SECURE CODE
WARRIOR

OS COMMAND INJECTIONS

We will explain

What an OS Command Injection is, its causes and preventions,
&
some potential hazards.

WHAT ARE OS COMMAND INJECTIONS?

An “OS Command Injection” is a vulnerability that allows arbitrary commands to be executed on the operating system of the application.

WHAT ARE OS COMMAND INJECTIONS?

WHAT CAUSES OS COMMAND INJECTIONS?

This vulnerability can happen when user controlled input, through parameters, cookies, HTTP headers, etcetera

WHAT CAUSES OS COMMAND INJECTIONS?

USER INPUT

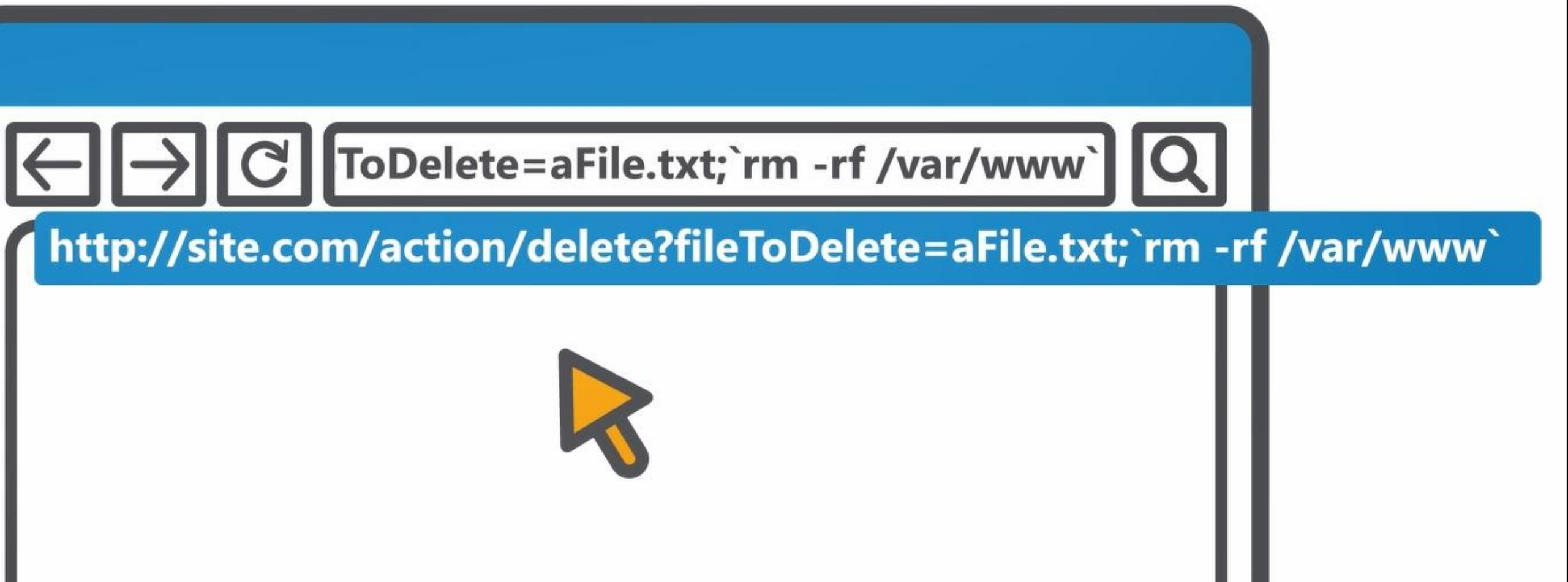
```
webapp@server$ cp -r $FileToDelete /bac
webapp@server$ rm -rf $FileToDelete
```

are passed to the system shell without any prior validation.

