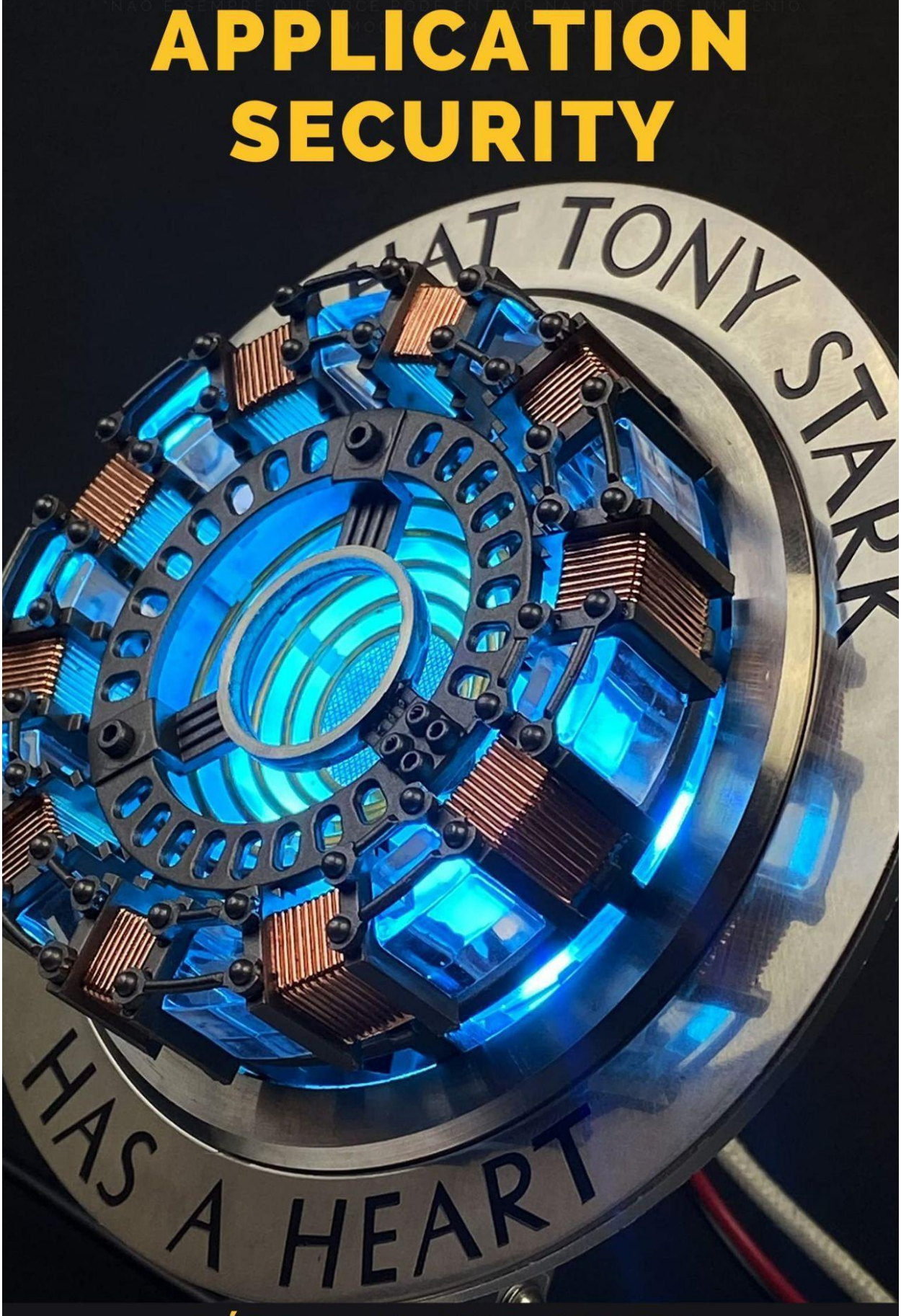
APPLICATION SECURITY



CÁSSIO BATISTA PEREIRA

LESSONS LEARNED FROM IRON MAN ABOUT SECURE SOFTWARE DEVELOPMENT

AppSec - Homem de Ferro

Segurança de Aplicações pela mente do Homem de Ferro

Cássio Batista Pereira

AppSec - Iron Man

Segurança de Aplicações pela mente do Homem de Ferro

Cássio Batista Pereira

Publicação independente

Contato:

Cássio Batista Pereira
cassio@cassiobp.com.br

SUMARIO

Prefácio

Cássio B. Pereira
Especialista em Segurança de Aplicações

Autor

Muito prazer, me chamo Cássio Batista Pereira, ou @cassiodeveloper, sou desenvolvedor e arquiteto de software por formação. Trabalho como Especialista em Segurança de Aplicações e ajudo empresas e profissionais a construírem software seguro. Possuo mais de duas décadas de experiência no mercado de desenvolvimento e segurança nos mais variados segmentos de negócios, onde adquiri conhecimento para trabalhar com diferentes tecnologias, linguagens, estruturas e processos. Sou evangelista da cultura do Desenvolvimento Seguro e compartilho conhecimento sobre AppSec, DevSecOps e SDLC, veja mais detalhes aqui.

Você também pode encontrar uma versão reduzida desse livro em formato de apresentação aqui, aproveite.

Aviso!

Este livro abordará alguns filmes do UCM - Universo Central da Marvel, e por isso muitos spoilers aparecerão, todos os filmes são bem antigos, então, a essa altura, você já deve ter assistido a todos eles. Mas, caso você não seja fã de filmes de super-heróis e não tenha assistido até agora, esteja avisado que haverá muitos spoilers. Então, eu recomendo que você assista a todos os filmes antes de prosseguir, ou melhor ainda, leia este livro agora mesmo, e aproveite a experiência do filme depois (já que você saberá detalhes que a maioria das pessoas não tem ideia).

Capítulo 0

Breve introdução

"A segurança é responsabilidade de todos"
– Cássio B. Pereira

Tony Stark, também conhecido como Homem de Ferro, é um personagem fictício do universo da Marvel Comics. Ele foi criado pelo escritor e editor Stan Lee, desenvolvido pelo roteirista Larry Lieber, e desenhado pelos artistas Don Heck e Jack Kirby. Tony Stark fez sua primeira aparição em "Tales of Suspense" #39 em 1963.

No Universo Cinematográfico Marvel (UCM), Tony Stark é interpretado pelo ator Robert Downey Jr. Ele é uma figura central no UCM e desempenhou um papel crucial na formação dos Vingadores.

Histórico:

Tony Stark é um brilhante industrialista e engenheiro que herda a Stark Industries, uma fabricante multinacional de armas, de seu pai, Howard Stark. Ele é conhecido por seu intelecto e inventividade excepcionais. A vida de Stark toma um rumo dramático quando ele é capturado por terroristas e forçado a construir uma arma de destruição em massa. Em vez disso, ele constrói uma armadura motorizada para escapar, marcando o nascimento do Homem de Ferro.

Armadura do Homem de Ferro:

A armadura do Homem de Ferro é um exoesqueleto motorizado tecnologicamente avançado que aumenta a força, durabilidade e habilidades de Stark. Com o tempo, Stark continuou a atualizar e refinar o traje, criando inúmeras variações com diferentes capacidades.

Desenvolvimento do personagem:

Tony Stark passa por um desenvolvimento de personagem significativo ao longo dos quadrinhos e do UCM. Inicialmente retratado como um playboy e um gênio egocêntrico, ele enfrenta dilemas morais que o levam a reavaliar suas prioridades. Stark evolui para um indivíduo mais responsável e altruísta, usando sua tecnologia para lutar contra ameaças à humanidade.

Papel no UCM:

No UCM, Tony Stark desempenha um papel fundamental na trilogia "Homem de Ferro", "Os Vingadores" e vários outros filmes. Ele é um membro fundador dos Vingadores e é fundamental na batalha contra vários adversários formidáveis, incluindo a invasão Chitauri, Ultron e Thanos

Legado:

O arco do personagem de Tony Stark chega a uma conclusão pungente em "Vingadores: Ultimato" (2019), onde ele faz o sacrifício final para salvar o universo. Seu

legado continua vivo, e o impacto de suas contribuições para a tecnologia e heroísmo continua a moldar o UCM.

Influência:

O personagem de Tony Stark se tornou icônico, representando a intersecção de inteligência, carisma e heroísmo. Sua jornada de um gênio egocêntrico a um herói altruísta ressoou com o público, tornando o Homem de Ferro um dos personagens mais amados e duradouros do universo Marvel.

Visão do Autor:

Agora que você sabe (se não sabia antes) um pouco mais sobre o Homem de Ferro e Tony Stark (a mesma pessoa, mas ao ler este livro, você pode ver referências aos dois), deixe-me trazer minha opinião/visão pessoal do personagem.

Antes de tudo, eu sei que você está ansioso sobre a parte de AppSec e tecnologia, mas é crucial entender a mente dele para entender suas conquistas e feitos notáveis, então não pule esta parte ou mais tarde algumas coisas não farão sentido algum. Então, Tony é um humano, certo? Apenas um homem, sim, com muito dinheiro e eu quero dizer, MUITO DINHEIRO, um conhecimento enorme e personalidade forte, mas ele é apenas um homem, com problemas também, especialmente em relacionamentos, considerando que seus pais estavam ausentes na maior parte do tempo, ele geralmente se sentia solitário, é por isso que (talvez) parte de seu hobby, era aproveitar festas e vida noturna gastando sua fortuna com garotas legais e carros incríveis. Mas ok, isso é apenas parte de sua vida e caráter, um fato importante para nós aqui é que, ele cresce como ser humano, ele aprende com seus próprios erros, os reconhece e cresce, e por isso ele quer que todos ao seu redor também cresçam e sejam melhores, como seres humanos. E vemos tudo isso durante seu desenvolvimento nos filmes. Não sou muito fã de quadrinhos, devo ter lido um ou dois quando era adolescente, mas é só isso, meu foco está nos filmes do UCM, então, por favor, leve em consideração que posso perder algumas informações sobre ele.

Como eu disse, como homem, Tony conhece suas limitações e pontos fortes, como, eu diria, todo homem deveria conhecer, pois isso permite e "facilita" nossa jornada chamada vida, e sendo um gênio como Tony é, pode parecer arrogante às vezes (o tempo todo), ou até mesmo descuidado em algumas situações, mas não se engane, ele não é.

Neste livro, eu o levarei em uma jornada pelo UCM (com foco no Homem de Ferro), considerando a trilogia de filmes Homem de Ferro 1, 2 e 3 e, é claro, suas aparições em outros filmes como Os Vingadores, Vingadores - Era de Ultron, Vingadores - Guerra Infinita, Vingadores - Endgame, Capitão América - Guerra Civil, Homem-Aranha - De Volta ao Lar e talvez outros (nunca lembrarei de todos), e, durante essa jornada, aprenderemos como a mente de Tony funciona e nos beneficiarmos dela para

implementar a segurança de aplicações em nossas vidas reais, e não importa se você é um desenvolvedor, um engenheiro de QA, um DevOps ou um engenheiro de segurança cibernética, se estiver envolvido no SDL de alguma forma, você deve conhecer AppSec. Veremos muitas referências a filmes e aprenderemos com essas referências como Tony pensa e como nós também devemos pensar para garantir que nosso software seja, de certa forma, mais seguro.

Recomendo enfaticamente que você assista a TODOS os filmes, seja durante esta leitura, antes da leitura ou depois de ler este livro. As coisas farão mais sentido, você terá as mesmas emoções e sentimentos que eu tive ao escrever este livro, e você se divertirá e passará um bom tempo assistindo, talvez você possa fazer uma "maratona de filmes" em um fim de semana frio sozinho, ou até mesmo com seu/sua parceiro(a), um bom momento para estarem juntos e descobrirem/aprenderem algo novo. Ah, mas eu não gosto de super-heróis, Marvel etc... NÃO ME IMPORTO. ASSISTA A TODOS ELES. Talvez você comece a gostar, talvez veja que a fantasia e a ficção são boas para nossa imaginação, veja esse livro!

Capítulo 1

Modelagem de Ameaças

"A arte de encontrar problemas antes mesmo do software existir."
– Cássio B. Pereira

Contextualizando

No primeiro filme Homem de Ferro 1, Tony é sequestrado por terroristas e mantido como refém em uma caverna, o que é o começo de tudo para que o Homem de Ferro se torne quem ele é. E não vou escrever todo o roteiro do filme aqui, mas é importante estabelecer algumas bases. Durante o ataque dos terroristas ao comboio em que Tony se encontra, alguns soldados morrem e Tony se vê ao lado de um míssil/bomba ativo onde está escrito "Indústrias Stark", uma ironia que seu próprio produto, agora, está ameaçando sua vida, e seu rosto não demonstra surpresa do tipo "ah, merda, uma bomba aqui...", mas sim "ah, merda, MINHA BOMBA aqui?", porque ele sabia quem eram seus clientes, e definitivamente aqueles terroristas não estavam em sua lista, então como diabos seu míssil estava ali? E esse momento é crucial, porque ele percebe que o legado de seu pai e agora seu próprio trabalho não está sendo usado para proteger os Estados Unidos, mas está nas mãos de bandidos que estão matando pessoas inocentes, e ele não quer isso.

