



Microsoft Online Tech Forum

微软在线技术峰会

如何通过 SDL 和 SecDevOps 实现软件及
应用的原生安全

朱长明
安全架构师



如何通过 SDL 和 SecDevOps 实现软件及应用的 原生安全

最佳安全实践介绍

Agenda

- SDL是什么
 - SDL的安全实践
 - SDL和DevOps的融合
- 

SDL是什么

什么是安全开发周期?

●是什么!

- 通过最佳实践来提高软件开发生命周期的过程, 以提高软件安全性
- 试图解决安全问题并减少不安全软件的成本的尝试。

●不是什么!

- SDL并非无所不能, 彻底的治本!



Microsoft安全历史

2002-2003

- 比尔·盖茨在2002年初撰写“可信赖计算”备忘录 memo early 2002
- 全推送和FSR扩展到其他产品

2004

- 高级领导团队同意对所有产品要求SDL
 - 面临重大风险和/或
 - 处理敏感数据

2005-2007

- 添加了SDL增强功能
 - “模糊测试”，代码分析，密码设计要求
 - 隐私，禁止的危险函数等...
- Windows Vista是第一个通过完整SDL的操作系统

2008-2015

- 通过反馈，分析和自动化来优化流程
- 开始对外宣传SDL

2015-2019

- 更广泛的宣传，开发统一知识库系
- 统一合规系统的介绍和开发
- 安全性已完全集成到DevOps（DevSecOps）中
- 更多的安全自动化
- 工具轻量化

为什么采纳SDL

网络犯罪演变

1986-1995



局域网
第一台PC病毒
动机: 损害

1995-2003



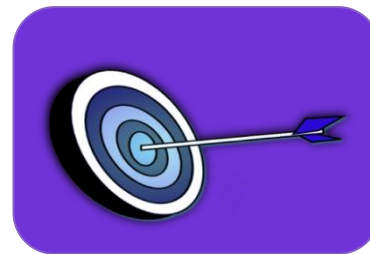
互联网时代
“大蠕虫”
动机: 损害

2004+



操作系统, 数据库攻击
间谍软件, 垃圾邮件
动机: 财务

2006+



针对性攻击
社会工程学
金融+政治

2009+



关键基础设施攻击
间谍
网络战

→ **Cost of U.S.
cybercrime:
> \$100B**

2007 Market prices:

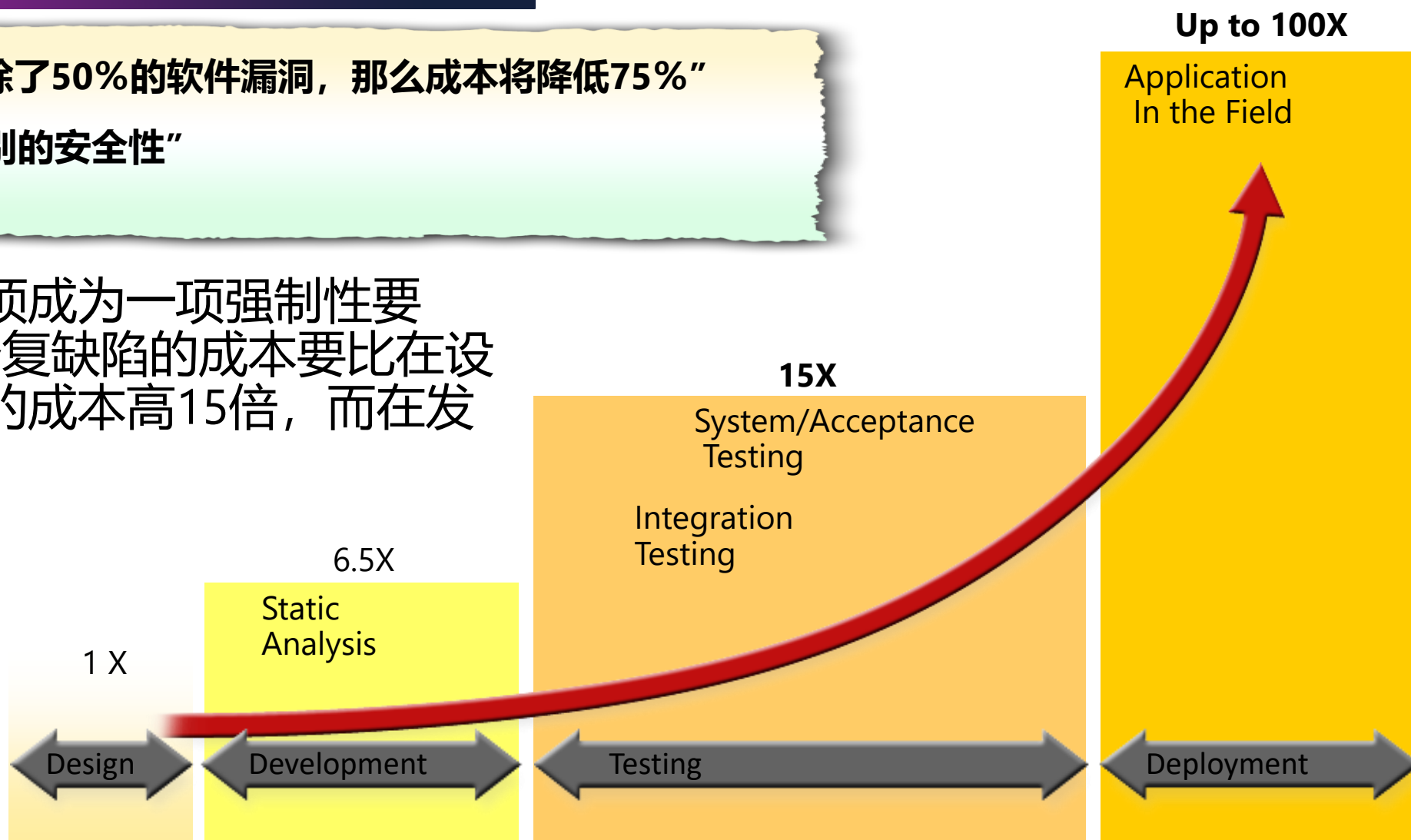
Credit Card Number	\$0.50 - \$20
Full Identity	\$1 - \$15
Bank Account	\$10 - \$1000