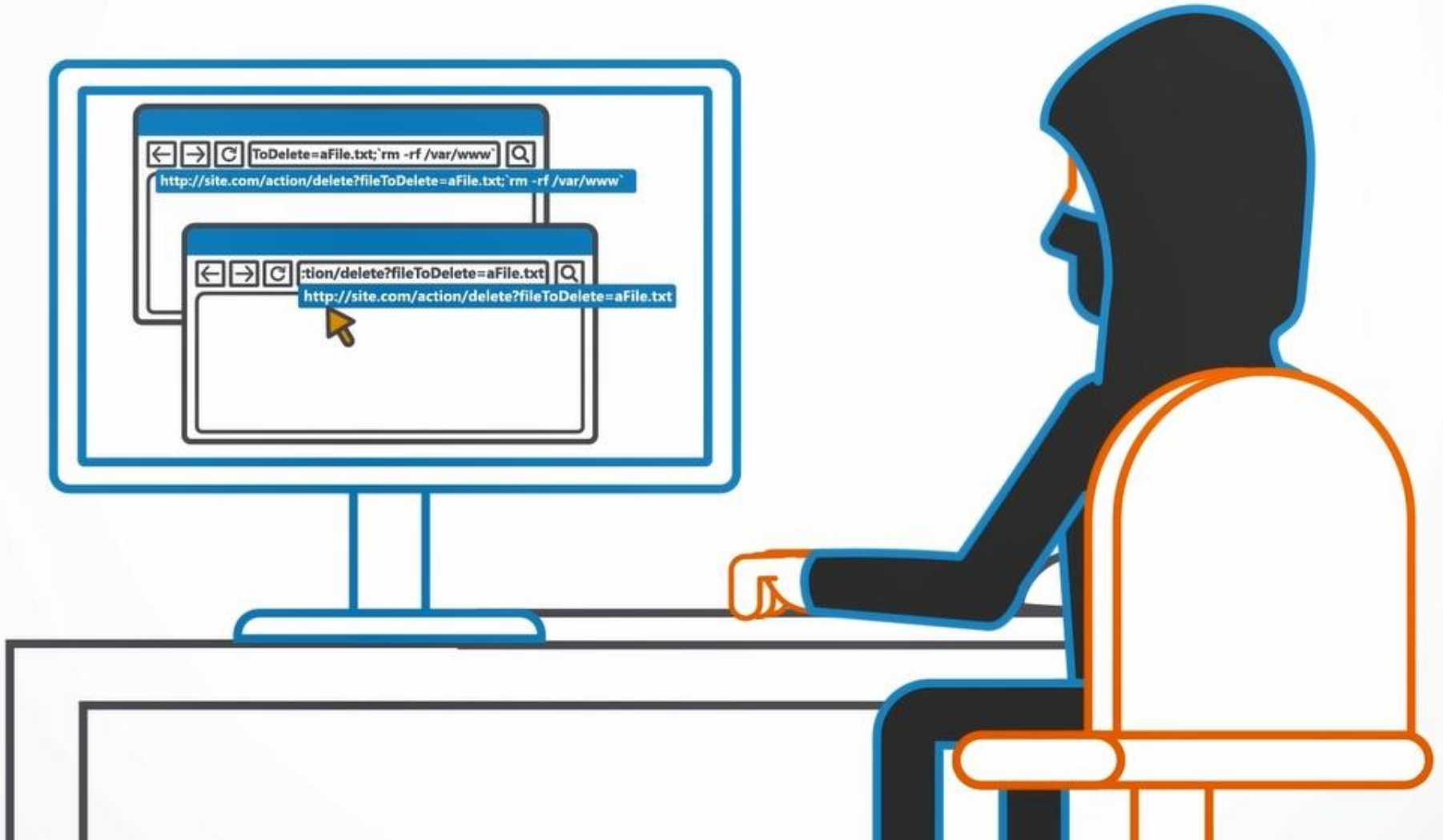
To understand

OS Command Injections,
let's look at an example
of an application with this vulnerability