Na caverna, após alguns dias, Tony percebe, com a ajuda de outro refém, que precisa fugir, pois se ele criar o que os terroristas querem, a América e o mundo estarão ainda mais em perigo, considerando que eles queriam a tecnologia do míssil Jerico, uma nova e poderosa arma projetada por Tony. Ele, então, finge estar construindo o míssil Jerico, mas está construindo algo diferente, algo que seria seu maior legado mais tarde e, para aquele momento, sua salvação, o Mark I, o primeiro traje, o traje do Homem de Ferro, e literalmente de Ferro, pois eram os materiais que ele tinha na caverna.

Modelagem de Ameaças

Não fica claro no filme, é claro, mas Tony faz um tipo de Modelagem de Ameaça para criar o primeiro traje, caso contrário, ele não o criaria se não tivesse certeza de que seria bem-sucedido, ou pelo menos que teria uma alta porcentagem de sucesso. Porque, lembremos, ele é um homem, qualquer soco forte, bala, faca podem matá-lo. Então ele fez o quê?

Ameaças

A principal ameaça eram os homens armados, considerando alguns tipos de armas e metralhadoras.

Objetivos

Os principais objetivos incluíam:

- Escapar da caverna - Sair;
- Ir o mais longe possível - Não ficar na entrada;
- Não sofrer nenhum dano - Ele não queria morrer;
- Atacar - Para evitar algumas ameaças;
- Esperar ser encontrado pelos EUA;

Requisitos

Os principais requisitos de segurança incluíam:

- Escapar da caverna - Possibilidade de quebrar portas e evitar inimigos;
- Ir o mais longe possível - Voar;
- Não sofrer danos - À prova de balas;
- Atacar - lança-chamas, socos e chutes;
- Esperança de ser encontrado pelos EUA - Estando fora da caverna, seria fácil para o Exército dos EUA encontrá-lo;

Lições aprendidas

Podemos tirar algumas lições disso:

- Com o material que tinha, ele fez o melhor que pôde;
    - Mark I não derrotaria Thanos nem seria suficiente para lutar contra o Capitão América e o Soldado Invernal mais tarde, mas, para AQUELE MOMENTO, foi suficiente.
- Feito é melhor que perfeito!

Reflexão

É importante que aprendamos com esse primeiro momento do livro fazendo algumas perguntas a nós mesmos:

- Quantas vezes você já pensou sobre as ameaças que seu software pode enfrentar antes de colocá-lo em produção?
- Já pensou que sua empresa, marca, software etc. podem ser alvo de criminosos cibernéticos?
- Já pensou no impacto na sociedade caso seu software seja afetado por um ataque cibernético?
- Você já imaginou que é possível fazer um brainstorming de algumas ameaças apenas com a lista de requisitos funcionais?

Sejamos sinceros, se Tony Stark não tivesse 99% de certeza de que o primeiro traje o libertaria dos terroristas, você acha que ele o faria mesmo assim? Ou ele fez isso porque tinha certeza de que seria bem-sucedido de alguma forma, considerando as ameaças e a proteção que tinha, ele estava basicamente pensando, as ameaças são de nível 3 e eu tenho nível de defesa 5, então vamos lá.

Em nosso ciclo de vida de desenvolvimento de software, essa é a mesma mentalidade que devemos ter:

- Quais são as ameaças que a funcionalidade X pode enfrentar?
- Quais são as mitigações que devo implementar para reduzir o impacto ou até mesmo eliminar a ameaça?
- E fazer tudo isso antes de o software entrar em produção. Caso contrário, correremos riscos.

Conhecimento profundo

Modelagem de Ameaças - COMO

Quero destacar algumas dicas para você com relação à modelagem de ameaças:

- A modelagem de ameaças deve ser usada em ambientes em que haja risco significativo à segurança e não em todo o sistema.
- A modelagem de ameaças pode ser aplicada no nível do componente, da aplicação ou do sistema.
    - Recomendo o mais granular possível (funcionalidade), para não ter um modelo enorme que será difícil de ler e entender.

Caso de Uso - Veronica ou Hulk Buster ou Mark XLIV

Em Vingadores - Era de Ultron, Hulk foi enfeitiçado por Wanda Maximoff, a Feiticeira Escarlate, e ficou fora de controle como nunca. A essa altura, Bruce Banner já era (quase) capaz de controlar Hulk, como vimos em Os Vingadores e nas cenas pós-créditos de O Incrível Hulk. Mas, juntos, Bruce e Tony projetaram a Veronica para o caso de Hulk enlouquecer novamente, agora seria possível, pelo menos, lutar com ele para controlar os danos e perdas, o que antes não era possível.

Então, a Verônica foi um trabalho de Modelagem de Ameaças 100%, foi projetada exclusivamente para combater/conter o Hulk no caso de uma mudança não intencional de Bruce Banner, e quase não funcionou. Durante a luta, podemos ver que Veronica tem muitas funcionalidades para conter Hulk, não para destruí-lo/matá-lo, mas para parar seus ataques, tentar fazê-lo dormir com algum tipo de veneno de fumaça verde, bloquear seus socos com um armário de mão... e Tony quase perdeu. Sem mencionar que, no início, uma espécie de gaiola prende Hulk, sem sucesso. O filme não mostra isso, mas Bruce e Tony trabalharam juntos, discutiram os ataques de Hulk, os pontos fortes e fracos, as possibilidades e as consequências para criar o traje HulkBuster perfeito, não apenas para deter Hulk, mas também para proteger a vida de Tony. Observe que, na imagem acima, assim que eles dão um soco um no outro ao mesmo tempo, o rosto de Tony fica "surpreso", pois ele conhece o poder de seu traje, tem todos os cálculos de quão forte ele é e, ainda assim, o soco de Hulk estava no mesmo nível, de fato a ameaça é mais assustadora quando acontece do que quando apenas pensamos nela.

Ameaça

A ameaça principal era o **Hulk**.

Objetivos

Os principais objetivos incluíam:

- Parar o Hulk - Fazer com que ele pare de atacar/danificar a inocência, a cidade etc.;
- Ir o mais longe possível - Não ficar na cidade, mas em um local não povoado;
- Sofrer danos leves - Ele sabia que levaria alguns socos;
- Atacar - Para conseguir o primeiro objetivo também;

Requisitos

Os principais requisitos de segurança incluíam:

- Para deter Hulk - Uma Gaiola/Jaula, forte e ágil o suficiente.
- Ir o mais longe possível - Voar, trava de braço;
- Sofrer danos leves - Somente o ferro não seria suficiente;
- Atacar - Voar, trava de braço, laser e veneno;

Lições aprendidas

Podemos tirar algumas lições disso:

- A ameaça real é diferente da ameaça teórica, geralmente é pior;
- Mesmo com todas as mitigações em vigor, os danos ainda podem ser prejudiciais;
- É necessário tempo e pesquisa;
- A comunicação é muito importante;
- Esteja pronto para improvisar a qualquer momento;

Resposta a incidentes - Aprendendo com os fatos e os erros

Outra maneira muito eficaz de fazer a Modelagem de Ameaças é usar os incidentes que aconteceram no passado, o histórico do que e como aconteceu ajuda as equipes a criarem softwares à prova de balas, pelo menos à prova de ameaças conhecidas que já eram realidade para o seu contexto. Como no caso abaixo:

Em Homem de Ferro 3, Tony voou para o Tennessee depois de ser atacado pelo grupo terrorista liderado pelo Mandarim, seu traje era um protótipo, por isso o voo aconteceu, antes ele estava apenas planejando o voo com Jarvis. Quase chegando ao Tennessee e seu traje estava ficando sem energia, Jarvis o alertou e ele se sentiu quase com 0% de energia. É claro que ele precisava sair do traje para ir a qualquer lugar, agora a pé. Mas era inverno, ele estava congelando na neve. É bom lembrar que, como homem, Tony

agora estava com roupas leves em um inverno rigoroso, o que é perigoso para qualquer ser humano nesse tipo de condição climática. Além disso, ele estava bastante machucado após a "luta" com o grupo terrorista que o atacou, estava faminto, sujo e sedento e longe dos recursos básicos.

E então, quando ele criou o traje do Homem-Aranha mais tarde no filme Homem-Aranha: De Volta ao Lar, bem.... Ele era equipado com um aquecedor! Quando Peter caiu em um lago e foi resgatado pelo Homem de Ferro, Tony disse:

Já está bem claro que a mente do Tony é incrível e que ele está sempre melhorando, ele teve um problema e consertou, isso é bem básico, não? Bem, não... nossos softwares apresentam diariamente bugs e vulnerabilidades que nem sequer nos damos ao trabalho de verificar, apenas os colocamos em algum tipo de lista com um nome bonito, como Backlog, e um dia, se não houver nada mais importante, como um problema inesperado, trataremos disso. Tony nos mostra que podemos cometer erros, mas não podemos mantê-los, precisamos fazer melhor da próxima vez. Nesse caso, não apenas para si mesmo, mas também para os outros, pois nos preocupamos com os outros, certo? Nosso software geralmente tem a intenção de agregar valor ao nosso cliente, que tem a intenção de agregar valor ao cliente final, mas, no caso de um problema, todos perdem.

Vamos também deixar algo claro aqui: ele tinha um problema menor, considerando todo o universo de ameaças do Homem de Ferro. Ser exposto à neve não é um "grande erro", mas algo que, considerando o ambiente e as circunstâncias, poderia matá-lo. Então, por que não o corrigir?

Resumindo o que foi feito até o momento

Vamos fazer uma pausa no fluxo do tópico e resumir um pouco do que vimos até agora:

- A versão 1.0.1 do seu software deve ser melhor que a 1.0.0, melhor em termos de segurança, e não apenas novos recursos interessantes que "agregam valor ao seu cliente". FODA-SE. Se o seu novo recurso for uma porta para um ataque cibernético, você não está agregando valor, está RETIRANDO VALOR.
- Pense como Tony Stark! Ok, você pode não ser um gênio, mas tente!
- Além de corrigir bugs, você deve melhorar a segurança. As vulnerabilidades e as falhas de lógica comercial devem ser corrigidas.
- Não há segurança sem o processo adequado de SDL e o comprometimento da equipe.
- Imagine que cada versão sua é um novo traje Mark que Tony construiu.
- Cada traje foi projetado para proteger a vida dele. Seu software não pode apenas agregar valor se esse valor não proteger também o cliente/usuário. Portanto, cada versão de seu software também deve proteger seu cliente/negócio.

De volta ao Homem de Ferro 1

Logo depois de ser resgatado pelo Coronel Rodes após escapar dos terroristas, Tony chegou em casa e começou a trabalhar para melhorar sua grande criação, o traje do Homem de Ferro. Ele criou o primeiro traje totalmente operacional, depois de algumas sessões de teste que falharam (é para isso que servem os testes, certo?), ele agora era capaz de ter um voo estável, tinha o monitoramento em tempo real e a assistência da IA Jarvis, e não apenas o ferro como material para o traje, mas um material melhor e mais durável e, é claro, uma versão aprimorada do principal componente do traje, a fonte de energia, o Arch React. Sendo capaz de voar, ele fez seu primeiro voo atingindo diretamente à estratosfera e tendo um problema de formação de gelo.

No final do filme, com o Mark III, ele corrigiu o problema e agora podia voar até a estratosfera sem gelo, mas o vilão não conseguiu porque estava usando uma versão baseada no Mark II, com o problema do gelo. Fork errado, cara, Fork errado!

Homem De Ferro: Problema de formação de gelo.

Quando uma vulnerabilidade vira um recurso

1. Em Homem de Ferro 2, na luta contra o Whiplash, Tony sofreu muitos danos com a energia dos chicotes do Whiplash. Naquela época, já podíamos ver uma grande

melhoria no traje do Homem de Ferro, agora Tony não precisava estar em um lugar para vestir (montar) o traje, agora o traje era portátil em sua maleta, muito mais eficaz, certo?

2. Mais tarde, em Os Vingadores, durante a luta (mal-entendida) contra Thor, ele agora era capaz de absorver a energia do trovão e sofrer menos danos. Está vendo? Agora seu traje podia sofrer menos danos devido a um material melhor e também absorver a energia dos ataques, ele aprendeu que, caso algum inimigo tivesse "uma arma de energia", ele poderia se beneficiar disso, não é brilhante? Os aprimoramentos no traje em relação ao material são constantes, a cada nova versão eles são melhores, de alguma forma, e isso, por definição, é um processo de aprimoramento contínuo que ele tem.

3. Depois, em Vingadores: Ultimato, seu traje agora está equipado com tantos recursos que é quase impossível descrever, com todos os recursos de

nanotecnologia, as limitações eram apenas sua criatividade, eu diria. Mas, durante o confronto com Thanos, junto com Thor e Capitão América, ele agora foi capaz de abrir suas costas como uma "flor" para poder absorver toda a energia do trovão que Thor invocou e se tornar uma poderosa arma de ataque disparando uma forte combinação de trovão e seu laser Arc React contra Thanos. Sim, tudo bem que Thanos conseguiu lidar com isso facilmente, mas isso é genial.

