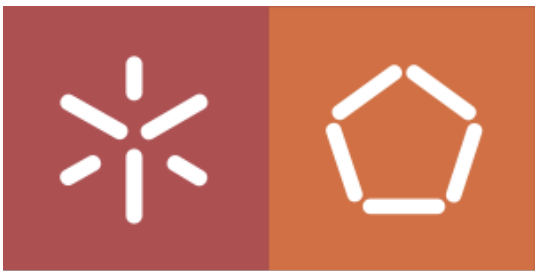

Universidade do Minho
Escola de Engenharia

Universidade do Minho
Mestrado Integrado em Engenharia Informática
Engenharia de Segurança
Projeto 1 - Software Security Takes a Champion

Diogo Duarte
(pg41843@alunos.uminho.pt)
Mateus da Silva Ferreira
(pg37159@alunos.uminho.pt)
Ricardo Cunha Dias
(pg39295@alunos.uminho.pt)

23 de Março de 2020

Conteúdo

1. Introdução

Este relatório apresenta informações sobre como construir e manter um programa *Security Champions* (SC) e baseia-se no *guidebook* "*How to Build and Sustain a Successful Security Champions Program*"elaborado pela SAFECode. Um SC é um desenvolvedor, geralmente sénior, que tem conhecimentos sobre a área de segurança e faz o papel de ligação entre a equipa de desenvolvimento e a equipa de segurança, tomando conhecimento de potenciais problemas relacionados a segurança e traduzindo-os e repassando-os para os especialistas em segurança.

Os tópicos apresentados a seguir abordam temos como: O que significa ser um SC, como identificar um possível SC, como medir se o programa SC esta a funcionar satisfatoriamente, entre outros.

2. Construindo um Software Seguro: *It Takes a Champion*

As empresas e suas equipas de desenvolvedores frequentemente enfrentam dificuldades para gerenciar seus Secure Development Lifecycle (SDL). Isso se deve, geralmente, à um ou mais dos motivos citados abaixo:

- **Limitação de pessoal:** equipas centralizadas de segurança, principalmente em empresas de médio e grande porte, geralmente operam com um número muito restrito de colaboradores e por isso eles não têm recursos suficientes para oferecer suporte à cada equipa de produto ou aplicação da empresa.
- **Conhecimento limitado:** Segurança de software ainda não é uma componente muito explorada nos cursos engenharia de software e por isso, os desenvolvedores podem não ter um conhecimento profundo sobre as necessidades de segurança tanto internas, como relacionadas ao cliente. As empresas, ao contratar desenvolvedores ou engenheiros, geralmente não levam em consideração as habilidades e histórico em segurança dos candidatos e por isso, as equipas de desenvolvimentos, para prestarem suporte à atividades de segurança, dependem que equipas de segurança forneçam orientações.
- **Diversidade dos processos de desenvolvimento:** Empresas com vários processos de desenvolvimento e linhas de produção podem enfrentar dificuldades em traduzir os requisitos do SDL para o processo de desenvolvimento de cada equipa de cada produto.
- **Percepção do "*Bad Cop*":** As equipas de segurança são, comumente, vistas como um "policial mau" que fica a espera de uma violação de segurança para advertir os desenvolvedores. Esta relação entre equipas de desenvolvimento e equipas de segurança, muitas vezes, levam à resistência por parte dos desenvolvedores em conceder um lugar ao AppSec nas discussões sobre os projetos e incentiva a cultura de exclusão dos riscos do AppSec para realizar lançamentos.
- **Desafios de Prioritização:** Os requisitos de desenvolvimento de segurança são frequentemente priorizados abaixo dos requisitos funcionais de desenvolvimento devido à perspectiva orientada ao valor definido pelo cliente. Como existe uma falta de cálculo de risco-valor estabelecido para atividades de desenvolvimento de segurança, a priorização é, geralmente, complicada ou até mesmo ausente, e muitas vezes a pessoa mais capacitada da equipa é escolhida para lidar com esse cálculo, mesmo que ela não tenha habilidades para isso.