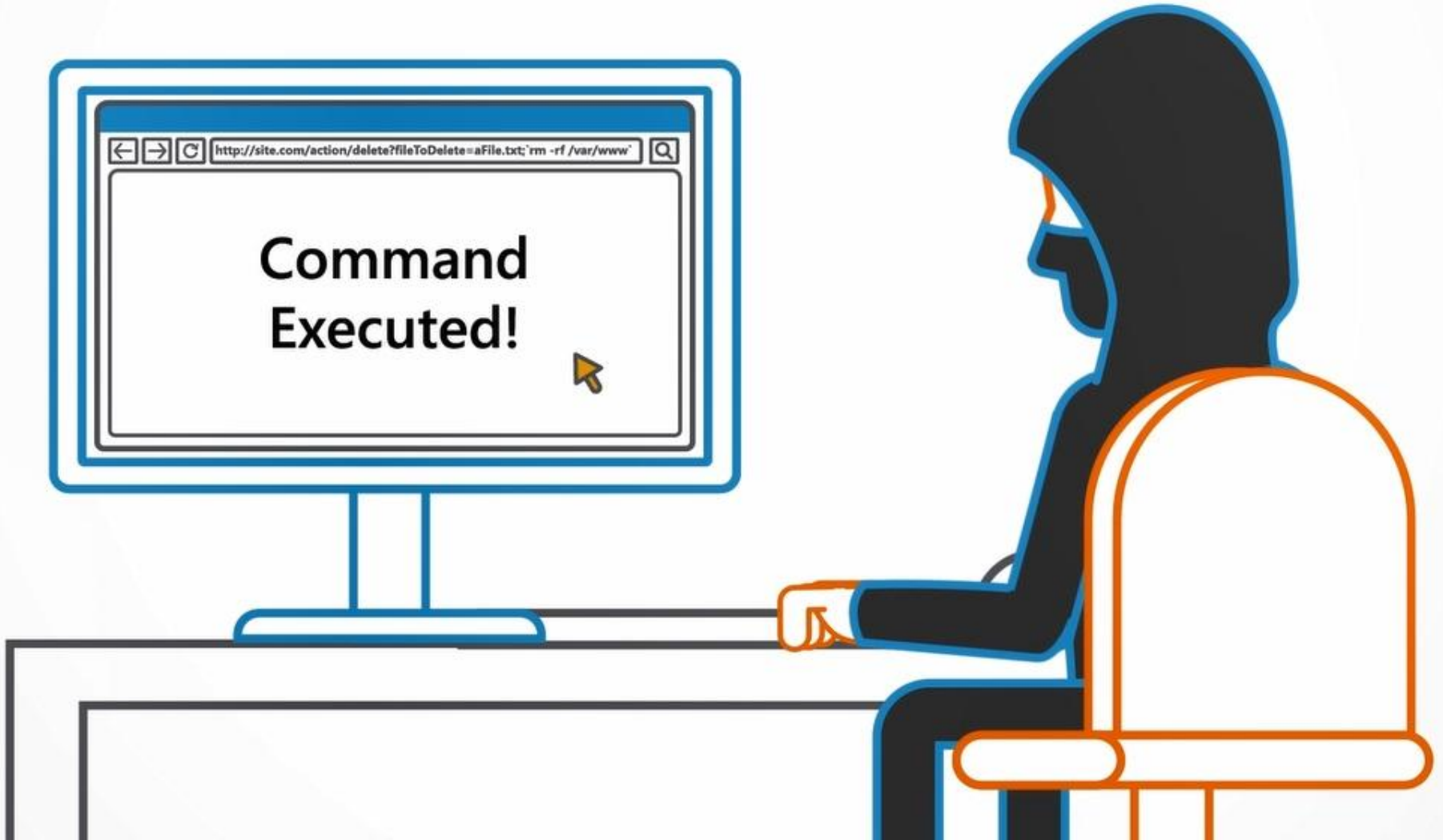
Here, a GET parameter ‘filetoDelete’ is passed to the system shell without prior validation.