Essa sequência nos mostra que ele não está preocupado apenas em se proteger com o melhor material de traje, ele também está sempre melhorando a capacidade de ataque, e o que antes era um bug, material fraco, muitos danos e nenhum benefício, agora era um traje com material mais forte e absorvedor de energia capaz de defender e atacar ao mesmo tempo. Sem mencionar alguns detalhes relacionados à absorção de energia para os quais toda a fonte de energia do traje precisava estar preparada, certo? Quanta energia um trovão pode produzir? Quanta energia ele era capaz de armazenar antes? Todos esses cálculos e detalhes não estão lá, mas implicitamente nos mostram que foi feito um trabalho duro pra caramba. Sem mencionar a portabilidade do traje em si, de uma instalação em um local na garagem, para uma mala e agora uma espécie de "distintivo de peito" simples que pode carregar tudo isso com o milagre da nanotecnologia.

Agora, com isso em mente, imagine os bugs e as vulnerabilidades no software em que você está trabalhando, que funciona ou funcionará, não importa, eles estão em uma pilha de tickets no Jira em um backlog perdido que ninguém jamais tocará? Ou pior, eles nem mesmo são documentados de alguma forma? Ou você os avalia e testa constantemente para garantir que eles passem por uma análise de impacto ou de risco para garantir um melhor gerenciamento de resposta a incidentes quando as coisas dão errado? Ou até mesmo para saber quais são as consequências desse bug/vulnerabilidade em seu ambiente, caso ele se torne um ataque?

É muito comum ver empresas lidando com seus bugs, elas têm equipes de Q&A cheias de engenheiros para procurar problemas em seus produtos e isso é muito bom, elas têm desenvolvedores cada vez mais experientes, o que tende a evitar muitos bugs, bem,

pelo menos os mais bobos. Não é tão difícil encontrar empresas que aplicam técnicas como TDD diariamente ou até mesmo equipes de Q&A capazes de criar testes automatizados, isso é fantástico. Demorou algum tempo para que isso acontecesse. Lembro-me de quando comecei a programar, aos 15 anos, por volta de 2002 (última vez que o Brasil ganhou a Copa do Mundo na época em que este livro foi escrito), o desenvolvedor era o testador, havia equipes de teste, mas não tão comuns como agora. Especialmente porque, naquela época, o Visual Basic e o Deplhi eram o auge das linguagens de programação e o desenvolvimento da Web ainda estava crescendo e se tornando popular. Portanto, os sistemas de desktop cliente-servidor eram muito mais populares e, sejamos honestos, muito mais fáceis de testar e validar. Porém, o mesmo não se aplica às ameaças ou vulnerabilidades.

Os bugs são esperados, todos os conhecem, aceitam e até se divertem com eles. Como desenvolvedor, eu ficava feliz (às vezes) em encontrar um bug e corrigi-lo, o que significa que não "prejudicaria" meu cliente ou usuário final, certo? Mesmo quando o engenheiro de controle de qualidade relatava alguns bugs estúpidos, os chamados não bloqueadores, eu ficava feliz por fazer parte do aprimoramento do produto (software) para que ele fosse o mais perfeito possível, considerando, é claro, o tempo e os recursos para isso. Mas, quando você muda a palavra BUG para a palavra VULNERABILIDADE, a mágica acontece, ninguém mais se importa, e o que eu mais ouço sobre isso é (na ordem):

1. É um falso positivo; (99% dos casos)
2. Nosso produto/empresa/marca não é um alvo para hackers (crackers);
3. Isso nunca acontecerá, quem saberia disso?
4. É muito difícil para um invasor fazer isso.
5. Ah, se isso acontecer, teremos problemas maiores.

Com as vulnerabilidades, todo mundo parece ficar louco e a primeira reação é defensiva, evasiva ou protetora, como se fosse apenas uma categoria diferente de bug, nada mais. (Discutiremos isso mais tarde). Por enquanto, vamos voltar ao nosso super-herói e a mais coisas sobre Modelagem de Ameaças. Uma coisa boa que gosto de dizer sobre a Modelagem de Ameaças é que você precisa pensar no inimaginável.

As ameaças inimagináveis

Às vezes, você enfrentará ameaças que nunca imaginou. Bem, se você não está preparado para as mais fáceis, imagine as mais difíceis.

Cena que o Homem-Formiga destrói a armadura do Homem de Ferro - Capitão América: Guerra Civil 2016

Esta é uma das minhas cenas favoritas, há muito a aprender, portanto, mantenha-se concentrado agora. Primeiro, podemos ver que a primeira reação de Tony é perguntar a Friday o que está acontecendo quando ele percebe alguma falha nos propulsores de suas mãos. Veremos nos próximos capítulos deste livro mais detalhes sobre o Monitoramento de Aplicativos, mas, por enquanto, veja que ele não pergunta nada para ela diretamente, ele apenas chama seu nome e ela sabe o que responder na hora. Ela então diz: " Temos alguns sistemas de armas desligados", e a resposta de Tony é: "Eles o quê?", ele fica muito surpreso porque conhece seu "software", conhece seu "produto" porque o projetou, então como poderia ter alguns "sistemas de armas desligados" de repente? Depois, a parte mais engraçada é quando o Homem-Formiga começa a falar com ele: "É a sua consciência, não temos nos falado muito ultimamente." Nesse momento, você pode ver o rosto desesperado de Tony chamando por Friday mais uma vez, e ela decide automaticamente acionar o sistema de supressão de incêndio. Ela também não sabe ao certo o que está acontecendo, eles sabem o impacto, mas o que está causando esse impacto, não fazem ideia. Felizmente, o sistema de supressão de fogo funcionou e expulsou o Homem-Formiga de seu traje.

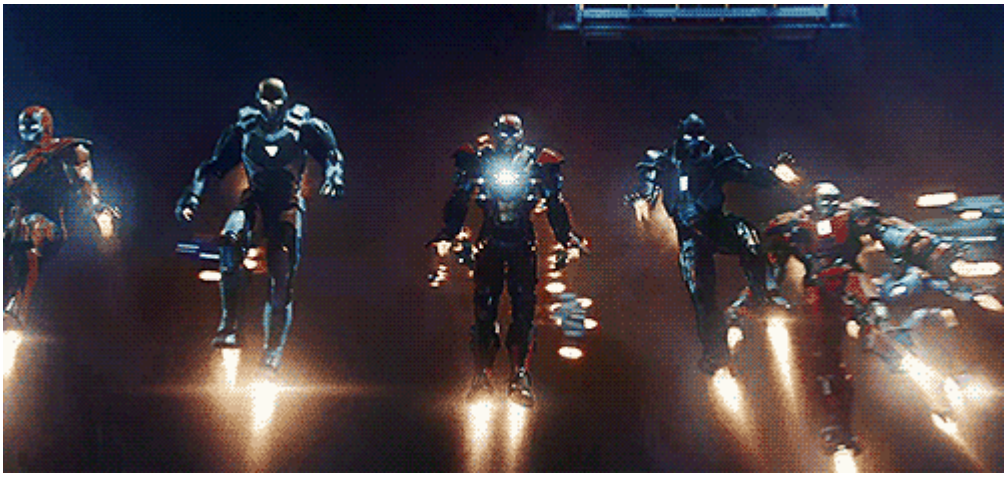
Uma das maiores responsabilidades dos participantes (especialmente os de segurança) durante uma sessão de Modelagem de Ameaças é imaginar as "coisas malucas", ou seja, pensar nas possibilidades de ataques, mesmo que às vezes pareçam impossíveis ou engraçados, a questão aqui é: qual é o impacto? Ele pode nos destruir ou apenas fazer cócegas? Tony nunca imaginou que um homem pequeno iria primeiro existir, depois entrar em seu traje e "desligar" alguns sistemas, e tudo bem porque, no final das contas, não podemos pensar em todas as possibilidades, isso é impossível por natureza. Mas considerando seu trabalho até agora, ele tinha um traje capaz de lidar com isso, se não ele estaria fora de combate, considerando também que o Homem-Formiga não o mataria, era uma luta de parar um ao outro, não de matar um ao outro. E terminamos com algumas perguntas: como está seu software hoje? Ele está preparado para ameaças leves? E quanto às ameaças graves? Ele consegue lidar com um simples ataque de SQLi? Ele consegue se manter resiliente durante um ataque DDoS? E quanto a um insider que tem acesso ao banco de dados? Pense nisso! Ah, e por falar nisso, ele lidou com a ameaça do Homem-Formiga com o traje nanotecnológico 🙂

Quando a Modelagem de Ameaças não funciona

Uma solução para cada problema?

Em Homem de Ferro 3, vemos a Legião de Ferro, basicamente um grupo de trajes independentes do Homem de Ferro, que eram controlados por Jarvis e podiam agir de forma autônoma para apoiar Tony Stark em qualquer situação. Ele chegou a esse ponto porque já estava obcecado com o fato de que os alienígenas poderiam invadir a Terra,

como vimos em Os Vingadores, e por isso começou a aprimorar ainda mais seus trajes e seu poder, pois tinha certeza de que ameaças maiores estavam por vir.



É claro que ele tentou criar uma solução para cada possível problema (ameaça) em Homem de Ferro 3 com a Legião de Ferro, mas é impossível dimensionar isso a essa altura. Sim, ele era um bilionário, mas ainda assim... como criar algo para uma ameaça que você não sabe como é? Mas ainda assim, uma boa tentativa que o ajudou muito nas versões futuras, esse foi um projeto MVP muito bom para o futuro traje nanotecnológico, eu diria.

E aqui está o ponto: se você tentar aplicar a modelagem de ameaças a todo o sistema, não funcionará, sabe por quê? Porque o sistema inteiro é muito grande, e nem tudo em seu sistema é crítico, sim, pode haver alguns cenários específicos, mas, em geral, os softwares ao redor são críticos em algumas funcionalidades específicas. Portanto, você só aplica a modelagem de ameaças aos cenários em que o risco de segurança significativo é visível e você precisa protegê-lo. Por exemplo, você tem um aplicativo de internet banking e, claramente, as transações em dinheiro têm mais risco do que apenas ver o histórico de transações. A modelagem de ameaças é um trabalho manual, difícil de automatizar e escalonar, portanto, o foco é muito importante. Outro conselho que posso dar é que você opte pelo nível granular, como, por exemplo, usar uma funcionalidade como login ou registro de usuário para modelar ameaças, e não o fluxo de trabalho da jornada do usuário, que é maior e parece ter muitas funcionalidades pequenas. Além disso, não faça o modelo de ameaça de uma função ou método, isso é um absurdo.

Lições aprendidas, sempre

Precisamos de bons exemplos, precisamos aprender e ter inspirações para sermos proativos, caso contrário, continuaremos a fazer segurança cibernética de forma reativa e, muitas vezes, não há tempo para reagir, por exemplo:

1. Em Capitão América Guerra Civil, durante a luta entre a Equipe do Homem de Ferro e a Equipe do Capitão, o Visão acertou o Patriota de Ferro por engano. Ele estava mirando o Falcão, mas acertou o Patriota de Ferro, fazendo com que ele perdesse a energia da fonte do traje, o que o fez cair.

Tony tentou voar e salvá-lo de cair no chão, mas não havia recursos suficientes naquele momento, então o Patriota de Ferro, também conhecido como Coronel Rhodes, amigo de Tony, foi atingido e ficou paraplégico, pois a queda poderia tê-lo matado.

2. Mais tarde, em Vingadores - Guerra Infinita, quando a nave espacial estava voltando para o espaço depois de invadir a Terra, Tony a perseguiu e, por um momento, a nave espacial foi mais rápida do que ele. Então, ele pediu um "apoio" a Friday e ela lhe deu um impulso, uma espécie de modo de velocidade de voo turbo, que o fez alcançar a nave espacial.

Mais uma vez, Tony está nos mostrando que os aprimoramentos constantes são necessários e NUNCA serão suficientes. Sempre haverá uma situação em que problemas,

riscos e danos o atingirão e como você lidará com isso? Quanto isso vai doer? Quanto custará? Ele tinha uma falha (potência de voo insuficiente), ele melhorou com o modo de impulso, fácil, certo? Pergunto novamente: seus softwares também são assim? Cada nova versão é aprimorada do ponto de vista da segurança?

Frameworks

Existem algumas estruturas de modelagem de ameaças por aí, para todos os tipos de modelos de ameaças que você deseja fazer, você pode encontrar algo que atenda às suas necessidades, mas aqui eu quero trazer a você, talvez, a lista mais popular de frameworks que você pode querer examinar:

- STRIDE
- PASTA
- LINDDUN
- Attack Trees
- Persona non grata (PnG)
- Security Cards
- Hybrid Threat Modeling Method (hTMM)
- Quantitative Threat Modeling Method
- Trike
- VAST Modeling
- Octave

Fique à vontade para experimentá-los. Uma simples lista de "Hello World" em cada um deles será útil para que você entenda seus pontos fortes e fracos, quais podem ser mais fáceis do que outros e quais podem ser perfeitos para o seu processo.

