Pós-Graduação em Ciência da Computação

“Aegis: Um Modelo de Proteção à Dados Sensíveis em Ambientes Client-side"

Por

Carlo Marcelo Revoredo da Silva

Dissertação de Mestrado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, FEVEREIRO/2014

Universidade Federal de Pernambuco
Centro de Informática
Pós-graduação em Ciência da Computação

Carlo Marcelo Revoredo da Silva

"Aegis: Um Modelo de Proteção à Dados Sensíveis em Ambientes Client-side"

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: *Vinicius Cardoso Garcia*

RECIFE, FEVEREIRO/2014

Catalogação na fonte
Bibliotecário Jefferson Luiz Alves Nazareno, CRB 4-1758

Silva, Carlo Marcelo Revoredo da.
Aegis: um modelo de proteção à dados sensíveis em ambientes client-side. – Recife: O Autor, 2014.
130f. : fig., tab.

Orientador: Vinicius Cardoso Garcia.
Dissertação (Mestrado) - Universidade Federal de Pernambuco. Cin. Ciência da computação, 2014.
Inclui referências e apêndice.

1. Ciência da computação . 2. Segurança da informação. I. Garcia, Vinicius Cardoso. (orientador). II. Título.

004 (22. ed.) MEI 2014-63

Dissertação de Mestrado apresentada por **Carlo Marcelo Revoredo da Silva** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título "**Aegis: Um Modelo de Proteção à Dados Sensíveis em Ambientes Client-side**" orientada pelo **Prof. Vinicius Cardoso Garcia** e aprovada pela Banca Examinadora formada pelos professores:

_____

Prof. Ruy José Guerra Barretto de Queiroz
Centro de Informática / UFPE

_____

Prof. Rodrigo Elia Assad
Departamento de Estatística e Informática / UFRPE

_____

Prof. Vinicius Cardoso Garcia
Centro de Informática / UFPE

Visto e permitida a impressão.
Recife, 28 de fevereiro de 2014.

_____

Profa. Edna Natividade da Silva Barros
Coordenadora da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

A minha mãe, Arimar Revoredo da Silva,
Ao meu pai, Jose Oliveira da Silva,
Ao meu irmão, Adrianno Marcio Revoredo da Silva,
A minha namorada Adriana da Silva Freitas.

Agradecimentos

Agradeço...

- Em primeiro lugar a Deus por sua graça e pelas bênçãos derramadas constantemente sobre mim de maneira maravilhosa;
- A minha família, meu pai, meu irmão, e especialmente minha mãe, esta por ser a maior incentivadora e por fornecer amor e suporte emocional imprescindível em todos os momentos de minha vida;
- A Adriana, por ser minha companheira fiel, me apoiando com seu carinho e cuidado e por estar ao meu lado mesmo nas horas mais difíceis.
- Ao professor Vinicius Garcia pela paciência e dedicação na orientação desta pesquisa.
- Aos professores Ruy de Queiroz e Rodrigo Assad por terem aceitado participar de minha banca.
- A todos os meus amigos e colegas que contribuíram direto e indiretamente para a realização deste sonho;

Muito Obrigado!

"E aqueles que foram vistos dançando foram julgados insanos por aqueles que não podiam escutar a música."

—NIETZSCHE (Friedrich Nietzsche)

Resumo

Os Navegadores Web, do inglês, Web Browsers, são ferramentas de extrema importância no que diz respeito ao consumo de informações na internet, consequentemente são os softwares dos quais os usuários utilizam a maior parte do tempo, sejam domésticos ou corporativos. Com o passar dos anos, a forma do que estava sendo publicado na web foi evoluindo, o que antes era estático e apenas leitura, agora se apresenta como um cenário onde os internautas e serviços interagem, compartilhando informações em uma grande variedade de cenários, como e-mail, redes sociais e comércio eletrônico, tornando-se então em um ambiente dinâmico e colaborativo.

Em meio ao advento da web, foram surgindo possibilidades para prestadores de serviços especializados, como bancos ou fontes pagadoras de transações comerciais, dos quais através da internet encontraram oportunidades para disponibilizarem seus serviços com alta disponibilidade, possibilitando que seus clientes consigam utilizar seus serviços através de um browser. O resultado disso é uma série de benefícios, como redução de filas, interatividade, foco no domínio, que podemos traduzir em uma palavra: comodidade.

Não obstante, em consequência ao seu advento, a web se apresenta como um ambiente cada vez mais hostil, onde pessoas mal-intencionadas executam crimes cibernéticos, dos quais visam roubar ou adulterar informações sigilosas dos usuários da web, violando os dados do usuários. Essas atividade ilícitas muitas vezes são executadas através de ataques que exploram fatores como Engenharia Social, Falhas nas Aplicações Web ou Vulnerabilidades no Browser do usuário. Estes vetores de ataques apresentam-se em diversas formas: capturando entradas dos usuários, forjando sites oficiais, injeção de scripts maliciosos, propagação de Malwares, entre outros. Com o dinamismo da web, o processo de controle destes vetores de ataque torna-se cada vez mais difícil, e como a tendência da web é manter-se em constante evolução, estas ameaças tornam-se uma preocupação contínua.

Nesta pesquisa é proposta uma ferramenta, intitulada Aegis, como modelo de proteção para interagir diretamente no browser do usuário, buscando minimizar os ataques mencionados através de uma solução puramente cliente-side. Para isso, foram realizados estudos e revisão na literatura sobre o tema, buscando uma justificativa bem fundamenta para as decisões e técnicas aplicadas. Adicionalmente, são apresentados os resultados obtidos na avaliação da Aegis em conjunto com demais ferramentas relacionadas.

Palavras-chave: Segurança da Informação, Vulnerabilidades em Aplicações Web, Vulnerabilidades em Navegadores Web, Proteção à Fraudes.

Abstract

Web Browsers are extremely important tools with respect to the use of information on the Internet, are therefore the software which users utilize most of the time, whether domestic or corporate. Over the years, the way of what was being published in web has evolved, what was once static, read-only, now presents itself as a scenario where Internet users and services interact by sharing information on a wide variety of scenarios, such as email, social networking and e-commerce , thus becoming a dynamic and collaborative environment.

Amid the advent of web, possibilities for specialized service providers have emerged as banks or payers of commercial transactions, including over the internet found opportunities to make available their services with high availability , enabling its customers to be able to use their services through a browser.

The result is a series of benefits such as shorter queues, speed, interactivity , focus in the field, we can translate a word: convenience. Nevertheless, due to its advent, the website is presented as an environment every once hostile, where people perform malicious cyber crimes, the which aim to steal or tamper with sensitive information of the web users, violating data users. These illegal activities are often implemented through attacks that exploit factors such as Social Engineering, Web Applications Flaws or Vulnerabilities in the browser user. These attack vectors are shown in several ways: capturing input from users, forging official websites, injection malicious scripts, malware propagation, among others. With the dynamism of the web , the control process of these attack vectors becomes increasingly difficult, and how the trend of the web is to keep evolving, these threats become one ongoing concern.

In this research we propose a tool, called Aegis, a protection model to interact directly in the user's browser in order to minimize the attacks mentioned purely by a client -side solution. To this were done studies and literature review on the topic, seeking a justification well founded for decisions and techniques applied. Additionally, the results are presented obtained in the evaluation of Aegis in conjunction with other related tools.

Keywords: *Information Security, Vulnerabilities in Web Applications, Vulnerabilities in Web Browsers, Fraud Protection.*

Sumário

1 Introdução **12**

1.1 Justificativa e Motivação 13
1.2 Objetivos 15
1.3 Fora do Escopo 15
1.4 Metodologia 16
1.5 Estrutura da Dissertação 17

2 Contextualização da Proposta **18**

2.1 Metodologia de Pesquisa 19
2.2 Definição de Pesquisa e Fonte Primária 20
2.3 Critérios de Inclusão 21
2.4 Critério de Exclusão 21
2.5 Critério de Relevância 22
2.6 Triagem 22
2.7 Classificação dos Resultados 23
2.8 Análise e discussão dos Resultados 24
2.8.1 Resultado Obtido da QP 1 24
Vetor 1: Injeção de Códigos 25
Vetor 2: Fraudes 25
Vetor 3: Fraca Proteção aos Dados 26
Vetor 4: Componentes ou Complementos Inseguros 26
Vetor 5: Ataques de Domínio Cruzado 27
2.8.2 Resultado Obtido da QP 2 27
Ataque 1: Cross-Site Scripting (XSS) - A1 28
Ataque 2: Injeção SQL (SQLi) - A2 33
Ataque 3: Clickjacking - A3 34
Ataque 4: Redirecionamentos e Encaminhamentos Inválidos - A4 36
Ataque 5: Complementos Maliciosos - A5 39
Ataque 6: Exposição de Dados Sensíveis - A6 41
Ataque 7: Utilização de Componentes Vulneráveis Conhecidos - A7 43
Ataque 8: Cross-Site Request Forgery (CSRF) - A8 45
Ataque 9: Falhas nas Politicas de Mesma Origem - A9 46

Considerações Finais 48
2.8.3 Resultado Obtido da QP 3 49
Principais Grupos de Pesquisa nas fontes acadêmicas. 50
Trabalhos Relacionados. 51
Ferramentas de proteção integradas ao navegador 52
2.9 Considerações Finais 54

3 Implementação da Proposta **55**
3.1 Decisões de Projeto 55
3.1.1 Publico alvo e cenário de atuação 55
3.1.2 Tipo de ferramenta a ser desenvolvida 56
3.2 Arquitetura do Aegis 57
3.2.1 A Extensão Aegis 57
Nível de Confiança 58
Listas 59
Scripts 59
Manipulação das Requisições 59
Manipulação de Dados Sensíveis (A6) 61
Manipulação de Complementos (A7) 62
Manipulação de Dados em Transferência (A5) 62
Notificações ao Usuário 62
Operações Internas 62
3.2.2 Arquitetura do Serviço do Aegis 63
Biblioteca Própria 63
Bibliotecas Open Source 63
Serviços Externos 64
3.3 Funcionalidade dos Filtros 65
3.3.1 Filtro A1: Ataques baseados em A1 66
3.3.2 Filtro A2: Ataques baseados em A2 67
3.3.3 Filtro A3: Ataques baseados em A3 69
3.3.4 Filtro A4: Ataques baseados em A4 72
3.3.5 Filtro A5: Ataques baseados em A5 73
3.3.6 Filtro A6: Ataques baseados em A6 74
3.3.7 Filtro A7: Ataques baseados em A7 78
3.3.8 Filtro A8: Ataques baseados em A8 79
3.3.9 Filtro A9: Ataques baseados em em A9 81

3.4 Limitações e Considerações Finais 82

4 Avaliação da Proposta **83**
4.1 Metodologia 83
4.1.1 Planejamento 84
4.1.2 Definição 85
4.1.3 Coleta de Dados 93
4.1.4 Análise dos Resultados 95
Resultado da Questão 1 96
Resultado da Questão 2 97
Resultado da Questão 3 97
Resultado da Questão 4 98
Resultado da Questão 5 99
Resultado da Questão 6 99
Resultado da Questão 7 100
Resultado da Questão 8 100
Resultado da Questão 9 101
4.2 Limitações da Avaliação 101
4.3 Considerações Finais 102

5 Conclusões e Trabalhos Futuros **103**
5.1 Contribuições 104
5.2 Limitações da Proposta 105
5.3 Trabalhos Futuros 106

Referências **107**

A Apêndice **144**

1 Introdução

Segundo a pesquisa da Mashable Mas (2013) ilustrada na Figura 1.1, em 1992 a web era utilizada quase que exclusivamente por usuários norte-americanos, onde chegavam a representar 66% destes usuários que navegavam na rede mundial de computadores. Contudo, a web nos dias atuais está de fato considerada globalizada, um bom exemplo disso é que no mesmo estudo, foi identificado que a partir do ano de 2012, os usuários de outras partes do mundo já representavam 87% de todo o trafego da web, fazendo a hegemonia norte-americana despencar para 13%.

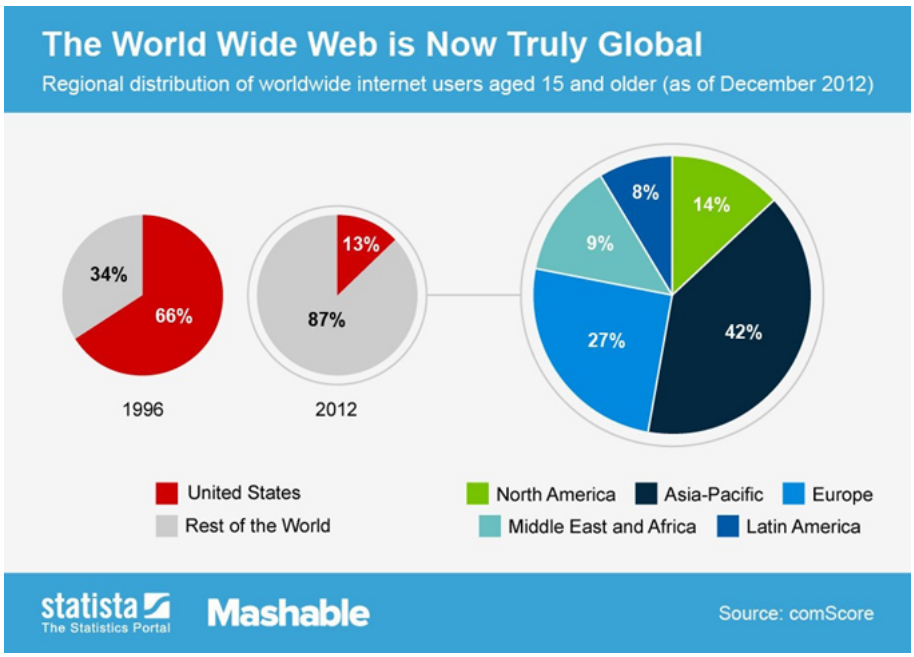

Figura 1.1 Pesquisa realizada pela Mashable em 2013.

Como justificativa deste crescimento é a evolução que a web e seus meios de navegação sofreram nos últimos anos. Neste período a Internet ganhou força e impulsionou

o surgimento de mecanismos que proporcionavam usabilidade durante a navegação na web. Estas ferramentas ficaram conhecidas como Navegadores Web, e em pouco tempo tornaram-se de extrema importância no que diz respeito ao consumo de informações na Internet, atualmente são os softwares dos quais os usuários utilizam a maior parte do tempo, sejam domésticos ou corporativos. Outra consequência dessa evolução é o comportamento e heterogeneidade do conteúdo trafegado na web. Antes as aplicações web eram estáticas, traziam quase sempre apenas textos e havia pouca transição entre páginas, além disso, a interação do usuário com a aplicação era quase que nula. Atualmente as aplicações proporcionam um ambiente participativo e colaborativo de informações trafegadas, apresentando seu conteúdo em diversos tipos e formatos de arquivos multimidia O'Reilly (2005).

Durante este processo evolutivo, foram surgindo possibilidades para prestadores de serviços especializados, como bancos ou fontes pagadoras de transações comerciais, os quais através da internet encontraram oportunidades para disponibilizarem seus serviços com alta disponibilidade, possibilitando que seus clientes consigam utilizar seus serviços com praticidade, sendo necessário apenas um navegador web Hertzum *et al.* (2004). O resultado disso é uma série de benefícios, como o fortalecimento do e-commerce, redução de filas, agilidade, interatividade, foco no domínio, que podemos traduzir em uma palavra: comodidade.

1.1 Justificativa e Motivação

Apesar das inúmeras possibilidades de uso, institutos como o Centro de Estudos para Resposta e Tratamento de Incidentes, CERT, alertam da necessidade de cuidados a serem tomados durante a navegação na Internet, em especial aos vetores de ataques envolvidos na utilização dos navegadores web CER (2008). Com tantos valores agregados proporcionados pela comodidade da Internet, operações como compras online, Internet Banking e compartilhamento de informações pessoais tornaram-se populares, sendo executadas durante o dia a dia de milhares de pessoas. Consequentemente, o dado trafegado começa a conter informações sensíveis, representando assim maior valor de negócio, a partir disso as preocupações quanto a uma proteção eficaz destes dados começa a receber crescimento.

O advento da web trás oportunidades também para pessoas mal-intencionadas, que executam crimes cibernéticos, dos quais visam roubar ou adulterar informações sigilosas, comprometendo a segurança da informação (SI) dos usuários da web, ou seja, violando

a Conformidade, Integridade e Disponibilidade (CID) de seus dados. Estes ataques apresentam-se em diversas formas: capturando entradas dos usuários, forjando sites oficiais, injeção de scripts maliciosos, exploração de vulnerabilidades nos navegadores, propagação de softwares maliciosos, entre outros. Com o dinamismo da web, um sólido controle destes ataques torna-se cada vez mais difícil, e como a tendência da web é manter-se em constante evolução, estas ameaças tornam-se uma preocupação contínua TRENDMicro (2011). Os ataques são executados por agentes mal intencionados e projetados através de vetores, que são os métodos pelos quais os agentes os executam, e a consequência do sucesso de um ataque é a violação dos dados do usuário.

Como a maioria dos conteúdos da web são direcionados aos navegadores, um dos vetores mais comuns é a exploração por vulnerabilidades contidas nestas aplicações. No decorrer dos anos surgiu um crescente interesse na exploração e combate destas vulnerabilidades, como mostram algumas competições focadas no descobrimento de falhas nos navegadores mais populares da atualidade G1 (2009), com casos alarmantes como o comprometimento do sistema operacional Mac (2009)IDG (2009).

Outro vetor muito explorado é o qual o agente utiliza engenharia social para propagar um esquema ilícito, conhecido como fraude, com objetivo de obter informações do usuário, como dados ou até mesmo cliques, sem a devida autorização CER (2012b). Um dos meios mais comuns de aplicar uma fraude é através de programas maliciosos, denominado de Malwares CER (2012a). Outra possibilidade de fraude é através de Phishing, prática bastante explorada em todo o mundo Greg Aaron (2012).

Além disso, muitas vezes por falta de uma sólida politica de desenvolvimento seguro, alguns sites ou serviços são disponibilizados na web apresentando falhas de segurança OWASP (2013g). Essa negligencia dar origem a uma série de vetores, dentre os quais possibilita que um mal intencionado explore vulnerabilidades existentes na aplicação.

O objetivo deste estudo trata da elaboração de um mecanismo que auxilie o navegador a oferecer maior segurança na navegação do usuário, visando minimizar os ataques relacionados aos principais vetores apresentados. A ferramenta, intitulada Aegis, irá monitorar as requisições e respostas produzidas no navegador, a fim de analisar diversos aspectos com o intuito de detectar atividades maliciosas na requisição ou vulnerabilidade existente na requisição, removendo comportamentos suspeitos ou restringindo recursos do navegador, como leitura de mídias, exibição de imagens, execução de scripts ou plug-ins, tudo em tempo de execução e conforme a confiabilidade do site em questão, o objetivo é impedir que o navegador torne-se suscetível a ataques que comprometam a integridade dos dados trafegados.

A elaboração deste projeto é centrada em uma revisão sistemática da literatura sobre os ataques relacionados, bem como uma fundamentação extraída de principais trabalhos sobre o tema, que envolvem diversos pesquisadores e instituições.

1.2 Objetivos

Este estudo tem como objetivo propor o desenvolvimento de uma ferramenta capaz de proteger o navegador web das principais ameaças que proporcionam violação aos dados.

- Apresentar os navegadores mais populares da atualidade, a história e evolução dos mesmos e as tecnologias envolvidas, sobre a ótica da navegação segura.
- Realizar uma revisão sistemática sobre as questões de segurança e estado atual nas principais fontes literárias.
- Propor uma taxonomia dos principais ataques web direcionados aos ambientes Client-side.
- Discutir o estado atual das ameaças com base nos resultados obtidos na literatura, direcionando o estudo baseado nas principais fontes literárias do tema seja academia ou indústria.
- Desenvolver e disponibilizar um protótipo da ferramenta que possa interagir com o navegador em tempo real.
- Analisar o comportamento da ferramenta proposta sendo executada nos ambientes controlados disponíveis na atualidade.
- Desenvolver e disponibilizar um ambiente controlado, com o objetivo de uma validação mais completa e voltado para a ótica do tema proposto.
- Discutir os resultados obtidos na validação da proposta.

1.3 Fora do Escopo

Alguns assuntos foram excluídos do escopo de pesquisa desta dissertação, a saber:

- **Soluções que envolvam melhores praticas de implementação:** O foco do problema está direcionado na segurança que os navegadores atuais oferecem aos seus

usuários, ou seja, a proposta desta dissertação não pretende apresentar boas práticas no desenvolvimento de aplicações, mas sim proteger o usuário de ameaças que dentre as quais algumas exploram falhas nestes cenários.

- **Soluções que envolvam consequências diferentes de uma violação de dados:** Na pesquisa desta dissertação, os objetivos estão focados em minimizar ameaças que violem os dados do usuário de navegador web, ou seja, a integridade da mesma, bem como a confiabilidade do site ou aplicação em questão que provê um determinado serviço. Consequências que envolvam disponibilidade não estão no escopo deste estudo, pelo fato das mesmas não impactarem diretamente aos dados de um respectivo usuário, mas sim a operabilidade de um determinado serviço.

1.4 Metodologia

Para atender os objetivos propostos, será adotada uma metodologia que consiste em uma serie de passos a serem executados.

O primeiro passo será realizar uma revisão sistemática na literatura sobre o tema que apresenta relação com a proposta deste projeto. A revisão deverá ser aplicada em bibliotecas ou repositórios de publicações, e devem ser executadas através de palavras chaves bem definidas. O objetivo será analisar as principais ameaças que estão relacionadas ao problema abordado no tema, e consequentemente mapear as soluções apresentadas até hoje e mitigar possíveis lacunas.

Com todo o conhecimento de pesquisa já realizada sobre este tema revisado, o segundo passo será partir para as especificações da solução. A ferramenta proposta neste estudo denominada Aegis deverá ser acoplada aos navegadores considerados mais populares da atualidade. Também deverão ser avaliadas outras ferramentas que possuam propósitos similares ao do Aegis, que através desta avalição deverá ser gerado um relatório com as características e peculiaridades encontradas durante a investigação e tal relatório servirá para uso durante o desenvolvimento deste projeto e na escrita desta dissertação.

No terceiro passo serão atividades de definição da arquitetura que modela a ferramenta Aegis, para isso serão realizadas analises e estudos, bem como validações para que se possa fomentar a viabilidade de desenvolvimento da ferramenta. Este processo de decisão arquitetural deverá observar o ambiente de execução da ferramenta, ou seja, uma ferramenta que interaja com o navegador e proporcione uma experiência com o usuário.

No quarto passo será descrito o desenvolvimento de um protótipo funcional da ferramenta proposta, e para isso serão utilizadas ferramentas para apoio ao desenvolvimento

em cada um dos navegadores mais populares.

Para que os resultados sejam o mais próximo da realidade, será realizada uma validação da ferramenta proposta. Porém, devido aos inúmeros cenários propensos a violação de dados, bem como as consequências da exposição de dados sensíveis, a execução em ambientes reais extrapolaria o cronograma da pesquisa. Portanto foi definido que a ferramenta será avaliada em ambientes controlados que reproduzem ambientes reais com vulnerabilidades existentes. A validação será realizada no quinto passo, que se caracteriza em uma analise dos principais ambientes controlados atuais que se encontram disponíveis de forma gratuita e de código aberto, e que tenha apoio ou tutela de alguma organização de grande relevância ao tema.

No sexto passo, baseando-se na ótica das principais ameaças obtidas no primeiro passo, este trabalho tem o intuito de adicionar novas funcionalidades para que o respectivo ambiente cubra as ameaças relacionadas ao tema que estavam até então ausentes na reprodução do seu cenário.

As conclusões, bem como variações e comparações entre os resultados da experimentação serão descritas no passo sete, trazendo um comparativo entre as ferramentas mitigadas no segundo passo, em comparação com o Aegis.

1.5 Estrutura da Dissertação

Neste Capítulo de introdução foi apresentada uma breve fundamentação, levantando problemas que justificam e motivam a execução de tal pesquisa, bem como apresentando os objetivos da mesma. O resto deste trabalho está dividido nos seguintes capítulos:

- **Capítulo 2:** apresenta uma contextualização da pesquisa que contribuiu para aderir embasamento no desenvolvimento da ferramenta proposta.
- **Capítulo 3:** apresenta a implementação e estrutura da ferramenta proposta intitulada Aegis.
- **Capítulo 4:** apresenta o ambiente de validação da ferramenta, relatando resultados detalhes da execução, resultados e adequação do mesmo aos vetores abordados nesta pesquisa.
- **Capítulo 5:** apresenta a análise de resultados obtidos com a execução deste estudo e desfeche conclusivo da pesquisa apontando possíveis trabalhos futuros.

2

Contextualização da Proposta

Nesta pesquisa é apresentada uma taxonomia para as principais ameaças, vetores e ataques mais atuantes em ambientes Client-side, dos quais proporcionam a violação de dados sensíveis. Primeiramente é importante conceituar alguns termos. Segundo Allen (2001) a violação é o roubo ou perda acidental de dados sensíveis. Já um dado sensível é qualquer dado que possui seu acesso restrito às pessoas autorizadas. E segundo os estudos publicados pelas entidades SANS/MITRE SANS/MITRE (2011), um vetor de ataque é um método do qual um usuário malicioso utiliza para realizar um ataque à uma determinada aplicação ou serviço web. Já um ataque na web é qualquer ação realizada por uma aplicação ou pessoa, seja de forma proposital (maliciosa) ou involuntária (através de vulnerabilidades), que resulta em algum tipo de Ameaça. E por fim, uma Ameaça é qualquer evento, circunstancia ou pessoa que possa causar danos a uma determinada pessoa ou aplicação, a exemplo da violação de dados sensíveis.

Este capítulo visa apresentar uma taxonomia que auxilie na didática e classificação das principais ameaças neste cenário. O primeiro passo foi realizar uma revisão sistemática na literatura sobre o tema no intuito de apresentar os principais problemas e soluções, proporcionando assim maior fundamentação e embasamento na abordagem da proposta. Como ponto de partida, foi necessário realizar um levantamento das considerações básicas quanto às definições e nomenclaturas já estabelecidas em publicações reconhecidas mundialmente sobre o tema proposto, além de serem mantidas por entidades com alto impacto e respaldo. Diante os critérios pré-estabelecidos como ponto de partida, foi possível selecionar sete publicações consideradas o prisma da proposta desta dissertação, as quais estão distribuídas entre cinco entidades que desenvolveram e mantêm seus respectivos trabalhos. Segue uma breve introdução sobre as referencias selecionadas, agrupadas por sua respectiva instituição mantenedora.

- *OWASP:* É uma entidade sem fins lucrativos e de respaldo internacional que visa

melhorias e boas praticas de segurança, avaliando riscos e prevenções aos principais ataques realizados em aplicações web. Sua principal publicação relacionada ao tema deste estudo é o OWASP Top Tem 2013 OWASP (2013g), que descreve os principais ataques e riscos mais explorados através de vulnerabilidades em aplicações Web. Outras obras também relacionadas são os guias de prevenções, conhecidos como Owasp Cheat Sheets OWASP (2012b), que a principio voltados para o desenvolvimento, também abordam técnicas que podem ser utilizadas em proteções Client-side.

- *SANS /MITRE:* São duas entidades que também não possuem fins lucrativos e utilizam o mesmo respaldo da OWASP. Essa parceria possui diversas publicações disponíveis na literatura, entretanto as consideradas mais intrínsecas a esta pesquisa são as publicações: "Top 25 Most Dangerous Software Error"SANS/MITRE (2011) e a "Common Attack Pattern Enumeration and Classification"MITRE (2013a).

- *CERT:* É um centro de estudo para resposta e tratamento de incidentes em computadores, com diversos comitês de gestão em diversos países. Sua contribuição relacionada a este estudo é o "Securing Web Browser"CER (2008), no qual apresenta definições e boas práticas em ambientes client-side.

- *W3C:* O W3C é a principal organização padronizadora da WWW. A principal obra relacionada é a publicação intitulada "The World Wide Security FAQ"W3C (2002), no qual propõem padrões e boas práticas que visam minimizar os principais problemas relacionados ao conteúdo distribuído na Web.

- *HTML5Security:* É um grupo de pesquisa direcionado aos aspectos de segurança em tecnologias que utilizam HTML5. Sua principal publicação é a "HTML5 Security Cheatsheet"HTML5Sec (2013), que aponta uma serie de causas e prevenções relacionadas a ameaças nesta API nativa do navegador.

2.1 Metodologia de Pesquisa

Com os principais Ataques definidos, o próximo passo foi utilizar uma metodologia para uma busca sistêmica em fontes literárias disponíveis online. A metodologia é baseada nas orientações de Kitchenham Kitchenham (2004) que baseia-se em uma triagem segmentada em Questões de Pesquisa (QP) e tem o proposito de determinar um conteúdo e concepção de uma revisão na literatura de um determinado tema. A revisão com o tema deste estudo

visa identificar os principais Vetores, Ataques, definições e suas consequências, além de mapear trabalhos relevantes ao tema que se encontram presentes até o momento na literatura. O estudo pretende responder três questões de pesquisa, a saber: (1) Quais Vetores à Segurança mais relatados quanto aos dados sensíveis trafegados nos navegadores web? (2) Qual a relevância destes ataques durante os anos e como estão classificados? (3) Quais as principais propostas e grupos de pesquisa mais aderente ao tema durante a evolução da web?

2.2 Definição de Pesquisa e Fonte Primária

Foi definido o IEEEXplore[1] como a fonte primária deste trabalho, apesar do seu mecanismo de busca não possuir tantas funcionalidades como em outras fontes literárias, esta possui um considerável número de publicações. Também foi observado que se dispõe de um grande número de indexações de trabalhos de outras fontes. Outras fontes literárias também foram escolhidas por contemplarem trabalhos relevantes ao tema e indispensáveis para a elaboração das conclusões desta revisão, a saber: Elsevier Scopus[2], ACM Digital Library[3], SpringerLink[4], Science Direct[5] e Engineering Village[6].

Além das fontes acadêmicas, também foram consideradas algumas fontes direcionadas à indústria, pelo fato de possuírem publicações aderentes ao tema e de acesso totalmente gratuito ao seu conteúdo, estas fontes foram representadas como uma única Fonte literária nomeada como "Outros".

Como pesquisa inicial foi realizado no IEEEXplore uma busca por trabalhos relacionados a segurança nos navegadores web, utilizando palavras-chave e termos baseados nas sete publicações apresentadas anteriormente, que definiram as nomenclaturas dos ataques acerca deste estudo, chegando a seguinte regra de filtragem: [(palavras-chave relacionadas à segurança)] E [palavra-chave relacionada à navegador web] E [palavras-chave relacionadas as ameaças ao navegador web] diretamente no meta-data das publicações, formado pelo titulo, abstract e palavras-chaves de cada artigo, resultando no seguinte comando de busca conforme ilustrado na Tabela 2.1: O comando da Tabela 2.1 listou 345 publicações[7] no IEEEXplore , somando os resultados em todas as fontes de pesquisa,

[1] IEEE Xplore: http://ieeexplore.ieee.org
[2] Elsevier Scopus: http://www.scopus.com/home.url
[3] ACM Digital Library: http://dl.acm.org
[4] Springer Link: http://link.springer.com/
[5] Science Direct: http://www.sciencedirect.com/
[6] Engineering Village: http://www.engineeringvillage.com/
[7] Busca realizada em Agosto de 2013

("threat"OR "vulnerabilit* OR "secur*"OR "flaw"OR "risk"OR "unsafe"OR "safe"OR "trust"OR "untrust") AND ("browser"OR "client-side"OR "web client"OR "web application") AND (("XSS"OR "cross-site*"OR "injection") OR ("SQLi"OR "SQL"OR "injection") OR ("Clickjacking"OR "UI Redressing"OR "fraud") OR ("phishing"OR "malwares"OR "malversiting"OR "advertsing"OR "spoofing"OR "pharming"OR "fraud") OR ("CSRF"OR "XSRF"OR "Cross-site*"OR "cross-domain") OR ("XHR"OR "Same Origin"OR "SOP"OR "cross-domain") OR ("plugin"OR "extension*"OR "NPAPI"OR "short url"OR"add-on") OR ("sensitive data"OR "Weak protection"))

Tabela 2.1 Comando de busca para o IEEEXplore

resultou em 1.383 publicações.

Muitas das ocorrências não estavam no contexto da pesquisa e foi necessário realizar um refinamento manual baseado em critérios de triagem. O pouco refinamento da pesquisa foi proposital para evitar o risco de alguma publicação relevante ser excluída, sendo preferível deixar o comando de busca com escopo maior e deixar o refinamento a cargo de uma inspeção manual mais minuciosa composta por uma série de critérios e posteriormente submetida a uma triagem. Em seguida, é apresentado o processo de triagem considerando os seguintes critérios:

2.3 Critérios de Inclusão

a) Relate sobre soluções de Segurança Client-side;
b) Relate qualquer aspecto a Segurança dos dados no Navegador Web;
c) Relate sobre Falhas em Aplicações Web, mas que tenha soluções baseadas em Client-side;
d) Relate sobre Engenharia Social, mas que tenha soluções baseadas em Client-side;
e) Relate sobre Falhas no Navegador Web e seus Complementos;

2.4 Critério de Exclusão

a) Duplicidades de publicação (indexação);
b) Periódicos que não forem acessíveis online;
c) Publicações da indústria onde o acesso é restrito a pagamento;
d) Publicações que relatem problemas específicos à dispositivos móveis;
e) Publicações que tragam uma revisão sem alguma proposta como solução;
f) Publicações voltadas para recomendações de desenvolvimento seguro;
g) Publicações que estejam direcionadas a ataques que não acarretam em violação de

dados, mas sim disponibilidade, como Denial of Service (DoS) e seus variantes, ou ataques de Buffer Overflow;
h) Publicações que não estejam escritas no idioma inglês;

2.5 Critério de Relevância

a) Artigos com a solução de propostas puramente Client-side;
b) Artigos que possuam algum tipo de validação da sua proposta;
c) Artigo que relacione o maior número de Ataques sobre o tema;

2.6 Triagem

Nesta seção é descrita a triagem a qual faz uso dos critérios apresentados anteriormente com o objetivo de refinar o resultado inicial das publicações selecionadas. Inicialmente foi aplicada uma analise superficial, baseada na leitura do abstract das publicações, na qual eram aplicados os critérios previstos, uma vez contemplados, a publicação era considerada. Foi constatado que neste tipo de análise existiam casos onde o processo superficial não era suficiente, como por exemplo, quando o abstract era muito resumido ou não era possível extrair dele as informações necessárias aos cumprimentos dos critérios. Essas publicações eram alocadas separadamente para uma futura avaliação mais detalhada onde seria analisada a introdução ou demais capítulos ou seções da publicação. O resultado da triagem resultou nas publicações conforme ilustrado na Figura 2.1. Esta triagem

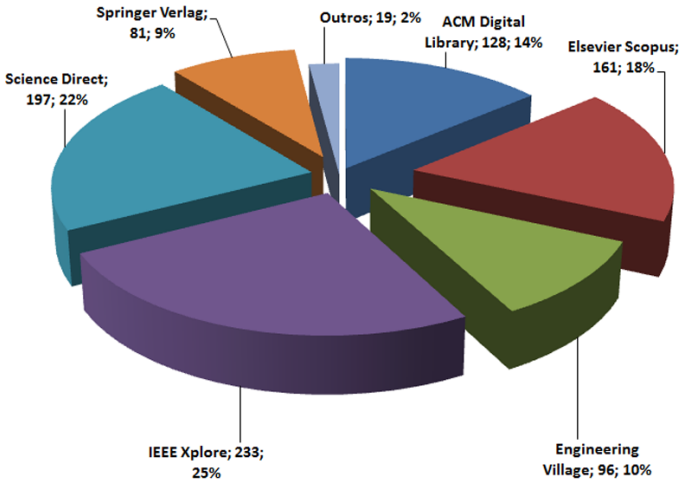

Figura 2.1 Publicações distribuídas por fontes literárias.

resultou em 915 publicações, e revelou que o IEEEXplore é a fonte com maior numero de trabalhos disponíveis sobre o tema, com 233 publicações, ou seja, a triagem descartou 112 artigos que foram considerados na pesquisa inicial. O mesmo procedimento foi realizado nas demais fontes e isso trouxe ao resultado final a garantia que as publicações selecionadas apresentam de fato aderência ao tema deste estudo. Conforme especificado anteriormente, a fonte demarcada como "Outros", representam as obras encontradas nas fontes industriais, e nestas estão incluídas as sete publicações que fundamentam os termos e nomenclaturas desta pesquisa.

2.7 Classificação dos Resultados

Nesta etapa foi realizada a análise e classificação de quais são os Vetores e Ataques relacionados ao tema, distribuídas conforme ilustrado na Figura 2.2. A partir deste estágio, se dá inicio a construção da Taxonomia, sendo definida em três tipos de ameaças:

- **Falhas em Aplicações (i):** Esta ameaça tange as vulnerabilidades oriundas de falhas no desenvolvimento da aplicação web, seja pela ausência ou ineficiência de um solido processo de desenvolvimento seguro, seja por brechas nas entradas da aplicação ou pela utilização de componentes com vulnerabilidades conhecidas.

- **Engenharia Social (ii):** A engenharia social possui um conceito muito amplo, portanto é importante salientar que no escopo desta pesquisa está intrinsecamente relacionada às ações ilícitas de usuários mal intencionados que vão bem além dos aspectos tecnológicos, como por exemplo persuadir um usuário à uma determinada ação que resulte na violação de seus dados sensíveis.

- **Falhas em Navegadores Web (iii):** Esta ameaça é proporcionada por componentes ou complementos com vulnerabilidades conhecidas instalados no navegador. Uma vez que o navegador trafega dados sensíveis constantemente, esta ameaça quando presente proporciona um impacto de negócio equivalente à propagação da popularidade destas ferramentas.

Durante a triagem foi observado que diante as três ameaças apresentadas, foram detectados cinco vetores e nove ataques relacionados que serão descritos em detalhes mais adiante, chegando à taxonomia proposta na Figura 2.2. Na Figura 2.2 demonstra que com base na classificação dos resultados na literatura, a ameaça (i) possui três Vetores relacionados, dos quais estão contidos cinco Ataques distintos. Já a ameaça (ii) possui

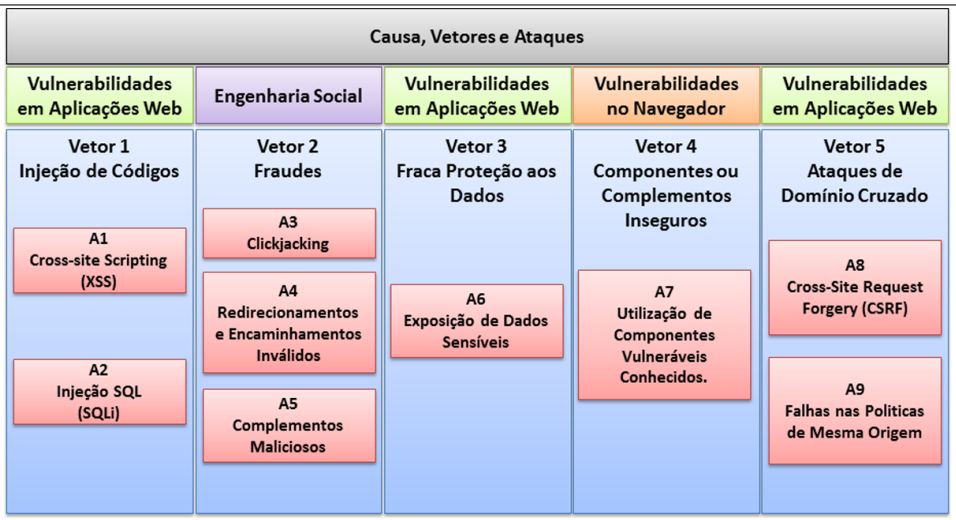

Figura 2.2 Taxonomia das Ameaças, Vetores e Ataques.

um Vetor disseminador, do qual possui três Ataques. E por fim, a ameaça (iii) possui também apenas um Vetor e um Ataque relacionado. A classificação também gerou como resultado um agrupamento dos trabalhos com maior aderência ao tema proposto desta pesquisa e o tipo da solução proposta em cada publicação. Estes resultados serão apresentados nas próximas seções de forma sistemática e em conjunto com análises e discussões desenvolvidas com o objetivo de responder as questões de pesquisas levantas pela metodologia da revisão proposta.

