Before we start:

```
$ git clone https://github.com/PentesterLab/codereview-php

$ git clone https://github.com/PentesterLab/codereview-golang
```

https://github.com/snyff/Talks/blob/master/Intro_Code_Review_Owasp_BA.pdf

Web Security Code Review Workshop

Louis Nyffenegger
louis@pentesterlab.com

ABOUT ME:

- Founder and CEO of PentesterLab
- Ex: Pentester, Code Reviewer, AppSec Engineer

- Online Platform to Learn Code Review and Web Hacking / Web Penetration Testing
- Online Live Training Sessions on Web Security Code Review

This WorkShop

- Introduction
- Routing
- Patterns
- CVE Analysis
- CVE-2008-1930
- Conclusion
- Hands-On Code Review

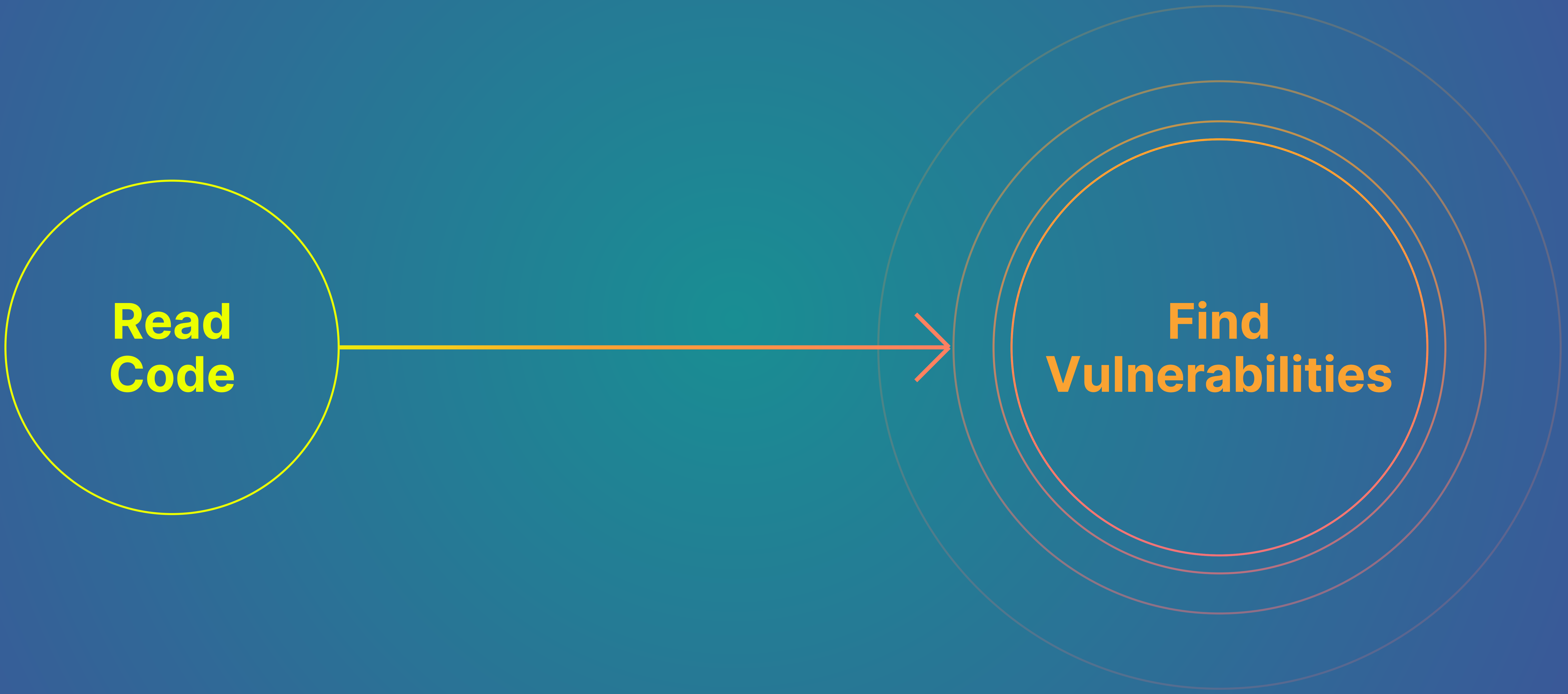
INTRODUCTION

Security Code Review is in demand

- Ability to find complex bugs
- Ability to find bugs that scanners can't find
- Ability to review changes prior to deployment (Agile, AppSec)
- Ability to find new classes of vulnerabilities

- Powerful skill for:
  - Developers
  - Penetration Testers
  - Security Engineers
  - Vulnerability Researchers / Exploit writers
  - QA/Test Engineer

