Universidade do Minho
Escola de Engenharia

Managing Security Risks Inherent in the Use of Third-party Components

Investigação do tópico de desenvolvimento seguro de *software*

Submetido por:

Diogo Araújo A78485

Diogo Nogueira A78957

Conteúdo

Contextualização

O presente trabalho surge no sentido de investigar um tópico de desenvolvimento seguro de *software* de seu título "*Managing Security Risks Inherent in the Use of Third-party Components*, que como o próprio nome o indica refere-se ao estudo dos riscos de segurança que estão inerentes no uso de componentes de terceiros para o desenvolvimento final de um determinado produto.

Eis os objetivos planeados para esta investigação:

- Compreender o significado de *Third-party Components* e aquilo que representam no produto final
- Quais os riscos associados ao seu uso e possíveis resoluções
- Como é possível fazer a gestão destes componentes de modo a contornar os ricos de segurança

Para se alcançarem estes objetivos existirá uma fase inicial que se focará em entender todo o artigo usado como base para esta investigação, através de um pequeno resumo relativamente ao que aborda e ainda informações extra que possam ser necessárias à sua compreensão. Com esta base teórica perfeitamente compreendida surge uma fase de pesquisa extra, que procura investigar outras fontes de informação para complementar aquilo que ficou entendido.

A metodologia utilizada para esta investigação foi o uso do documento de apoio fornecido pelo docente e alguma pesquisa bibliográfica enriquecida com fontes *online* que abordam este assunto dos *Third-party Components*.

Third-party Components

De modo a introduzir o tema que irá ser abordado no decorrer desta investigação é importante estabelecer alguns conceitos e ideias que permitirão compreender melhor o estudo em causa. O conceito mais importante é o de *Third-party Components*, não só pelo que representam estes componentes, mas também pela sua importância num produto final que é entregue a um consumidor.

O que são?

"*Third Party Components means, with respect to a product, all software that is embedded in, used in, incorporated into, combined with, linked with, distributed with, provided to any Person as a service with, provided via a network as a service or application with or made available with such product, in each case that is owned, in whole or in part, by a third party.*" [1]

Com isto se entende que um *Third-Party Component* consiste essencialmente num componente de terceiros produzido com o objetivo de ser reutilizável para que possa ser distribuído de forma livre ou até mesmo vendido posteriormente por uma entidade que não o fornecedor original do mesmo.

Qual o seu significado no produto final?

O uso de componentes de terceiros tem sido visto como uma forma eficaz de diminuir os custos de um produto ao mesmo tempo que reduz o tempo do ciclo do desenvolvimento de *software*.

[1] Fonte: https://www.lawinsider.com/contracts/8NVe5f8UNc0#third-party-components

Por isso se diz que estes componentes têm um impacto maior no início de um projeto, dado que ao fazer-se a sua integração podemos avançar de forma mais rápida e, tal como já dito, economizar alguns custos a curto prazo.

Claro está que esta integração poderá causar impacto a longo prazo, tanto através de vantagens como desvantagens:

- A maior vantagem/benefício a longo prazo é estes componentes fornecem funcionalidades que são constantemente atualizadas e com os *standards* mais recentes.
    - Isto é útil para quem está focado em garantir que o produto final se mantenha seguro e perfeitamente funcional.
- A maior desvantagem a longo prazo é que ao usar-se estes componentes estamos também a abdicar do controle total sobre os mesmos.
    - Isso significa que se um determinado componente for descontinuado será necessário investir numa implementação alternativa.
    - Caso exista uma atualização que provoque problemas no produto final será necessário resolver esses problemas e acarretar com os custos associados a esse processo.
    - Se porventura o componente for personalizado muito após a sua integração no produto, pode ser praticamente impossível e muito caro instalar atualizações.

Isto deixa claro que há muitos fatores a ter em consideração quando se quer decidir se um *Third-party Component* é adequado ou não para um projeto de *software* personalizado e que o significado destes componentes nesse produto final vai depender de muitas decisões/implementações que se tomem ao longo de todo o processo de desenvolvimento.

A investigação serve essencialmente para compreender todas estas vertentes e de certa forma clarificar as ideias que são realmente precisas quando se faz uso destes componentes de terceiros.

Adversidades inerentes ao uso *de Third-party Components*

O caso de estudo em análise apresenta-se como sendo uma tentativa em perceber qual a influência do uso destes componentes na produção de determinados produtos finais ao consumidor. O caso apresenta-se como sendo o caso do Bob, e neste ponto em específico procura-se responder a uma série de questões que surgem logo após a venda do produto final.