被动安全的代价

“如果在生产之前仅消除了50%的软件漏洞，那么成本将降低75%”

-Gartner“应用程序级别的安全性”

交付安全应用程序必须成为一项强制性要求.....在集成过程中修复缺陷的成本要比在设计时检测和消除缺陷的成本高15倍，而在发布时则要高达100倍。



SDL的安全实践

实践#1 - 提供培训

- 安全是每个人的工作。开发人员、服务工程师以及计划和产品经理必须了解安全基础知识，并知道如何将安全性构建到软件和服务中，使产品更安全，同时仍满足业务需求并提供用户价值。
- 有效的培训将补充和重新实施安全策略、SDL 实践、标准和软件安全要求，并遵循通过数据或新可用的技术能力获得的见解。
- 尽管安全是每个人的工作，但重要的是要记住，不是每个人都需要成为安全专家，也不是努力成为熟练的渗透测试人员。但是，确保每个人都了解攻击者的观点、目标以及可能的艺术将有助于吸引每个人的注意力，并提高集体知识标准。

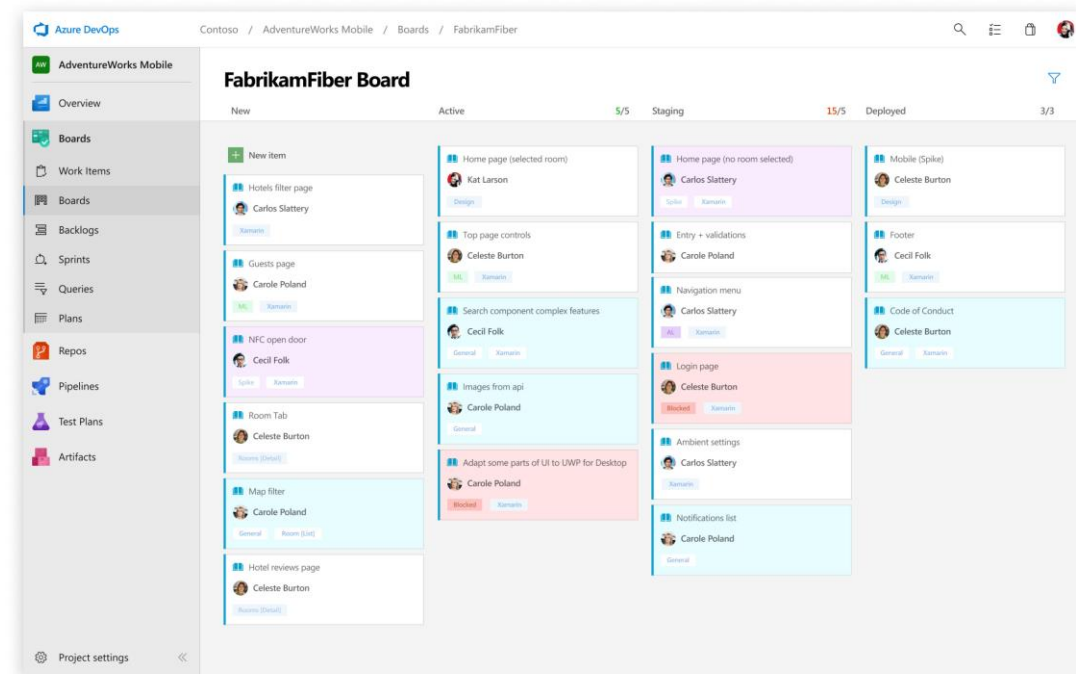
实践#2 - 定义安全要求

需要考虑安全和隐私是开发高度安全的应用程序和系统的一个基本方面，无论使用何种开发方法，都必须不断更新安全要求，以反映所需变化功能和威胁环境的变化。显然，定义安全要求的最佳时间是在初始设计和规划阶段，因为这允许开发团队以最小化中断的方式集成安全性。

影响安全要求的因素包括（但不限于）法律和行业要求、内部标准和编码实践、对以前事件的审查以及已知的威胁。这些要求应通过工作跟踪系统或通过从工程管道派生的遥测来跟踪。

跟踪看板、积压工作、团队仪表板和自定义报告的工作

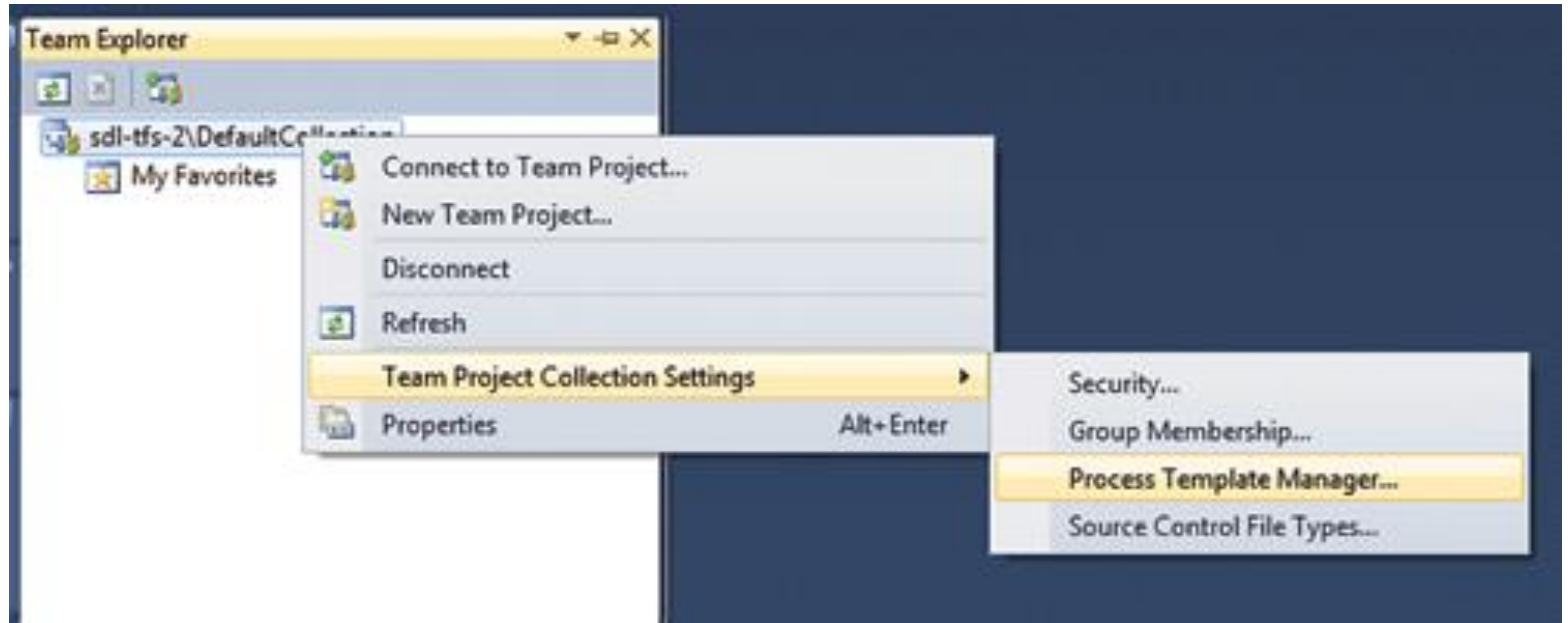
- 将拖放冲刺（sprint）规划和灵活的工作项目跟踪与全面的可追溯性相结合，为您的所有想法（大小）提供完美的家。