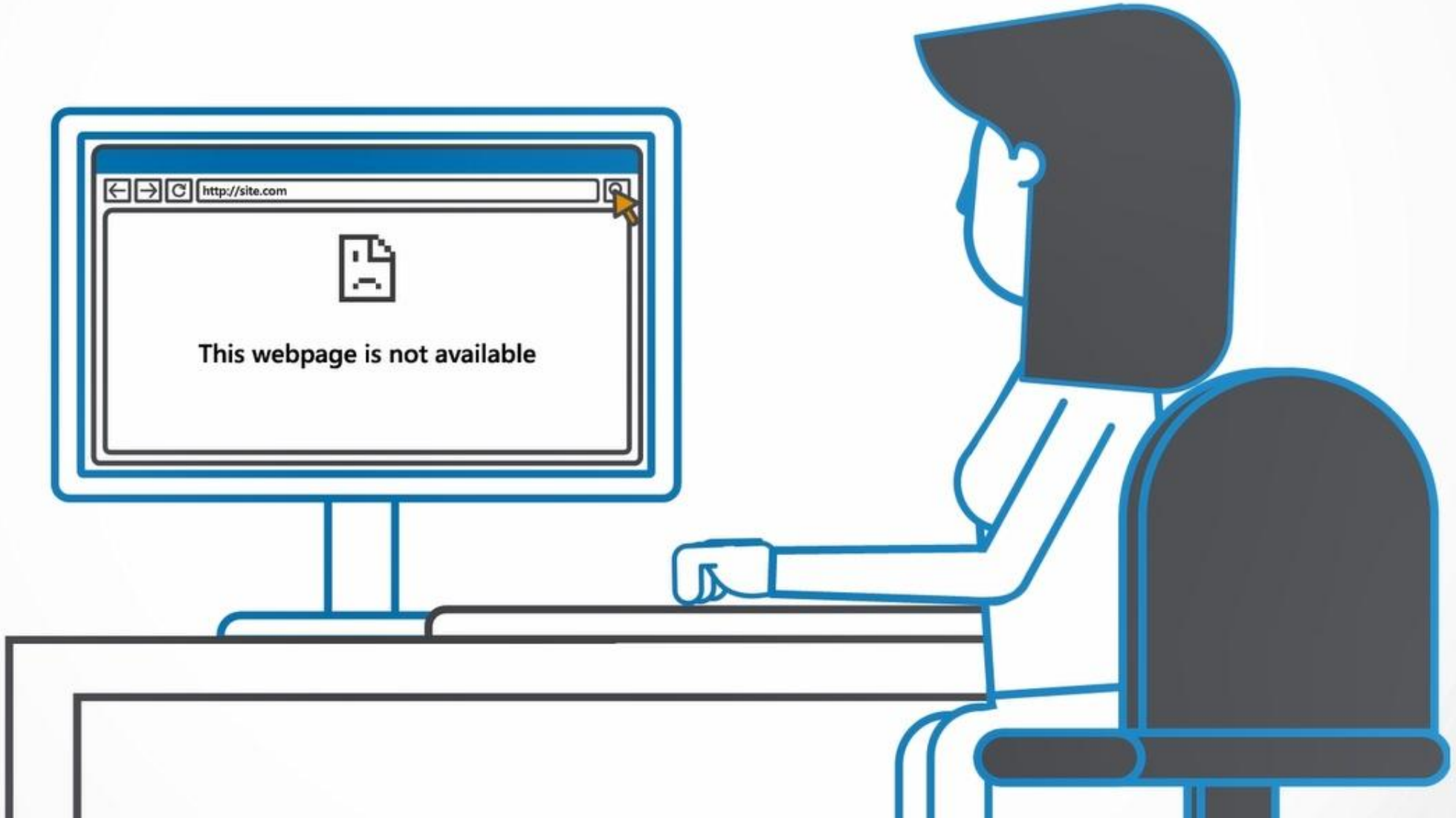
An attacker edits the URL to craft a new, malicious URL: he appends a shell command to the parameter value of a request, like so…



The application appends the GET parameter to the command string and the malicious command is executed.



In this example, all the web application files are deleted. The web application becomes unavailable.



OS Command Injection could have major impacts.

Injected commands will run with the privileges of the vulnerable application.

So, for example, user passwords or other sensitive data could be displayed on the application output.