2.8 Análise e discussão dos Resultados

Aqui são respondidas as três questões de pesquisas propostas nesta revisão sistemática.

2.8.1 Resultado Obtido da QP 1

Apesar de cada vetor possuir uma característica específica, nada impede que o mesmo atue simultaneamente com outros vetores em um mesmo cenário, resultando assim em diversas intersecções entre as publicações e Vetores. Conforme os cinco vetores apresentados na Figura 2.2, o resultado da QP1 gerou o gráfico da Figura 2.3 no qual demonstra a distribuição das 915 publicações entre os cinco Vetores, com o objetivo de observar quais vetores são mais explorados. É importante salientar que o total de 915 artigos é divergente do somatório dos números desprendidos no gráfico da Figura 2.3 (1063), o motivo é o acumulo de ocorrências de intersecções já mencionadas, gerando

sobreposições publicações atribuídas em mais de um Vetor. O gráfico da Figura 2.3

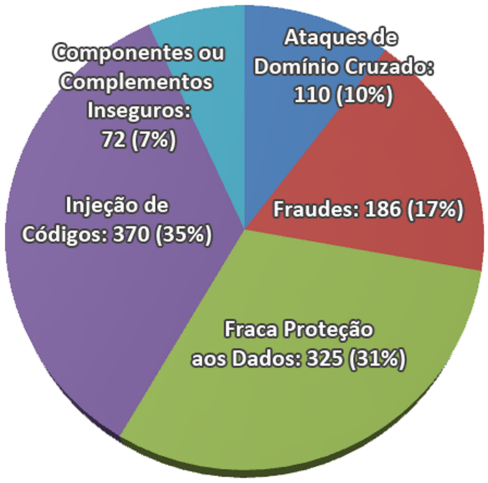

Figura 2.3 Distribuição das publicações por Vetores.

distribuiu as publicações que possuem Ataques relacionados a cada Vetor correspondente. Foi possível então dividir os Ataques entre cinco vetores distintos, sendo também possível concluir que o Vetor com maior número de publicações na literatura é a Exploração por Injeção de Códigos, e as publicações referentes a Componentes ou Complementos Inseguros fazem parte do seleto menos explorado na literatura. Nas próximas seções são descritos em mais detalhes a classificação dos Vetores de Ataques.

Vetor 1: Injeção de Códigos

Trata-se do vetor mais explorado na literatura, com 370 publicações selecionadas e seus relatos iniciaram-se na literatura em meados do ano 2000. A injeção de código é resultante de uma falha no tratamento de entradas da aplicação, como dados em formulários ou arquivos. Essas falhas são muito predominantes principalmente em sistemas legados, e dentre todos os vetores, esse é a que possui maior facilidade de detecção OWASP (2011). Em contrapartida, ainda é bastante recorrente nas aplicações SANS/MITRE (2011) MITRE (2013a) e possui um maior fator de impacto nos negócios, pois uma vez vulnerável, uma aplicação pode comprometer completamente um servidor web ou toda sua base de dados OWASP (2013g).

Vetor 2: Fraudes

O terceiro Vetor mais explorado na literatura, com 186 publicações encontradas, com foco nas fraudes, que são frutos de sites mal intencionados que furtam informações dos

visitantes através de engenharia social, sequestrando dados confidenciais ou cliques. Esse vetor é muito disseminado pelas ações dos Malwares, técnicas de Phishing, entre outros golpes dos quais simulam um cenário real de um determinado site ou serviço com o objetivo de levar o usuário a crer que o site em questão é quem se diz ser CER (2012b).

Vetor 3: Fraca Proteção aos Dados

Com 325 publicações é um vetor intermediário em termos de exploração literária, encontrando-se como segundo mais relatado. Sua consequência se caracteriza por ausência ou falha em técnicas como a criptografia ou ofuscação de dado sensível, quando não aplicadas ou aplicadas de forma ineficiente. Exemplos comuns são senhas ou dados de cartão de crédito, muitas vezes estes dados são trafegados de forma inadequada por aplicações que adotam uma fraca política de proteção à dados sensíveis OWASP (2013g)Bina (1994).

Vetor 4: Componentes ou Complementos Inseguros

Uma área bastante relevante, mas até então menos explorada na literatura. Foram encontrados apenas 72 artigos, que representa em torno de 7% de todos os artigos selecionados. Este tipo de vetor ocorre quando existe alguma vulnerabilidade no navegador ocasionada por uma falha em seu desenvolvimento CER (2008). Apesar dos desenvolvedores de navegadores estarem sempre corrigindo as falhas e incentivando seus usuários a atualizarem suas versões, é muito rápida a disseminação de uma notícia de nova falha em uma determina versão do navegador, como em casos de ataques de Dia Zero, do inglês, Zero-Day Symantec (2011).

Uma vez que muitos navegadores utilizam o mesmo motor de desenvolvimento, não é incomum encontrar inúmeras vulnerabilidades em navegadores distintos for Internet Security (2013), dando margens ao usuário ser suscetível a esta vulnerabilidade.

Outra forma de exploração deste vetor ocorre quando uma ferramenta externa é adicionada ao navegador e traz consigo alguma vulnerabilidade, que consequentemente compromete a segurança do navegador. Essa ocorrência é muito comum em Extensões ou Plug-ins, o fato é que são ferramentas desenvolvidas por terceiros, e a segurança do navegador fica em cheque devido alguma negligencia no desenvolvimento dessas ferramentas adicionais Symantec (2010).

Vetor 5: Ataques de Domínio Cruzado

Este vetor resulta em ataques que são executados de fora dos domínios da aplicação que recebeu o ataque. Casos comuns são originados de serviços externos maliciosos que se aproveitam da confiança obtida de um usuário ou aplicação W3C (2002). Existem milhares de APIs disponíveis para serem consumidas, podendo também construir combinações de outras APIs, das quais são conhecidas como Mashups James Governor (2009). Estas interfaces têm sérios problemas de padronização Silva *et al.* (2013), isto dificulta aplicar uma política de segurança consistente, e a consequência disto é que muitas vezes o controle de acesso, autenticação, tratamento das entradas, trafego de dados criptografado, monitoramento das atividades, entre outros aspectos de segurança que são negligenciados, oferecendo grande risco ao usuário de navegador web. Foi mapeado como o quarto vetor mais explorado, com 110 publicações relacionadas.

2.8.2 Resultado Obtido da QP 2

Nesta seção foram identificados os Ataques envolvidos em cada vetor da QP1, onde se fez uma classificação de forma mais granular a fim obter maior visibilidade dos resultados da pesquisa agrupados entre os cinco Vetores e os nove Ataques correspondentes. Alguns Ataques podem ocorrer em um ou mais vetores, portanto também são passiveis a intersecções, conforme a distribuição ilustrada na Figura 2.4. Com o resultado do

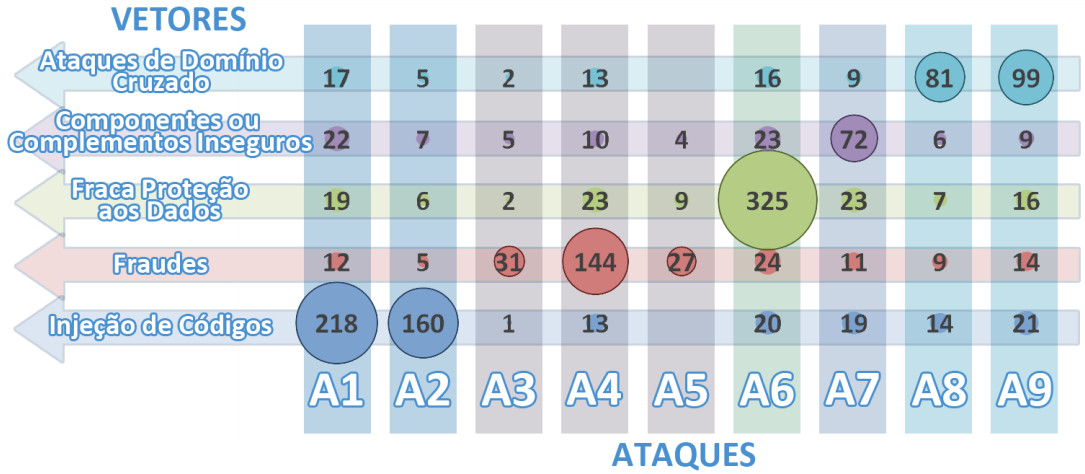

Figura 2.4 Distribuição das publicações por Vetores e Ataques.

gráfico da Figura 2.4 podem ser observados os Ataques que mais trabalham em conjunto com outros, podendo-se concluir que é algo bastante muito comum de acontecer. Além disso, pode-se concluir que as pesquisas que relatam sobre o Ataque relacionado ao vetor

de Fraca Proteção aos Dados frequentemente também relatam sobre outros vetores e consequentemente outras ameaças distintas, conFigurando o cenário de Ataques que atuam em conjunto. Nesta seção também serão apresentados os ataques que envolvem os vetores mapeados, descrevendo seu respectivo vetor, definição e impactos. Estes resultados seguem a seguinte estrutura:

- Os itens que relatam as definições são baseados principalmente nas sete principais publicações;
- Os itens que relatam as consequências são baseados nas noticias vinculadas na mídia, considerando a relação de impacto de negócio.
- Os itens que relatam as principais técnicas de ataque e defesa são baseados nos principais trabalhos obtidos nesta revisão sistemática.

Ataque 1: Cross-Site Scripting (XSS) - A1

Vetor Relacionado

Vetor 1: Injeção de Códigos.

Definição Com base nos resultados da revisão desta pesquisa, o XSS (em alguns casos também leva o acrônimo de CSS), encontra-se como a segunda ameaça mais explorada na literatura, registrando 218 publicações. Além disso, segundo a pesquisa da WhiteHat Security (2013a), trata-se da vulnerabilidade mais predominante em aplicações web, onde informa que 53% das aplicações web possuem vulnerabilidade XSS. Na mesma linha, segundo OWASP (2013g) XSS consta como o terceiro ataque mais crítico nos ambientes de aplicação web, e segundo os dados de SANS/MITRE (2011)MITRE (2013a) XSS é o problema com mais ocorrências registradas.

Esse ataque em muitas de suas facetas utilizam recursos de JS do navegador, onde permite que seu executor envie dados não confiáveis para o sistema, como scripts maliciosos que sem uma devida validação nas entradas, a aplicação acaba por permitir este tipo de dado e faz com que o navegador interprete o código e reproduza um comportamento que traga algum transtorno ao usuário MITRE (2013a). Existem três tipos de XSS:

XSS Persistente: Possibilita que um código malicioso seja permanentemente armazenado na aplicação, fazendo com que essa informação quando consumida, fará com que o navegador execute uma interpretação arbitrária, tornando qualquer usuário visitante uma vítima OWASP (2013g). Um exemplo dessa prática é conseguir inserir um código malicioso em algum campo de um formulário que persiste suas informações em um banco de dados e que posteriormente esses dados possam ser recuperados por qualquer usuário

da aplicação OWASP (2013g). Este tipo de XSS é o mais destrutivo, pois possibilita que um grande número de usuários tornem-se vitimas através de uma única falha na aplicação. Conforme na Figura 2.5, é ilustrado um exemplo mais comum de XSS Persistente.

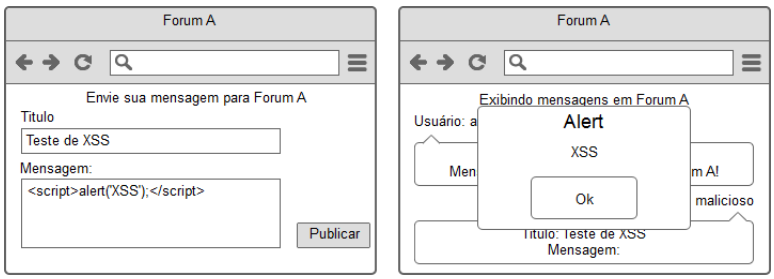

Figura 2.5 Exemplo de XSS Persistente.

Na Figura 2.5 é demonstrado um cenário onde a aplicação não faz uma devida validação em suas entradas, permitindo que seu usuário possa inserir código JS em um campo texto que quando a aplicação submeter o formulário, a mesma irá persistir essa informação em sua base de dados. Essa informação persistida será posteriormente requisitada por outro usuário da mesma aplicação, que irá se tornar uma vitima do ataque.

XSS Refletido: Esse cenário ocorre quando um usuário mal intencionado altera os parâmetros de uma URL com códigos maliciosos, e encaminha para um usuário vitima, que este ao tentar abrir o endereço, fará com que seu navegador interprete os códigos injetados via parâmetros Acunetix (2010). Muitas vezes essas ações também podem utilizam técnicas de embaralhamento OWASP (2013j) na URL para dificultar a detecção da alteração nos parâmetros, conforme na Figura 2.6.

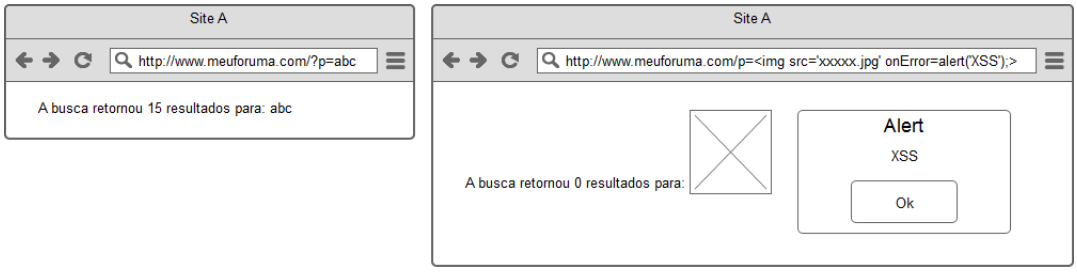

Figura 2.6 Exemplo de XSS Refletido.

A Figura 2.6 demonstra uma aplicação que não realiza uma validação nos valores atribuídos ao parâmetro "p"em sua seção de busca por termos ou palavras chaves que são fornecidos pelo usuário e processados na pagina seguinte. Neste contexto um usuário mal intencionado pode explorar essa vulnerabilidade inserindo código JS ou HTML neste parâmetro, resultando em um resultado desastroso para a aplicação na página seguinte.

XSS Baseado em DOM: DOM é o acrônimo para Document Object Model, que é um padrão utilizado para construir elementos HTML através de javascript. Quando se diz que um XSS é baseado em DOM se caracteriza por modificar das propriedades destes elementos sem a necessidade de realizar requisições no servidor, já que a modificação pode ser feita diretamente no navegador do usuário OWASP (2013f)

Selecione seu idioma:
<select><script>
document.write(«OPTION value=1>"
+document.location.href.substring(document.location.href.indexOf("default=")+8)
+«/OPTION>");
document.write(«OPTION value=2>Ingles</OPTION>");
</script></select>
http://www.meusitea.com/pagina.html?default=Frances
(Link esperado pela aplicação)
http://www.meusitea.com/pagina.html?default=<script>alert('XSS')</script>
(Link enviado pelo atacante)

Tabela 2.2 Exemplo de XSS baseado em DOM (retirado do OWASP OWASP (2013g))

No exemplo da Tabela 2.2, o código HTML em questão faz o navegador renderizar uma tela onde fornece ao usuário um menu suspenso que representa o conjunto de idiomas disponível na aplicação, disponibilizando ao usuário a opção que define o idioma padrão na página. O menu suspenso em questão está sendo criado pela aplicação através de DOM, no qual espera em seu parâmetro um conteúdo texto que represente o idioma desejado pelo usuário, quando na verdade um usuário mal intencionado pode simplesmente modificar o parâmetro para que no momento que o elemento DOM seja renderizado, irá executar o conteúdo nocivo passado via parâmetro pelo mal intencionado OWASP (2013f).

Principais Técnicas de Ataque

Nesta seção serão descritas as principais técnicas utilizadas nos ataques de XSS.

Forma Direta: São ataques que utilizam técnicas mais comuns, ou quando o código injetado é enviado sem codificação. Geralmente estas ações são minimizadas com a utilização de filtros MITRE (2013c). Basicamente apresentam-se nas seguintes formas:

a) Explorando falhas em tratamento de Escape.: São códigos injetados em uma aplicação que não faz um devido tratamento de escape nas Strings a serem exibidas ao usuário. Atualmente muitos frameworks de web por padrão ativam esse este tipo de filtragem nas saídas de String. Na tabela 2.3 é ilustrada a funcionalidade desta técnica.

b) Utilizando elementos críticos ou de conteúdo sensível: São técnicas que exploram vulnerabilidades em componentes como o Cross Site Flashing OWASP (2013h) ou ele-

String html = «b>Teste"
Aplicando a técnica de Escape a String resulta em: &lt&gtTeste&lt/&gt
Ao ser renderizada, será exibido ao usuário o seguinte texto: Teste

Tabela 2.3 Exemplo da funcionalidade da técnica de Escape

mentos HTML como frames, através de Cross Frame Scripting Network (2013a). Outros aspectos são comandos JS que manipulam dados sensíveis como o document.cookie() Sun (2012a)De Groef *et al.* (2012).

Evasão de Filtro

Evasão ocorre quando se utiliza técnicas para burlar os critérios dos filtros, como a utilização de filtros está cada vez mais comum, os atacantes começam a elaborar técnicas que contornam esta medida de prevenção OWASP (2013j).

a) Conteúdo injetado através de codificação da String: Podem ser baseadas em Unicode, Charcode, Hexa, decimal ou base64 OWASP (2013g)SANS/MITRE (2011)HTML5Sec (2013), ou baseadas em URIComponent, que é uma função nativa do JS que codifica caracteres especiais Alvarez and Petrovic (2003), ou mesmo escapes de elementos HTML ou JS SANS/MITRE (2011)MITRE (2013a).

b) Conteúdo injetado através de função dinâmica: Funções como eval(), Function(), document.write(), setTimeOut(), através de DOM ou técnicas que utilizam expressão regular em uma String em combinação da função replace do JS Anderson (2011)Bozic (2013).

Consequências

As consequências de um XSS podem ser um simples Defacement na aplicação, que segundo Anderson et al. Anderson (2011) é a desFiguração de uma pagina em um site na internet, onde seus danos são apenas estéticos. Ou até mesmo executar um roubo de dados sensíveis do usuário como dados de sessão, cookies ou redirecionamentos para sites maliciosos. Um caso que chamou atenção foi quando em 2005, o usuário Samy Kankar realizou um ataque no MySpace.com, que com o intuito de aumentar sua lista de amigos, ele injetou um script em seu perfil, fazendo com que qualquer visitante que visualizasse seu perfil se tornasse seu amigo na rede do MySpace OWASP (2012a). Além disso, o scripts se propagava de forma viral, fazendo com que também adicionasse amigos ao Samy quando alguém visualizasse o perfil de um usuário contaminado. Com isso, em menos de 20 horas Samy já possuía milhões de amigos e o site do MySpace ficou fora do ar por algumas horas para solucionar o problema em suas entradas de formulários Mook (2006). O MySpace tinha como objetivo permitir que seus usuários customizassem cores e efeitos nas informações de seus perfis, através de códigos HTML

e CSS. Apesar de possuir algumas restrições baseada em filtros, Samy identificou uma falha nos critérios de restrição, conseguindo assim inserir um código JavaScript malicioso OWASP (2012a). Uma das medidas de prevenção de XSS adotadas pelos navegadores atuais é a utilização de uma Política de Segurança de Conteúdo, do inglês, Content Security Policy (CSP) Network (2013b), que trata-se de uma especificação desenvolvida pela W3C que possibilita o navegador adotar uma política de quais recursos são permitidos ou não de serem processados em uma requisição, e essas configurações são transmitidas através dos cabeçalhos HTTP ou por Meta Tag HTML Network (2013b). Isso possibilita que o desenvolvedor de uma aplicação restrinja recursos que serão utilizados em uma determinada página, garantindo assim legitimidade de seu conteúdo. Apesar de diversas medidas de prevenção do navegador, ainda se faz ineficiente as técnicas de proteção nativa dos navegadores, seja por técnicas que se baseiam em evasão de filtros ou por falhas nas configurações da CSP OWASP (2013c).

Principais Técnicas de Defesa

Analise Estática: A Análise Estática de Código consiste em realizar uma auditoria no código, sendo necessária a intercepção da requisição antes do processo de resposta. É uma técnica muito alinhada aos Padrões de Comportamento Wassermann (2008)Johns (2013)Livshits and Chong (2013).

Padrões de Comportamento: Refere-se às prevenções relacionadas ao código JS com o objetivo de detectar comportamentos suspeitos. As medidas aplicadas são análises de códigos pré-definidos combinados a políticas de listas negras ou brancas, baseados em expressão regular Bates *et al.* (2010)V. and Selvakumar (2011)Heiderich *et al.* (2013).

Proxy Reverso: Uma camada adicional de um terceiro (provedor da solução de segurança) que através de separação ou isolamento dos dados no qual irá interceptar as requisições e aplicar um conjunto de outras técnicas Kirda (2006)Johns (2006).

Proteção de Parâmetros ou Atributos: São medidas que visam ocultar informações sensíveis no cabeçalho ou formulários, a fim de impedir a intercepção por um atacante, o que caracteriza no ataque Man-in-The-Middle. Outra medida é a analise dos valores passados em elementos, como HTML, nos quais os atacantes manipulam a fim de escapar XSS Shar and Tan (2013)Sun *et al.* (2009).

Prevenção à Evasão de Filtros: São técnicas que analisam a utilização de codificações para escapar entradas com comandos JS Venkatakrishnan *et al.* (2010)Johns and Beyerlein (2007).

Dynamic Data Tainting: Também conhecida como Data Flow Analysis, é uma analise diferente das abordagens focadas em falhas no lado do navegador. Através dela

é possível monitorar um código JS diretamente no mecanismo de renderização do JS, analisando sua semântica em tempo de execução, possibilitando assim interromper ou continuar um processamento Nentwich *et al.* (2007). É uma técnica bastante utilizada em conjunto com Análise estática e padrão de comportamento.

Ataque 2: Injeção SQL (SQLi) - A2

Vetor Relacionado

Vetor 1: Injeção de Códigos.

Definição Seguindo a revisão sistemática desta pesquisa, esta ameaça foi identificada como a terceira mais explorada na literatura, com 160 publicações, e segundo OWASP (2013g) é o ataque com maior criticidade em aplicações web devido ao alto impacto nos negócios. Injeção SQL, ou SQLi, é um ataque que se aproveita da falha no tratamento de entradas em formulários, que enviam informações das quais alimentam um banco SQL SANS/MITRE (2011).

Principais Técnicas de Ataque

a) Manipulação de parâmetro: Pode ser realizado através de parâmetros da URL (GET) ou de formulários (POST) ou em qualquer entrada que resulte em um recurso que acesse a camada de persistência de uma aplicação que utilize SQL ANSI SANS/MITRE (2011)MITRE (2013a).

b) Conteúdo injetado através de codificação: Mesma técnica de codificação utilizada em XSS para escapar caracteres especiais como aspas ou --". Também envolvem aspectos relacionados a expressão regular OWASP (2013i).

A ação ilustrada na Figura 2.7 reproduz uma concatenação na sintaxe SQL que resulta em critérios com uma verdade absoluta, no caso o '0'='0', que sempre será true, e seguindo a regra de condicionais, a sintaxe resultante seria FALSE or TRUE, como na regra do OR basta um lado ser verdadeiro, o SELECT da sintaxe SQL irá retornar TRUE na operação WHERE e todos os registros da tabela usuários. Como a consulta trará resultados, o atacante será autorizado na tentativa de efetuar login na aplicação, recebendo um acesso indevido, que irá assumir o primeiro registro da listagem retornada.

Outro exemplo mais destrutivo é quando o atacante compromete a base através de SQLi, quando ao concatenar a String com ';', sintaticamente está possibilitando a execução de outro comando na mesma String de consulta SANS/MITRE (2011)MITRE (2013a). Para destruir a base da aplicação o atacante pode acrescentar um comando que utilize DROP TABLE, conforme na Tabela 2.4. O atacante insere as aspas simples precedidas do comando DROP TABLE, do qual irá remover a tabela usuários, e finaliza

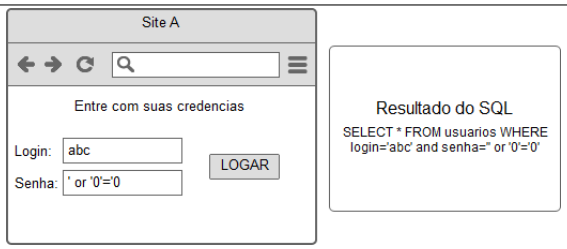

Figura 2.7 Exemplo de SQLi.

Select * from usuarios where login =’adm’ drop table usuarios; –’ and senha=’123’;

Tabela 2.4 Exemplo de SQLi destrutivo

com ;– indicando o final do comando e que após o - todo o restante da sintaxe se tornou comentário, na prática o banco de dados fará dois comandos, o SELECT, e logo em seguida o DROP TABLE, apagando a tabela usuários por completa.

Consequências

SQLi possui um alto impacto nos negócios OWASP (2013g), pois sua execução traz como o comprometimento de toda a base de dados, e consequentemente a violação dos dados dos usuários de navegador web. Além disso, o atacante pode realizar um chamado by-pass ou quebra de acesso, onde este adquire privilégios indevidos, podendo realizar acesso, exclusão, adulteração ou roubo de informações sigilosas dos usuários.

Principais Técnicas de Defesa

As técnicas utilizadas são bastante similares as aplicadas para XSS, conforme em Shar and Tan (2013)Venkatakrishnan *et al.* (2010)Johns and Beyerlein (2007). Também faz uso da Análise Estática, conforme em Halfond and Orso (2005b)Fu *et al.* (2007)Junjin (2009)

Ataque 3: Clickjacking - A3

Vetor Relacionado

Vetor 2: Fraudes.

Definição

Conforme os resultados desta revisão sistemática, o Clickjacking foi identificado como o ataque menos explorado na literatura, ao contexto do tema desta pesquisa foram mapeadas 31 publicações relacionadas. Segundo OWASP (2013b), Clickjacking, que também é conhecido na literatura por UI Redressing, é um ataque que sequestra o clique do mouse, mais precisamente quando o usuário deseja clicar em algum link ou botão disponível na pagina, fazendo com que o usuário acredite que realmente clicou no

botão, quando na verdade há elementos HTML sobrepondo o alvo, muitas vezes DIV ou IFRAME.

O Clickjacking apresenta-se como uma ou múltiplas camadas transparentes ou opacas que se sobrepõem com o objetivo de enganar o usuário que tinha a intenção de clicar em um link ou botão sobreposto MITRE (2009), conforme ilustrado na Figura 2.8.

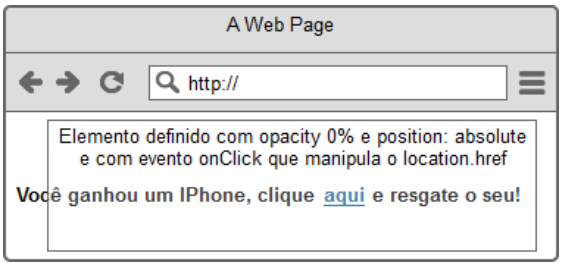

Figura 2.8 Exemplo de sequestro de Click.

Na Figura 2.8 é ilustrado um cenário onde uma pagina introduz um elemento DIV que sobrepõem um link. O DIV sobreposto possui atributos definidos para que o mesmo fique transparente e com posicionamento flutuante, fazendo com que o usuário não o enxergue e que fique por cima do link, do qual será visível e o usuário quando tentar clica-lo, na verdade estará clicando no DIV transparente, do qual irá disparar um evento que pode redirecionar o usuário para outra página. Outra técnica específica que é tão prevalente que também tem seu próprio termo cunhado, conhecido como "Likejacking", que possui o mesmo mecanismo de um Clickjacking, porem é focado contra o Facebook, onde engana um usuário sobrepondo o botão "Curtir", sequestrando seu click, fazendo com que páginas ou imagens sejam curtidas sem seu consentimento TecMundo (2013a).

Também existe uma variação do Clickjacking, denominada Strokejacking, que ao invés do click, este captura o evento das teclas do teclado. Devido ao pequeno numero de publicações encontradas na literatura sobre este ataque, este estudo considerou Strokejacking e Clickjacking como um mesmo ataque Niemietz (2011). Na Figura 2.9 é ilustrada outra utilização para sequestrar informações sigilosas como credencias de acesso, com base nas técnicas já apresentadas.

No cenário ilustrado na Figura 2.9, o usuário informa suas credencias de acesso sem desconfiar que na verdade existem campos sobrepondo os campos visíveis, estes campos ocultos podem ser submetidos em formulários ou frames cruzados, técnica conhecida como Double Framing OWASP (2013a).

Principais Técnicas de Ataque

a) Sobreposição de Elementos em Tela: Essa sobreposição é possível através de

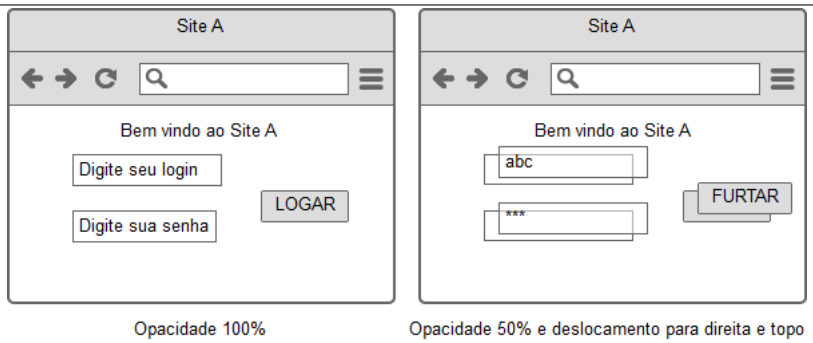

Figura 2.9 Exemplo de Clickjacking/Strokejacking para roubo de dados em formulário.

recursos do próprio HTML como o estado flutuante do objeto e a sua opacidade MITRE (2009).

b) Manipulação em Frames: Modificações no atributo parente.location.href, reproduzindo um Double Framing OWASP (2013a)Kantor (2011).

Consequências

Na prática, o Clickjacking é uma fraude. O atacante faz com que o usuário clique em um determinado recurso na pagina, onde imagina que tenha um determinado comportamento, quando na verdade seu clique foi em outro recurso, desta forma o atacante "sequestra"este clique para outros fins. Algum dos objetivos do furto do clique é redirecionar o usuário vitima de um ambiente legitimo para outro hostil onde possa extrair seus dados sensíveis, que é uma ação muito intrínseca ao redirecionamento inválido. Clickjacking atualmente está muito comum em redes sociais, uma vez que a usabilidade destes ambientes é direcionada ao simples clique das coisas, como botões curtir ou compartilhar. Mas nada impede que esta técnica também seja usada para sequestrar cliques de um teclado digital, que são muitas vezes utilizados em internet banking InfoSecurity (2012)Lundeen (2012).

Principais Técnicas de Defesa

a) CSP: Algumas medidas de prevenção adotadas pelos navegadores atuais, assim como em XSS, também utiliza a Política de Segurança de Conteúdo Network (2013b).

b) Prevenção em Frames: Utilização das opções do Cabeçalho HTTP X-Frame, impedindo modificações nos atributos de um Frame e técnicas de Frame Busting Balduzzi (2010)Gustav Rydstedt and Jackson (2010).

Ataque 4: Redirecionamentos e Encaminhamentos Inválidos - A4

Vetor Relacionado

Vetor 2: Fraudes.

Definição

A revisão sistemática mapeou-a como a quarta ameaça mais explorada na literatura, com 144 publicações relacionadas. Neste contexto estão às preocupações relacionadas ao redirecionamento do usuário durante a navegação, Segundo OWASP (2013g), as aplicações web frequentemente redirecionam e encaminham usuários para outras páginas e não usam uma validação adequada quanto aos parâmetros de redirecionamento de seu domínio, fazendo a vítima achar que uma determinada URL irá direciona-la ao respectivo domínio confiável, quando na verdade são encaminhados para sites forjados e não confiáveis.

Este contexto também está associado ao usuário ser enganado para cair em sites forjados, já que na prática tudo é uma fraude, e os objetivos são baseados em uma engenharia social da qual redireciona o usuário para um ambiente hostil, que representa visualmente muito fielmente o site do serviço oficial, mas que na verdade este irá intercepta os dados sensíveis para um golpista, fazendo o usuário pensar que suas informações estão sendo enviada para o serviço legítimo. Esta técnica de ataque é denominada Phishing, que segundo CER (2012b), Phishing, Phishing-scam ou Phishing/scam, é o tipo de fraude por meio da qual um golpista tenta obter dados pessoais e financeiros de um usuário, pela utilização combinada de meios técnicos e engenharia social, conforme a Figura 2.10.

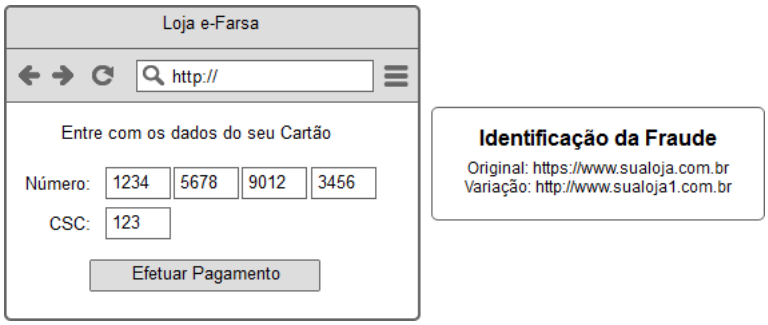

Figura 2.10 Exemplo de Phishing.

Na Figura 2.10 é ilustrado um exemplo de um site forjado com intuito de roubar os dados do usuário.

Principais Técnicas de Ataque

a) Encaminhando ao Phishing: O objetivo é introduzir o usuário em um cenário forjado para que o mesmo acredite que se trata de algo legítimo ou confiável. Com isso, o usuário fornecerá informações pessoais como senhas ou numerações de cartão de credito, e estas informações serão enviadas para o golpista que criou o ambiente forjado. CER

(2012b)

b) Redirecionando ao Phishing: Um atacante pode se utilizar de diversas técnicas para conseguir realizar um redirecionamento ao usuário, seja por modificar os parâmetros em uma URL legitima, ou até mesmo ocultar uma URL suspeita através de serviços de URL curtas, que são muito comuns em serviços de troca de mensagens sucintas como o Twitter MITRE (2013b). No caso de URL legitimas, algumas aplicações utilizam parâmetros que redirecionam o usuário para uma determinada pagina, isso é bastante comum quando uma sessão é expirada em uma aplicação, e visando funcionalidade, redireciona o usuário para a tela de autenticação preservando a URL anterior através de um parâmetro na URL do redirecionamento da autenticação. O objetivo é quando o usuário efetuar seu login, o mesmo ser encaminhado para o endereço que consta no parâmetro. Estes parâmetros podem ser alterados por usuário mal intencionados que encaminham a URL modificada para usuários vitimas. Se a aplicação não realizar uma devida validação, poderá redirecionar a vitima para uma pagina maliciosa.

Consequências

Apesar de inúmeras possibilidades, o único objetivo é conseguir roubar os dados sensíveis do usuário através das técnicas descritas de encaminhamento e redirecionamento. Atualmente o Phishing ainda segue realizando grandes prejuízos em todo o mundo Greg Aaron (2012).

Principais Técnicas de Defesa

a) Confiabilidade no Domínio: Algumas medidas de prevenção foram aplicadas, como a sigla de segurança "b.br"no Brasil, para endereço eletrônico dos sites de banco de S.Paulo (2008), gerenciado pela organização do Núcleo de Informação e Coordenação do Ponto BR, NIC.br. Porém, é uma medida ineficiente visto que poucos usuários detectam estas minucias no endereço, por muitas vezes não precisar digitar o endereço no navegador, já que os principais vetores são disseminados através de mensagens de e-mail, e o alvo em potencial são usuários de comercio eletrônico, redes sociais, companhias aéreas, e algum tipo de recadastramento que solicita dados sensíveis.

b) Controle de Phishing baseado em Lista Branca ou Negra: Muito dos navegadores atuais tem a politica de proteger o usuário baseado em uma Lista Negra atualizada periodicamente, mas a tarefa é bastante árdua devido a grande velocidade e demanda da atuação dos golpistas. Prova disso é conforme o Phishing da Figura A.1, onde foi registrado por este estudo que no dia 01/10/2013, as 16:15, este Phishing ainda era possível de ser executado por qualquer navegador popular, e todos atualizados, a exemplo do Google Chrome que nesta data e horário estava com suas atualizações em dia, na

versão 30.0.1599.66m e mesmo assim foi possível acessar o conteúdo da Figura A.1. Seguindo a mesma proposta da proteção nativa dos Navegadores, existem soluções de terceiros que visam proteger a navegação do usuário em tempo real, conforme em Tan (2009) Raffetseder *et al.* (2007) Aggarwal (2012) Yue Wang; Agrawal (2008) Joshi (2008) Nakayama (2009). Para o caso das modificações em parâmetros de URL legitimas, as aplicações podem utilizar uma politica de Lista Branca para endereços validos em seus redirecionamentos.

Ataque 5: Complementos Maliciosos - A5

Vetor Relacionado

Vetor 2: Fraudes.

Definição

Com base nesta revisão sistemática, com apenas 27 publicações, este ataque foi considerado o menos explorado na literatura. Este ataque representa outro viés da Fraude, e ocorre quando o usuário é redirecionado para um ambiente onde promete aos seus visitantes ferramentas para otimizar o seu navegador ou auxiliar na reprodução de algum formato de mídia através dele, mas que na verdade trata-se de um Trojan ou cavalo de Tróia como no caso em IDGNow (2013c), que segundo CER (2012a) é um programa que atua como hospedeiro, e embora em alguns casos ainda execute as funções para as quais foi aparentemente projetado, seus principais objetivos são outros, normalmente maliciosas, como o monitoramento do trafego ou coleta de dados, sem o consentimento e devida autorização do usuário.

