**1.**

**Stakeholders:**

**> cliente**

**> fornecedores de hardware**

**> alguem do departamento legal (unidade certificadora)**

**Como se procede o registo?**

**> Online (website) ou no quiosque**

**Como confirmar os pagamentos da pessoa?**

**> Ex: pedir q seja enviada copia digital do CC**

**>> Definir requisitos**

**-> n se pode criar conta sem um nome (com + de 2 letras)**

**> Objetivos da fase**

**> Quem está envolvido**

**> Quais os outputs esperados**

**--------------------------------**

**2. Atributos de segurança:**

**-> CIA e AAA + Não Repudio**

**-> dar exemplos dos que se aplicam e qual o seu impacto no caso de estudo**

**3. Escrever requisitos**

**4. Escrever um teste por requisito**

**5. interfaces -> ex: email**

**Considere uma aplicação/solução para sistemas de carregamento de viaturas eléctricas (quiosque com monitor táctil).**

**Antes de iniciar o carregamento o utilizador deve autenticar-se (no ponto, por ligação wifi ou Bluetooth, através de app específica)**

**Acesso através de registo prévio**

**Na sua conta o utilizador consegue ver e alterar: Nome, Email, Morada, dados da viatura, dados do CC, nº de contribuinte, histórico dos consumos, histórico das faturas (mensais), cartão de crédito associado para pagamento direto, e outras configurações de conta (tipo de pagamento, Código de desconto, etc.**

**1: Descrever um plano genérico cobrindo, pelo menos, a parte da especificação do sistema (não da app), o desenvolvimento, testes e aceitação/instalação.**

O primeiro passo é, sempre, treino da equipa nas áreas necessárias, sendo cibersegurança uma delas.

Depois disto, a fase de especificação do sistema surge. Nesta fase, é necessário definir todos os standards que se vá usar, quer seja de criptografia (funções de hash que serão usadas, modos de cifra, etc.) ou métricas de bom funcionamento (quanto tempo de loading é demasiado, etc.). Também se definem as ferramentas que serão usadas ao longo do ciclo de vida do sistema.

Na fase de análise e requisitos é feita uma análise de risco, e alguns requisitos de segurança são concluídos daí. De notar que nem todos os requisitos de segurança surgem a partir desta análise, alguns surgem independentemente dos riscos.

Depois disto, surge o desenho detalhado da arquitetura, no qual se definem, também, os requisitos de arquitetura.

Após, inicia-se a implementação, em conjunto com os testes. Nesta fase usam-se os standards e ferramentas estabelecidas na fase de especificação do sistema. Sempre que se usarem componentes de outrem, será necessário controlar e documentar o risco de segurança destes. Também será necessário efetuar análise estática e dinâmica de segurança, e penetration testing.

Finalmente, chega a fase de lançamento, deployment e manutenção. Nesta fase deverá ser definido o processo que se deve seguir como resposta a um incidente.

**2: À luz de uma análise de ameaças de segurança (threats analysis) aplicável a este sistema:**

**Identificar os principais atributos de segurança que se devem ter em conta, e explicar brevemente cada um deles com alguns exemplos (3 atributos no mínimo, 6, no máximo).**

Há três grandes ângulos de segurança que devem ser garantidos: segurança física do utilizador, segurança da infraestrutura do sistema, e segurança dos dados do utilizador.

A segurança física do utilizador passa por garantir que o carregamento do automóvel é feito seguramente. Isto significa que não deverá haver descargas de eletricidade inesperadas (isto é, o utilizador não deverá receber um choque elétrico ao carregar o carro) e o carro será carregado (isto é, o carro não sofre danos devido a voltagem inadequada do quiosque e de facto é carregado).

A segurança do quiosque é mais difícil de garantir. O quiosque deverá feito de materiais que não deterem ao ar livre (expostos a chuva, vento, calor, frio, sol, etc.) e deverão ser fortes o suficiente que não sejam facilmente vandalizados.

Finalmente, temos a segurança dos dados dos utilizadores, que é a segurança da base de dados. Esta não deverá ser acessível a pessoas externas, e os dados nela deverão corresponder à realidade.

**3: Escrever entre 5 e 10 requisitos relacionados com a segurança funcional do sistema, cobrindo tópicos relacionados com os atributos apresentados na questão 2. Fazer esse mapeamento, mesmo que alguns requisitos mapeiem com mais do que um atributo.**

Requisito 1:

Quiosques não podem ter down time maior do que 5 horas por dia.