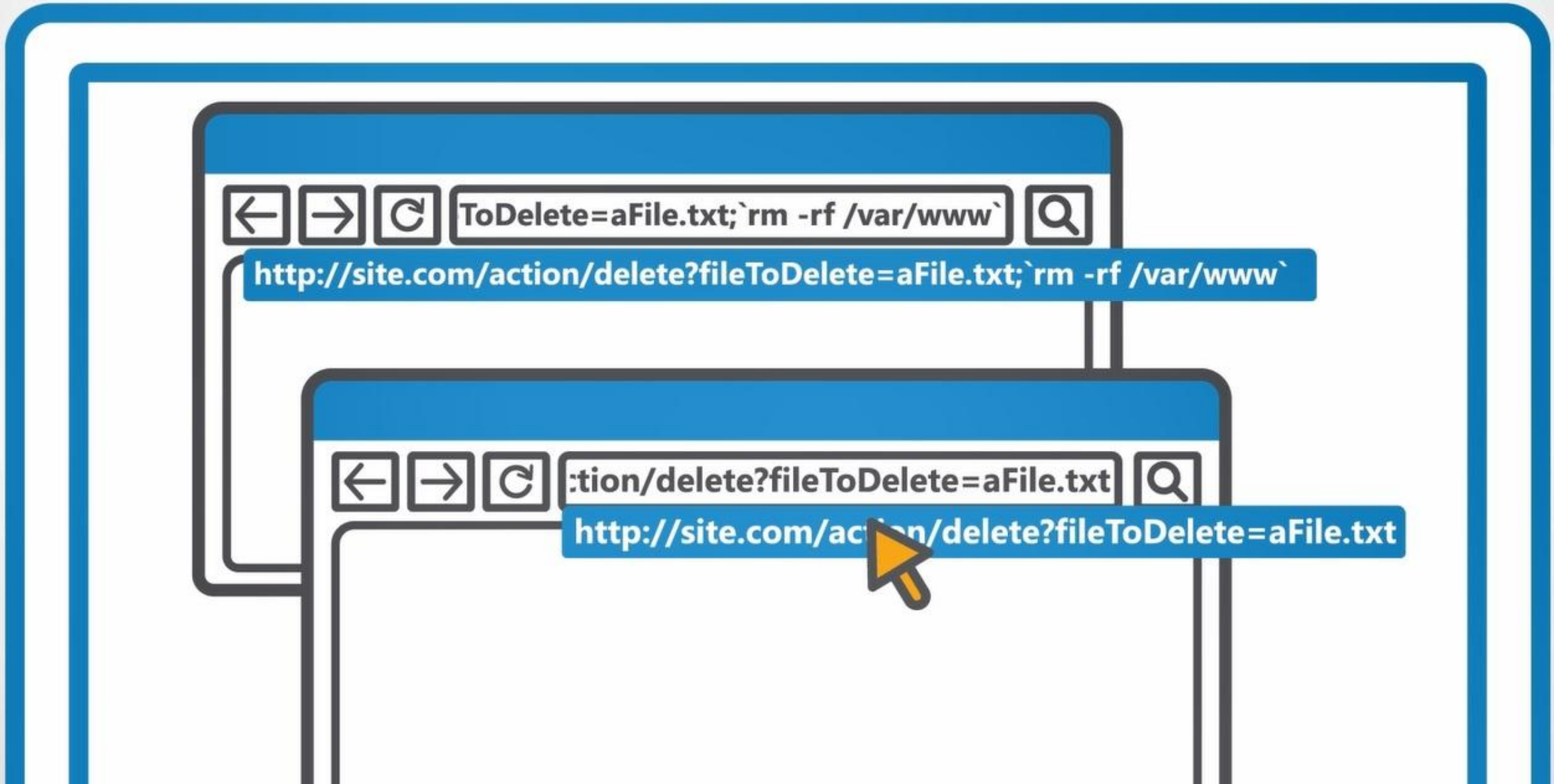
First Name	Last Name	Email Id	Password
John	Doe	john.doe@xyzmail.com	e10adc3949ba72
David	Stuart	david.stuart@xyzmail.com	e10adc332oba59
Jenny	Helen	jenny.helen@xyzmail.com	e10adc3976ba8i
Robert	Brian	david.stuart@xyzmail.com	e10adc39iaba45
Linda	Smith	linda.smith@xyzmail.com	e10adc3649ab63
Mark	Jason	mark.jason@xyzmail.com	e10adc3i43ba22
Betty	Jones	betty.jones@xyzmail.com	e10adc395ab55

Customer data could get exposed leading to privacy issues and a damaged reputation.

