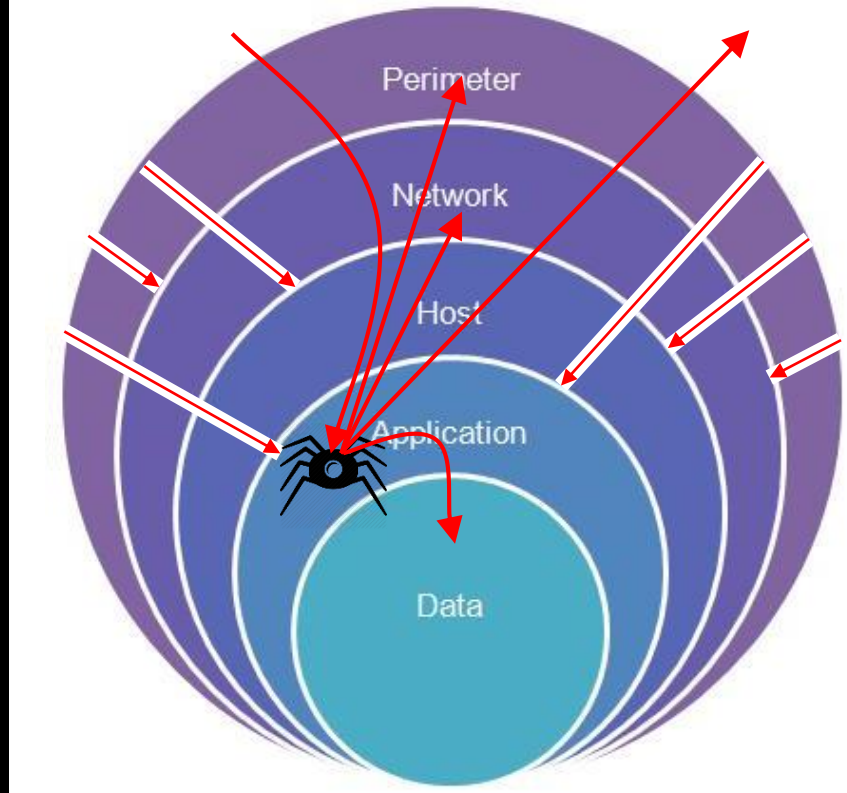


防御早就体系化了？ — 纵深防御 or 多层堆砌

理想目标

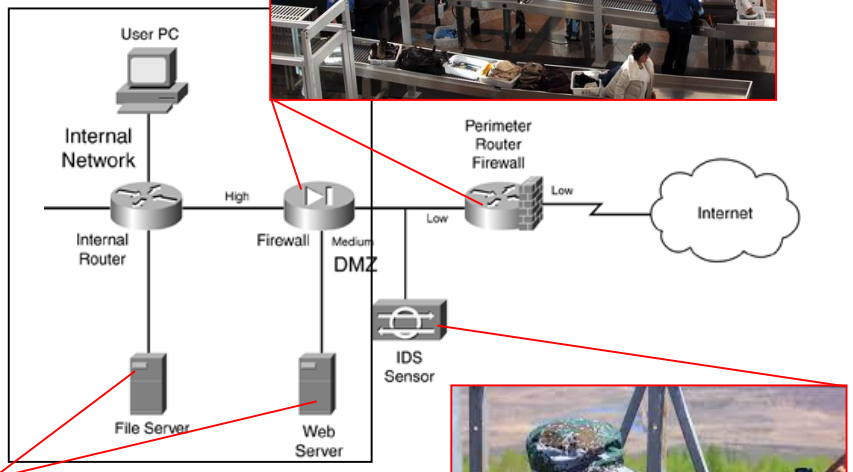


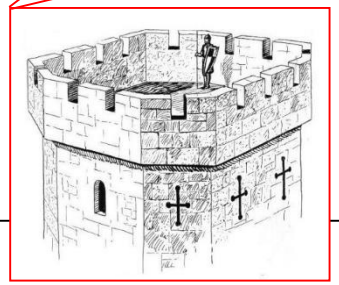
无奈现状



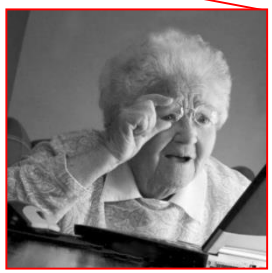
防御早就体系化了？ — 纵深防御 or 多层堆砌