Principais Técnicas de Ataque

a) Através de Plug-ins: Um tipo de plug-in muito explorado neste contexto são os baseados em ActiveX, dentre tantas possibilidades, um ActiveX mal intencionado pode propagar Malwares ou exclusão e roubo de arquivos. Estas técnicas são as mais comuns de serem aplicadas neste cenário. CER (2012a)Kumar (2013)TRENDMicro (2013)Security (2013b)Cluley (2013)Malwarebytes (2013)Symatec (2013).

b) Através de Extensões: Além dos plug-ins, as extensões também se enquadram em possibilidades de ataque nestes cenários. No decorrer dos tempos, a confiança entre o desenvolvedor da extensão e o usuário encontra-se em cheque. Normalmente as extensões tem um nível privilegiado de acesso aos recursos do navegador, como os formulários, campos do cabeçalho HTTP, Cookies, e até mesmo em alguns casos, recursos do sistema operacional. Com a ascensão destes aplicativos, as explorações de vulnerabilidades neste nível atuam descontroladamente, tornando-se comum encontrar casos como roubo de

informações e quebra de privacidade Guha (2011)Ross *et al.* (2005).

Consequências

Estes usuários são vítimas de Trojans, Malwares, Adwares, Spywares (malicioso), Adversiting, Scareware, e Malversiting, estes se apresentam como Plug-ins ou Extensões do navegador. As consequências, além a violação dos dados sigilosos, também a privacidade do usuário, já que estas ferramentas intrusas monitoraram todo o tráfego do usuário, principalmente em ambientes como internet banking CER (2012a). Muitos destes adicionais, por terem privilégios de escrita e leitura, modificam diversas configurações do usuário e afetam o comportamento não apenas do navegador, mas de todo o sistema operacional. Além disso, o computador da vitima pode se tornar um Botnet, sendo controlado remotamente.

Muitos desses Malwares atuam como trojan bancários, conforme o estudo em Bestuzhev (2013), no qual descreve o Brasil como um pais bastante explorado por este tipo de pragas. Na Figura 2.11 é descrito o fluxo de atuação dos Malwares, que quando instalado navegador do usuário, ou em alguns casos é instalado diretamente no sistema operacional, mas que frequentemente infectado através do navegador.

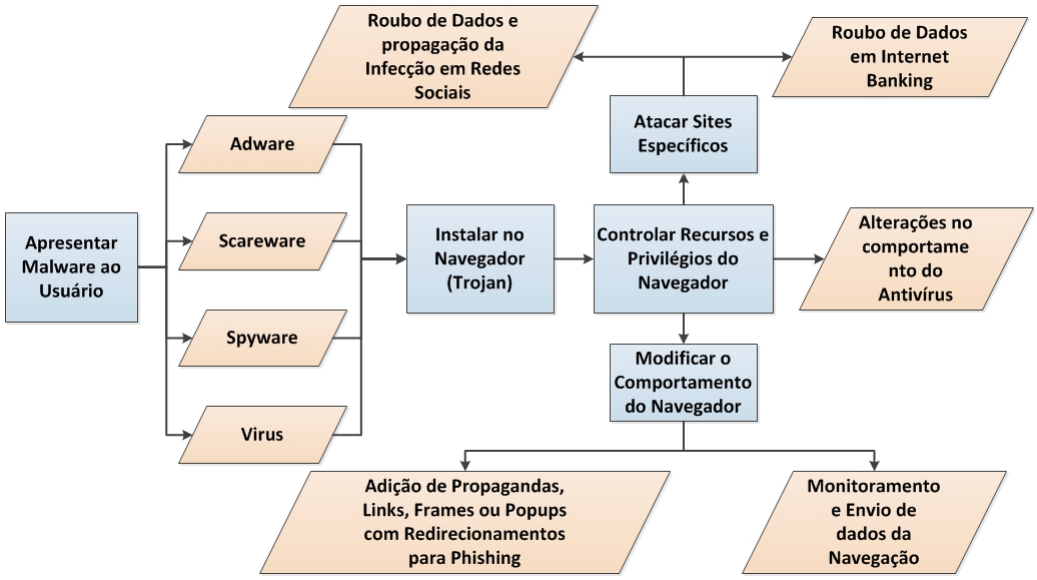

Figura 2.11 Fluxo de atividade de um Malware.

Quando instalado, o mesmo inicializa recursos para diversas atividades ilícitas, como roubo de contas em redes sociais, bloqueio no funcionamento do antivírus instalado no sistema operacional, ou monitoramento de informações e atividades da navegação do usuário infectado Fir (2013).

Principais Técnicas de Defesa

a) Politicas de restrições aos Plug-ins: A execução de um ActiveX é baseada na autenticidade do seu código, isso garante que nem todo ActiveX possa ser instalado no navegador sem previamente ter assinado um certificado através de alguma autoridade de certificação digital MITRE (2012b), como a Certsign[8], isso garante que seu código realmente foi criado e mantido por um determinado autor. O grande problema é que este autor pode estar no anonimato, visto que qualquer desenvolvedor pode criar seu ActiveX e solicitar o certificado. Uma vez que o ActiveX malicioso é autorizado para execução, todos os dados do navegador, como também o computador do usuário podem ser comprometidos MITRE (2012b). Algumas publicações na literatura propõem o monitoramento de plug-ins instalados no navegador em tempo real, como em Ford (2009)Guha (2011)Malkhi (1998).

b) Politicas de restrições as Extensões: Recentemente os desenvolvedores dos navegadores mais populares adotaram uma nova politica, da qual restringe que o navegador instale apenas extensões mantidas em seus próprios domínios, no caso do Google Chrome, o mesmo não permite que extensões fora da Chrome Store[9] sejam instaladas em seu navegador TecMundo (2013b). Um bom exemplo foi na ocasião onde um grupo de criminosos conseguiu publicar extensões maliciosas na Chrome Store Protalinski (2012).

Ataque 6: Exposição de Dados Sensíveis - A6

Vetor Relacionado

Vetor 3: Fraca Proteção aos Dados.

Definição

Com base desta pesquisa, a Exposição de Dados Sensíveis foi identificada como a ameaça mais explorada de toda a literatura, com 325 publicações registradas. Este ataque está intrínseco as politicas de segurança utilizadas durante o desenvolvimento de uma aplicação, direcionada aos requisitos relacionados a proteção dos dados OWASP (2013g)MITRE (2012a).

Principais Técnicas de Ataque

Desenvolvimento inapropriado da Aplicação: Segundo OWASP (2013g), os problemas mais comuns são na forma incorreta ou ausência na criptografia de dados sensíveis. Outras ameaças descritas na literatura, como Man-in-the-Middle e Falha no Gerenciamento de Sessões de Aplicações, também estão envolvidas neste cenário e ambos são frutos de uma falha na proteção do respectivo dado sensível OWASP (2009)OWASP

[8]Certisign - Certificadora Digital: http://www.certisign.com.br/

[9]Chrome Web Store: https://chrome.google.com/webstore

(2013g). Conforme ilustrado na Figura 2.12, este tipo de ataque também está diretamente associado às técnicas aplicadas a segurança nos dados de Cookies, Históricos, Cache ou outras funcionalidades do navegador como o auto completar habilitados em formulários ou armazenamentos indevidos em Cache ou o Storage do HTML5 MITRE (2012a)OWASP (2009). Outras falhas são ausências de HTTPS para o trafego de dados sensíveis OWA (2013f)OWASP (2013g).

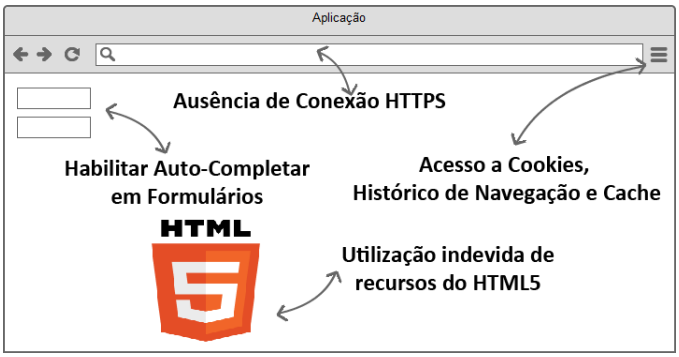

Figura 2.12 Aspectos da Exposição de Dados Sensíveis.

Consequências

Alguns dos grandes incidentes durante os anos foi o caso da empresa Adobe em outubro de 2013, onde sua base de dados foi invadida, comprometendo mais de 130 milhões de senhas de usuários SCHROEDER (2013). Outro caso é da Target Corporation, a segunda maior rede varejista da América, e hackers conseguiram explorar brechas de segurança das quais acarretaram em um roubo de mais de 40 milhões de cartões de crédito Goldman (2013)Target (2013).

Principais Técnicas de Defesa

a) Tráfego seguro para dado sensível: Algumas medidas de segurança como O HTTPS e a criptografia são bastante utilizadas em ambientes com trafego de dados sensíveis, o grande problema é que nem sempre são aplicadas nas aplicações que manipulam dados sensíveis. Algumas publicações na literatura direcionam seus estudos baseado nas medidas aplicadas pelo HTTPS, a saber: Fung (2010)Jackson and Barth (2008b)Chen *et al.* (2009).

b) Conformidades aos dados sensíveis: Outras medidas podem ser associadas à politicas de conformidades a serem exercidas nestes ambientes. A conformidade se caracteriza como uma condição de alguém ou grupo de pessoas ou processos estar conforme o pretendido ou previamente estabelecido, o pretendido em questão são os padrões de especificações. Muitas vezes essas especificações são elaboradas por empresas

que realizam auditoria externa com o objetivo de analisar se um determinado ambiente está de acordo com os requisitos exigidos.

Um exemplo é a HIPAA, sigla para Health Insurance Portability and Accountability Act, que é um padrão com o intuito de proteger os dados relacionados à saúde, garantindo privacidade e prevenção de fraudes Information (2010). Outro exemplo é o Data Loss Prevention, DLP, que é uma técnica para evitar em tempo hábil incidentes de violação ou acesso indevido a dados sensíveis, as consequências podem variar, desde a inibição de acesso ao arquivo ou a autodestruição do mesmo Shabtai *et al.* (2012). Um grande problema é que as conformidades não satisfazem todos os domínios, primeiramente muitas delas são regionais ou frutos de leis estatutárias, a outra questão é que o custo de manter o controle quanto aos requisitos é de certa complexidade e em alguns casos atingem custos muito inviáveis a serem aplicáveis devido à realização periódica de auditorias externas.

Ataque 7: Utilização de Componentes Vulneráveis Conhecidos - A7

Vetor Relacionado

Vetor 4: Componentes ou Complementos Inseguros.

Definição

Segundo este estudo, este ataque possui 72 publicações encontradas na literatura relacionada ao tema da pesquisa. Esta ameaça está direcionada as falhas existentes no navegador e seus componentes OWASP (2013g). Como dito anteriormente, diversas informações sensíveis são trafegadas durante a navegação e quando precisam ser armazenadas, é responsabilidade do navegador garantir que pessoas mal intencionadas não tenham acesso a essa informação.

Além disso, o desenvolvedor do navegador precisam adotar políticas sólidas que garantam que tais informações estarão bem protegidas, caso contrário uma falha comprometerá sua reputação e os dados do usuário Acer and Jackson (2010)Lab (2013). Outra questão é que os componentes que são adicionados ao navegador visam diretamente à responsabilidade do desenvolvedor do navegador ou de seus componentes terceiros. Outra questão é que esta ameaça é bastante similar ao de Complementos Maliciosos, porem na vulnerabilidade dos componentes as consequências não são propositais, vistos que são resultantes de falhas não intencionais, diferentemente de um complemento malicioso.

Principais Técnicas de Ataque

a) Vulnerabilidades nos Componentes: São técnicas que exploram vulnerabilidades de componentes nativos ou de terceiros que se encontram instalados no navegador. Como

exemplo a existência de inúmeras vulnerabilidades encontradas no motor de navegador WebKit Details (2013c)Vulnerabilities (2013) e Gecko Details (2013a), motores utilizado em diversos navegadores da atualidade.

b) Ataques de Dia Zero: Ocorrem quando mal intencionados se aproveitam de falhas ainda não corrigidas ou recentemente detectadas para realizar a exploração destas vulnerabilidades McAfee (2013)Symantec (2011).

c) Vulnerabilidades em Componentes de Terceiros: Além das vulnerabilidades oriundas dos componentes nativos do navegador, o usuário também está suscetível a vulnerabilidades encontradas em componentes de terceiros que são incorporados ao navegador. Estes recursos adicionais, como leitores de arquivos PDF do Adobe Reader ou vídeos Stream do Adobe Flash ou Quick Time, possibilitando a reprodução de diversos arquivos multimídia, além de oferecer recursos com maior interação do usuário em sites ou serviços disponibilizados na internet, como a Rich Internet Application, RIA, a exemplo do Microsoft Silverlight e Java FX. Comprometimento aos dados do usuário através de vulnerabilidades exploradas em recursos desta natureza não são incomuns McAllister (2013)IDGNow (2013a)Details (2013b). Outro exemplo é a vulnerabilidade no componente SilverLight que permite o acesso indevido ao arquivo clientaccesspolicy.xml, permitindo exploração de CSRF Couvreur (2006).

Consequências

As consequências deste tipo de vulnerabilidade são de alto impacto, capazes de proporcionar comprometimento de todo o sistema operacional, como os casos do Mac OS com o Safari IDGNow (2009) em tempo recorde, e do Windows com Internet Explorer Tecnologia (2009).

Principais Técnicas de Defesa

a) Atualizações periódicas dos desenvolvedores de Navegador Web: Um exemplo disso é o navegador Google Chrome recentemente ter recebido um patch de atualização que corrigiu cerca de 25 vulnerabilidades IDGNow (2013b). Apesar dos extensos históricos de incidentes e o continuo esforço por parte dos desenvolvedores, atualmente a segurança oferecida pelos navegadores é bastante questionada entre os especialistas Florian (2012)Estadão (2012).

b) Bloqueio de Recursos defasados: Como exemplo, uma medida de prevenção do Chrome foi informar aos seus usuários que deixará de permitir plug-ins NPAPI a partir de 2014. Segundo os próprios desenvolvedores do Chrome, NPAPI é uma tecnologia ultrapassada e de difícil manutenção quanto às questões de segurança GizModo (2013).

c) Politicas de Restrições aos Plug-ins e Extensões: São técnicas similares às adotas

para prevenção aos Componentes Maliciosos. As questões relacionadas às restrições nos repositórios de extensões também estão relacionadas a este contexto TecMundo (2013b). Algumas publicações na literatura propõem mecanismos de defesa neste contexto Watanabe *et al.* (2010)Silic (2010).

Ataque 8: Cross-Site Request Forgery (CSRF) - A8

Vetor Relacionado

Vetor 5: Ataques de Domínio Cruzado.

Definição

O Cross-Site Request Forgery, CSRF, às vezes também chamado por XSRF ou algumas vezes também pronunciado como Sea-Surf, e em alguns outros casos também conhecido como "Riding Session". Segundo W (2001), é um ataque do qual obriga o usuário-final para execução de ações indesejadas em uma aplicação em que ele encontra-se autenticado. Apesar de ter um impacto semelhante ao XSS, o CSRF não é uma injeção de código, mas sim uma manipulação nos parâmetros que são trafegados na requisição, ou seja, uma requisição forjada OWASP (2013g).

Apesar de ser na prática uma fraude, ela é executada fora dos domínios da aplicação, portanto o mesmo não foi considerado como um redirecionamento inválido por este estudo, pois apesar das manipulações em seus parâmetros, o fluxo é invertido: a requisição parte de um ponto de partida externo e segue direcionada para o domínio do site ou serviço legitimo. Essas foram algumas conclusões para que o CSRF fosse classificado como um ataque Cross-Domain. Foram encontradas 81 publicações relacionadas na literatura.

Principais Técnicas de Ataque

a) Manipulação de Parâmetros: Conforme ilustrado na Figura 2.13, ocorre quando o atacante envia para uma vitima um link que quando esta acessar irá disparar o método de atualização de seus dados pessoais em um determinado site, do qual serão persistidos os dados que o atacante informou nos parâmetros da requisição OWASP (2013g).

Outro exemplo pode ser feito numa compra ou transação financeira, alterando campos de quantidade ou endereço de entrega, ou qualquer outra operação que envolva o envio de dados através de uma requisição com parâmetros modificados. A princípio parece que uma simples solução seria realizar todas as requisições como POST OWASP (2013e), mas na prática não resolve o problema, visto que o atacante também pode forjar este tipo de requisições. Outra questão é que muitas vezes os usuários por imprudência acabam realizando salvamento automático de credencias ou permissões de relembrar autenticações, este tipo de ações podem resultar em CSRF através da manipulação de

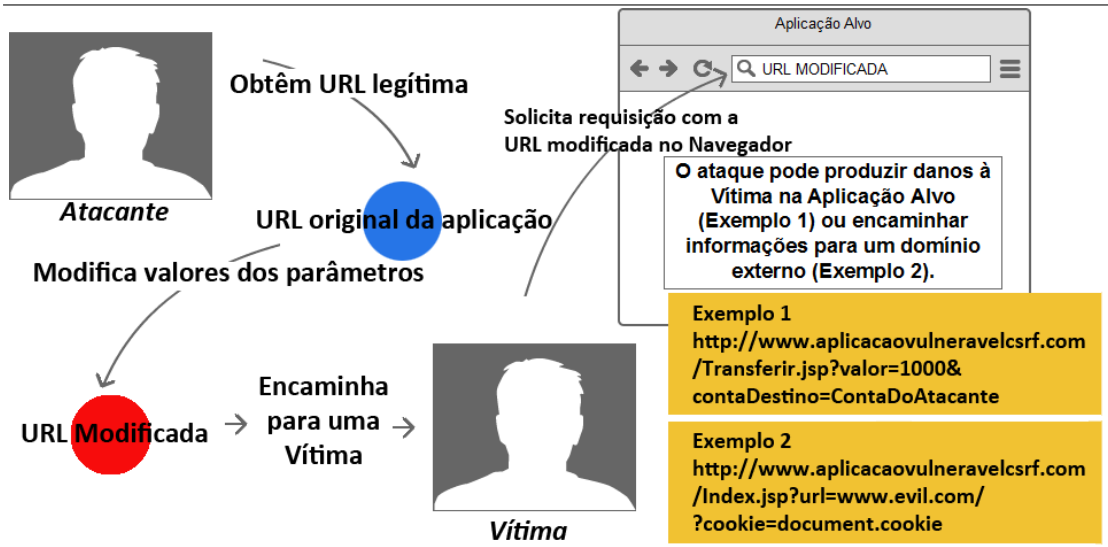

Figura 2.13 Exemplo de CSRF através da Manipulação de Parâmetros.

parâmetros OWASP (2013d).

b) Através de recursos da Aplicação: Existe também uma modalidade do CSRF denominada persistente, da qual em formulários sem uma devida validação, aceita componentes HTML como uma imagem, e no endereço desta é informado uma URL forjada, e que será executada sempre que algum usuário acessar a referida imagem, combinando assim conceitos de CSRF e XSS Cittzen (2007).

Consequências

O CSRF herda a identidade e os privilégios da vitima para realizar uma operação indesejada em nome do usuário final, um exemplo foi à ocorrência de vulnerabilidade de CSRF no Office365 Rich Lundeen (2011).

Principais Técnicas de Defesa

a) Utilização de Tokens nas Requisições: Exige que a requisição possua um Token para garantir a legitimidade da mesma. Dacosta *et al.* (2012)Mao *et al.* (2009)Wang (2013a)

b) Utilização de Cabeçalhos HTTP: Preventiva utilização dos parâmetros Referer e Origin para prevenir requisições de domínios não validados. De Ryck (2011)Adam Barth and Mitchell (2008a)

Ataque 9: Falhas nas Politicas de Mesma Origem - A9

Vetor Relacionado

Vetor 5: Ataques de Domínio Cruzado.

Definição

Com 99 publicações, é um ataque relativamente explorado na literatura. Os browsers atuais possuem algumas medidas de segurança, como a Política de Mesma-Origem (PMO) W3C (2013b), responsável por inibir que uma mesma página carregue conteúdos de domínios distintos, resultando em restrições em ambientes que utilizam o objeto XMLHttpRequest (XHR) através de JavaScript (JS).

Principais Técnicas de Ataque

a) Exploração através de XHR: Um domínio externo injeta XSS em um domínio alvo, essa técnica também é conhecida como Cross-Site Script Inclusion, XSSI Symatec (2013). Outro ataque também à exploração a PMO é o Cross Site History Manipulation (XSHM) OWASP (2010) que possibilita uma manipulação no histórico do navegador, violando a privacidade do usuário.

b) API ou Mashups não confiáveis: Outra exploração é através de APIs ou Mashups inseguros, fato muito emergente na atualidade. Com isso, o resultado do serviço pode ser originado de fontes não confiáveis, que possam explorar e executar ataques como XSS ou algum tipo de fraude Ben-Itzhak (2013).

c) Exploração de Cookies: Um dos ataques relacionados a esta técnica é o Cookie Tossing, que se define na exploração de dados que são dados armazenado e enviados pelo navegador através de uma requisição Gentile (2013)Rich Lundeen (2011), esses dados são cookies, que são chamados de CookieHTTP Livshits and Chong (2013). Este conjunto de cookies por padrão não são acessíveis via script, porem são passiveis ao acesso por seguirem as regras do escopo de cookie, podendo ser acessados através de uma requisição.

Consequências

As falhas em PMO basicamente proporcionam vetores para diversos ataques já conhecidos. Conforme na Tabela A.1, ilustra uma quebrar da Política de Mesma Origem, utilizando convenções como o jQuery , JSONP e o YQL . Para explorar um ataque de XSS, através de HTML baseado em Hex encoding, conforme a Tabela A.2, do qual reflete uma interpretação do navegador que resulta em um ataque ao usuário-final, ilustrado na Figura A.2.

Principais Técnicas de Defesa

a) CORS: Alguns padrões como o Cross-Origin Resource Sharing (CORS) W3C (2013a) são alternativas que contornam as restrições da PMO, e que ao mesmo tempo adotam práticas que não comprometem a segurança. Porém, existem convenções baseadas em JavaScript, como JSONP ou o WebSocket do HTML5, estes são capazes de quebrar a

PMO, sem necessariamente alavancar práticas voltadas para as questões de segurança Takesue (2008)Karlof (2007).

b) Utilização de Cabeçalhos HTTP: Parâmetros como Origin para prevenir requisições de domínios não validados Jackson and Wang (2007)De Ryck (2010).

Considerações Finais

E concluído a QP2, na Figura 2.14 é ilustrada a projeção das ameaças durante os anos registrados na literatura, registrando as publicações desde 1994 até o primeiro semestre de 2013.

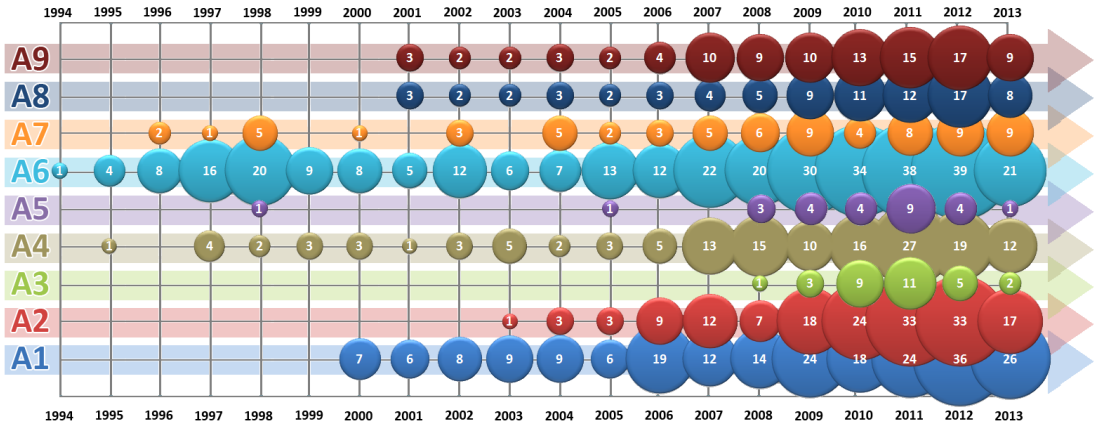

Figura 2.14 Projeção das publicações relacionadas às ameaças durante os anos.

Com base nos dados distribuídos na Figura 2.14, é perceptível que ao longo dos anos todos os Ataques estão sendo cada vez mais explorados, refletindo o ritmo de crescimento das preocupações emergentes quanto aos impactos de negócio. Em 1994 consta o primeiro registro de uma publicação na literatura Bina (1994), na ocasião o artigo trata a respeito de aspectos da Exposição de Dados Sensíveis, Ataque do qual ao longo dos anos passou a ser bastante desenvolvido e hoje detém como o mais explorado nas fontes literárias, uma das justificativas para este fato é que foi bastante comum ocorrer de uma publicação relatá-lo em combinação com outros ataques, conforme observado anteriormente na Figura 2.4.

Na mesma linha, Injeção de Código apresentou-se como o tipo de vetor mais explorado na literatura, reflexo do expressivo numero de publicações de XSS e SQLi, ocupando respectivamente o segundo e terceiro lugar na lista de Ataques mais explorados, segundo os resultados desta revisão sistemática. Não tão obstante, as preocupações referentes aos Redirecionamentos e Encaminhamentos Inválidos e Falhas em Políticas de Mesma

Origem tiveram números expressivos, ao contrário das preocupações com os ataques de Clickjacking e Complemento Maliciosos, este ultimo classificado como o Ataque menos explorado, com apenas 27 publicações.

E por fim, o CSRF e Utilização de Componentes Vulneráveis Conhecidos podem ser classificados como Ataques intermediários em respeito ao interesse dos pesquisados, e suas ocorrências na literatura apresentam certa estabilidade quanto a quantidade de publicações.

2.8.3 Resultado Obtido da QP 3

Nesta etapa foram identificadas e classificadas as publicações mais relevantes para a proposta desta pesquisa durante os anos da evolução da web, resultando em uma seleção de 225 publicações, conforme ilustrado na Figura A.3 em combinação com as Tabelas A.5 e A.6.

Durante os primeiros seis anos foram identificadas 17 publicações aderentes ao tema desta pesquisa. A web 1.0 se caracterizava por um ambiente estático e não interativo, e focado em funcionalidades, apesar de limitadas, poucas foram às medidas de prevenção à segurança, consequência foi um período que despertou as principais preocupações Naik (2008). Um bom exemplo são as antigas salas de bate-papo das quais permitiam que seus usuários inserissem códigos HTML diretamente em suas mensagens. Um serviço desta natureza nos dias atuais estaria comprometido em questões de segundos. Com a mudança de um ambiente estático para um colaborativo e o surgimento das redes sociais, a web muda radicalmente sua forma de publicar suas informações. No Período da web 2.0 foram encontradas 71 publicações.

As principais abordagens são sobre a comunicação de componentes que possibilitam as interfaces ricas e o consumo de dados através de terceiros. O dinamismo da web trouxe uma série de melhorias, mas em contrapartida também resultou em diversos problemas que envolvem todas as ameaças acerca desta pesquisa Naik (2008). Com a Web 3.0, forão detectados 137 publicações. A web agora é capaz de criar aplicações com integrações automatizadas entre fontes de dados distintas, com as chamadas maquinas social Jim Hendler (2009), que se definem como um conjunto de conectores de entrada e saída de dados, de requisição e resposta. Isso gera um fortalecimento da web semântica Naik (2008)Eftekhari *et al.* (2011), em contrapartida desperta preocupações quanto à portabilidade de dados e aplicativos interativos dos quais permitem que usuários compartilhem informações em diversas aplicações e dispositivos.

Principais Grupos de Pesquisa nas fontes acadêmicas.

a) Bina et al: Um grupo de pesquisa da universidade de Illinois, composto por engenheiros de software que desenvolveram o primeiro navegador gráfico, o Mosaic. Também foi o primeiro registro de uma publicação relatava sobre o ataque de Exposição de Dados Sensíveis, registrado em 1994 por Bina (1994) focando client-side, onde relatava sobre falhas na proteção de dados sensíveis durante a navegação do navegador Mosaic.

b) Jackson et al: O grupo de pesquisa da universidade de Stanford constituído por Collin Jackson, Adam Barth, Helen Wang, Zack Weinberg, Pavithra Jayaraman, entre outros. Estes realizaram diversos trabalhos sobre a proteção Client-side, considerando aspectos como privacidade e segurança aos dados, técnicas de proteção para domínios cruzados e segurança no navegador. Seus principais trabalhos são relacionados aos ataques A1 Bates *et al.* (2010), A3 Gustav Rydstedt and Jackson (2010), A4 Jackson *et al.* (2007a), A6 Gaurav Aggrawal and Boneh (2010)Ross *et al.* (2005)Jackson *et al.* (2007b), A7 Acer and Jackson (2010), A8 Adam Barth and Mitchell (2008a) e A9 Collin Jackson and Mitchell (2006)Jackson and Wang (2007)Huang *et al.* (2010)Wang *et al.* (2007)Adam Barth and Mitchell (2008b). Estes estudos fundamentaram boa parte da segurança na arquitetura do Google Chrome Team (2009).

c) Kirda et al.: O grupo de pesquisa da universidade de Northeastern, formado por Engin Kirda, Nenad Jovanovic, Christopher Kruegel, entre outros, desenvolveram uma serie de estudos sobre análise estática de ataques para a proteção Client-side, mais precisamente focados nos ataques A1 Jovanovic *et al.* (2006)Nentwich *et al.* (2007)Kirda (2006), A3 Balduzzi (2010) e A6 Jovanovic *et al.* (2006).

d) Gajek et al. & Heiderich et al: Um grupo de pesquisa da universidade Ruhr de Bochum composto por Sebastian Gajek, Florian Kohlar, Jorg Schwenk, entre outros e voltado para estudos de definições de segurança e proteção aos dados, com fortes estudos direcionados ao desenvolvimento de protocolos e canais de comunicação segura para dados sensíveis, relacionados aos ataques A1 Heiderich *et al.* (2013)Heiderich *et al.* (2012a), A4 Gajek *et al.* (2007), A6 Gajek (2009)Gajek *et al.* (2009b)Gajek *et al.* (2009a)Gajek (2009)Gajek *et al.* (2008a)Gajek (2008) e A9 Gajek *et al.* (2008b). Um detalhe importante é que um dos pesquisadores, Mario Heiderich, é um dos fundadores do grupo de boas praticas, o HTML5Security, uma das principais referencias HTML5Sec (2013) desta pesquisa.

e) Piessens et al: Um grupo de estudo da universidade de Leuven, onde participam pesquisadores com diversas publicações aderentes ao tema, como Frank Piessens, Philippe De Ryck, Steven Van Acker. São diversos estudos que envolvem vários ataques

relacionados neste estudo, com destaque de pesquisas que envolvem A1 Philippaerts *et al.* (2013)Van Acker *et al.* (2012)Nikiforakis *et al.* (2012), A2 Philippaerts *et al.* (2013), A6 De Ryck *et al.* (2012b)Ryck *et al.* (2012)Groef *et al.* (2011)Bielova *et al.* (2011), A8 De Ryck (2011) e A9 De Ryck *et al.* (2012a)Lekies *et al.* (2012)Van Acker *et al.* (2011)De Ryck (2010).

Trabalhos Relacionados.

Proteção contra A1: Ross Ross *et al.* (2005) propõe uma proteção nativa ao navegador Internet Explorer 8, que controla arbitrariamente qualquer código considerado suspeito, filtrando conteúdos na tag <script> utilizando o parâmetro X-XSS no cabeçalho HTTP. Na mesma linha, Bates et al Bates *et al.* (2010) propõem um componente, intitulado XSS Auditor, que atua diretamente no mecanismo de renderização do Google Chrome. O componente é responsável por realizar uma filtragem baseada em expressão regular. Chandra & Selvakumar V. and Selvakumar (2011) apresentam uma solução intitulada BIXSAN, que foca na sanitização das entradas e técnicas de evasão por meio de DOM. No mesmo princípio, Heiderich et al Heiderich *et al.* (2012b) propõem uma triagem nas manipulações maliciosas nos elementos HTML através do atributo innerHTML. E por fim, Kirdaet al Kirda (2006) apresentam o Noxes, uma solução que visa mitigar ataques por meio de análise estática baseada em uma heurística que cobre diversos tipos de XSS.

Proteção contra A2: Shahriar et al Shahriar and Zulkernine (2012) propõem uma proteção de quatro métricas de entropia condicional, na qual avaliam a proposta. As métricas analisam comportamentos da aplicação, como mensagens de erro e ausência de sanitização nas entradas em formulários.

Proteção contra A3: Lundeen & Alves-Foss Lundeen (2012) propõem defesas baseadas na análise do contexto visual, identificando sobreposições e opacidades dos elementos considerados como suspeitos. Balduzzi et et al Balduzzi (2010) apresentam soluções quanto ao controle e manipulações de frames na página, além de comportamentos de elementos flutuantes.

Proteção contra A4: No trabalho de Yue & Wang Wang (2008) é proposto um método de detecção e classificação de páginas maliciosas. Eshete et al Eshete *et al.* (2012) estendem a ideia para a análise de Phishing, como também o monitoramento e prevenção para possíveis técnicas de Malversiting.

Proteção contra A5: Carlini et al Wagner (2012) apresentam um mecanismo de avaliação de extensões para Google Chrome, que consiste em analisar as políticas de privilégio, isolamento e permissão definidas no gerenciador de Extensões. Ford et al Ford

(2009) propõem uma proteção que monitora downloads e acessos à possíveis malwares baseados em Flash nas páginas web.

Proteção contra A6: West & Pulimood West and Pulimood (2012) propõem uma análise de ameaças e prevenções na utilização de recursos do HTML5 como o WebStorage.Toubiana et al Toubiana *et al.* (2012) analisam as questões de privacidade quanto aos cookies dos serviços da Google, visando aspectos de persistência e identificação. Fung & Cheung Fung (2010) apresentam uma camada de armazenamento client-side que restringe o acesso exclusivamente baseado em em HTTPS. E por fim, Dong et al Dong *et al.* (2013) propõem detecções de ameaças e proteções contra a exposição de dados sensíveis utilizando um mecanismo denominado Cryptons.

Proteção contra A7: Walden et al Walden *et al.* (2010) abordam os impactos das vulnerabilidades existentes em plug-ins, considerando o poder de propagação dos mesmos. Schmitt et al Schmitt *et al.* (2012) propõem proteções quanto à leitura de arquivos PDF com ataques injetados baseados em JS.

Proteção contra A8: Ryck et al De Ryck (2011) proporcionam uma abordagem sobre defesas de CSRF no lado cliente, baseando-se nas permissões definidas em parâmetros do cabeçalho HTTP. Maes et al Maes *et al.* (2009) propõem uma política de proteção baseada em uma análise da troca de Tokens entre servidor e cliente.

Proteção contra A9: Bhargavan et al Maffeis (2013) apresentam uma proposta de defesa das falhas, quanto às definições de permissões de domínios cruzados, por meio dos parâmetros de cabeçalho HTTP sobre um mecanismo que estende o JS. Wang et al Wang *et al.* (2007) propõem um proxy para permissões de requisições remotas cruzadas sobre o contexto da aplicação.

Ferramentas de proteção integradas ao navegador

Nesta seção são apresentadas as principais ferramentas selecionadas no estudo que possuem seus objetivos correlatos e atuam diretamente nos navegadores mais populares. Conforme listado na Tabela 2.5, são apresentadas aquelas selecionadas que atendem os critérios previstos. Estes são baseados em considerar as ferramentas mais atuantes durante os últimos 12 meses e que sejam disponibilizadas gratuitamente. O resultado da triagem resultou em 23 ferramentas. Foi constatado que o Firefox é o navegador com um maior número disponível de opções, totalizando em 13. Em seguida o Chrome com 10 ferramentas selecionadas. E por fim, nenhuma foi contabilizada para o navegador IE, mostrando uma grande carência em relação à disponibilidade desse tipo de ferramenta.

Tabela 2.5: Ferramentas disponíveis e Ataques relacionados

Extensões	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**	**A9**	**Navegador**	**Atualização**
avast! Online Security	O	O	O	X	X	O	O	O	O	Chromeava (2013)	16/11/2013
Anti-Phishing	O	O	O	X	O	O	O	O	O	FirefoxAnt (2013)	20/06/2013
Clickjacking Reveal	O	O	X	O	O	O	O	O	O	FirefoxCli (2013a)	19/11/2013
Clickjacking Test	O	O	X	O	O	O	O	O	O	ChromeCli (2013b)	05/08/2013
CSRF Finder	O	O	O	O	O	O	O	X	O	FirefoxCSR (2013)	06/08/2013
ImmuniWeb Self-Fuzzer	X	X	O	O	O	O	O	O	O	FirefoxImm (2013)	21/08/2013
Netcraft Extension	X	O	O	X	O	X	O	O	O	ChromeNet (2013b) e FirefoxNet (2013a)	17/11/2013 e 07/10/2013
No-Script Security Suite	X	O	X	O	O	X	O	X	O	FirefoxNo- (2013)	08/11/2013
Request Policy	O	O	O	X	O	X	O	X	X	FirefoxReq (2013)	30/06/2013
Right ClickXSS	X	O	O	O	O	O	O	O	O	FirefoxRig (2013)	19/06/2013
Safe Preview	O	O	O	O	X	O	O	O	O	ChromeSaf (2013a) e FirefoxSaf (2013b)	14/11/2013 e 23/06/2013
Script Safe	X	O	O	X	O	X	O	O	O	ChromeScr (2013)	02/07/2013
SQL Inject Me	O	X	O	O	O	O	O	O	O	FirefoxSQL (2013)	02/10/2013
Tamper Monkey	X	O	O	X	O	X	O	O	O	ChromeTam (2013)	11/11/2013
Toogle Cookies	O	O	O	O	O	X	O	O	O	FirefoxToo (2013)	30/06/2013
Web of Trust (WOT)	O	O	O	X	X	O	O	O	O	ChromeWeb (2013b) e FirefoxWeb (2013c)	18/11/2013 e 18/11/2013
Websecurify	X	X	O	O	O	X	O	X	X	ChromeWeb (2013d)	04/11/2013

XSS chef	X	O	O	O	O	O	O	O	O	ChromeXSS (2013a)	20/06/2013
XSS Me	X	O	O	O	O	O	O	O	O	FirefoxXSS (2013b)	11/04/2012
Zscaler Likejacking Prevention	O	O	X	O	O	O	O	O	O	ChromeZsc (2013)	12/04/2012

2.9 Considerações Finais

Este estudo apresentou uma contextualização baseada em uma revisão sistemática de literatura, na qual através de um critério de seleção explícito extraiu os principais pontos a serem considerados quanto à problemática do tema abordado nesta pesquisa. Foram apresentados os principais ataques que um usuário de navegador web pode ser suscetível, baseando-se nas publicações de maior impacto e aderência ao tema, sendo possível extrair as definições, bem como as principais técnicas de defesa e ataque destas ameaças.

Quanto aos resultados obtidos foi possível ser identificada uma grande discrepância em relação à quantidade de publicações durante a evolução da web. Onde na Web 3.0 durante seus 3 anos e meio (de 2010 até Agosto 2013) foi contabilizado quase o dobro da quantidade de publicações durante os 10 anos da Web 2.0, este ultimo por sua vez tem um somatório maior que quatro vezes a soma das publicações na Web 1.0. Com base nestes resultados é possível considerar que a preocupação quanto à segurança em ambientes Client-side está em considerável projeção de crescimento, comprovando a relevância quanto ao problema apresentado por este estudo.

3 Implementação da Proposta

A solução proposta nesta dissertação, denominada Aegis[1], apresenta-se como uma solução leve e acoplável aos navegadores web mais populares da atualidade. Tem o intuito de proteger a navegação do usuário, objetivando minimizar as ações dos principais ataques mencionados no Capítulo 2. Nas próximas seções são apresentados, em detalhes, os seguintes aspectos do desenvolvimento da ferramenta proposta nesta dissertação, a saber: (i) Decisões de Projeto, (ii) Arquitetura e (iii) Principais Funcionalidades. Todas as conclusões desses aspectos tiveram base nos resultados obtidos no Capítulo 2.

