Avaliação de Abordagem para Definição de Requisitos de Cibersegurança em Projetos Ágeis

Daniel M. D. Crespo
CEFET-MG
Leopoldina, Minas Gerais, Brasil
mescolincefet@gmail.com

Túlio Taveira Macioci
CEFET-MG
Leopoldina, Minas Gerais, Brasil
taveiramaciocitulio@gmail.com

Maria Júlia M. Schettini
CEFET-MG
Leopoldina, Minas Gerais, Brasil
mariajuliaschettini97@gmail.com

Gabriella C. B. Costa
CEFET-MG
Leopoldina, Minas Gerais, Brasil
gabriella@cefetmg.br

ABSTRACT

In agile software development contexts, the definition and consistent interpretation of cybersecurity requirements remain a challenge, especially when such requirements are expressed through user stories with varying levels of detail and clarity. The lack of technical and architectural context can compromise the uniform identification of security requirements throughout the development process. This study presents an analysis of the application of cybersecurity requirements from the OWASP ASVS standard to 66 user stories, conducted by four distinct analysts. The Agile Security Framework (ASF) tool was used to annotate the requirements, and Python scripts were employed for data processing. The study evaluated the frequency, distribution, and convergence of 1,634 security requirement annotations. The results indicate a significant predominance of category V5 (Validation, Sanitization, and Encoding), accounting for 51.5% of the annotations, followed by V3 (Session Management) with 16.9% and V2 (Authentication) with 8.4%. The analysis revealed that, while satisfactory convergence exists in user stories with a clear scope, such as those related to authentication, textual ambiguity and the absence of architectural context in more vague stories lead to significant interpretative divergences among analysts. It is concluded that the textual clarity of user stories is a critical factor for consistency in the identification of security requirements.

KEYWORDS

Cybersecurity, Software Development, , Requirements.

1 Introdução

A Engenharia de Software fornece métodos, ferramentas e práticas essenciais para gerenciar a complexidade do desenvolvimento de sistemas, desde a sua concepção até a manutenção [1]. Tradicionalmente, busca garantir qualidade, confiabilidade e eficiência no processo de construção de software. Contudo, com a crescente presença de sistemas em aplicações críticas e amplamente conectadas à internet, a cibersegurança emergiu como um aspecto central e indispensável desse processo. A segurança deixou de ser uma reflexão tardia para ser compreendida como um requisito de qualidade fundamental, que deve permear todas as etapas do ciclo de vida do desenvolvimento de software. Esse movimento foi intensificado pela consolidação de regulamentações nacionais e internacionais voltadas à proteção de dados e à segurança da informação, como a Lei Geral de Proteção de Dados (LGPD) [2], a Política Nacional de Segurança da Informação (PNSI) [3], a Política Nacional de Cibersegurança (PNCiber) [4], a Lei de Acesso à Informação (LAI) [5] e o Regulamento Geral de Proteção de Dados da União Europeia (GDPR) [6], que impõem a adoção de medidas técnicas e organizacionais de segurança desde a fase de elaboração dos sistemas.

Nesse contexto, consolidou-se o conceito de *Security by Design*, que preconiza a consideração sistemática de aspectos de segurança desde as fases iniciais do desenvolvimento, incluindo requisitos, modelagem e design [7]. Para operacionalizar essa abordagem, o *Open Worldwide Application Security Project* (OWASP) [8] propõe *frameworks* como o *Application Security Verification Standard* (ASVS) [9], que define controles técnicos e níveis de verificação para aplicações *web*. Contudo, um desafio no desenvolvimento ágil é aplicar eficientemente esses controles nos artefatos de requisitos, como histórias de usuário.

Nesse sentido, foi desenvolvida a ferramenta ASF (Agile Security Framework), que auxilia na definição de requisitos de cibersegurança baseada na análise de histórias de usuário alinhadas ao ASVS. A ferramenta organiza o processo em níveis hierárquicos, agrupando requisitos em questões específicas e gerais. Para avaliar a abordagem empregada pelo ASF, este artigo apresenta um estudo conduzido a partir de dados extraídos de um repositório público hospedado no *Mendeley Data*[1]. O repositório original abriga uma coleção de 22 conjuntos de dados, obtidos de fontes online ou cedidos por empresas de software sob permissão,

1 https://data.mendeley.com/datasets/7zbk8zsd8y/1

cada um contendo mais de 50 requisitos expressos como histórias de usuário. Para esta pesquisa, foi selecionado especificamente o conjunto "g21-badcamp.txt", composto por 66 histórias de usuário. Esse subconjunto é oriundo de um repositório no GitHub referente ao BADCamp, uma conferência anual dedicada a sites de código aberto em Drupal. O site do evento atua como uma ferramenta de apoio aos participantes, e seu escopo engloba diversos recursos adicionados ao longo do tempo. Dessa forma, o estudo consistiu na análise dessas histórias por quatro analistas de desenvolvimento de software, a finalidade central é compreender como os analistas utilizaram a ferramenta ASF, avaliando a frequência, a distribuição, a convergência interpretativa e a cobertura temática dos requisitos identificados. Dessa forma, busca-se validar a metodologia e identificar padrões e desafios na aplicação de requisitos de cibersegurança no desenvolvimento ágil.

O restante deste trabalho está organizado da seguinte forma: a Seção 2 apresenta a fundamentação teórica. A Seção 3 contextualiza o problema investigado. A Seção 4 detalha a metodologia adotada. A Seção 5 apresenta e analisa os resultados, bem como as ameaças à validade do estudo realizado. A Seção 6, por fim, reúne as considerações finais.