Após a realização e venda do produto surgiram alguns *reports* negativos relativamente a uma vulnerabilidade grave de segurança descoberta num *open source component* pertencente ao mesmo. Esta vulnerabilidade foi listada pela própria MITRE e levantou a primeira questão - "*What third-party components are included in my product?*".

Após se compreender quais os TPC's incluídos no produto surge a segunda questão - "*Is the product affected by the CVE?*".

Constatou-se que o CVE em si listava o componente afetado com um nome que não era 100% familiar ao Bob. Com alguma revisão ao código de *software*, chegou à conclusão que correspondia ao mesmo componente que suspeitava e que o produto estava verdadeiramente afetado.

Apesar desses componentes não serem fabricados pela empresa em si, como são usados pelo Bob é como se o mesmo estivesse 100% responsável pelos seus problemas (tal como abordado no capítulo *Third-party Components*). Por essa razão foi necessário lidar com problemas financeiros, de reputação e de perda de credibilidade por parte dos consumidores. Tudo isto enquanto existiam ataques ao servidor da base de dados e à própria base de dados em si, formando-se uma nova questão - "*What should we do to maintain the TPCs within our product?*".

A partir daqui a ideia passou por listar todos os componentes conjuntamente com os riscos associados a cada um deles, na tentativa de responder a uma nova questão - "*What TPCs should we use and what is the security risk associated with them?".*

Assim, surge a questão geral de toda esta análise inicial - "*How should we manage TPCs overall?*".

Possíveis Resoluções

Estando todas as questões levantadas torna-se imperativo responder a todas elas e tentar ao mesmo tempo deduzir quais os riscos/obstáculos que estes problemas podem representar em termos de produto final.

Questão	Possível Resolução	Possíveis Obstáculos
"*What third-party components are included in my product?*"	Uso de uma *Bill of Materials* (BOM) que permite obter uma lista de componentes incluídos no produto. Esta lista facilita a localização dos produtos afetados em casos bem preparados.	1. O uso de várias linguagens de programação distintas pode complicar na interpretação de quais TPC's estão incluídos no produto; 2. Um TPC pode ter vários nomes, o que pode dificultar a identificação; 3. Um TPC pode ter vários subcomponentes (dada a sua hierarquia).
"*Is the product affected by the CVE?*"	Determinar se o TPC afetado está incluído no produto final e se o produto em si está afetado pelo CVE em causa. Não é incomum um TPC listado com CVE estar incluído no produto e o mesmo não se encontrar afetado.	1. Obstáculos similares aos anteriores; 2. Dificuldade em estabelecer uma correspondência entre os TPC's listados pelo BOM com os listados pelo CVE, de forma a determinar se os TPC's afetados estão ou não incluídos no produto; 3. O conteúdo do CVE pode não ser suficiente para determinar com 100% de certezas se o componente está afetado, podendo ser necessária uma análise mais profunda.

"*What should we do to maintain the TPCs within our product?*"	Selecionar os componentes que foram desenvolvidos já com a segurança em mente, não pensando apenas na sua funcionalidade futura.	1. Tentar estabelecer uma relação minimamente ideal entre aquilo que é realmente necessário para o funcionamento do produto e o quão seguro o mesmo deve ser.
"*What TPCs should we use and what is the security risk associated with them?"*	Manter os TPC's usados ou incorporados ao longo do tempo é uma tarefa importante que inclui responder às vulnerabilidades descobertas.	1. Nem sempre é fácil responder às vulnerabilidades que vão surgindo, dado que é necessário manter um plano organizado daquilo que o produto realmente contém.

Tabela 1 Resposta às adversidades inerentes ao uso de TPC's e possíveis obstáculos

Gestão de *Third-party Components*

O uso de *Third-party Components* traz consigo desafios que requerem um processo robusto e essencialmente bem preparado. Este processo de gestão que trata do ciclo de vida de um TPC deve começar o mais cedo possível no *Software Development Life Cycle* (SDLC) estando por isso integrado internamente no mesmo. A ideia deste processo de gestão é que as organizações definam e consequentemente adotem um modelo para controlar os riscos de segurança dos seus TPC's ao mesmo tempo que encaixam esse processo num SDLC já existente nas suas organizações.

Third-party Components Life Cycle

O que é?