实践#3 - 定义指标和合规性报告

- 必须确定可接受的安全质量最低级别，并让工程团队负责满足这些标准。尽早定义这些可帮助团队了解与安全问题相关的风险，在开发过程中识别和修复安全缺陷，并将标准应用于整个项目。设置有意义的 Bug 栏需要明确定义安全漏洞的严重性阈值（例如，使用"严重"或"重要"严重级别发现的所有已知漏洞都必须在指定的时间范围内进行修复），并且在设置后永远不会放松它。
- 为了跟踪关键性能指标（KPI）并确保安全任务的完成，组织（如 Azure DevOps）使用的错误跟踪和/或工作跟踪机制应允许安全缺陷和安全工作项明确标记为安全，并标有相应的安全严重性。这允许准确跟踪和报告安全工作。

向Azure DevOps/TFS添加bugbar



· 创建安全流程时需要考虑的基本标准

关键

服务器摘要：网络蠕虫或服务器"拥有"的*不可避免*的情况。

- 特权提升：执行任意代码或获得比授权权限更多的权限的能力
 - 远程匿名用户
 - 例子：
 - 未经授权的文件系统访问：任意写入文件系统
 - 执行任意代码
 - SQL 注入（允许代码执行）
 - 所有写入访问冲突（AV）、可利用的可利用的可访问 AV 或远程匿名可调用代码中的整数溢出

重要

服务器摘要：非默认关键方案或存在有助于*预防*关键情况的缓解情况。

- 拒绝服务：必须通过发送少量数据"易于利用"，否则必须快速诱导
 - 匿名
 - 持久 DoS
 - 例子：
 - 发送单个恶意 TCP 数据包会导致死亡蓝屏（BSOD）
 - 发送导致服务故障的少量数据包
 - 带放大的临时 DoS
 - 例子：
 - 发送少量数据包，导致系统在一段时间内无法使用
 - Web 服务器（如 IIS）停机一分钟或更长时间
 - 单个远程客户端通过建立会话并保持打开会话来消耗服务器上的所有可用资源（会话、内存）

SDL 隐私错bugbar (示例)