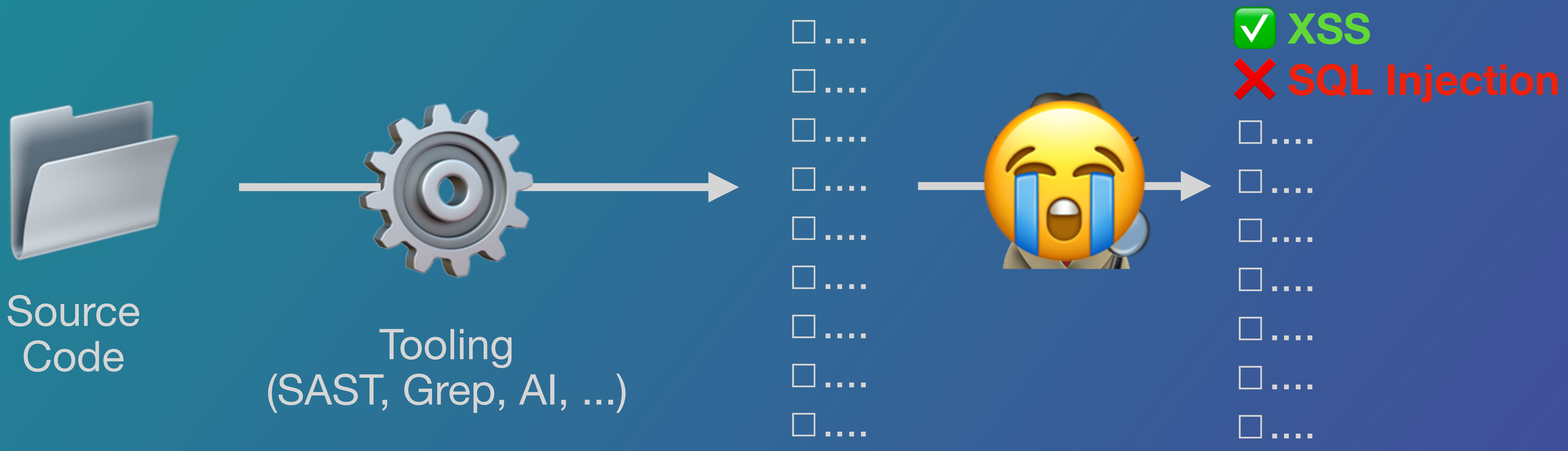
Expectations…

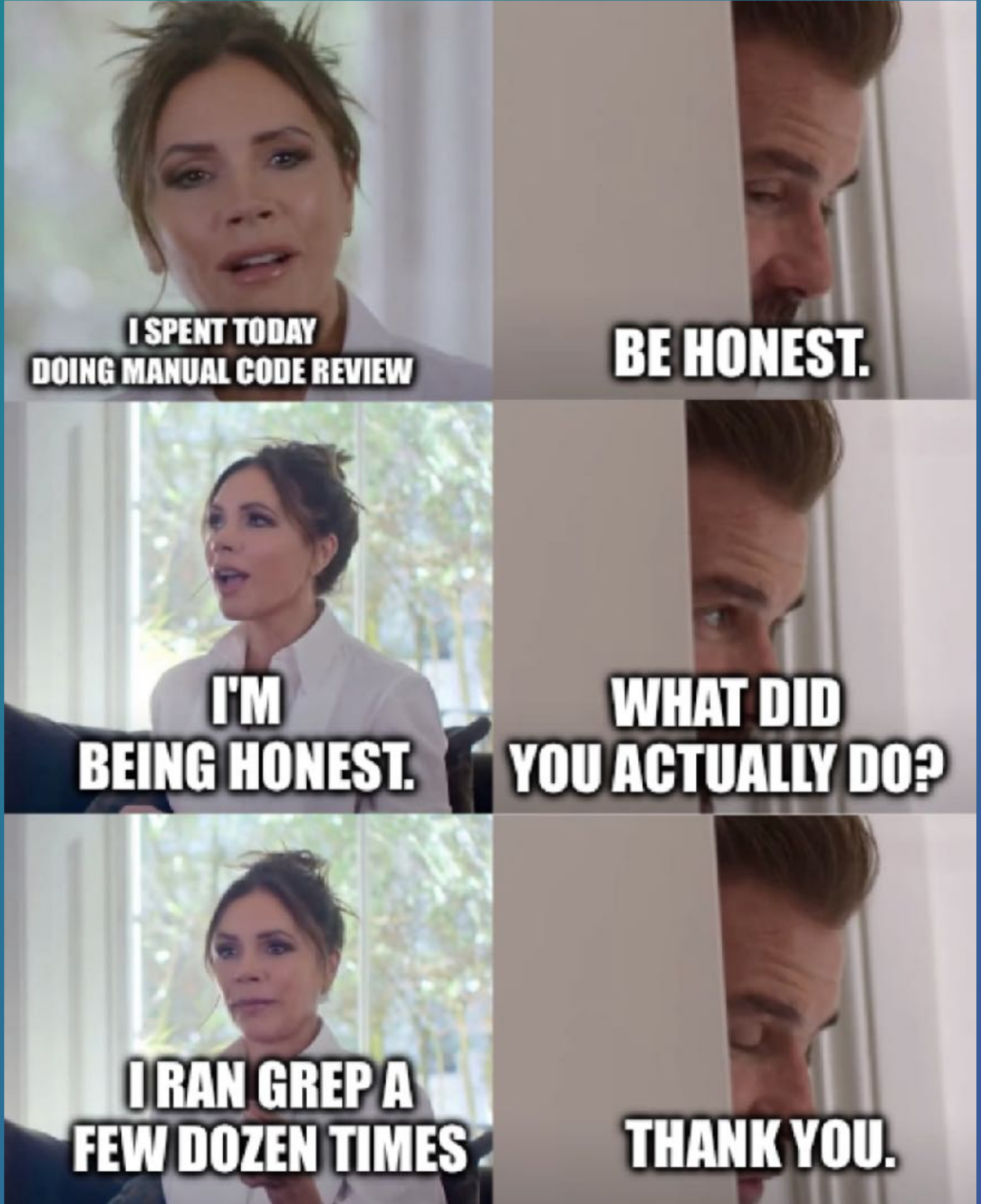


Reality…

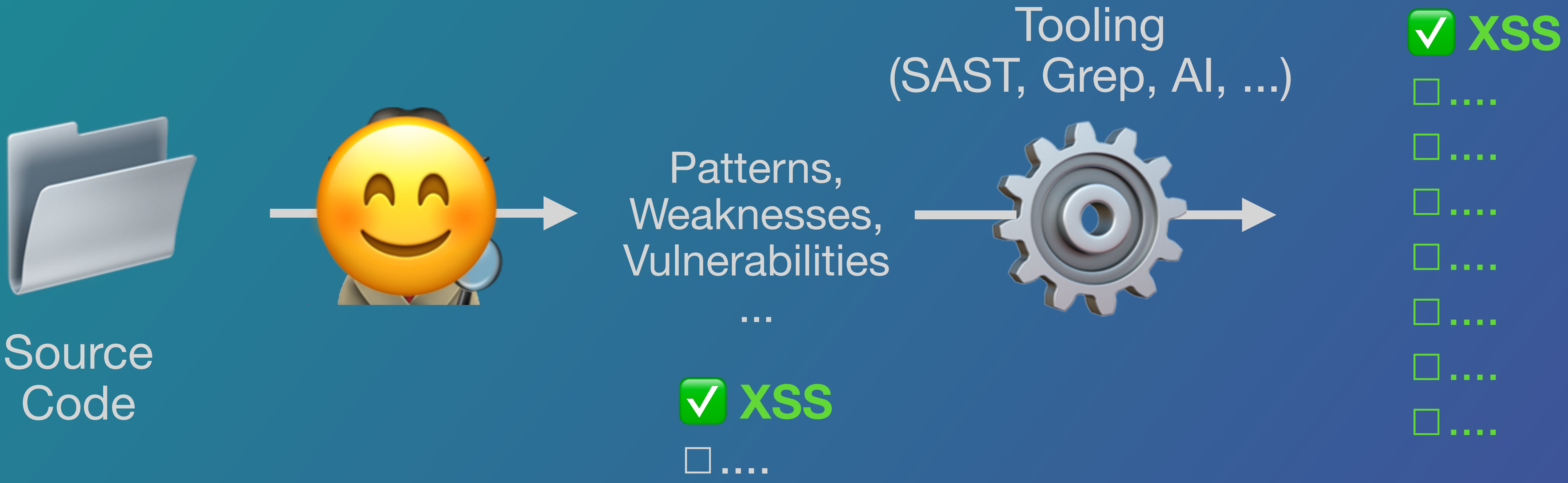
Find Nothing...
Read Code
Fuzz Snippet...
Find Vulnerabilities
Read Documenation
Read *More* Code
Read *More* Code
Deploy Test Environment
Find Weaknesses

Security Code Review...




I SPENT TODAY DOING MANUAL CODE REVIEW
BE HONEST.
I'M BEING HONEST.
WHAT DID YOU ACTUALLY DO?
I RAN GREP A FEW DOZEN TIMES
THANK YOU.

Security Code Review...


Source Code
Patterns, Weaknesses, Vulnerabilities ...
✅ XSS
☐ ....
Tooling (SAST, Grep, AI, ...)
✅ XSS
☐ ....
☐ ....
☐ ....
☐ ....
☐ ....
☐ ....
☐ ....

One of the main advantages of this approach is that it helps identify "unknown unknowns"—issues that automated tools may overlook.

Should you know the language?

A good rule of thumb is that you need to know things that developers don't know:

- Something about a format used?
- A way to bypass a filter?
- Something about threat modelling?
- Something about the language?

The more things you know that the developers don't, the more likely you are to find vulnerabilities

Should you know how to write code?

- It definitely helps!
- You don't need to be a "real" developer but knowing how applications are developed will speed up your work
- The more code you write, the more likely you are to guess:
  - What mistakes developers will make?
  - What shortcuts developers will take?
- The less you know, the more patient you will have to be

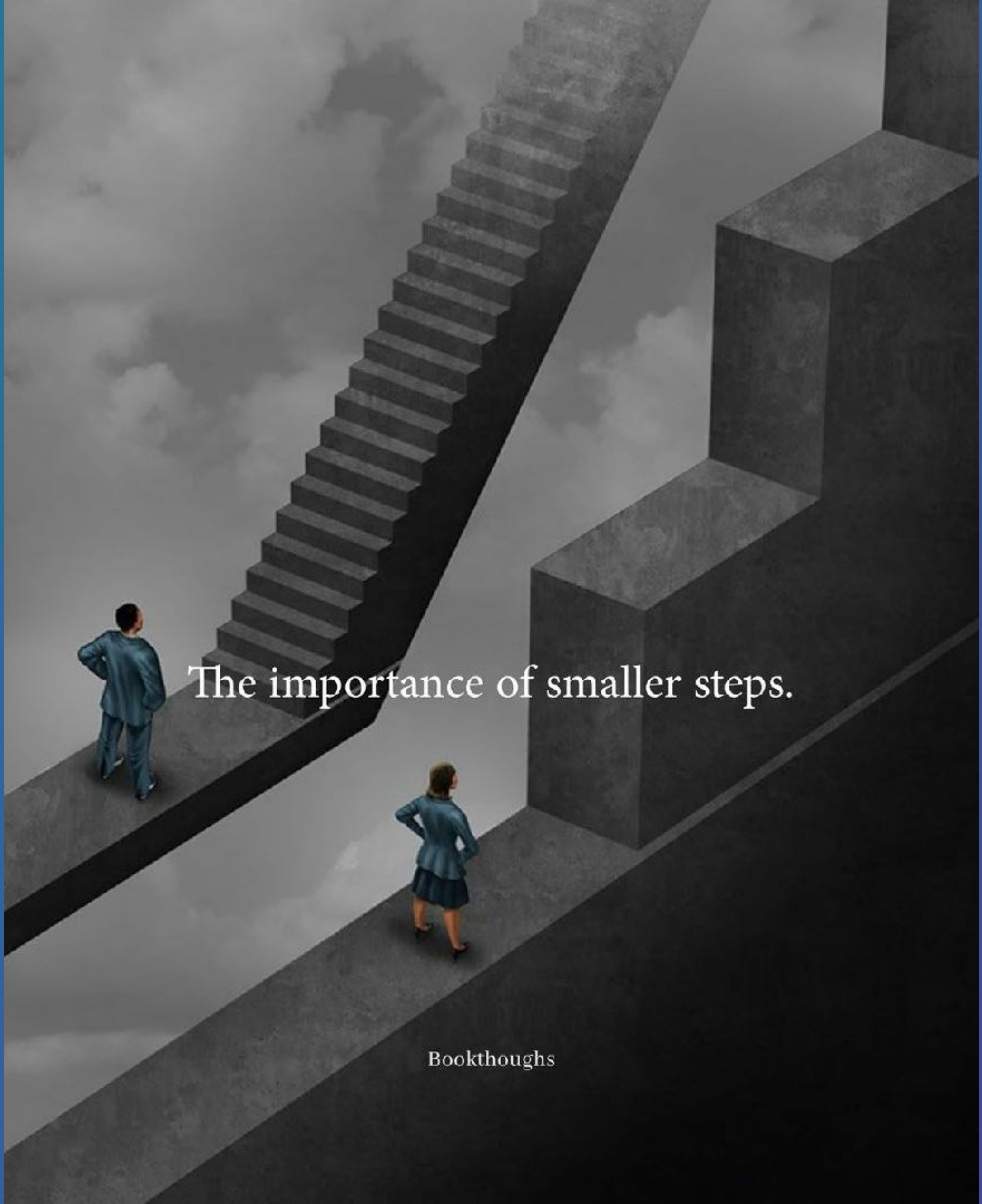
Threat modeling

- Key component of code review
- If you don't know what can go wrong, you don't know what to check for
- Knowing about common bug classes for each type of feature or application is key
- Threat modeling gives "direction" to your review

Threat modeling: How to learn?

- There are many methodologies for threat modeling
- For web security code reviews, your best options are to:
  - Read pentest reports
  - Read bug bounty findings and write-ups
  - Follow research presented at conferences
  - Analyze CVE

Picking your targets to learn...



Picking your targets to learn...

- You need to find targets that are not too easy
- You need to find targets that are not too hard

- You need to find targets that allow you to grow
- You need to find targets to **build resilience**

Picking your targets to learn...

1. Snippets
2. Diff from known/public vulnerabilities/CVE
3. Small or simple Libraries
4. Bigger or more complex Libraries
5. Small Applications
6. Larger Applications
7. Hard Targets

Defining Success in Security Code Review

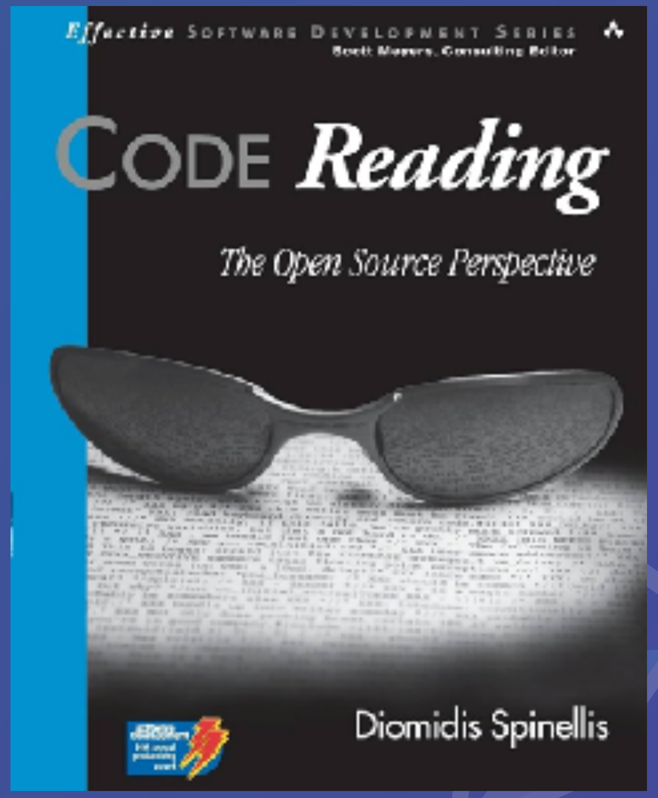
- You don't want to base your success based on the number of vulnerabilities you find or the impact of the vulnerabilities you find (especially when learning)

- Success should be based on:
  - Your progression in understanding a codebase
  - Learning ways a check or filter is implemented
  - Finding small weaknesses or potential improvements
  - Understanding complex patterns
  - Discovering new patterns (with and without security implications)

READING CODE

🤓

"An hour of code reading can save you a minute of reading the documentation"


Effective Software Development Series
Scott Meyers, Consulting Editor
Code Reading
The Open Source Perspective
Diomidis Spinellis

READING THE CODE

Notice:

- Things that are unusual
- When people reinvent the wheel
- Sketchy code
- Complexity
- Unchecked return values
- Checks:
  - What are they trying to prevent?
  - Are they preventing it properly?
  - Is there something else they should take care of but they don't?

Reading code

- Every time you encounter a new function or method:
  - Read the documentation
  - Look for potential security improvements and issues
  - "Fuzz" it (REPL or docker)
  - Keep notes

**Bonus point for doing this over multiple versions of the same method/function...*

Routing

Routing?

How an application maps:

https://...../foo/1234/bar to actual code...