3.1 Decisões de Projeto

Para uma solução que visa apoiar o navegador proporcionando uma navegação segura para o usuário, algumas decisões foram levantadas, considerando o público alvo, cenário de atuação e tipo de ferramenta.

3.1.1 Publico alvo e cenário de atuação

O público alvo é o usuário de navegador web que carece de uma solução única que possa minimizar os principais ataques que violam seus dados sensíveis. A ferramenta precisa interagir o máximo possível com o usuário e sua navegação pela web, portanto, o cenário ideal da proposta está nos navegadores mais populares.

[1] O nome Aegis vem da mitologia grega, era o escudo revestido com a pele da Medusa e pertencia à deusa Atena.

3.1.2 Tipo de ferramenta a ser desenvolvida

Escolhidos o público alvo e cenário de atuação, o próximo passo foi analisar o tipo de ferramenta a ser construída. Foram observados três possíveis caminhos para o desenvolvimento da proposta:

Desenvolver um novo navegador voltado para as Questões de Segurança

Esta opção foi imediatamente descartada. Primeiramente pela complexidade, seria uma tarefa muito custosa ter que manter regularmente um navegador atualizado, sem contar com o esforço que seria desprendido em outras funcionalidades e preocupações que fogem do escopo desta pesquisa. O outro fator seria da aceitação, com tantos navegadores disponíveis na atualidade, o mercado é bastante competitivo. E por fim, este caminho cairia em contradição com um dos requisitos da proposta desta dissertação, que seria proporcionar segurança às opções mais populares de navegação na web.

Desenvolver um aplicativo independente e instalável no sistema operacional

Esta opção torna-se interessante quando se deseja proteger um usuário independente do aplicativo em uso, seja um navegador, seja um cliente de e-mail. Porém, ela carece de maior integração e controle de conteúdo a ser trafegado entre o usuário e o Navegador. Um exemplo seria possibilitar uma análise do conteúdo de resposta de uma requisição antes que a mesma fosse renderizada no navegador, inibindo assim ação de diversos ataques que atuam neste contexto.

Desenvolver um aplicativo interativo

Aplicativos interativos são aplicações leves que podem oferecer certas variedades de recursos. Têm um comportamento parecido com um complemento do navegador, porém, a grande diferença é que uma aplicação dessa natureza tem uma interação mais limitada com o usuário quanto a sua atividade e monitoramento, uma de suas características, que justifica essa conclusão é o fato do usuário sempre precisar iniciar o aplicativo quando o navegador for inicializado. Isso dificulta um eficiente autogerenciamento durante a navegação do usuário Oqu (2013).

Desenvolver um Complemento para um navegador existente

Esta opção foi identificada como a mais adequada para a implementação da proposta. Primeiramente pelo fato que os navegadores mais populares, se predispõem a aceitar ferramentas desta natureza, oferecendo suporte e manutenção ao desenvolvimento, além de conceder certos privilégios sobre o conteúdo transitado pelo usuário, e até mesmo controle de certas funcionalidades do Navegador. Conforme já especificado nesta dissertação, um Complemento pode se caracterizar como uma Extensão ou um Plug-in.

Dentre as opções, a Extensão é a mais viável para a proposta, por ser a mais adequada

em nível de design de projeto: leve e funcional. Além disso, os navegadores incentivam a construção de ferramentas desta natureza, primeiramente porque disponibilizam um repositório de aplicações onde os desenvolvedores podem publicar suas Extensões. E por fim, oferecem uma vasta documentação quanto ao desenvolvimento deste tipo de aplicativo.

3.2 Arquitetura do Aegis

Nesta seção são descritos os aspectos arquiteturais da proposta, apresentando os componentes que fomentam as principais funcionalidades.

3.2.1 A Extensão Aegis

A extensão Aegis, internamente, possui a arquitetura de uma Extensão padrão de navegador que se comunica com outros recursos externos. Nesta seção serão apresentados os aspectos particulares e adicionais de suas funcionalidades, considerando seu cenário de atuação, conforme a estrutura interna da extensão ilustrada na Figura 3.1. Por questões didáticas, as nomenclaturas e descrições quanto aos componentes e APIs internas utilizadas pelas extensões terão como base a documentação das Extensões para Google Chrome, mas cabe a ressalva que os conceitos seguem um padrão onipresente nos navegadores mais populares.

Todo navegador web possui um recurso chamado Interpretador, que recebe o conteúdo da web e o interpreta em formato de leitura padrão. Em seguida, antes de ser renderizado ao usuário, o Aegis irá intervir neste conteúdo, composto por HTML, CSS e JS, que pode ter sido manipulado ou criado por um atacante, podendo o mesmo ser abortado, no caso de fraudes ou vulnerabilidades de um determinado componente. Outra medida de prevenção é o bloqueio de recursos do navegador, que tem por base uma política de confiança quanto ao conteúdo em sites com comportamentos suspeitos. Apresentando a estrutura externa da extensão Aegis, conforme na Figura 3.2, ilustrando a interface da Extensão e as funcionalidades que podem ser ativadas pelo usuário a qualquer momento em sua navegação.

O botão "Atualizar", ilustrado na Figura 3.2, serve para o usuário checar atualizações da extensão a qualquer momento. O botão "denunciar"serve para o usuário colaborar informando fraudes, sites vulneráveis ou vulnerabilidades existentes. Adicionalmente todas as opções podem ser acionadas por meio do menu no botão direito do mouse. Na próxima seção é descrita, em detalhes, as funcionalidades dos Níveis de Confiança.

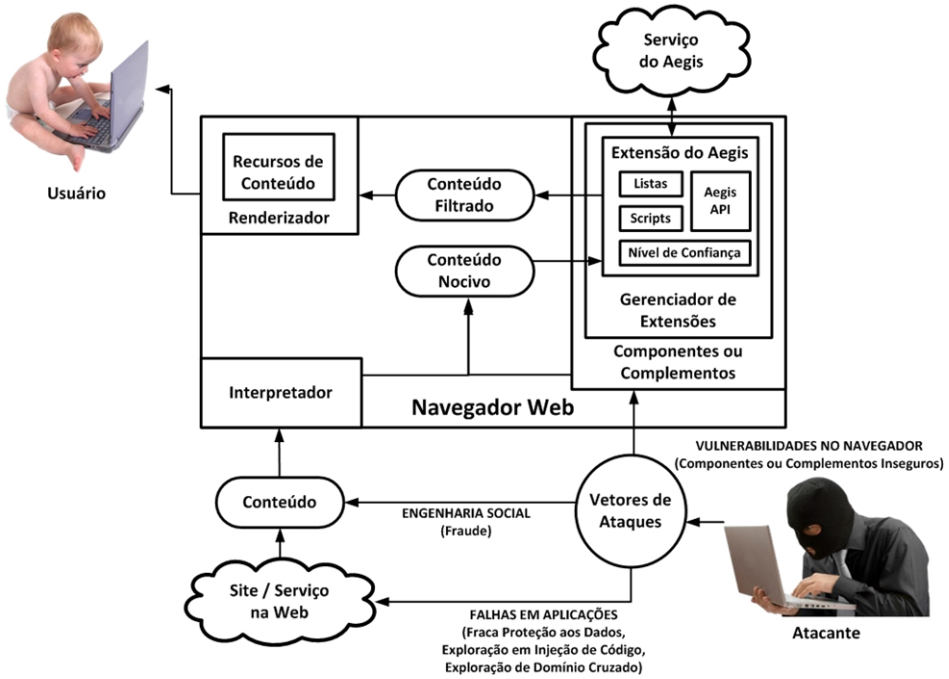

Figura 3.1 Cenário de atuação do Aegis.

Nível de Confiança

Relacionado ao nível de confiança que a extensão considera um determinado site. A avaliação é realizada de acordo com os resultados das análises dos filtros das Ameaças, uma vez identificado algum comportamento suspeito, o nível de confiança é reduzido. A confiança corresponde à liberação de recursos do navegador que podem ser desabilitados quando o nível de confiança é reduzido. Isso garante que a requisição será abortada em último caso, ocorrendo apenas quando um determinado recurso não for bloqueado por alguma interferência do navegador ou quando o site continue perigoso mesmo com o nível de confiança reduzido. Essa avaliação se faz possível através da API chrome.contentSettings, que possibilita um gerenciamento das configurações e controle dos recursos de conteúdo.

Recursos de Conteúdo

a) Requisições AJAX;
b) Cookies;
c) Imagens e CSS;
d) JavaScript;
e) Plug-ins;
f) Pop-ups;

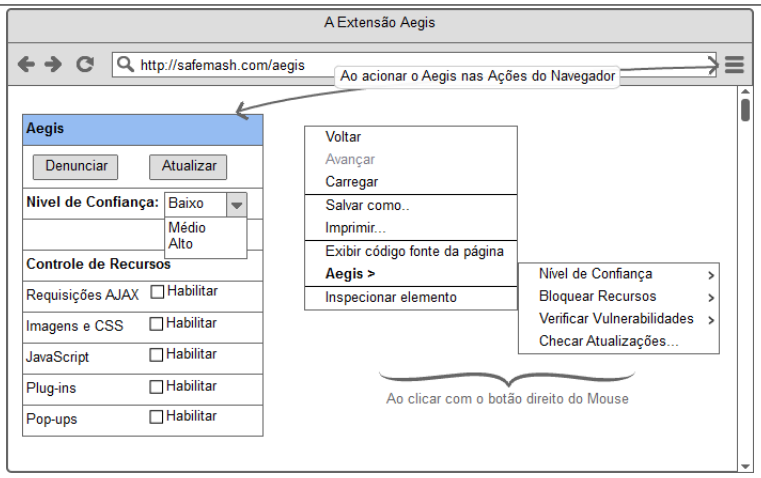

Figura 3.2 Acessos a Extensão Aegis pela interface do Navegador.

Classificação dos Níveis

a) ALTO: Permite todos os Recursos controláveis;
b) MÉDIO: Permite de acordo com o contexto do ataque;
c) BAIXO: Proíbe todos os Recursos Controláveis;

Listas

A extensão do Aegis possui um conjunto de listas que são sincronizadas periodicamente. São arquivos JSON que representam diversos recursos, a saber: listas negras de Phishing e Malwares, listas brancas para as exceções nos filtros, e um dicionário de URL curtas.

Scripts

Uma extensão armazena scripts baseados em página de eventos e de ação do Chrome. O script principal da extensão é o AegisAPI, contém os métodos que executam as chamadas de todos os filtros contra as 9 ameaças. É nessa API que fica armazenado um conjunto de regras relativas à definição dos filtros e critérios a serem executados para minimizar as ameaças ao usuário. Essa biblioteca JS ficará armazenada no navegador do usuário por meio da extensão, proporcionando uma validação local, evitando latências. Periodicamente essa API será sincronizada pelo serviço do Aegis.

Manipulação das Requisições

Os navegadores disponibilizam uma API que facilita bastante o trabalho de manipulação de requisições. No Chrome ela é chamada de webRequestWeb (2013a), no Firefox recebe

o nome de HTTPRequestsHTT (2013). Esse componente é crucial para a realização dessa proposta, pois por meio dele, uma requisição pode ser totalmente manipulada, possibilitando acessar cabeçalhos HTTP, estado de cookies, e proporciona a execução da requisição em SandBox. Além disso, a maioria dos métodos do WebRequest podem ser acionados de forma assíncrona, garantido escalabilidade e performance.

Por sua diversidade de funcionalidades esse componente será bastante detalhado nessa seção. O WebRequest é um componente que utiliza o padrão Observer. Segundo Gamma et alGamma *et al.* (1995), o referido padrão permite que um objeto alvo, nomeado Listener, seja notificado de uma mudança de estado em um método de outro objeto, denominado Source. Isso possibilita que o Source delegue a execução de um método no Listener. No exemplo da implementação desta proposta, o componente WebRequest é um Listener, que será notificado e acionado pela Extensão, que atua como um Source. Portanto a Extensão deverá adicionar eventos aos métodos do WebRequest.

Na Figura 3.3, retirada da documentação do WebRequest, estão ilustradas as funcionalidades e processos internos que são responsáveis por interceptar uma requisição.

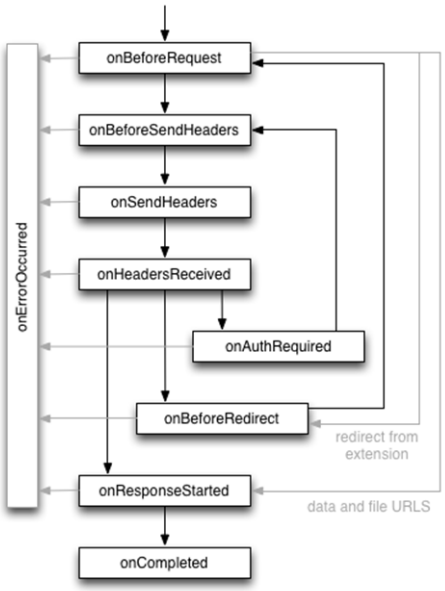

Figura 3.3 Estrutura dos Métodos do WebRequest.

A Figura 3.3 representa o ciclo de vida de um processo de captura do WebRequest. Um processo de captura corresponde a um conjunto de métodos que controlam todos os estágios de uma requisição. Para facilitar o entendimento, este estudo se aprofundou no mecanismo descrito na Figura 3.3 para apresentar o conjunto de métodos de forma mais granular. Partindo do princípio, no ato em que o usuário realiza uma requisição, o

processo de captura é iniciado, possibilitando que a Extensão intercepte uma requisição através do método onBeforeRequest. Nesse estágio, a requisição chegou à Extensão antes mesmo de atingir o servidor de destino.

É nesta fase que a Extensão possui acesso ao conteúdo do corpo da página, possibilitando uma análise de ataque que se baseia em requisições como SQLi (A2) e CSRF (A8). Adicionalmente, a Extensão tem permissão de modificar o cabeçalho HTTP, removendo ou adicionando parâmetros, usando o método onBeforeSendHeaders, e posteriormente de encaminhá-lo modificado pelo método onSendHeaders, permitindo assim mensurar comportamentos que resultam em ataques, como Falhas na PMO (A9) e Clickjacking (A3), e aplicar as devidas ações preventivas. Durante a transição entre requisição e resposta, os métodos onHeadersReceived e onAuthRequired podem ser disparados quando uma requisição requer autenticação, possibilitando à Extensão prever e intervir nestes procedimentos, gerenciando os parâmetros do cabeçalho HTTP.

Dando continuidade ao processo de captura, ainda na transição de requisição e resposta, redirecionamentos podem ser desencadeados pelo servidor destino, a Extensão pode interceptar esse evento através do método onBeforeRedirect, aprovando a detecção de Redirecionamentos ou Encaminhamentos Inválidos (A4). E concluindo o processo de captura, a Extensão poderá interceptar a resposta assim que o primeiro byte do corpo do conteúdo for recebido, significando que os cabeçalhos de resposta estão disponíveis para análise, pelo método onResponseStarted. E por fim, no método onCompleted, a Extensão pode analisar uma resposta processada, podendo mensurar ataques como XSS (A1).

É importante ressaltar que durante o ciclo do processo de captura, todos os métodos são passíveis de erro de execução, sendo possível a realização de um tratamento desses erros utilizando o método onErrorOcurried.

Manipulação de Dados Sensíveis (A6)

O navegador também permite que a Extensão tenha acesso a algumas APIs que podem modificar dados armazenados, possibilitando ações preventivas quanto à Exposição de Dados Sensíveis (A6), a saber:

chrome.browsingData

Manipula dados do diretório referente ao perfil do usuário do navegador no sistema operacional;

chrome.cookies

Possibilita a manipulação de Cookies;

chrome.history

Possibilita a manipulação do histórico de páginas acessadas;

chrome.privacy

Possibilita a manipulação das configurações pessoais do usuário no navegador, como por exemplo: salvamento de senhas em formulários, lembranças de logon, etc;

chrome.Storage

Possibilita a manipulação de dados do usuário via especificação sessionStorage, permitindo o armazenamento de dados do usuário no escopo de sessão;

chrome.system.storage

Possibilita a detecção de dispositivos removíveis de armazenamento no computador do usuário.

Manipulação de Complementos (A7)

chrome.extension

Possibilita a comunicação com outras extensões instaladas;

chrome.management

Possibilita gerenciar extensões/aplicações instaladas no navegador.

Manipulação de Dados em Transferência (A5)

chrome.downloads

Monitorar e manipular downloads de arquivos sendo processados no Chrome;

Chrome.fileBrowserHandler

Monitorar e manipular uploads de arquivos sendo processados no Chrome.

Notificações ao Usuário

chrome.notifications Possibilita a notificação ao usuário por meio de templates ou pelo próprio system tray do sistema operacional.

Operações Internas

chrome.alarms

Possibilita o agendamento automático de rotinas. Um exemplo seria a checagem da Extensão em busca de atualizações.

3.2.2 Arquitetura do Serviço do Aegis

A extensão Aegis possui um serviço externo, desenvolvido como um Web Service em REST, que fornece atualizações periódicas e disponibiliza em forma de API recursos que inibem as nove ameaças, provenientes dos cinco vetores de ataques mais explorados e mapeados no Capítulo 2, e implementados conforme especificado na seção de Filtros dos Ataques. Algumas funcionalidades dessa API compartilham recursos de API de terceiros, conforme na Figura 3.4.

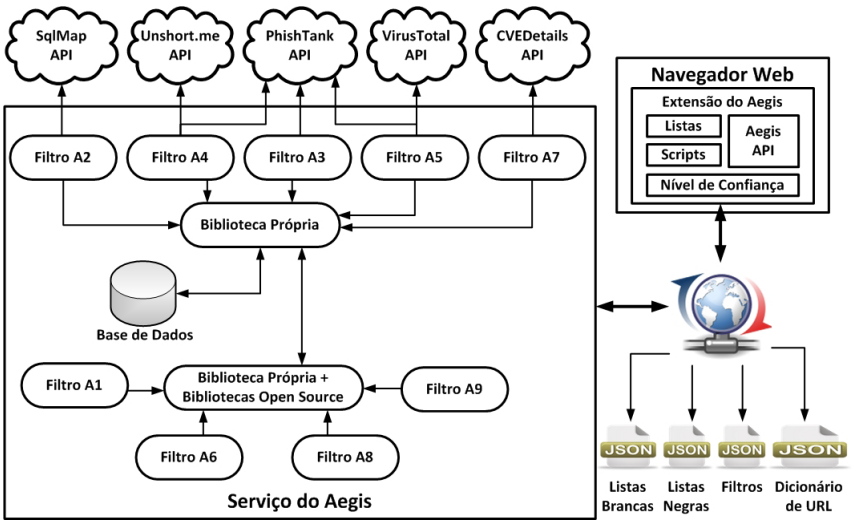

Figura 3.4 Arquitetura do Serviço do Aegis.

Biblioteca Própria

O serviço do Aegis dispõe de uma biblioteca própria, que oferece soluções de prevenção para Ataques que não estão cobertos por bibliotecas de terceiros, como os Ataques relacionados ao A3, A4 e A5.

Bibliotecas Open Source

A OWASP Enterprise Security API, também conhecida como OWASP ESAPI[2], é uma documentação desenvolvida pela OWASP que traz uma série de bibliotecas voltadas para linguagens em ambientes Server-side, de código aberto, além de um conjunto de boas práticas no desenvolvimento seguro. Apesar de ser Server-side, alguns de seus

[2]OWASP ESAPI: https://code.google.com/p/owasp-esapi-java/

conceitos e técnicas foram adaptadas para uma versão Client-side intitulada JavaScript-based ESAPIOWA (2013b). Até o presente momento, esta biblioteca encontra-se em fase alfa e sua última versão foi liberada em 2011.

Para atender os requisitos desse estudo algumas funcionalidades precisaram ser incorporadas, sendo necessário realizar algumas modificações e adições de funcionalidades nesta biblioteca. Além disso, por ser a biblioteca de código aberto facilitou o processo de adaptação, garantindo apoio à plenitude dos requisitos. A documentação da biblioteca do OWASP ESAPI padrão é apoiada por muitos especialistas, que auxiliam em constantes modificações e melhorias, esse é um dos motivos para as atualizações periódicas das regras dos filtros do Aegis. Esse estudo tem o objetivo de disponibilizar essa adaptação da biblioteca em JS para que possa ser incorporada ao projeto Client-side do ESAPI.

Serviços Externos

Os serviços externos são funcionalidades disponíveis online e gratuitas que interagem em tempo real com o Serviço Aegis, provendo recursos que auxiliam nos objetivos a serem alcançados com base em entidades de excelência em soluções relacionados aos problemas em questão.

Serviço Unshort.me API

O Unshort.me[3] é um serviço gratuito que descobre a URL real de uma URL curta, disponibilizando o seu serviço por meio de uma API em REST de fácil utilização e rápida resposta nas requisições.

Serviço VirusTotal API

O VirusTotal[4] é um serviço gratuito que contempla mais de 40 serviços de antivírus online diferentes, proporcionando maior precisão nas varreduras de diversos tipos de Malwares e uma rica análise de URL suspeitas. Seu serviço também é disponível em REST e internacionalizável em diversos idiomas, incluindo o português.

Serviço SQLMap API

O SQLMap API[5] é um serviço disponível também em REST que analisa um site e verifica se o mesmo está vulnerável ou não à SQLi. O serviço ainda encontra-se em fase experimental, mas já possui alguns resultados satisfatórios.

Serviço CVE Details

CVE é um acrônimo para Common Vulnerabilities and Explosures, apresentado no

[3]Unshort.me: http://www.unshort.me
[4]VirusTotal: https://www.virustotal.com
[5]SQLMap API: https://github.com/sqlmapproject/sqlmap

site CVE Details[6], é uma base de registros de vulnerabilidades de segurança em diversos recursos computacionais. É mantida pelo MITRE e possui um grande acervo, no que diz respeito a vulnerabilidades em recursos computacionais em geral. Nele consta uma considerável base de informações quanto aos motores de navegadores, bem como diversos componentes e plug-ins.

Serviço PhishTank

O PhishTank[7] é um serviço desenvolvido por uma comunidade voltada para o anti-phishing, lançado em outubro de 2006, uma célula de desenvolvimento do grupo OpenDNS[8]. Os navegadores mais famosos utilizam o serviço denominado SafeBrowsing[9], porém em testes realizados nesta proposta, uma denúncia de Phishing recentemente catalogada no PhishTank ainda era possível de ser acessada no Chrome e Firefox.

Portanto, o estudo considerou necessária a utilização desse serviço da OpenDNS. Além disso, outros navegadores utilizam o serviço do PhishTank internamente, porém, a lista só é sincronizada quando um novo release do navegador está disponível. E, a exemplo dos demais serviços de terceiros integrados nesta proposta, o serviço do Phishtank também pode ser acessado através de uma API em REST.

3.3 Funcionalidade dos Filtros

Nesta seção são descritos os filtros que visam minimizar as nove ameaças mais exploradas quanto à violação de dados sensíveis do usuário de navegador web. As principais medidas tomadas pelos filtros, para alcançarem seus objetivos, é bloquear os recursos do navegador ou aplicar ações preventivas quando o conteúdo em questão possui algum comportamento considerado hostil ao usuário. Um ponto positivo, por ser uma defesa client-side, é que o usuário pode ter uma interação maior com a ferramenta, como por exemplo, ele poderá indicar que um determinado site enquadra-se em uma regra de exceção, representado por uma lista branca para cada filtro.

Essa lista serve para indicar para a ferramenta, que apesar do site em questão ter um comportamento suspeito, o mesmo é considerado confiável pelo usuário, resolvendo assim os problemas de falsos positivos. Por questões didáticas, serão apresentados fluxogramas que representam a funcionalidade de cada filtro específico, sendo importante frisar que, apesar das diferenças no desenvolvimento entre os três navegadores mais populares,

[6]CVE Details RSS: http://www.CVE Details.com/vulnerability-feeds-form.php
[7]PhishTank: http://www.phishtank.com
[8]OpenDNS: http://www.opendns.com/
[9]SafeBrowsing: https://developers.google.com/safe-browsing/

os padrões estruturais e seus componentes são bastante similares, fazendo com que a representação em fluxogramas proporcione o entendimento necessário, indiferente do navegador a ser desenvolvido. Outro fator que vale ser ressaltado é que toda a heurística e as funcionalidades dos filtros foram fundamentadas com base nas técnicas de defesa e ataque que foram obtidas nos resultados apresentados no Capítulo 2.

3.3.1 Filtro A1: Ataques baseados em A1

A filtragem de XSS possui regras pré-estabelecidas baseadas em análise estática no conteúdo da requisição, uma vez interceptado pelo WebRequest no método onCompleted, conforme ilustrado na Figura 3.5. Numa visão geral, o filtro do Aegis visa robustez quanto à prevenção de técnicas de evasão, com base nas técnicas de defesa descritas no Capítulo 2, mais precisamente na seção 2.8.2. Este estudo visa contribuir com uma evolução e atualização contínua quanto à detecção dos padrões de ataques em conjunto às prevenções utilizadas na biblioteca do OWASP ESAPI, que são em código livre e passíveis de adaptações.

Conforme ilustrado na Figura 3.5, o filtro possui três componentes principais, cada um deles possui uma heurística especifica que relaciona as principais técnicas de defesa descritas no Capítulo 2. O filtro inicia seu processo no método onCompleted do WebRequest, verificando se o site em questão encontra-se na lista de exceções. Essa lista serve para catalogar os sites que o usuário considera como confiáveis, identificando que nesses casos a requisição poderá ser processada. No próximo passo, o filtro captura o conteúdo a ser apresentado ao usuário e submete-o a uma análise estática, buscando elementos considerados suspeitos, mais precisamente tags HTML como <iframe> ou <embed>, entre outras previamente catalogadas em uma lista negra.

Ocorrendo a utilização de algum elemento presente na lista, o filtro irá questionar se o domínio em questão está presente em uma lista branca, que representa sites anteriormente considerados confiáveis pelo usuário. O processo seguirá normalmente caso o site em questão conste na lista branca, caso contrário o filtro irá bloquear algum recurso do Navegador. Por ser um ataque de XSS, geralmente o recurso a ser bloqueado é o JS. Porém, outros recursos podem ser bloqueados de acordo com a semântica do conteúdo, um bom exemplo é o caso de XSS sendo aplicado via CSS OWA (2013g).

Além disso, um atenuante nestes cenários é que uma vez a aplicação suscetível a XSS, ela também está vulnerável a ataques XSS explorados através dos componentes de APIs do HTML5, como WebWorkers, WebSocket, WebMessaging, WebDatabase e WebStorage, portanto estes recursos também serão bloqueados no navegador do usuário.

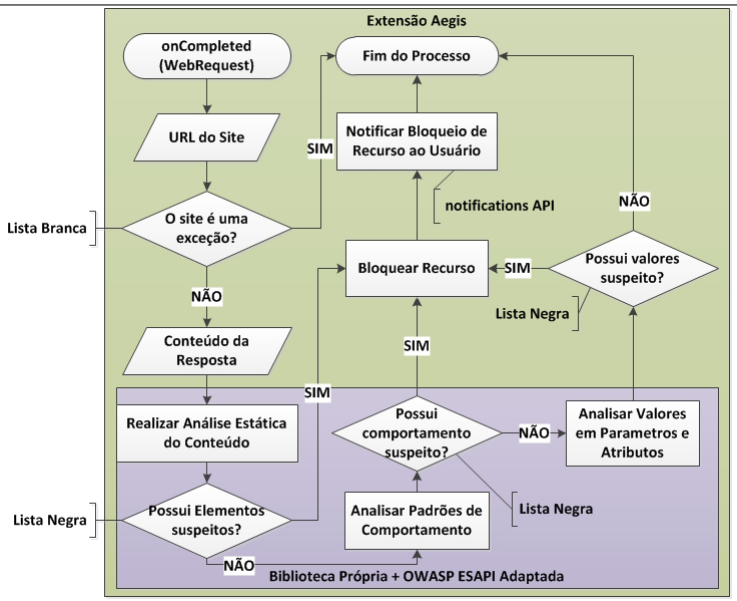

Figura 3.5 Fluxograma da funcionalidade do filtro de XSS (A1).

Caso a análise estática não encontre elementos suspeitos, o próximo passo será analisar o conteúdo em busca de padrões de comportamento, conforme descrito anteriormente no Capítulo 2, como utilização suspeita de expressões regulares ou mesmo evasões de filtro. Por fim, o conteúdo será submetido para uma nova análise, esta baseada em uma triagem quanto aos valores atribuídos em atributos de elementos declarados na página.

3.3.2 Filtro A2: Ataques baseados em A2

O processo de filtragem de SQLi é dividido em dois fluxos: a solicitação e resposta de filtragem, conforme ilustrado nas Figuras 3.6 e 3.7 respectivamente. No primeiro caso, o usuário manualmente solicita a Extensão para que verifique se um site A é vulnerável ou não à SQLi. Essa filtragem é baseada em duas respostas: primeiramente em uma análise baseada em comportamento da aplicação, e caso não seja suficiente, o próximo passo será obter uma resposta através da API do SQLMap. No primeiro caso, o processo tem uma resposta rápida e geralmente resolve vulnerabilidades mais evidentes. O segundo caso é uma operação com maior profundidade, que ocasionalmente resulta em um processamento custoso e, quando solicitado pelo usuário, a Extensão irá definir o status do site como "Em Análise". Como é um processo assíncrono, o usuário poderá posteriormente acessar outros sites ou até mesmo cancelar o pedido pendente de análise. O objetivo deste fluxo é analisar se o site não é vulnerável em SQLi, caso positivo o mesmo será adicionado na lista branca de sites não vulneráveis.

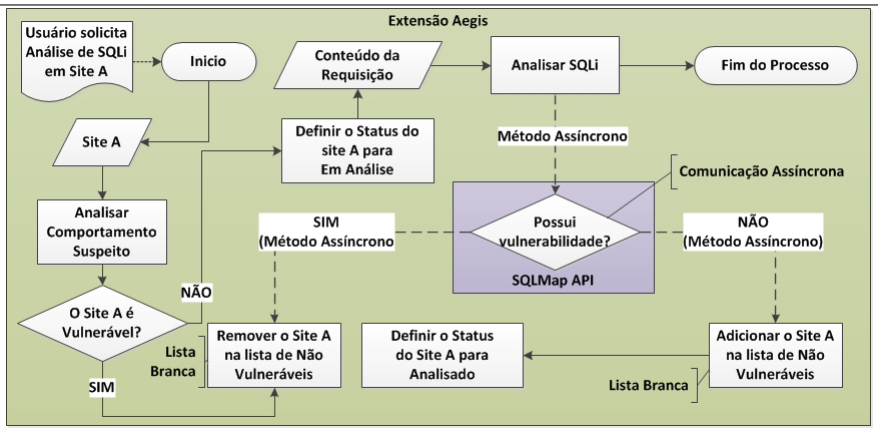

Figura 3.6 Primeiro fluxograma da solicitação de filtragem de SQLi (A2).

O segundo fluxograma, com base na Figura 3.7, descreve a filtragem aprimorada no resultado do primeiro. Caso o site esteja "Em Análise", o processo de requisição do mesmo será bloqueado e notificará ao usuário o ocorrido até que a análise, em comunicação assíncrona, seja concluída. Caso contrário, será verificado se o site consta na lista branca de não vulneráveis à SQLi. Uma vez ausente na lista a requisição será bloqueada e o motivo será notificado ao usuário. Caso conste na lista, o processo será concluído normalmente. A mecânica do filtro consiste em observar que enquanto o site não estiver considerado como "Não Vulnerável"qualquer requisição atribuída a ele será bloqueada até que ocorra uma as seguintes hipóteses: a primeira é quando próprio usuário define o site como confiável, resolvendo um falso positivo.

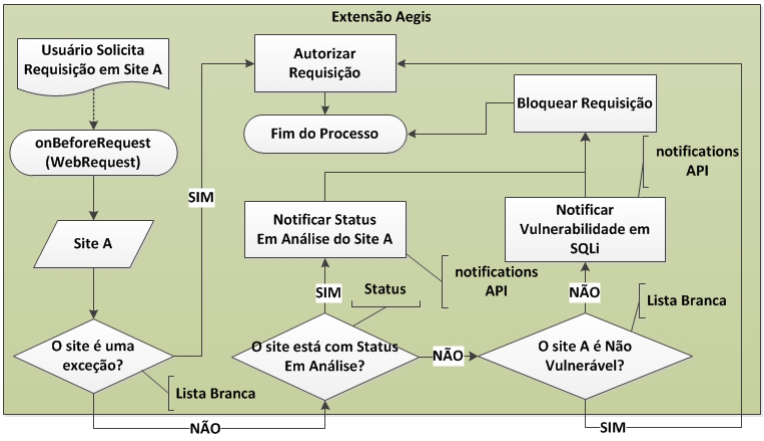

Figura 3.7 Segundo fluxograma da solicitação de filtragem de SQLi (A2).

A segunda seria quando o desenvolvedor da aplicação corrigir a falha e posteriormente o usuário solicitar uma nova solicitação de filtragem para que a ferramenta classifique

o site como Não Vulnerável. Uma vez sem falhas, o site se mantém protegido por um longo tempo, portanto a classificação por lista branca se faz eficaz. Mas, por questões de prevenção, o Serviço do Aegis irá reavaliar periodicamente mediante primeiro fluxo, os sites considerados como Não Vulneráveis em determinados períodos estratégicos, como em horários mais ociosos. É importante ressaltar que todos os sites que possuam alguma característica básica do SQLi, como parâmetros em GET ou POST, a Extensão irá sugerir ao usuário uma análise de SQLi, essa,quando acatada, irá executar o primeiro fluxo. Um pedido recusado pelo usuário faz com que a ferramenta entenda que se trata de um site confiável, sendo assim adicionado à lista de exceções.

Adicionalmente, o filtro de SQLi também realiza uma checagem de vulnerabilidades em aplicações que utilizam o WebSQL. A premissa básica é de, por meio de expressão regular, observar sintaxes SQL que estejam utilizando implementações que previnam SQLi nesses cenários, como por exemplo, a presença de concatenação de parâmetros utilizando o vars OWA (2013c), ao invés de concatenar parâmetros diretamente na String. Essa prevenção se faz importante porque uma vez um site vulnerável a XSS, um atacante pode roubar informações através de WebSQL.

3.3.3 Filtro A3: Ataques baseados em A3

Este ataque ainda carece de boas soluções na literatura, principalmente no nível de proteção Client-side, um fator que comprova isso é a pequena quantidade de publicações encontradas, conforme os resultados do capítulo 2. Todo o desenvolvimento da prevenção deste Ataque nesta proposta é originário da biblioteca própria do Aegis. A heurística do combate, além de se ater às prevenções em frames e parâmetros do cabeçalho HTTP, conforme especificado no Capítulo 2, também se baseia na análise estática da requisição do usuário, com o intuito de checar o posicionamento de elementos na tela, além de aspectos como opacidade, características que classificam um elemento como suspeito.

Todas as filtragens relacionadas à Fraude serão submetidas a uma checagem da URL a fim de proporcionar ações anti-Phishing, por ser bastante reusado e visando facilitar seu entendimento, esse processo foi separado como um fluxo interno denominado de Sub-rotina anti-Phishing, conforme ilustrado na Figura 3.8.

O primeiro passo desta sub-rotina é com relação às URL curtas. Com a grande demanda e descontrole no surgimento de serviços encurtadores de URL, todas as URLs informadas ao navegador serão submetidas a uma checagem do destino da URL encurtada através do Unshort.me. Uma vez recuperada a URL original, a mesma será armazenada no dicionário de URL. O próximo passo é avaliar a URL original junto à lista branca de

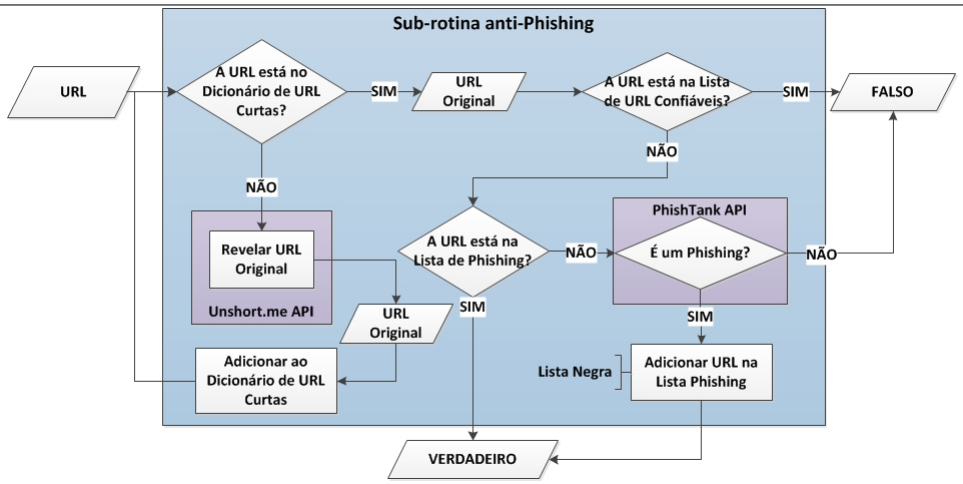

Figura 3.8 Fluxograma da Sub-rotina anti-Phishing.

URL confiáveis, uma vez constando o resultado da sub-rotina será FALSO, indicando que não se trata de um Phishing. No caso de ausência da URL na listagem de confiáveis, o próximo passo será verificar se a URL consta na listagem de Phishing da Extensão Aegis. Essa listagem proporciona redução de latência, uma vez que armazena confirmação de Phishing em análises anteriores da mesma URL.

Caso a URL conste na lista negra, o resultado da sub-rotina será VERDADEIRO. Uma vez ausente, posteriormente a URL será avaliada no serviço externo do PhishTank. O resultado do serviço do PhishTank é uma análise da URL em seu banco de dados de Phishing catalogados. Uma vez constatado que a URL trata-se de um Phishing, o resultado dessa confirmação será adicionado à listagem de Malwares e o resultado do fluxo retornará VERDADEIRO, caso contrário o fluxo apenas retornará o valor FALSO. Na figura 3.9 está ilustrada a filtragem de Clickjaking, a qual possui, como primeira etapa, a execução da sub-rotina anti-Phishing. Posteriormente será verificado se o site em questão trata-se de uma exceção, se quando verdadeiro, ignora o restante do processo de filtragem e prosseguirá com a requisição. Caso seja falso, será recuperado o conteúdo da requisição em busca dos elementos contidos na tela.

De posse desses elementos, verificar-se-á a distância entre um elemento e outro, identificando, assim, as sobreposições ou manipulações maliciosas em frames. Caso alguma dessas características seja detectada, a Extensão considerará o site como uma Fraude, bloqueando sua requisição e informando ao usuário sobre o bloqueio. Caso contrário, o próximo passo será interceptar o conteúdo do cabeçalho HTTP usando o método onBeforeSendHeaders, do WebRequest em busca de comportamentos suspeitos em parâmetros do cabeçalho.