2 Fundamentação Teórica

Esta seção apresenta os principais conceitos envolvidos no desenvolvimento deste trabalho. Na Seção 2.1, é apresentado o conceito de engenharia de software. Já na Seção 2.2 encontra-se a definição de cibersegurança. Por fim, a Seção 2.3 apresenta o OWASP e o ASVS para o entendimento geral deste documento.

2.1 Engenharia de Software

A Engenharia de Software é definida como uma disciplina de engenharia sistemática, disciplinada e quantificável, focada na produção de sistemas de software em larga escala. Conforme Sommerville [1], o objetivo primário desta disciplina é fornecer os métodos e ferramentas para gerenciar a complexidade inerente ao desenvolvimento de software, garantindo que os produtos finais sejam entregues dentro do prazo, do orçamento e, crucialmente, que atendam aos padrões de qualidade de software. O conceito de "qualidade" em Engenharia de Software é multidimensional. Embora historicamente focada na correção funcional, isto é, se o software executa as funções especificadas no levantamento de requisitos, a disciplina moderna expande esta definição para abranger um conjunto complexo de atributos, frequentemente classificados como requisitos funcionais e não funcionais (RNFs) [1].

Tradicionalmente, a segurança era classificada como um desses RNFs, ao lado de atributos como desempenho, usabilidade e manutenibilidade. Contudo, o contexto contemporâneo de sistemas de software, caracterizado pela conectividade global e integração em aplicações críticas (financeiras, médicas, infraestruturais), tornou essa classificação insuficiente. A segurança, no cenário atual, exibe uma propriedade de dominância de falha sobre os demais atributos de qualidade. Um sistema de software pode ser funcionalmente correto, ter excelente desempenho e alta usabilidade, mas uma única falha de segurança (como a exposição de dados confidenciais) pode anular o valor de todos os outros atributos, com implicações financeiras, legais e de reputação.

Desta forma, a segurança transcende sua classificação como um RNF secundário. Ela evoluiu para ser entendida como um pilar primário e indispensável da qualidade do software. A Engenharia de Software moderna deve, por definição, tratar a segurança não como uma restrição acessória ou uma reflexão tardia, mas como um requisito de qualidade fundamental e não negociável, que deve permear todo o processo de desenvolvimento [1, 7, 10].

2.2 Cibersegurança

A cibersegurança, definida por Pfleeger [11], foca na proteção de sistemas contra danos ou acesso não autorizado. Esta proteção é articulada pela "Tríade CIA", que busca preservar: 1) Confidencialidade (acesso apenas por autorizados); 2) Integridade (garantia contra modificações impróprias); e 3) Disponibilidade (acesso ao sistema quando necessário).

A crescente complexidade do software expôs a falha do paradigma tradicional de segurança. Historicamente, o modelo reativo, o qual trata o software como uma "caixa preta" a ser protegida externamente, se tornou uma estratégia falha por ignorar as vulnerabilidades, fraquezas exploráveis no design ou na implementação, intrínsecas ao próprio software [12]. Um ataque explora ativamente essa fraqueza interna, não apenas o perímetro. Demonstra-se, assim, a insuficiência do modelo reativo, que ignora a origem das falhas na lógica de negócios e impõe a necessidade de integrar controles (salvaguardas) diretamente ao ciclo de vida de desenvolvimento de software (SDLC).

Em contrapartida, a Engenharia de Software Segura adota o paradigma proativo do *Security by Design*. Sua premissa é que a segurança deve ser intrínseca ao design, não "aplicada" externamente após o desenvolvimento [10]. A principal prática para isso é a Modelagem de Ameaças (Threat Modeling), uma atividade de engenharia estruturada onde a equipe identifica riscos e controles antes da codificação. A lógica é econômica: encontrar e corrigir falhas de design precocemente é exponencialmente mais barato do que em produção. Essa prática é a ponte conceitual que força engenheiros a pensar como adversários durante o design, alinhando cibersegurança e engenharia de software.

2.3 OWASP e ASVS

Para orientar a implementação prática do Security by Design, as equipes de desenvolvimento recorrem a organizações de referência e padrões consolidados. A principal entidade global neste domínio é a OWASP, uma fundação internacional sem fins lucrativos cuja missão é "trabalhar para melhorar a segurança do software" [8]. A organização produz uma vasta gama de recursos gratuitos, incluindo documentação, ferramentas, padrões e

capítulos locais, tornando-se a "fonte de fato" para desenvolvedores e profissionais de segurança.

A publicação mais conhecida da OWASP é o OWASP Top 10, que identifica e classifica os dez riscos de segurança mais críticos para aplicações web, como A01:2021-Broken Access Control, A02:2021-Cryptographic Failures e A03:2021-Injection. O Top 10 desempenha papel fundamental na educação e priorização de riscos. No entanto, embora seja uma ferramenta inestimável de conscientização, não foi projetado como padrão completo de verificação nem guia exaustivo de implementação de controles.

O OWASP ASVS preenche essa lacuna, definindo requisitos verificáveis para segurança de aplicações web e fornecendo base sistemática para testar controles técnicos ao longo do ciclo de desenvolvimento. Em sua versão 4.0.3, o ASVS normaliza a cobertura e o rigor da verificação de segurança, atuando como métrica para avaliar o nível de confiança de uma aplicação, como guia de desenvolvimento ao estabelecer requisitos claros desde as fases iniciais, e como base de contratação para especificação objetiva de requisitos.