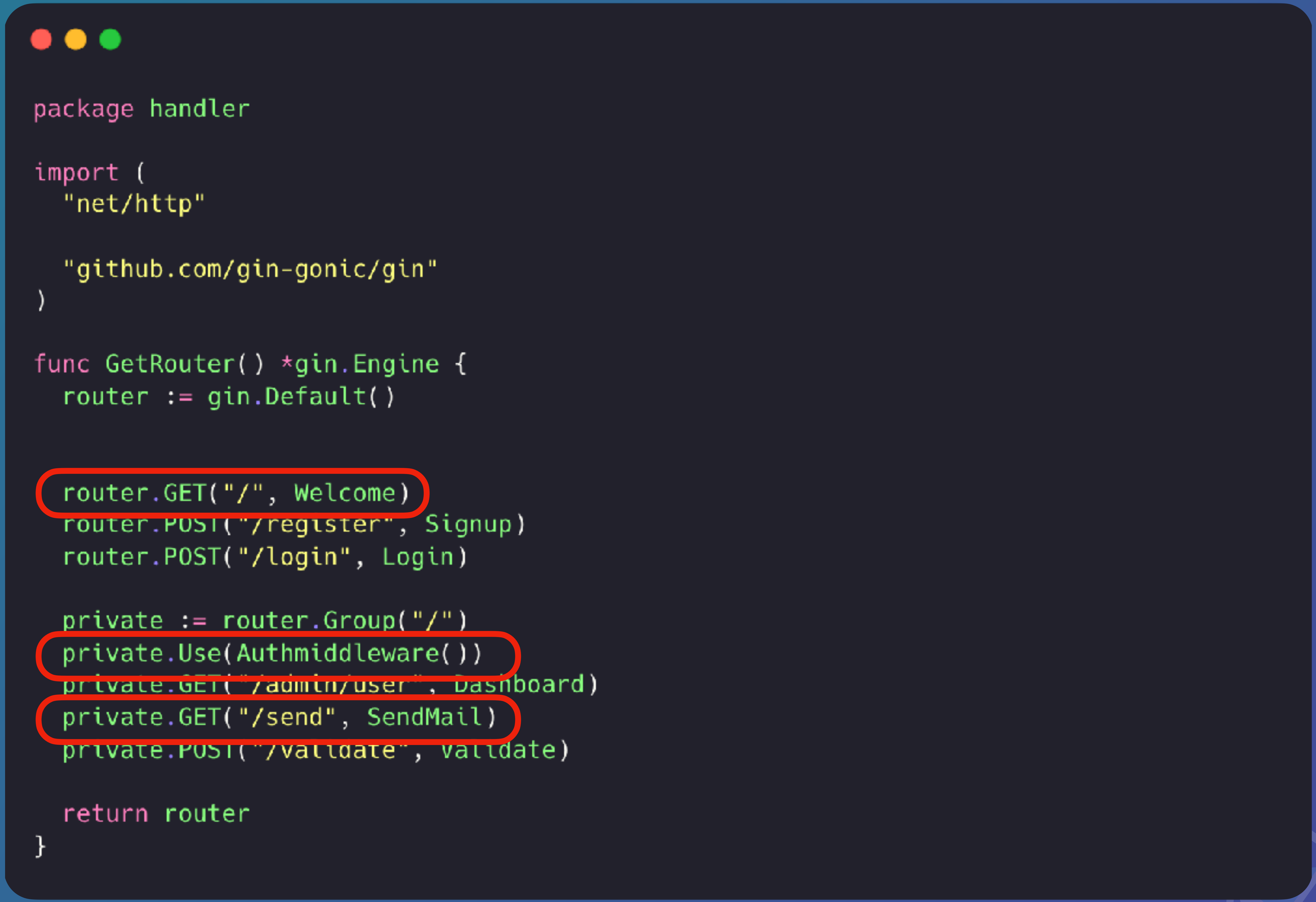
- And what is the impact on
  - What you need to review?
  - How you will perform your review?
- Multiple ways to define routing: FS, programmatically, configuration

File System based

- Very common with (old, small, pure, immature) applications (mainly PHP)
- Accessing /index.php is mapped to running the code in the file [WEBROOT]/index.php
- Any file in the web root can potentially be accessed.
- The file's extension or the file's location will decide if the file gets:
    - interpreted/executed: the result of the execution is returned to the client.
    - served: the content of the file is returned to the client.

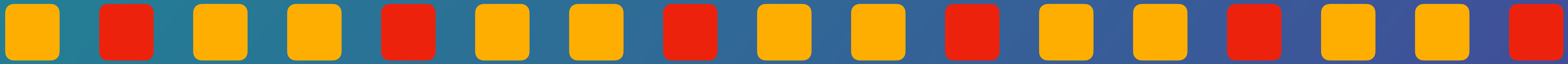
Programmatically defined

- Code is used to map a route to code



```go
package handler

import (
  "net/http"

  "github.com/gin-gonic/gin"
)

func GetRouter() *gin.Engine {
  router := gin.Default()


  router.GET("/", Welcome)
  router.POST("/register", Signup)
  router.POST("/login", Login)

  private := router.Group("/")
  private.Use(Authmiddleware())
  private.GET("/admin/user", Dashboard)
  private.GET("/send", SendMail)
  private.POST("/validate", Validate)

  return router
}
```

PATTERNS…



Patterns

When the sage points at the moon, the fool looks at the finger.

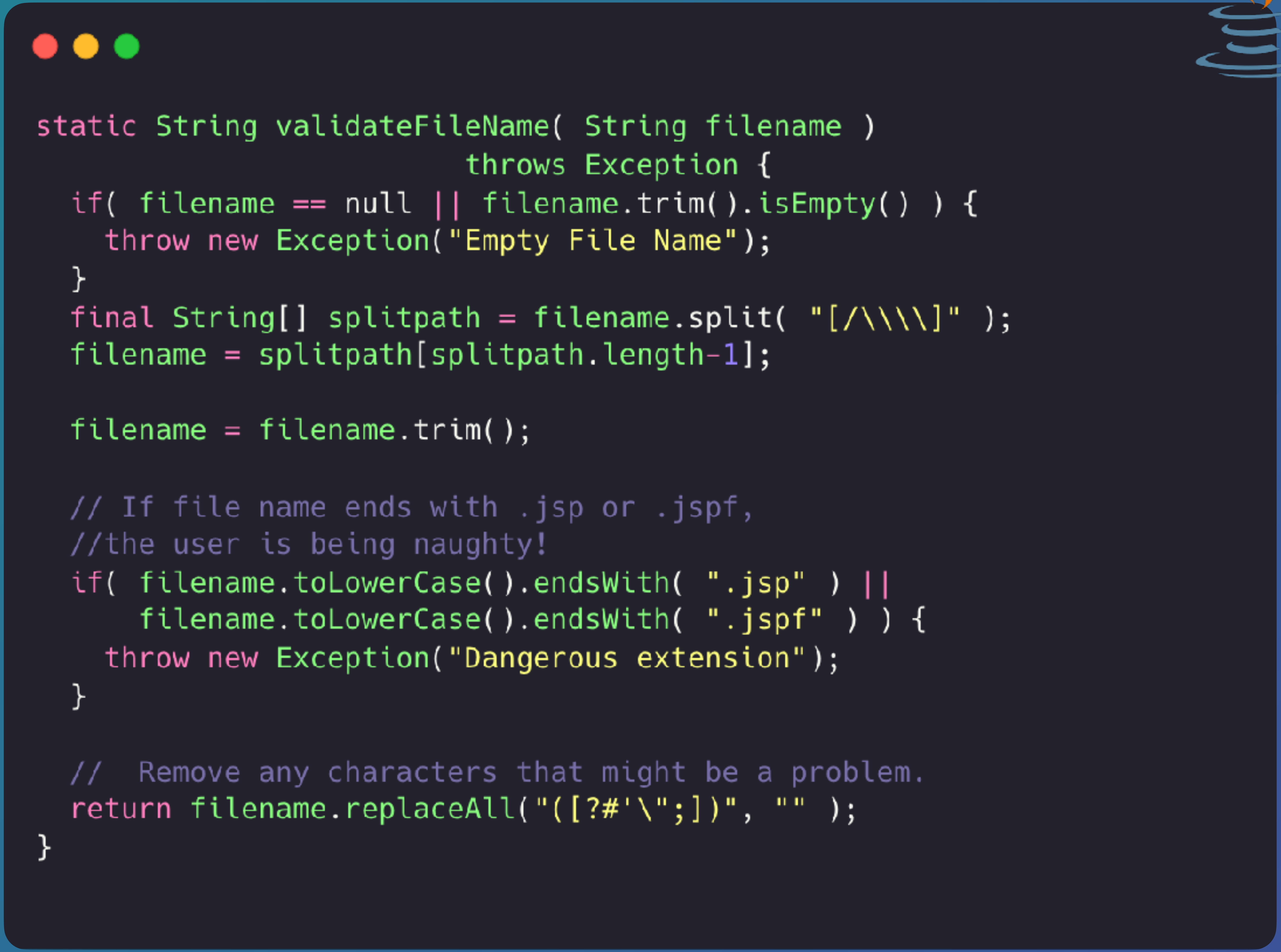
- A lot of issues in security are completely independent of the programming language
- In this section, we are going to explore patterns with implementation in multiple programming languages
- Make sure you focus on the pattern

Filter -> Modify -> Use

- The code does three things:
  1. Filters for malicious values
  2. Modifies the value
  3. Uses the value

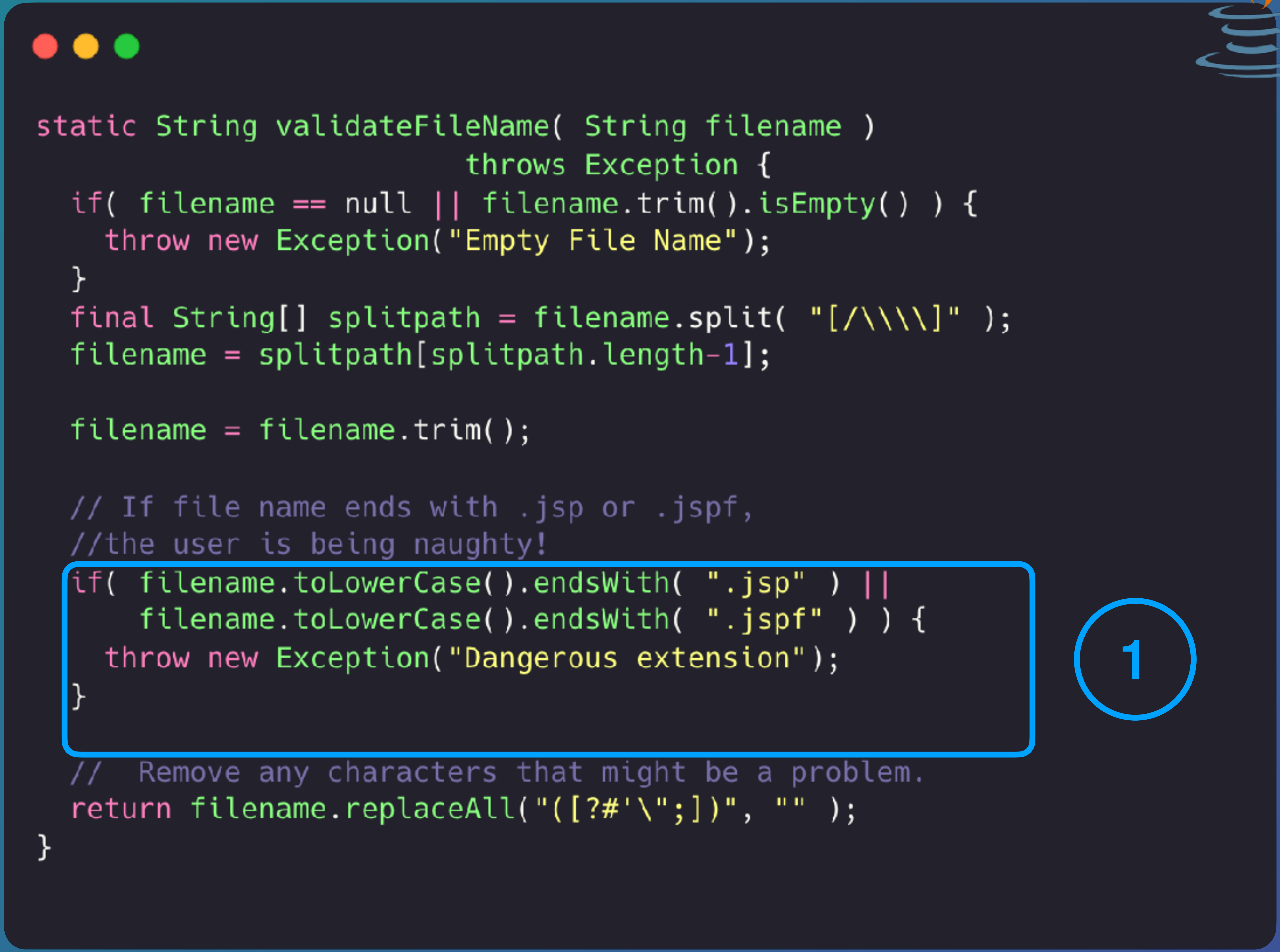
CAN WE REINTRODUCE SOME OF THE FILTERED VALUES BACK USING THE MODIFICATION?