O processo de gestão de um TPC consiste em etapas *high-level* que são essenciais definir para que tudo decorra com normalidade e sucesso. Dado que se trata de um ciclo, fica patente a necessidade de realizar as etapas de forma sequencial. Atente-se na imagem abaixo anexada que permite entender melhor este conceito cíclico.

Em que consiste?

Baseia-se num conjunto de etapas que definem o ciclo de vida útil de um *Third-party Component.* No capítulo Ingredientes-chave para a Gestão de um TPC irão ser discutidas as medidas a ter em conta para cada uma das etapas deste processo e de que forma amenizam a probabilidade de existência de riscos de segurança.

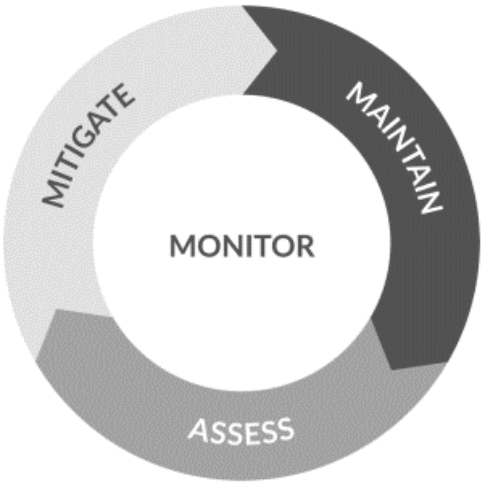

Figura 1 *Ciclo de Vida de um TPC e as suas respetivas etapas*

Etapa	Descrição da Etapa
Maintain – Manter uma lista de todos os TPC's.	Primeiro passo para a gestão de TPC's. Consiste na listagem de todos os TPC's em uso ou a ser usados futuramente.
Assess – Avaliar os riscos de segurança a partir desses TPC's.	Estando a lista de TPC's disponível, pretende-se avaliar esses componentes, de modo a medir os riscos inerentes à sua utilização. O resultado desta etapa pode resultar num *score* de risco para o TPC.
Mitigate – Mitigar ou aceitar riscos inerentes aos TPC's vulneráveis.	Tendo-se agora acesso ao perfil de risco de um TPC, esta etapa centra-se em decidir se o uso de um determinado TPC é ou não aceitável e se precisa de ser mitigado.
Monitor – Acompanhamento contínuo do TPC.	Estando todas as etapas anteriores concluídas, a ideia passa por estabelecer um processo de acompanhamento ao longo do tempo que permita garantir que o perfil de risco de um TPC permaneça aceitável. Espera-se que esta etapa sirva também para estabelecer um cenário de risco quando o TPC se torna inaceitável.

Tabela 2 *Etapas do Ciclo de Vida de um TPC e a sua respetiva descrição*

Software Development Life Cycle

O que é?

O *Software Development Life Cycle* (SDLC) consiste numa *framework* que define todo o processo usado pelas organizações por fim a desenvolver uma aplicação desde o início até ao fim do seu ciclo de vida. (Mohino, Higuera, Higuera, & Montalvo, 2019)

A sua definição importa para compreender a sua importante relação no processo de gestão de um *Third-party Component*.

Em que consiste?

Perante algumas pesquisas, verificou-se que este ciclo de vida consiste num conjunto de etapas e algumas sub-etapas que não constam no documento de apoio fornecido. Apesar disso, para a ideia que se quer transmitir e de forma a conseguir estabelecer uma relação entre ambos os ciclos, as etapas que são mencionadas neste artigo para investigação representam a generalidade do processo e são suficientes para este entendimento.