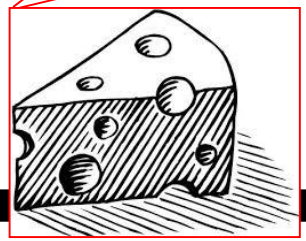
理想目标



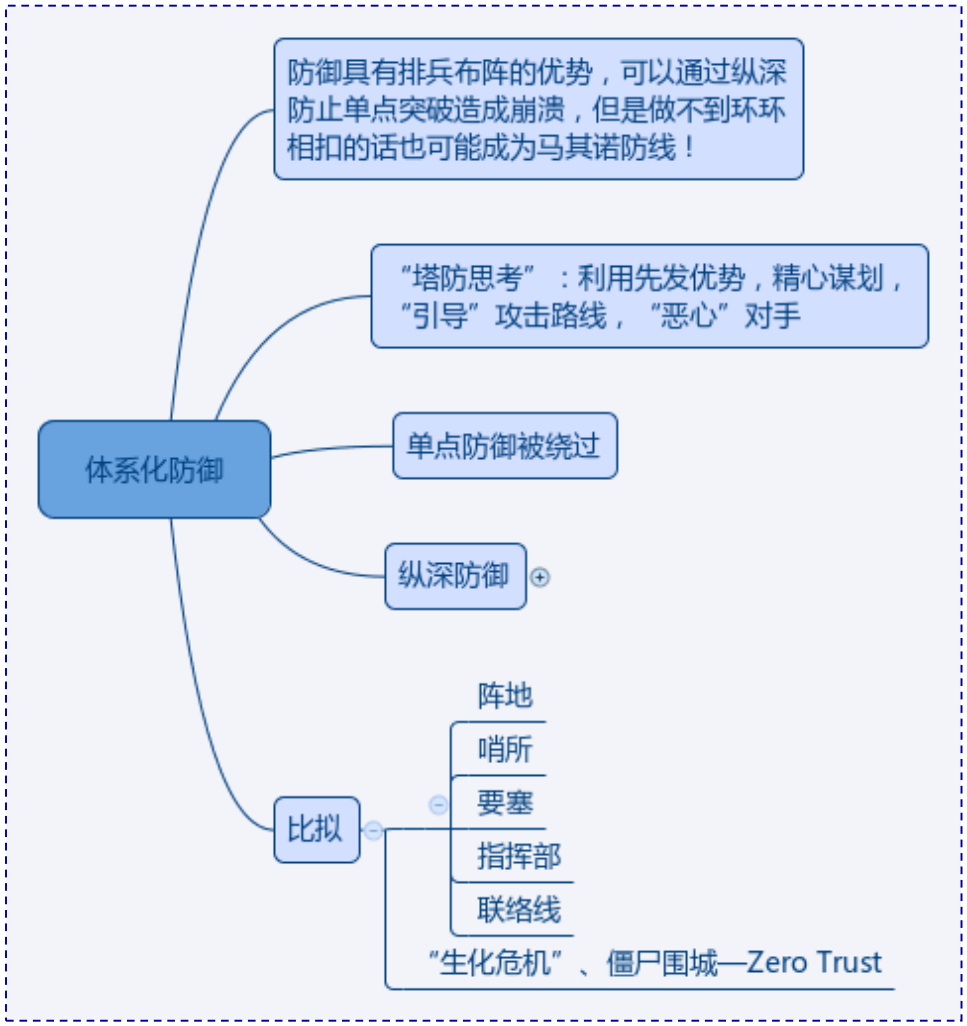


无奈现状





来自“塔防游戏”的启示 — 体系化的纵深防御

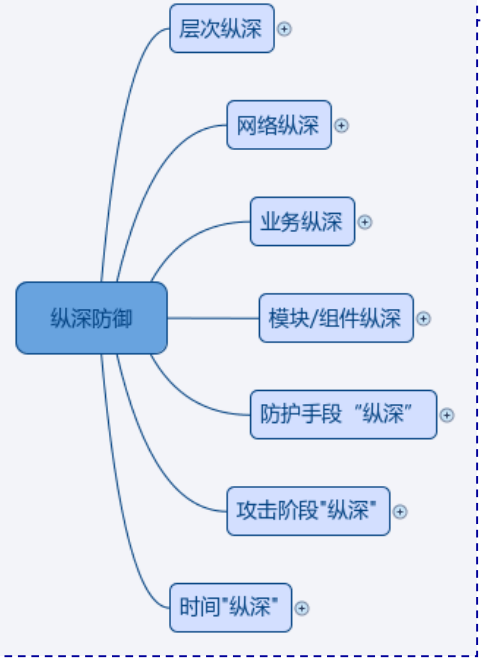


纵深防御“解决方案”？

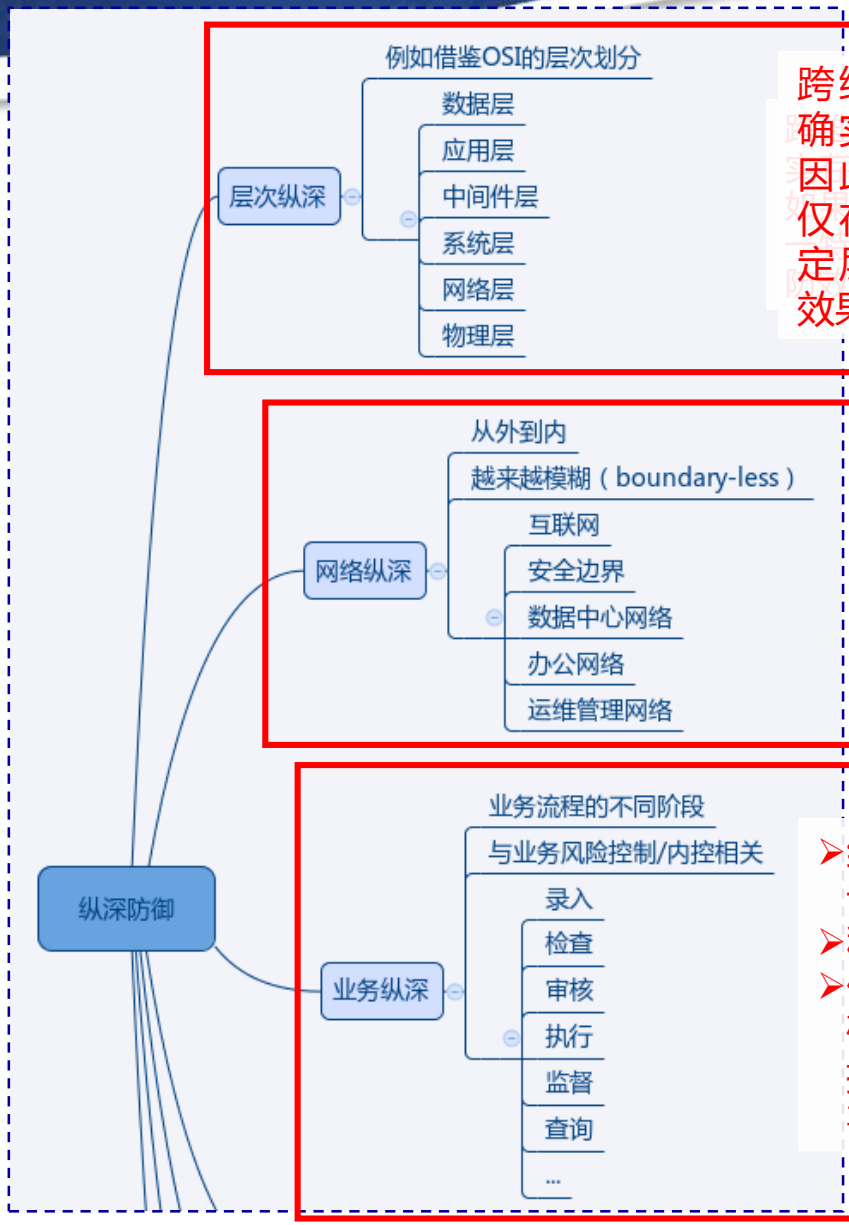
照理此处应该出现一张
“纵深防御整体解决方案架构图”