· 创建隐私过程时需要考虑的基本标准

关键

- 缺乏通知和同意

示例：在传输之前，在 UI 中没有明确通知和明确选择加入同意的情况下，从用户系统中传输敏感的个人身份信息（PII），
- 缺少用户控件

示例：正在进行的非必需 PII 的收集和传输，而 UI 中用户无法停止后续收集和传输。
- 缺乏数据保护

示例：PII 被收集和存储在持久性常规数据库中，没有允许用户访问和更正存储的 PII 的身份验证机制。
- 缺乏儿童保护

示例：对于对儿童具有吸引力或针对儿童的网站或服务，不会收集年龄，网站收集、使用或披露用户的 PII。
- 不当使用 Cookie

示例：存储在 Cookie 中的敏感 PII 未加密。
- 缺乏内部数据管理和控制

示例：对存储在组织中 PII 的访问不仅仅限于那些有效业务需求的用户，或者在不再需要访问后没有撤销访问的策略。
- 法律控制不足

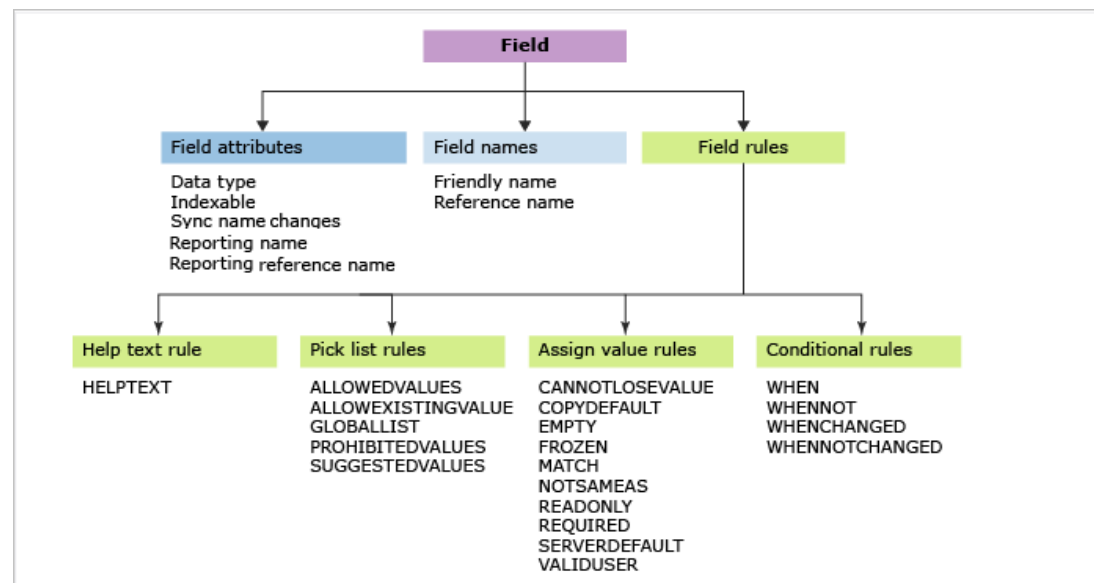
示例：产品或功能将数据传输到尚未签署合法合同的代理或独立第三方。

重要

- 缺乏通知和同意

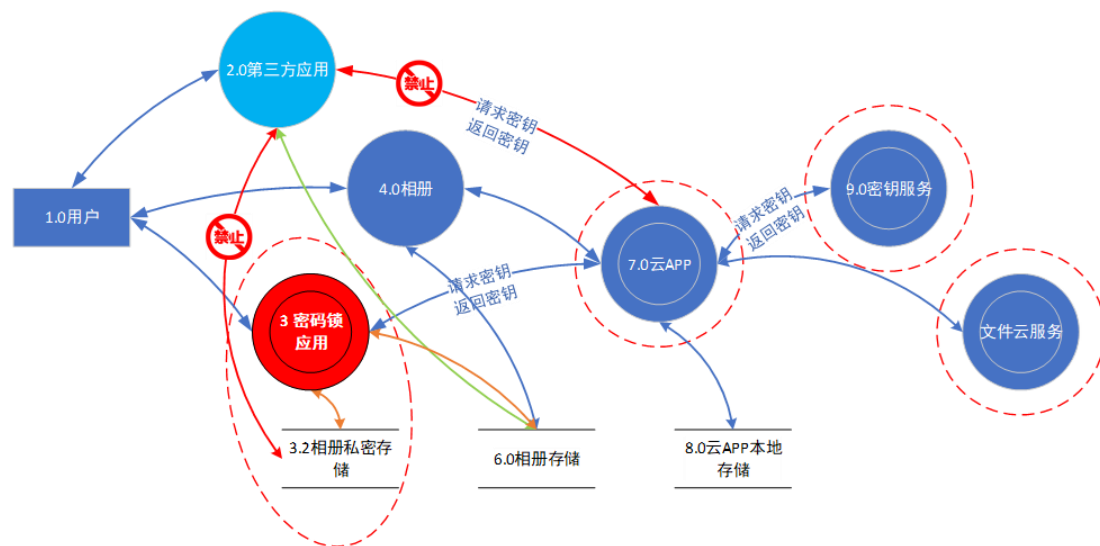
添加或修改字段以跟踪工作

- Azure DevOps 服务器 2019
|TFS 2018 |TFS 2017 |TFS
2015 |TFS 2013



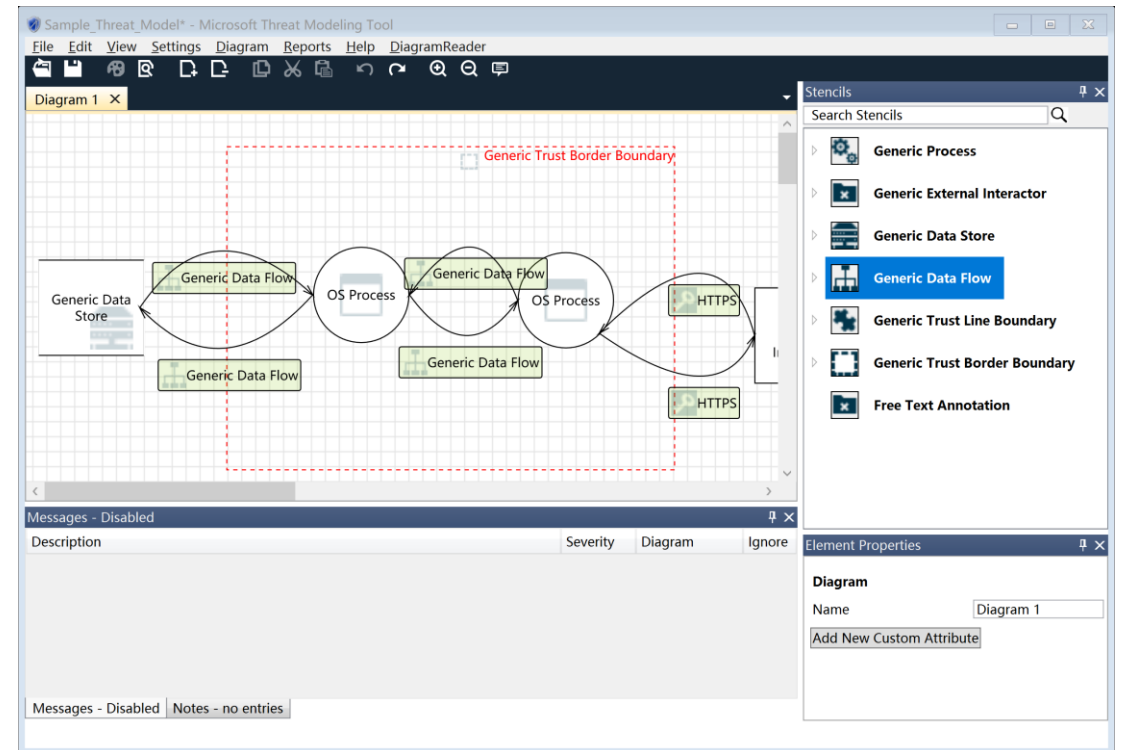
练习#4 - 执行威胁建模

- 威胁建模应在存在有意义的安全风险的环境中使用。威胁建模可以在组件、应用程序或系统级别应用。这种做法允许开发团队考虑、记录和（重要）在其计划的运营环境中结构化方式讨论设计的安全影响。
- 将结构化方法应用于威胁方案有助于团队更有效、更便宜地识别安全漏洞，确定这些威胁的风险，然后进行安全功能选择并建立适当的缓解措施。