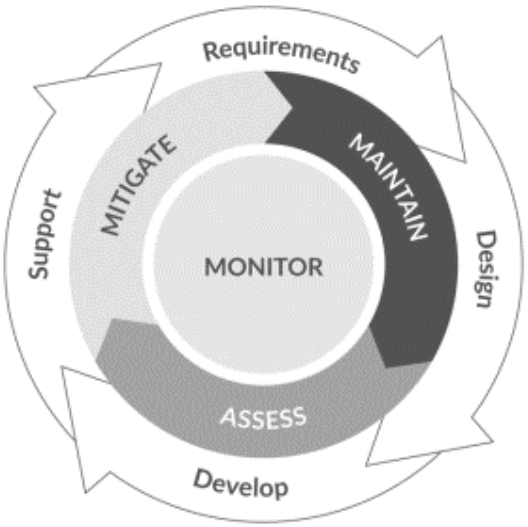

Figura 2 SDLC e as suas respetivas etapas

Etapa	Descrição da Etapa
Requirements – Apuramento das informações necessárias para o desenvolvimento do produto.	Espera-se que nesta etapa se recolham todas as informações relevantes acerca daquilo que se pretende criar, para que se consiga desenvolver um produto final consoante aquilo que é esperado.
Design – Preparação do *design* de sistema e *software*	Nesta etapa prepara-se todo o *design* de sistema e *software* de acordo com os requisitos iniciais. Isto ajuda a definir a arquitetura geral de todo o sistema.
Develop – Programar o sistema através de linguagens de programação à escolha	A ideia é começar a construir o sistema em si através do código necessário e de uma linguagem de programação adequada.
Support – Manutenção do produto final	Após ficar feita a implementação do produto deve ser feita a manutenção do mesmo, no sentido de procurar possíveis problemas que possa surgir e aperfeiçoar outros.

[2]

Tabela 3 Etapas do SDLC e a sua respetiva descrição

Relação do Ciclo de Vida de um TPC com o SDLC

Uma das formas mais eficazes de começar a gerir um TPC é fazer com que se integre num processo SDLC já estabelecido conforme a imagem abaixo o sugere.

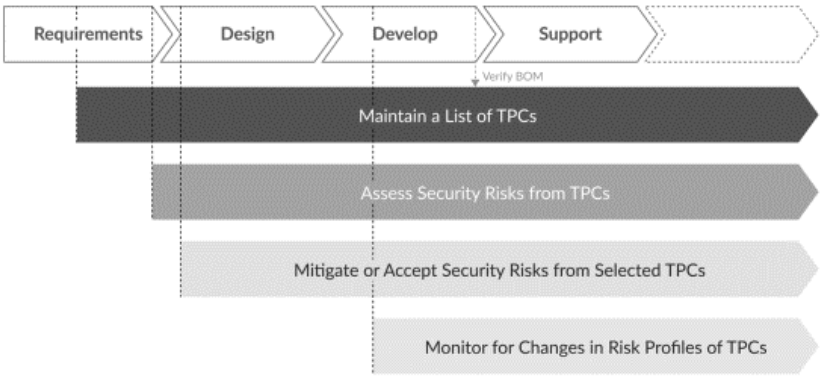

Figura 3 Relação das fases do Ciclo de Vida de um TPC e o SDLC

[2] Fontes: https://www.guru99.com/software-development-life-cycle-tutorial.html
https://www.softwaretestinghelp.com/software-development-life-cycle-sdlc

- Os TPC's devem ser listados a partir da fase de ***Requirements*** do SDLC. Com isso estabelecemos uma ligação da fase ***Maintain*** do ciclo de vida do TPC e conseguimos que os requisitos funcionais possam ditar o uso de TPC's específicos.
    - A seleção dos TPC's geralmente acontece na fase de ***Design*** e ***Develop*** do SDLC.
    - Para aplicações já antigas e sem qualquer tipo de processos de gestão do ciclo de vida dos TPC's, esta listagem de componentes acaba por acontecer na fase de ***Suport*** do SDLC.
- A etapa de ***Assess*** que consiste em avaliar os riscos de um componente, inicia-se assim que um TPC candidato é identificado através da listagem anterior.
    - Caso exista um TPC com alto risco, acaba por ser necessário mudar a lista de TPC's e começar novamente a avaliação de riscos para os TPC's recém-selecionados.
- A etapa de ***Mitigate*** acompanha o projeto desde a fase do ***Design*** do SDLC até ao restante das fases, dado que pode ser necessário salvaguardar certos níveis do código e do próprio projeto em si para que consiga mitigar os riscos de um TPC que deve ser usado.
- Dado que a fase de ***Monitor*** visa acompanhar a lista de TPC's, pode ser necessário acionar a avaliação de riscos no caso de TPC's novos ou atualizados serem adicionados à BOM.
    - Isto deve ser feito antes de se enviar o produto final ao consumidor e de entrar na fase de ***Support*** do SDLC.
    - Por isso que vimos o "*Verify BOM"* a laranja antes de entrarmos na fase de ***Support***.

Estas relações deixam claro o porquê de até mesmo as fases do processo de gestão do ciclo de vida de um TPC acabarem por coincidir. Basta pensar em tudo o que pode correr mal ao longo do processo. Apenas é preciso existir uma avaliação de risco que justifique uma mudança de componente para que seja pertinente estabelecer uma nova lista de componentes e consequentemente executar uma nova avaliação de riscos.