Para construir segurança dentro do seu DNA, uma empresa deve criar uma cultura de segurança de cima a baixo e para gerar esta cultura, as empresas de desenvolvimento empregam uma mistura ad hoc de atividades, como:

- **Exercícios CTF (*Capture-the-flag*):** Exercícios no estilo videojogo baseados no tempo, voltados para ajudar os membros da equipa de desenvolvimento a aprender conceitos de segurança de uma maneira "vamos hackear aplicativos simulados". Acredita-se é que os participantes começarão a aplicar automaticamente o conhecimento adquirido durante esses exercícios em suas funções diárias, porém essa esperança muitas vezes é perdida quando esses exercícios são realizados de maneira inadequada.

- **Encontrar os "*firefighters*":** Membros das equipas de desenvolvimento, geralmente desenvolvedores séniores, que se voluntariam para ajudar em uma ou mais atividades de segurança devido à alguma necessidade urgente como auditorias ou alguma solicitação de clientes.
- **Programas informais ou ainda não concluídos:** Frequentemente, as equipas de segurança estabelecem um entendimento informal com alguns membros da equipa de desenvolvimento acerca da execução de algumas atividades de segurança. Entretanto, por não serem formais, esse tipo de relacionamento, apesar de úteis, não têm uma visibilidade à nível gerencial e carecem de incentivos bem definidos.

Porém, em consequência da forma como as equipas de segurança muitas vezes são vistas pelas equipas de desenvolvimento e devido, também, a falta de suporte e planejamento a nível de programa, estes tipos de esforços ad hoc geralmente enfrentam muita dificuldade e resistências na sua implementação e até mesmo falham completamente.

Para fazer a ligação entre as equipas e conduzir de modo coerente as atividades de segurança na empresa, falta nestas abordagens, uma "camada de cola". Esta camada é onde se encaixam os *Security Champions* (SC), que são membros da equipa de desenvolvimento, identificados e habilitados para ajudar a guiar e executar as atividades de segurança no dia a dia da empresa. Diferentemente de equipas de especialistas em segurança com disponibilidade limitada, os Scs estão sempre disponíveis para dar suporte à execução do SDL e atividades de segurança juntamente a equipa de desenvolvimento.

3. Identificando um *Software Security Champion*

Após apresentar o porquê da necessidade de se implementar um programa *Security Champion*, será expostos, neste capítulo, as qualificações e funções deste colaborador e demonstrar que segurança não precisa ser um obstáculo, como é visto por grande parte das equipas de desenvolvimento.

3.1 Quem pode ser um SC?

Geralmente, um SC além de ser um desenvolvedor ativo, é também um membro com um vasto conhecimento sobre a comunidade de segurança de software. Eles estão inseridos nas equipas de desenvolvimento de software para acelerar o desenvolvimento, identificando e resolvendo problemas de segurança antecipadamente no ciclo de vida do desenvolvimento. Os SCs garantem que a segurança seja algo presente ao longo de todo o ciclo de desenvolvimento e são autorizados a encaminhar possíveis problemas para as equipas de gerenciamento e de segurança.

Em resumo, os SCs são facilitadores dentro das equipas de desenvolvimento, que impedem que a segurança se torne um componente bloqueador para o lançamento do produto, além de capacitarem as equipas para que juntos, possam produzir, continuamente, segurança para os seus clientes.

3.2 O que caracteriza um SC?

Os SCs enfrentam o desafio de garantir que a segurança seja integrada ao procjeto desde o início. Eles trabalham ajudando os desenvolvedores a produzir código seguro desde a primeira linha, assegurando assim, que a segurança esteja presente em todo do produto. Além disto, os SCs também precisam certificar que a segurança não esteja apenas no código de sua equipa, mas também, nos códigos de fornecedores.