Conclusão

A modelagem de ameaças consiste em imaginar o inimaginável. Os bandidos não têm limites quando vão atrás de você, por que você deveria ter limitações em relação ao seu processo de segurança de aplicativos? Lembre-se de que, depois de Os Vingadores, Tony começa a se assustar com as ameaças que a Terra pode sofrer, porque em Homem de Ferro 1 e 2 as ameaças eram "homem", "tecnologia", armas etc., mas em Os Vingadores, alienígenas saíram do espaço em um buraco quente no meio de Nova York, então precisamos estar preparados, certo? Por causa disso, surgiram Utron (Vingadores 2) e a Legião de Ferro (Homem de Ferro 3), porque ele sabia que algo pior poderia acontecer a qualquer momento, e ele precisava estar preparado para salvar a Terra a qualquer momento de qualquer ameaça.

Não quero que você se apavore com a segurança (talvez eu me apavore), mas quero que você realmente se preocupe com o software que fornece, quantas pessoas podem ser afetadas por um problema de segurança que você possa ter? Como a sociedade pode ser afetada? A Modelagem de Ameaças não só ajuda a criar uma cultura de segurança, mas afeta diretamente a qualidade do seu software no final, já que você se preocupa com os requisitos de segurança desde o início do seu SDL.

Exemplo simples de uma Modelagem de Ameaça

Imagine esse Requisito Funcional

RF 1 - Login

A página de login deve ser a primeira página que os usuários veem no aplicativo modificado. Ela deve fornecer dois campos de texto - um para inserir um nome de login e outro para inserir uma senha. Além disso, ela deve ter um botão de comando que inicie a ação de verificação de senha. Se um dos campos de texto for deixado em branco, é um erro que deve ser informado ao usuário. Se ambos os campos forem preenchidos, mas não houver registro do nome de usuário ou se a senha estiver incorreta, isso também deverá ser informado ao usuário.

Agora, imagine algumas questões simples:

- Ataques de força bruta - deve haver um captcha que precisa ser resolvido antes do login.
- Criptografia - Tanto o nome de usuário quanto a senha devem ser criptografados usando o algoritmo xyz, seja durante o transporte dos dados ou durante o armazenamento.
- Engenharia social - a mensagem de login incorreto não deve especificar se o nome de usuário ou a senha estão errados para evitar que se diga que um ou outro está correto.
- Registro e monitoramento - cada tentativa de login deve ser registrada com pelo menos um mínimo de dados, considerando os "5Ws" (Where, What, Who, When e Why), inclusive o status (logins bem-sucedidos ou não)

O resultado dessas perguntas simples são alguns requisitos de segurança poderosos que tornariam o software mais protegido desde o início. Se essa página de login já estiver em produção, todos esses riscos também estarão, se as medidas de segurança forem implementadas desde o início, esses riscos serão mitigados por padrão. Além disso, e se esse "brainstorming de modelagem de ameaças" acontecer com base na lista de requisitos funcionais? Então, o próprio requisito de funcionamento poderia ser uma fusão

de requisitos funcionais e de segurança, evitando algum "desvio" nos requisitos de segurança, pois, como eles "não são requisitos de negócios", é fácil "não fazê-los" durante a implementação do projeto de software.

Capítulo 2

DevSecOps

"AppSec e DevSecOps são coisas diferentes!"
– Cássio B. Pereira

É óbvio que Tony Stark percebeu que carregar uma bagagem ou ir até a "garagem" para pegar seu traje não era a melhor maneira de tê-lo. Ele deve ser portátil, até mesmo de bolso, fácil de montar onde quer que ele esteja, fácil de carregar. Vimos em Homem de Ferro III que o protótipo do traje agora podia ter partes independentes, fáceis de montar, escalonáveis, cada parte podia literalmente ir até ele onde quer que ele estivesse e até mesmo ele podia optar por montar o traje em outra pessoa. O mesmo vimos em Capitão América Guerra Civil, quando ele usava um relógio simples que tinha recursos de defesa e ataque, e foi muito útil durante a pequena luta contra o Soldado Invernal, ele tinha um "flash bang" e um pulso sonoro para deixar o inimigo atordoado, além do material à prova de balas que o salvou de ser atingido.

Vamos pensar juntos: o que é melhor ou mais fácil, proteger um sistema monolítico inteiro ou diferentes microsserviços? A questão é que você não precisa proteger toda a sua casa, apenas o cômodo onde armazena os ativos mais valiosos. Ou a casa inteira, se for o caso, dependendo da importância de seus negócios.

Para Tony, o principal ativo a ser protegido é ele mesmo, sua vida como ser humano. Ok, o traje poderia ser um protetor de peito para o coração e um capacete para

a cabeça? Sim, mas ainda assim ele estaria vulnerável, portanto, o traje para o corpo todo é mais viável. Por sorte, nosso desenvolvimento de software moderno não é assim, talvez queiramos proteger a API de pagamentos (api.system.com/payment) em vez da API de categorias de lista (api.system.com/listCategories), por exemplo, certo? Ambas fazem parte do mesmo sistema, mas é evidente que uma tem muito mais impacto nos negócios do que a outra, as PII devem estar envolvidas e assim por diante.

Tentarei concentrar este capítulo nas coisas do DevSecOps, pontos de contato do AppSec. Toda a abordagem do AppSec diz respeito a cuidar do aplicativo em si, do software em si, do código, dos arquivos .exe .jar etc. etc. A abordagem DevSecOps protege tudo o que está envolvido na criação do aplicativo, toda a cadeia de suprimentos (sim, não apenas as bibliotecas de terceiros), mas também o IDE, as ferramentas de CI/CD, o sistema de tíquetes, os bate-papos etc., tudo o que faz parte do SDL. Para entender melhor, vamos ver o Manifesto DevSecOps abaixo e também recomendo que você consulte o site deles:

Leaning in *over Always Saying "No"*
Data & Security Science *over Fear, Uncertainty and Doubt*
Open Contribution & Collaboration *over Security-Only Requirements*
Consumable Security Services with APIs *over Mandated Security Controls & Paperwork*
Business Driven Security Scores *over Rubber Stamp Security*
Red & Blue Team Exploit Testing *over Relying on Scans & Theoretical Vulnerabilities*
24x7 Proactive Security Monitoring *over Reacting after being Informed of an Incident*
Shared Threat Intelligence *over Keeping Info to Ourselves*
Compliance Operations *over Clipboards & Checklists*

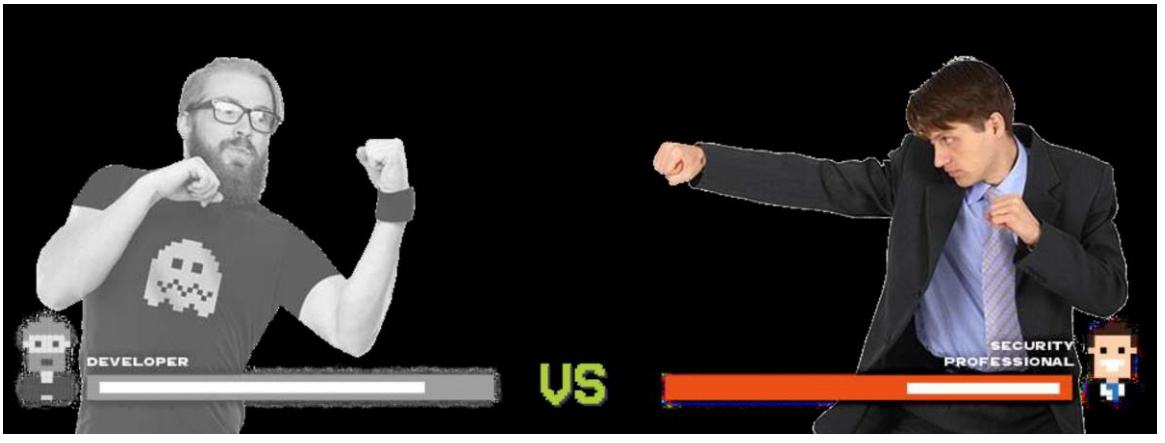
Esses nove pontos descrevem como as coisas devem ser e podem ser um pouco complicados de entender em um primeiro momento, mas tentarei abordar os tópicos durante este livro. Por enquanto, quero me concentrar em 3 ou 4 pilares que considero necessários para que o DevSecOps realmente funcione sem problemas:

- Cultura (pessoas) - Fazer com que todos na SDL participem.
- Processo (estratégia) - Para ter uma forma estruturada de como fazer.
- Ferramentas - Caso contrário, você não conseguirá expandir.
- Automação - a automação é a chave para que seu AppSec/DevSecOps seja dimensionado e funcione adequadamente.

Para entender um pouco melhor, criei o seguinte quadro para deixar claro como é o mundo hoje em dia se não considerarmos a abordagem DevSecOps:

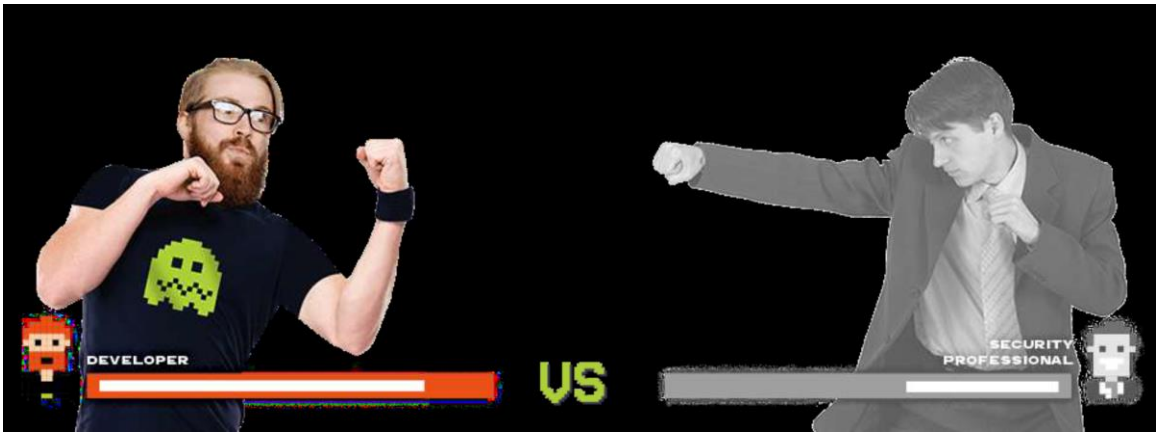
DevSecOps Antes - Times de Segurança

- Diga não à maioria das coisas.
- Descobrir vulnerabilidades na produção.
- Não sabe como lidar com outras equipes.



DevSecOps Antes - Times de Desenvolvimento

- Não prioriza a correção de vulnerabilidades;
- Acredita que a melhor solução para o problema é a sua;
- Seu código não trava;



Claramente, as equipes não conversam nem interagem de forma adequada, toda a comunicação e interação tem a ver com poder e ego, e os desenvolvedores e os profissionais de segurança podem ser muito irritantes com isso, acredite, já estive em ambos os lados, então a filosofia DevSecOps é:

Cultura - Pessoas 🤝

Bom ✅

- Saber dizer sim com responsabilidade;
- Disseminar as preocupações com a segurança;

Ruim ❌

- Querer participar de todas as decisões;
- Praticar a segurança por obscuridade;

Processo - Estratégias ⚙️

Bom ✅

- Comece com processos simples;
- Sempre produza métricas;

Ruim ❌

- Ignorar processos para implementações rápidas;
- Querer automatizar tudo;

Ferramentas - Escala ⛰️

Bom ✅

- Adotar soluções Open Source;
- Veja como outras empresas fazem isso;

Ruim ❌

- Não querer gastar $$$ em ferramentas;
- Querer usar todas as disponíveis;

Automação 😈

Bom ✅

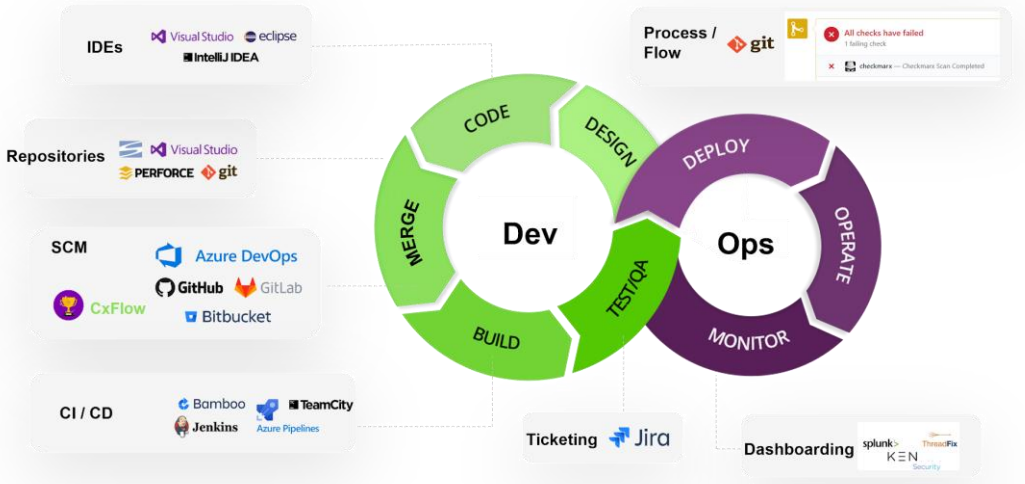
- Automatizar um fluxo inteiro, não parte dele.