但是“纵深防御”是一种应该体现在信息安全防御体系设计各个方面的基本原则，而不是一种“可以独立堆叠形成的解决方案”。

为了对抗体系化的攻击，防御体系的设计应用好“先发优势”，针对威胁行为模式，在各个方面防御手段的考虑纵深覆盖。



多样化的纵深（1/9）

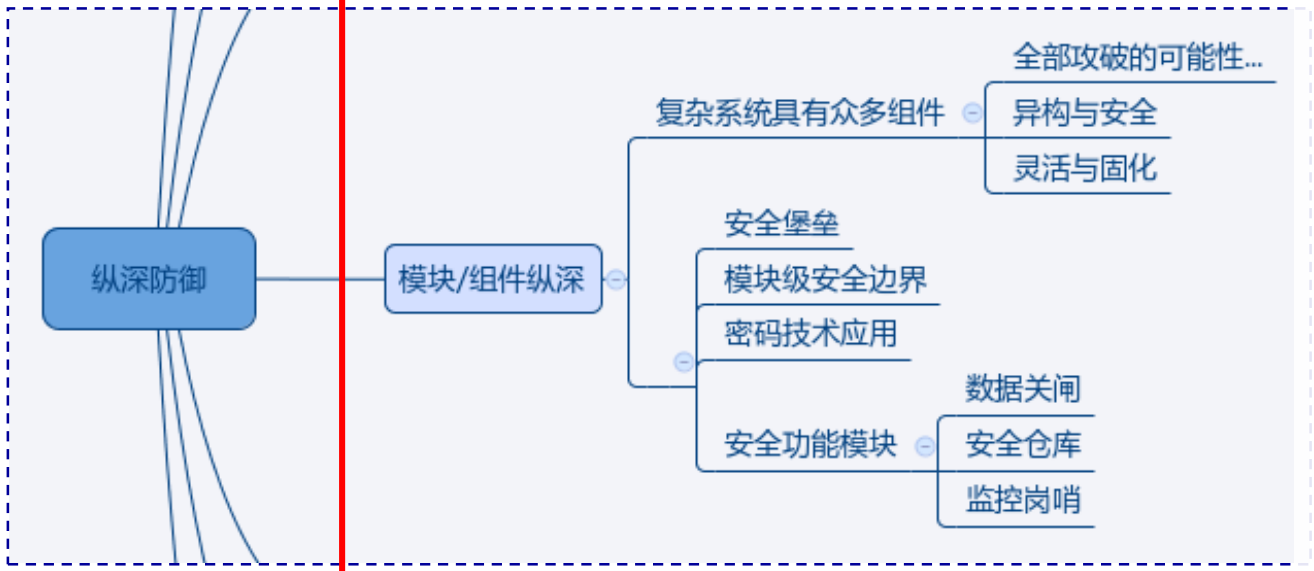


跨维度攻击确实存在，因此如果仅仅在某一特定层次设防效果有限。

网络纵深最传统，是排兵布阵的基础，但要打破“唯安全边界论”难度也不小！

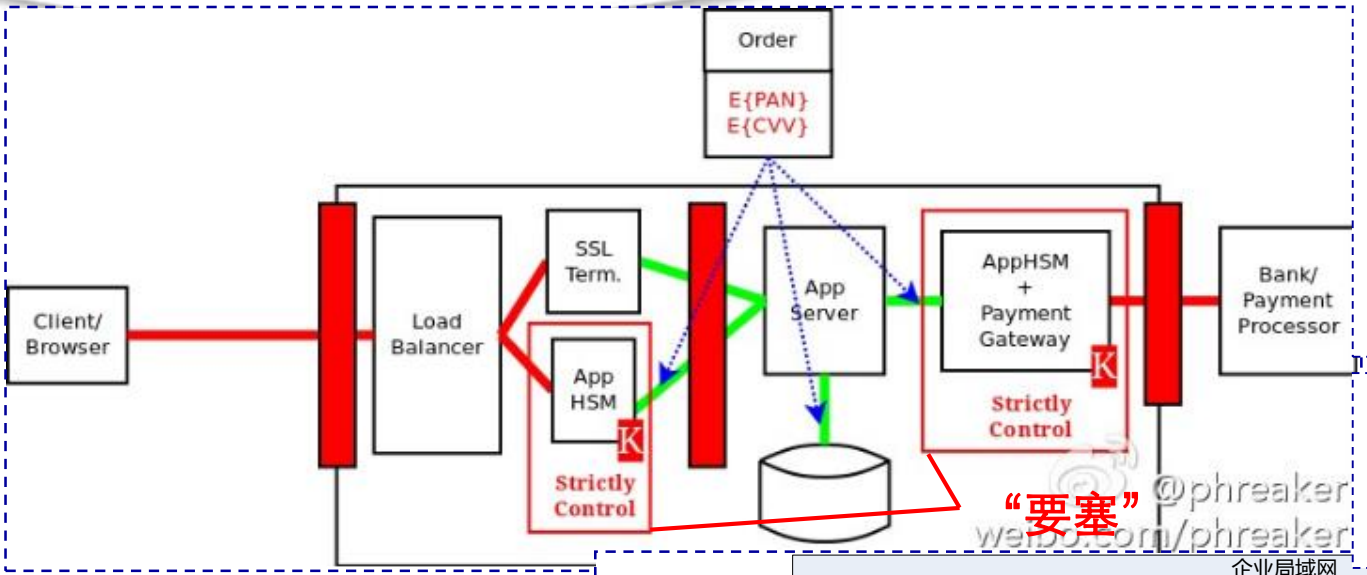
- 结合业务流程设置多道防线，有助于阻断攻击获利环节！
- 秘密信息的解密环节可重点防护；
- 信息系统在加密等防御保护措施基础上，对解密操作等行为的重点监控，可能给攻击获利环节造成难度，甚至形成威慑效果。

多样化的纵深（2/9）



- 根据经验，任何单一节点的系统都有被攻破的可能性；
- 现实中业务系统往往是多节点复杂系统，由相对独立的多模块组成；
- 借鉴MTD基于多样化和动态配置的防御原理，构建多模块纵深防御体系；
- 根据各业务功能模块业务特点不同，在保留一定灵活性的情况下进行加固，并有机结合安全功能模块，降低攻击者“通打”的可能性；
- 对安全功能模块进行功能裁剪和高强度加固，将攻击面降低至最小，形成可依靠“要塞”；
- **借鉴MTD的思路，分析攻击者可能的进攻路径，基于“Design for failure”原则，设计多样化多层次防御“后手”。**