Filter -> Modify -> Use



```java
static String validateFileName( String filename )
                         throws Exception {
  if( filename == null || filename.trim().isEmpty() ) {
    throw new Exception("Empty File Name");
  }
  final String[] splitpath = filename.split( "[/\\\\]" );
  filename = splitpath[splitpath.length-1];

  filename = filename.trim();

  // If file name ends with .jsp or .jspf,
  //the user is being naughty!
  if( filename.toLowerCase().endsWith( ".jsp" ) ||
      filename.toLowerCase().endsWith( ".jspf" ) ) {
    throw new Exception("Dangerous extension");
  }

  //  Remove any characters that might be a problem.
  return filename.replaceAll("([?#'\";])", "" );
}
```

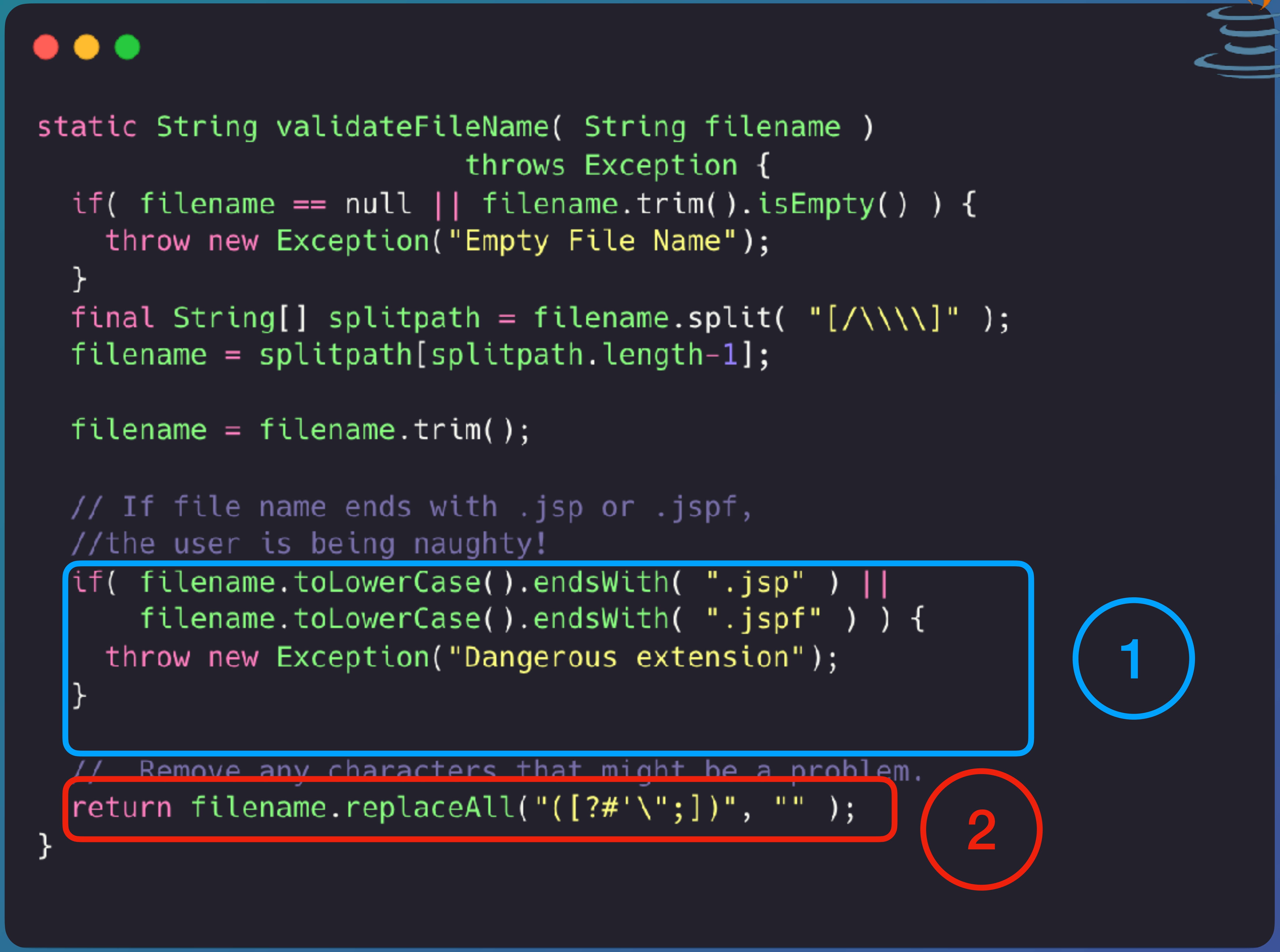
Filter -> Modify -> Use



```java
static String validateFileName( String filename )
                             throws Exception {
  if( filename == null || filename.trim().isEmpty() ) {
    throw new Exception("Empty File Name");
  }
  final String[] splitpath = filename.split( "[/\\\\]" );
  filename = splitpath[splitpath.length-1];

  filename = filename.trim();

  // If file name ends with .jsp or .jspf,
  //the user is being naughty!
  if( filename.toLowerCase().endsWith( ".jsp" ) ||
      filename.toLowerCase().endsWith( ".jspf" ) ) {
    throw new Exception("Dangerous extension");
  }

  //  Remove any characters that might be a problem.
  return filename.replaceAll("([?#'\";])", "" );
}
```

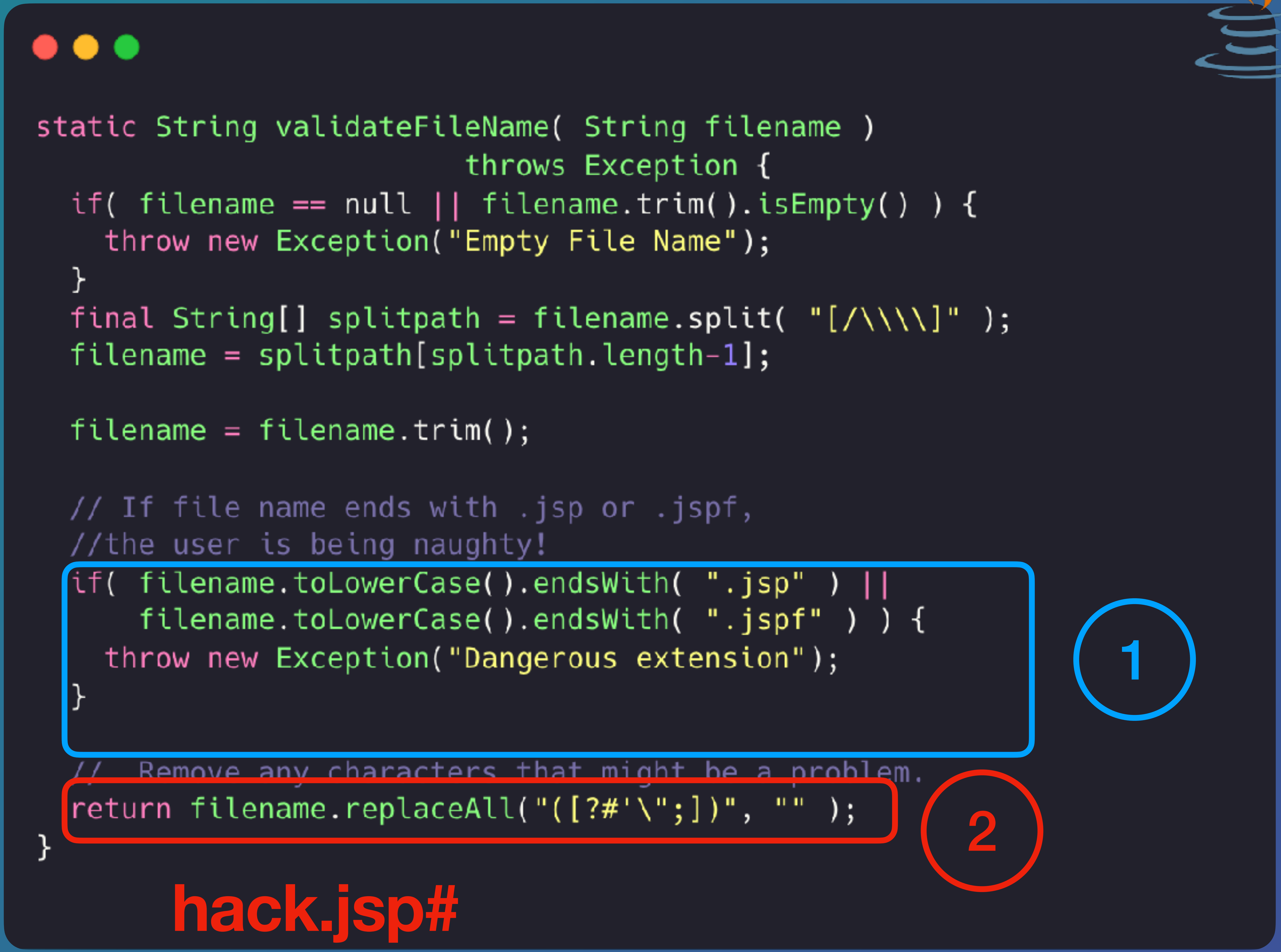
1

Filter -> Modify -> Use



```java
static String validateFileName( String filename )
                             throws Exception {
  if( filename == null || filename.trim().isEmpty() ) {
    throw new Exception("Empty File Name");
  }
  final String[] splitpath = filename.split( "[/\\\\]" );
  filename = splitpath[splitpath.length-1];

  filename = filename.trim();

  // If file name ends with .jsp or .jspf,
  //the user is being naughty!
  if( filename.toLowerCase().endsWith( ".jsp" ) ||
      filename.toLowerCase().endsWith( ".jspf" ) ) {
    throw new Exception("Dangerous extension");
  }

  //  Remove any characters that might be a problem.
  return filename.replaceAll("([?#'\";])", "" );
}
```