Ruim ❌

- Tentar automatizar tudo.

Automação como um fator crítico de sucesso

Todo o SDL (Software Development Life Cycle) é um ecossistema complexo, considerando o número de ferramentas, processos e integrações existentes, cuja escala já é incontrolável:



E sem alguma automação, o simples objetivo de criar um software simples pode se tornar um pesadelo. Da IDE à nuvem, passando pelo sistema de tickets, SCM, ferramentas de CI/CD, painéis de controle e fluxos, como tudo pode funcionar manualmente? Eu diria que é uma mistura, automatize sempre que puder, mas você sobreviverá com uma ou duas tarefas manuais.

Conclusão

O DevSecOps faz parte do AppSec, o que significa que o DSO é outra área que requer atenção. Não são apenas as ferramentas de segurança integradas no pipeline ou a verificação automática de seus PRs, mas muito mais, por exemplo:

- Essas ferramentas estão configuradas corretamente? Seus scanners AST?
- O SAST tem uma taxa adequada de falsos positivos? Você o verificou? E quanto aos verdadeiros positivos?
- O SCA está resolvendo as dependências corretamente? Não está faltando nada?
- As etapas do pipeline têm gerenciamento de acesso adequado? Quem pode desativar uma etapa de segurança?
- Você coleta KPIs dos seus scanners AST?
- Como está o endurecimento das máquinas dos desenvolvedores?

Capítulo 3

Gerenciamento de Vulnerabilidades

"Conheça seus problemas!"
– Cássio B. Pereira

Ousaria dizer que o gerenciamento de vulnerabilidades é o ponto em que a maioria das empresas falha na implementação de um programa de AppSec. Esse é apenas um processo normal e, ainda assim, as empresas o negligenciam. Por que é tão difícil rastrear um bug ou uma vulnerabilidade? Vamos imaginar um fluxo simples: a equipe de segurança ou qualquer pessoa no mundo encontrou uma vulnerabilidade e a relatou à empresa, alguém a validou e a encaminha para a equipe/pessoa responsável por corrigi-la. Deve haver alguma priorização, pode ser algo muito urgente ou algo que pode esperar X tempo. Em seguida, uma correção é aplicada, alguém a testa, implementa na produção e pronto. Simples, certo?

Acho que, a essa altura, está bem claro que Tony Stark cuida de suas coisas. Não é por engano que cada novo traje, ou marca, tem uma melhoria. Seja no poder de defesa ou no poder de ataque, às vezes até em ambos. Lembremos que, quando ele lutou contra o Capitão América e o Soldado Invernal no filme Guerra Civil, seu sistema de mira foi danificado e ele não conseguiu atirar corretamente, obrigando-o a fazer a mira "manualmente". Mais tarde, quando o Homem de Ferro lutou contra Thanos em Vingadores - Guerra Infinita, ele agora era capaz de disparar mísseis guiados remotamente, mesmo sem a necessidade de mirar.

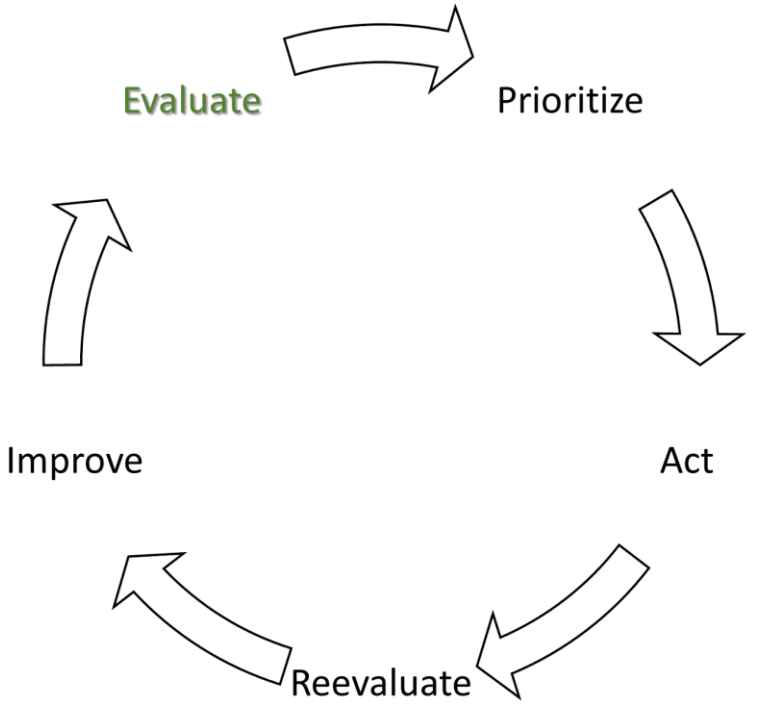
E quanto ao seu software? Você tem uma lista de bugs? E uma lista de vulnerabilidades? E uma lista de melhorias de segurança? Talvez requisitos de segurança? Ok, não se preocupe…

Processo de Gerenciamento de Vulnerabilidades

Para atingir a meta de realmente gerenciar suas vulnerabilidades, você precisa se preparar para isso. Não é possível monitorar/manusear todos os ativos da sua empresa, pois a maioria deles nem é crítica o suficiente para que você gaste tempo/dinheiro com eles (para essa finalidade). Portanto, primeiro você deve fazer uma preparação, que é composta por:

- Determinar o escopo;
    - O que fará parte de seu gerenciamento de vulnerabilidades.
- Definir funções e responsabilidades;
    - Quem será responsável pelo quê?
- Selecione as ferramentas de avaliação;
    - Quais serão seus scanners?
- Defina políticas;
    - Quais são as regras para os resultados?
- Identifique os ativos;
    - Quais sistemas, repositórios etc. você vai verificar?

Em seguida... você aplica esse fluxo simples:



Vou explicar um por um usando um exemplo de varredura SAST:

- **Avaliar (varredura)**

    - Execute a verificação SAST em seu repositório.
- **Priorizar**
    - Obtenha os resultados e priorize-os. Por exemplo, os resultados da varredura SAST mostraram apenas uma vulnerabilidade ALTA. Mas, como se trata de um impacto em um sistema legado que não tem impacto comercial crítico para você, você pode marcá-lo como MÉDIO.
- **Agir (corrigir o problema)**
    - Resolva essa vulnerabilidade usando seu sistema de emissão de tickets, por exemplo, e defina o SLA esperado considerando a vulnerabilidade MÉDIA.
    - O desenvolvedor fará a correção e a disponibilizará para teste.
- **Reavaliar (verificar novamente)**
    - Examine novamente o mesmo repositório, com as mesmas configurações.
    - A vulnerabilidade não deve mais estar presente.
- **Melhorar**
    - Pense em maneiras de automatizar:
        - A varredura
        - O rastreamento dos resultados
        - A notificação dos resultados
        - O tempo de fixação
        - A nova varredura
        - etc.
- Repetir…

Esse macroprocesso funciona muito bem para qualquer verificação/teste de segurança, com pequenas alterações de acordo com cada teste, se realmente for necessário, mas a ideia é a mesma: você testa, prioriza, corrige, válida e melhora.

Ferramentas

Você pode usar qualquer sistema de gerenciamento de bugs, sistema de tarefas, sistema de emissão de tickets etc. que funcionará da mesma forma. O segredo aqui é centralizar suas vulnerabilidades em um único local. Você pode ter uma varredura SAST, SCA e DAST e deseja consolidar as vulnerabilidades para otimizar o gerenciamento. Uma boa ferramenta para usar é o Defect Dojo da OWASP. Ou você ainda pode usar:

- Jira (Popular)
- Excel (😶‍🌫️)
- Github Issues (Um bem legal)
- Notepad! (Brincadeira, **ou não**)

Referência

Não sou Tony Stark (ainda), portanto não criei nada disso, apenas pesquisei e apliquei o que, na minha opinião, é o processo mais simples de fazer o gerenciamento de vulnerabilidades. Aqui está a referência e o PDF.

Capítulo 4

Monitoramento de Aplicações

"Conheça sua aplicação antes do seu inimigo."
– Cássio B. Pereira

Sir, Take a deep Breath Iron Man 3 (Clip) 2013

Jarvis ajudou, Jarvis SABIA O QUE DEVERIA FAZER NO MOMENTO CERTO, você deve estar se perguntando porque estou gritando? PORQUE OS DESENVOLVEDORES, PORRA, NÃO CONHECEM SEUS PRÓPRIOS SISTEMAS, e isso me irrita. Portanto, vou lhe fazer algumas perguntas:

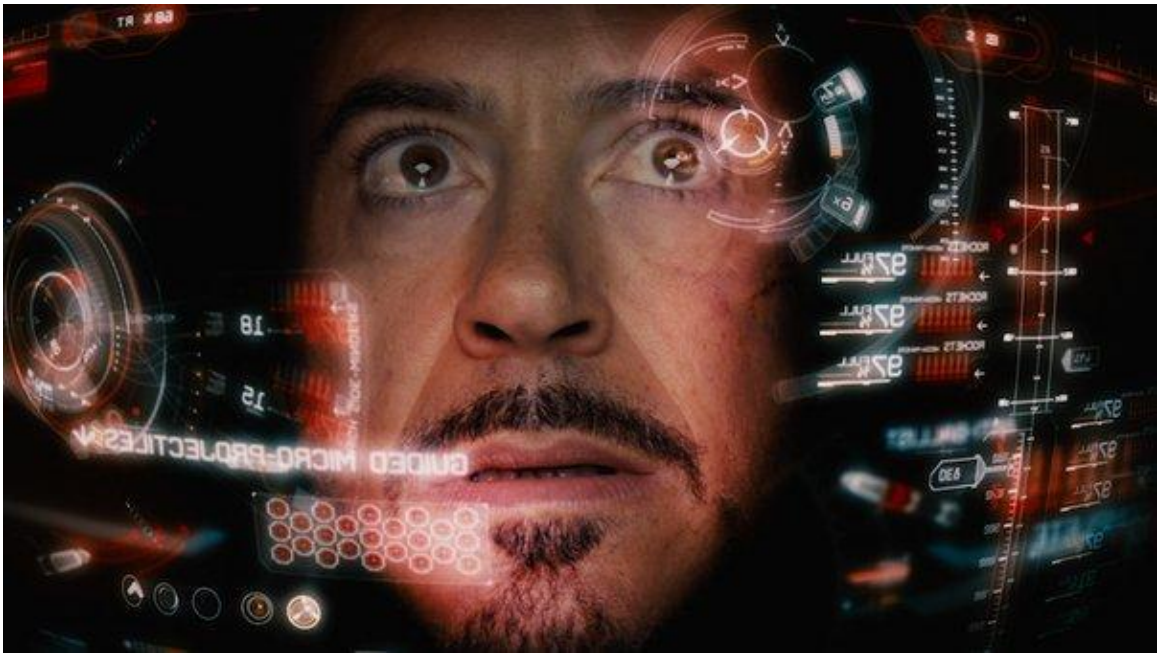
- Como a sua aplicação se comporta sob ataque?
- Ela simplesmente cai?
- Você tem algum registro?
- Alertas automáticos?
- Você sabe ao menos que está sendo atacado?

Depois de duas décadas de trabalho no setor, tive a oportunidade de conhecer uma variedade de pessoas e empresas diferentes e, sem nenhum arrependimento, posso afirmar que 90% dos desenvolvedores nem sequer sabem o que estão construindo, eles conhecem a tarefa, o recurso, mas o todo? Não. Eles conhecem o produto, o nome, o objetivo, sim... mas quando você pergunta sobre o aplicativo como um todo, pelo menos uma visão macro, eles não têm a menor ideia. E tudo bem, eles devem se concentrar em sua "tarefa, recurso, ticket etc." e não no todo, mas, durante um incidente de segurança, quando a equipe de DFIR precisa agir para conter a ameaça ou apenas investigar, ninguém tem a menor ideia de todo o sistema, quais são os serviços, quem fala com cada um deles, onde está hospedado o banco de dados, quem é o administrador, quem é o proprietário do produto responsável pelas decisões... Tantas perguntas que, em teoria, são fáceis de responder, mas durante uma crise, as equipes de DFIR estão SOZINHAS, elas precisam fazer uma PESQUISA interna primeiro para saber tudo isso, depois podem começar a investigar o incidente em si, e esse tempo geralmente é muito importante.

Portanto, o que quero dizer aqui é que a colaboração dos desenvolvedores deve ser levada muito mais a sério. Eles sabem como o aplicativo funciona/foi criado, devem saber, não apenas para facilitar seu próprio trabalho, mas também para apoiar toda a estrutura da empresa em relação à postura de segurança. Certa vez, tive um incidente em uma empresa para a qual trabalhei: algumas APIs estavam sendo abusadas por bots, aumentando a taxa de solicitações para 500.000 em uma hora durante a manhã. A primeira coisa foi: me dê os registros, quero ver o IP da solicitação, a geolocalização para entender o cenário, mas não havia registros. Tudo bem. Ligar os registros, ESPERAR O BOT EXECUTAR DE NOVO, basicamente, esperar ser atacado de novo para que eu pudesse ter dados para verificar, não poderia apenas bloquear solicitações, bloquear o quê? Depois o BOT novamente, endereços IP vindos da Amazon... Depois de duas semanas tentando

entender (e eu já tinha as suspeitas), durante um café informal com um desenvolvedor, ele mencionou que seus TESTES NAS APIs estavam bem, pedi detalhes e para minha surpresa (ou não), era ele apenas testando as APIs que estavam "sob ataque". Mas até então, ninguém sabia nada sobre essas APIs, quem tinha as URLs, quem tinha acesso a elas (eram públicas, mas supostamente apenas para alguns clientes), quem era o responsável pelo desenvolvimento (poucas equipes eram), enfim, muitas questões simples, que em uma empresa sem o mínimo de organização e disciplina, durante uma crise, podem ser fatais.