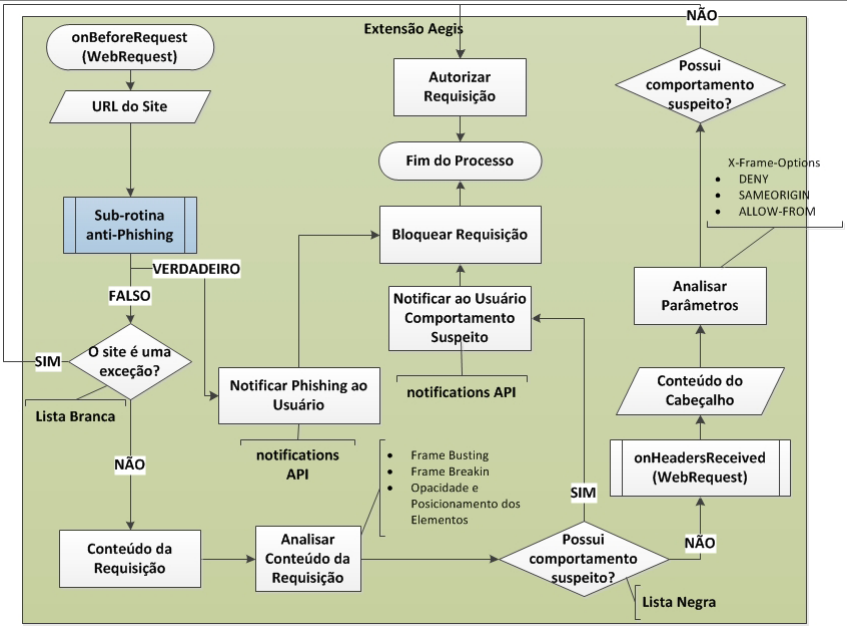

Figura 3.9 Fluxograma da filtragem de Clickjacking (A3).

Vale uma ressalva a respeito da decisão de sempre checar as URLs e a utilização dos serviços do Unshort.me e PhishTank. Foi observado durante os testes realizados neste estudo, e até o fechamento deste capítulo[10], que em serviços como goo.gl[11], bitly.com[12], e o mcaf.ee[13] foram bloqueadas as tentativas de criar uma URL curta em uma URL catalogada no PhishTank. Em contrapartida, em serviços como Ow.ly[14], Tiny.cc[15] e o Bit.do[16], foi possível criar uma URL encurtada que redireciona para uma URL maliciosa. Outra questão foi que as URL encurtadas geradas nos serviços mencionados, quando inseridas no PhishTank, o serviço anti-phishing informou que não há nada a respeito sobre a URL, ou seja, concretizando em um ByPass no serviço do PhishTank. Com base nestes resultados, este estudo concluiu que se faz necessária a integração entre o serviço do Unshort.me, para em seguida enviar a URL original a ser avaliada no PhishTank.

[10]Sexta-feira, 10 de Janeiro de 2014.
[11]Google URL Shortener: http://goo.gl/
[12]Bit.ly your bitmarks: https://bit.ly
[13]McAfee Secure Short URL Service: http://mcaf.ee
[14]Ow.ly Shorten URL: http://ow.ly
[15]Tiny URL: http://tiny.cc
[16]it.do URL Shortener: http://bit.do

3.3.4 Filtro A4: Ataques baseados em A4

A extensão irá monitorar toda a navegação do usuário e imediatamente bloquear qualquer redirecionamento ou encaminhamento inválido detectado. O fluxograma na Figura 3.10 descreve os métodos de análise e bloqueio de um link nocivo. Neste filtro, o processo é realizado em dois métodos do WebRequest. A filtragem se inicia com o método onBeforeRequest, que realizará a sub-rotina anti-Phishing, anteriormente apresentada na seção da ameaça A3.

Posteriormente a ação do filtro ocorrerá no recebimento da resposta da requisição, através do método onCompleted, e o comportamento do filtro será coletar o conteúdo a ser exibido ao usuário. De posse desse conteúdo, o próximo passo será coletar todos os elementos que possuam URL, sejam links ou elementos com eventos JS que possuam uma URL. O objetivo é analisar a URL de cada elemento e submeter aos resultados da sub-rotina anti-Phishing, uma vez contendo uma URL nociva, o elemento será modificado. Importante lembrar que o procedimento é assíncrono e limita-se em até 100 links por página, o restante ficaram desabilitados e serão filtrados pela confirmação do usuário.

Entrando em mais detalhes quanto ao processo de modificação dos elementos, quando um link for detectado como nocivo, terá sua funcionalidade desabilitada, e a modificação será anunciada ao usuário por meio de um balão do tipo "ToolTip". A checagem é realizada no momento em que toda a página é carregada, garantindo assim que todos os elementos sejam capturados.

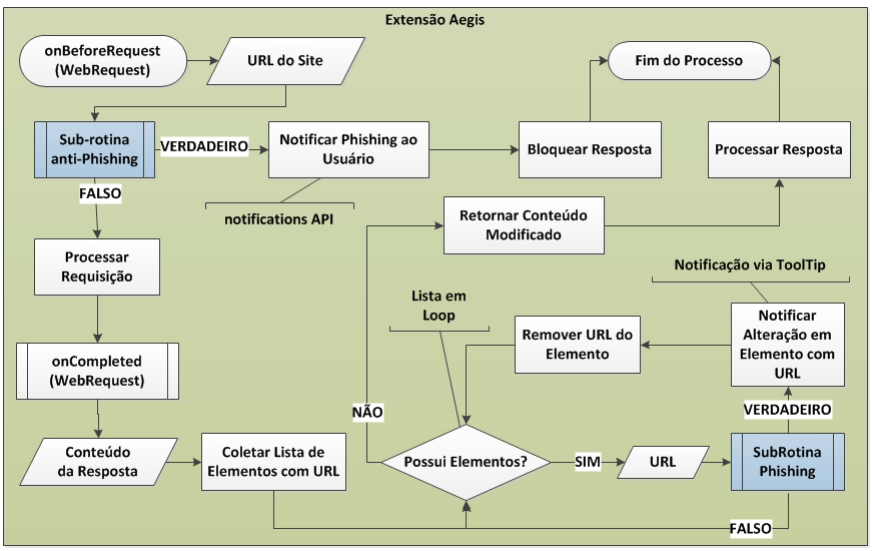

Figura 3.10 Fluxograma do Redirecionamentos ou Encaminhamentos Inválidos (A4).

3.3.5 Filtro A5: Ataques baseados em A5

A filtragem de complementos maliciosos é baseada em um mecanismo que previne que o usuário instale, a partir de seu navegador, algum plug-in ou extensão maliciosa, que muitas vezes são hospedados em sites classificados como não confiáveis. Essa filtragem divide-se em dois fluxogramas: o primeiro descreve a prevenção de Malware no momento em que o mesmo está sendo transferido para o usuário, mediante download do arquivo malicioso, conforme ilustrado na Figura 3.11.

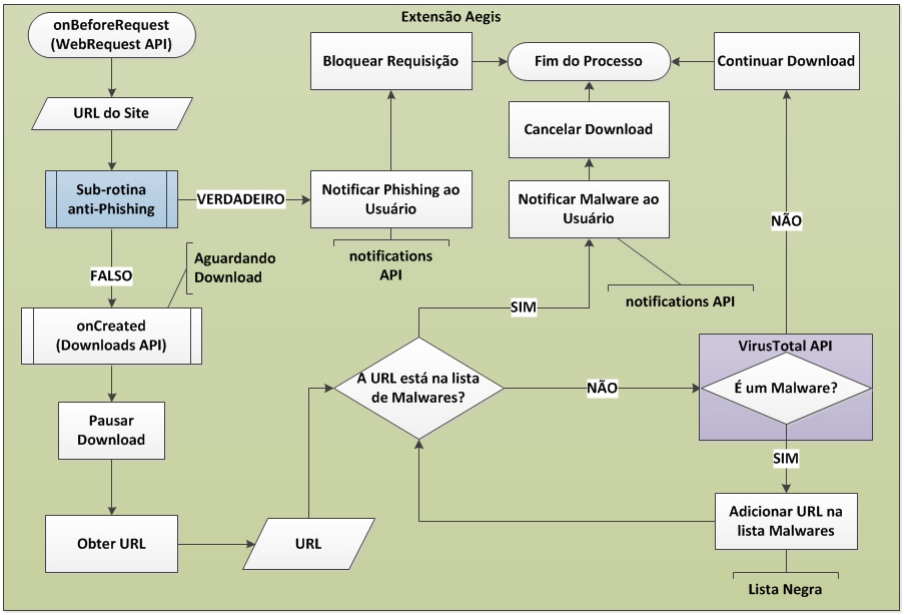

Figura 3.11 Primeiro processo de Filtragem de Complementos Maliciosos (A5).

A primeira etapa desta filtragem é verificar se o site do plug-in ou extensão em questão é classificado como um site mal-intencionado, através da sub-rotina anti-Phishing, resultando em bloqueio ou autorização de uma requisição. A segunda etapa é detectar algum download de Malware, detecção que se faz possível com uma API do navegador denominada "downloads", usando o método onCreated, que é disparado no momento que um download for inicializado. Além de oferecer o gerenciamento dos downloads em execução, ela também provê funcionalidades como captura do nome e da URL do arquivo em questão. De posse da URL, a extensão irá pausar a transferência do arquivo e analisará a URL no serviço VirusTotal.

Uma vez o conteúdo considerado sem perigo, a transmissão do download terá procedimento, caso contrário o download é cancelado. A extensão também possui uma lista negra de Malwares, analisados anteriormente, seguindo o mesmo mecanismo da

lista de Phishing, proporcionando os similares benefícios em relação a latências. O segundo fluxograma, descrito na Figura 3.12, ilustra o processo de filtragem e remoção das extensões e plug-ins já existentes antes do Aegis ter sido instalado.

O processo é realizado através da API do navegador intitulada "management". Ela proporciona a detecção das informações de todas as extensões instaladas no navegador pelo método getAll, como também desabilita ou desinstala uma extensão pelo método setEnable e uninstall, respectivamente. Além de extensões, essa API também consegue controlar aplicativos interativos ou temas instalados no navegador. As Extensões que possuam comportamentos suspeitos, como ter acesso a cookies e demais dados sensíveis, estarão sujeitas aos critérios dos filtros, condicionando suas funcionalidades à liberação manual do usuário.

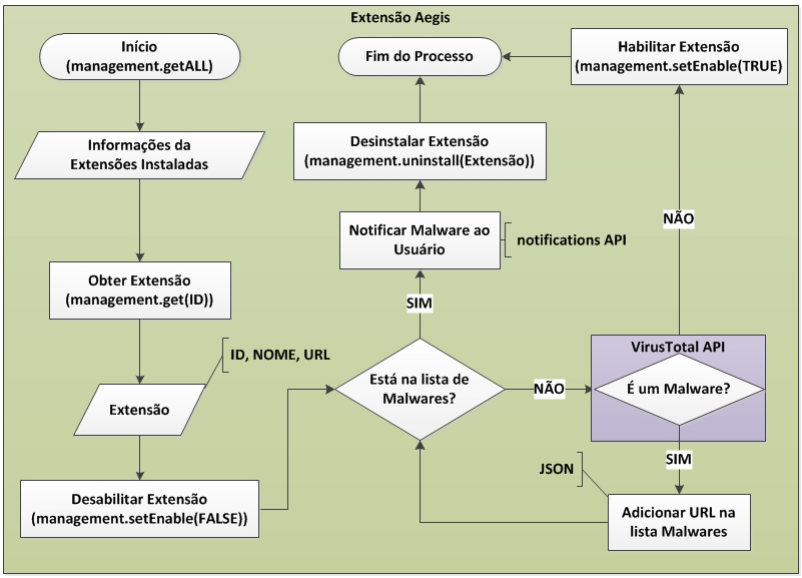

Figura 3.12 Segundo processo de Filtragem de Complementos Maliciosos (A5).

3.3.6 Filtro A6: Ataques baseados em A6

Este é o ataque mais explorado na literatura, e não seria diferente demandar funcionalidades com um maior número de questões e comportamentos inseguros a serem considerados. Ao todo são 3 fluxos distintos, descritos na Figura 3.13, 3.14 e 3.15, para a filtragem com o intuito de minimizar os ataques relacionados à exposição de dados sensíveis, que são falhas não intencionas ocorridas no processo de desenvolvimento da aplicação. O primeiro fluxo, apresentado na Figura 3.13, é dedicado às questões de armazenamento de dados sensíveis através de recursos do HTML5, que irá capturar o conteúdo da página a

ser exibida ao usuário, através do método onCompleted do WebRequest.

Uma vez que um mesmo computador possa ser compartilhado por diversas pessoas, uma aplicação que utiliza a API WebStorage [19] de forma imprudente, como armazenar dados sensíveis, como número de cartões de crédito ou estado de sessões, através do LocalStorage [19], acaba comprometendo a segurança dos dados de seus usuários. Diferente de Cookies, o LocalStorage não criptografa os dados, portanto não deve ser utilizado para armazenar informações sensíveis que deveriam constar apenas no escopo de sessão da aplicação, ou seja, no SessionStorage [19], que garante a exclusão dos dados no momento em que a sessão for encerrada.

A estruturação das informações no LocalStorage são baseadas em parâmetros com chave e valor, conforme descrito na especificação. Uma vez detectada a utilização suspeita, a Extensão irá bloquear o recurso utilizado. Da mesma forma do WebSQL descrito no filtro A2, os dados do LocalStore também podem ser explorados, caso o site em questão tenha alguma vulnerabilidade de XSS. O mesmo vale para armazenamento utilizando o IndexDB OWA (2013c). Esses recursos são acessíveis pela Extensão da API storage.

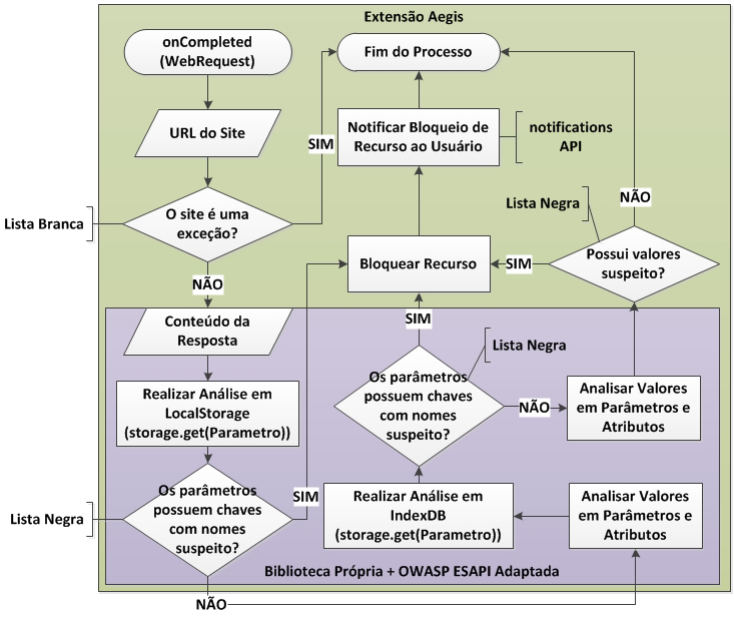

Figura 3.13 Primeiro processo de Filtragem de Exposição de Dados Sensíveis (A6).

O segundo fluxo, apresentado na Figura 3.14, é voltado para as questões de transmissão de conteúdos através de Cookies e cabeçalhos HTTP e a utilização de HTTPS. A filtragem de exploração de dados sensíveis é baseada na biblioteca OWASP ESAPI, que possui checagens e prevenções deste tipo de cenário, analisando o tráfego desse

tipo de dados. O objetivo é verificar se um determinado tráfego do cabeçalho HTTP é considerado inseguro como, por exemplo, exposição de cookies persistentesOWA (2013e). Outro exemplo é checar se um determinado serviço trafega sem utilizar HTTPS, ou WSS para WebSocket, para dados de um usuário com parâmetros críticos, como authorization.

A checagem do protocolo de comunicação se faz possível na própria URL, e a captura do cabeçalho é realizada utilizando o método onBeforeSendHeaders do WebRequest. Além disso, uma vez que o navegador identifica que o certificado HTTPS de um determinado site foi expirado, muitas vezes os usuários declaram consentimento e desejam prosseguir o processo, tornando-se vitimas em potencial para ataques Man-in-The-Middle, em que um atacante pode interceptar a comunicação ou mesmo se passar por um site ou serviço remoto.

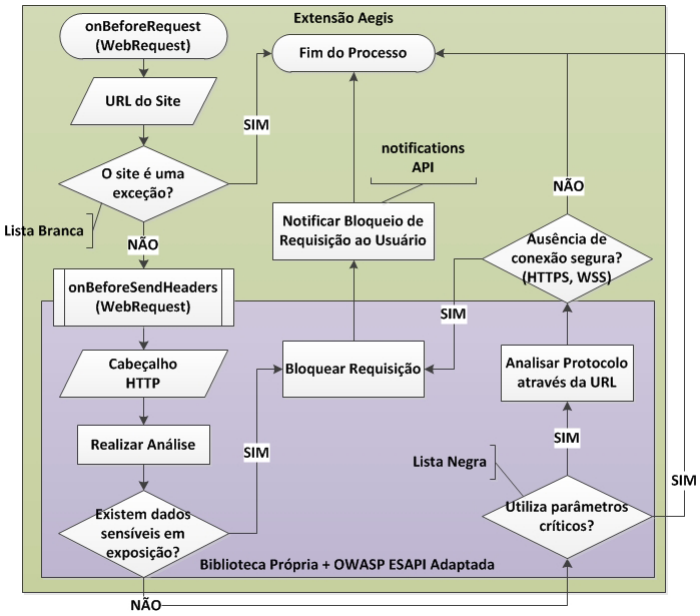

Figura 3.14 Segundo processo de Filtragem de Exposição de Dados Sensíveis (A6).

E por fim, o terceiro fluxo, apresentado na Figura 3.15, busca minimizar comportamentos perigosos, principalmente quando são realizados em um computador compartilhado por diversos usuários, muito comum em locais públicos como em laboratórios ou as conhecidas Lan Houses. A filtragem é baseada em buscar elementos na tela que manipulem os dados sensíveis do usuário, considerando dois aspectos: responsáveis em prover proteções aos dados do usuário, e que possam ocasionar armazenamento inseguro.

Aprofundando mais aos detalhes sobre as funcionalidades nesses aspectos, no primeiro caso a extensão realizará uma busca pelo botão "deslogar", que em algumas aplicações há variações no termo, como "logout", "sair", entre outros. Esse tipo de comportamento

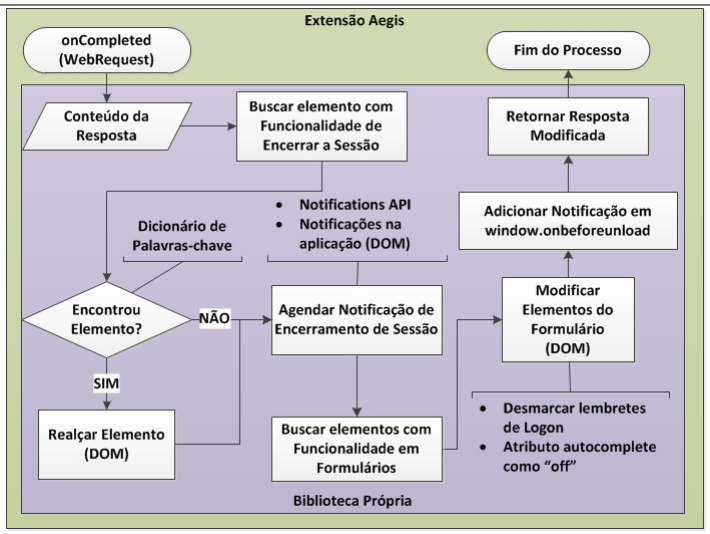

Figura 3.15 Terceiro processo de Filtragem de Exposição de Dados Sensíveis (A6).

será tratado em um dicionário de palavras-chave com possíveis variações para o link ou botão "deslogar". Uma vez identificado o elemento, a Extensão irá realçar o botão ou link em questão e se programar para notificar ao usuário, sugerindo ao mesmo que acione o elemento sempre que o navegador for fechado abruptamente. Caso esse elemento não seja encontrado, a Extensão também irá agendar uma notificação ao usuário, para que ele certifique-se que efetuou o devido logon.

A Notificação será disparada quando o navegador ou a aba for fechada no método window.onbeforeunload via JS que será adicionada através de DOM pela Extensão e também será exibida através da API notifications, que utiliza o sistema operacional para efetuar notificações do systray. No segundo aspecto é realizada uma análise de elementos, muitas vezes representados como um checkbox, que realiza a criação de um cookie na máquina para lembrar a autenticação em acessos futuros. O mais agravante é que muitas aplicações insistem em trazer para o usuário essa opção já selecionada por padrão, e geralmente a mesma situa-se após o botão de confirmar a operação, fazendo que em alguns casos este elemento passe despercebido ao usuário.

Nesse mesmo contexto, outro vetor é o salvamento automático no navegador de informações sensíveis inseridas em formulários. A prevenção está em identificar campos como login, email, número do cartão de crédito, e principalmente, o elemento do tipo password. Identificado esses elementos, será analisado se esses possuem o atributo autocomplete definidos como "off". Uma vez esse atributo ausente ou definido como on, o navegador irá questionar ao usuário se ele deseja que essas informações sejam armazenadas, e alguns casos o mesmo poderá confirmar, podendo acarretar em um acesso

indevido por outra pessoa que utilizar o navegador no mesmo computador.

De posse do conteúdo de resposta, a extensão, pela ação "Modificar Elementos do Formulário"conforme ilustrado na Figura 3.15, irá desmarcar as opções de lembrete de autenticação, além de adicionar o atributo autocomplete com valor "off"nos elementos encontrados, tudo através de DOM, e repassará ao usuário a resposta modificada.

3.3.7 Filtro A7: Ataques baseados em A7

O processo de filtragem deste ataque visa identificar componentes, que são considerados componentes e complementos do navegador, com vulnerabilidades conhecidas. Em âmbito geral, os plug-ins nem sempre são atualizados automaticamente, fazendo com que muitas vezes o usuário não perceba que utiliza em seu navegador uma versão com alguma vulnerabilidade já corrigida, ou em casos mais graves, o usuário nem faz ideia que está utilizando um componente com uma vulnerabilidade difundida, porém, ainda não solucionada, que são os casos denominados de dia zero. Na Figura 3.16 é representado o fluxograma do processo de filtragem para a prevenção desse tipo de ataque. A extensão inicia o processo abrindo uma aba para a página de checagem de plug-ins do Firefox que funciona para todos os navegadores.

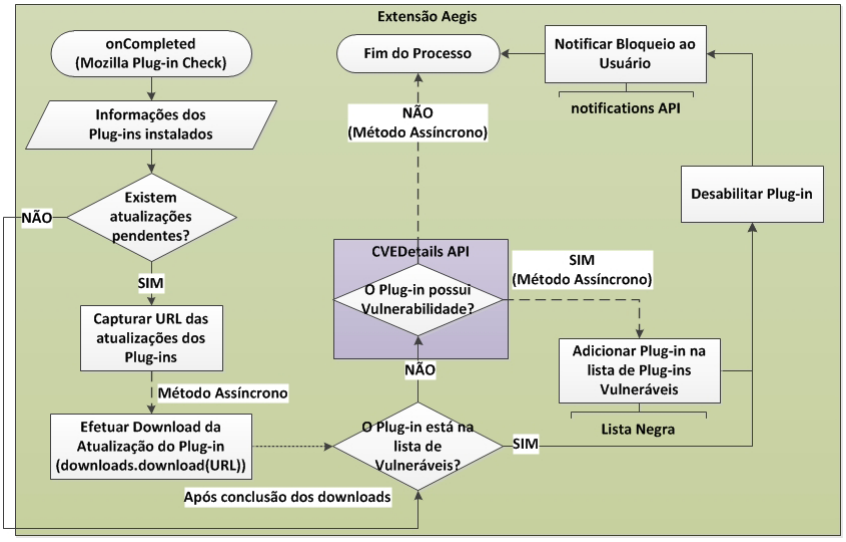

Figura 3.16 Processo de Utilização de Componentes Vulneráveis Conhecidos (A7).

Nesta página se faz possível capturar os links das atualizações disponíveis dos plug-ins desatualizados. Para identificar os casos de dia zero, a Extensão fará uma comunicação assíncrona mediante componente XMLHttpRequest para o site do CVE Details em busca de vulnerabilidades divulgadas. A consulta no CVE Details é baseada no retorno de um

arquivo JSON que armazena informações como o nome do plug-in, versão e descrição da vulnerabilidade. Conforme visto nos demais serviços externos, o resultado do CVE Details também será armazenado numa lista negra, evitando latências. A heurística do filtro trabalha da seguinte forma: se a versão do plug-in constar como vulnerável, o mesmo será desabilitado e será recomenda sua atualização.

Caso não exista ainda uma correção, situação conhecida como dia zero, o plug-in permanecerá desabilitado. Adicionalmente, para desabilitar um plug-in, a extensão irá abrir uma aba com o endereço chrome://plug-ins no caso do Chrome, ou about:plug-ins para o Firefox, que exibe uma listagem dos plug-ins instalados no navegador, descrevendo informações como a versão instalada, além de links para habilitá-los ou desabilitá-los.

3.3.8 Filtro A8: Ataques baseados em A8

O processo de filtragem do CSRF é dividido em dois fluxos, conforme apresentado nas Figuras 3.17 e 3.18. No primeiro fluxo, representado na Figura 3.17, são considerados os ataques baseados na requisição, como a manipulação em URL e cabeçalhos HTTP. Uma vez que uma URL possui seus parâmetros modificados de forma maliciosa, possivelmente o atacante utilizará serviços de URL curta para camuflar seus objetivos. Portanto o processo de prevenção se inicia logo após a análise do filtro A4, quando revela o endereço legítimo da URL. De posse da URL original, o próximo passo será analisar seus parâmetros, identificando ausências de Token baseados na técnica Synchronizer Token PatternOWASP (2013d), seja nos parâmetros da URL, para as requisições GET, ou campos com atributo HIDDEN para as requisições POST.

O próximo passo é capturar o cabeçalho HTTP e realizar uma análise quanto aos parâmetros e Cookies. A questão dos Cookies é que muitas vezes um Cookie pode ser combinado a um Token com o intuito de reforçar a legitimidade da requisição. Porém, algumas aplicações não se atentam ao atributo HTTPOnly, que tem o objetivo de garantir que seu estado não seja visível ao lado cliente. Quanto aos parâmetros, se aplica a presença dos valores em REFERER e Origin no cabeçalho HTTP. E por fim, é analisada uma possível ausência de criptografia nos valores em Tokens, visto que expor esses dados pode proporcionar vulnerabilidades, comprometendo a integridade das requisições.

O segundo fluxo, ilustrado na Figura 3.18, reforça a legitimidade avaliada pelo primeiro, que faz uma análise do conteúdo da resposta. Uma vez a URL considerada confiável, o próximo passo será capturar o conteúdo de resposta e analisar elementos que contenham comportamentos suspeitos, como por exemplo, elementos <form>, <a>, <img>, entre outros com atributos SRC definidos com URL dinâmicas, bem como

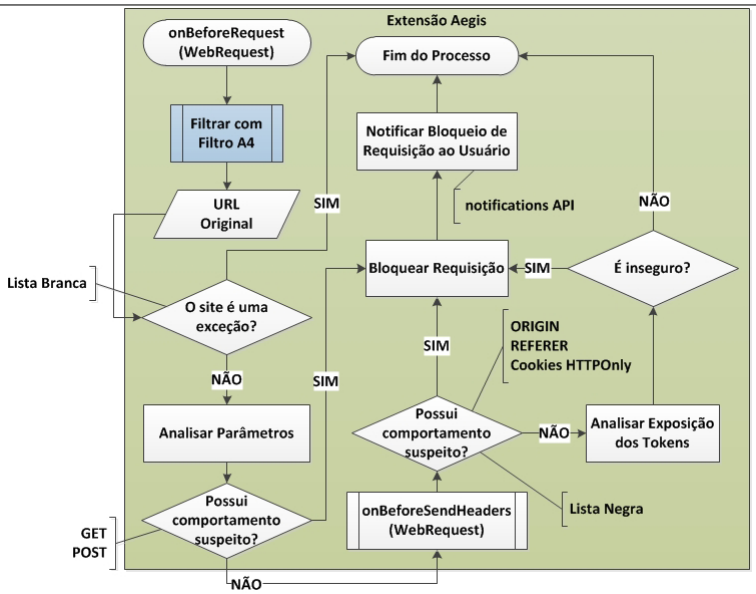

Figura 3.17 Primeiro processo de Filtragem de CSRF (A8).

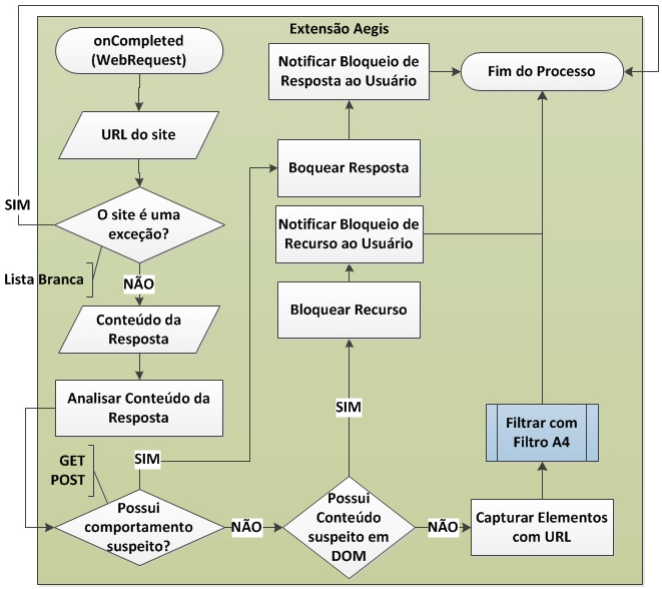

Figura 3.18 Segundo processo de Filtragem de CSRF (A8)

<iframes> e seus elementos filhos, considerando também as tentativas de construção destes elementos por DOM. Adicionalmente as URL apontadas aos serviços externos, as quais não são capturadas pelo WebRequest, por serem disparadas enquanto a página é carregada, serão capturadas e submetidas a uma filtragem usando o filtro A4.

3.3.9 Filtro A9: Ataques baseados em em A9

A filtragem do A9 será realizada mediante análise de requisições para domínios externos, a fim de evitar exploração em falhas de PMO ou CORS. Serão analisados aspectos como: cabeçalhos de origem, técnicas de Cookie TossingSecurity (2013a), e XSSI, observando consumo de serviços de Mashups ou APIs, que podem realizar injeção de XSS quando a aplicação consumidora não realizar uma devida prevenção no conteúdo de resposta. Tentativas de quebras da PMO, como por exemplo, utilizando recursos como JSONP em combinação com outras técnicas, também serão avaliadas. Na Figura 3.19 é ilustrado o fluxograma do processo de filtragem por falhas em PMO.

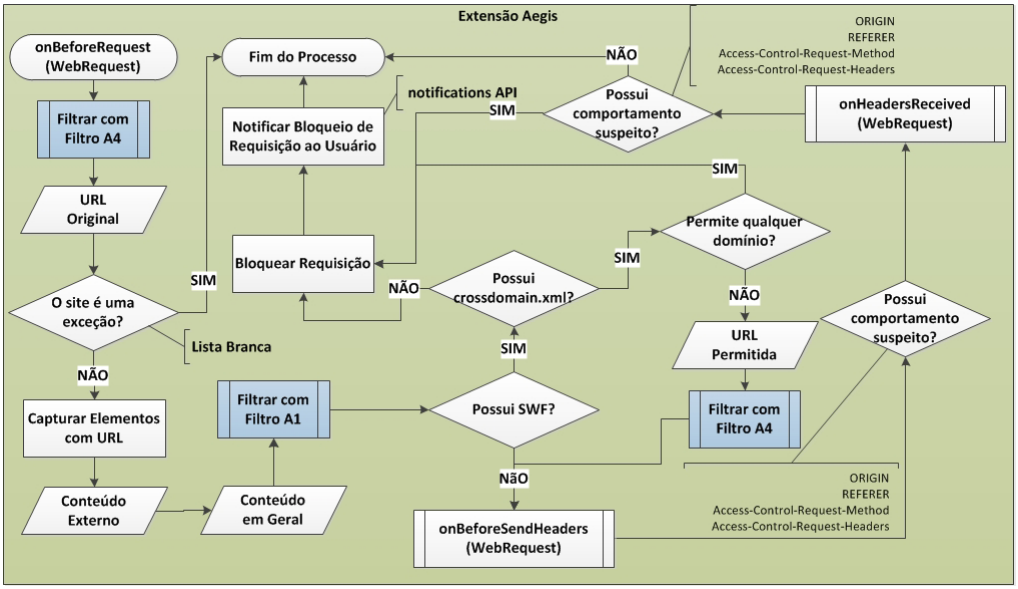

Figura 3.19 Processo de Filtragem de Falhas em Políticas de Mesma Origem (A9)

Identificado esses comportamentos, o Filtro irá acionar o filtro A4 para as prevenções de fraudes, e o filtro A1 para análise de XSSI. Tais comportamentos são oriundos de Mashups maliciosas. Outro aspecto de prevenção é uma página que possua arquivos SWF e que em suas definições permita acesso a qualquer domínio. Estas definições são armazenadas no arquivo crossdomain.xml e a extensão acessará este arquivo e analisará seu conteúdo. Uma vez encontrada a tag <allow-access-from domain="*"/>, a extensão considerará a página como suspeita. Além disso, essa prevenção impede consequências como CSRF. É importante ressaltar que esses aspectos serão analisados tanto em conexões HTTP como também por meio de WebSocket [30].

O processo de filtragem faz uso de recursos já realizados em filtros anteriores, como a checagem do cabeçalho de origem, descrita nas prevenções de CSRF. Outro reuso é

com relação à análise de URLs. Uma vez o cabeçalho sendo avaliado como confiável, o próximo passo será analisar o conteúdo de resposta, buscando por respostas de elementos que realizam requisições em serviços externos.

3.4 Limitações e Considerações Finais

O desenvolvimento da extensão, bem como seu comportamento, é bastante similar entre os navegadores Chrome e Firefox. Em contrapartida, o desenvolvimento para o IE apresenta-se como algo bastante trabalhoso, que requer linguagem compilável e dispõe de pouca documentação. Portanto, até o presente momento não foi possível concluir uma versão funcional para o IE, bem como submetê-la aos testes de avaliação. Vale ressaltar que, pelo fato de ser uma proteção para o lado cliente, esta ferramenta sempre estará sujeita aos fatores intrusivos da ação humana.

O fato é que certas funcionalidades irão depender diretamente do comportamento do usuário, já que são suscetíveis a intervenções equivocadas, como considerar um site fraudulento como confiável ou mesmo desabilitar a ferramenta durante a sua navegação. Em contrapartida, esta proposta apresentou uma solução focada em uma ferramenta leve, com funcionalidades objetivas e fundamentadas, e, ao mesmo tempo, que interage com técnicas e serviços de terceiros com bastante respaldo quantos aos problemas.

A proposta visa proteger o usuário de falhas oriundas de desenvolvimentos imprudentes em aplicações, como também protegê-lo de falhas internas do navegador e seus componentes, além de identificar explorações baseadas em engenharia social, e alertá-lo sobre possíveis fraudes. A principal característica da ferramenta é interceptar comportamentos suspeitos no conteúdo apresentado ao usuário, limitando recursos do navegador ou, caso seja preciso, bloqueando todo o processo de requisição ou resposta, reduzindo assim bastante a margem de sucesso de ataques aos seus dados sensíveis. Para os casos de sites idôneos que apresentam um comportamento suspeito, o fator da intervenção humana torna-se um benefício, já que o usuário poderá definir uma lista de sites confiáveis para cada filtro, proporcionando uma solução aos falsos positivos.

No próximo Capítulo será descrita a avaliação da ferramenta proposta, em comparativo com as ferramentas correlatas, apresentando os resultados obtidos em um conjunto de ambientes controlados.

4 Avaliação da Proposta

Este Capítulo descreve a avaliação da proposta, em comparativo com trabalhos correlatos, na busca de analisar o desempenho das ferramentas em um ambiente voltado para testes. A abordagem dessa avaliação inicia-se pela descrição da metodologia utilizada, objetivos e componentes necessários para reproduzir os cenários de testes. Em seguida são relatados os resultados obtidos quanto às vulnerabilidades conhecidas, considerando os falsos positivos. E por fim, são apresentadas as conclusões acerca dos resultados dos testes.

4.1 Metodologia

A metodologia da avaliação baseia-se na abordagem GQM (Goal, Question, Metric) proposta por Basili & Caldiera Victor R. Basili and Rombach (1994), com o intuito de proporcionar um formalismo e planejamento quanto à medição dos resultados a serem extraídos nas ferramentas submetidas aos testes. Conforme ilustrado na Figura 4.1, essa abordagem divide o processo em quatro fases: (i) Planejamento; (ii) Definição; (iii) Coleta de Dados; e (iv) Interpretação e Análise dos Resultados.

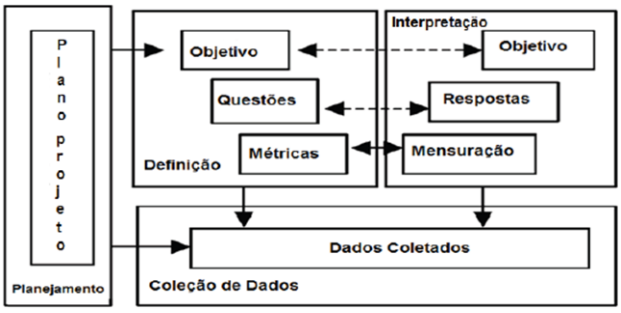

Figura 4.1 Fases da metodologia GQM.

4.1.1 Planejamento

Segundo Soligen & Berghout Van Solingen and Berghout (1999), o objetivo desta fase é definir o plano de projeto, no qual se descreve o ambiente, participantes e o que será avaliado. O primeiro passo deste estudo foi realizar uma busca por ambientes controlados disponíveis com base nas principais publicações descritas no Capítulo 2. Um ambiente controlado tem o objetivo de simular um cenário real, a fim de testar uma determinada causa e efeito. Para a realização dessa avaliação, foi utilizado o projeto OWASP Broken Web ApplicationOWA (2013d), OBWP, que é mantido pela OWASP e apresenta-se como um conjunto de aplicações que visam auxiliar no estudo de vulnerabilidades em aplicações.

O estudo também detectou a existência de outras aplicações que não fazem parte do projeto do OWASP, mas se encontram no contexto deste estudo. Como triagem inicial, os únicos critérios estabelecidos foram que a ferramenta cobrisse ao menos uma das ameaças e que tivesse seu conteúdo disponível de forma gratuita, seja por download ou acesso online. Na Tabela A.3 são listadas as 28 aplicações resultantes da triagem inicial e sua respectiva cobertura quanto às ameaças abordadas neste estudo.

Analisando o resultado ilustrado na Tabela A.3, foi observado que não existe um ambiente controlado disponível que contenha os ataques A4, A5 e A7. Além disso, também não há avaliação de certos recursos do HTML5 relacionados os ataques A1, A2 e A6, assim como alguns comportamentos apresentados por este estudo aos ataques A1, A3, A8 e A9, que não foram encontrados nos ambientes disponíveis.

Baseando nessas conclusões, este estudo desenvolveu uma aplicação intitulada "Aegis Web Threats"ou AegisWT, uma aplicação J2EE com vulnerabilidades intencionais que utiliza o framework HTML5 attack HTM (2013), desenvolvido pela AppSec Lab[3], na qual traz cobertura em todos os ataques e reúne comportamentos aderentes ao tema proposto. O objetivo da criação dessa aplicação foi atender às ameaças não cobertas pelos ambientes disponíveis, possibilitando a plena avaliação desta proposta.