Filter -> Modify -> Use



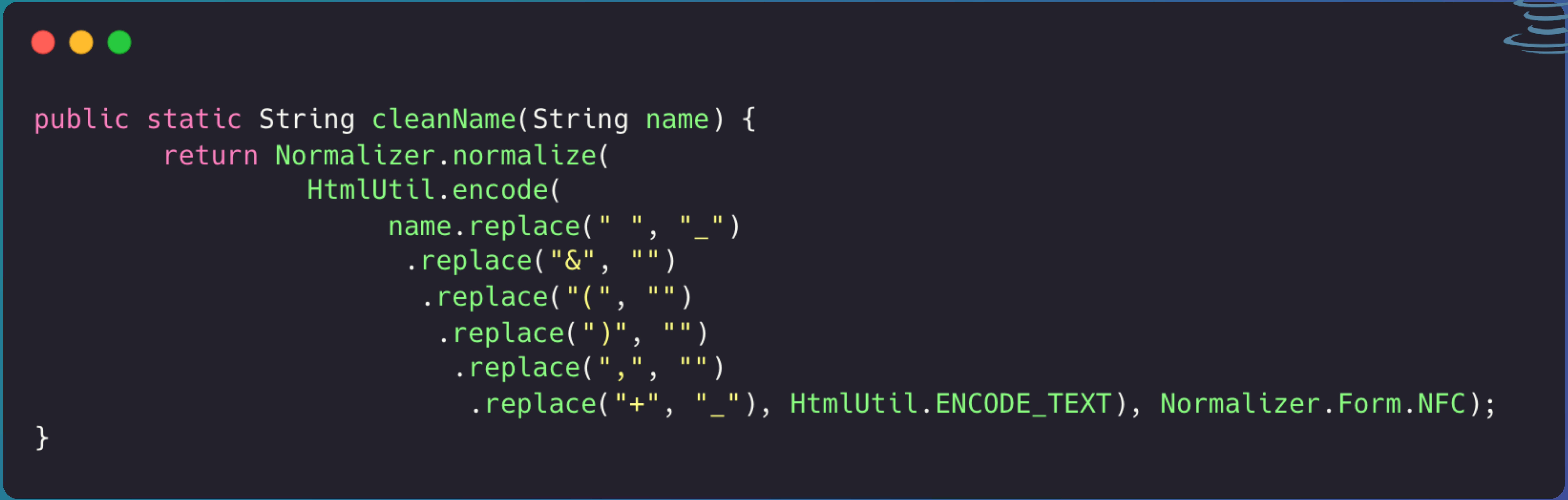
```java
static String validateFileName( String filename )
                            throws Exception {
  if( filename == null || filename.trim().isEmpty() ) {
    throw new Exception("Empty File Name");
  }
  final String[] splitpath = filename.split( "[/\\\\]" );
  filename = splitpath[splitpath.length-1];

  filename = filename.trim();

  // If file name ends with .jsp or .jspf,
  //the user is being naughty!
  if( filename.toLowerCase().endsWith( ".jsp" ) ||
      filename.toLowerCase().endsWith( ".jspf" ) ) {
    throw new Exception("Dangerous extension");
  }

  // Remove any characters that might be a problem.
  return filename.replaceAll("([?#'\";])", "" );
}
```

1

2

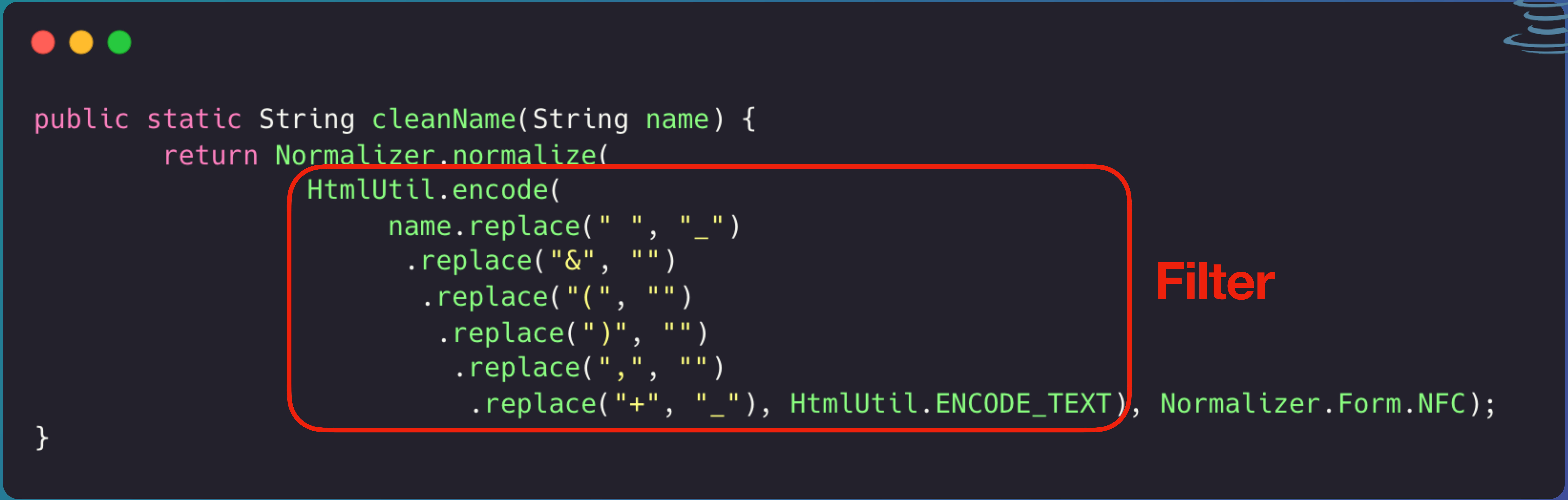
hack.jsp#

Filter -> Modify -> Use



```java
public static String cleanName(String name) {
        return Normalizer.normalize(
                 HtmlUtil.encode(
                      name.replace(" ", "_")
                       .replace("&", "")
                        .replace("(", "")
                         .replace(")", "")
                          .replace(",", "")
                           .replace("+", "_"), HtmlUtil.ENCODE_TEXT), Normalizer.Form.NFC);
}
```

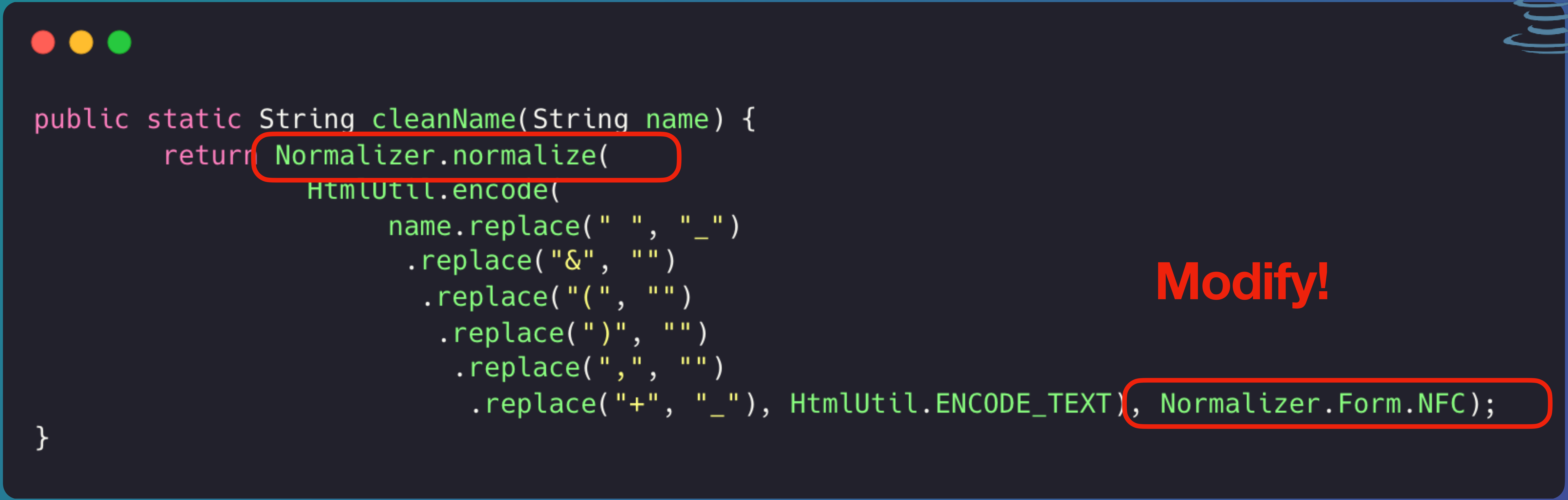
Filter -> Modify -> Use



```
public static String cleanName(String name) {
        return Normalizer.normalize(
                  HtmlUtil.encode(
                       name.replace(" ", "_")
                        .replace("&", "")
                         .replace("(", "")
                          .replace(")", "")
                           .replace(",", "")
                            .replace("+", "_"), HtmlUtil.ENCODE_TEXT), Normalizer.Form.NFC);
}
```

Filter

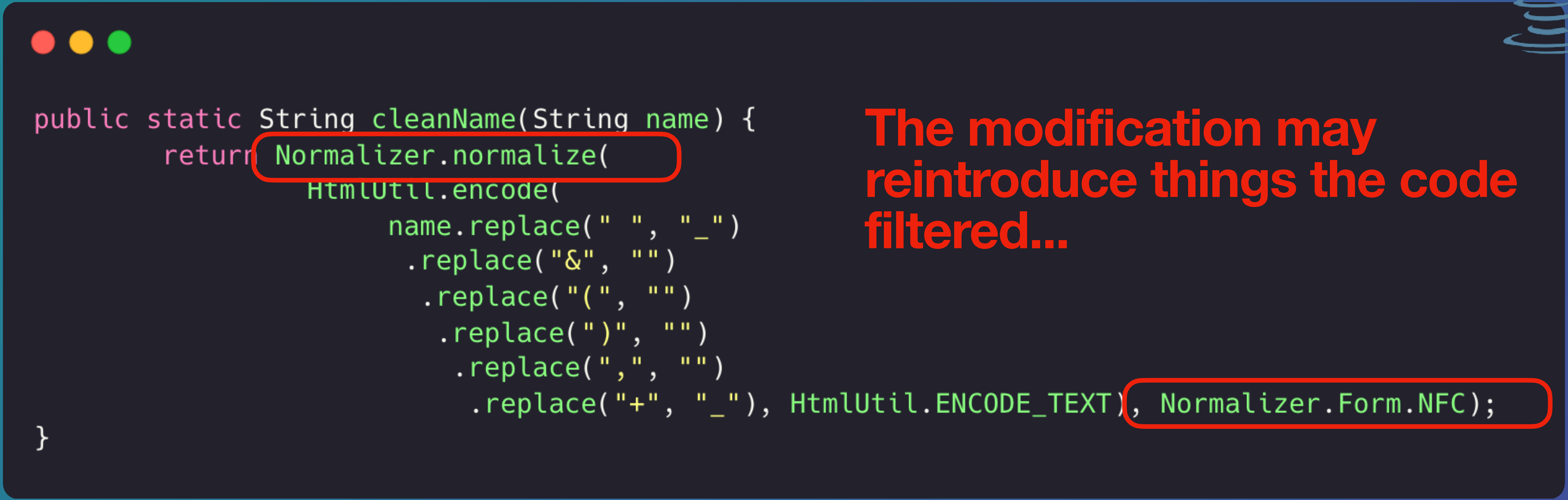
Filter -> Modify -> Use



```
public static String cleanName(String name) {
        return Normalizer.normalize(
                HtmlUtil.encode(
                        name.replace(" ", "_")
                         .replace("&", "")
                          .replace("(", "")
                           .replace(")", "")
                            .replace(",", "")
                             .replace("+", "_"), HtmlUtil.ENCODE_TEXT), Normalizer.Form.NFC);
}
```

Modify!

