A PDF Vulnerability, and a possible solution

Colin Walker, 2007-09-01

Over recent weeks conversation regarding a pervasive security risk making use of Adobe software on users' systems has cropped up among security groups and hacker forums alike. The Adobe vulnerability quickly garnered a large amount of attention, even from mainstream media.

For a detailed description of the problem, I'd direct you to the webapp security list, (webappsec.org - linked above), but basically this vulnerability is a Cross Site Scripting attack which causes victims to run an attackers script while in the context of the attacked site - any PDF can be used since the problem is with the way the PDF is presented in the browser Adobe has acknowledged the problem and posted information at: http://www.adobe.com/support/security/advisories/apsa07-01.html

Amit Klein has posted on the webappsec mailing list an algorithm that helps mitigate the risk of this vulnerability and once again, we are able to quickly implement this logic with the power of iRules.

The iRule below forces a redirection on PDF requests and uses the redirection to remove fragments (anchors) that were on the original link. It also verifies that the requestor hasn't changed IP addresses, and that they're making the request to the redirect destination URL within 10 seconds of their initial request by passing some additional, encrypted info in the URL and verifying it after the redirection. If things don't line up, or things look suspicious...the iRule simply forces the user to download the PDF, which is the current solution proposed by many.

What does all of this mean? It means people hosting PDF files on their websites that want to greatly reduce the risk of this wide-spread problem, but don't want to force users to download every PDF, every time now have a way.

```
when RULE_INIT {
  set static::pdfKey [AES::key "256"]
}

when HTTP_REQUEST {
  set replaceContentTypeHeader 0
  set uri_lowercase [string tolower [HTTP::uri] ]

  # Do case insensitive check to see if request was for a PDF file
  if { $uri_lowercase contains ".pdf" } {
    # request was for a PDF.  See if the URI contains token_query
    if { not ($uri_lowercase contains "token_query") } {
      # URI didn't contain token_query
      set index [expr [string last ".pdf" $uri_lowercase] + 3]
      set newUri [string range [HTTP::uri] 0 $index]
      set encrypted [b64encode [AES::encrypt $::pdfKey "[clock seconds],[IP::client_addr]" ] ]
      if {[catch {PROFILE::clientssl mode}] == 0} {
        HTTP::redirect "https://[HTTP::host]$newUri?token_query=$encrypted#a"
      } else {
        HTTP::redirect "http://[HTTP::host]$newUri?token_query=$encrypted#a"
      }
     } elseif { [string length [getfield $uri_lowercase "token_query=" 2] ] >0} {
      # since IE6 sp2 (and other IE versions?) includes the anchor after a 302 redirect, remove it if
      set uri [getfield [HTTP::uri] "#" 1]
      set tokenVal [getfield [HTTP::uri] "token_query=" 2]

      if { not ( [catch {b64decode $tokenVal} ]  &&  [catch {AES::decrypt $static::pdfKey [b64decode
```

```
        set decryptedToken [AES::decrypt $static::pdfKey [b64decode $tokenVal]]
        set tokenVal_time [getfield $decryptedToken "," 1]
        set tokenVal_IP [getfield $decryptedToken "," 2]
        HTTP::header remove "If-Modified-Since"
        HTTP::header remove "If-None-Match"
        if { $tokenVal_IP == [IP::client_addr] and ([expr $tokenVal_time + 10] >= [clock seconds] ) }
          # token is valid, so rewrite URI to remove the token_query parameter
          HTTP::uri [getfield [HTTP::uri] "?token_query" 1]
        } else {
          # token is not valid, so force client to download file
          set replaceContentTypeHeader 1
        }
      } else {
        # decryption failed, force client to download PDF"
        set replaceContentTypeHeader 1
      }
    }
  }
}

when HTTP_RESPONSE {
  if { $replaceContentTypeHeader == 1 && [HTTP::status] == 200 }{
    #Replacing Content-Type and Content-Disposition headers to force download"
    HTTP::header replace Content-Disposition "attachment"
    HTTP::header replace Content-Type "application/octet-stream"
    set replaceContentTypeHeader 0
  }
}
```

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc. Corporate Headquarters info@f5.com	F5 Networks Asia-Pacific apacinfo@f5.com	F5 Networks Ltd. Europe/Middle-East/Africa emeainfo@f5.com	F5 Networks Japan K.K. f5j-info@f5.com