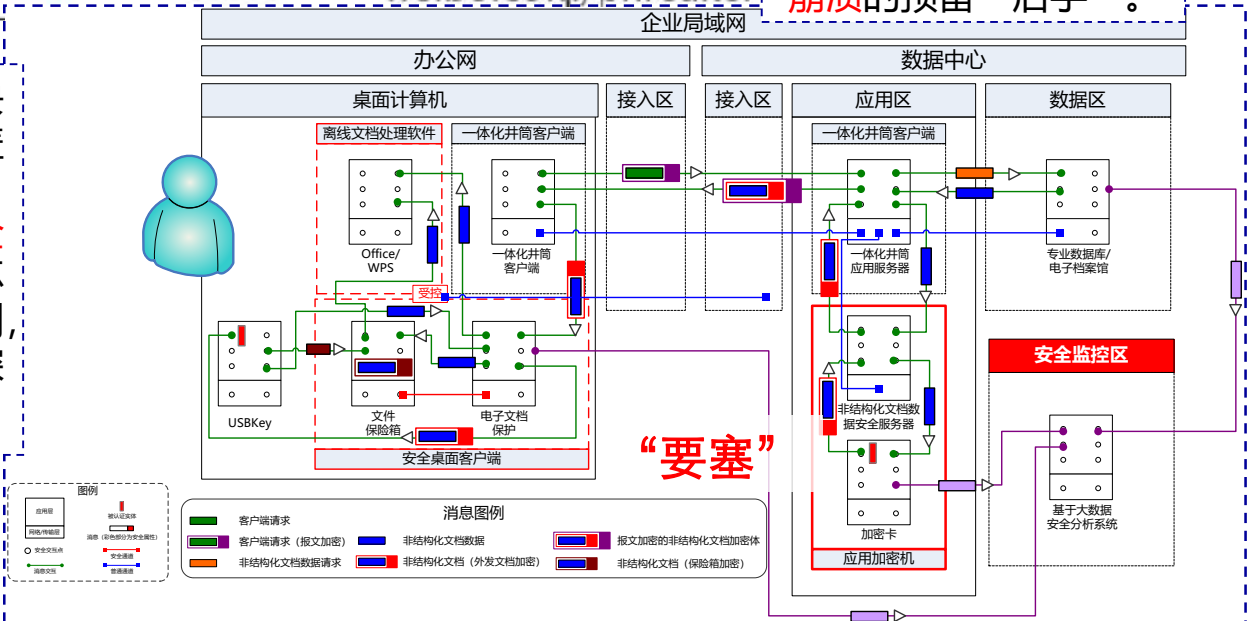
多样化的纵深（3/9）— “模块/组件纵深”覆盖



例如在多组件构成的支付处理系统中，可以引入内置应用逻辑并强加固的密码设备作为**“安全要塞”**组件，降低其它业务功能组件的加固要求，在保持系统灵活性的同时实现安全防护。

“要塞” — 在其它业务功能组件被攻破的情况下，依然能避免整体崩溃的预留“后手”。

多组件系统实现“模块/组件纵深”防御覆盖时必须实现可信可靠、环环相扣的组件间安全交互机制。只有做到环环相扣的安全交互机制，才能确保实现的是纵深防御而不是多层堆叠。

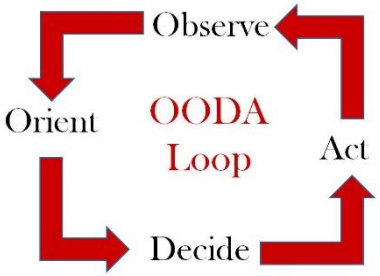