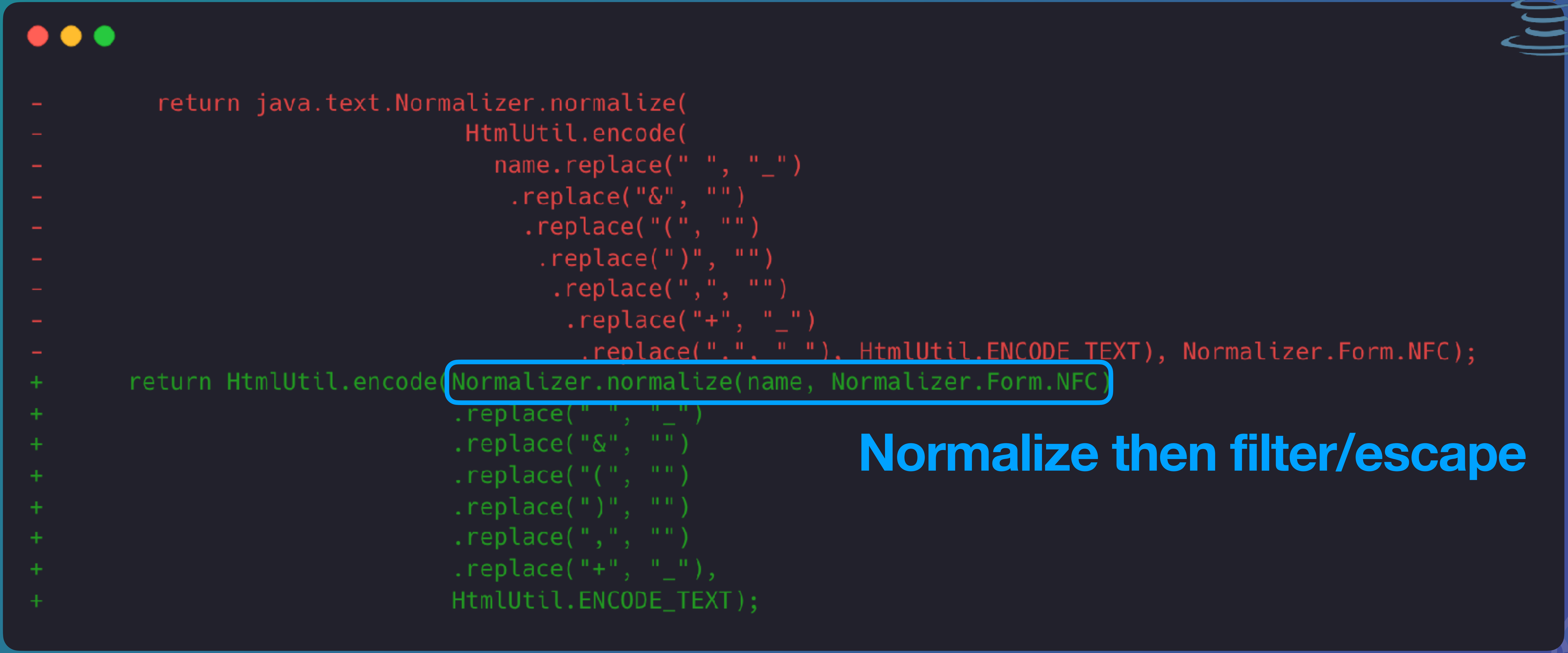
Filter -> Modify -> Use



```
public static String cleanName(String name) {
        return Normalizer.normalize(
                HtmlUtil.encode(
                        name.replace(" ", "_")
                         .replace("&", "")
                          .replace("(", "")
                           .replace(")", "")
                            .replace(",", "")
                             .replace("+", "_"), HtmlUtil.ENCODE_TEXT), Normalizer.Form.NFC);
}
```

The modification may reintroduce things the code filtered...

Filter -> Modify -> Use



```
-        return java.text.Normalizer.normalize(
                              HtmlUtil.encode(
                                name.replace(" ", "_")
                                 .replace("&", "")
                                  .replace("(", "")
                                   .replace(")", "")
                                    .replace(",", "")
                                     .replace("+", "_")
                                      .replace(".", "_"), HtmlUtil.ENCODE_TEXT), Normalizer.Form.NFC);
+      return HtmlUtil.encode(Normalizer.normalize(name, Normalizer.Form.NFC)
+                            .replace(" ", "_")
+                            .replace("&", "")
+                            .replace("(", "")
+                            .replace(")", "")
+                            .replace(",", "")
+                            .replace("+", "_"),
+                            HtmlUtil.ENCODE_TEXT);
```

Normalize then filter/escape

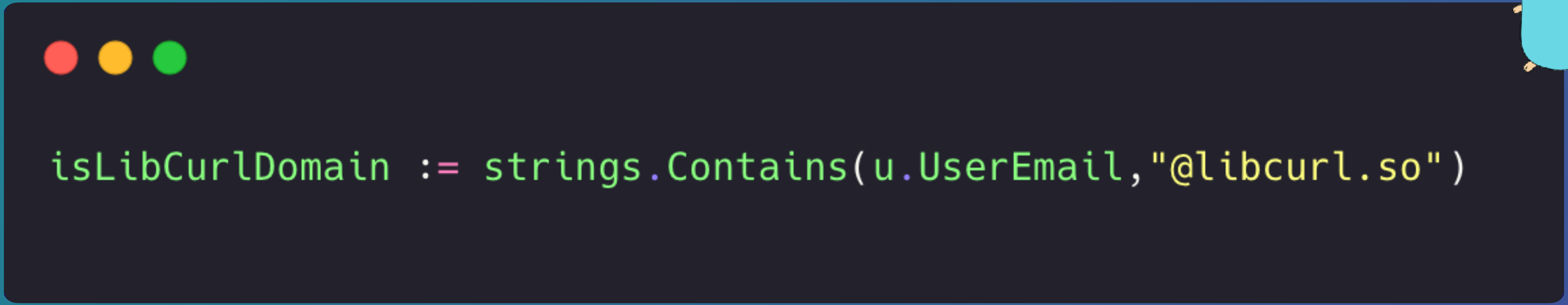
Matching is hard

Ends with, contains, starts with...

- When matching strings without using a Regular Expression, a lot of people get confused on what they are trying to achieve.
- "ends with", "contains", "starts with" may feel like they work similarly for the happy path but they rarely do in reality.

Matching is hard

Ends with, contains, starts with...



```go
isLibCurlDomain := strings.Contains(u.UserEmail,"@libcurl.so")
```

Matching is hard

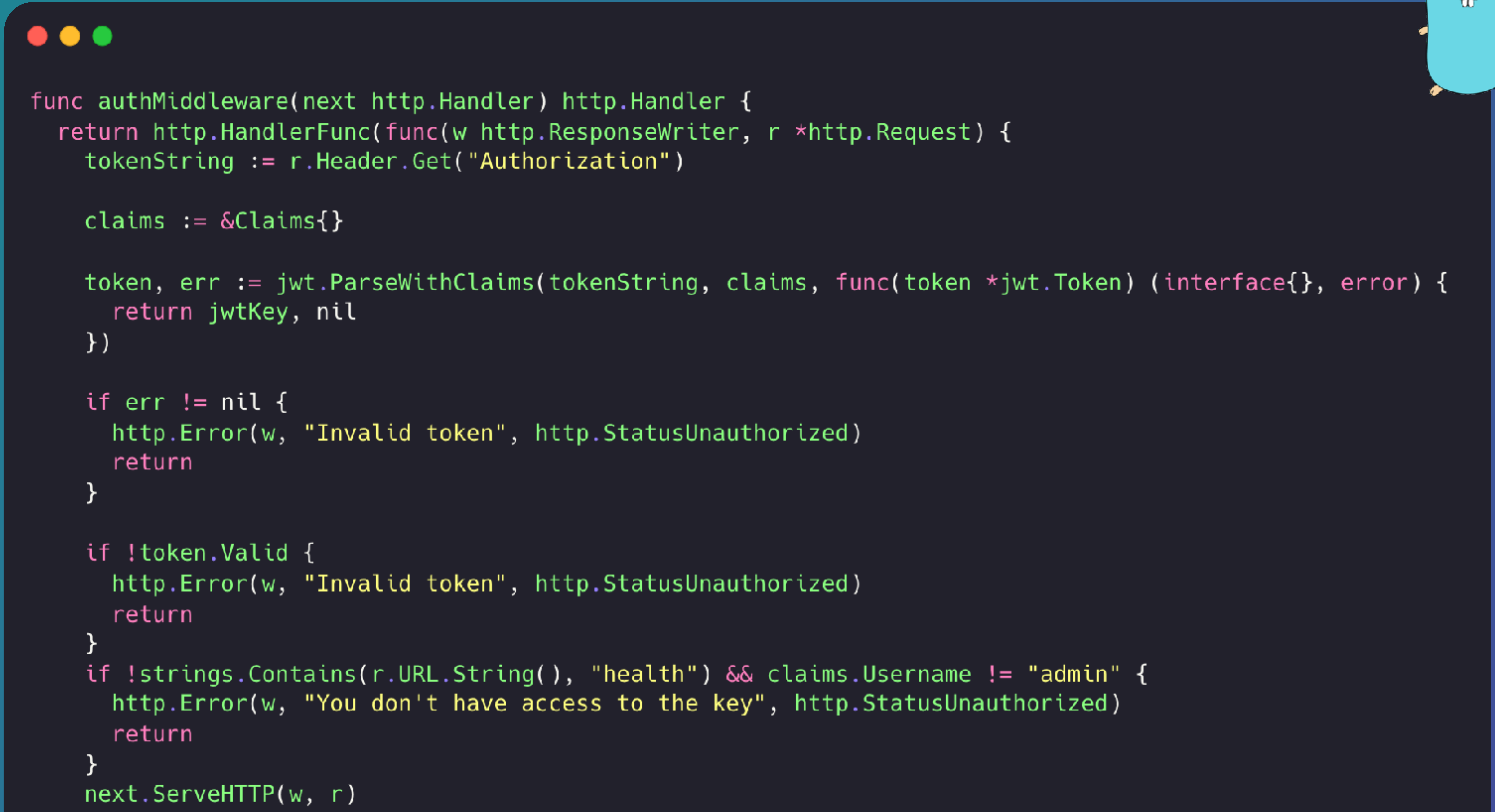
Ends with, contains, starts with...

```
isLibCurlDomain := strings.Contains(u.UserEmail,"@libcurl.so")
```

louis@libcurl.so.pentesterlab.com

Not matching the correct value...