Requisito 2:

Caso um quiosque seja danificado, o tempo entre reportar este acontecimento e a reparação não poderá exceder 4 horas

Requisito 3:

O quiosque terá de ser feito de material que apenas precise de ser trocado uma vez de sete em sete anos.

Requisito 3.5:

O quiosque será feito de material que tem certificado de tempo útil igual ou superior a sete anos.

Estes requisitos associam-se à segurança do quiosque.

Requisito 4:

O quiosque não poderá ter voltagem tal que danifique carros elétricos. A voltagem recomendada são 120V.

Requisito 5:

O quiosque não terá descargas elétricas que possam injuriar o utilizador enquanto este liga o carro ao carregador.

Estes requisitos associam-se à segurança física do utilizador e dos seus bens.

**4: Definir um conjunto de testes para confirmar a correta implementação dos requisitos escritos na questão 3. Mapear cada caso de teste ao(s) respetivo(s) requisito(s).**

Teste 1:

Instalar alguns protótipos de quiosques, pré release. Comunicar de hora a hora com cada um para verificar se algum está em baixo.

Aplica-se ao Requisito 1 e, em parte, ao Requisito 3, apesar deste não ser testável em tempo útil.

Teste 2:

Durante o teste 1, caso haja alguma avaria física, medir o tempo entre a descoberta e a solução. Este tempo deverá ser menos de 4 horas.

Aplica-se ao Requisito 2.

Teste 3:

Durante a construção dos quiosques, e durante o Teste 1, a voltagem que carrega os carros será medida.

Aplica-se ao Requisito 4.

Teste 4:

Durante a construção dos quiosques, e durante o Teste 1, os quiosques serão usados por uma equipa que, devidamente equipada de modo a não sofrerem danos físicos, verificam se há descargas elétricas indevidas.

Aplica-se ao Requisito 5.

**5: Para testar a robustez da solução, fornecer os seguintes dados:**

**• Lista de todas as interfaces internas e externas da solução.**

**• Nomear e descrever (até 100 palavras cada) dois ataques de segurança que poderiam ser descobertos ao aplicar uma metodologia de penetration testing.**

**• Apresentar 3 soluções para evitar ataques de DOS (Denial-of-Service).**

As interfaces externas que estão a ser tidas em conta são todos os pontos de contacto do utilizador com o sistema, ou seja, o quiosque e a aplicação.

As interfaces internas serão, portanto, o backend de todo o sistema (quiosque e aplicação) e todos os serviços usados no sistema (base de dados).

Um possível ataque poderia ser uma SQL injection no login da aplicação móvel. Neste login, usar-se-ia um e-mail como “asd@asd.com or 1==1; -- “. Se SQL injections forem possíveis, este valor dará meio a um atacante de começar sessão como o primeiro utilizador da base de dados.

Outro possível ataque seria exploração da possibilidade de submeter dados, sendo esta funcionalidade parte do sistema de confirmação de identidade – será suposto o utilizador submeter uma imagem do CC, mas qualquer tipo de ficheiro pode ser enviado. Assim, será possível enviar uma reverse shell, e assim ganhar acesso à máquina host.

Um método para evitar ataques DOS será proteger os cabos do quiosque. Se estes forem acessíveis, isto é, estiverem expostos, um atacante poderá cortá-los, o que origina denial of service naquele quiosque.

Outro método será limitar o número de caracteres de cada campo de input. Isto deverá ser feito porque, hipoteticamente, um atacante poderia criar um pedido com volume tal de dados que o sistema ficaria inacessível a qualquer utilizador, uma vez que este estaria a lidar com o pedido impossivelmente grande.

Finalmente, os equipamentos responsáveis pela rede não deverão estar acessíveis a estranhos, uma vez que, se uma pessoa tiver acesso aos equipamentos da rede, conseguirá modificar as configurações da rede, ou danificá-los. Qualquer uma destas duas opções criará denial of service.