微软威胁建模工具

- Microsoft 威胁建模工具通过可视化系统组件、数据流和安全边界的标准表示法，使所有开发人员的威胁建模更加轻松。它还可帮助威胁建模人员根据软件设计的结构识别他们应该考虑的威胁类别。我们设计了该工具时考虑到了非安全专家，通过提供创建和分析威胁模型的明确指导，使所有开发人员都更容易进行威胁建模。



实践#5 - 建立设计要求

- SDL 通常被视为帮助工程师实现"安全功能"的保证活动，因为功能在安全性方面经过精心设计。为此，工程师通常依赖于安全功能，如加密、身份验证、日志记录等。在许多情况下，安全功能的选择或实现已证明非常复杂，以至于设计或实现选择可能会导致漏洞。因此，至关重要的是，在一致地应用这些保护时，必须始终如一地应用这些保护。

实践#6 - 定义和使用加密标准

- 随着移动和云计算的兴起，确保所有数据（包括安全敏感信息以及管理和控制数据）在传输或存储时受到保护，防止意外泄露或更改至关重要。加密通常用于实现此目的。在使用加密的任何方面时做出错误的选择可能是灾难性的，最好制定明确的加密标准，提供有关加密实现的每个元素的详细说明。这应该留给专家。一个好的一般规则是只使用行业审查的加密库，并确保它们实现的方式允许在需要时轻松替换它们。

实践#7 - 管理使用第三方组件的安全风险

- 如今，绝大多数软件项目都是使用第三方组件（商业和开源）构建的。选择要使用的第三方组件时，了解其中的安全漏洞可能对集成它们的大型系统的安全性产生的影响非常重要。准确清点第三方组件，并在发现新漏洞时制定响应计划，将大有作为，以减轻这种风险，但应考虑进行额外的验证，具体取决于组织的风险偏好，使用的组件类型以及安全漏洞的潜在影响。

开源的好处

将开源软件作为开发过程的一部分有很多好处，其中一些包括：

- **增加上市时间。**通过将现有组件连接在一起，而不是从零开始实现所有组件，更快地创建软件。
- **质量更高。**所有软件组件都可能包含缺陷，但专注于专用软件组件通常比让许多工程师多次单独解决相同问题更高质量。
- **社区。**通过贡献新功能、报告 Bug 或与所使用的开源项目进行交互，您可以分担代码库的成本和收益。

正确管理此风险必须采取的最低步骤。

库存开源

- 了解正在使用哪些开源组件以及在哪里使用。

执行安全分析

- 确保所有已识别的组件没有安全漏洞。

使开源保持最新

- 使开源组件保持最新。

调整安全响应流程

- 准备一种与您的整体安全响应计划保持一致的方法。

实践#8 - 使用已批准的工具

- 定义并发布已批准的工具及其关联的安全检查的列表，例如编译器/链接器选项和警告。工程师应努力使用最新版本的已批准工具（如编译器版本），并利用新的安全分析功能和保护；
- 开发团队只需要关注自己的开发任务，而开发工具、编译器等的选择让专业的工具团队来完成，工具团队通过自己的研究会有更专业的指导和选择，然后提供给开发团队；
- iPhone XCode Ghost**苹果手机病毒事件**的教训；

SDL推荐的工具

代码安全性用

微软威胁建模
全问题的设计

Roslyn分析器
分析)，但也
Roslyn 分析器

微软DevSkim
中提供内联安

微软安全风险
安全风险检测
技术。

攻击表面分析
态、运行时参

cusexsvcex 属性页

配置(C): Release 平台(P): x64 配置管理器(O)...

配置属性

常规

高级

调试

VC++ 目录

C/C++

常规

优化

预处理器

代码生成

语言

预编译头

输出文件

浏览信息

高级

所有选项

命令行

链接器

常规

输入

清单文件

调试

系统

优化

嵌入的 IDL

查找选项或开关:

ASM 列表位置

C++ 语言标准

OpenMP 支持

SDL 检查

Spectre 缓解

XML 文档文件名

安全检查

保留注释

编译为

程序数据库文件名

创建可热修补映像

对象文件名

多处理器编译

浮点模型

符合模式

附加包含目录

附加选项

公共语言运行时支持

\$(IntDir)

默认值

是 (/sdl)

已启用 (/Qspectre)

\$(IntDir)

启用安全检查 (/GS)

否

默认值

\$(IntDir)vc\$(PlatformToolsetVersion).pdb

\$(IntDir)

是 (/MP)

精度 (/fp:precise)

否

Spectre 缓解

CVE 2017-5753 的 Spectre 缓解。 (/ Qspectre)

确定








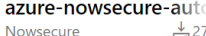










取消

应用(A)

实践#9 - 执行静态分析安全测试 (SAST)

- 在编译之前分析源代码提供了一种高度可扩展的安全代码审查方法，有助于确保遵循安全编码策略。SAST 通常集成到提交管道中，以便每次构建或打包软件时识别漏洞。但是，某些产品集成到开发人员环境中，以发现某些缺陷，例如存在不安全或其他被禁止的功能，并在开发人员正在积极编码时用更安全的替代方法替换这些缺陷。没有一种尺寸适合所有解决方案，开发团队应决定执行 SAST 的最佳频率，并可能部署多种策略，以平衡生产率和足够的安全覆盖范围。