A estrutura do ASVS 4.0.3 é organizada em 14 categorias de verificação (prefixadas com 'V'), codificando explicitamente o princípio do Security by Design. A categoria V1: Arquitetura, Design e Modelagem de Ameaças, posicionada como primeiro conjunto de controles, estabelece que a segurança proativa no design é o alicerce de qualquer aplicação segura. Outras categorias críticas incluem V2: Autenticação, V3: Gerenciamento de Sessão, V4: Controle de Acesso, e V5: Validação, Sanitização e Codificação. O ASVS define três níveis de verificação, permitindo adequar o rigor da avaliação ao risco da aplicação. A transição do Nível 1 para o Nível 2, recomendado para a maioria das aplicações com dados sensíveis [9], representa mudança de paradigma: de teste de penetração (postura reativa, caixa-preta) para verificação (postura proativa, caixa-branca, baseada em design), reforçando que a conformidade exige integração da segurança ao ciclo de vida do desenvolvimento.

3 Contextualização do Problema

A integração efetiva de práticas de segurança no ciclo de vida de desenvolvimento de software (SDLC) apresenta desafios que transcendem a simples adoção de padrões técnicos ou princípios de design. Embora ASVS forneça um "o quê", um padrão técnico para verificação, e o Security by Design estabeleça o "como", uma abordagem proativa, persiste uma lacuna em como aplicar ambos no "onde": o processo de desenvolvimento moderno [9].

Nas últimas duas décadas, a indústria de software consolidou a adoção de metodologias ágeis, com Scrum e Kanban tornando-se predominantes [13]. Tais métodos, que priorizam a flexibilidade, a entrega incremental e a rápida resposta a mudanças, alteraram fundamentalmente a forma como os requisitos são gerenciados. No epicentro desse paradigma está a História de Usuário, que se tornou a técnica mais difundida para a elicitação e documentação de requisitos [14]. Este artefato foca em expressar o valor da funcionalidade sob a perspectiva direta do usuário final, constituindo o principal veículo para a definição do que deve ser construído. Apesar de sua eficácia na entrega de valor funcional, a literatura acadêmica, incluindo revisões sistemáticas, identifica um desafio persistente nas metodologias ágeis: a tendência a negligenciar requisitos não funcionais (RNFs) [15]. Como os processos ágeis são otimizados para entregar valor de negócio visível em ciclos curtos , os RNFs (como segurança, desempenho ou manutenibilidade), que são frequentemente "invisíveis" para o usuário final, são rotineiramente despriorizados, negligenciados ou acumulados como "dívida de documentação" [14].

No caso específico da segurança, o problema é mais profundo do que a simples negligência de RNFs; é um conflito fundamental de paradigma entre o artefato ágil e a natureza dos requisitos de segurança. A pesquisa da SAFECode [7] articula claramente essa dicotomia nos seguintes pontos, o primeiro de forma que histórias de usuário são escritas da perspectiva do usuário final e focam em “casos de uso”. Elas descrevem o comportamento esperado e desejado do sistema (ex.: "Como um cliente, eu quero poder ver meu extrato bancário"). Já o segundo ponto, aborda sobre os requisitos de segurança, eles devem ser escritos da perspectiva de um adversário e focam em “casos de abuso” (abuse cases). Eles descrevem o comportamento inesperado e indesejado do sistema (ex.: “Como um invasor, eu não devo conseguir ver o extrato bancário de outro usuário”). O conflito reside no fato de que o ator central da história de usuário (o usuário final) “não tem visibilidade ou participação” [7] nos casos de abuso. É, portanto, conceitualmente difícil, senão impossível, para o *Product Owner* ou o usuário definir exaustivamente os requisitos de segurança usando o formato padrão da história de usuário.

Isso expõe uma lacuna de pesquisa clara e crítica, que é o foco deste artigo. De um lado, a Engenharia de Software Segura (Seção 2.1 e 2.2) exige uma abordagem de *Security by Design*, verificada por padrões técnicos granulares como o OWASP ASVS Nível 1 (Seção 2.3). Do outro lado, o método de desenvolvimento dominante (Ágil) usa um artefato (História de Usuário) que é paradigmaticamente inadequado para capturar requisitos de segurança (casos de abuso) , e as soluções alternativas (Evil Stories) são consideradas impraticáveis. Portanto, permanece o desafio de como traduzir e mapear eficientemente os controles de segurança técnicos, granulares e abrangentes (como os do ASVS) diretamente nos artefatos de requisitos funcionais (as Histórias de Usuário) de uma maneira que seja sistemática, rastreável e que se integre nativamente aos fluxos de trabalho ágeis, sem sobrecarregá-los. Esta lacuna na tradução de controles técnicos para artefatos ágeis é precisamente o problema que a abordagem ASF (Agile Security Framework) busca endereçar.

4. Metodologia

A implementação e a validação da abordagem de desenvolvimento ágil com requisitos de cibersegurança baseiam-se em um método de pesquisa misto, com ênfase quantitativa e qualitativa, com o objetivo de compreender o comportamento de marcação dos analistas na identificação de requisitos de cibersegurança com base no padrão OWASP ASVS.

O procedimento experimental envolveu a participação de quatro analistas distintos, dois analistas com nível de desenvolvimento pleno, um com dois anos de experiência em desenvolvimento de sistemas, o outro com 3. Os outros dois analistas têm nível de desenvolvimento júnior, sendo um deles técnico de informática. Nesse sentido, foi atribuído a estes, o mesmo conjunto de 66 histórias de usuário, provenientes do dataset descrito na Seção 1. Cada analista teve como tarefa identificar os requisitos de segurança aplicáveis a cada história de usuário, de acordo com os critérios definidos pelo ASVS, utilizando a ferramenta ASF, ilustrada na Figura 1. Essa ferramenta foi desenvolvida pelos estudantes do projeto, em *typescript*, e visa, a partir da inserção de histórias de usuários, auxiliar o desenvolvimento de software desde sua parte inicial, para que a programação já seja efetuada visando a segurança.