**6: O cliente identificou 4 ameaças que pretende ver resolvidas com a própria solução (sem intervenções externas, nem interrupções no serviço), e estas são:**

**• A) Instalação de vírus ou malware através do terminal**

**• B) Acesso indevido a funções de configuração do terminal (deve ser permitido apenas com o utilizador “maintenance”)**

**• C) Obtenção de dados confidenciais de outro utilizador**

**• D) Utilização de uma conta alheia para carregamento**

**No âmbito de uma análise de ameaças (hazard analysis) fornecer, para cada ameaça listada pelo cliente, a seguinte informação:**

**- Possíveis consequências;**

**- Possíveis causas;**

**- Algumas medidas a implementar/adoptar para eliminar a ameaça.**

No que toca à instalação de malware através do terminal, algumas consequências são: comprometimento de dados de utilizadores, denial of service e denial of service com o acréscimo de pedido de resgate para “salvar” a máquina.

A causa disto poderá ser a exposição de portas USB que tornem possível a injeção de malware na máquina.

Assim, uma medida para eliminar a ameaça será desligar todas as portas USB que estejam acessíveis ao público.

No que toca a acesso indevido a funções de configuração do terminal, a consequência principal será a alteração das configurações. Isto poderá criar vários problemas: desde mudar a voltagem de saída, danificando carros que sejam carregados mais tarde, a pura e simplesmente desligar o quiosque.

Possíveis causas são PINs fracos (por exemplo, ‘1234’ ou ‘1111’) e a exposição do number pad ao público.

Mitigar este problema passará por forçar PINs fortes (PINs mais compridos, regras de inexistência de padrões, etc. – bancos têm algumas medidas interessantes como não autorizar dois valores repetidos juntos) e, se possível, dificultar o acesso ao numberpad. Colocá-lo atrás de um painel aparafusado com parafusos que requeiram chaves específicas será uma opção. Também poderá ser implementada uma política de duplo fator: o number pad apenas aceitará input quando o quiosque for marcado, noutro sistema, como precisando de manutenção. Depois de ser marcado, apenas alguns dos membros da equipa de manutenção (ou seja, apenas alguns dos PINs existentes e ‘aceitáveis’) poderão desbloquear o modo de manutenção. Finalmente, requerer o número mecanográfico usado na empresa e o PIN será uma opção.

No que toca à divulgação de dados de clientes, algumas consequências para a empresa serão multas (por causa de violação de proteção de dados) e, maioritariamente, um problema de imagem – confiança será perdida. No entanto, utilizadores também sofrerão consequências. Considerando que o sistema tem dados críticos (dados de CC, dados bancários) um mau ator poderá aproveitar-se deles e roubar identidade. Também existe o facto de os dados terem a localização dos quiosques usados, que poderá fazer com que maus atores consigam perceber as rotinas de utilizadores, e aproveitar essa informação para cometer crimes variados. Finalmente, considerando que a password dos utilizadores está guardada, se esta for divulgada e utilizadores reutilizarem a password (ou a alterarem minimamente) com o mesmo e-mail, será difícil ter em conta quantas outras contas ficarão comprometidas.

A causa de uma destas situações poderá ser más credenciais de acesso a bases de dados, inexistência de cifração na base de dados (todos os dados estão guardados em plaintext) ou acessos indevidos à base de dados (via SQL injections ou apenas acedendo-lhe diretamente).

Algumas medidas que poderão minimizar o risco são mudar as credenciais de acesso à base de dados regularmente, usando credenciais fortes (difíceis de adivinhar). Também é recomendado que os dados presentes na base de dados sejam cifrados, e que passwords sejam hashed. Também deverão ser feitos testes exaustivos a todos os pontos que possam facilitar a existência de SQL injections. Finalmente, acessos diretos à base de dados terão de ser limitados de modo a apenas pessoas trusted o possam fazer.

A utilização de conta alheia para carregamento é difícil de lidar com.

As consequências são na maioria para o utilizador, uma vez que serão os dados deste que serão postos em risco. No entanto, se isto for uma ocorrência comum, as consequências serão parecias com as estabelecidas na divulgação de dados de clientes.

A causa destas situações será variada. Credenciais fracas de utilizadores, divulgação de dados prévia, SQL injections e outras vulnerabilidades.

Algumas maneiras de prevenir esta possibilidade são usar two factor authentication, pedir credenciais fortes para login, e, mais uma vez, garantir que as bases de dados e dados do utilizador são guardados seguramente.