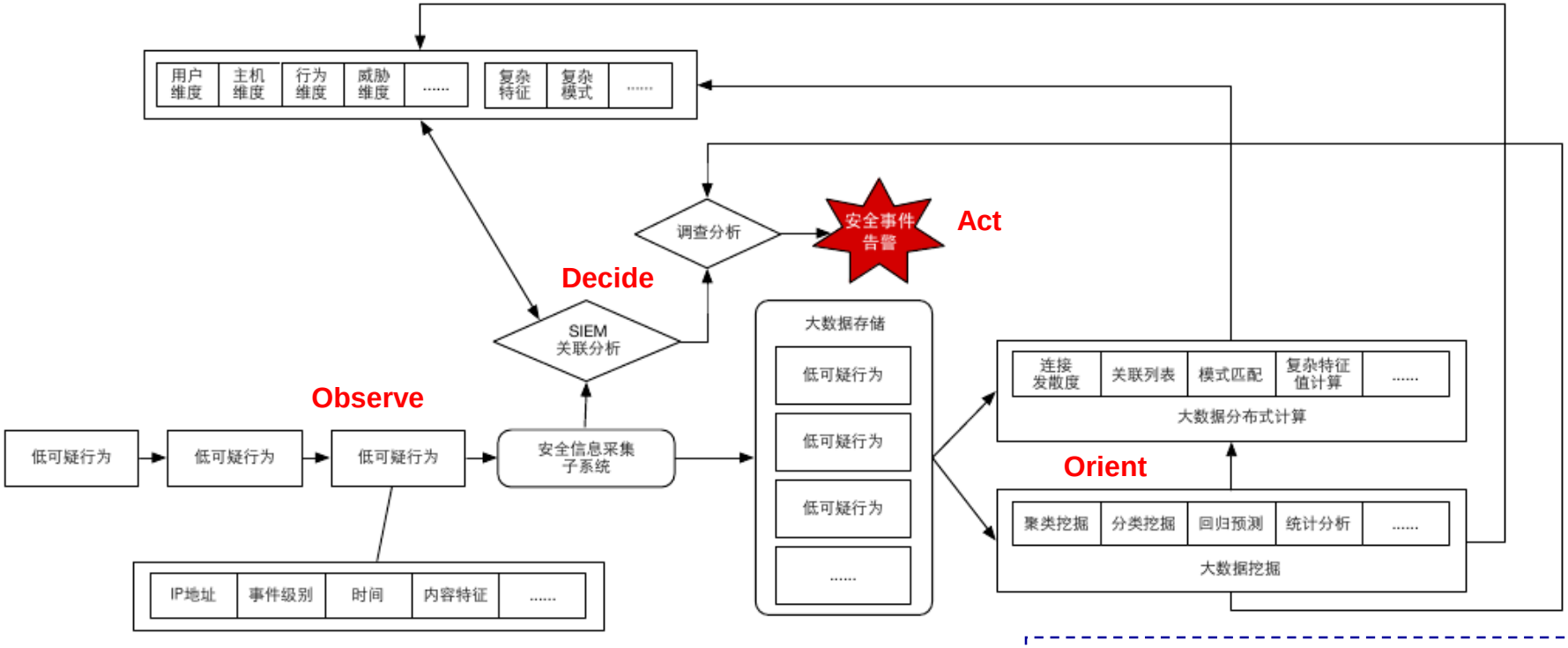
多样化的纵深（4/9）


纵深防御
防护手段“纵深”
PDRR
“D+R”应成为底线
点面结合
点:如exploit、shellcode检测
面:如全面监控、多层次防御
攻击阶段"纵深"
有针对性地面向各个攻击阶段
博弈思考：敌人此时陷于什么状态？
Design for Failure
默认攻击各个阶段都可能成功
精心设计“后手”
时间"纵深"
以事件为中心
事前
事中
事后
系统生命周期
SDLC
Build Security In
运维过程
Observe
Orient
OODA Loop
Act
Decide