Adicionalmente, é pretendido submeter essa aplicação para o projeto OBWP para que tal aplicação faça parte do projeto e assim possa ser disponibilizada para outros estudos, fazendo exemplo das aplicações Google Gruyere[4], Peruggia[5] e BodgeIt[6].

De posse das aplicações disponíveis, o próximo passo foi realizar uma triagem mais

[3]AppSec Application Security Labs: https://appsec-labs.com
[4]Google Gruyere: http://google-gruyere.appspot.com/
[5]Peruggia: http://peruggia.sourceforge.net/
[6]BodgeIt: https://code.google.com/p/bodgeit/

criteriosa com o objetivo de selecionar as aplicações mais aderentes ao tema, selecionando assim as que serão utilizadas na avaliação. Essa seleção foi baseada em dois critérios, a saber: (i) a aplicação com maior número de ataques relacionados; e (ii) a aplicação com atualizações mais recentes, considerando o semestre e ano. Como critério de desempate, foi considerada a ferramenta com maior período de atividade e documentação disponível.

Seguindo os critérios anteriormente mencionados, a aplicação selecionada foi a OWASP WebGoatOWA (2013a) , porque além de se tratar de uma aplicação com mais de 10 anos de atividade, também é rica em documentação. De posse dessas aplicações, o ambiente controlado contempla assim as vulnerabilidades necessárias para reproduzir todos os ataques relacionados neste estudo.

4.1.2 Definição

As abordagens de Soligen & Berghout Van Solingen and Berghout (1999) expõem que é na fase da definição que devem ser descritos os objetivos, questões e métricas, seguindo a estrutura conforme ilustrada na Figura 4.2 e desenvolvido na Tabela 4.1. Os ambientes controlados irão disponibilizar páginas vulneráveis nas quais o participante, a Aplicação Teste, aguardará um determinado comportamento, resultante de um ataque bem sucedido. Nesse momento, as Extensões, que representam o contexto, deverão tomar alguma ação que impeça o ataque de ser realizado, seja bloqueando recursos do navegador, cancelando a requisição ou resposta ou mesmo solicitando uma confirmação ao usuário para continuar o processo.

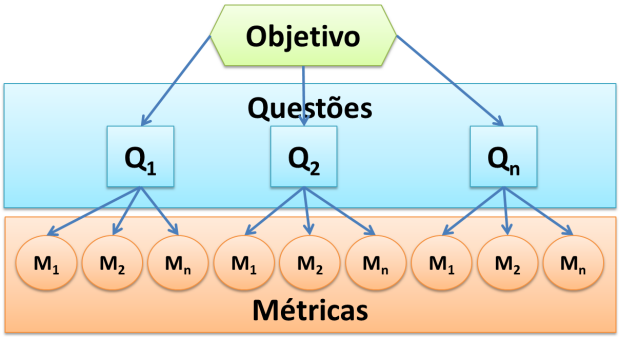

Figura 4.2 Estrutura da Fase de Definição

O objetivo esperado é que a Extensão modifique o comportamento final, resultando em algo que não seja esperado pela Aplicação Teste, ou seja, impedindo assim a reprodução do ataque. As métricas são baseadas nas principais técnicas de ataques descritas neste estudo. A escala foi definida de acordo com a complexidade do ataque e prevenção de

cada métrica atingida pelas Extensões.

Tabela 4.1: Definição das Questões e Métricas para avaliação

Questões	Descrição	Métricas	Contexto	Escala
[Q1] Quão eficaz é a proteção das ferramentas diante dos ataques A1?	Os testes de XSS serão divididos em 5 etapas, a saber: (1).[WebGoat] A Aplicação Teste acessará uma página na qual injetará o comando »<script>alert('XSS');</script> no campo do formulário e irá submetê-lo. Posteriormente a Aplicação Teste irá visualizar o resultado, esperando uma janela de alerta, indicando que o ataque foi realizado com sucesso. (2).[WebGoat] A Aplicação Teste acessará uma página na qual injetará o comando <script>alert('XSS');</script> no campo do formulário, que ao ser submetido fará uma busca por palavras-chave na base de dados. A Aplicação Teste irá esperar a janela de alerta na resposta, indicando sucesso no ataque. (3).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui um método JS que insere uma string com valor "Olá Mundo"na página. Esta string possui o comando replace combinada a uma expressão regular maliciosa, conforme o seguinte código: "Olá Mundo".replace(/Olá Mundo/g,alert). A Aplicação Teste irá esperar a janela de alerta quando a página for carregada. (4).[Aegis Web Threats] A Aplicação Teste acessará uma página que irá carregar um conteúdo malicioso que será interpretado apenas no carregamento da página, por meio de DOM, o qual irá reproduzir mediante document.write o seguinte código: <script>alert('XSS');</script>. A Aplicação Teste irá esperar a janela de alerta quando o carregamento da página for concluído. (5).[Aegis Web Threats] A Aplicação Teste acessará uma página que irá carregar um arquivo .js através de WebWorker. O conteúdo do arquivo possui o comando document.write(<script>alert('XSS');</script>);. A Aplicação Teste irá esperar a janela de alerta quando o carregamento da página for concluído.	(M1).XSS Persistente Padrão (sem escapes). (M2).XSS Refletido Padrão (sem escapes). (M3).XSS através de escapes com expressão regular. (M4).XSS através de DOM. (M5).XSS baseado em DOM através de Web Workers	(1).Aegis (2).ImmuniWeb Self-Fuzzer (3).Netcraft Extension (4).No-Script (5).RightClikXSS (6).ScriptSafe (7).TamperMonkey (8).Websecurify (9).XSS chef (10).XSS Me	Métricas atingidas 0 - Não se aplica 1 - Fraco 2 a 3 - Razoável 4 - Satisfatório 5 - Excelente

[Q2] Quão eficaz é a proteção das ferramentas diante dos ataques A2?	Os testes de SQLi serão divididos em 3 etapas, a saber: (1).[WebGoat] A Aplicação Teste acessará uma página que possui vulnerabilidade SQLi através de concatenação de parâmetros numéricos. Ela irá manipular os parâmetros do POST, aplicando uma verdade absoluta na sintaxe do SQL. A aplicação irá aguardar o carregamento da página, indicando que o acesso à página foi permitido. (2).[WebGoat] A Aplicação Teste acessará uma página na qual possui vulnerabilidade SQLi através de concatenação de parâmetros texto. Ela irá manipular os parâmetros do POST, aplicando uma verdade absoluta na sintaxe do SQL. A Aplicação irá aguardar o carregamento da página, indicando que o acesso à mesma foi permitido. (3).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui parâmetros inseguros na utilização do WebSQL. Ela irá aguardar o carregamento da página, indicando que o acesso a ela foi permitido.	(M1).Vulnerabilidade via GET (M2).Vulnerabilidade via POST (M3).Vulnerabilidade em Web SQL	(1).Aegis (2).ImmuniWeb Self-Fuzzer (3).SQL Inject Me (4).Websecurify	Métricas atingidas 0 - Não se aplica 1 - Fraco 2 - Razoável 3 - Satisfatório
[Q3] Quão eficaz é a proteção das ferramentas diante dos ataques A3?	(1).[Aegis Web Threats] A Aplicação Teste acessará uma página a qual possui um formulário malicioso com campos sobrepostos. Ela irá submeter a página e aguardar o envio concluído com sucesso, indicando que o ataque foi realizado com êxito. (2).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui links sobrepostos com elementos que no evento click redirecionam para Phishing. A aplicação irá aguardar o carregamento da página Phishing, indicando que o sequestro do clique foi realizado com êxito. (3).[Aegis Web Threats] A Aplicação Teste acessará uma página com frames cruzados e o cabeçalho com ausência do parâmetro X-Frame-Options. A permissão de acesso à página já garante ameaça ao usuário, portanto a Aplicação Teste aguardará apenas o carregamento da página.	(M1).Roubo de informações em Formulário (M2).Roubo de cliques (M3).Análise de X-Frame-Options	(1).Aegis (2).Clickjacking Reveal (3).Clickjacking Test (4).No-Script (5).Zscaler Likejacking Prevention	Métricas atingidas 0 - Não se aplica 1 - Fraco 2 - Razoável 3 - Satisfatório

[Q4] Quão eficaz é a proteção das ferramentas diante dos ataques A4?	(1).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui um elemento <a> que direciona para um Phishing. Após o carregamento da página, a Aplicação Teste irá acionar o link e espera direcionar a página para a URL, indicando que o elemento não foi bloqueado pela extensão. (2).[Aegis Web Threats] A Aplicação Teste acessará uma página na qual possui um link com uma URL curta que redireciona para um Phishing. Após 10 segundos do carregamento da página, o próximo passo da Aplicação Teste será conseguir clicar no link mencionado, encontrando o link ainda funcional.	(M1).Redirecionamento para Phishing (M2).Checagem de URL curtas	(1).Aegis (2).avast! Online Security (3).Anti-Phishing (4).Netcraft Extension (5).RequestPolicy (6).SafePreview (7).ScriptSafe (8).TamperMonkey (9).Web of Trust (WOT)	Métricas atingidas 0 - Não se aplica 1 - Razoável 2 - Satisfatório
[Q5] Quão eficaz é a proteção das ferramentas diante dos ataques A5?	(1).[Aegis Web Threats] A Aplicação Teste acessará uma página na qual disponibiliza um link para download, o arquivo em questão trata-se de um Malware e a Aplicação Teste aguardará a sua conclusão. (2).[Aegis Web Threats] A Aplicação Teste acessará uma página que está programada para receber a adição de um elemento <a> que direciona para um Phishing. A injeção deste elemento será realizada por uma Extensão instalada no navegador. A avaliação do resultado é baseada no estado do navegador, na qual será verificado se a Extensão maliciosa está desabilitada.	(M1).Download de Malware (Plug-in) (M2).Detecção de Phishing em Complemento (Extensão)	(1).Aegis (2).avast! Online Security (3).Safe Preview (4).Web of Trust (WOT)	Métricas atingidas 0 - Não se aplica 1 - Razoável 2 - Satisfatório

[Q6] Quão eficaz é a proteção das ferramentas diante dos ataques A6?	(1).[WebGoat] A Aplicação Teste acessará uma página na qual possui exposição dos dados do identificador da sessão. Esse identificador não possui uma data definida de expiração, indicando que a mesma estará sempre armazenada no navegador. A Aplicação Teste irá obter o identificador deste cookie, comprovando o acesso à página em questão. (2).[WebGoat] A Aplicação Teste irá acessar uma página que contém um formulário com dados sensíveis sem criptografia, utilizando codificações fracas como Base64. A Aplicação Teste irá aguardar a conclusão da requisição da página. (3).[Aegis Web Threats] A Aplicação Teste irá acessar uma página que armazena dados sensíveis através do LocalStorage. O objetivo da Aplicação Teste será recuperar um parâmetro que armazena o ID da sessão do usuário, indicando que o recurso, apesar do comportamento inseguro, ainda encontra-se disponível. (4).[Aegis Web Threats] A Aplicação Teste irá acessar uma página na qual realiza a autenticação por meio do parâmetro Authorization do tipo Basic. O objetivo da Aplicação Teste será conseguir efetuar logon mesmo que o protocolo em questão não seja HTTPS. (5).[Aegis Web Threats] A Aplicação Teste irá acessar uma página que armazena dados sensíveis através de IndexDB. O objetivo da Aplicação Teste será recuperar um parâmetro que armazena o ID da sessão do usuário, indicando que o recurso, apesar do comportamento inseguro, ainda encontra-se disponível. (6).[Aegis Web Threats] A Aplicação Teste irá acessar uma página na qual realizará um logon em um formulário. Em seguida a Aplicação Teste irá fechar o navegador com a sessão ativa. Ao final do teste se a instância do navegador estiver ativa, significa que o processo foi realizado de forma insegura. (7).[Aegis Web Threats] A Aplicação Teste irá acessar uma página na qual possui um formulário para autenticação. Os dados permitem armazenar as informações imputadas, além de por meio de padrão traz marcada a opção para lembrar a autenticação. Uma vez submetido o formulário de autenticação com os comportamentos inseguros, a Aplicação Teste irá considerar a operação realizada de forma insegura.	(M1).Utilização de Cookies Persistentes (M2).Armazenamento indevido baseado codificação Base64 (M3).Armazenamento indevido através do LocalStorage (M4).Autenticação Basic sem HTTPS (M5).Armazenamento indevido através do IndexDB (M6).Fechamento do navegador com sessão aberta (M7).Campos com autocompletar habilitados e lembretes de Logon	(1).Aegis (2).Netcraft Extension (3).No-Script (4).RequestPolicy (5).ScriptSafe (6).TamperMonkey (7).Toogle Cookies (8).Websecurify	Métricas atingidas 0 - Não se aplica 1 a 2 - Fraco 3 a 4 - Razoável 5 a 6 - Satisfatório 7 - Excelente

[Q7] Quão eficaz é a proteção das ferramentas diante dos ataques A7?	(1).[Aegis Web Threats] A Aplicação Teste irá acessar uma página a qual possui um link para instalar um plug-in com vulnerabilidade conhecida. O resultado final será conseguir instalar o referido plug-in no navegador. (2).[Aegis Web Threats] Primeiramente será instalada no navegador uma Extensão com uma versão com vulnerabilidade. Posteriormente a Aplicação Teste acessará uma página em branco. O resultado final será observar se a extensão foi removida.	(M1).Plug-in instalado com vulnerabilidade conhecida (M2).Extensão instalada com vulnerabilidade conhecida	(1). Aegis	Métricas atingidas 0 - Não se aplica 1 - Razoável 2 - Satisfatório
[Q8] Quão eficaz é a proteção das ferramentas diante dos ataques A8?	(1).[WebGoat] A Aplicação Teste acessará uma página que possui um formulário de envio de email no qual é possível injetar um elemento <img> que realiza uma requisição através do atributo SRC com parâmetros modificados. A Aplicação Teste interpretará o ataque como concluído caso seja possível submeter o formulário. (2).[WebGoat] A Aplicação Teste acessará uma página a qual possui um formulário de envio de e-mail no qual controla a requisição através de um Token. O Token em questão está armazenado de forma insegura, como um campo hidden com seu valor exposto no código fonte da página. A Aplicação Teste interpretará o ataque como concluído caso seja possível submeter o formulário. (3).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui um método que será disparado em seu carregamento, e do qual renderizará através de DOM um elemento <img> com um SRC atribuído com uma requisição forjada. A Aplicação Teste interpretará o ataque como concluído caso a imagem seja carregada.	(M1).CSRF através de <img> (M2).ByPass em Token (M3).Forjando requisição através de DOM	(1).Aegis (2).CSRF Finder (3).No-Script (4).RequestPolicy (5).Websecurify	Métricas atingidas 0 - Não se aplica 1 - Fraco 2 - Razoável 3 - Satisfatório

[Q9] Quão eficaz é a proteção das ferramentas diante dos ataques A9?	(1).[WebGoat] A Aplicação Teste acessará uma página que possui dois links, um que aponta para o mesmo domínio e outro que fará uma requisição em um domínio externo da aplicação e que o mesmo não consta nas permissões de domínios da PMO da aplicação. A Aplicação Teste irá clicar no primeiro e no segundo link, aguardando logo em seguida a presença de um elemento DIV com ID="message". Indicando que a falha foi explorada com sucesso. (2).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui um método que será disparado um consumo via AJAX em um serviço externo. A resposta deste serviço contém um ataque de XSS que executa um alert através de codificação URI-Component. A Aplicação Teste irá aguardar como resposta a mensagem de alerta, indicando que o ataque foi bem sucedido. (3).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui um arquivo SWF. O arquivo crossdomain.xml está definido para permitir acesso de qualquer domínio externo. A Aplicação Teste irá aguardar o carregamento da página, indicando que a falha de segurança foi desprezada pela Extensão. (4).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui uma conexão via protocolo WebSocket. O "handshake"entre cliente e servidor possui falhas de PMO. A Aplicação Teste interpretará o ataque como bem sucedido caso consiga recuperar a informação contida no retorno do WebSocket. (5).[Aegis Web Threats] A Aplicação Teste acessará uma página que possui uma troca de mensagens via Web Messaging. O controle de segurança para restringir domínios possui falhas. A Aplicação Teste interpretará o ataque como bem sucedido caso consiga recuperar a mensagem de retorno.	(M1).Proteção de PMO (M2).Aplicando XSSI via AJAX (M3).Checagem do arquivo crossdomain.xml (M4).Falhas de PMO através de WebSocket (M5).XSS através de Web Messaging	(1).Aegis (2).RequestPolicy (3).Websecurify	Métricas atingidas 0 - Não se aplica 1 - Fraco 2 a 3 - Razoável 4 - Satisfatório 5 - Excelente

4.1.3 Coleta de Dados

A coleta de dados foi realizada com base nas questões e métricas definidas na seção Descrição. As métricas foram obtidas por meio da execução de Testes Funcionais que, na Engenharia de Software, são responsáveis por realizar uma inspeção dos componentes de um determinado sistema com o objetivo de analisar resultados esperados. A técnica valida se o comportamento e funcionalidades do sistema estão de acordo com os objetivos definidos Mateus Felipe Tymburibá Ferreira (2012). Como o objetivo é a proteção do lado cliente, os comportamentos internos dos sites não são visíveis à extensão, portanto o método de avaliação será baseado em testes funcionais, também conhecidos como caixa preta Mateus Felipe Tymburibá Ferreira (2012).

A participante dessa avaliação foi uma aplicação, denominada Aplicação Teste, que executa as métricas para avaliar o comportamento de uma Extensão instalada no navegador. Cada extensão foi avaliada individualmente, ou seja, quando uma extensão era avaliada, as demais ficavam desabilitadas no navegador. Os testes foram executados de forma automatizada. A automação em testes é a utilização de um software para controlar a execução dos testes da funcionalidade de outro software Mateus Felipe Tymburibá Ferreira (2012).

Quanto à adoção do processo automatizado na realização dos testes, se faz partindo do princípio que uma vez no contexto, uma Extensão, seja permitida a utilização de simulação como trabalho automatizado, isso proporciona um experimento mais quantitativo, podendo alternar variáveis e comparar comportamentos distintos, ou seja, possibilita um comparativo entre um considerável número de Extensões submetidas aos testes. Outra questão é que a confiabilidade aumenta quando é eliminado o caráter subjetivo da avaliação, no caso, a intervenção manual. E diante do contexto desse estudo não há nada que um ser humano possa fazer que a automação não seja capaz de reproduzir.

A ferramenta escolhida para proporcionar automação aos testes foi a Selenium Server[8], que oferece a realização dos testes em praticamente todos os navegadores disponíveis, fazendo uso de mecanismos que realizam operações diretamente em uma instância do navegador.

A Aplicação Teste foi desenvolvida na tecnologia J2SE e representa um conjunto de testes baseados em Testes Unitários. Testes Unitários, ou também conhecidos como Testes de Unidade, é um processo que garante que o retorno de um determinado método seja esperado pela aplicação Mateus Felipe Tymburibá Ferreira (2012). Nessa avaliação

[8]http://docs.seleniumhq.org

foi utilizado o Framework JUnit[10] como apoio na execução dos Testes Unitários. Além do JUnit, a Aplicação Teste também utiliza as bibliotecas do Selenium WebDriver[11], um mecanismo que realiza os testes definidos de forma automática em uma instância do navegador, dando suporte a praticamente todos os navegadores disponíveis.

Estes procedimentos proporcionam a reprodução dos passos do teste em uma execução em tempo real diretamente no navegador, possibilitando a integração com a Extensão Aegis previamente instalada, além de auxiliar nas manipulações de cabeçalhos e requisições, sem a necessidade de utilizar ferramentas que oferecem proxy intermediário, como o BurpSuite[12]. Na Figura 4.3 é exibido um fluxograma da execução de um Teste Unitário. Cada classe de teste possui uma série de métodos, baseados nas métricas definidas na seção de Definição, que aguardam um determinado comportamento na página.

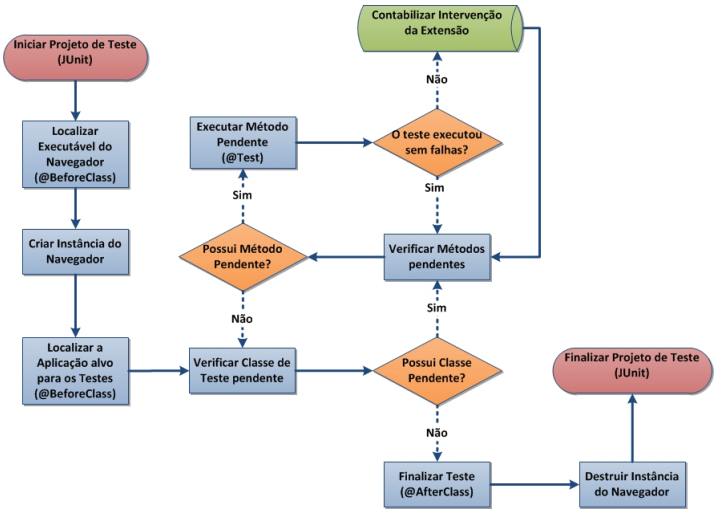

Figura 4.3 Workflow do ciclo de vida dos testes do ambiente proposto.

O primeiro passo do fluxo é a inicialização da Aplicação Teste, que irá localizar no computador o caminho absoluto do executável do navegador a ser utilizado. Uma vez localizado, o Selenium WebServer irá criar uma instância do navegador e executará os métodos dos Testes Unitários. Entretanto, uma vez o método não recuperando um comportamento esperado, causará um erro ao processo de teste, e é nesse momento que a Aplicação Teste irá contabilizar o evento, significando assim a intervenção da ferramenta. Para garantir maior precisão nos resultados, após a execução dos testes, foi realizada uma triagem no Log resultante da conclusão da Aplicação Teste, verificando assim o motivo das falhas nos métodos, para confirmar a intervenção das Extensões quanto ao

[10]JUnit: http://junit.org/
[11]Selenium WebDriver: http://docs.seleniumhq.org/projects/webdriver/
[12]BurpSuite: http://portswigger.net/burp/

comportamento do resultado. Outro ponto que vale ser ressaltado é que em nenhum dos casos, a proteção nativa do navegador interviu na reprodução dos testes.

Finalmente fica estabelecido o ambiente controlado para a Coleta de Dados, conforme a Figura 4.4 na qual descreve o cenário de extração de dados necessários para uma avaliação das Extensões submetidas aos testes. E na Tabela A.4 são listadas as Extensões que foram submetidas aos testes, com base na Tabela 2.5, com suas respectivas versões utilizadas nessa avaliação. Os testes foram realizados utilizando um Computador Desktop Core i5 com 4GB de RAM, com o Windows 7 Ultimate como sistema operacional, e uma conexão com a Internet por meio de um link da Oi Velox de 10mb. Estes recursos compõem o ambiente controlado para a execução das Questões, Q, com suas respectivas Métricas, M, com base das definições descritas na Tabela 4.1.

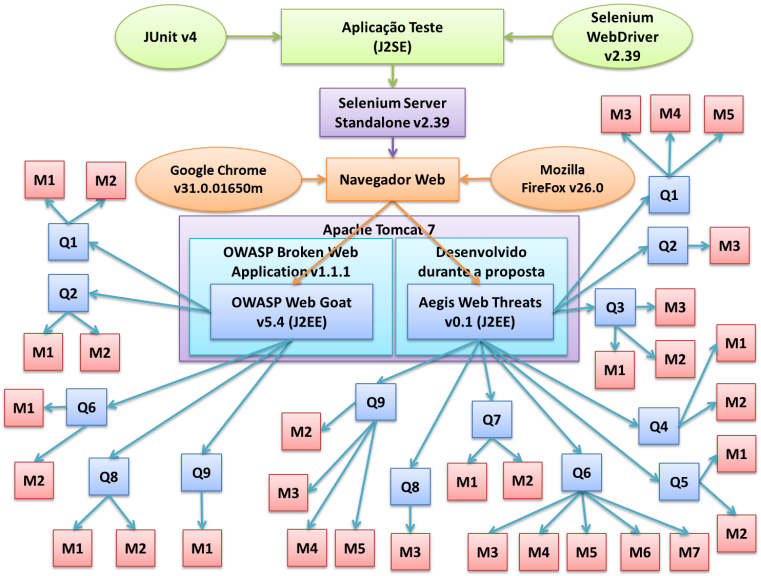

Figura 4.4 Ambiente Controlado para a Coleta dos Dados.

4.1.4 Análise dos Resultados

Com base na Tabela 4.1, foram submetidas 23 ferramentas aos testes desta experimentação, com os resultados obtidos por Ataques e as respectivas ferramentas que oferecem proteção. Pôde ser observado que em alguns casos a escala possui um nível qualificado como "Satisfatório", outros se estendem até o "Excelente". Os casos com ausência do nível "Excelente"referem-se aos ambientes de testes que, devido à falta de cenários prontos e o tempo e esforço para desenvolvê-los, não contemplam todos os aspectos vulneráveis descritos no estudo.

Em contrapartida, esses cenários abrangem os comportamentos mais elementares levantados nesta pesquisa.

Além disso, todos os aspectos pertinentes às vulnerabilidades em HTML5 estão de acordo com os cenários propostos. Outro fato que vale ser ressaltado é que os falsos positivos não foram considerados nestes testes, uma vez que o estudo atribui o fator humano responsável pelas correções dos erros de filtragem. Sendo assim os cenários apresentados foram definidos como ameaças reais ao usuário. Nesta seção serão descritas as considerações do estudo realizado com base nos resultados extraídos para as questões citadas na Tabela 4.1.

Resultado da Questão 1

Conforme observado na Figura 4.5, a ferramenta Aegis conseguiu um resultado excelente nos testes de proteção à XSS em ambos os navegadores. A ferramenta No-Script teve um resultado de mesma qualidade, porém essa solução encontra-se disponível apenas no navegador Firefox. Também foi observado que ferramentas exclusivamente disponíveis para o Chrome, como a Websecurify e a XSS Chef, tiveram resultados satisfatórios. Outra questão importante é que no geral, apenas uma ferramenta teve sua funcionalidade classificada como fraca, mas nenhuma delas teve um comportamento nulo em relação à proteção de XSS. Estes índices, combinados com a quantidade de ferramentas disponíveis demonstram que o ataque de XSS é o mais explorado quanto às soluções propostas no contexto de extensões.

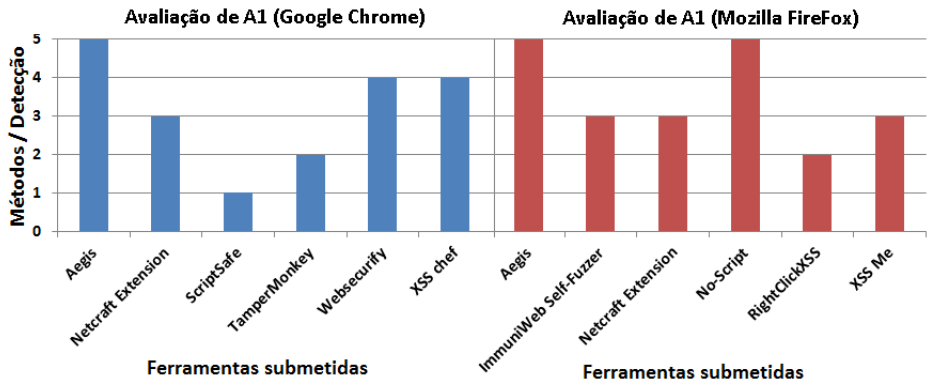

Figura 4.5 Resultados da avaliação da Questão 1.

Resultado da Questão 2

Seguindo para a análise dos resultados de SQLi, conforme ilustrado na Figura 4.6, os dados obtidos vão em contradição aos de XSS, tanto em respeito à quantidade de ferramentas disponíveis, quanto à eficácia. O fato é que as técnicas de detecção no contexto de uma extensão são bastante limitadas, primeiramente pela heurística aplicada por todas as ferramentas, as quais realizam uma inspeção baseada em um nível mais superficial, observando comportamentos como erro da aplicação ou mensagens ao usuário quando os parâmetros são manipulados. Com base nos resultados obtidos, a ferramenta Aegis apresentou-se com uma heurística com maior riqueza quanto aos comportamentos descritos. A primeira métrica do teste possui um comportamento de fácil detecção, mesmo assim uma ferramenta teve seu comportamento de proteção considerado nulo. A ferramenta Aegis se sobressaiu devido às questões de vulnerabilidade no WebSQL, as quais não são avaliadas pelas demais ferramentas.

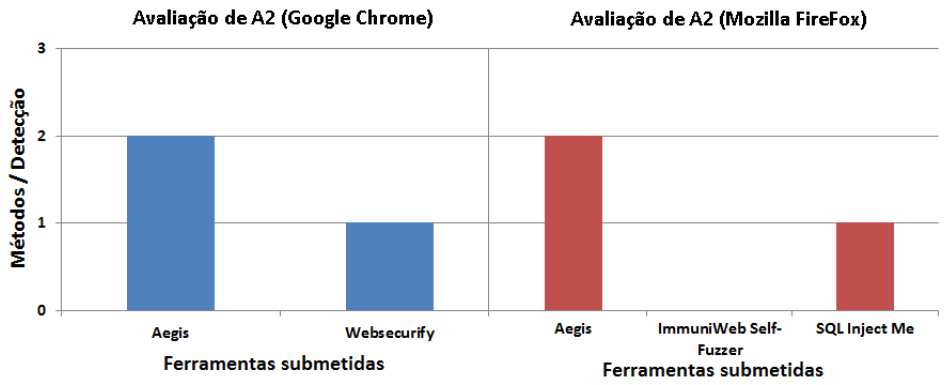

Figura 4.6 Resultados da avaliação da Questão 2.

Resultado da Questão 3

Nos dados extraídos quanto ao ataque de Clickjacking, conforme ilustrado na Figura 4.7, a maioria das soluções candidatas apresentaram um comportamento classificado como fraco em relação a sua funcionalidade de proteção, e dentre elas, uma apresentou um comportamento nulo. Em contrapartida, a ferramenta No-Script além de detectar sobreposição para furto de cliques, também obteve controle dos cabeçalhos HTTP referente a permissões de frames. Contudo a ferramenta Aegis foi à única que além desses aspectos, também cobriu a sobreposição de campos em formulários.

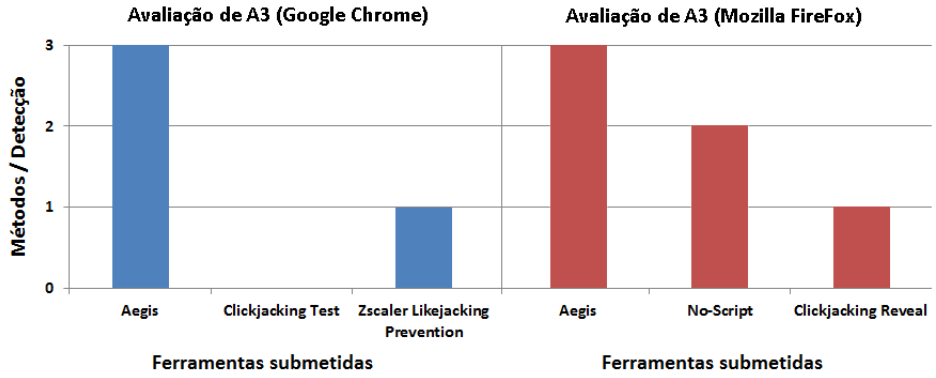

Figura 4.7 Resultados da avaliação da Questão 3.

Resultado da Questão 4

Conforme ilustrado na Figura 4.8, o destaque negativo quanto à proteção aos ataques A4. Com exceção da Aegis, nenhuma outra ferramenta foi capaz de detectar o Phishing em uma URL curta, definido como a segunda métrica desta avaliação. Tal fato possibilita que o usuário clique em um link e com isso acaba jogando a responsabilidade de prevenção do Phishing para o navegador. Dessas ferramentas, a única prevenção apresentada foi referente ao ataque de forma mais básica, descrito na primeira métrica. Ainda sim, duas ferramentas tiveram um comportamento nulo devido ao fato de serem baseadas em uma lista negra que possui um considerável tempo de depreciação. O fato foi comprovado pois os testes utilizaram Phishing catalogados e confirmados no dia anterior no PhishTank, um tempo mais que suficiente para um mal intencionado realizar ações ilícitas em centenas de usuários.

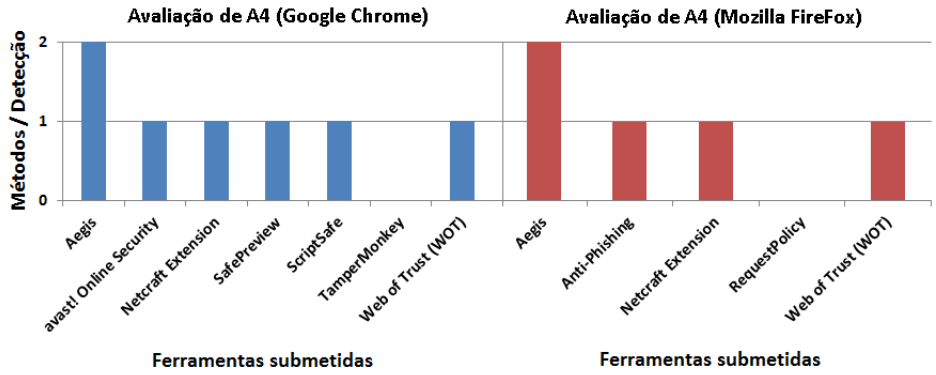

Figura 4.8 Resultados da avaliação da Questão 4.

Resultado da Questão 5

Em relação aos resultados obtidos nos testes do ataque A5, com exceção a SafePreview, que apresentou seu comportamento como nulo, as demais ferramentas conseguiram detectar o download de um Malware, conforme descrito na Figura 4.9. Porém, apenas a ferramenta Aegis conseguiu realizar a prevenção contra uma Extensão maliciosa.

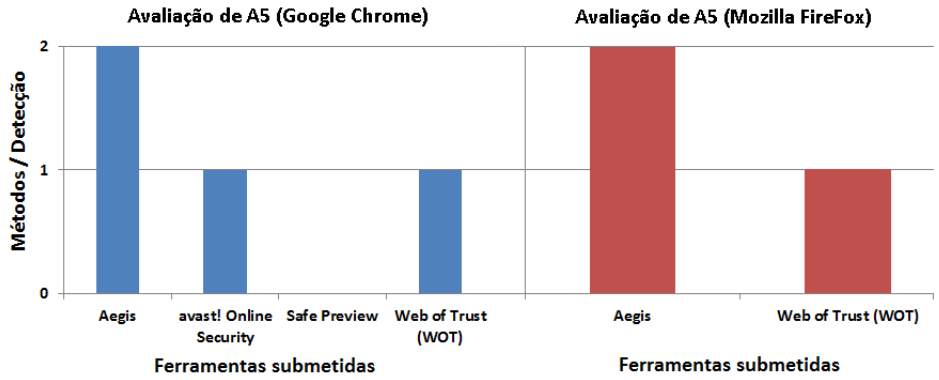

Figura 4.9 Resultados da avaliação da Questão 5.

Resultado da Questão 6

O comportamento padrão das ferramentas de prevenção disponíveis é o controle de Cookies. Conforme descrito na Figura 4.10, todas as ferramentas tiveram uma ação preventiva. A TamperMonkey observou o armazenamento inseguro em LocalStorage, porém, apenas a Aegis cobriu todas as vulnerabilidades quanto aos componentes do HTML6. Além disso, as demais ferramentas apresentaram-se com baixa cobertura quanto às observações e comportamentos inseguros descritos neste estudo.

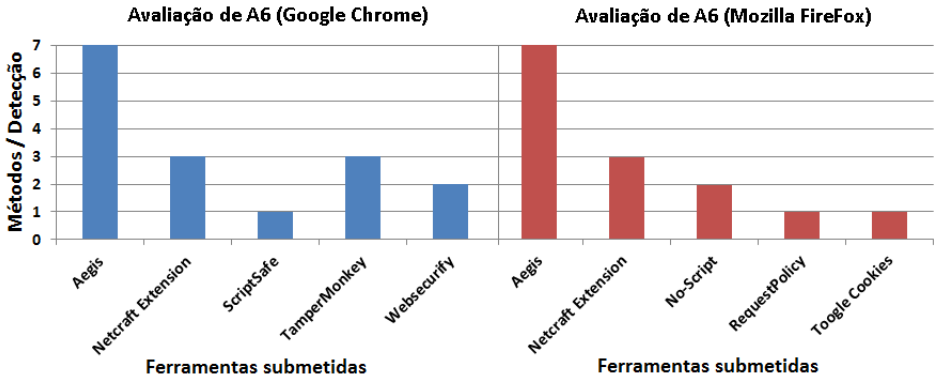

Figura 4.10 Resultados da avaliação da Questão 6.

Resultado da Questão 7

Em relação aos ataques A7, nenhuma outra ferramenta foi encontrada disponível para um possível comparativo com a Aegis. Isso também reflete na pequena quantidade de publicações disponíveis na literatura sobre esse ataque. Além disso, conforme descrito na Figura 4.11, apenas no Chrome foi possível realizar uma prevenção satisfatória, uma vez que o navegador em questão disponibiliza uma listagem de seus plug-ins instalados, por meio do endereço chrome://plugins, onde se faz possível desabilitar um determinado plug-in, o que não ocorre na listagem about:plugins do Firefox. Portanto a solução proposta ainda possui um comportamento minimamente preventivo.

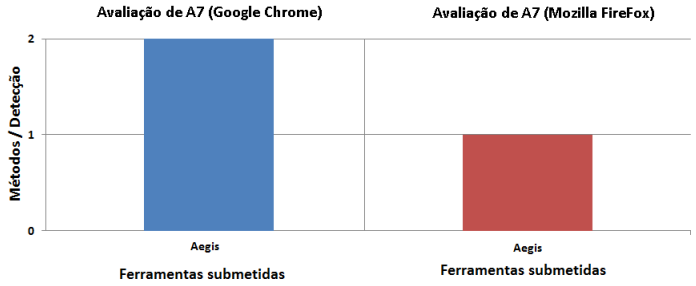

Figura 4.11 Resultados da avaliação da Questão 7.

Resultado da Questão 8

Conforme descrito na Figura 4.12, a ferramenta Aegis e a No-Script apresentaram um comportamento satisfatório nos testes que envolvem ataques de A8. As demais ferramentas tiveram ineficiência quanto a detecção da métrica 2, a qual é baseada na forja de Token.

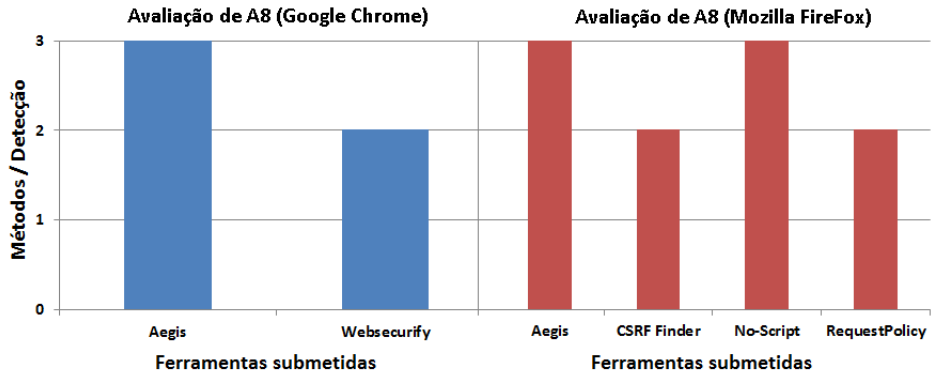

Figura 4.12 Resultados da avaliação da Questão 8.