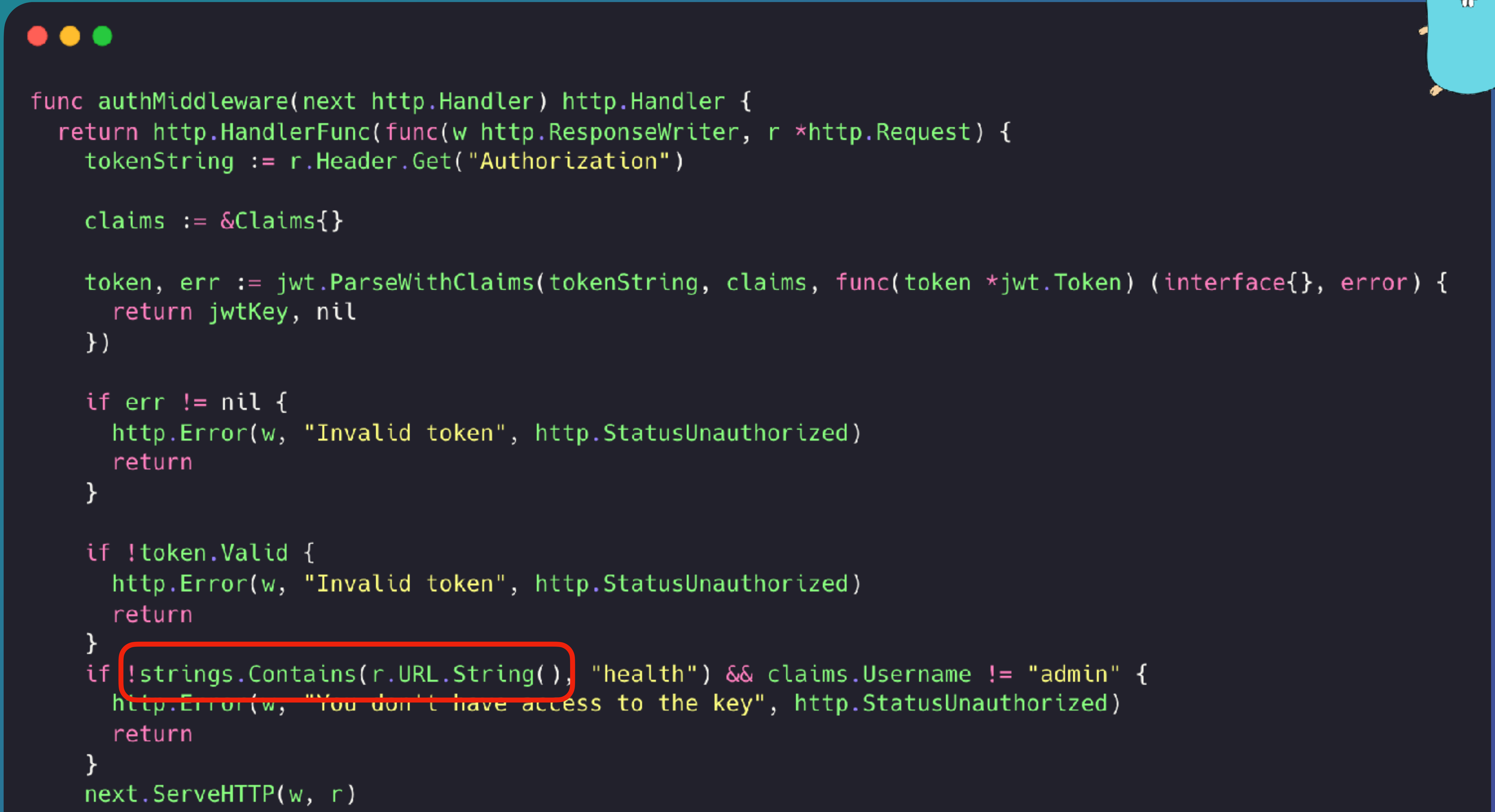
Ends with, contains, starts with...



```
func authMiddleware(next http.Handler) http.Handler {
  return http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
    tokenString := r.Header.Get("Authorization")

    claims := &Claims{}

    token, err := jwt.ParseWithClaims(tokenString, claims, func(token *jwt.Token) (interface{}, error) {
      return jwtKey, nil
    })

    if err != nil {
      http.Error(w, "Invalid token", http.StatusUnauthorized)
      return
    }

    if !token.Valid {
      http.Error(w, "Invalid token", http.StatusUnauthorized)
      return
    }
    if !strings.Contains(r.URL.String(), "health") && claims.Username != "admin" {
      http.Error(w, "You don't have access to the key", http.StatusUnauthorized)
      return
    }
    next.ServeHTTP(w, r)
```

Not matching the correct value...

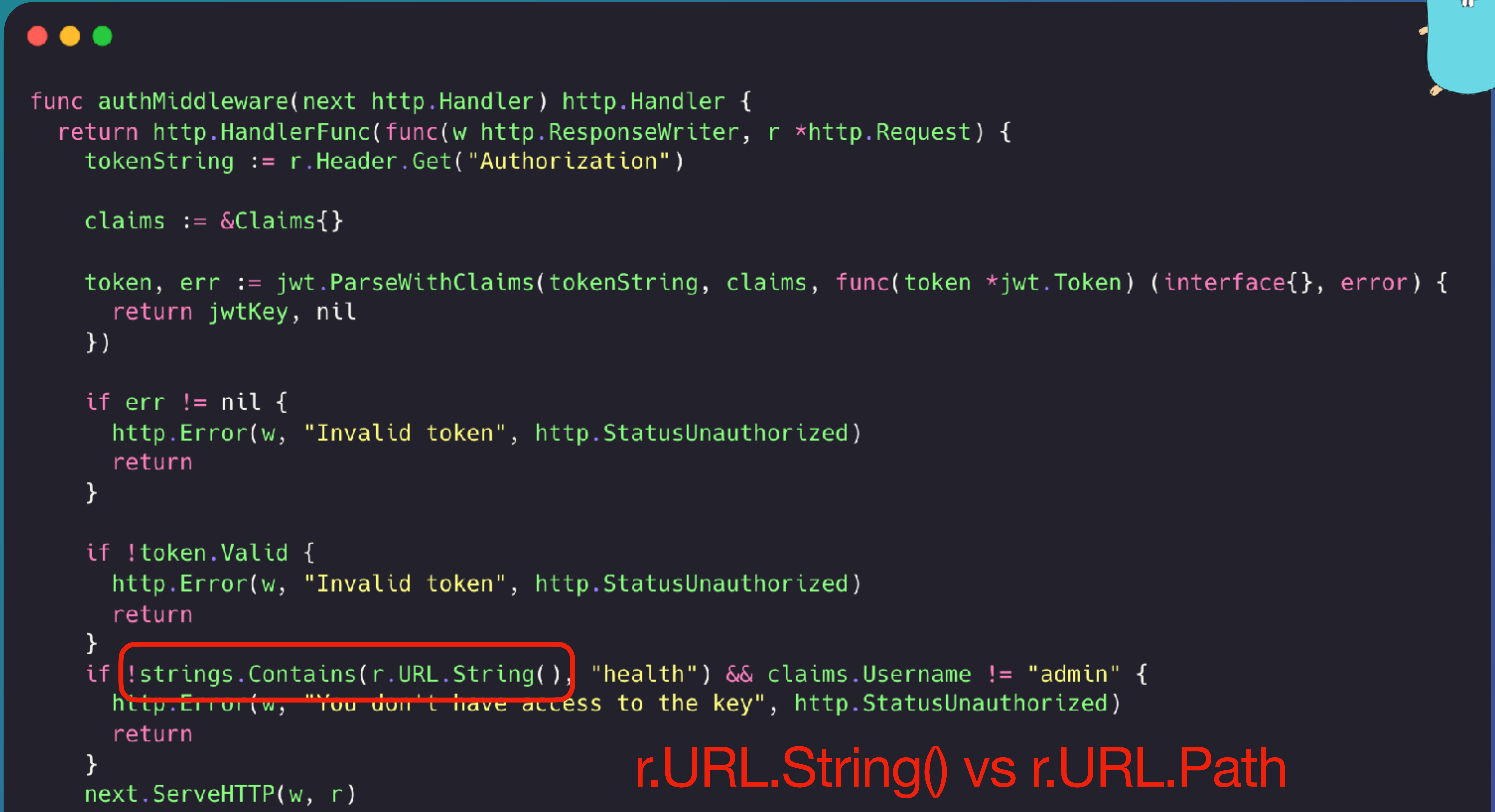
Ends with, contains, starts with...



```go
func authMiddleware(next http.Handler) http.Handler {
  return http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
    tokenString := r.Header.Get("Authorization")

    claims := &Claims{}

    token, err := jwt.ParseWithClaims(tokenString, claims, func(token *jwt.Token) (interface{}, error) {
      return jwtKey, nil
    })

    if err != nil {
      http.Error(w, "Invalid token", http.StatusUnauthorized)
      return
    }

    if !token.Valid {
      http.Error(w, "Invalid token", http.StatusUnauthorized)
      return
    }
    if !strings.Contains(r.URL.String(), "health") && claims.Username != "admin" {
      http.Error(w, "You don't have access to the key", http.StatusUnauthorized)
      return
    }
    next.ServeHTTP(w, r)
```

Not matching the correct value...

Ends with, contains, starts with...



```go
func authMiddleware(next http.Handler) http.Handler {
  return http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
    tokenString := r.Header.Get("Authorization")

    claims := &Claims{}

    token, err := jwt.ParseWithClaims(tokenString, claims, func(token *jwt.Token) (interface{}, error) {
      return jwtKey, nil
    })

    if err != nil {
      http.Error(w, "Invalid token", http.StatusUnauthorized)
      return
    }

    if !token.Valid {
      http.Error(w, "Invalid token", http.StatusUnauthorized)
      return
    }
    if !strings.Contains(r.URL.String(), "health") && claims.Username != "admin" {
      http.Error(w, "You don't have access to the key", http.StatusUnauthorized)
      return
    }
    next.ServeHTTP(w, r)
```