Sendo assim, é necessário que o papel de um SC seja encarado como uma responsabilidade primária e preferencialmente, deve ser um trabalho de período integral e portanto, não deve ser um *Scrum Master*, um PO (*Product Owner*) ou uma pessoa que já tenha várias funções e não tenha tempo para lidar com as questões relacionadas com a segurança.

Como citado anteriormente, um SCs deve atuar como uma ligação entre equipas de desenvolvimento e a equipa de segurança, fazendo o reconhecimento dos problemas das equipas de desenvolvimento e os traduzindo para a equipa de segurança. O SC pode, também, ser responsável por fazer revisões de segurança e design nos códigos, juntamente com as equipas, além de delegar funções como nomear membros para executar as ferramentas de teste e garantir que estes saibam interpretar os resultados.

3.3 Deveres de um SC

É importante observar que a definição detalhada dos deveres de um SC pode variar entre organizações de diferentes portes, domínios e maturidade do processo de segurança, porém as principais funções de um SC podem ser caracterizadas em três áreas:

1. ***Security Awareness***: Apoiar diretamente os membros da equipa em atividades de segurança e promover o conhecimento e a conscientização de segurança dentro da equipa de desenvolvimento, além de estabelecer a comunicação sobre o status de segurança entre as equipas.
2. ***Enable Security***: Garantir que os requisitos de segurança sejam prioritizados adequadamente e não sejam tratados com menos urgência que os requisitos funcionais, além de assegurar que a equipa de desenvolvimento possua as ferramentas necessárias para garantir que a segurança seja integrada e injectada em todo o processo, desde o começo.
3. ***Risk-oriented***: Garantir uma abordagem orientada ao risco (*Risk appetite*), o que inclui conhecer os riscos que a empresa está disposta a assumir e verificar se esta abordagem está integrada em todas as etapas do processo.

Vários cargos no contexto de engenharia de software podem se tornar SCs ou assistentes de SCs e contribuir para as medidas de segurança, como:

- Engenheiros que têm conhecimento sobre algum mecanismo de segurança e podem verificar se as melhores práticas deste mecanismo são aplicadas.
- Desenvolvedores que conhecem as aplicações implementadas nos mínimos detalhes e podem detectar problemas de segurança no código (princípio dos 4 olhos ou programação em pares).
- Designers ou arquitetos de soluções que entendem como as partes do sistema interagem e onde ocorrem problemas de segurança na composição.
- Gerentes de programa que esclarece à equipa de desenvolvimento o valor que a segurança cria para seu produto e por que ela fornece confiança aos seus clientes, entre outros.

Os SCs devem ter conhecimento das principais áreas relevantes para a realização de tarefas de segurança, pelo menos as seguintes competências devem existir ou ser obtidas com algum tipo de treinamento:

1. **Engenharia de Software:** Ferramentas e métodos de desenvolvimento de software seguro.
2. **Análises de ameaças e riscos:** Identificação e mitigação de ameaças, análise do caminho do ataque e dos riscos (por exemplo, *Threat Modeling*).
3. **Monitoramento de vulnerabilidades e manipulação de incidentes:** Listas de vulnerabilidades e de classificação (por exemplo, CVE e CVSS) e suporte à SIRT e PSIRT.
4. **Habilidades interpessoais:** Resolução de conflitos, habilidades de discussão e apresentação, além de saber motivar os membros.

Em alguns contextos, certificações profissionais também podem ser essenciais para os SCs, dependendo das necessidades da empresa.
Em resumo, um SC deve possuir conhecimentos tanto de engenharia de segurança quanto de engenharia de software, além de ser capaz de dialogar com especialistas em ambos os campos e ser um catalisador para unificá-los.

4. Como Construir um programa eficiente de *Security Champions*

O primeiro passo lógico seria contratar ou adicionar um SC (*Security Champion*) para o projecto.
Porém nem sempre é possível e os requisitos necessários para estabelecer as fundações iniciais são os seguintes.