多样化的纵深（5/9）— “攻击阶段纵深”与“事件纵深”

基于OODA循环的安全事件响应机制是实现“攻击阶段纵深”和“事件纵深”覆盖的关键。采用大数据技术增强现有的安全监控与事件管理体系，有可能大幅提高对攻击及事件的“广度与纵深覆盖”能力。

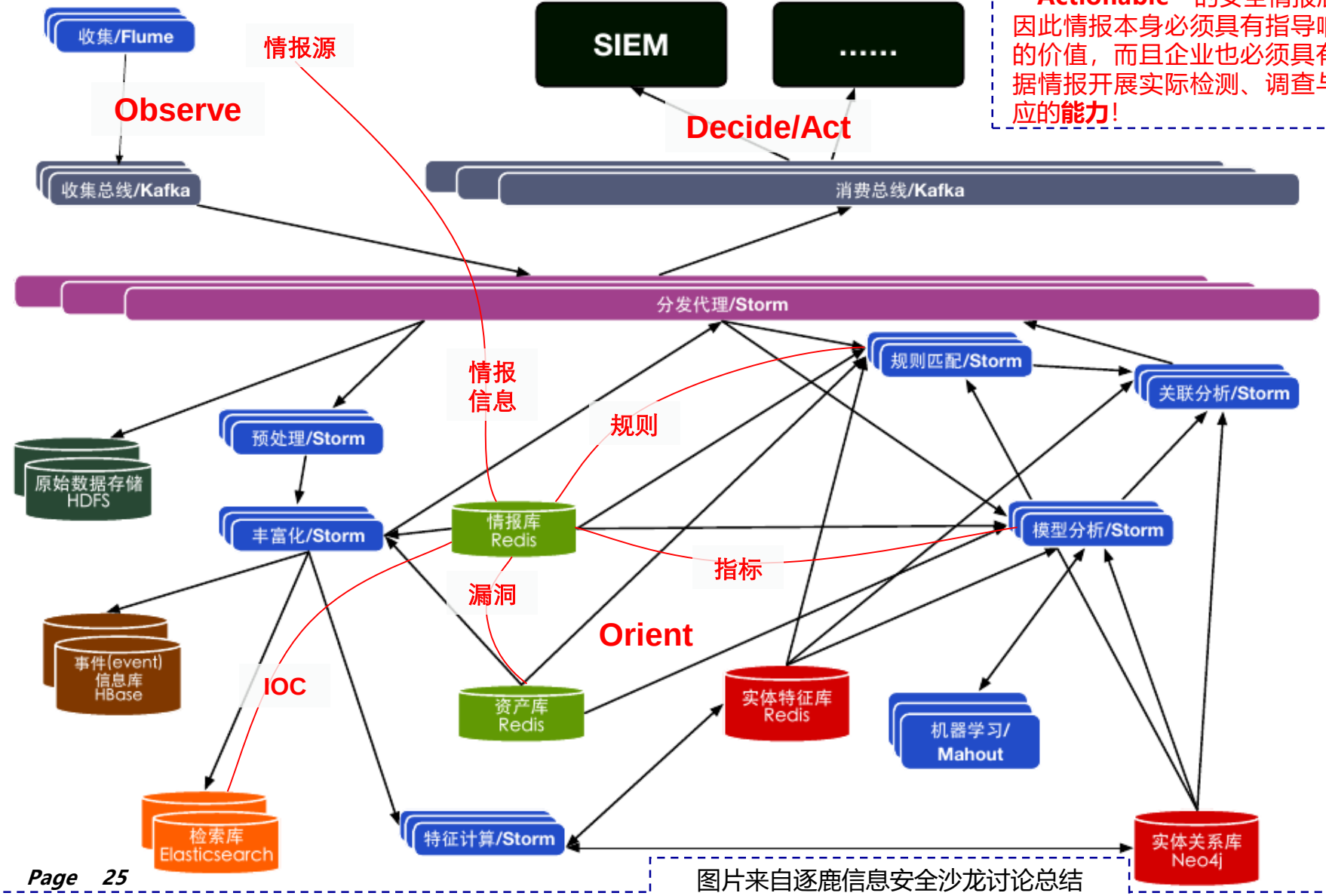