Para cada história, o analista inicialmente preenchia os campos "As a…", "I want…" e "So that…" com as informações correspondentes e, em seguida, prosseguia para a análise das questões gerais e específicas pertinentes, culminando na marcação dos requisitos individuais do ASVS associados àquela história de usuário, conforme exemplificado na Figura 2.

Figura 1: Software ASF, desenvolvido para realizar a análise de requisitos através das histórias de usuário.

O primeiro parâmetro é feito a partir da análise da questão geral, caso seja contemplada, é feita a análise da questão específica, nisso, se a resposta à questão for sim, é realizada análise de cada requisito e sua marcação. Caso contrário, se a questão geral e a questão específica não forem contempladas, é prosseguido o questionário para as demais questões gerais e as específicas, e assim por diante. Culminando em marcações feitas de requisitos do ASVS que visam aprimorar a ciber-segurança do software que está sendo desenvolvido, pois, a partir do momento que aquele requisito for marcado, torna-se uma tarefa do desenvolvedor trabalhar em prol dele.

Após a etapa de marcação manual pelos analistas, os resultados foram exportados para o formato JSON, contendo cada história de usuário e as marcações realizadas. Esses arquivos foram processados por um script desenvolvido em Python, responsável por extrair, consolidar e organizar as marcações em um formato estruturado. Esse processamento permitiu a geração de relatórios auxiliares, os quais forneceram uma visão abrangente sobre o comportamento das avaliações dos participantes em cada história de usuário. Entre as análises realizadas, destacam-se a verificação da frequência dos requisitos, observando quantas vezes cada item do ASVS foi selecionado nas histórias de usuário, e a distribuição das marcações por analista e por história, conforme se pode observar na Figura 2, o que possibilitou avaliar o grau de consistência e a variabilidade entre as interpretações individuais. Além disso, foi examinada a convergência entre as marcações por questão geral, permitindo identificar padrões de consenso ou divergência entre os analistas, bem como a concorrência de requisitos, a qual analisou a coocorrência de diferentes controles de segurança dentro de uma mesma história. Por fim, realizou-se uma análise por seção do ASVS, mapeando a cobertura temática e a distribuição dos requisitos segundo as áreas de segurança definidas pelo padrão.

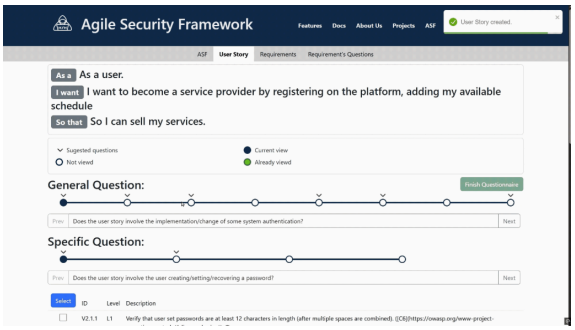

Figura 2: Demonstração da ferramenta ASF

Esses resultados formam a base da análise estatística e interpretativa conduzida neste estudo, permitindo avaliar tanto a consistência interavaliadora quanto a abrangência temática das marcações realizadas. Dessa forma, a metodologia adotada possibilita compreender não apenas a frequência com que determinados requisitos de segurança são aplicados, mas também de que maneira os analistas interpretam, associam e operacionalizam as diretrizes do ASVS no contexto prático da análise de histórias de usuário.

5. Resultados

Esta seção apresenta os resultados obtidos na pesquisa. A Seção 5.1 aborda as frequências de marcação dos requisitos de segurança. A Seção 5.2 analisa a atuação dos analistas por categoria de requisitos, considerando a distribuição e a convergência das marcações. A Seção 5.3 apresenta a coocorrência dos requisitos. Por fim, a Seção 5.4 discute os resultados alcançados.

5.1 Frequência de Requisitos

A análise dos 66 artefatos resultou na identificação de um volume total de 1634 requisitos de segurança individuais. Este volume significativo demonstra a viabilidade da metodologia em extrair requisitos de segurança a partir de histórias de usuário. Nesse sentido, a cobertura da análise foi abrangente, com marcações identificadas em 8 seções distintas do ASVS. Contudo, a distribuição desses requisitos não foi uniforme,o que pode ser

observado na figura 3. A análise de frequência revelou uma concentração massiva de marcações em categorias específicas, a seção V5 - Validação, Sanitização e Codificação foi a mais proeminente, acumulando 51,5% de todas as identificações. Em segundo lugar, a Seção V3 - Gerenciamento de Sessão foi responsável por 16,9% das marcações. Juntas, essas duas seções representam mais de dois terços (68,4%) de todos os requisitos de segurança identificados.

Dentre os requisitos individuais mais recorrentes que impulsionaram essa estatística, destacam-se: V5.1.3: Verifique se toda entrada é validada usando validação positiva, e o V5.3.1: Verifique se o escape de saída sensível ao contexto protege contra XSS refletido. Essa concentração sugere que, na prática, os analistas tendem a focar primariamente em vulnerabilidades clássicas de injeção (como XSS e SQL Injection), que são mitigadas pelas práticas da V5.

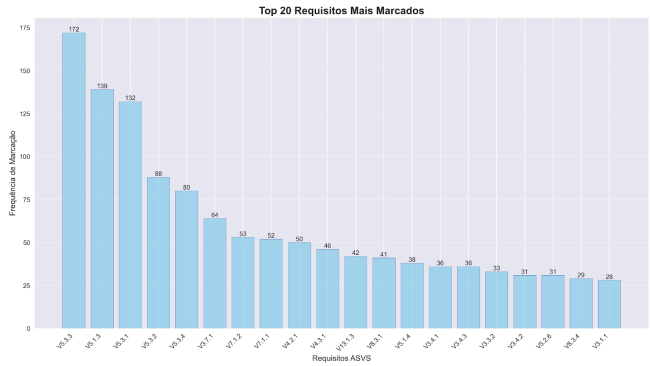

Figura 3: Gráfico com os 20 requisitos mais marcados acerca da análise das histórias.