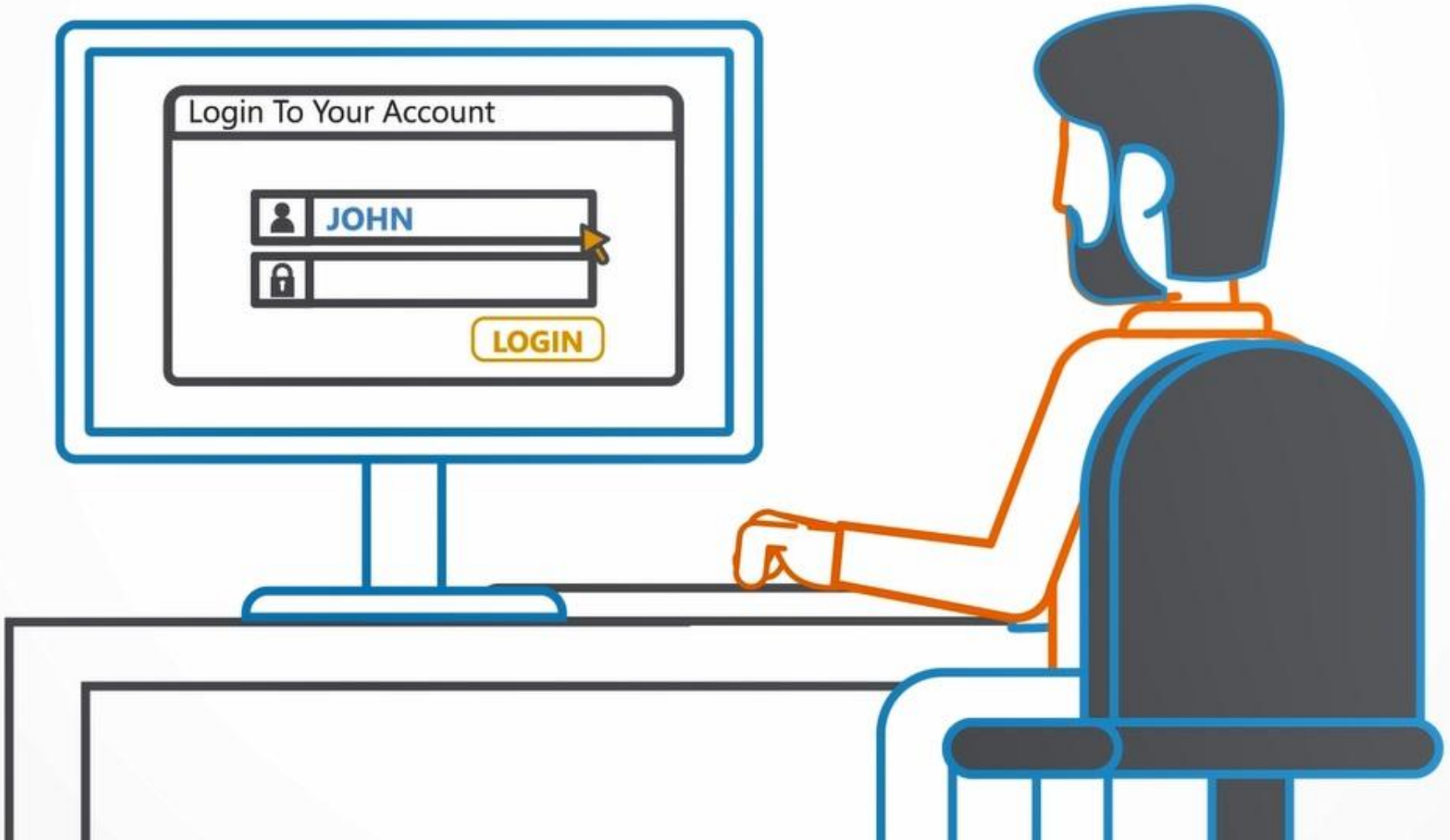
First Name	Last Name	Email Id	Password
John	Doe	john.doe@xyzmail.com	j0#NdOe
David	Stuart	david.stuart@xyzmail.com	D4v!d$tuart
Jenny	Helen	jenny.helen@xyzmail.com	J3nny#eLen
Robert	Brian	david.stuart@xyzm	R08ert8r4!n
Linda	Smith	linda.smith@xyz ail.com	!nd4$m!t#
Mark	Jason	mark.jason@xy nai	n4rkJ4$0n
Betty	Jones	betty.jones@xy	BettyJ0ne$

Commands executed as the application owner could lead to repudiation issues.