**A tua empresa está a desenvolver uma solução para promover compra e venda online, de produtos, de todo o tipo de produtos.**

**Terá de haver um particionamento dos utilizadores, como todos os produtos são permitidos, certos utilizadores, por exemplo, menores de 18 anos, não poderão ver nem aceder a produtos derivados do tabaco, alcool ou pornografia.**

**A solução, donominada de Amazon++, deverá ser acessível por internet em qualquer browser (provavelmente terá de se definir uma lista de browsers inicial, onde a aplicação será efectivamente testada e deverá ser barrada a sua utilização para outros browsers, a não ser que se convença o cliente de que a aplicação pode funcionar noutros browsers sem problemas de segurança).**

**Do ponto de vista de utilizadores, haverá 3 tipos:**

**1. Os gestores da plataforma, com acesso total, podendo colocar restrições em certos utilizadores ou ver informação apenas acessível a eles, nomeadamente contactos das pessoas e a imagem de um documento oficial que é obrigatório para o completo registo no sistema. Eles serão também responsáveis pela manutenção e resolução de conflitos caso não haja entendimento entre comprador e vendedor.**

**2. Utilizador-vendedor, utilizadores que tanto podem comprar como vender, para atingirem este tipo, os utilizadores devem fornecer mais informações à plataforma como o número de identificação nacional, uma morada válida para onde é enviado um pin, e a imagem de um documento oficial;**

**3. Utilizador comprador, com menos restrições que o anterior, mas que apenas pode efectuar compras no site.**

**Para além destes tipos, os utilizadores são ainda categorizados conforme a idade legal, havendo certos produtos que não podem ser adquiridos por certos utilizadores.**

**O sistema deve integrar com diversos meios de pagamento (MBWay, Multibanco, Transferência Bancária, Cartão de Crédito ou Débito, Paypal, etc).**

**O sistema deve ainda ter um mecanismo de detecção se o produto ou produtos foram colocados na categoria correcta, este mecanismo utiliza um algoritmo inovador de inteligência artificial que a tua empresa desenvolveu, mas que ainda não foi totalmente posto à prova.**

**A arquitectura da aplicação resta a definir bem como requisitos de backup, segurança, encriptação de dados, certificação do algoritmo de categorização dos produtos, etc.**

**Explica, em dois parágrafos, quais as duas fases que consideras mais importantes, para o desenvolvimento da solução, e porque essas duas fases são importantes do ponto de vista de security e robustez da solução.**

As duas fases que considero mais importantes são a fase de análise de riscos e a fase de inception.

A fase de inception é importante, uma vez que é nesta que são definidas as ferramentas e sistemas criptográficos que serão usados. Isso significa que nesta fase já é considerado quais linguagens e IDEs são aprovados, o que leva a toda uma lista de vulnerabilidades conhecidas que se terá de lidar com. É importante saber-se quais as vulnerabilidades comuns de uma ferramenta e linguagem no início do desenvolvimento. Descobri-las a meio ou, no pior caso, depois da solução já ter sido released, pode ser catastrófico. Também acho esta fase importante porque é a primeira depois do treino, e se esta tiver pouco ênfase em segurança e robustez será difícil que as outras tenham.

A fase de análise de riscos também é extremamente importante, na minha opinião, porque será a partir desta análise que alguns dos requisitos de segurança serão criados. E também é necessário ter em conta que será nesta fase que “fringe behavior” é tido em conta: o que é suposto fazer quando uma componente não funciona? Qual o impacto? E se houver um terramoto? A segurança e robustez de uma solução são tão boas quanto os requisitos criados, e a fase de análise de riscos é uma das fases que mais contribui para que estes sejam bons.

**À luz de uma análise de ameaças de segurança (threats analysis) aplicável a este sistema:**

**a) Identifica os principais atributos de segurança que se devem ter em conta, e explicar brevemente cada um deles com alguns exemplos (3 atributos no mínimo, 6, no máximo).**

**b) Lista 3 tipos de atacantes potenciais e quais seriam as suas motivações.**

**c) Identifica uma vulnerabilidade (genérica) que deveria ser analizada para as possíveis tecnologias envolvidas.**

Os atributos de segurança que devem ser tidos em conta são a tríade CIA e o não repúdio. Assim, os atributos de segurança são a confidencialidade, integridade, availability e não repúdio.