VSTS安全工具市场

 Snyk Security Scan Snyk 306 Snyk scan for open source vulnerabilities ★★★★★ FREE	 Container Security Aqua Security 763 Vulnerability scanner for container images ★★★★★ FREE	 ADO Security Scanner Microsoft DevLabs 324 ADO Security Scanner provide security guidelines for devops components like Org, Projec... ★★★★★ FREE	 Serverless Security Aqua Security 90 Vulnerability scanner for serverless ★★★★★ FREE	 Codified Security Codified Security 90 This step uploads your app to Codified Security for automated mobile app... ★★★★★ FREE	 REST API Static Security 42Crunch 6 Locate OpenAPI/Swagger files (JSON and YAML, v2 and v3) and run 42Crunch Security... ★★★★★ FREE
 Beagle Security Test Beagle Security 12 Test for security vulnerabilities using Beagle Security ★★★★★ FREE	 azure-nowsecure-auto Nowsecure 27 Azure Extension for NowSecure Auto Security Test ★★★★★ FREE	 Application Security Test HCL Technologies 283 Perform static, dynamic, mobile and open source security tests for your... ★★★★★ FREE	 SonarQube SonarSource 40.7K Detect bugs, vulnerabilities and code smells across project branches and pull... ★★★★★ FREE	 API Management Suite Stephane Eyskens 1.9K Broad support of Azure API Management ★★★★★ FREE	 Secure DevOps Kit (Azure) Microsoft DevLabs 3.2K Collection of extensions that empower DevOps teams to build and deploy application... ★★★★★ FREE
 Alcide Kubernetes Adonis Alcide 45 A Kubernetes Security & Hygiene Scanner for CI+CD pipelines ★★★★★ FREE	 Kiuwan Kiuwan Software 254 Analyze your applications with Kiuwan in your build definitions. Find relevant... ★★★★★ FREE	 Appknox Appknox 11 Appknox Automated Security Scanning ★★★★★ FREE	 OWASP Zed Attack Proxy Kasun Kodagoda 1.3K Visual Studio Team Services build/release task for running OWASP ZAP automated... ★★★★★ FREE	 SD Elements Integration Security Compass 30 Security Compass SD Elements Platform Integration ★★★★★ FREE	 NeuVector NeuVector Inc. 2 Scan container images for vulnerabilities with the NeuVector Container Security... ★★★★★ FREE

实践#10 - 执行动态分析安全测试 (DAST)

- 对完全编译或打包的软件执行运行时验证检查功能，这些功能仅在集成和运行所有组件时才明显。这通常是使用一套预先构建的攻击工具或工具实现的，这些工具专门监视应用程序行为以查找内存损坏、用户特权问题和其他关键安全问题。与 SAST 类似，没有一刀切的解决方案，虽然某些工具（如 Web 应用程序扫描工具）可以更容易地集成到连续集成/连续交付管道中，但其他 DAST 测试（如模糊化）需要不同的方法。

实践#11 - 执行渗透测试

- 渗透测试是一种软件系统的安全分析，由模拟黑客操作的熟练安全专业人员执行。渗透测试的目的是发现编码错误、系统配置错误或其他操作部署弱点导致的潜在漏洞，因此测试通常发现最广泛的漏洞。渗透测试通常与自动和手动代码审查结合执行，以提供比通常可能更高的分析级别。

使用白盒模糊功能进行自动渗透测试

- 安全研究人员和黑客越来越多地使用模糊作为查找漏洞的主要技术之一。黑客通常练习黑盒模糊 - 生成数据的各种排列，而实际上不会将其与分析数据的代码相关联。
- 模糊是一种系统的方法来查找代码中的错误，这些缺陷是安全错误最严重的原因。模糊化能够查找*远程代码执行*（缓冲区溢出）、*永久拒绝服务*（未处理的异常、读取 AV、线程挂起）和*临时拒绝服务*（泄漏、内存峰值）。
- 模糊不仅发现缓冲区边界验证的缺陷，还发现状态机逻辑、错误处理和清理代码中的故障。

攻击表面分析器

- 攻击表面分析器的核心功能是在安装软件组件之前和之后"分散"操作系统的安全配置的能力。这一点很重要，因为大多数安装过程都需要提升权限，一旦授予，可能会导致意外的系统配置更改。
- 攻击 Surface 分析器当前报告对以下操作系统组件的更改：
 - 文件系统（提供静态快照和实时监视）
 - 用户帐户
 - 服务
 - 网络端口
 - 证书
 - 注册表
 - COM 对象（新！）
 - 事件日志（新建！）
 - 防火墙设置（新建！）

实践#12 - 建立标准事件响应流程

- 制定事件响应计划对于帮助解决随时间而出现的新威胁至关重要。它应与组织的专用产品安全事件响应团队（PSIRT）协调创建。该计划应包括在发生安全紧急情况时与谁联系，并建立安全服务协议，包括从组织内其他组继承的代码和第三方代码的计划。在需要之前，应测试事件响应计划！

SDL和DevOps的融合

安全 DevOps

- 使安全原则和实践成为 DevOps 不可或缺的一部分，同时保持更高的效率和生产率
- 从一开始，Microsoft SDL 就确定安全性需要成为每个人的工作，并在 SDL 中包括项目经理、开发人员和测试人员的做法，所有这些都旨在提高安全性。此外，它认识到一个大小并不适合所有开发方法，因此描述了经过验证的灵活实践和活动，这些实践和活动使用经典瀑布或较新的敏捷来提高软件应用程序在开发每个阶段的安全性方法。但是，除了考虑生产环境外，SDL 还不包括操作工程师的活动。
- DevOps 方法改变了这一点。现在，开发和操作已紧密集成，以便快速、持续地向最终用户交付价值。DevOps 已取代孤立的开发和运营，以创建多学科团队，与共享和高效的实践、工具和 KPI 协同工作。要在这种快速移动的环境中提供高度安全的软件和服务，安全以相同的速度移动至关重要。实现此目的的一个方法是在开发（SDL）和操作（OSA）流程中构建安全性。

实践#1— 提供培训

- 培训对成功至关重要。确保每个人都了解攻击者的观点、目标，以及他们如何利用编码和配置错误或体系结构弱点，将有助于吸引每个人的注意力，并提高集体知识标准。

实践#2_定义需求

- 建立考虑到安全性和合规性控制的最低安全基准。确保这些已烘焙到 DevOps 流程和管道中。至少，确保基线考虑到实际威胁，如 **OWASP 前 10 名**或**SANS 前 25 名**以及已知存在或可能由您选择的技术堆栈中的人为错误引起的行业或法规要求和问题。