Tudo isto internamente ligado com o suposto ciclo de vida do desenvolvimento de um *software*, garantindo-se que o produto final está livre de riscos de segurança e perfeitamente preparado para uso.

Ingredientes-chave para a Gestão de um TPC

Como falado na secção anterior, os quatro maiores passos/fases, ou seja, *Maintain*, *Assess*, *Mitigate* e *Monitor* têm uns ingredientes-chave para serem considerados como uma excelente gestão do ciclo de vida dum TPC.

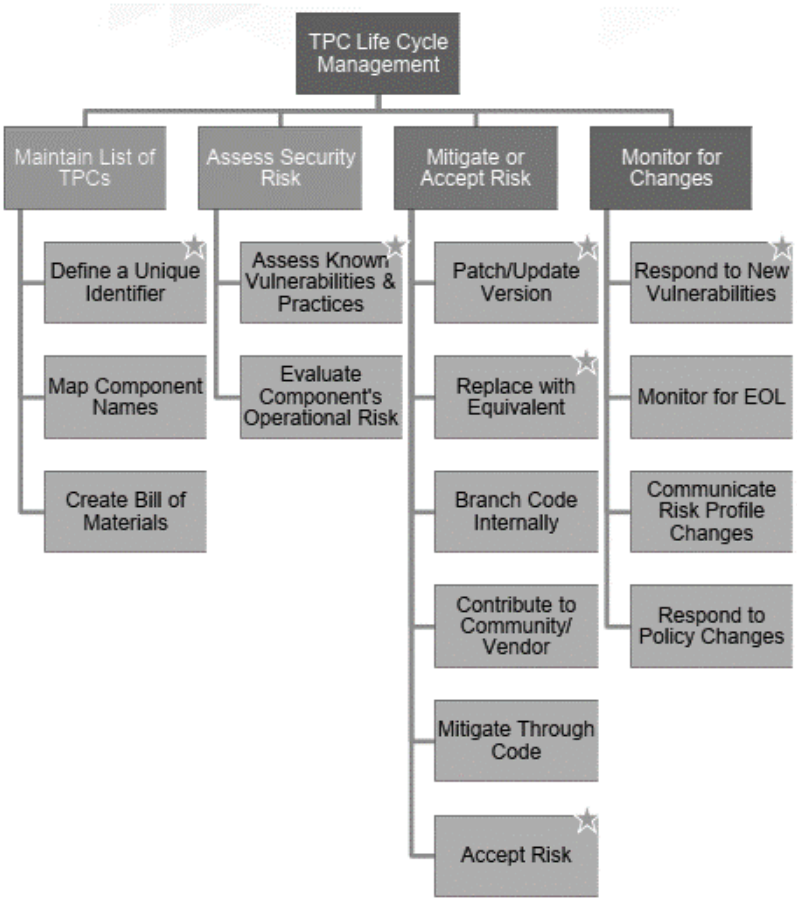

Figura 4 Etapas do processo de gestão do Ciclo de Vida de um TPC

Pela figura acima observamos uma panóplia de ingredientes para as quatro fases essenciais para a gestão do ciclo de vida dos TPC. Alguns destes ingredientes (que contêm a estrela) são considerados o mínimo para cada fase de forma a existir um controlo essencial e basilar dos TPC.

O que vamos ver a seguir é o aprofundar destes ingredientes-chave, através da descrição das várias possibilidades a ter em conta para cada uma destas fases/etapas e de certa forma compreender o seu impacto num produto final.

Maintain - Manter uma lista de TPC's

1. Definição dum identificador único de forma a ajudar na criação e manutenção do *Bill of Materials* (BOM). Para estes identificadores não terem colisões aconselha-se a utilização dum *Global Unique Identifier* (GUID) em junção com outra convenção mundialmente usada, como por exemplo *Common Platform Enumeration* (CPE) utilizado pelo MITRE, o *Maven Project Object Model* (POM) feito em XML ou ainda uma especificação da ISO/IEC para identificação única de software.

2. Mapear corretamente os TPC's aos seus identificadores corretos para colocar no BOM, associando toda a informação útil sobre cada TPC's e as possíveis vulnerabilidades encontradas, *patches*, etc.

   Assim cada TPC iria no seu *Unique Identifier* uma variada informação sobre o que era, representava e links externos para as bases de dados de segurança e vulnerabilidade.

3. Criação da *Bill of Materials* (BOM) utilizado uma abordagem o mais correta para a escala da empresa e os seus *softwares* com TPC.