Um exemplo de não repúdio aplicável a este sistema é um vendedor não poder alegar que não foi ele que colocou um item à venda. No que toca à confidencialidade, um exemplo será não ser possível saber a identidade de todos os utilizadores da plataforma, não devendo também ser possível saber certos dados privados de utilizadores (idade, morada, etc.). No que toca à integridade, isto significa que os dados que estão na plataforma deverão estar corretos. Isto significa que não só os dados dos utilizadores deverão estar corretos (não ser possível colocar uma morada que não exista, ou requerir prova de habitação na morada) e que os dados dos itens que estão a ser vendidos são os corretos (não ser possível mentir sobre dimensão, qualidade, etc.) No que toca a availability, isto significa que o sistema deverá manter-se em funcionamento.

Na minha opinião, haverá três tipos de atacantes possíveis. O primeiro, e possivelmente menos perigoso, será um hack kiddie – alguém que está a tentar atacar o sistema por diversão, ou para poder dizer que o fez. O segundo tipo de atacante será um hacktivist, que poderá, ou não, ser perigoso. Este atacante atacará a plataforma como ato de ativismo, há algo na plataforma ao qual se opõe. No caso do sistema, poderá ser algo como a limitação de venda de álcool e outras substâncias a menores de 18. Finalmente, existe o atacante extrator – este atacante tem como objetivo extrair informação de utilizadores para uso próprio (roubo de identidade) ou para venda. A motivação deste será, portanto, a obtenção de dados para futuro benefício monetário.

Uma vulnerabilidade que não pode existir neste tipo de sistema é um “leaky system”. Isto é, o sistema não poderá revelar, em qualquer altura, mais dados do que os estritamente necessários. Quer seja da existência (ou não) de uma conta com um certo e-mail, ou um cartão já estar associado a outra conta na plataforma, mensagens de erro default de ferramentas usadas, etc. Mesmo que estes dados sejam úteis para utilizadores, são extremamente perigosos. Um atacante com acesso a mais dados do que deveria ter é um atacante com mais uma arma no seu arsenal.

**A FMEA é uma técnica para analisar concretamente modos de falha, e quais seriam as causas, efeitos, o método de detecção e possíveis alterações ao sistema para reduzir o impacto das falhas individuais.**

**Para o sistema a desenvolver, identifica 3 funções do mesmo (por exemplo “fazer o log de todas as operações”, “emitir um alarme”, "gerar um relatório") e analisa com os modos de falha básicos (função não executada, função tardiamente executada e função mal executada) quais seriam:**

* **Possíveis métodos de detecção da falha,**
* **Efeitos previstos se a falha não for eliminada, e**
* **Que sugestões farias para eliminar a falha ou para controlar os seus efeitos.**

As funcionalidades que serão tidas em conta são:

- a funcionalidade que esconde itens indevidos a menores de idade (para simplicidade, será referido como age locking)

- pagamentos

- deteção de categorias

Na resposta, são discutidos métodos de deteção, uma vez que, para eliminar as falhas, será necessário perceber em que casos é que esta ocorre. Depois de perceber os casos comuns, deverá ser feito o debug da função ou, se impossível, criar avisos para utilizadores, ou outros métodos vistos como necessários e apropriados.

No que toca à funcionalidade de age locking, a função não executada significa que qualquer utilizador de qualquer idade terá acesso a qualquer item da plataforma. Isto não só poderá expor menores a conteúdo que não deveriam ver (conteúdo pornográfico, uso de drogas, etc.) como poderá ter repercussões de imagem, monetárias e talvez até legais para a empresa.

Para simplicidade, será explorada apenas a possibilidade de venda de tabaco. As repercussões de imagem são auto explanatórias: utilizadores que saibam que a plataforma vende tabaco a menores terão uma imagem negativa da empresa. As repercussões monetárias são multas que a empresa receba, porque venda de tabaco a menores de idade é ilegal. Considerando que esta venda é ilegal, poderá haver um caso em tribunal.

Este modo de funcionamento poderá ser testado ao aceder à plataforma, regularmente, como utilizador menor de idade, e assim tentar aceder a itens que não se deverá ter acesso a.

A função tardia será analisada como itens serem apenas marcados como “not suitable” para menores (e assim age locked) algumas horas depois de serem publicados. Isto terá as mesmas consequências mencionadas acima. O mesmo acontece para o método de deteção.