图片来自逐鹿信息安全沙龙讨论总结

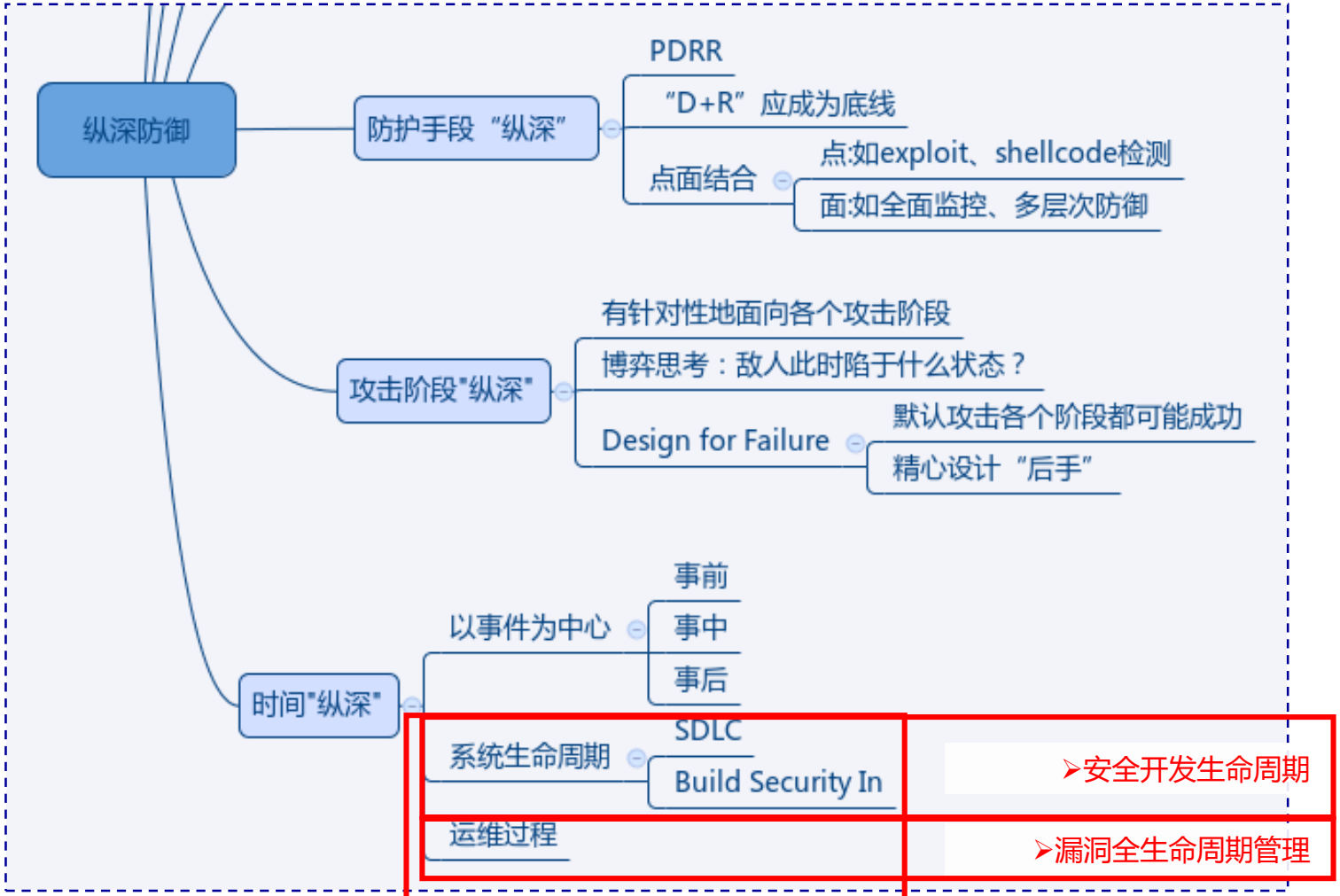
多样化的纵深（6/9）— “攻击阶段纵深”与“事件纵深”

关于**安全情报共享**—核心是围绕“**Actionable**”的安全情报展开，因此情报本身必须具有指导响应的价值，而且企业也必须具有根据情报开展实际检测、调查与响应的**能力**！