Para avaliar a consistência e a reprodutibilidade do método, foi analisada a concordância entre os diferentes perfis de analistas. Os resultados indicam um alinhamento satisfatório, mas com pontos de melhoria claros. Observou-se uma concordância parcial, onde 30 das 66 histórias (45,5%) tiveram ao menos um requisito de segurança em comum identificado por todos os analistas. Adicionalmente, em 44 das 66 histórias (66,7%), a variação na quantidade e no tipo de requisitos identificados foi considerada moderada, o que significa requisitos únicos identificados pelos analistas. O principal achado desta análise, no entanto, foi o ponto de atenção qualitativo: a dispersão nas marcações (ou seja, a baixa concordância) estava diretamente relacionada a histórias de usuário que apresentavam ambiguidade técnica. HUs com descrições vagas ou que não detalhavam o fluxo de dados geraram uma variedade muito maior de interpretações de segurança, diminuindo a consistência. Este achado sugere que a clareza e o detalhamento técnico na escrita da história de usuário são fatores críticos e pré-requisitos para uma identificação de requisitos de segurança consistente e alinhada.

5.2 Analistas por Categoria de Requisitos

A análise da distribuição das 1.634 marcações de requisitos revelou uma hierarquia clara de priorização entre as categorias ASVS pelos especialistas. A categoria V5 (Validação, Sanitização e Codificação) dominou significativamente as avaliações, representando 841 marcações (51.5% do total), seguida por V3 (Gerenciamento de Sessão) com 276 marcações (16.9%) e V2 (Autenticação) com 138 marcações (8.4%). Esta distribuição pode ser observada na figura 4, além disso, sugere que os especialistas reconhecem sistematicamente certos domínios de segurança como fundamentais e universalmente aplicáveis. A predominância da categoria V5 pode ser atribuída à sua natureza transversal, requisitos de validação de entrada e codificação de saída são aplicáveis à maioria das funcionalidades web, independentemente da complexidade específica da história de usuário.

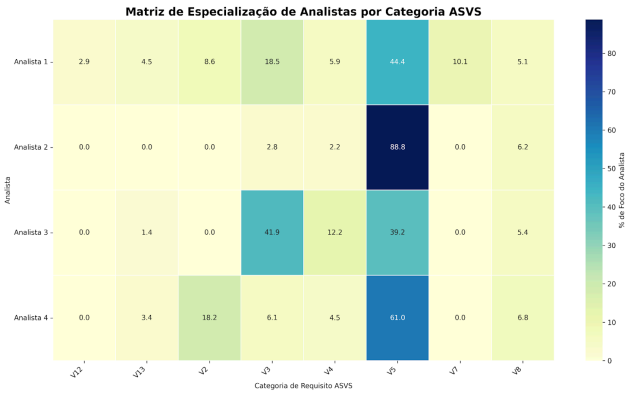

Figura 4: Matriz de especialização dos analistas por categorias ASVS

A análise revelou padrões distintos de especialização entre os analistas, para o analista 1, um perfil abrangente, pois demonstrou cobertura equilibrada entre categorias, com foco em V5 (44.4%), V3 (18.5%) e V7 (10.1%). Único analista a identificar sistematicamente requisitos nas categorias V7 (Tratamento de Erros) e V12 (Arquivos e Recursos), sugerindo expertise em aspectos operacionais e de infraestrutura. Já o analista 2, um perfil mais focado, apresentou uma concentração extrema em V5 (88.8% de suas marcações), com cobertura limitada em outras categorias. Este padrão sugere especialização específica em validação e sanitização, potencialmente refletindo background em desenvolvimento seguro. Nesse sentido, o analista 3 apresentou perfil balanceado, uma vez que a distribuição foi mais equilibrada entre as marcações dos tópicos V3 (41.9%) e V5 (39.2%), com atenção significativa a V4 (Controle de Acesso) (12.2%), o que sugere expertise em aspectos de autenticação e autorização. Por fim, o analista 4 apresentou um perfil moderado, pois apresentou foco principal em V5 (61.0%) complementado por forte atenção a V2 (Autenticação) (18.2%), indicando abordagem que prioriza fundamentos de segurança.

As histórias com maior divergência, apresentando 0% de consenso, revelaram padrões sistemáticos. A História 12, “As a Trainee I want to see which trainings I have signed up for, So that they can schedule their event experience.” por exemplo, demonstrou 31 requisitos únicos identificados, com uma distribuição de (29, 3, 3, 1) entre os analistas. Essa amplitude de 28 requisitos entre o analista mais e menos rigoroso ilustra interpretações completamente divergentes da mesma funcionalidade. De forma similar, a História 13, “As a trainer I want to have a view that is attached to their training node that shows all attendee information Trainers can see all class attendance at first because there are so little Trainers and they are trusted with privacy so that the training coordinator doesn't have to do all the work to get login info for environment setup.” registrou 29 requisitos únicos, com distribuição (29, 5, 3, 1), sugerindo que histórias de maior complexidade aparente tendem a ampliar significativamente as divergências interpretativas. A História 17, “As a AnonymousUser I want to have a training landing page with an intro section and a list of all training offerings and promote call for trainings so that I can shop for training all with one page and we can have a call for trainings for selection.” por sua vez, também obteve 29 requisitos únicos, com distribuição (28, 3, 1, 1), onde a concentração extrema de identificação do Analista 1, em contraste com a identificação mínima pelos demais, evidencia uma lacuna interpretativa fundamental. A análise identificou múltiplos fatores sistemáticos que contribuem para essa baixa convergência, incluindo a ambiguidade léxica em user stories que continham termos como "gerenciar", "sistema" ou "processo", os quais geraram interpretações divergentes sobre o escopo e a complexidade da implementação. Adicionalmente, a ausência de contexto arquitetural, ou seja, a falta de especificações sobre tecnologia, arquitetura e integrações, levou a pressupostos distintos sobre os requisitos aplicáveis. A experiência diferencial entre os perfis de especialização dos analistas também resultou em sensibilidades variadas para categorias específicas de risco. Por fim, a granularidade interpretativa, que se manifestou em divergências sobre o nível apropriado de detalhamento, com alguns analistas aplicando requisitos granulares enquanto outros mantiveram o foco em controles de alto nível, foi outro fator crucial para a disparidade nas avaliações.