Na função mal executada, esta será tida em conta como alguns itens serem age locked quando não deviam ser. Isto levará a menos vendas de itens, que poderá ter consequências na imagem da plataforma: esta será vista como estando a censurar vendedores quando estes estão a vender itens apropriados para todas as idades. O método de deteção passará por, regularmente, fazer as mesmas pesquisas em contas menores e maiores de idade, e comparar os itens que surgem. Outro método será rever, regularmente, os itens que foram age locked, e a partir daí corrigir possíveis erros.

No que toca aos pagamentos, o não funcionamento significará que pagamentos não serão processados. Isto tem repercussões, também, na imagem da empresa, uma vez que vendedores não serão capazes de vender itens, o que é o core do funcionamento do sistema. Um método de deteção poderá ser simular vendas regularmente, testando assim se as vendas estão a ser feitas.

O funcionamento tardio significará que o pagamento será feito horas após o utilizador o fazer. Isto não terá consequências tangíveis para vendedores, uma vez que o pagamento será feito na mesma, com a ressalva que o pagamento ainda será recebido em tempo útil. Um método de deteção será o mencionado acima, mas o tempo entre o pagamento e o recebimento deste deverá ser temporizado.

O funcionamento errado significará que o pagamento será feito, mas enviado para o vendedor errado. Isto significará que vendedores não recebem o que venderam, o que significa que perdem lucro. Isto será péssimo para a imagem da empresa e até poderá dar origem a casos em tribunal.

No que toca à deteção de categorias, o não funcionamento significará que itens publicados não terão categorias associadas. Isto significa que a procura será mais difícil, uma vez que os itens não estão categorizados. Também poderá significar que o mecanismo de age locking não funcione, o que traz consigo as consequências já mencionadas. Para detetar este modo de funcionamento, deverá ser introduzido, regularmente, um item virtual, e deverá ser verificado se este foi categorizado.

O funcionamento tardio significa que a categorização ocorre passado algum tempo. Isto poderá não ter consequências notáveis para vendedores caso a categorização ocorra em tempo útil. Se este não for o caso, as consequências são as mencionadas acima. Para detetar isto, ao efetuar o teste acima, deverá ser cronometrado o tempo entre a introdução do item e a categorização.

O funcionamento errado significa que itens não são categorizados corretamente. As consequências são as mencionadas no não funcionamento desta mesma função, com a ressalva que agora os itens estão categorizados, mas erradamente. O método de deteção será parecido com o discutido, também, nesse modo de funcionamento, mas deverá ser verificado se a categoria está correta.

**Descrever 6 requisitos relacionados com a segurança funcional do sistema. Caso a descrição do sistema a desenvolver não permita identificar claramente requisitos fazer pressupostos (o cliente na maioria das vezes não faz bem ideia do que precisa, é perfeitamente normal fazer propostas lógicas e com sentido).**

**Para cada requisito identificar quais os atributos de segurança, garantido que existe pelo menos um requisito relacionado com "Auditability".**

Requisito 1 (Integrity):

O sistema categorizará itens corretamente 90% das vezes.

Requisito 2 (Integrity):

Caso um vendedor ou utilizador marque um item como mal categorizado, a categorização será feita manualmente por um membro da equipa.

Requisito 3 (Auditability, Non-Repudiation):

De cada vez que haja uma categorização manual, esta será guardada na lista de categorizações, com a identificação do membro da equipa que a fez, o item categorizado, e ambas as categorizações (automática e manual).

Requisito 4 (Confidentiality):

O histórico de compra de um utilizador não será acessível a outros utilizadores

Requisito 4.1 (Confidentiality):

O histórico de vendas de um utilizador será a única exceção ao requisito 4, uma vez de um vendedor terá acesso ao seu próprio histórico de vendas, que terá em si partes do histórico de compras de um utilizador (o vendedor V terá acesso apenas ao registo da compra C do utilizador U se esta compra for feita, pelo utilizador U, ao vendedor V).

Requisito 5 (Availability):

O sistema não terá falhas de quality of service quando 10 000 utilizadores estão ativos.