Método	Prós	Contras
Lista manual dos TPC's	Gratuito	Pouca certidão, principalmente nas sub-dependências
Ferramentas automáticas	Muito preciso e eficiente	Preço e disponibilidade de múltiplas ferramentas para diferentes linguagens
Combinação das duas	Combinação dos prós falados acima	Combinação dos contras falados acima

Tabela 4 Listagem de diferentes métodos para criar e manter uma lista de TPC's

Assess – Avaliar o risco de segurança

1. Avaliar as vulnerabilidades conhecidas e as práticas usadas pensando sempre quais vulnerabilidades de segurança públicas ainda não foram remediadas na última versão (ou na versão em uso) ou se existe passos já disponíveis para mitigar as mesmas.

   Do outro lado também se deve verificar se a comunidade/vendedor tem claro relatórios de segurança sobre o(s) TPC incluído no nosso software e se existe algum *website* dedicado a receber os problemas e/ou soluções para as vulnerabilidades descobertas.

2. Avaliar o risco operacional do componente acaba por ser muito importante. As organizações que utilizam TPC têm de entender que pode ser uma manutenção de *short* ou *long term* e efetuar as devidas atualizações para ter sempre o componente *third-party* estável e atualizado.

   Assim todo o processo deve ser analisado para o TPC e entender se é algo bem suportado pela comunidade, se existe há muito tempo, se tem atualizações frequentes, qual a reputação da comunidade/vendedor do mesmo, entre outros.

Mitigate – Mitigar ou aceitar o risco

Este passo acaba por ser o de maior risco e importância porque quando são encontradas vulnerabilidades nos TPC usados no(s) nosso(s) produto(s), a equipa de *software development* deve tomar a melhor decisão da maneira duma maneira proporcional à gravidade do CVE dado.

1. Atualizar a versão do TPC pensando sempre no trabalho adicional que pode advir de tal.

2. Substituir por um TPC equivalente de forma a obter um critério de segurança melhor para integrar no produto.

3. Criar um novo *Branch* internamente e ficar o responsável por remediar a vulnerabilidade.

4. Contribuir com soluções para a comunidade/vendedor de forma a criar um ecossistema mais sustentável de soluções para mitigar o problema o mais rápido possível para toda as empresas e individuais que utilizam o TPC envolvido.

5. Mitigar através de novo código por cima de formar a atenuar o problema encontrado.

6. Aceitar o risco envolvido e não mitigar a vulnerabilidade. Isto pode acontecer quando o TPC está numa porção do produto que a equipa não está a usar ativamente ou então a vulnerabilidade tem um *score* baixo de perigo.

   Assim podem deixar a mitigação para o próximo ciclo de *update/ release*.

Monitor – Monitorizar mudanças aos TPC's

1. Responder a novas vulnerabilidades colocadas nas bases de dados de segurança edorecomo por exemplo a NIST CVE de forma a ver caso os componentes possam ter sido afetados.

   Também é importante o registo em mecanismos de notificação por parte da comunidade/vendedor dos TPC usados como por exemplo se manter a par dos *issues* do GitHub, ou a *mailing list* do vendedor.

2. Monitorizar a *End of Life* (EOL) dos TPC e verificar se todo o componente foi descontinuado ou apenas a versão usada. Para isso existe *software* que identifica corretamente estas situações e avisa para a atualização ou substituição/remoção do componente ou componentes em causa.

3. Comunicar as mudanças sobre o perfil de risco da empresa quando uma equipa centralizada ou não descobre uma vulnerabilidade na sua porção do produto, informando assim toda a empresa da situação enquanto mitiga a mesma.

4. Responder às mudanças de política de segurança sobre os TPC, dado que as vulnerabilidades hoje em dia são quase diárias e para manter a melhor postura sobre segurança nos produtos a desenvolver, a empresa deve sempre revisitar e modificar processos internos e práticas que tenham para a utilização de TPC e a sua manutenção constante.

Pesquisa Adicional

O conceito pensado para esta secção é enriquecer a investigação feita para esta área, de modo a conseguir obter novos dados ou informações que possam ser importantes estudar.

Numa fase inicial, pesquisaram-se outros documentos de apoio/artigos, tendo-se verificado que existem, à data atual, muito poucas fontes que se foquem neste tema em específico.