5.3 Coocorrência de Requisitos

A análise de frequência (Seção 5.1) demonstrou quais requisitos foram mais comuns, enquanto a análise de coocorrência, apresentada na Figura 5, revela como esses requisitos foram agrupados. Esta análise expõe os padrões de pensamento e os modelos mentais aplicados pelos analistas ao decompor uma HU em controles de segurança. A observação mais saliente da matriz de coocorrência (Figura 5) é a formação de um "cluster" de alta intensidade no quadrante superior esquerdo, composto quase inteiramente por requisitos da Seção V5 (Validação, Sanitização e Codificação). Este achado não apenas reforça a dominância da Seção V5, já identificada na análise de frequência (51,5% das marcações), mas demonstra sua coesão interna. O par com maior coocorrência em todo o estudo foi (V5.1.3, V5.3.3), com 130 instâncias, ambos pertencentes à V5.

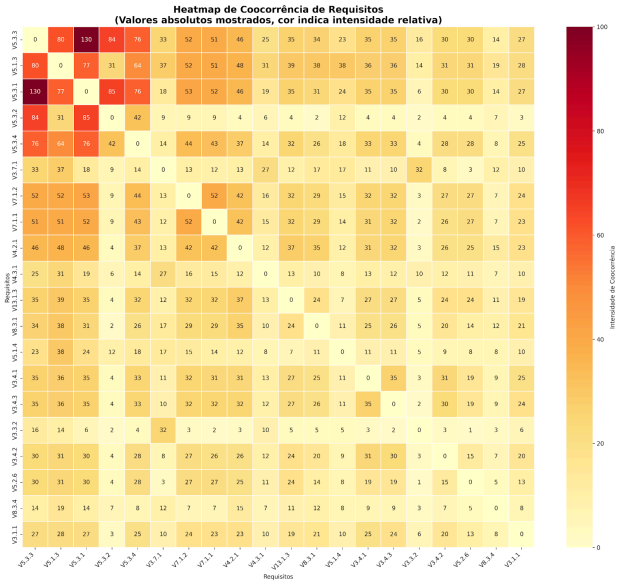

Figura 5: HeatMap contendo os dados das marcações da coocorrência de requisitos.

Este agrupamento dominante não é acidental; ele representa a identificação de um padrão de segurança fundamental e completo: o ciclo de vida dos dados. O cluster V5 combina requisitos de Input Validation (Validação de Entrada), como o V5.1.3 ("validação positiva"), com requisitos de Output Encoding (Codificação de Saída), como o V5.3.1 ("escape de saída sensível ao contexto") e V5.3.3 ("escape contra XSS"). A alta coocorrência entre eles, como o par (V5.1.4, V5.3.1) com 76 marcações, argumenta que os analistas não se limitaram a identificar um único ponto de falha. Em vez disso, eles aplicaram consistentemente o princípio de segurança de "Validar na Entrada e Codificar na Saída" para histórias de usuário que manipulam dados. Além desse cluster principal, a matriz revela padrões secundários que se alinham a funcionalidades específicas. Conforme apontado, requisitos da Seção V2 (Autenticação), como o par (V2.1.1, V2.1.3) (valor de 52 no gráfico), apresentaram coocorrência significativa. Isso indica que HUs focadas em "login" ou "cadastro" disparam um conjunto previsível e padronizado de controles de autenticação. De forma similar, um cluster mais fraco, mas notável, foi observado entre os requisitos de Controle de Acesso (Seção V4).

Em contraste, requisitos com baixa coocorrência geral, como os pertencentes à V8 (Proteção de Dados), atuam como "ilhas". Isso sugere que tais controles não são universais (como o cluster V5), mas sim situacionais, sendo corretamente identificados apenas para HUs que tratam de contextos específicos, como persistência de dados sensíveis ou backups. Portanto, a análise de

coocorrência valida a metodologia ao demonstrar que os analistas não apenas identificaram requisitos, mas o fizeram seguindo padrões lógicos e alinhados às melhores práticas de segurança, agrupando controles de forma consistente por funcionalidade (Autenticação) e por padrão de fluxo de dados (Validação/Codificação).

5.4 Discussão

Os dados apresentados anteriormente indicam uma concentração significativa de marcações nas seções V2 (Autenticação) e V5 (Gerenciamento de Sessão) do ASVS. Isso é esperado, já que muitos sistemas modernos lidam com autenticação de usuários, sessões de login e controle de acesso como componentes essenciais. Em diversas histórias analisadas, funcionalidades como "cadastro de novo usuário", "login seguro", "recuperação de senha" e "manutenção da sessão" estavam diretamente envolvidas, o que naturalmente leva os analistas a marcarem requisitos como V2.1.1 (senhas complexas), V2.2.2 (autenticação multifator), e V5.1.4 (ID de sessão aleatório).