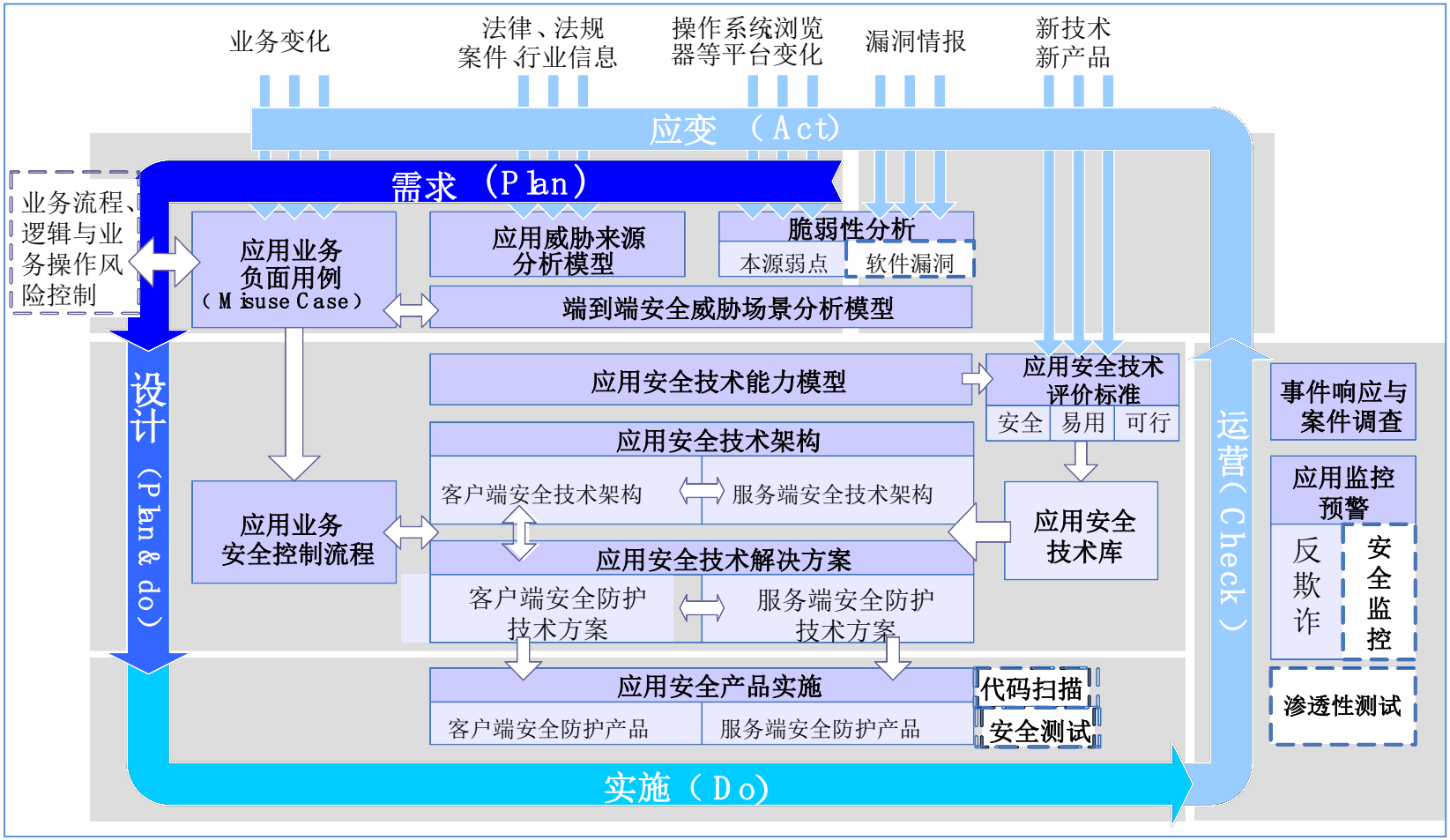


图片来自逐鹿信息安全沙龙讨论总结

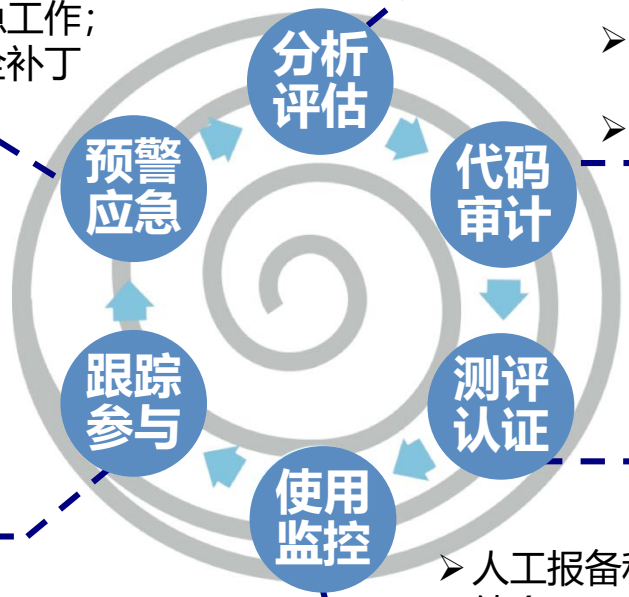
多样化的纵深（7/9）


纵深防御
防护手段“纵深”
PDRR
“D+R”应成为底线
点面结合
点:如exploit、shellcode检测
面:如全面监控、多层次防御
攻击阶段"纵深"
有针对性地面向各个攻击阶段
博弈思考：敌人此时陷于什么状态？
Design for Failure
默认攻击各个阶段都可能成功
精心设计“后手”
时间"纵深"
以事件为中心
事前
事中
事后
系统生命周期
SDLC
Build Security In
运维过程
➢安全开发生命周期
➢漏洞全生命周期管理
➢围绕应用的深度加固

多样化的纵深（8/9）— “系统生命周期纵深”覆盖



多样化的纵深（9/9）— “运维过程纵深”覆盖



- 分析开源软件机制、原理与协议安全性；
- 分析架构安全性等；
- 整体评估可控性。

- 自动化与手工代码审计；
- 关注实现安全性。

- 鼓励开源社区或应用行业送测开源软件；
- 尤其适用于开源安全软件。

- 人工报备和自动化发现相结合；
- 掌控开源软件使用情况。

- 积极主动参与开源与安全社区
- 跟踪了解基础开源软件的安全动态；
- 及早获知漏洞信息。

- 结合使用情况及安全情报信息，及早发布预警；
- 指导安全团队开展虚拟补丁、攻击取证等应急工作；
- 支持系统团队打安全补丁或升级。

总结

- 需要切换到积极地体系化防御模式
- 体系化的攻击虽然强大但无法消除薄弱环节
- 用好“先发”优势实现体系化纵深防御
- 纵深覆盖需要落实在防御的各个方面
- 亟待更多覆盖方面、更体系化的纵深防御

感谢聆听