4.1 Fundações

- **Compreender a cultura de segurança existente**: Significa que em vez, de saltar entre equipas de desenvolvimento, é necessário conhecer o estado actual da cultura de segurança dentro da organização, antes de se começar a adicionar sistematicamente elementos de segurança ao projecto.

- **Convencer Equipas de Gestão/Executivo**: Deve-se procurar ter a equipa de executivo do lado do SC, isto é alcançável, através da amostragem de dados de como o programa pode ajudar a empresa a conter custos, minimizar riscos e acrescentar valor, procurar mostrar o valor que se espera obter a longo prazo enquanto se implementa os blocos de fundação do programa.

- **Convencer Equipa de Desenvolvimento**: A equipa tem de estar a par de que não é o objectivo do SC de patrulhar o seu trabalho, mas sim de resolver futuros problemas desde o início, de uma forma prática e orientada ao risco, é essencial ter a equipa do lado do SC.

- **Políticas e Regras**: Deve-se usar nos documentos palavras e conceitos comuns entre os desenvolvedores, devem também ser utilizados conceitos como *Security by Design, Security by default*, isto permite criar uma ligação maior com a equipa de desenvolvimento.

- **Financiamento**: É também importante ter apoio financeiro, para ajudar a gerar recompensas e reconhecimento dentro da equipa, é aconselhado o investimento em *training*/*workshops*/conferências ou actividades como CTF(*capture the flag*), para manter as equipas unidas, aumentar o espírito de equipa e impulsionar o empenho destas.

4.2 Definição e Operacionalização do Programa

Esta é a fase é onde se define as principais as componentes do programa.
Qual é a estrutura da organização?

- O programa vai ser centralizado ou descentralizado?, baseando-se no tamanho e no *layout* da equipa de desenvolvimento da empresa?
- Quais os principais problemas que o SC vai ter de batalhar nas diferentes áreas?
- Quantas equipas é suposto o SC dar suporte.
- A Distribuição Geográfica das equipas.
- A Tolerância ao risco ao longo da organização.

Quais são os papéis e responsabilidades?

- A persona do SC.
- Definir claramente quais são os papéis e responsabilidades do SC, para não só definir as *skills* necessárias mas para também definir as expectativas esperadas.
- A influência do SC com base na função que ocupa na organização.
- Algumas possíveis funções adicionais para o SC são Engenheiro de qualidade ou Gestor de projetos(*Project Manager*), entre outras.
- Ter uma estrutura definida, sobre a quem deve ser reportado os relatórios, como por exemplo, equipa de desenvolvimento, gestão ou segurança.

Onde encontrar um SC?
Estes foram alguns dos métodos utilizado com sucesso por membros da SAFECode.

- Identificar Engenheiros que demonstrem vontade para aprender sobre Segurança de software e tenham demonstrado *skills* de segurança.
- Vender o lugar como uma oportunidade de desenvolvimento de emprego-
- Vender o lugar como uma posição voluntária, apenas para aqueles que estejam interessados.
- Contratar um *Expert* na área de Segurança, para cumprir esta função.

Operational Playbook?
Uma espécie de plano de carreira, isto pq o SC necessita de estar em constante aprendizagem, e é importante definir um plano de crescimento pessoal e de carreira, isto significa que o SC necessita de tempo para expandir o seu leque de conhecimento, como tal é necessário o SC estar presente nas seguintes actividades:

- **CTF(*Capture the flag*)**: Mantêm os seus conhecimentos relevantes e envolventes, permite também maior ligação com os membros da equipas.
- **Ferramentas**: Permite automatizar alguns processos de segurança, poupando tempo precioso.
- **Train the trainer**: Treinar o SC no "O quê?"and "Como?"
  "O quê"passa por estudar, desde princípios de design segurança, a testes de segurança.
  "Como"pode incluir desde Interno ou Externo CBT(*computer based training*)
  Um sistema buddy dentro da equipa.
  Workshops em temas como Modelação de ameaças e analise de código.
  Treino de *awareness* para que estejam a par das ultimos problemas de segurança.
- **Conferências**: permite *networking* com outras empresas e partilhar informações.
- **Fóruns de segurança**: semelhante a conferências mas é mais dedicado a aprendizagem pois permite explorar novos vectores de ameaças, ou apresentar novos dilemas à comunidade.