A forte marcação de requisitos relacionados à Q1 (Questão geral 1 : A história de usuário envolve a implementação ou alteração de algum sistema de autenticação?) e Q2 (Questão geral 2 : A história do usuário envolve algum tipo de token?) também confirma esse padrão. Essas questões específicas tratam, respectivamente, da criação/recuperação de senha e da implementação de autenticação. Por exemplo, uma história que obteve alta convergência entre os analistas descrevia uma funcionalidade de redefinição de senha com envio de código por e-mail, o que envolve tanto autenticação quanto recuperação segura, justificando a marcação de requisitos da seção V2 e da questão Q1.

Por outro lado, a baixa marcação de requisitos relacionados à Q13 (analisadores XML), Q15 (análise de JSON) e Q12 (serialização de objetos) sugere que as histórias apresentadas não abordavam diretamente o processamento de dados estruturados ou a integração com sistemas externos via XML/JSON, ou que os analistas não identificaram claramente esses elementos como críticos em termos de segurança. Em histórias com menor convergência (como a História 33 : As a AnonymousUser I want to read the blog so that I can stay informed about all the latest happenings with BADCamp.), observou-se que os analistas divergem sobre se certas entradas do usuário implicam riscos de injeção ou manipulação. Isso sugere que alguns requisitos, especialmente os relacionados à entrada, saída ou armazenamento de dados, são mais abertos à interpretação, dependendo do nível de detalhe da história ou do conhecimento técnico do analista.

A ocorrência frequente entre pares como V2.1.1 e V2.2.2 reforça que a autenticação multifator é frequentemente considerada como uma extensão ou reforço dos controles de senha. Já os pares V5.1.4 e V5.3.1 indicam que,ao identificar qualquer uso de sessão, os analistas tendem a marcar também requisitos adicionais de segurança da sessão (como proteção contra fixação).

Para além dos agrupamentos por questões e temas, uma análise focada no nível mais granular, os requisitos individuais, revela padrões importantes de interpretação por parte dos analistas. O requisito V2.1.1 (complexidade mínima para senhas) foi o mais marcado. Isso se explica pelo fato de diversas histórias envolverem o cadastro ou a autenticação de usuários. Nesse sentido, o requisito V2.2.2 que trata de autenticação multifator (MFA), também se destacou. Em histórias onde o usuário acessava informações sensíveis ou realizava ações administrativas (como alteração de dados cadastrais ou aprovações), os analistas indicaram a necessidade de múltiplos fatores como reforço de segurança. Em muitos casos, mesmo sem a presença explícita do MFA na história, os analistas informam sua necessidade com base na criticidade da ação descrita. Outro requisito recorrente foi V5.1.4, referente à aleatoriedade e unicidade de identificadores de sessão. Sempre que uma funcionalidade de login ou permanência do usuário logado era mencionada, os analistas tendiam a marcá-la, demonstrando preocupação com sequestro ou reutilização de sessão. Já requisitos como V13.4.2 (validação segura de XML) ou V12.3.1 (controle de serialização de objetos) foram muito pouco marcados. Isso se deve ao fato de que a maioria das histórias avaliadas abordava funcionalidades voltadas a interações diretas com o usuário e não aspectos internos de comunicação entre sistemas, em que esses requisitos ganham relevância.

A interpretação dos requisitos pelos analistas também variou conforme a clareza das histórias. Quando o texto era vago ou deixava margem para diferentes entendimentos técnicos, como na História 33, a diversidade de marcações aumentava. Isso mostra que, mesmo com um framework como o ASVS, a clareza da história influencia fortemente a identificação dos requisitos de segurança aplicáveis.

5.5 Ameaças à Validade

Como ameaças internas ao estudo realizado, cabe ressaltar a ambiguidade técnica e a falta de detalhamento das histórias de usuário, que atuaram como fatores de confusão, pois descrições vagas ou a ausência de fluxos de dados detalhados geraram uma variedade muito maior de interpretações de segurança, diminuindo a consistência dos resultados. Além disso, a experiência e os perfis de especialização distintos dos analistas resultaram em sensibilidades variadas para categorias específicas de risco, o que influenciou diretamente a quantidade e o tipo de requisitos identificados para a mesma funcionalidade. No que tange às ameaças de constructo, que avaliam a conexão entre a teoria e as observações práticas, destaca-se o conflito de paradigma entre o formato das histórias de usuário e a natureza dos requisitos de segurança. Enquanto a teoria exige uma abordagem de *Security by Design* focada em casos de abuso, as histórias de usuário são tradicionalmente escritas sob a ótica do valor para o usuário final, o que torna conceitualmente difícil a definição exaustiva de requisitos de segurança usando esse formato padrão. A ausência de um contexto arquitetural claro nas histórias também forçou os analistas a basearem suas marcações em pressupostos subjetivos sobre tecnologias e integrações, distanciando a observação da

realidade técnica do sistema. Quanto às ameaças de conclusão, relacionadas à confiabilidade da análise estatística, são evidenciadas pela baixa convergência interavaliadora encontrada. O estudo revelou que apenas 45,5% das histórias tiveram ao menos um requisito em comum identificado por todos os analistas, apontando uma dispersão significativa nas marcações. Além disso, a concentração massiva de dados em categorias específicas (como a V5, que acumulou 51,5% das marcações) e a subnotificação de requisitos mais granulares, como processamento de XML ou JSON, sugerem que a análise pode ter sido limitada pela percepção superficial de riscos clássicos, como injeção, em detrimento de vulnerabilidades mais complexas. Por fim, como ameaças externas ao estudo realizado, que tratam da capacidade de generalização do estudo, estas residem na especificidade do conjunto de dados e do domínio analisado. O estudo utilizou um subconjunto de histórias de usuário relacionadas ao site de uma conferência específica (BADCamp). Essa natureza do domínio, focada em interações diretas com usuários, pode não refletir as necessidades de segurança de sistemas de infraestrutura crítica ou outros tipos de aplicações específicas, limitando a aplicação dos achados a contextos de desenvolvimento de software mais genéricos.