Você sabe qual é o comportamento normal de suas aplicações?



Tony Stark sabia (com o apoio de sua assistente de I.A., é claro), a qualquer momento, um monte de informações úteis, ele tinha todos os tipos de dados relativos ao seu traje, é claro, além de muitos dados sobre o ambiente externo, como tráfego de controle aéreo, clima, perímetro terrestre etc. Veja a próxima cena: em tempo real, Jarvis define uma rota de voo para ultrapassar os drones, fazendo até mesmo com que alguns deles caiam:

Esse é um exemplo perfeito de monitoramento de aplicativos: você precisa ter as informações para saber o que está acontecendo e decidir o que fazer, especialmente se estiver sendo atacado. E não apenas ter as informações, mas as informações certas no momento certo.

Informação é a chave

A informação é TUDO para a tomada de decisão correta. O traje de Tony está conectado a várias fontes de dados, JARVIS sabe tudo para mantê-lo informado a qualquer momento, e às vezes sem pedir, apenas algumas coisas que Jarvis "TEM DÚVIDA". Não apenas confiando nas fontes de dados, mas também processando informações em tempo real, como nesta cena:

Iron Man Plane Rescue Scene - Iron Man 3 (2013) Movie CLIP HD

Viu que, durante todo o tempo, Jarvis o mantém informado sobre a altitude e mostra na tela o "plano de rota" para pegar cada pessoa? Isso é incrível e um bom exemplo dos dados corretos no momento certo durante uma crise. E tudo isso com um traje de controle remoto, pois Tony não o estava usando, como podemos ver no final da cena. Está vendo como às vezes precisamos de informações rápidas para tomar uma decisão difícil?

Logs

Em computação, data log é uma expressão usada para descrever o processo de registro de eventos relevantes em um sistema de computador. Esse registro pode ser usado para restaurar um sistema ao seu estado **original** ou para permitir que alguém conheça seu **comportamento** no **passado**.

NO

LOGS

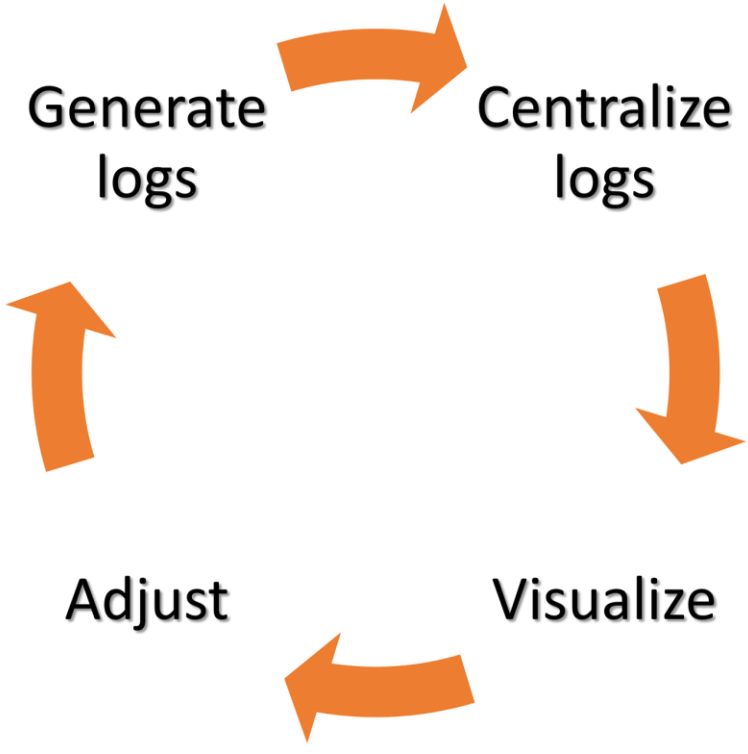
NO

CRIME

É tudo uma questão de **comportamento**. Conhecer o comportamento do seu software é essencial para a proteção adequada. Somente com o monitoramento contínuo é possível conhecer o comportamento do seu software. Depois de conhecer/determinar o comportamento normal, é fácil conhecer/determinar o comportamento anômalo, por exemplo:

Sua API de autenticação tem 500 solicitações diárias com base nos dados observados durante 30 dias. Podemos presumir que esse é o comportamento normal de sua API de autenticação, o número de solicitações diárias é 500. Então, um dia, você tem 500 por minuto em sua API durante as primeiras horas do dia. Qual é a primeira reação? Claramente, algo **anormal** está acontecendo, você dedica algum tempo para entender e reagir/tomar uma decisão sobre o que fazer, mas, sem esses dados, seria apenas um "dia normal", talvez o sistema ficasse um pouco lento, mas nada muito preocupante.

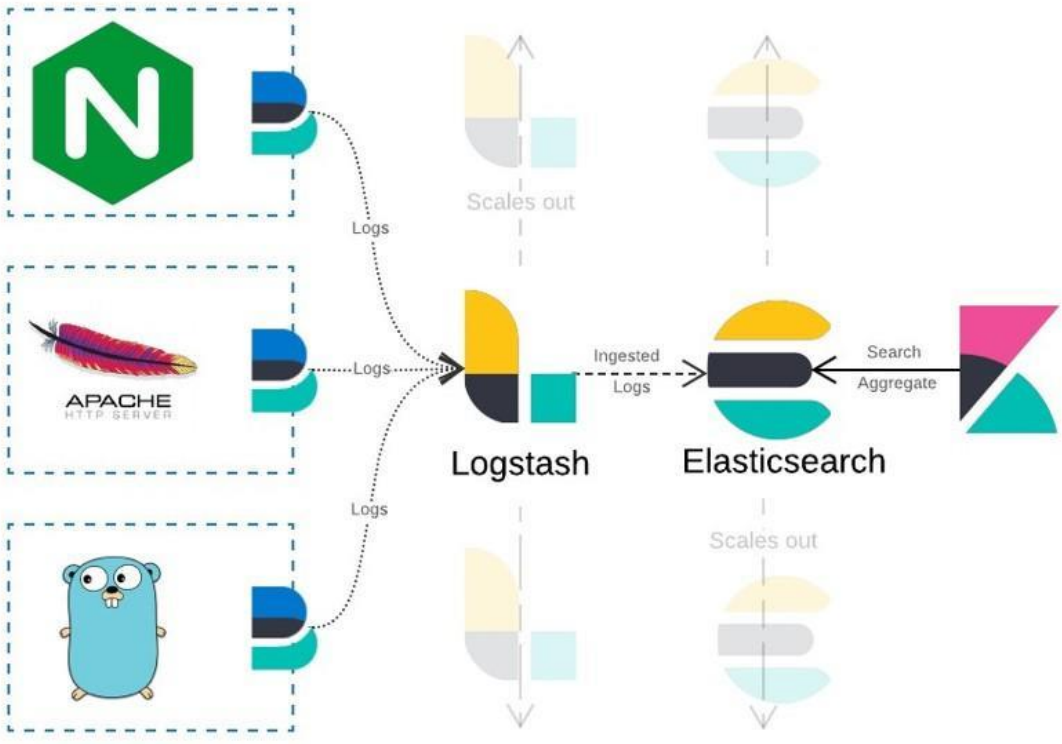
O gerenciamento de registros é simples, mesmo que possa ser caro:



- **Geração de registros**
  - Faça com que seu aplicativo produza os logs de que você precisa, inclusive a arquitetura de perímetro, como firewalls, wafs, cdns etc.
- **Centralize os registros**
  - Armazene esses logs de forma centralizada, facilmente acessível e fácil de gerenciar e correlacionar.

- **Visualize**
    - Crie exibições e dashboards que apresentem os dados de registro de forma amigável, pois a leitura de registros pode ser demorada e desagradável.
- **Ajuste**
    - Exclua dados irrelevantes e adicione dados relevantes. Talvez você queira saber o endereço IP de uma solicitação, mas não o tamanho dos bytes, dependendo do contexto.

Uma maneira muito fácil de fazer isso é usar o ELK Stack, mas você pode fazer isso usando qualquer outra solução de registro ou SIEM:



Capítulo 5

Programa de Segurança de Aplicações

"Segurança é trabalho de todos."
– Cássio B. Pereira

Pontos a se considerar

Todos nós já sabemos que Tony Stark não só ficou "louco" com sua segurança e proteção, mas também com a segurança e proteção do planeta Terra, principalmente depois de Os Vingadores, quando alienígenas saíram de um buraco de minhoca no meio da cidade de Nova York. Ele, então, começou a contaminar todos com suas ideias, Ultron foi o projeto de criar um "traje ao redor do mundo", que fracassou, sim, mas ainda assim foi uma tentativa 🙂. Então, para evitar que seu programa AppSec fracasse, aqui estão alguns pontos a serem considerados:

- Definir projetos críticos;
    - Quais são os projetos realmente críticos para sua empresa?
- Comunique as equipes;
    - Elas devem saber sobre o programa AppSec, como, quando e por que são as informações mínimas que você precisa fornecer.
- Realize treinamentos;
    - Cada público tem necessidades específicas; os desenvolvedores devem aprender a corrigir uma vulnerabilidade, mas as perguntas e respostas devem saber como explorá-las, por exemplo.
- Nomeie o proprietário da segurança;
    - Deve haver um proprietário, alguém da equipe, do esquadrão, da tribo ou de qualquer outro setor que seja responsável pelo chapéu de segurança.
- Defina linhas de base;
    - Quais são os requisitos mínimos que você deseja implementar? Por exemplo, todo o código deve passar por uma verificação SAST antes de ser mesclado com a ramificação principal.
- Defina os níveis de aceitação;
    - Quais são os riscos que você aceita? Por exemplo, você pode definir que todas as vulnerabilidades ALTAS e CRÍTICAS devem ser corrigidas antes da implementação.
- Defina a estratégia de CI/CD;
    - A segurança deve fazer parte dela, não importa como você faça isso, inclua a segurança em sua CI/CD. Varreduras, verificações e testes são exemplos de tarefas que podem fazer parte dela.
- Gerar métricas e KPIs;
    - É importante saber como está o desempenho da sua estratégia de AppSec, e definir e coletar KPIs é importante para medir isso.
- Gamificação
    - Às vezes é útil, mesmo que você considere que os desenvolvedores não devam ser recompensados por corrigir vulnerabilidades, por exemplo,

porque eles não deveriam criá-las em primeiro lugar, é uma ferramenta a ser considerada, com certeza.

Voltando ao Tony Stark, ele pensa em segurança, ele respira segurança, ele vive segurança, ele é neurótico com relação à segurança. Normalmente, essa é a mentalidade da equipe de segurança, e isso está errado. Todos são responsáveis pela segurança. Imagine o seguinte cenário: você mora com alguém, um de vocês sai de casa mais cedo do que o outro, mas alguém deixou a porta da frente aberta, alguém entrou durante a sua ausência e roubou todos os seus bens, como TV, videogame, barra de som, laptop etc., causando um enorme prejuízo financeiro para ambos. De quem é a culpa? Na verdade, quem deixou a porta da frente aberta? Você, que saiu primeiro? Seu parceiro que saiu depois? Se foi você, por que seu colega não verificou? Você percebe que isso não importa? No final, a perda afetou vocês dois, todos os seus bens valiosos foram roubados, se você ou seu parceiro deixaram a porta aberta é o mínimo de seus problemas.

Espero que, ao ler este livro, você descubra que também é responsável pela segurança, dado seu cargo e posição, com mais ou menos responsabilidades ou tarefas, isso é certo, mas, de uma forma ou de outra, você não está excluído da equipe de segurança, talvez deixe a porta da frente aberta, talvez não verifique se alguém deixou a porta da frente aberta, mas você também tem responsabilidade, e a próxima cena explica por quê:

Iron Man 3 2013 ► Tony Stark vs Ellen Brandt Bar Bud Light Scene ► Movie CLIP 4K Ultra HD

Tony improvisa quando necessário, mesmo sem seu traje, ele conseguiu afastar uma ameaça real e difícil. Sua vida estava em perigo, então, mais uma vez, com as ferramentas e as condições que tinha, ele lidou com a situação. Muitas vezes, vejo engenheiros de SDL (qualquer pessoa envolvida em SDL) dizendo: "mas não tenho o IDE X, não tenho o orçamento Y, prefiro usar a estrutura Z, blá blá blá", todas desculpas para não fazer a coisa certa, todas "saídas" para não assumir a responsabilidade de segurança inerente à sua função.

Capítulo 6

Por que?