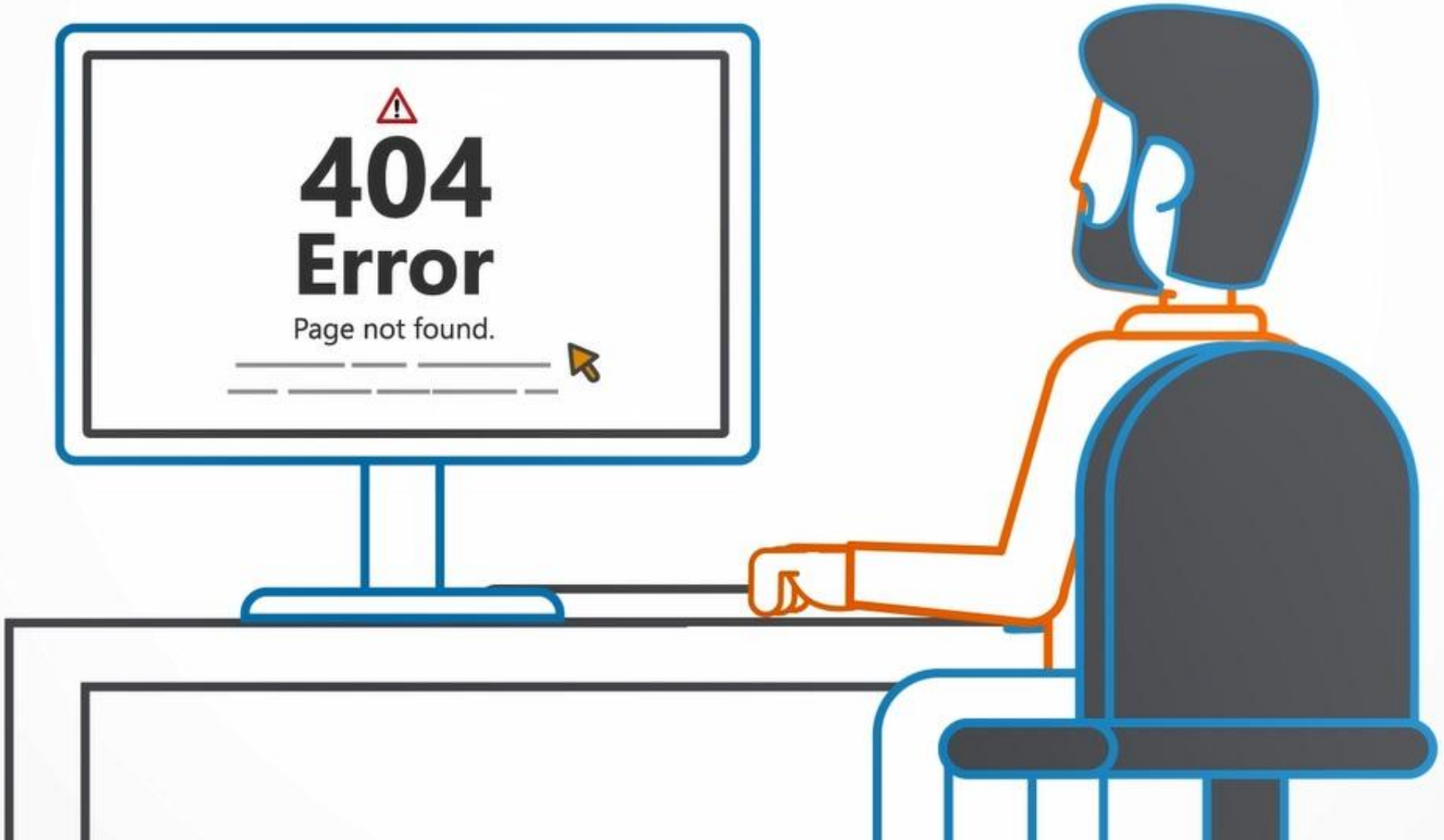
Files or database records could be manipulated or deleted. And services could even be started or stopped.