6. Considerações Finais

Este trabalho avaliou a ferramenta ASF, baseada no framework ASVS, para identificar requisitos de cibersegurança em Histórias de Usuário (HUs) ágeis. A análise de 1634 requisitos relacionados a 66 histórias de usuários demonstrou que a abordagem é viável e robusta para integrar a segurança no início do SDLC. Os analistas aplicaram o framework de forma coerente, identificando "clusters" lógicos de controles (como V2-Autenticação e V5-Validação), conforme detalhado nas Seções 5.1 e 5.3. No entanto, a eficácia do método provou ser diretamente moderada pela qualidade da HU. O principal argumento derivado do estudo (Seções 5.2 e 6) é a dicotomia entre a alta concordância em HUs funcionalmente explícitas (30/66) e a alta dispersão em HUs tecnicamente ambíguas (ex.: História 33). Isso explica por que riscos complexos, como serialização (Q12) ou processamento de JSON/XML (Q13, Q15), foram subnotificados. O gargalo para uma segurança abrangente não reside na ferramenta ASF ou no ASVS, mas na falta de contexto técnico da própria HU. Em termos organizacionais, o ASF atua como uma ferramenta estratégica de contenção de despesas. Ao permitir a identificação de falhas de design precocemente, ele "poupa" tempo e o custo exponencialmente maior de remediar vulnerabilidades em produção, mitigando o débito técnico. Por conseguinte, este estudo valida o ASF como um mecanismo prático para fortalecer a cultura de cibersegurança. Ele reforça a necessidade de melhorar a clareza textual e o detalhamento técnico das HUs, sugerindo avanços como a padronização e o uso de checklists para maximizar a eficácia da análise de segurança desde a concepção.

Agradecimentos

O presente trabalho foi realizado com o apoio do Programa de Apoio à Iniciação Científica e Tecnológica da Fundação de Amparo à Pesquisa do Estado de Minas Gerais (PIBIC-FAPEMIG) e do Laboratório de Iniciação Científica e Extensão da Computação (LINCE) do Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG), campus Leopoldina.

REFERÊNCIAS

[1] SOMMERVILLE, Ian. Engenharia de software. 10. ed. São Paulo: Pearson, 2021.
[2] BRASIL. Lei nº 13.709, de 14 de agosto de 2018. Dispõe sobre a proteção de dados pessoais e altera a Lei nº 12.965, de 23 de abril de 2014 (Lei Geral de Proteção de Dados Pessoais – LGPD). Diário Oficial da União, Brasília, DF, 15 ago. 2018.
[3] BRASIL. Decreto nº 9.637, de 26 de dezembro de 2018. Institui a Política Nacional de Segurança da Informação. Diário Oficial da União, Brasília, DF, 27 dez. 2018.
[4] BRASIL. Decreto nº 11.856, de 26 de dezembro de 2023. Institui a Política Nacional de Cibersegurança. Diário Oficial da União, Brasília, DF, 27 dez. 2023.
[5] BRASIL. Lei nº 12.527, de 18 de novembro de 2011. Regula o acesso a informações previsto na Constituição Federal (Lei de Acesso à Informação – LAI). Diário Oficial da União, Brasília, DF, 18 nov. 2011.
[6] UNIÃO EUROPEIA. Regulamento (UE) 2016/679 do Parlamento Europeu e do Conselho, de 27 de abril de 2016. Relativo à proteção das pessoas singulares no que diz respeito ao tratamento de dados pessoais e à livre circulação desses dados (Regulamento Geral sobre a Proteção de Dados – GDPR). Jornal Oficial da União Europeia, L 119, 4 maio 2016.
[7] SAFECode. Fundamental practices for secure software development: a guide for agile development & security. [S.l.]: SAFECode, 2012. Disponível em: https://safecode.org/publication/SAFECode_Agile_Dev_Security0712.pdf. Acesso em: 5 nov. 2025.
[8] OPEN WEB APPLICATION SECURITY PROJECT (OWASP). About OWASP. [S.l.], c2024. Disponível em: https://owasp.org/about/. Acesso em: 5 nov. 2025.
[9] OPEN WEB APPLICATION SECURITY PROJECT (OWASP). OWASP Application Security Verification Standard – ASVS 4.0.3. [S.l.]: OWASP Foundation, 2021. Disponível em: https://owasp.org/www-project-asvs/. Acesso em: 5 nov. 2025.
[10] SHOSTACK, Adam. Threat modeling: designing for security. Indianapolis: Wiley, 2014.
[11] PFLEEGER, Charles P.; PFLEEGER, Shari Lawrence. Security in computing. 5. ed. Boston: Pearson, 2015.
[12] NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (NIST). SP 800-30 Rev. 1: Guide for conducting risk assessments. Gaithersburg, MD: U.S. Department of Commerce, 2012.
[13] DIGITAL.AI. 17th annual state of agile report. [S.l.]: Digital.ai, 2024. Disponível em: https://digital.ai/state-of-agile-report. Acesso em: 3 nov. 2025.
[14] COHN, Mike. User stories applied: for agile software development. Boston: Addison-Wesley, 2004.
[15] SILVA, A. et al. Conceptual modeling versus user story mapping: which is the best approach to agile requirements engineering? [S.l.], 2021. Disponível em: https://www.researchgate.net/publication/351400926. Acesso em: 5 nov. 2025.