"Crackers, criminosos cibernéticos, guerra cibernética... Somos apenas uma empresa comum."
– Uma empresa comum

Uma vez Tony Stark disse:

"Deuses, alienígenas, outras dimensões…Eu sou só um homem numa lata."

Stark, Tony - Homem de Ferro 3

Uma vez algumas empresas disseram:

"Crackers, criminosos cibernéticos, guerra cibernética... Somos apenas uma empresa comum."

Uma empresa comum

E essa mentalidade tomou conta das mentes de milhões de executivos, engenheiros e profissionais do setor de software (T.I.), o que é mais ou menos aceitável se analisarmos a natureza de alguns negócios, como um simples site estático para mostrar o que uma empresa faz, que pode não ser um alvo para um criminoso cibernético à primeira vista, em comparação com um aplicativo bancário em que bilhões de transações em dinheiro real são processadas diariamente, mas, da perspectiva de um grupo de crime cibernético organizado, ambos são iguais e têm, talvez, o mesmo potencial de ganho.

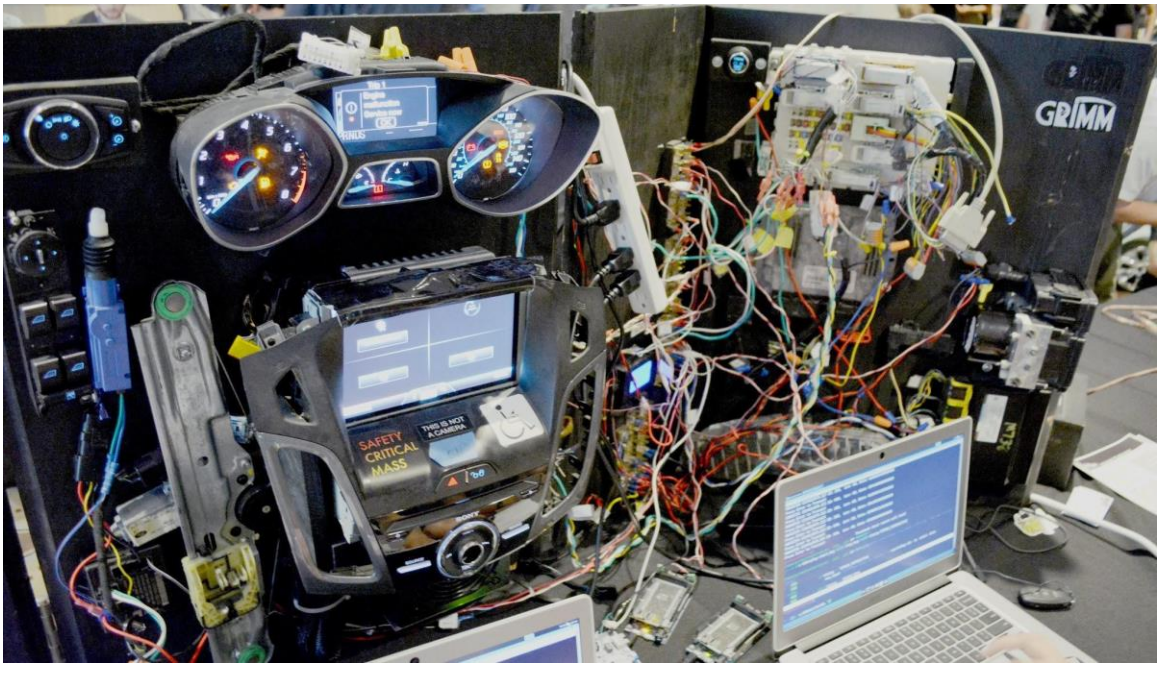
E você pode pensar o mesmo, Cássio, eu trabalho em uma empresa de software, mas nosso software não é crítico, é apenas um sistema de gerenciamento de armazém para materiais de marketing, como canetas, banners e brindes, o que pode dar errado? Não se trata da operação da empresa nesse caso, mas dos dados que essa empresa possui. Se eu quiser prejudicar alguém, posso me interessar pelos eventos que essa empresa promove, já que você armazena o material de marketing, quando um grande pedido é feito, algo pode acontecer, então posso rastrear para entender a operação e conseguir o que quero como um "ator malicioso". Ou posso me concentrar apenas no desvio de materiais para venda, por exemplo. Lembre-se sempre de que, para os agentes mal-intencionados, não há limites para o que eles podem fazer.

Portanto, sua empresa pode ser um alvo apenas por ser uma marca ligada a algum marketing que deu errado, ou um alvo pelos próprios ativos, como dinheiro, materiais ou recursos valiosos, às vezes por se posicionar publicamente para o lado A ou B do governo ou até mesmo devido a um caso público em que essa empresa pode ter feito algo ruim com um ex-funcionário. Não importa o motivo (neste caso), mas se você é um alvo para alguém, talvez ainda não tenha sido encontrado.

Por que? Bem...

Algumas pessoas talvez ainda não tenham percebido, mas a porra da nossa vida é controlada e guiada por software EM TODA PARTE, desde a cadeia de produção de alimentos até os pagamentos e compras que fazemos, há um software em algum lugar,

portanto, proteger o software hoje em dia não é mais proteger seus bolsos ou empresas, é proteger a realidade e a própria sociedade.



Se seu carro tiver uma falha de software, e daí? Não é necessário ser o modelo das notícias, pois qualquer carro antigo com injeção eletrônica de combustível já é afetado por um software. Portanto, se você for um executivo poderoso de uma grande empresa ou um membro do governo, poderá ser um alvo para alguém que queira prejudicar você ou sua empresa de alguma forma.

Conclusão

Talvez sua empresa não seja um provedor de infraestrutura crítica ou um negócio em si, mas qual seria o impacto na sociedade (na vida das pessoas) se você tivesse um incidente neste momento? Pense nisso na próxima vez que você codificar!

- Algumas pessoas perdem dinheiro?
- Algumas pessoas não poderão pagar por algo?
- Os funcionários de sua empresa perderão seus empregos?
- O que mais?

Lembrem-se:

- Tudo é controlado por software;
- Precisamos construir uma sociedade melhor;
- Precisamos de mais proteção individual;
- Porque nossas vidas dependem disso.

Capítulo 7

Bônus

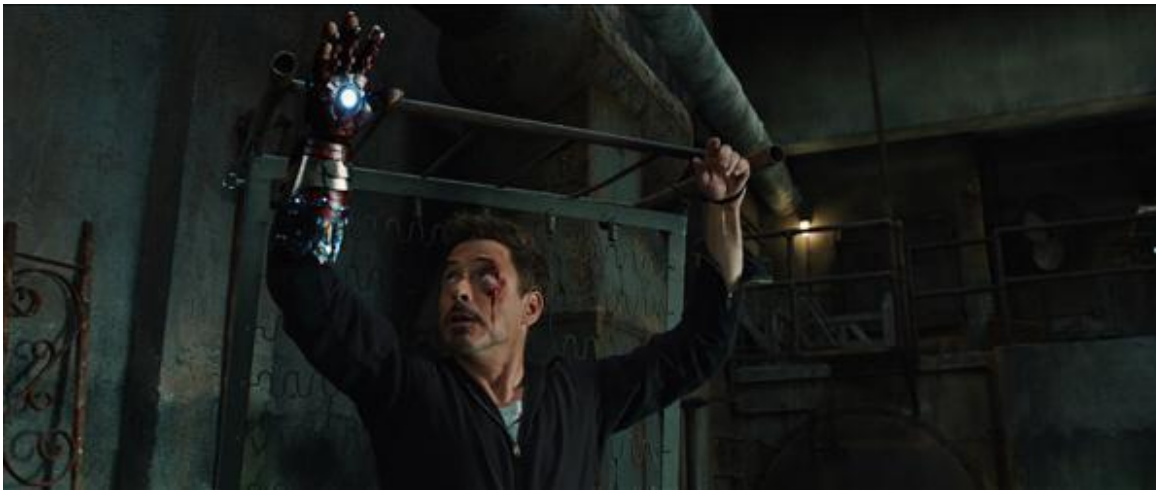
"Porque eu gosto muito do Homem de Ferro."
– Cassio Pereira

Nessa cena de Vingadores - Guerra Infinita, a equipe tentou tomar a manopla de Thanos e quase conseguiu, mas no último momento ele acordou e a recuperou.

Sabendo disso, Tony Stark preparou seu traje nanotecnológico para ser capaz de se incorporar a qualquer outro material, de modo que, quando teve a oportunidade de pegar a manopla de Thanos novamente em Vingadores - Ultimato, ele conseguiu fazer isso apenas tocando-a.

Vemos mais dessa capacidade depois em Homem-Aranha Sem Volta Para Casa, quando Peter (o verdadeiro) foi atacado pelo Dr. Octopus e seus tentáculos tocaram o traje nanotecnológico de Peter, a nanotecnologia foi automaticamente incorporada ao tentáculo, tornando-o capaz de ser controlado por Peter.

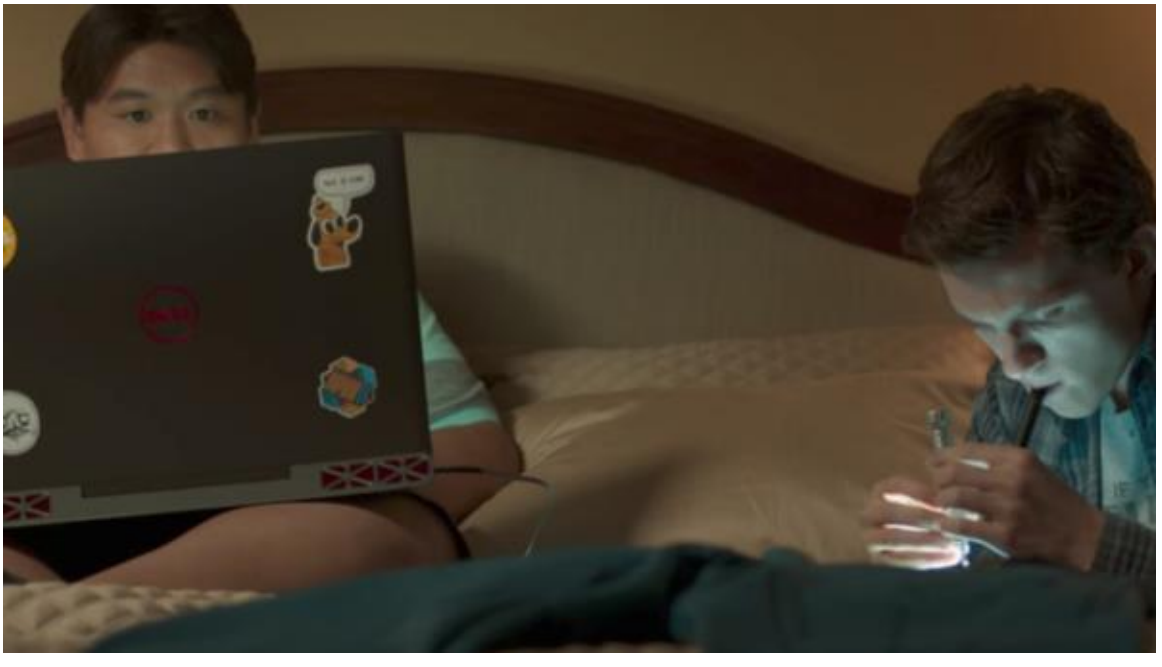
Em Homem de Ferro 1, ele foi sequestrado pelos terroristas e mantido em uma caverna onde era impossível rastreá-lo e encontrá-lo.



Depois, em Homem de Ferro 3, seu traje agora podia vir de qualquer lugar, então havia um sistema de rastreamento, além da capacidade de ser armado com seu traje, ou pelo menos parte dele, como vimos no filme. Além disso, ele era capaz de improvisar, já que seu traje vinha parte por parte, ele tinha que lutar para se libertar mesmo sem o traje completo.

E não vamos nos esquecer do poderoso relógio que ele tinha, naquele momento em que o traje não podia ser usado, pois ele se submeteu ao Tratado de Sokovia que deu origem à Guerra Civil. Mas, ainda assim, sem o traje, ele não estava totalmente vulnerável e desarmado.

O Homem-Aranha também tinha um sistema de rastreamento em seu traje, quando Tony Stark o convidou para os Vingadores e fez um novo traje para ele, pois queria proteger o garoto (já que sabia o que poderia acontecer). É claro que Peter hackeou o traje e removeu o rastreador para fazer alguma coisa, mas tudo bem.