Resultado da Questão 9

Finalmente, com base nos resultados da avaliação da Questão 9, conforme ilustrado na Figura 4.13, poucas foram as ferramentas disponíveis para as proteções de falhas em PMO, sendo detectadas apenas três ferramentas. A Aegis foi à única que considerou os critérios abordados neste estudo, a saber, a segurança em arquivos SWF e as falhas ocasionais em recursos do HTML5 como o WebSocket e o WebMessaging.

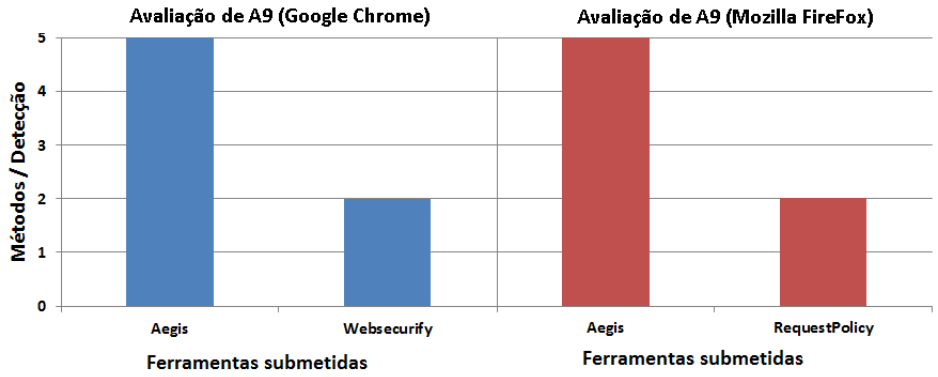

Figura 4.13 Resultados da avaliação da Questão 9.

4.2 Limitações da Avaliação

Devido às restrições de tempo, não foi possível desenvolver um ambiente controlado capaz de avaliar todas as variações dos aspectos inseguros acerca dos dados do usuário, os quais foram apresentados por este estudo, oferecendo assim um número ainda maior de métricas a serem submetidas nas questões de avaliação. Este crescimento na diversidade das métricas poderia proporcionar uma análise mais quantitativa, o que em tese garantiria uma maior precisão quanto aos resultados obtidos nos testes. Além disso, não foi possível realizar uma análise comparativa em relação ao ataque 7 devido as limitações disponíveis no âmbito das ferramentas apresentadas.

E por fim, não foi possível submeter os testes considerando a navegação através do IE. A versão atual da ferramenta proposta voltada para este navegador ainda apresenta-se em um estágio sem muitas funcionalidades, conforme descrito no Capítulo 3, na seção 3.4. Outra observação que vale ser ressaltada é que até o presente momento este navegador possui uma carência considerável quanto à soluções correlatas disponíveis, o que impossibilitaria análises comparativas.

4.3 Considerações Finais

Foram avaliadas um total de 23 extensões. Dentre as ferramentas submetidas, a Aegis apresentou resultados excelentes e satisfatórios tanto no Chrome, como no Firefox. A No-Script, para Firefox, e Websecurify, para Chrome, tiverem excelentes resultados. Contudo, com exceção apenas da Aegis, e da No-Script quanto ao Clickjacking (A3), foi observado que nenhuma das ferramentas avaliadas possui cobertura para os ataques relacionados aos aspectos de HTML5, concluindo assim que no estado atual das extensões disponíveis, não há uma proteção eficiente quanto aos aspectos mencionados. Outra observação é sobre as Extensões que oferecem um combate às fraudes, com um rico acervo disponível para o usuário.

Em contrapartida, em relação à URL curtas, com exceção da Aegis nenhuma outra ferramenta foi capaz de ao mesmo tempo em que obtem a URL original, verificar se a mesma trata-se de um Phishing. E ao contrário do grande número de ferramentas anti-Phishing disponíveis, são raras as defesas contra falhas em PMO oferecidas, e as poucas existentes carecem de boas soluções, pois não consideram diversos aspectos mencionados neste estudo.

E por fim, foi identificado que a proposta desta pesquisa, apesar de obter excelentes resultados em ambos os navegadores e uma plenitude na maioria dos testes, ainda carece de boas soluções quanto aos ataques A2 e A7. Em relação ao ataque A2, é preciso aperfeiçoar a heurística em um modelo que trabalhe em conjunto com ferramentas de análise com maior profundidade. Quanto ao ataque A7, o ponto positivo da proposta é que ela oferece uma solução que ainda não existe em nenhuma outra extensão para os navegadores mais populares. Porém, a proposta até o momento não possui uma prevenção satisfatória, indicando que precisa de uma evolução nesse quesito. O próximo Capítulo irá abordar sobre essas evoluções, bem como apontar melhorias e trabalhos futuros.

5

Conclusões e Trabalhos Futuros

Até o presente momento, e, muito provavelmente em alguns anos pela frente, as soluções Server-side ainda serão consideradas como as mais eficientes contra os vetores de ataques aos dados sensíveis do usuário na web. Em contrapartida, nesse contexto há pressuposto de que a responsabilidade da segurança dos dados está totalmente direcionada aos proprietários da aplicação disponível na web. E isso vai ao encontro de um grande dilema quanto aos serviços prestados nesses ambientes, determinando um questionamento: como garantir que os proprietários irão de fato adotar fortes políticas de segurança no desenvolvimento e manutenção de suas aplicações? Tais políticas, uma vez ausentes, podem propiciar inúmeras vulnerabilidades na aplicação Assad (2011), que podem comprometer os dados sensíveis de seus usuários.

Na mesma linha de raciocínio, outro questionamento é direcionado aos desenvolvedores de navegadores: essas ferramentas são consideradas os principais mecanismos de uso no ambiente da web. Nesse contexto, surge um questionamento: como garantir que essas ferramentas sejam disponibilizas aos usuários sem falhas que possibilitem a intervenção de mal-intencionados? Ou ainda que possam trazer consequências como vulnerabilidades em aplicações que até então não eram vulneráveis [2]. Ou mesmo resultando no comprometimento de todo o sistema operacional Kaneshige (2009). E não fosse o bastante, os usuários ainda são suscetíveis a ações de golpistas que forjam sites para roubar dados financeiros, causando enormes prejuízos Bestuzhev (2013). Esses fatos dão origem a um terceiro questionamento: Quem de fato é o responsável em prover prevenções contra essas fraudes? Os proprietários dos serviços legítimos ou os gestores de navegadores web?

A proteção client-side está longe de ser a solução mais eficaz, porém, ainda, demonstra-se necessária, pois atua como uma solução adicional, que visa minimizar os ataques relacionados às incertezas das respostas para esses questionamentos.

Este estudo apresentou uma abordagem que visa aplicar técnicas de defesa que são executadas diretamente no navegador, a fim de oferecer ao usuário uma solução preventiva intrínseca ao contexto. A proposta foi submetida a uma avaliação, na qual foi possível observar que a mesma conseguiu inibir um grande número de ataques consequentes por ausência ou falhas nas prevenções de segurança em recursos disponíveis na web usando um navegador.

As diversas técnicas aplicadas nesta proposta, a exemplo da análise estática, são as verificações mais recomendadas no nível client-side, mas em um âmbito geral essas técnicas não são as mais eficientes. Mesmo assim, a iniciativa dessa proposta torna-se válida. Primeiro porque as análises mais eficazes são as realizadas no nível de implementação, e não se pode garantir que todos os desenvolvedores irão atribuir às devidas técnicas em suas aplicações. Segundo, porque apesar de limitada, essa técnica trouxe resultados satisfatórios conforme a avaliação apresentada no Capítulo 4.

5.1 Contribuições

As principais contribuições desta dissertação foram:

- Uma revisão sistemática da literatura, apresentada no Capítulo 2, que descreveu os ataques mais comuns que resultam na violação de dados do usuário de navegador web. Essa revisão também relacionou as principais técnicas de defesa e ataque, combinadas às considerações na visão deste estudo.

- Abordagem e implementação de uma ferramenta de proteção client-side, apresentada no Capítulo 3. A proposta foi desenvolvida para interagir diretamente no navegador do usuário, visando minimizar as ameaças apresentadas. É pretendido disponibilizar sua biblioteca principal ao projeto Client-side do OWASP ESAPI.

- E por fim, no Capítulo 4, foi apresentado um ambiente controlado que proporciona uma avaliação mais ampla para as ferramentas que tenham um propósito similar ao da ferramenta proposta neste estudo. É pretendido disponibilizar este ambiente para ser incorporado ao projeto OWASP Broken Web.

Além disso, três artigos foram escritos com o objetivo de difundir parte do conhecimento produzido nesta pesquisa, os quais foram aceitos para publicação, a saber:

1) Silva, C. M. R.; Garcia, V. C. (2013) "SafeMash: Uma Plataforma como Serviço para a Composição Segura e Colaborativa de Mashups". Em: VI Workshop de Teses e

Dissertações em Sistemas de Informação, 2013, João Pessoa. IX Simpósio Brasileiro de Sistemas de Informação, 2013. v. 1. p. 19-24.

2) Silva, C. M. R., Rodrigues, R. B., Campos, G. M. M., Ferreira, W. O., Ferraz, F. S., and Garcia, V. C. (2013). "An approach for Secure Mashup Composition". Em: IADIS 2013 - International Conference WWW/INTERNET 2013 (ICWI) v.1 p.378-380.

3) Silva, C. M. R., Rodrigues, R. B., and Garcia, V. C. (2014). "SafeMash: A Platform for Safety Mashup Composition"Em: WEBIST 2014 - 10th International Conference on Web Information Systems and Technologies.

5.2 Limitações da Proposta

Devido às restrições de tempo, alguns dos problemas abordados nesta dissertação ainda estão em aberto na solução apresentada neste estudo. Além disso, algumas características da proposta carecem de uma funcionalidade mais eficaz. Eis alguns pontos que podem ser investigados como melhorias desejáveis, a saber:

- **Melhorias nas funcionalidades do Filtro A2**: Conforme descrito no Capítulo 3, seção 3.3.2, é planejada uma integração com os resultados do SQLMap, a fim de proporcionar uma análise com maior nível de profundidade, buscando precisão aos resultados.

- **Melhorias nas funcionalidades do Filtro A5**: Conforme descrito no Capítulo 3, seção 3.3.5, é planejada uma melhoria em relação à checagem de arquivos em download, uma vez que a Extensão precise pausar a transferência, a fim de coibir o término do carregamento do arquivo, antes da conclusão da inspeção, isso trouxe alguns problemas quando realizado download em servidores que não suportam reinícios.

- **Melhorias nas funcionalidades do Filtro A7**: Conforme descrito no Capítulo 3, seção 3.3.7, o filtro ainda carece de boas funcionalidades, principalmente no navegador Firefox. O estudo identificou que, devido às limitações do navegador, esse filtro talvez apresente um comportamento mais eficiente se for executado em um contexto diferente àquele proporcionado a uma extensão.

5.3 Trabalhos Futuros

Além das limitações apresentadas, também é pretendido adicionar funcionalidades até então não planejadas no escopo deste estudo, mas que garantem um desempenho mais satisfatório da ferramenta proposta:

- **Melhoria quanto aos Falsos Positivos**: O estado atual da proposta sugere o fator humano como solução para o controle de falsos positivos, porém, como evolução da ferramenta, é planejada uma realização de execuções dos filtros em Sandbox, utilizando uma análise baseada em comportamento do conteúdo, resultante da execução em isolamento.Chr (2013) Adicionalmente, também é pretendido utilizar inteligência artificial para as decisões do tipo de recurso a ser bloqueado.

- **Prevenção a ataques baseados em Esteganografia**: Um ataque relacionado à injeção de código, porém, não muito difundida na literatura, no qual o ataque é injetado em um arquivo de imagem um código malicioso capaz de ser processado pelo servidor web. O código em questão é ofuscado na imagem através de esteganografia, cifrado em algum método de codificação MIME, como o base64, e armazenado no meta-dados do arquivo. O processo é executado de forma silenciosa, visto que arquivos JPEG e PNG são bem flexíveis Suc (2013).

- **Rotinas de Notificação de Vulnerabilidades**: Será proposta uma política de notificar os proprietários do site quando uma vulnerabilidade for encontrada. Essa funcionalidade ainda não foi incrementada no estágio atual da proposta devido à limitação das regras estabelecidas quanto aos falsos positivos. Além disso, é necessário um bom planejamento quanto ao envio das notificações.

- **Melhorias no desempenho dos Filtros**: Também se planeja melhorias na extensão para o Google Chrome, considerando o fato que está sendo desenvolvido o declarativeWebRequest Dec (2013), que no momento encontra-se em fase beta. Essa nova API será uma versão melhorada do webrequest, onde oferece maiores funcionalidades e técnicas de execução em Sandbox.

- **Disponibilizar o Aegis como API para o desenvolvimento seguro**: Por fim, é planejado disponibilizar as funcionalidades do Aegis em uma API JS para auxiliar na segurança de aplicações web em nível de implementação. Isso dará maiores possibilidades de prevenção, uma vez que o proprietário da aplicação permita uma maior proximidade entre os filtros do Aegis e o servidor da aplicação.

Referências

ACCORNERO R.; RISPOLI, D. B.-F. Privacy-enhanced identity via browser extensions and linking services. **Network and System Security (NSS), 2011 5th International Conference on**, [S.l.], 2011.

ADAM BARTH, C. J.; MITCHELL, J. C. Robust Defenses for Cross-Site Request Forgery. **15th ACM Conference on Computer and Communications Security**, [S.l.], 2008.

ADAM BARTH, C. J.; MITCHELL, J. C. Securing Browser Frame Communication. **17th USENIX Security Symposium**, [S.l.], 2008.

AGGARWAL A., R. A. K.-P. PhishAri: automatic realtime phishing detection on twitter. **eCrime Researchers Summit**, [S.l.], 2012.

AKIYAMA M., Y. T. H.-T. Improved blacklisting: inspecting the structural neighborhood of malicious urls. **IT Professional**, [S.l.], 2013.

AKIYAMA M., Y. T. I.-M. Searching structural neighborhood of malicious URLs to improve blacklisting. **11th IEEE/IPSJ International Symposium on Applications and the Internet, SAINT 2011**, [S.l.], 2011.

ALAZAB, A. et al. Web application protection against SQL injection attack. **7th International Conference on Information Technology and Application, ICITA 2011**, [S.l.], 2011.

ALDWAIRI M., A. R. MALURLS: a lightweight malicious website classification based on url features. **Journal of Emerging Technologies in Web Intelligence**, [S.l.], 2012.

ALEXENKO T., J. M. R.-S. Z. W. Cross-site request forgery: attack and defense. **7th IEEE Consumer Communications and Networking Conference, CCNC 2010**, [S.l.], 2010.

ALFANTOOKH, A. An automated universal server level solution for SQL injection security flaw. **Proceedings - 2004 International Conference on Electrical, Electronic and Computer Engineering, ICEEC'04**, [S.l.], 2004.

ALI, A. et al. SQL-injection vulnerability scanning tool for automatic creation of SQL-injection attacks. **Procedia Computer Science**, [S.l.], 2011.

ALI S., K. S. R.-A. A cryptography-based approach to web mashup security. **Proceedings - International Conference on Computer Networks and Information Technology**, [S.l.], 2011.

ALVAREZ, G.; PETROVIC, S. A new taxonomy of Web attacks suitable for efficient encoding. **Computers Security**, [S.l.], 2003.

ANDERSON B.; QUIST, D. L.-T. Detecting Code Injection Attacks in Internet Explorer. **Computer Software and Applications Conference Workshops (COMPSACW), 2011 IEEE 35th Annual**, [S.l.], 2011.

ANDERSON, R.; LEE, J.-H. Jikzi: a new framework for security policy, trusted publishing and electronic commerce. **Computer Communications**, [S.l.], 2000.

ANUPAM V.; MAYER, A. Secure Web scripting. **Internet Computing, IEEE**, [S.l.], 1998.

ARMANDO, A. et al. An authentication flaw in browser-based Single Sign-On protocols: impact and remediations. **Computers Security**, [S.l.], 2013.

ARULSUJU, D. Hunting malicious attacks in social networks. **Advanced Computing (ICoAC), 2011 Third International Conference on**, [S.l.], 2011.

ASLAM B.; LEI WU; ZOU, C. PwdIP-Hash: a lightweight solution to phishing and pharming attacks. **Network Computing and Applications (NCA), 2010 9th IEEE International Symposium on**, [S.l.], 2010.

ASSAD, R. E. **SSQA Software Security Quality Assurance**. 2011. Tese (Doutorado em Ciência da Computação) — Universidade Federal de Pernambuco Centro de Informca.

AUSTIN, A.; HOLMGREEN, C.; WILLIAMS, L. A comparison of the efficiency and effectiveness of vulnerability discovery techniques. **Information and Software Technology**, [S.l.], 2013.

BALASUNDARAM, I.; RAMARAJ, E. An Efficient Technique for Detection and Prevention of Injection Attack using Based String Matching. **Procedia Engineering**, [S.l.], 2012.

BALDUZZI M., E. M. K. E. B. D. K. C. A solution for the automated detection of clickjacking attacks. **Proceedings of the 5th International Symposium on Information, Computer and**, [S.l.], 2010.

BALFANZ., D. D. E. W. F. D. S. W. D. Java security: web browsers and beyond. **Internet besieged**, [S.l.], 1997.

BALZER, R. Safe email, safe office, and safe web browser demo description. **DARPA Information Survivability Conference and Exposition, 2003. Proceedings**, [S.l.], 2003.

BAMRARA A., S. G. B. M. Cyber attacks and defense strategies in India: an empirical assessment of banking sector. **International Journal of Cyber Criminology**, [S.l.], 2013.

BANDHAKAVI, S. et al. CANDID: preventing sql injection attacks using dynamic candidate evaluations. **Proceedings of the ACM Conference on Computer and Communications Security**, [S.l.], 2007.

BARONE G.B.; MARGARITA, N. M. A. M. N. R. L. Secure access to personalized Web services. **Dependable Computing, 2001. Proceedings. 2001 Pacific Rim International Symposium on**, [S.l.], 2001.

BECK K.; ZHAN, J. Phishing in Finance. **Future Information Technology (FutureTech), 2010 5th International Conference on**, [S.l.], 2010.

BESTUZHEV, D. **Brazil**: a country rich in banking trojans. [url: http://www.securelist.com/en/analysis/204792084/brazil_a_country_rich_in_banking _trojans?print_mode=1] acesso: 10/12/13. 2013.

BIELOVA, N. Survey on JavaScript security policies and their enforcement mechanisms in a web browser. **The Journal of Logic and Algebraic Programming**, [S.l.], 2013.

BINA E. ; MCCOOL, R. . J. V. . W. M. Secure access to data over the Internet. **Parallel and Distributed Information Systems, 1994., Proceedings of the Third International Conference**, [S.l.], 1994.

BLANC G.; ANDO, R. K. Y. Term-Rewriting Deobfuscation for Static Client-Side Scripting Malware Detection. **New Technologies, Mobility and Security (NTMS), 2011 4th IFIP International Conference on**, [S.l.], 2011.

BOYAN CHEN; ZAVARSKY P.; RUHL, R. L. D. A Study of the Effectiveness of CSRF Guard. **Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom)**, [S.l.], 2011.

BOYD, S.; KEROMYTIS, A. SQLrand: preventing sql injection attacks. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, [S.l.], 2004.

BOZIC J., W. F. XSS pattern for attack modeling in testing. **2013 8th International Workshop on Automation of Software Test, AST 2013 - Proceeding**, [S.l.], 2013.

CESSO: 10/12/13(2013)BUG 680771 - SEND X-XSS-PROTECTION HEADER FOR XSS PREVENTION/BLOCKING. [URL: HTTPS://BUGZILLA.MOZ]Bugzilla BUG 680771 - Send X-XSS-Protection header for XSS prevention/blocking. [URL: https://bugzilla.mozilla.org/show_bug.cgi?id=680771] acesso: 10/12/13. 2013.

CABALLERO, S. C. H. C. M. Residue objects: a challenge to web browser security. **EuroSys '10 Proceedings of the 5th European conference on Computer systems**, [S.l.], 2010.

CESSO: 10/12/13(2013)CATEGORY:OWASP WEBGOAT PROJECT. [URL: HTTPS://WWW.OWASP.ORG/INDEX.PHP/CATEGORY:OWASP_WEBGOAT_PROJECT] ACESSO: 10/12/13]OWASPWebGoat CATEGORY:OWASP webgoat project. [url: https://www.owasp.org/index.php/category:owasp_webgoat_project] acesso: 10/12/13. 2013.

CESSO: 10/12/13(2012)COS MALICIOSOS (MALWARE). [URL: HTTP://CARTILHA.CERT.BR/MALWARE/] ACESSO: 10/12/13]CERT.BR.Malwares COS maliciosos (Malware). [URL: http://cartilha.cert.br/malware/] acesso: 10/12/13. 2012.

CHANDRASHEKHAR, R. et al. SQL injection attack mechanisms and prevention techniques. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, [S.l.], 2012.

CHEN, F.-H. H. C.-K. T. Y.-C. Y. W.-J. W. L.-H. BrowserGuard: a behavior-based solution to drive-by-download attacks. **Selected Areas in Communications, IEEE Journal on**, [S.l.], 2011.

CHEN, P. Z. J. S. H. Insecure JavaScript Detection and Analysis with Browser-Enforced Embedded Rules. **Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2010 International Conference on**, [S.l.], 2010.

CHHABRA S., A. A. B. F. K.-P. Phi.sh$oCiaL: the phishing landscape through short urls. **ACM International Conference Proceeding Series**, [S.l.], 2011.

CHO, S.; CHA, S. SAD: web session anomaly detection based on parameter estimation. **Computers Security**, [S.l.], 2004.

CHOMSIRI, T. HTTPS Hacking Protection. **Advanced Information Networking and Applications Workshops, 2007, AINAW 07. 21st International Conference on**, [S.l.], 2007.

CHOUDHARY S.; DINCTURK, M. B. G. J.-G.-V. O.-I. I. P. Solving Some Modeling Challenges when Testing Rich Internet Applications for Security. **Software Testing, Verification and Validation (ICST), 2012 IEEE Fifth International Conference on**, [S.l.], 2012.

CESSO: 10/12/13(2013)CHROME.DECLARATIVEWEBREQUEST. [URL: HTTP://DEVELOPER.CHROME.COM/EXTENSIONS/DECLARATIVEWEBREQUEST.HTML] ACESSO: 10/12/13]DeclarativeWebRequest. CHROME.DECLARATIVEWEBREQUEST. [URL: http://developer.chrome.com/extensions/declarativewebrequest.html] acesso: 10/12/13. 2013.

CESSO: 10/12/13(2013)CHROME.WEBREQUEST. [URL: HTTP://DEVELOPER.CHROME.COM/EXTENSIONS/WEBREQUEST.HTMLIMPLEMENTATION] ACESSO: 10/12/13]WebRequest CHROME.WEBREQUEST. [URL: http://developer.chrome.com/extensions /webrequest.htmlimplementation] acesso: 10/12/13. 2013.

CIAMPA, A.; VISAGGIO, C.; DI PENTA, M. A heuristic-based approach for detecting SQL-injection vulnerabilities in web applications. **Proceedings - International Conference on Software Engineering**, [S.l.], 2010.

CLAESSENS, J. et al. On the Security of Today's Online Eletronic Banking Systems. **Computers Security**, [S.l.], 2002.

COLLIN JACKSON ANDREW BORTZ, D. B.; MITCHELL, J. C. Protecting Browser State from Web Privacy Attacks. **15th International World Wide Web Conference**, [S.l.], 2006.

CRISPO, B.; LANDROCK, P.; JR., V. M. WWW security and trusted third party services. **Future Generation Comp. Syst.**, [S.l.], v.16, 2000.

CRITES S., H. F. C. H. OMash: enabling secure web mashups via object abstractions. **Proceedings of the ACM Conference on Computer and Communications Security**, [S.l.], 2008.

DE RYCK P., D. L. H. T.-P.-F. J.-W. CsFire: transparent client-side mitigation of malicious cross-domain requests. **2nd International Symposium on Engineering Secure Software and Systems**, [S.l.], 2010.

DE RYCK P., D. L. J. W.-P.-F. Automatic and precise client-side protection against CSRF attacks. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, [S.l.], 2011.

DE ZOYSA K.; MUFTIC, S. Bi-directional Web document protection system for serious e-commerce applications. **Parallel Processing Workshops, 2002. Proceedings. International Conference on**, [S.l.], 2002.

DEVESA J., C. X. A. G. B.-P. An efficient security solution for dealing with shortened URL analysis. **Proceedings of the 8th International Workshop on Security in Information Systems,**, [S.l.], 2011.

DHAWAN M.; GANAPATHY, V. Analyzing Information Flow in JavaScript-Based Browser Extensions. **Computer Security Applications Conference, 2009. ACSAC 09. Annual**, [S.l.], 2009.

DI LUCCA G.A.; FASOLINO, A. M. M. T.-P. Identifying cross site scripting vulnerabilities in Web applications. **Telecommunications Energy Conference, 2004. INTELEC 2004. 26th Annual International**, [S.l.], 2004.

DYMOND P.; JENKIN, M. WWW distribution of private information with watermarking. **Systems Sciences, 1999. HICSS-32. Proceedings of the 32nd Annual Hawaii International Conference on**, [S.l.], 1999.

EISEN, O. Catching the fraudulent Man-in-the-Middle and Man-in-the-Browser. **Network Security**, [S.l.], 2010.

ELIA, I.; FONSECA, J.; VIEIRA, M. Comparing SQL injection detection tools using attack injection: an experimental study. **Proceedings - International Symposium on Software Reliability Engineering, ISSRE**, [S.l.], 2010.

CESSO: 10/12/13(2013)ESAPI JAVASCRIPT. [URL: HTTPS://WWW.OWASP.ORG/INDEX.PHP/ESAPI_JAVASCRIPT_README] ACESSO: 10/12/13]OWASPESAPIJS ESAPI JavaScript. [URL: https://www.owasp.org/index.php/esapi_javascript _readme] acesso: 10/12/13. 2013.

FORD S.; COVA, M. K. C. V. G. Analyzing and Detecting Malicious Flash Advertisements. **Computer Security Applications Conference, 2009. ACSAC 09. Annual**, [S.l.], 2009.

FREDRIKSON M.; LIVSHITS, B. RePriv: re-imagining content personalization and in-browser privacy. **Security and Privacy (SP), 2011 IEEE Symposium on**, [S.l.], 2011.

FRENZ C.M.; YOON, J. XSSmon: a perl based ids for the detection of potential xss attacks. **Systems, Applications and Technology Conference (LISAT), 2012 IEEE Long Island**, [S.l.], 2012.

FUKUSHIMA Y.; HORI, Y. S. K. Proactive Blacklisting for Malicious Web Sites by Reputation Evaluation Based on Domain and IP Address Registration. **Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on**, [S.l.], 2011.

FUNG A.P.H.; CHEUNG, K. HTTPSLock: enforcing https in unmodified browsers with cached javascript. **Network and System Security (NSS), 2010 4th International Conference on**, [S.l.], 2010.

FUNG B.S.Y.; LEE, P. A Privacy-Preserving Defense Mechanism against Request Forgery Attacks. **Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on**, [S.l.], 2011.

GALAN E.; ALCAIDE, A. O. A. B. J. A multi-agent scanner to detect stored-XSS vulnerabilities. **Internet Technology and Secured Transactions (ICITST), 2010 International Conference for**, [S.l.], 2010.

GAMMA, E. et al. **Design Patterns**: elements of reusable object-oriented software. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1995.

GASTELLIER-PREVOST S.; GRANADILLO, G. L. M. A Dual Approach to Detect Pharming Attacks at the Client-Side. **New Technologies, Mobility and Security (NTMS), 2011 4th IFIP International Conference on**, [S.l.], 2011.

GASTELLIER-PREVOST S.; GRANADILLO, G. L. M. Decisive Heuristics to Differentiate Legitimate from Phishing Sites. **Network and Information Systems Security (SAR-SSI), 2011 Conference on**, [S.l.], 2011.

GASTELLIER-PREVOST S.; LAURENT, M. Defeating pharming attacks at the client-side. **Network and System Security (NSS), 2011 5th International Conference on**, [S.l.], 2011.

GAURAV AGGRAWAL ELIE BURSZTEIN, C. J.; BONEH, D. An analysis of private browsing modes in modern browsers. **In Proc. of Usenix Security**, [S.l.], 2010.

GEBRE, M. K.-S. L. M. H. A robust defense against Content-Sniffing XSS attacks. **Digital Content, Multimedia Technology and its Applications (IDC), 2010 6th International Conference on**, [S.l.], 2010.

GEN LU; DEBRAY, S. Automatic Simplification of Obfuscated JavaScript Code: a semantics-based approach. **Software Security and Reliability (SERE), 2012 IEEE Sixth International Conference on**, [S.l.], 2012.

GIFFORD D.K.; STEWART, L. P. A. T. G. Payment switches for open networks. **Compcon 95.Technologies for the Information Superhighway, Digest of Papers.**, [S.l.], 1995.

CESSO: 10/12/13(2012)GOLPES NA INTERNET. [URL: HTTP://CARTILHA.CERT.BR/GOLPES/] ACESSO: 10/12/13]CERT.BR.Golpes GOLPES na Internet. [URL: http://cartilha.cert.br/golpes/] acesso: 10/12/13. 2012.

GOTH, G. IE security flaws spike interest in alternative browsers. **Internet Computing, IEEE**, [S.l.], 2004.

GREG AARON, R. R. **Global Phishing Survey**: trends and domain name use in 2h2012. [S.l.]: APWG, 2012.

GRIER C.; SHUO TANG; KING, S. Secure Web Browsing with the OP Web Browser. **Security and Privacy, 2008. SP 2008. IEEE Symposium on**, [S.l.], 2008.

GROSSMAN, H. . Clickjacking. **WWW**, [S.l.], 2008.

GUHA A.; FREDRIKSON, M. L. B. S. N. Verified Security for Browser Extensions. **Security and Privacy (SP), 2011 IEEE Symposium on**, [S.l.], 2011.

GUNDY, M. V.; CHEN, H. Noncespaces: using randomization to defeat cross-site scripting attacks. **Computers Security**, [S.l.], 2012.

GUO, Y. W. Z. L. T. Program Slicing Stored XSS Bugs in Web Application. **Theoretical Aspects of Software Engineering (TASE), 2011 Fifth International Symposium on**, [S.l.], 2011.

GUSTAV RYDSTEDT ELIE BURSZTEIN, D. B.; JACKSON, C. Busting frame busting: a study of clickjacking vulnerabilities at popular sites. **IEEE Oakland Web 2.0 Security and Privacy Workshop**, [S.l.], 2010.

CESSO: 10/12/13(2009)HACKER INVADE MAC EM 10 SEGUNDOS E GANHA US$ 5 MIL. [URL: HTTP://MA]MacWorldBR HACKER invade Mac em 10 segundos e ganha US$ 5 mil. [URL: http://macworldbrasil.com.br/noticias/2009/03/19/hacker-invade-mac-em-10-segundos-e-ganha-us-5-mil/] acesso: 10/12/13.
2009.

HALFOND, W.; ORSO, A. AMNESIA: analysis and monitoring for neutralizing sql-injection attacks. **20th IEEE/ACM International Conference on Automated Software Engineering, ASE 2005**, [S.l.], 2005.

HALFOND, W.; ORSO, A. Detection and Prevention of SQL Injection Attacks. **Advances in Information Security**, [S.l.], 2007.

HALLARAKER O.; VIGNA, G. Detecting malicious JavaScript code in Mozilla. **Engineering of Complex Computer Systems, 2005. ICECCS 2005. Proceedings. 10th IEEE International Conference on**, [S.l.], 2005.

HASSLER V.; THEN, O. Controlling applets behavior in a browser. **Computer Security Applications Conference, 1998. Proceedings. 14th Annual**, [S.l.], 1998.

HECHT, J. When web browsers turn bad. **New Scientist**, [S.l.], 2007.

HEIDERICH M., F. T. H. T. IceShield: detection and mitigation of malicious websites with a frozen dom. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial**, [S.l.], 2011.

HERNANDEZ-ARDIETA, J. L. et al. A taxonomy and survey of attacks on digital signatures. **Computers Security**, [S.l.], 2013.

HERTZUM, M.; JRGENSEN, N.; NRGAARD, M. Usable Security and E-Banking: ease of use vis-a-vis security. **Australasian J. of Inf. Systems**, [S.l.], v.11, n.2, 2004.

HODOVAN R.; KISS, A. Security Evolution of the Webkit browser engine. **Web Systems Evolution (WSE), 2012 14th IEEE International Symposium on**, [S.l.], 2012.

HOFF T.P.; CREVATIN, M. von. HTTP digest authentication in embedded automation systems. **Emerging Technologies and Factory Automation, 2003. Proceedings. ETFA 03. IEEE Conference**, [S.l.], 2003.

CESSO: 10/12/13(2013)HTML5 ATTACK. [URL: HTTPS://APPSEC-LABS.COM/HTML5/HTML5-ATTACK-FRAMEWORK] ACESSO: 10/12/13]HTML5Framework HTML5 Attack. [URL: https://appsec-labs.com/html5/html5-attack-framework] acesso: 10/12/13. 2013.

CESSO: 10/12/13(2013)HTML5 SECURITY CHEAT SHEET. [URL: HTTPS://WWW.OWASP.ORG/INDEX.PHP/HTML5_SECURITY_CHEAT_SHEET] ACESSO: 10/12/13]OWASPHTML5 HTML5 Security Cheat Sheet. [URL: https://www.owasp.org/index.php/html5_security_cheat_sheet] acesso: 10/12/13. 2013.

CESSO: 10/12/13(2013)HTTP REQUESTS. [URL: HTTPS://DEVELOPER.MOZILLA.ORG/EN/DOCS/OBSERVER_NOTIFICATIONSHTTP _REQUESTS] ACESSO: 10]HTTPRequests HTTP requests. [URL: https://developer.mozilla.org/en/docs/observer _notificationshttp_requests] acesso: 10/12/13. 2013.

HUANG S.-K., L. H.-L. L. W.-M. L.-H. CRAXweb: automatic web application testing and attack generation. **Proceedings - 7th International Conference on Software Security and Reliability, SERE 2013**, [S.l.], 2013.

IACONO L.L.; RAJASEKARAN, H. Secure Browser-Based Access to Web Services. **Communications, 2009. ICC 09. IEEE International Conference on**, [S.l.], 2009.

IHA G.; DOI, H. An Implementation of the Binding Mechanism in the Web Browser for Preventing XSS Attacks: introducing the bind-value headers. **Availability, Reliability and Security, 2009. ARES 09. International Conference on**, [S.l.], 2009.

JACKSON, C.; BARTH, A. Beware of Finer-Grained Origins. **Web 2.0 Security and Privacy**, [S.l.], 2008.

JACKSON, C.; BARTH, A. ForceHTTPS Cookies: a defense against eavesdropping and pharming. **17th International World Wide Web Conference**, [S.l.], 2008.

JAGNERE, P. Vulnerabilities in social networking sites. **Parallel Distributed and Grid Computing (PDGC), 2012 2nd IEEE International Conference on**, [S.l.], 2012.

JAKOBSSON M.; STAMM, S. Web Camouflage: protecting your clients from browser-sniffing attacks. **Security Privacy, IEEE**, [S.l.], 2007.

JARKOWSKI, M.; BINCZEWSKI, A.; STROINSKI, M. New approach for management services with a Web browser. **Computer Networks**, [S.l.], 1999.

JAYARAMAN K., L. G. T.-P. C. S. Enforcing request integrity in web applications. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial**, [S.l.], 2010.

JAYARAMAN K.; WENLIANG DU; RAJAGOPALAN, B. C. S. ESCUDO: a fine-grained protection model for web browsers. **Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on**, [S.l.], 2010.

JENKIN M.; DYMOND, P. A plugin-based Privacy scheme for World-Wide Web file distribution. **System Sciences, 1998., Proceedings of the Thirty-First Hawaii International Conference on**, [S.l.], 1998.

JIANG WANG; HUANG Y.; GHOSH, A. SafeFox: a safe lightweight virtual browsing environment. **System Sciences (HICSS), 2010 43rd Hawaii International Conference on**, [S.l.], 2010.

JOHNS, M. PreparedJS: secure script-templates for javascript. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, [S.l.], 2013.

JON HOWELL COLLIN JACKSON, H. J. W.; FAN, X. MashupOS: operating system abstractions for client mashups. **11th Workshop on Hot Topics in Operating Systems**, [S.l.], 2007.

JOSHI Y.; DAS, D. S. S. Mitigating man in the middle attack over secure sockets layer. **Internet Multimedia Services Architecture and Applications (IMSAA), 2009 IEEE International Conference on**, [S.l.], 2009.

JOSHI Y.; SAKLIKAR, S. D. D. S. S. PhishGuard: a browser plug-in for protection from phishing. **Internet Multimedia Services Architecture and Applications, 2008. IMSAA 2008. 2nd International Conference on**, [S.l.], 2008.

JOVANOVIC N. ; KIRDA, E. . K. C. Preventing Cross Site Request Forgery Attacks. **Securecomm and Workshops, 2006**, [S.l.], 2006.

JR., R. C. D.; CARVER, C.; FERGUSON, A. J. Phishing for user security awareness. **Computers Security**, [S.l.], 2007.

JUELS A.; JAKOBSSON, M. J. T. Cache cookies for browser authentication. **Security and Privacy, 2006 IEEE Symposium on**, [S.l.], 2006.

JUN LI; DONGTING YU; MAURER, L. A resource management approach to web browser security. **Computing, Networking and Communications (ICNC), 2012 International Conference on**, [S.l.], 2012.

JUNG, J. K. K. S. S. WebVaccine: a client-side realtime prevention system against obfuscated malicious web pages. **INC, IMS and IDC, 2009. NCM 09. Fifth International Joint Conference on**, [S.l.], 2009.

KAI ZHAO; KAJI, Y. Anti-Phishing mutual authentication using the visual secret sharing scheme. **Information Theory and its Applications (ISITA), 2010 International Symposium on**, [S.l.], 2010.

KANESHIGE, T. **Seguranmeaeputa do Safari. [URL**: http://macworldbrasil.uol.com.br/noticias/999/12/31/seguranca-ameaca-reputacao-do-safari/] acesso: 10/12/13. 2009.

KARLOF C., S. U. T. J. W. D. Dynamic pharming attacks and locked Same-Origin policies for web browsers. **Proceedings of the ACM Conference on Computer and Communications Security**, [S.l.], 2007.

KERSCHBAUM, F. Simple cross-site attack prevention. **Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on**, [S.l.], 2008.

KHOURY, N. et al. An analysis of black-box web application security scanners against stored SQL injection. **Proceedings - 2011 IEEE International Conference on Privacy, Security, Risk and Trust and IEEE International Conference on Social Computing, PASSAT/SocialCom 2011**, [S.l.], 2011.

KIM DOOWON; ACHAN, C. B. J. F. P. S. Implementation of framework to identify potential Phishing websites. **Intelligence and Security Informatics (ISI), 2013 IEEE International Conference on**, [S.l.], 2013.