CBT (*Computer based training*), consiste no uso de um computador para sessões de treino, aulas, via online, que podem ser internas feitas por membros da organização, ou externas por um membro não pertence à organização. Podemos concluir que apenas encontrar um SC e pedir para corrigir a segurança, não vai funcionar, para este ter sucesso, é necessário todo um suporte formal, e apoio das diferentes equipas da organização, com objectivos bem definidos.

5. 6 Sinais que o programa de *Security Champions* está com problemas

Programas do tipo *Security Champion* tal como qualquer outro, podem encontrar *bottlenecks*, seja na sua fase de design ou de implementação, e como tal SafeCoder's, já tiverem de passar por este tipo de problemas, e como tal decidiram deixar uma série situações a ter em conta na implementação deste tipo de programa.

1. **Diminuição de recursos**: quando os recursos começam a diminuir, significa que o programa foi desprioritizado pelo sistema, a solução passa por voltar ao início e reavaliar as fundações.
2. **Dependência excessiva em Computer-based training (CBT)**: Com tanta informação disponível é normal que as pessoas fiquem presas num ciclo de partilha de recursos online como fonte de estudo, porém uma abordagem de treino baseada apenas em CBT não é suficiente, esta deve ser combinada com treino com instrutor e mentor, e o conteúdo oferecido deve ser mais especifico e não abrangente, tal como os *workshops* ou cursos devem ser focados em vertentes específicas, esta mistura de diferentes conhecimentos irá permitir ter um programa mais actual e com um suporte maior para diferentes áreas de segurança.
3. **Falta de comunicação**: Se o SC não estiver completamente empenhado, o resto da equipa também não é de esperar que esteja.
4. **Ver segurança como uma característica e não como um processo**: Quando isto acontece é normal que a equipa acabe por desprioritizar a segurança, por isso a segurança deve ser sempre vista como um processo com o objectivo de melhorar a qualidade geral do produto.
5. **Perceção de "*Bad cop*" (Policial Mau)**: Guiar o processo de segurança através de medo e incerteza, faz com que as equipas vejam como algo negativo, por isso é importante ser parte integral das equipas.
6. **Falta de responsabilidade**: Se a equipa achar que a não é possível atingir determinado nível de segurança, estas dificuldades devem ser discutidas em aberto entra equipa em oposto a serem remediadas.

6. Iniciação e acompanhamento do programa *Security Champions*

Neste capítulo abordaremos a forma mais correta de implementar um programa *Security Champions*. É fundamental um bom começo para o início do programa e para isso deve-se considerar usar um tipo de estrutura de gestão de mudanças para ajudar a orientar a implementação, como por exemplo, os oito passos de Kotter para a mudança.
De seguida, recomenda-se iniciar o SC com uma fase de piloto/teste, ou seja, deve-se implementar o programa com uma equipa de pessoas dentro da organização (de preferência com um número pequeno de participantes) sem afetar o resto da companhia. Assim, esta fase de teste serveria para estabelecer se o programa correspondia aos objetivos pretendidos e, caso se verificasse, apresentá-lo à gerência da empresa. Para além da ferramenta de gestão é essencial que todos os elementos da equipa presente no programa estejam abertos, interessados e pretendam obter ajuda para integrar a segurança nos seus processos. Após a implementação do programa SC num empresa é necessário tomar algumas medidas para que seja possível sustentá-lo. O programa deve ser adoptado pela equipa principal de segurança para que tenha influência e permita uma alteração na cultura de segurança de uma empresa. Para que estas modificações sejam possíveis é fundamental construir um ecossistema de apoio ao programa SC. De seguida, serão apresentadas algumas sugestões de táticas que permitem obter este objetivo:

- **Criar um SC Cohort:** utilizar uma ferramenta de comunicação para que todos os SC designados partilhem anúncios/informações importantes sobre o programa SC. Paralelamente, deve-se criar e atualizar um *AppSec Wiki Central* para ajudar na partilha de informação. Por fim, recomenda-se o agendamento de reuniões para reconhecer os esforços e enfatizar o papel dos SCs e destacar como estes se encaixam no AppSec.