实践#3— 定义指标和合规性报告

- 通过定义推动行动和支持合规性目标的特定指标，与工程师一起推动您想要的行为。

实践#4 — 使用软件组合分析 (SCA) 和治理

- 选择第三方组件（商业和开源）时，了解组件中的漏洞可能对系统整体安全产生的影响非常重要。SCA 工具可帮助进行许可暴露，提供组件的准确清单，以及使用引用的组件报告任何漏洞。在使用高风险的第三方组件时，您也应该更加有选择性，并考虑在使用它们之前执行更彻底的评估。

实践#5_执行威胁建模

- 虽然**威胁建模**DevOps 中可能具有挑战性，因为它感知到速度缓慢，是任何安全开发过程的关键组成部分。在大多数情况下，将结构化方法应用于威胁方案有助于团队更有效、更便宜地识别安全漏洞，确定这些威胁的风险，然后进行安全功能选择并建立适当的缓解措施。至少，在存在有意义的安全风险的环境中，应使用威胁建模。

实践#6—使用工具和自动化







使用经过精心挑选的工具和智能自动化，这些工具和智能自动化已集成到工程师的世界（如集成开发环境）。在现代工程领域，很容易假设自动化是解决方案，自动化是关键，但在选择工具时必须要有选择性，并在部署工具时要小心。目标是解决问题，而不是使工程师在日常工程经验之外使用太多的工具或外星流程来超载。用作安全 DevOps workflow 一部分的工具应遵循以下原则：

- 工具必须集成到 CI/CD 管道中。
- 工具不需要安全专业知识。
- 工具必须避免报告问题出现高误报率。







将静态应用程序安全测试（SAST）集成到 IDE（集成开发环境）中，可以深入了解语法、语义，并提供及时学习，防止在应用程序代码提交到代码存储库之前引入安全漏洞。同样，将动态分析安全测试（DAST）工具集成到持续集成/连续交付管道中，将有助于快速发现只有在集成和运行所有组件时显而易见的问题。

Azure DevOps 的扩展







Featured

 Timetracker 7pace ★★★★★ 12.6K FREE TRIAL	 Azure Cost Insights Kees Schollaart ★★★★★ 2.9K FREE	 Code Quality NDepend ndepend ★★★★★ 1.4K FREE TRIAL	 Bravo Notes Agile Extensions ★★★★★ 555 FREE TRIAL	 Visual Studio IntelliCode Microsoft ★★★★★ 521 FREE	 Azure Boards Slack app Microsoft ★★★★★ 61 FREE
--	---	--	---	--	--

Most Popular [See more](#)

 Code Search Microsoft ★★★★★ 134K FREE	 Test & Feedback Microsoft ★★★★★ 131K FREE	 Azure DevOps Open icon Microsoft DevLabs ★★★★★ 93.6K FREE	 SonarQube SonarSource ★★★★★ 41.8K FREE	 Delivery Plans Microsoft ★★★★★ 34.9K FREE	 Replace Tokens Guillaume Rouchon ★★★★★ 34.4K FREE
---	---	---	--	---	---

Trending [this week](#)

 FlowViz Agile Extensions ★★★★★ 209 FREE	 Time Logging Extension TimeLog ★★★★★ 612 FREE	 Install .NET Core Runtime Ronald Bosma ★★★★★ 312 FREE	 UseGitVersion GitTools ★★★★★ 682 FREE	 Unified Functional Testing Micro Focus ★★★★★ 332 FREE	 Angular CLI Raul A. Ruiz ★★★★★ 457 FREE
---	---	---	---	---	---

实践#7— 保持凭据安全

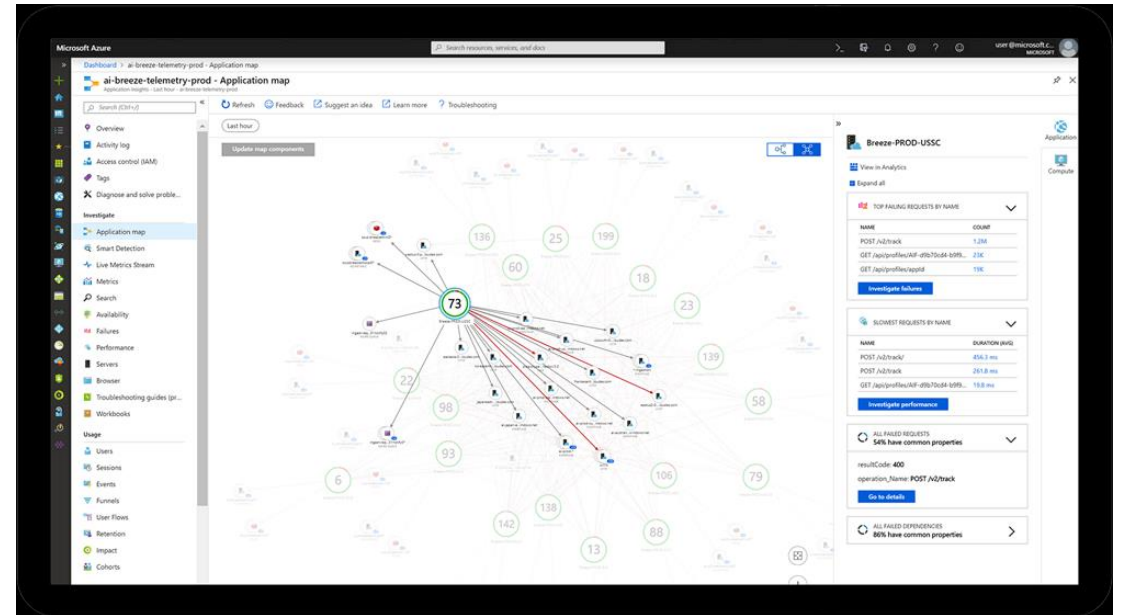
- 在预提交期间，需要扫描源文件中的凭据和其他敏感内容，因为它们降低了将敏感信息传播到团队的 CI/CD 流程的风险。请考虑使用自带密钥（BYOK）解决方案使用硬件安全模块（HSM）生成密钥，而不是将敏感密钥存储在代码中。