No entanto, seguem-se dois exemplos de artigos que abordam esta ideia de TPC conjuntamente com a área das vulnerabilidades/segurança:

Título do Artigo	Autores e Ano do Artigo	Descrição do Artigo
A New Detection Method for Stack Overflow Vulnerability Based on Component Binary Code for Third-Party Component	(2018)	*"This paper designs a stack buffer overflow vulnerability algorithm SBOD for the COM component, and implements a prototype system of detecting stack buffer overflow vulnerability."*
Shipboard ECDIS Cyber Security: Third-Party Component Threats	Boris Svilicic; Igor Rudan; Vlado Frančić; Mateo Doričić (2019)	*"The analysis os the cybers ecurity is based on the cyber security testing of the shipboard ECDIS using an industry vulnerability scanner."*

Tabela 5 Lista de artigos publicados baseados no tema dos TPC's e os seus riscos

Numa outra fase, direcionou-se a atenção a artigos *online*, tendo-se encontrado um artigo em específico e que interliga toda a ideia com um tema atualmente em alta.

Título do *Website*	Link e Ano do Artigo	Descrição do Artigo
The Hidden Risk in All IoT Devices: Third-Party Components	https://www.sternumiot.com/blog/2019/6/25/third-party-components-the-hidden-risk-in-all-iot-devices (2019)	*"Creating and designing secure IoT devices is a mission of high complexity. IoT devices often contain many functionalities that require using third-party software components within the device. In this article we will review the hidden risks in third-party components – as these components are highly critical in IoT devices, and affect device security greatly."*

Tabela 6 Lista de artigos online baseados no tema dos TPC's e os seus riscos

Este artigo fala sobre a criação e projeção de *Internet of Things Devices* e a sua complexidade. Refere que estes dispositivos exigem o uso de TPC's e que estes trazem consigo os tais riscos inerentes e ocultos, dado que afetam diretamente o dispositivo final de IoT.

- Para que são utilizados os TPC's? – Neste ponto referem a importância em se usar TPC's em dispositivos IoT.
    - Uso de alguns exemplos que deixam clara a ideia de que não existem este tipo de dispositivos sem o uso de TPC's;
    - Consciencialização da importância de se tentar proteger estes dispositivos.

- Porque é que os TPC's são realmente importantes? – Fala-se sobre a importância dos TPC's e a sua ligação interna com o dispositivo IoT em si.
    - Exposição a *cyber-attacks* e sua complexidade.
    - Dificuldade em proteger os TPC's, dada a forma como são disponibilizados.

- Ferramentas de segurança existentes são insuficientes – Deixa-se evidente que as ferramentas de segurança comuns necessitam de uma solução altamente eficaz para proteger os TPC's.
    - Cenário em que até os dispositivos mais protegidos podem estar expostos aos tais *cyber-attacks.*

- Conhecimento por parte dos atacantes – Ideia de que os atacantes têm um conhecimento prévio sobre os TPC's.
    - Pesquisa de vulnerabilidades de TPC's comuns de dispositivos IoT é algo simples de se fazer atualmente.
    - Descoberta de vulnerabilidades é algo altamente desejável para os atacantes.

- Importância de manter o dispositivo atualizado – O uso de TPC's pode dificultar a ideia de manter um dispositivo o mais atualizado possível.
    - Necessidade de atentar em novas versões dos vários componentes e sua posterior atualização.
    - Atentar nos CVE's dos componentes para tentar evitar ataques.

Através destes pequenos tópicos que o artigo aborda, a conclusão final a que chegam é que o uso de TPC's em dispositivo IoT é claramente inevitável. Dessa forma, tem de existir um cuidado extra quando se trata de segurança, tentando se mitigar ao máximo o risco de vulnerabilidades nestes componentes.

Este artigo encaixa perfeitamente na investigação que foi feita. Não só fortifica que existem riscos inerentes ao uso destes componentes de terceiros, como comprova também que este é um tema atual e para o qual é necessário tomar medidas.

Apesar dessas medidas não serem uma garantia a 100% para a segurança das várias aplicações/dispositivos/produtos finais, este artigo assegura que os métodos falados e referenciados no documento de apoio são os mais corretos a ter em conta e que devem ser postos em prática quando se pretende/precisar de usar *Third-party Components*.

Considerações Futuras

Ao abordarmos as considerações futuras estamos a dar um passo importante para a investigação no geral ao tentar fornecer uma visão daquilo que pode auxiliar a resolver os grandes problemas abordados ao longo de todo o documento de apoio.