KIRDA, E. et al. Client-side cross-site scripting protection. **Computers Security**, [S.l.], 2009.

KIRDA E., K. C. V. G. J. N. Noxes: a client-side solution for mitigating cross-site scripting attacks. **Proceedings of the ACM Symposium on Applied Computing**, [S.l.], 2006.

KIRDA E.; KRUEGEL, C. Protecting users against Phishing attacks with AntiPhish. **Computer Software and Applications Conference, 2005. COMPSAC 2005. 29th Annual International**, [S.l.], 2005.

KIRDA, K. O. M. B. B. W. R. E. Securing legacy firefox extensions with SENTINEL. **DIMVA'13 Proceedings of the 10th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment**, [S.l.], 2013.

KOIZUMI D., M. T. S. M. On the automatic detection algorithm of Cross Site Scripting (XSS) with the non-stationary Bernoulli distribution. **Proceedings of the 5th International Conference on Communications, Computers and**, [S.l.], 2012.

KOLBITSCH C.; LIVSHITS, B. Z. B. S. C. Rozzle: de-cloaking internet malware. **Security and Privacy (SP), 2012 IEEE Symposium on**, [S.l.], 2012.

KOLLETZKI, S. Secure internet banking with Privacy enhanced mail: a protocol for reliable exchange of secured order forms. **Computer Networks and Systems**, [S.l.], 1996.

KRUEGEL, C.; VIGNA, G.; ROBERTSON, W. A multi-model approach to the detection of web-based attacks. **Computer Networks**, [S.l.], 2005.

LARANJEIRO, N.; VIEIRA, M.; MADEIRA, H. A learning-based approach to secure web services from SQL/XPath Injection attacks. **Proceedings - 16th IEEE Pacific Rim International Symposium on Dependable Computing, PRDC 2010**, [S.l.], 2010.

LEE, J. H.; LIM, W. G.; LIM, J. I. A study of the security of Internet banking and financial private information in South Korea. **Mathematical and Computer Modelling**, [S.l.], 2013.

LEKIES S., T. W. J. M. Towards stateless, client-side driven Cross-site request forgery protection for Web applications. **Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI)**, [S.l.], 2012.

LIN, J.-K. J. Y.-Y. C. Y. The design of protocol for e-voting on the Internet. **Security Technology, 2001 IEEE 35th International Carnahan Conference on**, [S.l.], 2001.

LIU, A. X.; KOVACS, J. M.; GOUDA, M. G. A secure cookie scheme. **Computer Networks**, [S.l.], 2012.

LUNDEEN B.; ALVES-FOSS, J. Practical clickjacking with BeEF. **Homeland Security (HST), 2012 IEEE Conference on Technologies for**, [S.l.], 2012.

LWIN, N. Agent Based Web Browser. **Autonomic and Autonomous Systems, 2009. ICAS 09. Fifth International Conference on**, [S.l.], 2009.

MAFFEIS, K. B. A. D.-L. S. Language-based defenses against untrusted browser origins. **SEC'13 Proceedings of the 22nd USENIX conference on Security**, [S.l.], 2013.

MAFFEIS S.; TALY, A. Language-Based Isolation of Untrusted JavaScript. **Computer Security Foundations Symposium, 2009. CSF 09. 22nd IEEE**, [S.l.], 2009.

MALANDRINO D.; SCARANO, V. Supportive, Comprehensive and Improved Privacy Protection for Web Browsing. **Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom)**, [S.l.], 2011.

MALKHI D. ; REITER, M. . R. A. Secure execution of Java applets using a remote playground. **Security and Privacy, 1998. Proceedings. 1998 IEEE Symposium**, [S.l.], 1998.

CESSO: 10/12/13(2013)MALWARE HIDDEN INSIDE JPG EXIF HEADERS. [URL: HTTP://BLOG.SUCURI.NET/2013/07/MALWARE-HIDDEN-INSIDE-JPG-EXI]Sucuri MALWARE Hidden Inside JPG EXIF Headers. [URL: http://blog.sucuri.net/2013/07/malware-hidden-inside-jpg-exif-headers.html] acesso: 10/12/13. 2013.

MATEUS FELIPE TYMBURIBERREIRA THIAGO DE SOUZA ROCHA, G. B. M. E. F. e. E. S. Anse de vulnerabilidades em Sistemas Computacionais Modernos: conceitos, exploits e protes. In: MINICURSOS DO XII SIMP BRASILEIRO EM SEGURANA INFORMA E DE SISTEMAS COMPUTACIONAIS. **Anais...** [S.l.: s.n.], 2012.

MATSUDA T., K. D. S. M. Cross site scripting attacks detection algorithm based on the appearance position of characters. **Proceedings of the 5th International Conference on Communications, Computers and**, [S.l.], 2012.

MAURER M.-E., H. L. Sophisticated phishers make more spelling mistakes: using url similarity against. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, [S.l.], 2011.

MAURER M.-E., H. L. Sophisticated phishers make more spelling mistakes: using url similarity against. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial**, [S.l.], 2012.

MEYER, H. Browser bugs raise security concerns. **Computers Security**, [S.l.], 1997.

MEYER, H. Securing your users Web browsers. **Computers Security**, [S.l.], 1997.

MICHA? CHORA?, R. K. Evaluation of various techniques for SQL injection attack detection. **Advances in Intelligent Systems and Computing**, [S.l.], 2013.

CESSO: 10/12/13(2009)MICROSOFT LANE8 DESTACANDO MELHORIAS E]IDGNow MICROSOFT lanE8 destacando melhorias em seguran performance. [URL: http://idgnow.com.br/internet/2009/03/19/microsoft-lanca-ie8-destacando-melhorias-em-seguranca-e-performance/] acesso: 10/12/13.
2009.

NAKAYAMA S.; ECHIZEN, I. Y. H. Preventing False Positives in Content-Based Phishing Detection. **Intelligent Information Hiding and Multimedia Signal Processing, 2009. IIH-MSP 09. Fifth International Conference on**, [S.l.], 2009.

CESSO: 10/12/13(2009)NAVEGADORES POPULARES NESISTEM]G1 NAVEGADORES populares nesistem a ataques em competi. [URL: http://g1.globo.com/noticias/tecnologia/0,,mul1050769-6174,00-navegadores+populares+nao+resistem+a+ataques+em+competicao.html] acesso: 10/12/13. 2009.

NIKIFORAKIS N.; KAPRAVELOS, A. J. W. K. C. P. F. V. G. Cookieless Monster: exploring the ecosystem of web-based device fingerprinting. **Security and Privacy (SP), 2013 IEEE Symposium on**, [S.l.], 2013.

NTAGWABIRA, L.; KANG, S. Use of query tokenization to detect and prevent SQL injection attacks. **Proceedings - 2010 3rd IEEE International Conference on Computer Science and Information Technology, ICCSIT 2010**, [S.l.], 2010.

NUNAN A.E.; SOUTO, E. d. S. E. F. E. Automatic classification of cross-site scripting in web pages using document-based and URL-based features. **Computers and Communications (ISCC), 2012 IEEE Symposium on**, [S.l.], 2012.

CESSO: 10/12/13(2013)O QUE SPLICATIVOS, EXTENSES E TEMAS? [URL: HTTP://WWW.GOOGLE.COM/INTL/PT-BR_ALL/LAND]OqueSaoAplicativos O que splicativos, extenses e temas? [URL: http://www.google.com/intl/pt-br_all/landing/chrome/webstore/create/whats-in-store.html] acesso: 10/12/13. 2013.

OFUONYE E.; MILLER, J. Resolving JavaScript Vulnerabilities in the Browser Runtime. **Software Reliability Engineering, 2008. ISSRE 2008. 19th International Symposium on**, [S.l.], 2008.

OFUONYE, E.; MILLER, J. Securing web-clients with instrumented code and dynamic runtime monitoring. **Journal of Systems and Software**, [S.l.], 2013.

OJAMAA A.; DUUNA, K. Assessing the security of Node.js platform. **Internet Technology And Secured Transactions, 2012 International Conferece For**, [S.l.], 2012.

OLIVEIRA, C. K. O. A. O. S. L. S. Obtaining the threat model for e-mail phishing. **Journal Applied Soft Computing**, [S.l.], 2013.

O'REILLY, T. **What is Web 2.0**: design patterns and business models for the next generation of software. [S.l.]: O'Reilly, 2005.

ORIYANO, S.-P.; SHIMONSKI, R. Protecting Web Browsers. **Client-Side Attacks and Defense**, [S.l.], 2012.

ORMAN, H. Towards a Semantics of Phish. **Security and Privacy Workshops (SPW), 2012 IEEE Symposium on**, [S.l.], 2012.

OWASP. **OWASP Top Ten Project 2013**. [S.l.]: OWASP, 2013.

OWASP. **Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet. [URL**: https://www.owasp.org/index.php/cross-site_request_forgery_(csrf)_prevention_cheat_sheet] acesso: 10/12/13. 2013.

CESSO: 10/12/13(2013)OWASP BROKEN WEB APPLICATIONS PROJECT. [URL: HTTPS://WWW.OWASP.ORG/INDEX.PHP/OWASP_BROKEN_WEB_APPLICATION]OWASPBroken OWASP Broken Web Applications Project. [URL: https://www.owasp.org/index.php/owasp_broken_web_applications_project] acesso: 10/12/13. 2013.

PARK J.S.; SANDHU, R. Secure cookies on the Web. **Internet Computing, IEEE**, [S.l.], 2002.

PELIZZI R., S. R. A server- and browser-transparent CSRF defense for web 2.0 applications. **ACM International Conference Proceeding Serie**, [S.l.], 2011.

PINZ. et al. AIIDA-SQL: an adaptive intelligent intrusion detector agent for detecting sql injection attacks. **2010 10th International Conference on Hybrid Intelligent Systems, HIS 2010**, [S.l.], 2010.

PRINSLOO, J.; ELOFF, J. Web browsers: information security issues. **Computer Fraud Security**, [S.l.], 1999.

PUANGPRONPITAG S.; SRIWIBOON, N. Simple and Lightweight HTTPS Enforcement to Protect against SSL Striping Attack. **Computational Intelligence, Communication Systems and Networks (CICSyN), 2012 Fourth International Conference on**, [S.l.], 2012.

QINGXIAN, Z. C. W. Systematical Vulnerability Detection in Browser Validation Mechanism. **Computational Intelligence and Security (CIS), 2011 Seventh International Conference on**, [S.l.], 2011.

QIU, B. N. F. L. W. Detect Visual Spoofing in Unicode-Based Text. **Pattern Recognition (ICPR), 2010 20th International Conference on**, [S.l.], 2010.

QURASHI U.S.; ANWAR, Z. AJAX based attacks: exploiting web 2.0. **Emerging Technologies (ICET), 2012 International Conference on**, [S.l.], 2012.

RAICHAL, S. et al. Detection and avoidance of SQL injection attack in multi-tier web based application. **International Journal of Engineering and Technology**, [S.l.], 2013.

RITCHIE, P. The security risks of AJAX/web 2.0 applications. **Network Security**, [S.l.], 2007.

ROUAIX, F. A Web navigator with applets in Caml. **Computer Networks and Systems**, [S.l.], 1996.

SAIEDIAN H.; BROYLE, D. Security Vulnerabilities in the Same-Origin Policy: implications and alternatives. **Computer**, [S.l.], 2011.

SCHULTZ, E. E. Internet Explorer security: is there any hope? **Network Security**, [S.l.], 2005.

CESSO: 10/12/13(2008)SECURING YOUR WEB BROWSER. [URL: HTTP://WWW.CERT.ORG/TECH_TIPS/SECURING_BROWSER/] ACESSO: 10/12/13]CERT.Browser SECURING Your Web Browser. [URL: http://www.cert.org/tech_tips/securing_browser/] acesso: 10/12/13. 2008.

SECURITY, N. Web security for browsers. **Network Security**, [S.l.], 1998.

SECURITY, N. Browser attacks: can wipe out almost anything? **Network Security**, [S.l.], 2001.

SECURITY, W. **WEBSITE SECURITY STATISTICS REPORT**. [S.l.]: WhiteHat Security, 2013.

SEIFERT C.; WELCH, I. K. P. Identification of Malicious Web Pages with Static Heuristics. **Telecommunication Networks and Applications Conference, 2008. ATNAC 2008. Australasian**, [S.l.], 2008.

SERRHINI M.; DARGHAM, A. A.-M. A. Improve security of web Browser with stand-alone e-Learning awareness application. **Multimedia Computing and Systems (ICMCS), 2012 International Conference on**, [S.l.], 2012.

SHAHRIAR H., W. K. L.-T. Z. M. A model-based detection of vulnerable and malicious browser extensions. **Proceedings - 7th International Conference on Software Security and Reliability, SERE 2013**, [S.l.], 2013.

SHAHRIAR H., Z. M. Trustworthiness testing of phishing websites: a behavior model-based approach. **Future Generation Computer Systems**, [S.l.], 2012.

SHAHRIAR H.; ZULKERNINE, M. Client-Side Detection of Cross-Site Request Forgery Attacks. **Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on**, [S.l.], 2010.

SHAR, L. K. H. B. K. T. Defending against Cross-Site Scripting Attacks. **Computer**, [S.l.], 2012.

SHAR, L. K.; TAN, H. B. K. Automated removal of cross site scripting vulnerabilities in web applications. **Information and Software Technology**, [S.l.], 2012.

SHAR, L.; TAN, H. Predicting SQL injection and cross site scripting vulnerabilities through mining input sanitization patterns. **Information and Software Technology**, [S.l.], 2013.

SHAR L.K., T. H. Auditing the XSS defence features implemented in web application programs. **IET Software**, [S.l.], 2012.

SHAR L.K., T. H. Predicting SQL injection and cross site scripting vulnerabilities through mining input sanitization patterns. **Information and Software Technology**, [S.l.], 2013.

SHEPHERD, S. Lessons learned from security weaknesses in the Netscape World Wide Web browser. **Public Uses of Cryptography., IEE Colloquium on**, [S.l.], 1996.

SHI, C.-C. et al. A new approach for SQL-injection detection. **Advances in Intelligent and Soft Computing**, [S.l.], 2012.

SHUN-WEN HSIAO; SUN, Y. F.-C. A. M. C. C. A Secure Proxy-Based Cross-Domain Communication for Web Mashups. **Web Services (ECOWS), 2011 Ninth IEEE European Conference on**, [S.l.], 2011.

SIDDIQUI M.S.; VERMA, D. Cross site request forgery: a common web application weakness. **Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on**, [S.l.], 2011.

SILIC M.; KROLO, J. D. G. Security vulnerabilities in modern web browser architecture. **MIPRO, 2010 Proceedings of the 33rd International Convention**, [S.l.], 2010.

SKARUZ J.; SEREDYNSKI, F. Intrusion detection in web applications: evolutionary approach. **Computer Science and Information Technology, 2009. IMCSIT 09. International Multiconference on**, [S.l.], 2009.

SNELL, M. Secure shopping on the Internet. **Network Security**, [S.l.], 1995.

SONNTAG, M. AJAX Security in Groupware. **Software Engineering and Advanced Applications, 2006. SEAA 06. 32nd EUROMICRO Conference on**, [S.l.], 2006.

SOOD A., E. R. The state of HTTP declarative security in online banking websites. **Computer Fraud and Security**, [S.l.], 2011.

SOOD, A. K.; ENBODY, R. J. Spying on the browser: dissecting the design of malicious extensions. **Network Security**, [S.l.], 2011.

SOOD, A. K.; ENBODY, R. J. Malvertising: exploiting web advertising. **Computer Fraud Security**, [S.l.], 2011.

SOOD A.K., E. R. Frametrapping the framebusting defence. **Network Security**, [S.l.], 2011.

STEVANOVIC, D.; VLAJIC, N.; AN, A. Detection of malicious and non-malicious website visitors using unsupervised neural network learning. **Applied Soft Computing**, [S.l.], 2013.

STOCK, M. J. S. L. B. Eradicating DNS rebinding with the extended Same-Origin policy. **SEC'13 Proceedings of the 22nd USENIX conference on Security**, [S.l.], 2013.

SUBSORN P.; LIMWIRIYAKUL, S. A Case Study of Internet Banking Security of Mainland Chinese Banks: a customer perspective. **Computational Intelligence, Communication Systems and Networks (CICSyN), 2012 Fourth International Conference on**, [S.l.], 2012.

SUN J.-H., L. Z.-R.-C. H. Secure cross-domain communication mechanism for Web mashups. **Tongxin Xuebao/Journal on Communications**, [S.l.], 2012.

SUN Q.; SIMON, D. Y.-M. W. R. W. P. V. L. Q. Statistical identification of encrypted Web browsing traffic. **Security and Privacy, 2002. Proceedings. 2002 IEEE Symposium on**, [S.l.], 2004.

SUN Y., H. D. Model checking for the defense against cross-site scripting attacks. **Proceedings - 2012 International Conference on Computer Science and Service System,**, [S.l.], 2012.

SUNG Y.-C., C. M.-W. C.-W. H. C.-W. S. S. Light-weight CSRF protection by labeling user-created contents. **Proceedings - 7th International Conference on Software Security and Reliability, SERE 2013'**, [S.l.], 2013.

TAJPOUR, A. et al. SQL injection detection and prevention tools assessment. **Proceedings - 2010 3rd IEEE International Conference on Computer Science and Information Technology, ICCSIT 2010**, [S.l.], 2010.

TAKESUE, M. A Protection Scheme against the Attacks Deployed by Hiding the Violation of the Same Origin Policy. **Emerging Security Information, Systems and Technologies, 2008. SECURWARE 08. Second International Conference on**, [S.l.], 2008.

TAKESUE, M. An HTTP Extension for Secure Transfer of Confidential Data. **Networking, Architecture, and Storage, 2009. NAS 2009. IEEE International Conference on**, [S.l.], 2009.

TAN, H. H. S. Z. J. Browser-Side Countermeasures for Deceptive Phishing Attack. **Information Assurance and Security, 2009. IAS 09. Fifth International Conference on**, [S.l.], 2009.

TER LOUW M.; VENKATAKRISHNAN, V. Blueprint: robust prevention of cross-site scripting attacks for existing browsers. **Security and Privacy, 2009 30th IEEE Symposium on**, [S.l.], 2009.

CESSO: 10/12/13(2013)TESTING FOR COOKIES ATTRIBUTES (OWASP-SM-002). [URL: HTTPS://WWW.OWASP.ORG/INDEX.PHP/TESTING]OWASPTestCookies TESTING for cookies attributes (OWASP-SM-002). [URL: https://www.owasp.org/index.php/testing_for_cookies_attributes_(owasp-sm-002)] acesso: 10/12/13. 2013.

CESSO: 10/12/13(2013)THE WORLD WIDE WEB IS NOW TRULY. [URL: HTTP://MASHABLE.COM/2013/10/17/INTERNET-USERS-1996/?UTM_CID=MASH]Mashable THE World Wide Web is Now Truly. [URL: http://mashable.com/2013/10/17/internet-users-1996/?utm_cid=mash-com-g+-pete-photo] acesso: 10/12/13. 2013.

THOMAS K., G. C. M. J. P.-V. S. D. Design and evaluation of a real-time URL spam filtering service. **Proceedings - IEEE Symposium on Security and Privacy**, [S.l.], 2011.

TRAMPUS M.; CIGLARIC, M. P. M. V.-T. Are e-commerce users defenceless? **Intelligence and Lecture Notes in Bioinformatics)**, [S.l.], 2003.

TRENDMICRO. **The Basics of Web Threats**. [S.l.]: TRENDMicro, 2011.

URIEN, P. Internet card, a smart card as a true Internet node. **Computer Communications**, [S.l.], 2000.

CESSO: 10/12/13(2013)USING EVAL IN CHROME EXTENSIONS SAFELY. [URL: HTTP://DEVELOPER.CHROME.COM/APPS/SANDBOXINGEVAL.HTML] ACESSO: 10/12/13]ChromeSandbox USING eval in Chrome Extensions Safely. [URL: http://developer.chrome.com/apps/sandboxingeval.html] acesso: 10/12/13. 2013.

VAN SOLINGEN, R.; BERGHOUT, E. **The Goal/Question/Metric Method**: a practical guide for quality improvement of software development. [S.l.]: McGraw-Hill Higher Education, 1999.

MARCINIAK, J. J. (Ed.). **Goal Question Metric Paradigm**. [S.l.]: Encyclopedia of Software Engineering, 1994.

VYSHEGORODTSEV M.; MIYAMOTO, D. W. Y. Reputation Scoring System Using an Economic Trust Model: a distributed approach to evaluate trusted third parties on the internet. **Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on**, [S.l.], 2013.

WAGNER, N. C. A. P. F. D. An evaluation of the Google Chrome extension security architecture. **Security'12 Proceedings of the 21st USENIX conference on Security**, [S.l.], 2012.

WAHLBERG THOMAS; PAAKKOLA, P. W. C. L.-M. R. J. Kepler: raising browser security awareness. **Software Testing, Verification and Validation Workshops (ICSTW), 2013 IEEE Sixth International Conference on**, [S.l.], 2013.

WANG, A. C. A. M. T. K. H. J. Lightweight server support for browser-based CSRF protection. **WWW '13 Proceedings of the 22nd international conference on World Wide Web**, [S.l.], 2013.

WANG, C. Y. H. Anti-Phishing in Offense and Defense. **Computer Security Applications Conference, 2008. ACSAC 2008. Annual**, [S.l.], 2008.

WANG, J.; XIAO, N.; RAO, H. R. An exploration of risk information search via a search engine: queries and clicks in healthcare and information security. **Decision Support Systems**, [S.l.], 2012.

WANG PINGJIAN; WANG, L. X. J. L. P. G. N. J. J. MJBlocker: a lightweight and run-time malicious javascript extensions blocker. **Software Security and Reliability (SERE), 2013 IEEE 7th International Conference on**, [S.l.], 2013.

WANG, Z. L. K. Z. X. Mash-IF: practical information-flow control within client-side mashups. **Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on**, [S.l.], 2010.

WASSERMANN, G. Z. S. Static detection of cross-site scripting vulnerabilities. **Software Engineering, 2008. ICSE 08. ACM/IEEE 30th International Conference on**, [S.l.], 2008.

WATANABE T.; ZIXUE CHENG; KANSEN, M. H. M. A New Security Testing Method for Detecting Flash Vulnerabilities by Generating Test Patterns. **Network-Based Information Systems (NBiS), 2010 13th International Conference on**, [S.l.], 2010.

WATKINS, P. Cross-Site Request Forgeries. **WWW**, [S.l.], 2001.

WEINBERG Z.; CHEN, E. J. P. J. C. I Still Know What You Visited Last Summer: leaking browsing history via user interaction and side channel attacks. **Security and Privacy (SP), 2011 IEEE Symposium**, [S.l.], 2011.

WINDER, D. Does Web 2.0 need security 2.0? **Infosecurity**, [S.l.], 2010.

WOMBACHER A.; KOSTAKI, P. A. K. webXice: an infrastructure for information commerce on the www. **System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on**, [S.l.], 2001.

XING L., Z. Y. C. S. A client-based and server-enhanced defense mechanism for cross-site request forgery. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial**, [S.l.], 2010.

CESSO: 10/12/13(2013)XSS (CROSS SITE SCRIPTING) PREVENTION CHEAT SHEET. [URL: HTTPS://WWW.OWASP.ORG/IND]OWASPXSSPrevention XSS (Cross Site Scripting) Prevention Cheat Sheet. [URL: https://www.owasp.org/index.php/xss_(cross_site_scripting)_prevention_cheat_sheet] acesso: 10/12/13. 2013.

XU, H.; CROSSLER, R. E.; BéLANGER, F. A Value Sensitive Design Investigation of Privacy Enhancing Tools in Web Browsers. **Decision Support Systems**, [S.l.], 2012.

XU H., H. X. Z. D. A XSS defensive scheme based on behavior certification. **Applied Mechanics and Materials**, [S.l.], 2013.

YANG, Y. Y. Y. An efficient webpage information hiding method based on tag attributes. **Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on**, [S.l.], 2010.

YINZHI CAO ZHICHUN LI, V. R. Y. C. Virtual browser: a web-level sandbox to secure third-party javascript without sacrificing functionality. **Proceedings of the 17th ACM conference on Computer and communications security**, [S.l.], 2010.

YUAN, Y. C. Y. A comparative Study of the Safety between Internet Explorer and Firefox. **Information Science and Engineering (ISISE), 2012 International Symposium on**, [S.l.], 2012.

YUE, K. C. L. W. C. L. O. Hitchbot - Delivering Malicious URLs via Social Hitch-Hiking. **Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE**, [S.l.], 2011.

YUE WANG; AGRAWAL, R. B.-Y. C. Light Weight Anti-Phishing with User Whitelisting in a Web Browser. **Region 5 Conference, 2008 IEEE**, [S.l.], 2008.

ZARANDIOON S.; DANFENG YAO; GANAPATHY, V. OMOS: a framework for secure communication in mashup applications. **Computer Security Applications Conference, 2008. ACSAC 2008. Annual**, [S.l.], 2008.

ZAVOU A., A. E. P. G. K. A. Exploiting split browsers for efficiently protecting user data. **Proceedings of the ACM Conference on Computer and Communications Security**, [S.l.], 2012.

ZHANG, S. C. Z. M. Y.-M. W. M. Pretty-Bad-Proxy: an overlooked adversary in browsers https deployments. **Security and Privacy, 2009 30th IEEE Symposium on**, [S.l.], 2009.

ZHENGQIN LUO; REZK, T. Mashic Compiler: mashup sandboxing based on inter-frame communication. **Computer Security Foundations Symposium (CSF), 2012 IEEE 25th**, [S.l.], 2012.

A

Apêndice

Figura A.1 Exemplo de Phishing em atividade.

```
<html><head>
<script src="http://code.jquery.com/jquery-latest.js»</script>
<script src="https://dl.dropboxusercontent.com
/u/XXX/jquery.xdomainajax.js»
</script><script type="text/javascript»
$.ajax({url: 'http://servicoexterno.com.br/exemplo.html', type: 'GET',
success: function(res) {
var conteudo = $(res.responseText).text();
$("#resposta").html(conteudo);}});
</script></head><body><div id="resposta»</div></body></html>
```

Tabela A.1 Exemplo de requisição em domínio externo.

```
Bem Vindo! Consumo realizado, aproveite :)
&#x3C;&#x73;&#x63;&#x72;&#x69;&#x70;&#x74;
&#x3E;&#x61;&#x6C;&#x65;&#x72;&#x74;&#x28;
&#x27;&#x58;&#x53;&#x53;&#x27;&#x29;&#x3B;
&#x3C;&#x2F;&#x73;&#x63;&#x72;&#x69;&#x70;
&#x74;&#x3E;
```

Tabela A.2 Conteúdo retornado pelo serviço externo.

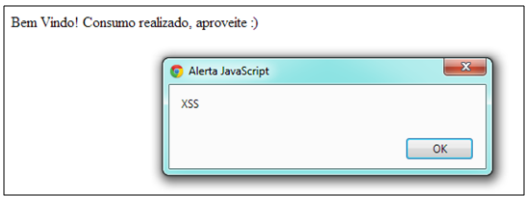

Figura A.2 Resposta renderizada pelo navegador para o usuário.

Aplicações	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**	**A9**
Acunetix Vulnweb	X	X	O	O	O	O	O	O	O
BadStore	X	X	O	O	O	O	O	O	O
Cenzic (CrackMe Bank)	X	X	O	O	O	O	O	O	O
Cyclone Transfers	X	X	O	O	O	X	O	O	O
Damn Vulnerable Web Application	X	X	O	O	O	O	O	O	O
EeSafe.org	X	X	O	O	O	O	O	O	O
EnigmaGroup	X	X	O	O	O	O	O	O	O
Exploit- DB	O	X	O	O	O	O	O	O	O
Exploit KB Vulnerable Web App	O	X	O	O	O	O	O	O	O
Google Gruyere	X	X	O	O	O	X	O	X	O
Hackxor	X	X	O	O	O	O	O	X	O
Hacme Apps	X	X	O	O	O	X	O	O	O
Moth	X	X	O	O	O	O	O	O	O
Mutillidae	X	X	O	O	O	X	O	O	O
OWASP Hackademic	X	X	O	O	O	O	O	O	O
OWASP SiteGenerator	X	X	O	O	O	O	O	O	O
OWASP Vicnum	X	X	O	O	O	X	O	O	O
OWASP Web Goat	X	X	O	O	O	X	O	X	X
PCTechtips Challenge	O	X	O	O	O	O	O	O	O
Peruggia	X	X	O	O	O	X	O	X	O
Stanford SecuriBench	X	X	O	O	O	X	O	O	O
The Bodgeit Store	X	X	O	O	O	X	O	X	O
The Butterfly Security Project	X	X	O	O	O	X	O	O	O
WackoPicko	X	X	O	O	O	X	O	O	O
Watchfire	X	X	O	O	O	O	O	O	O
Web Security Dojo	X	O	O	O	O	O	O	O	O
WebMaven/Buggy Bank	X	X	O	O	O	O	O	O	O
XSS Encoding Skills x5s	X	O	O	O	O	O	O	O	O

Tabela A.3 Ambientes Controlados e a cobertura aos Ataques

Extensões	Navegador	Versão
Aegis	Chrome / Firefox	ND
avast! Online Security	Chrome	9.0.2006.45
Anti-Phishing	Firefox	1.0
Clickjacking Reveal	Firefox	1.1
Clickjacking Test	Chrome	1.0
CSRF Finder	Firefox	1.2
ImmuniWeb Self-Fuzzer	Firefox	0.9.3
Netcraft Extension	Chrome / Firefox	1.3.0 / 1.8.3
No-Script Security Suite	Firefox	2.6.8.5
RequestPolicy	Firefox	0.6.28
RightClickXSS	Firefox	0.2.1
Safe Preview	Chrome / Firefox	2.0.0 / 1.1.2
ScriptSafe	Chrome	1.0.6.16
SQL Inject Me	Firefox	0.4.7
TamperMonkey	Chrome	3.7.3703
Toogle Cookies	Firefox	0.7.4
Web of Trust (WOT)	Chrome / Firefox	2.3.1 / 20131118
Websecurify	Chrome	4.0.0
XSS chef	Chrome	1.0
XSS Me	Firefox	0.4.6
Zscaler Likejacking Prevention	Chrome	1.1.9

Tabela A.4 Listagem das Extensões submetidas aos testes

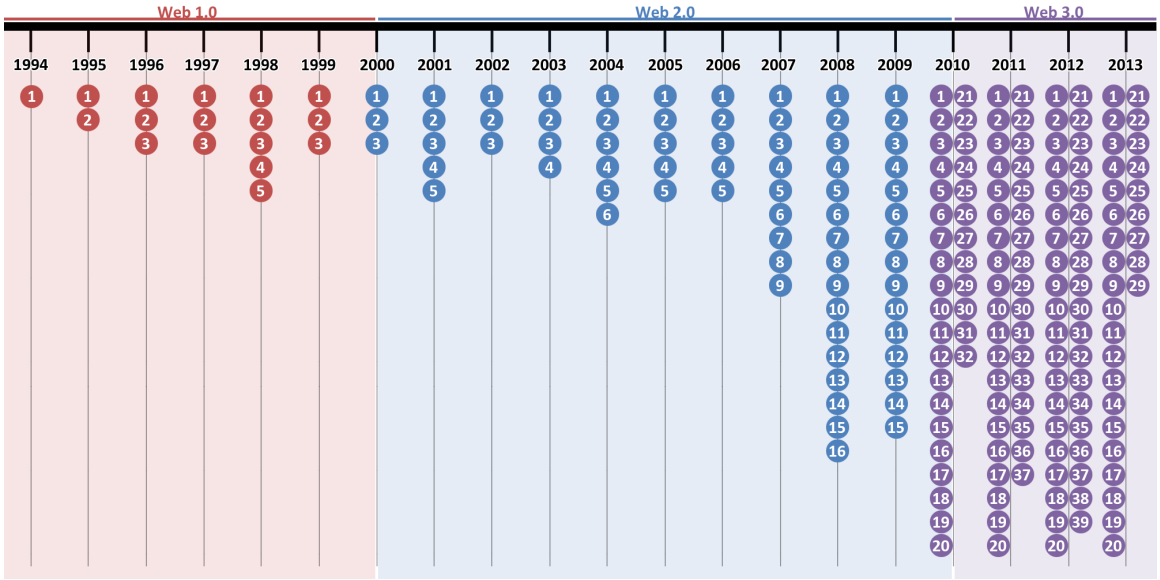

Figura A.3 Distribuição das principais publicações durante a evolução da web.

1994	1995	1996	1997	1998
1Bina (1994)	1Snell (1995), 2Gifford (1995)	1Shepherd (1996), 2Kolletzki (1996), 3Rouaix (1996)	1Meyer (1997a), 2Meyer (1997b), 3Balfanz. (1997)	1Anupam (1998), 2Jenkin (1998), 3Hassler (1998), 4Malkhi (1998), 5Security (1998)
1999	**2000**	**2001**	**2002**	**2003**
1Prinsloo and Eloff (1999), 2Jarkowski *et al.* (1999), 3Dymond (1999)	1Anderson and Lee (2000), 2Urien (2000), 3Crispo *et al.* (2000)	1Security (2001), 2Wombacher (2001), 3Watkins (2001), 4Lin (2001), 5Barone (2001)	1Park (2002), 2De Zoysa (2002), 3Claessens *et al.* (2002)	1Alvarez and Petrovic (2003), 2Trampus (2003), 3von Hoff (2003), 4Balzer (2003)

2004	2005	2006	2007
1Sun (2004), 2Di Lucca (2004), 3Goth (2004), 4Cho and Cha (2004), 5Alfantookh (2004), 6Boyd and Keromytis (2004)	1Halfond and Orso (2005a), 2Kruegel *et al.* (2005), 3Schultz (2005), 4Kirda (2005), 5Hallaraker (2005)	1Juels (2006), 2Kirda (2006), 3Sonntag (2006), 4Collin Jackson and Mitchell (2006), 5Jovanovic (2006)	1Jr. *et al.* (2007), 2Ritchie (2007), 3Hecht (2007), 4Bandhakavi *et al.* (2007), 5Halfond and Orso (2007), 6Jakobsson (2007), 7Karlof (2007), 8Jon Howell and Fan (2007), 9Chomsiri (2007)

Tabela A.5 Principais contribuições relacionadas ao tema nas fontes acadêmicas

2008	2009	2010
1Wang (2008), 2Takesue (2008), 3Crites (2008), 4Zarandioon (2008), 5Joshi (2008), 6Adam Barth and Mitchell (2008a), 7Seifert (2008), 8Jackson and Barth (2008a), 9Jackson and Barth (2008b), 10Yue Wang; Agrawal (2008), 11Adam Barth and Mitchell (2008b), 12Grossman (2008), 13Ofuonye (2008), 14Grier (2008), 15Wassermann (2008), 16Kerschbaum (2008)	1Iacono (2009), 2Skaruz (2009), 3Zhang (2009), 4Nakayama (2009), 5Maffeis (2009), 6Joshi (2009), 7Lwin (2009), 8Takesue (2009), 9Iha (2009), 10Ford (2009), 11Dhawan (2009), 12Ter Louw (2009), 13Tan (2009), 14Jung (2009), 15Kirda *et al.* (2009)	1Eisen (2010), 2Alexenko (2010), 3Ciampa *et al.* (2010), 4Elia *et al.* (2010), 5Laranjeiro *et al.* (2010), 6Ntagwabira and Kang (2010), 7Pinzón *et al.* (2010), 8Tajpour *et al.* (2010), 9Winder (2010), 10Gustav Rydstedt and Jackson (2010), 11Shahriar (2010), 12Xing (2010), 13Kai Zhao; Kaji (2010), 14Jayaraman (2010b), 15Fung (2010), 16Jayaraman (2010a), 17Qiu (2010), 18Balduzzi (2010), 19Yang (2010), 20Gebre (2010), 21Gala´n (2010), 22Watanabe (2010), 23Yinzhi Cao (2010), 24Wang (2010), 25Caballero (2010), 26Beck (2010), 27Chen (2010), 28Silic (2010), 29Jiang Wang; Huang (2010), 30Gaurav Aggrawal and Boneh (2010), 31Aslam (2010), 32De Ryck (2010)
2011	**2012**	**2013**
1Guha (2011), 2Malandrino (2011), 3Qingxian (2011), 4Blanc (2011), 5Fredrikson (2011), 6Saiedian (2011), 7Accornero (2011), 8Fukushima (2011), 9Guo (2011), 10Thomas (2011), 11Devesa (2011), 12Ali (2011), 13Gastellier-Prevost (2011b), 14De Ryck (2011), 15Shun-Wen Hsiao; Sun (2011), 16Boyan Chen; Zavarsky (2011), 17Fung (2011), 18Sood (2011a), 19Anderson (2011), 20Heiderich (2011), 21Arulsuju (2011), 22Weinberg (2011), 23Sood (2011b), 24Pelizzi (2011), 25Yue (2011), 26Chen (2011), 27Siddiqui (2011), 28Gastellier-Prevost (2011a), 29Gastellier-Prevost (2011c), 30Sood and Enbody (2011b), 31Alazab *et al.* (2011), 32Ali *et al.* (2011), 33Khoury *et al.* (2011), 34Maurer (2011), 35Chhabra (2011), 36Akiyama (2011), 37Sood and Enbody (2011a)	1Gundy and Chen (2012), 2Xu *et al.* (2012), 3Zavou (2012), 4Frenz (2012), 5Liu *et al.* (2012), 6Wang *et al.* (2012), 7Shar and Tan (2012), 8Balasundaram and Ramaraj (2012), 9Chandrashekhar *et al.* (2012), 10Shi *et al.* (2012), 11Oriyano and Shimonski (2012), 12Shar (2012a), 13Ojamaa (2012), 14Nunan (2012), 15Gen Lu; Debray (2012), 16Aldwairi (2012), 17Sun (2012a), 18Koizumi (2012), 19Sun (2012b), 20Qurashi (2012), 21Jun Li; Dongting Yu; Maurer (2012), 22Matsuda (2012), 23Wagner (2012), 24Subsorn (2012), 25Yuan (2012), 26Shahriar (2012), 27Lundeen (2012), 28Aggarwal (2012), 29Zhengqin Luo; Rezk (2012), 30Serrhini (2012), 31Maurer (2012), 32Puangpronpitag (2012), 33Shar (2012b), 34Kolbitsch (2012), 35Choudhary (2012), 36Hodovan (2012), 37Jagnere (2012), 38Lekies (2012), 39Orman (2012)	1Vyshegorodtsev (2013), 2Kim (2013), 3Wahlberg (2013), 4Wang (2013b), 5Huang (2013), 6Micha? Chora? (2013), 7Bamrara (2013), 8Johns (2013), 9Bozic (2013), 10Sung (2013), 11Nikiforakis (2013), 12Xu (2013), 13Bielova (2013), 14Ofuonye and Miller (2013), 15Maffeis (2013), 16Stock (2013), 17Wang (2013a), 18Kirda (2013), 19Raichal *et al.* (2013), 20Shar and Tan (2013), 21Shahriar (2013), 22Armando *et al.* (2013), 23Lee *et al.* (2013), 24Hernandez-Ardieta *et al.* (2013), 25Austin *et al.* (2013), 26Stevanovic *et al.* (2013), 27Akiyama (2013), 28Oliveira (2013), 29Shar (2013)

Tabela A.6 Principais contribuições relacionadas ao tema nas fontes acadêmicas