实践#8—使用持续学习与监控

- 通过高级分析监视应用程序、基础设施和网络有助于发现安全性和性能问题。当利用与监视工具相结合的持续集成/持续部署（CI/CD）实践时，您将能够更好地了解应用程序运行状况，并主动识别和降低风险，以减少遭受攻击的风险。监视也是支持纵深防御战略的重要组成部分，可以减少组织识别（MTTI）的平均时间，并减少包含（MTTC）指标的平均时间。

持续监控发布的应用

- 监控您的应用程序
- 获得监视Web应用程序的可用性，性能和使用情况所需的一切，无论它们是托管在Azure还是本地。 Azure Monitor支持流行的语言和框架，例如.NET，Java和Node.js，并与DevOps流程和工具（例如Azure DevOps，Jira和PagerDuty）集成。 跟踪实时指标流，请求和响应时间以及事件。



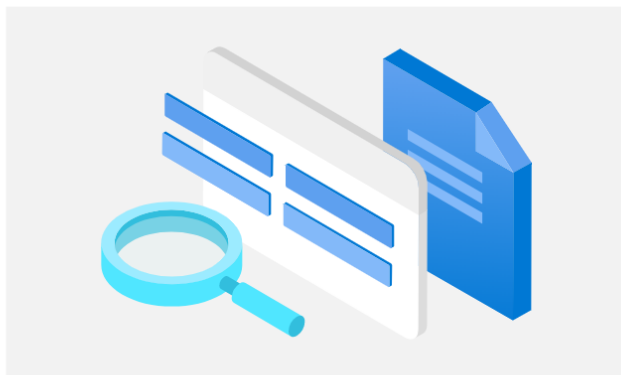
监视应用程序和基础结构



监视和可视化指标

指标是来自 Azure 资源的数字值，可帮助你了解系统的运行状况、操作和性能。

[浏览指标](#)



查询和分析日志

日志是来自监视解决方案的活动日志、诊断日志和遥测；分析查询帮助进行故障排除和可视化。

[搜索日志](#)

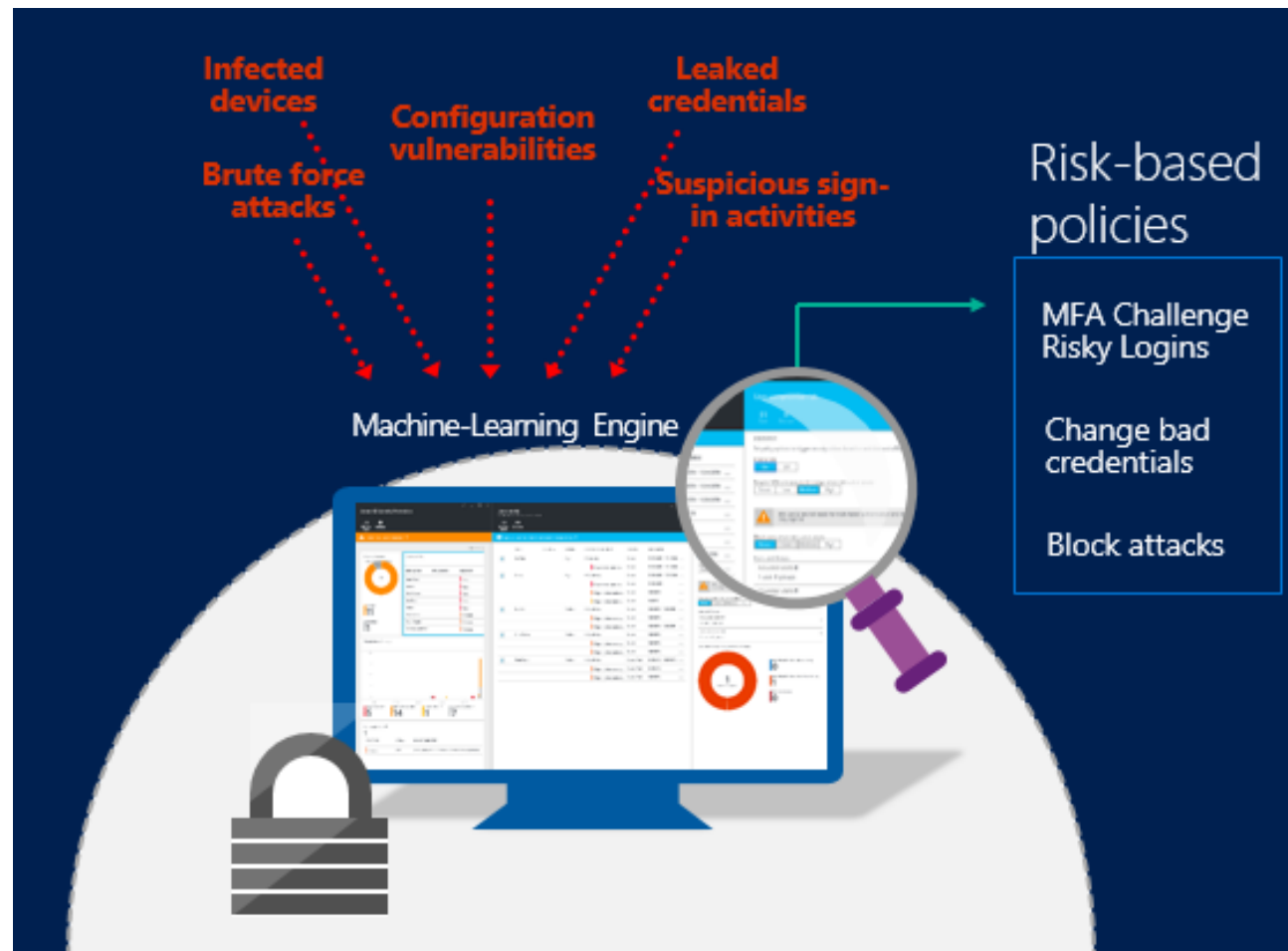


设置警报和操作

警报通知你严重的情况，并且可能根据指标或日志采取自动纠正措施。

[创建警报](#)

Azure 高级威胁检测





扫码下载讲师PPT
更多精彩尽在【微软市场活动】