O que se espera é que estes riscos e desafios sejam abordados pelas diferentes organizações, não de forma individual, mas algo cooperativo e que posso realmente fazer a diferença por fim a melhorar toda a situação global e com isso reduzir os tais *Risks Inherent in the Use of Third-party Components*.

Método	Descrição do Método
Crowdsourcing of Naming and Name Mapping	A ideia seria aprofundar a ideia de *unique names* abordada no subcapítulo *Maintain* - Manter uma lista de TPC's, onde as organizações atribuíssem um nome exclusivo para um dado TPC. O grande problema a lidar seria a grande escala de TPC's, pelo que se esperaria que as organizações colaborassem entre si de forma a conseguir criar um sistema de nomenclatura padrão.
Crowdsourcing of an End-of-life Repository	Criar uma *database* que providencie as datas de quando um TPC vai atingir o seu fim de ciclo de vida, ou seja, quando deixará de ser suportado pelo seu vendedor e quando as vulnerabilidades de segurança associadas a si estão sem qualquer tipo de resolução existente.
Crowdsourcing of a Vulnerability Source Listing	Criar uma lista das vulnerabilidades dos TPC's através das organizações em si como dos próprios fornecedores.

Tabela 7 Lista de possíveis implementações futuras a fazer-se na área

Conclusões

Este relatório serviu para abordar o assunto dos riscos de segurança existentes quando se usam *Third-party Components*. Para isso, teve-se como base um documento de apoio fornecido pelo docente e existente *online*, que permitiu compreender a definição e respetiva funcionalidade destes componentes na sua integração em futuros projetos de *software*.

Os *Third-party Components* são peças essenciais para a criação de novo *software* em pleno século XXI com as suas milhentas *frameworks* e métodos de incorporação. Com grande complexidade e escolha, vem também um perigo iminente sobre os componentes escolhidos para tal. Na análise deste projeto de investigação, bem como outros *websites* sobre o mesmo assunto temos como base que todos os métodos de vigilância, catalogação e proteção dos componentes que uma empresa deve ter em conta como necessários para minimizar prejuízos ou até mesmo prevenir casos vulneráveis facilmente identificáveis.

Esta gestão essencial de *Third-party Components* deve ser incorporada primordialmente na fase de construção de *software*, ou caso não seja possível dessa forma, o mais rápido possível para a prevenção dos projetos todos incluídos da empresa. Esta técnica, ainda que mais individual, não deixa de ser um passo essencial para a proteção basilar dos produtos a comercializar com os *TPCs*, dado que todos estes métodos são ainda atuais e excelentes soluções e *guidelines* a seguir para o mesmo.

Como consideração futura e pesquisa adicional entendeu-se que o passo essencial a seguir é realmente a criação de *standards* universais a todo o mundo da informática, como bases de dados independentes e criação de identificadores únicos transversais e utilizáveis para todas as entidades que emitem as vulnerabilidades e por aí em diante. Com isso em mente, a gestão, prevenção e remediação da segurança dos *Third-party Components* será universal e mais fácil de implementar.

Referências

AGREEMENT AND PLAN OF MERGER. (25 de julho de 2019). Obtido de Law Insider: https://www.lawinsider.com/contracts/8NVe5f8UNc0#third-party-components

Mohino, J. d., Higuera, J. B., Higuera, J. R., & Montalvo, J. A. (2019). The Application of a New Secure Software Development Life Cycle (S-SDLC) with Agile Methodologies. *Electronics 8.11*, 2.

SDLC (Software Development Life Cycle) Phases, Methodologies, Process, And Models. (10 de novembro de 2019). Obtido de Software Testing Help: https://www.softwaretestinghelp.com/software-development-life-cycle-sdlc/

SDLC (Software Development Life Cycle) Tutorial: What is, Phases, Model. (s.d.). Obtido de Guru99: https://www.guru99.com/software-development-life-cycle-tutorial.html

Svilicic, B., Rudan, I., Frančić, V., & Doričić, M. (2019). Shipboard ECDIS Cyber Security: Third-Party Component Threats. *Scientific Journal of Maritime Research*, 176-180.

The Hidden Risk in All IoT Devices: Third-Party Components. (25 de junho de 2019). Obtido de Sternum: https://www.sternumiot.com/blog/2019/6/25/third-party-components-the-hidden-risk-in-all-iot-devices

Xie, W., Hu, J., Kudjo, P. K., Yu, L., & Zeng, Z. (2018). A New Detection Method for Stack Overflow Vulnerability Based on Component Binary Code for Third-Party Component. *Institute of Electrical and Electronics Engineers*, 1-5.