Em Homem de Ferro 1, o Obadiah usou algum tipo de arma sonora para deixar Tony paralisado, você pode ver isso aqui Obadiah Stane Steals Tony Stark's Arc Reactor Scene | Iron Man (2008) Movie CLIP HD

Mais tarde, em Vingadores - Era de Ultron, o "mesmo som" afetou todos na sala, mas não Tony. Você pode ver a reação dele nos primeiros 5 segundos do vídeo a seguir, não é necessário assistir a tudo: Avengers vs Ultron First Fight in 4K Ultra HD [Fight Scene]

Capítulo 8

O FIM

"Preste atenção nessa cena!"
– Cassio Pereira

Tony Stark & Captain America Argument Scene - Avengers: Endgame Movie (Scene) | Movie Clip HD

Aqui está a transcrição da cena acima, e eu comentarei mais tarde, mas assista novamente! As partes mais importantes estão em **negrito**:

(Transcrição em português)

- 00:00:00.080 Faz 23 dias que o Thanos veio para a
- 00:00:02.080 Terra
- 00:00:04.720 governos mundiais estão
- 00:00:06.319 em pedaços… as partes que ainda estão
- 00:00:08.639 funcionando
- 00:00:09.679 estão tentando fazer um censo e parece que ele fez
- 00:00:11.840 ele
- 00:00:14.920 fez
- 00:00:17.199 exatamente o que disse que faria, Thanos eliminou
- 00:00:21.359 50%
- 00:00:22.720 de todas as criaturas vivas
- 00:00:28.560 Onde ele está agora? Onde?
- 00:00:30.800 não sabemos
- 00:00:32.640 ele simplesmente abriu um portal e
- 00:00:34.640 entrou
- 00:00:38.160 qual o problema dele?
- 00:00:40.160 uh ele está furioso
- 00:00:42.640 ele acha que falhou
- 00:00:45.200 o que é verdade
- 00:00:46.719 mas muitos estão sentindo o mesmo
- 00:00:48.079 não é? Sinceramente
- 00:00:50.559 pensei que você fosse de pelúcia talvez eu seja
- 00:00:52.320 estamos caçando o Thanos há 3
- 00:00:53.840 semanas vasculhamos o espaço profundo e
- 00:00:56.960 satélites, mas não achamos nada
- 00:01:00.800 Tony, você lutou com ele quem te disse isso?
- 00:01:03.280 eu não lutei com ele não ele acabou comigo
- 00:01:05.680 enquanto o mágico da Rua Bleecker
- 00:01:07.200 dava tudo de bandeja foi isso
- 00:01:09.200 não houve lita, ele não é derrotável
- 00:01:11.360 ele te deu alguma pista,
- 00:01:13.040 coordenadas, alguma coisa?

- **00:01:17.040 Eu previ isso há alguns anos eu tive**
- **00:01:19.119 uma visão não quis acreditar**
- **00:01:21.040 achei que era um sonho** Preciso que
- 00:01:22.880 se concentre **e eu precisei de você**
- **00:01:25.759 tipo no passado isso está acima do que você**
- **00:01:28.720 precisa tarde demais amigo**
- **00:01:30.560 desculpe**
- **00:01:32.240 sabe o que eu preciso**
- **00:01:33.520 fazer a barba**
- **00:01:35.840 e acho que me lembro de dizer a todos**
- 00:01:38.400 Tony Tony Tony… **sobreviventes ou não**
- **00:01:41.119 que precisávamos**
- **00:01:42.880 por uma armadura ao redor do mundo**
- **00:01:44.880 Lembram-se mesmo que prejudicasse a nossa**
- **00:01:46.720 preciosa liberdade era isso que**
- **00:01:49.520 precisávamos,** mas não adiantou, não é?
- **00:01:51.280 eu disse que perderíamos**
- **00:01:52.880 você disse faremos isso juntos também**
- **00:01:55.600 e adivinhe só cap**
- **00:01:57.040 nós perdemos**
- **00:01:58.719 e você não estava lá**
- **00:02:01.200 mas é isso que fazemos não é**
- **00:02:03.040 atuamos depois dos fatos somos os**
- **00:02:04.560 vingadores somos os vingadores não os**
- **00:02:06.960 preventores** ok você expos
- 00:02:08.878 seu argumento sente-se okay okay não não
- 00:02:10.479 aqui está meu porém só
- 00:02:12.000 sente-se nós precisamos de você é sangue novo
- 00:02:14.560 bando de velhos não tenho nada
- 00:02:16.959 para você cap nenhuma coordenada nenhuma
- 00:02:19.599 pista ou estratégia nenhuma opção zero nada
- 00:02:22.800 nenhuma
- 00:02:24.080 confiança mentiroso
- 00:02:30.160 aqui leve isso
- 00:02:31.680 se encontrá-lo, ponha isso
- 00:02:34.160 esconda-se
- 00:02:37.040 estou bem
- 00:02:38.879 me deixem..

(Transcrição Em ingles)

- 00:00:00.080 It's been 23 days since thanos came to
- 00:00:02.080 earth
- 00:00:04.720 world governments are
- 00:00:06.319 in pieces um the parts that are still
- 00:00:08.639 working
- 00:00:09.679 they are trying to take a census and it looks
- 00:00:11.840 like he
- 00:00:14.920 he did exactly what he said he was
- 00:00:17.199 going to do thanos wiped out
- 00:00:21.359 50%
- 00:00:22.720 of all living creatures
- 00:00:28.560 Where is he now? where?
- 00:00:30.800 we don't know
- 00:00:32.640 he just opened the portal and walked
- 00:00:34.640 through
- 00:00:38.160 what's wrong with him
- 00:00:40.160 uh he's pissed
- 00:00:42.640 he thinks he failed
- 00:00:45.200 which of course he did but you know
- 00:00:46.719 there's a lot of that going around ain't
- 00:00:48.079 there honestly until this exact second I
- 00:00:50.559 thought you were build-a-bear maybe i am
- 00:00:52.320 we've been hunting thanos for three
- 00:00:53.840 weeks now deep space scans and
- 00:00:56.960 satellites and we got nothing
- 00:01:00.800 Tony you fought him who told you that?
- 00:01:03.280 There was no fight no he went my face with
- 00:01:05.680 a planet while the Bleecker street
- 00:01:07.200 magician gave away the store that's what
- 00:01:09.200 happened there's no fights okay he said
- 00:01:11.360 did he give you any clues any
- 00:01:13.040 coordinates anything uh
- **00:01:17.040 I saw this coming a few years back I had a**
- **00:01:19.119 vision I didn't want to believe it**
- **00:01:21.040 maybe I was dreaming** Tony I'm gonna
- 00:01:22.880 need you to focus **and I needed you**
- **00:01:25.759 as in past tense that trumps what you**
- **00:01:28.720 need it's too late buddy**
- **00:01:30.560 sorry**

- **00:01:32.240 you know what I need**
- **00:01:33.520 i didn't shave**
- **00:01:35.840 and I believe i remember telling oh yes**
- 00:01:38.400 Tony Tony Tony… **alive or otherwise that**
- **00:01:41.119 what we needed**
- **00:01:42.880 was a suit armor around the world**
- **00:01:44.880 remember that? whether it impacted our**
- **00:01:46.720 precious freedoms or not that's what we**
- **00:01:49.520 needed** well that didn't work out did it
- **00:01:51.280 I said we'd lose**
- **00:01:52.880 you said we'll do that together too**
- **00:01:55.600 and guess what cap**
- **00:01:57.040 we lost**
- **00:01:58.719 and you weren't there**
- **00:02:01.200 but that's what we do right our best**
- **00:02:03.040 work after the facts what are the**
- **00:02:04.560 avengers we are the avengers not the**
- **00:02:06.960 pre-avengers** okay right you made your
- 00:02:08.878 point just sit down okay okay no no
- 00:02:10.479 here's my but you don't want me to just
- 00:02:12.000 sit down we need you your new blood
- 00:02:14.560 bunch of tired old meals i got nothing
- 00:02:16.959 for you cap i got no coordinates no
- 00:02:19.599 clues no strategies no options zero zip
- 00:02:22.800 not a
- 00:02:24.080 no trust fire
- 00:02:30.160 here take this
- 00:02:31.680 you find him you put that on
- 00:02:34.160 you hide
- 00:02:37.040 I'm fine
- 00:02:38.879 let me…

Agora vamos discutir esse momento icônico do UCM, em que Tony está basicamente dizendo: EU AVISEI VOCÊS, FILHOS DA PUTA! E isso reflete 99,99% das equipes de segurança cibernética hoje em dia, nós alertamos sobre alguma ameaça, risco etc. e ninguém se importa, então acontece uma merda e eles vêm até nós gritando em pânico, e temos que consertar...., mas eu gostaria de poder simplesmente dizer: EU DISSE A VOCÊS.

Infelizmente, essa é a realidade, o trabalho de uma equipe de segurança cibernética em geral é alertar sobre as ameaças, que podem ser um risco em algum momento, que podem explorar uma vulnerabilidade e se tornar um incidente, sim, são possibilidades que podem nunca acontecer, e para ser mais preciso sobre essa possibilidade, muitas ferramentas trazem bons recursos, juntamente com uma boa análise de impacto nos negócios, você pode basicamente concentrar seus esforços em corrigir o que importa, em vez de tentar lidar com centenas ou milhares de vulnerabilidades ao redor.

Nessa cena, Tony está fazendo uma referência a Ultron, que inicialmente era uma ideia de "terno ao redor do mundo" para protegê-lo, como sabemos, não funcionou corretamente e houve uma massa, mas depois meio que funcionou, com a criação do Visão no mesmo filme. Mas todos nós sabemos que Tony começou a dizer isso logo após Os Vingadores e a invasão de Nova York. Devido à luta por causa de Ultron, que o Capitão América não queria junto com outros, é isso que Tony está mencionando agora, porque agora Thanos veio e destruiu tudo, e Tony sabe claramente que, com mais tempo e testes, Ultron seria a arma/escudo perfeito contra Thanos, a maior ameaça até agora. Também vemos na série What If que Ultron mata Thanos em um segundo!!!

Outro ponto muito importante nesse momento do Tony é que ele menciona "nós precisávamos... Era uma armadura de fato em todo o mundo, lembra-se disso? Independentemente do impacto em nossas preciosas liberdades ou não, era disso que precisávamos!" Esse é um momento de reflexão, pois assim que surgiu a ideia de Ultron, a preocupação dos outros era exatamente a "liberdade", que Ultron estaria monitorando tudo o tempo todo, mais ou menos.

Aqui, o paralelo com a AppSec é muito claro: hoje em dia, os desenvolvedores (e outros) têm todo o poder de dirigir uma empresa e suas decisões de alguma forma. Se eles não quiserem executar varreduras de código, a empresa aceitará isso facilmente. Se o processo de CI/CD estiver lento devido a um teste de segurança, eles o removerão. Se o plug-in do IDE que verifica a qualidade do código não for "sofisticado o suficiente", eles gritarão e o removerão. Se for necessário executar um pentest, eles perguntam por quê? Não somos alvos. Às vezes, quando conseguimos executar uma varredura SCA ou SAST e apresentamos a eles algumas vulnerabilidades a serem corrigidas, a primeira reação é: é um falso positivo ou pior, sempre foi assim, por que precisamos corrigi-lo agora? A questão é que tudo é prioridade, menos a segurança, e em algum ponto eu concordo com isso, assumir riscos faz parte da vida humana desde a criação, mas isso não significa ser negligente, não significa que o que importa é apenas o dinheiro e o funcionamento dos negócios (sim, mas não apenas).

E o que eu mais gosto no discurso do Tony é quando ele diz: " **mas é isso que fazemos direito nosso melhor trabalho depois dos fatos**, nós somos os **vingadores** nós somos os vingadores não os **preventores**", esse deveria ser o slogan do Cyber Sec, *nós*

agimos depois dos fatos, mais especificamente o slogan do AppSec, já que todo o trabalho (ou a maior parte dele) é prevenir e proteger, acabamos sempre limpando a merda depois que algo aconteceu. E o mais engraçado é que, quando algo acontece, toda a empresa chama os Vingadores (equipe de segurança), e às vezes funciona, às vezes o incidente não é tão crítico que a equipe de segurança possa lidar, mas muitas vezes não é, muitas vezes a empresa está no noticiário mundial por causa de um incidente de segurança que impactou não apenas a empresa, mas a própria sociedade. Enquanto eu estava escrevendo este livro, houve o CloudStrike + microsoft incident, que nem sequer foi um incidente de segurança, apenas o velho e bom bug, que causou uma interrupção no agente CloudStrike para hosts Windows. Apenas como um exemplo do impacto na sociedade, os voos pararam / atrasaram, os sistemas de pagamento, as lojas etc., tudo o que estava sendo executado no Windows usando o agente CloudStrike, parou 🙂

Olhe, eu fui desenvolvedor de software por mais de uma década. Criei códigos para empresas financeiras, de turismo, de varejo, de comércio eletrônico, para agências de marketing, de logística e muitas outras. Sei como é o processo de SDL e como as empresas realmente se concentram na entrega e na manutenção do funcionamento, sim, tudo bem. Mas se um maldito problema de segurança, apenas um, for explorado, a empresa pode não estar funcionando, caramba. Portanto, desde que migrei minha carreira para a AppSec, tenho tentado fazer com que as empresas e os indivíduos percebam essa importância, não apenas verificando e encontrando/corrigindo problemas, mas também parando de criá-los em primeiro lugar. Para executar um programa de AppSec, eu diria que é fácil (não é), comprar um SAST / SCA e começar a fazer a varredura? Trabalho de 5 minutos, para consertar? Talvez uma hora, para parar de criar esses problemas de código? Talvez toda a jornada de treinamento de um desenvolvedor seja capaz de, pelo menos, pensar antes de configurar um parâmetro sem validação.