**Definir um conjunto de testes (sob a forma de uma especificação simples) para confirmar a correcta implementação dos requisitos escritos na questão 3. Mapear cada caso de teste ao(s) respectivo(s) requisito(s).**

**Nota: Exemplo de especificação simples:**

**Testar que no momento de registo de um novo utilizador, o mesmo recebe um email para confirmar o seu registo em menos de 5 minutos. Caso o mesmo nunca receber o email o registo fica pendente. Caso o email chegue depois de 5 minutos, o registo não deve ser confirmado e um novo email deve ser gerado.**

**Nota: Um requisito pode perfeitamente ser não testável de forma dinâmica, nesse caso apresenta como o mesmo poderá ser verificado.**

Teste 1 (Requisito 1):

Regularmente (4 vezes por dia) a equipa inserirá itens virtuais na plataforma, que não estarão acessíveis ao público, e verificará se a categorização foi feita corretamente. A partir disto, a percentagem de categorizações corretas será calculada

Teste 2 (Requisito 2+3):

Regularmente (4 vezes por dia), um membro da equipa marcará um item virtual como mal categorizado, e o processo de categorização será feito. Depois deste ser feito, a lista de categorizações manuais será consultada, e terá de ter a categorização mencionada aqui com os dados corretos.

Teste 3 (Requisito 4):

Regularmente, a equipa deverá aceder, como utilizador virtual com histórico de compras virtual, ao histórico de compras, e este terá de estar correto.

Teste 4 (Requisito 4):

Regularmente, a equipa deverá tentar aceder, como um utilizador virtual, ao histórico de compra de outro utilizador. Isto não deverá ser possível.

Teste 5 (Requisito 4.1):

Regularmente, como um vendedor virtual, deverá ser verificado se o histórico de vendas está completo.

Teste 6 (Requisito 5):

Serão criados 9 999 “utilizadores” (bots) com tarefas a cumprir na plataforma. Também haverá um utilizador humano, parte da equipa, que estará na plataforma, a efetuar uma tarefa pré-estabelecida. O tempo que o utilizador humano demora a completar a tarefa será temporizado. O tempo que o mesmo utilizador demora a completar a tarefa sem os 9 999 “utilizadores” na plataforma também será cronometrado, e ambos os tempos serão comparados.

**Com vista a testar a robustez da solução, fornecer os seguintes dados:**

**• Lista de todas as interfaces internas e externas da solução.**

**• Nomear e descrever (até 100 palavras cada) dois ataques de segurança que poderiam ser descobertos ao aplicar uma metodologia de penetration testing.**

**• Apresentar 2 soluções ou boas práticas para evitar ataques de XSS (Cross site scripting).**

Algumas interfaces internas são os servidores, bases de dados e serviços de verificação internos. Algumas interfaces externas são e-mail, itens de vendedores e comunicação entre compradores e vendedores.

Um ataque que pode ser descoberto usando penetration testing será o acesso a contas alheias via SQL injection. Por exemplo, no login, se o e-mail colocado for “asd’ or 1=1; – “ e a plataforma for vulnerável a SQL injections, um atacante poderá aceder ao sistema como o primeiro utilizador da base de dados.

Outro ataque poderá ser feito na inserção de imagens de itens. Se for aceite qualquer tipo de ficheiro, será possível criar uma reverse shell e submete-la, ganhando assim acesso à máquina host do sistema.

Uma boa prática no que toca a Cross Site Scripting é expor (em vez de embeber na página) todo o conteúdo inserido por utilizadores. Um método para tentar evitar cross site scripting é proibir a utilização de caracteres como ‘<’ ou ‘>’, mas estes caracteres também poderão ser usados fora do contexto de XSS.

**Pensando agora no sistema que deve ser implementado, no ambiente onde ele vai operar, nas possíveis tecnologias envolvidas e numa potential arquitectura, quais seriam, em conclusão, os 5 (cinco) principais princípios de design de segurança que não poderiam ser ignorados para este projecto?**

Os cinco princípios de design de segurança que não podem ser ignorados para este projeto são:

- aplicar defesa em profundidade, uma vez que se estão a lidar com dados críticos de utilizadores, que terão de ser mantidos em segurança em todos os níveis

- detetar intrusões o mais cedo possível, uma vez que o perigo para os utilizadores será maior quanto mais tempo estes não souberem que os seus dados foram comprometidos

- não confiar em serviços, uma vez que, mais uma vez, os dados críticos de utilizadores estão a ser usados nestes, então uma approach de ‘safety first’ será importante

- não confiar em infraestruturas, pela mesma razão mencionada acima

- modelo de segurança positivo, uma vez que será mais fácil estabelecer quem pode aceder (sendo estas pessoas de confiança) em vez de definir quem não o pode fazer – é mais seguro apenas ser acessível a quem já provou que é de confiança, em vez de dar acesso a quem ainda não provou que não é de confiança