- **Socializar a segurança de software em toda a equipa de desenvolvimento:** divulgar e enaltecer o conceito de segurança de aplicações em todos os níveis de uma equipa de desenvolvimento. Este compartilhamento de informação pode ser efetuado através do agendamento de sessões com *AppSec Subject Matter Experts* para discutir conceitos de segurança.

- **Tornar a segurança divertida e não uma atividade descabida:** fazer com que o SC designado lidere a organização periódica através da "gamificação"na forma de hackathons.

- **Mentalidade *train-the-trainer*:** o papel do SC deve evoluir gradualmente de guias de segurança para atuar como "treinadores que treinam"desenvolvedores para que possam resolver as coisas autonomamente, procurando esclarecimentos adicionais sempre que necessário.

Resumindo, a utilização de processos estabelecidos de gestão de mudanças e a inicialização de um programa piloto para identificar e corrigir problemas antecipadamente ajudarão um programa SC a começar de maneira positiva. Para sustentar este começo, os programas SC necessitam de medidas para implementar e enfatizar conceitos sobre segurança, de forma a criar uma cultura de suporte.

7. Definição de Sucesso: O programa *Security Champion* está a funcionar?

Para rastrear o nível de eficácia do programa SC devem-se utilizar uma combinação de métricas quantitativas com qualitativas. De forma a ajudar a guiar este processo, serão apresentadas uma lista de métricas quantitativas e qualitativas usualmente utilizadas.

7.1 Medidas quantitativas:

- **Percentagem de membros da equipa que adquiriram conceitos/treinos básicos e/ou específicos sobre código orientado a SC:** verificar quantos elementos ao longo do tempo vão recebendo estes conceitos.
- **Percentagem de membros séniores da equipa que adquiriram conceitos/treinos:** verificar se o número de elementos séniores que adquirem conhecimentos aumenta ao longo do tempo.
- **Número reduzido de perguntas de entrada para os SC's:** verificar se o número de perguntas de entrada reduz ao longo do tempo.
- **Diminuição do número de vulnerabilidades recorrentes:** verificar se o número de vulnerabilidades reduz ao longo do tempo.
- **Conclusão dos módulos de treino por parte dos elementos da equipa:** verificar quantos e quais os módulos concluídos pela equipa de desenvolvedores.

7.2 Medidas qualitativas:

- **Feedback:** à medida que os desenvolvedores vão adquirindo mais conhecimentos sobre o AppSec deve-se verificar se estes ainda resistem à utilização dos conceitos de segurança.
- **Pesquisas curtas periódicas:** estas pesquisas permitem que as métricas sejam saudáveis e que não dependam excessivamente apenas dos aspectos quantitativos.

O uso combinado destas métricas podem ajudar a obter informações que permitem orientar as decisões sobre a manutenção do programa, bem como o ajuste dos elementos específicos presentes. Resumindo, uma abordagem de métricas eficaz pode ajudar a determinar se o programa SC é bem sucedido e, caso não seja, permite fornecer dicas de como alcançar o sucesso.

8. Conclusão

Ao longo da realização deste projeto, fomos adquirindo e consolidando conhecimentos relativos à importância da realização de software seguro. Consideramos que esta implementação é de extrema relevância para o bom funcionamento do software e das aplicações desenvolvidas, garantindo uma base segura para que todos os funcionários possam contribuir, ajudando uma empresa a atingir os seus objetivos
Assim, a utilização de *Security Champions*, por parte das empresas, permite responder a questões, recomendar conceitos/treinos e interagir com especialistas em segurança possibilitando um investimento na produtividade, otimização e na qualidade do software.