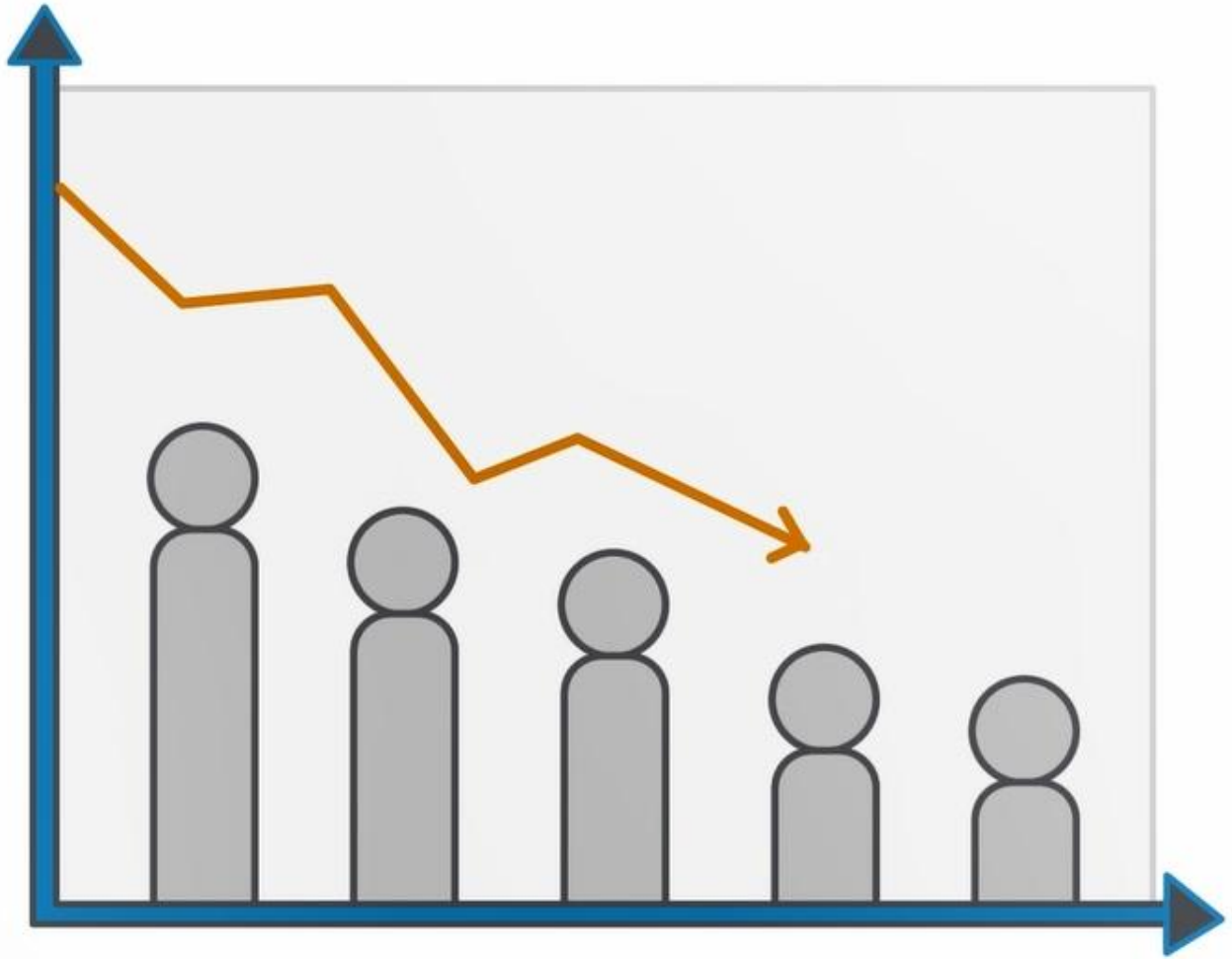
In a worst case scenario



all the files of your application could be deleted



denying service and causing reputation loss and financial damages



Preventing OS Command Injections

Use framework specific API calls instead of OS commands.

If this is not possible

you should validate all user controlled output against an allowlist before passing it to the shell.

This validation should include

- POST and GET parameters
- Cookies
- HTTP headers

Always apply the principle
of
least privilege to the application.

Congratulations,

you have now completed this module, OS Command Injections!

www.securecodewarrior.com