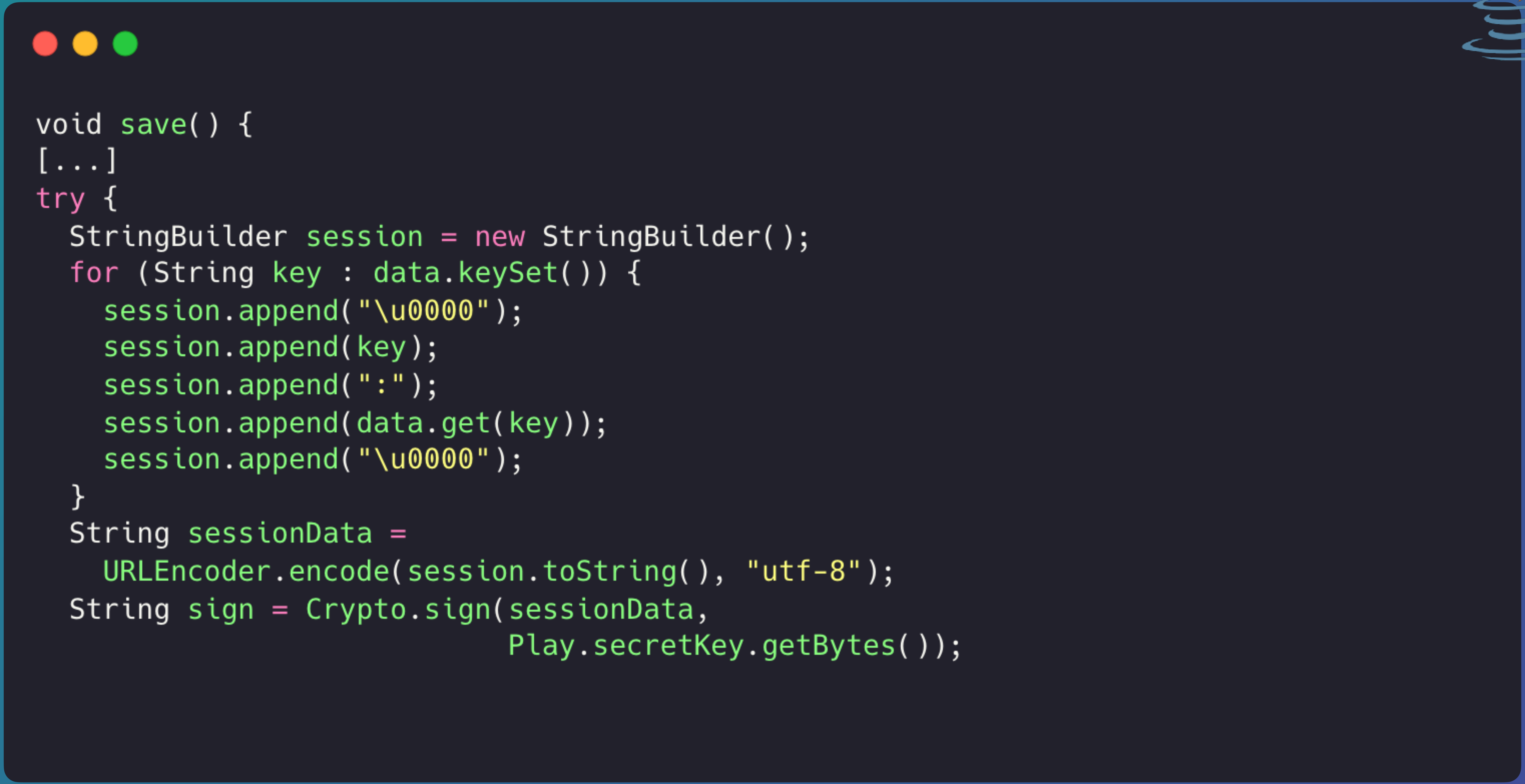
r.URL.String() vs r.URL.Path

Reinventing the wheel!

- Never a good idea (always a bad idea when dealing with crypto)

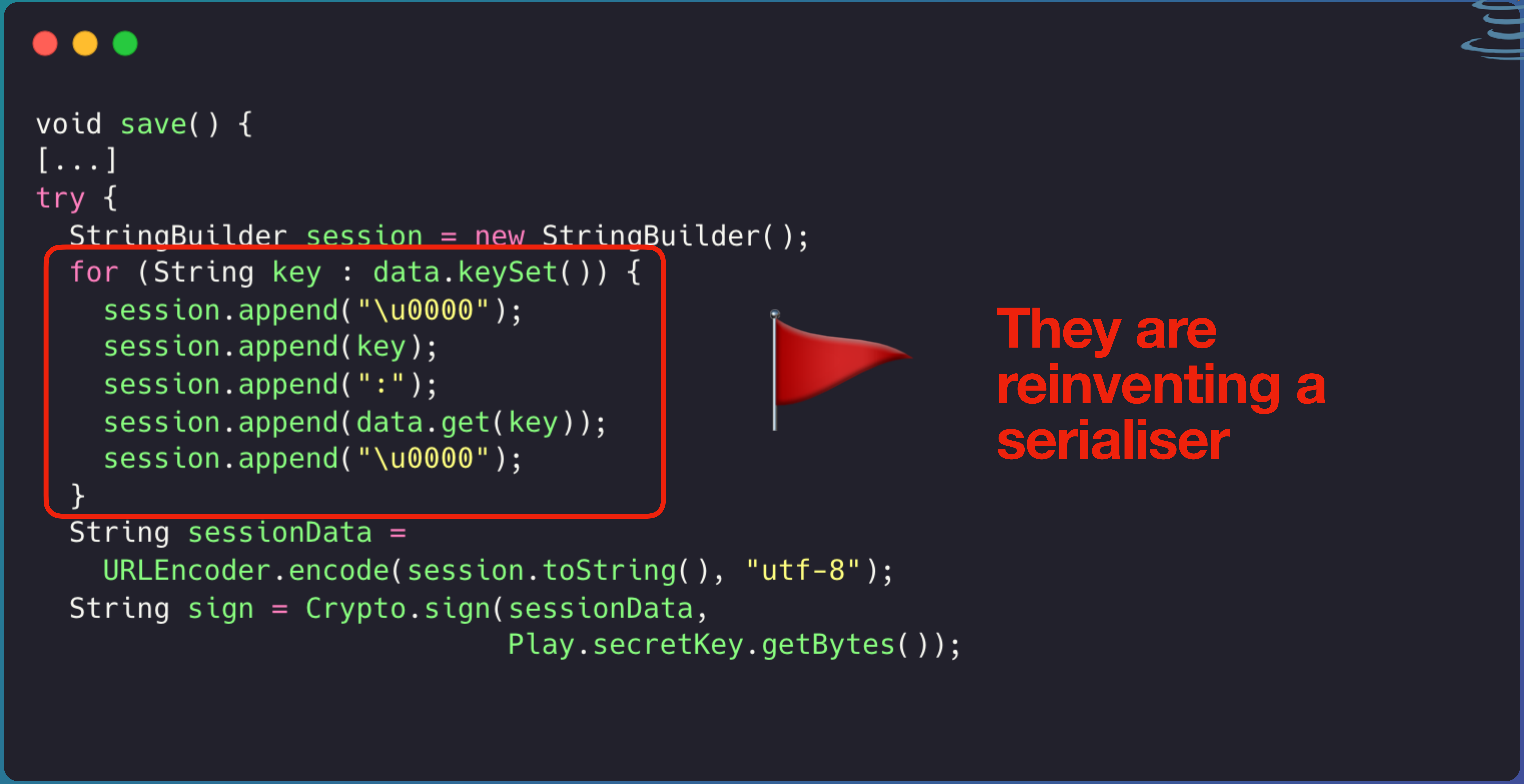
- For most common operations, programming languages provide built-in functions or methods, such as:
    - String manipulations: uppercase, lowercase, split, cut
    - File manipulations: getting the file extension from a filename, extracting the filename from a path, etc.
- When developers write their own versions, they’re likely to overlook odd edge cases that the built-in functions already handle
- Compare the built-in source code with the code written by the developers!

Play Session Injection



```java
void save() {
[...]
try {
  StringBuilder session = new StringBuilder();
  for (String key : data.keySet()) {
    session.append("\u0000");
    session.append(key);
    session.append(":");
    session.append(data.get(key));
    session.append("\u0000");
  }
  String sessionData =
    URLEncoder.encode(session.toString(), "utf-8");
  String sign = Crypto.sign(sessionData,
                            Play.secretKey.getBytes());
```

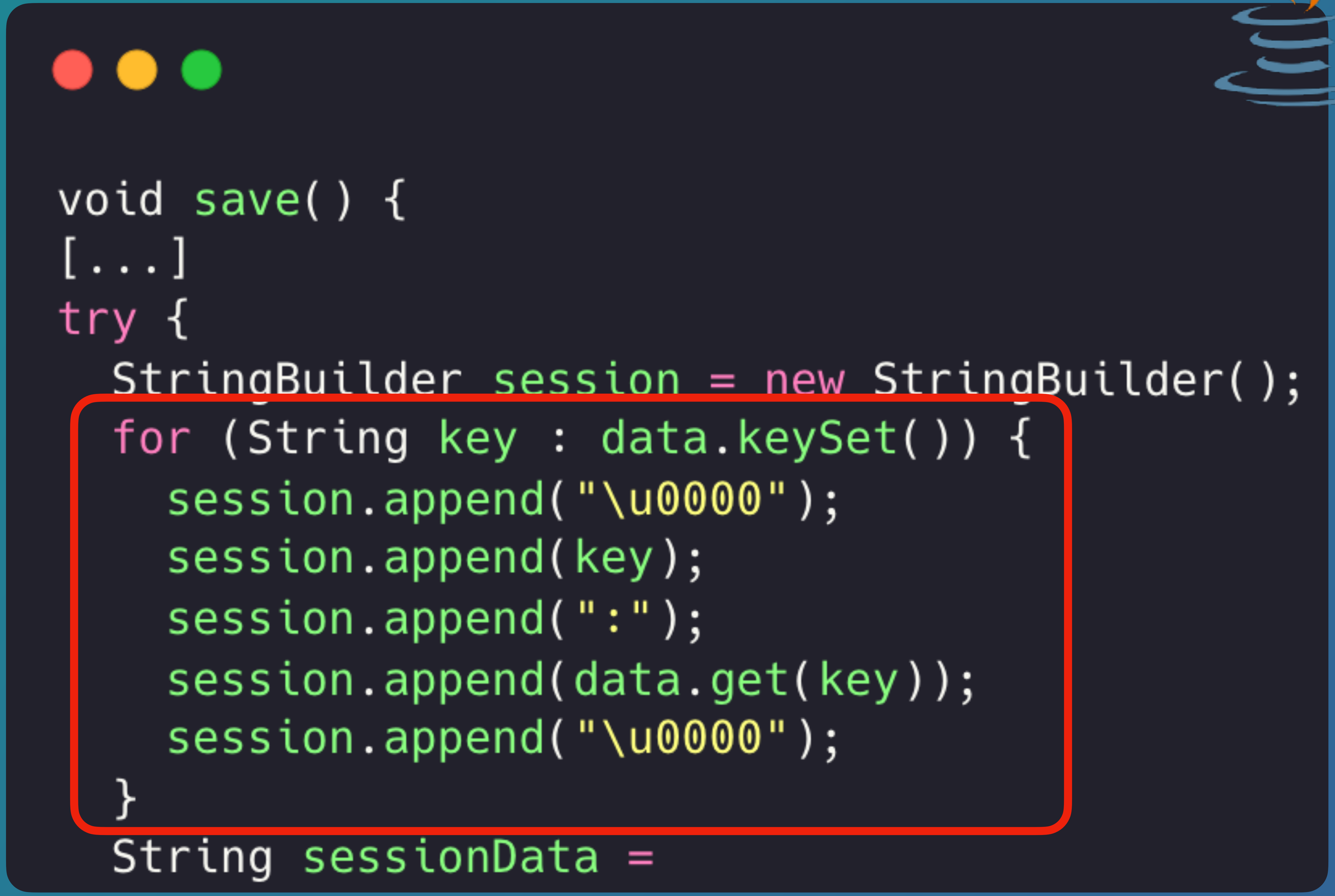
Play Session Injection



```
void save() {
[...]
try {
  StringBuilder session = new StringBuilder();
  for (String key : data.keySet()) {
    session.append("\u0000");
    session.append(key);
    session.append(":");
    session.append(data.get(key));
    session.append("\u0000");
  }
  String sessionData =
    URLEncoder.encode(session.toString(), "utf-8");
  String sign = Crypto.sign(sessionData,
                            Play.secretKey.getBytes());
```

🚩 They are reinventing a serialiser

Play Session Injection

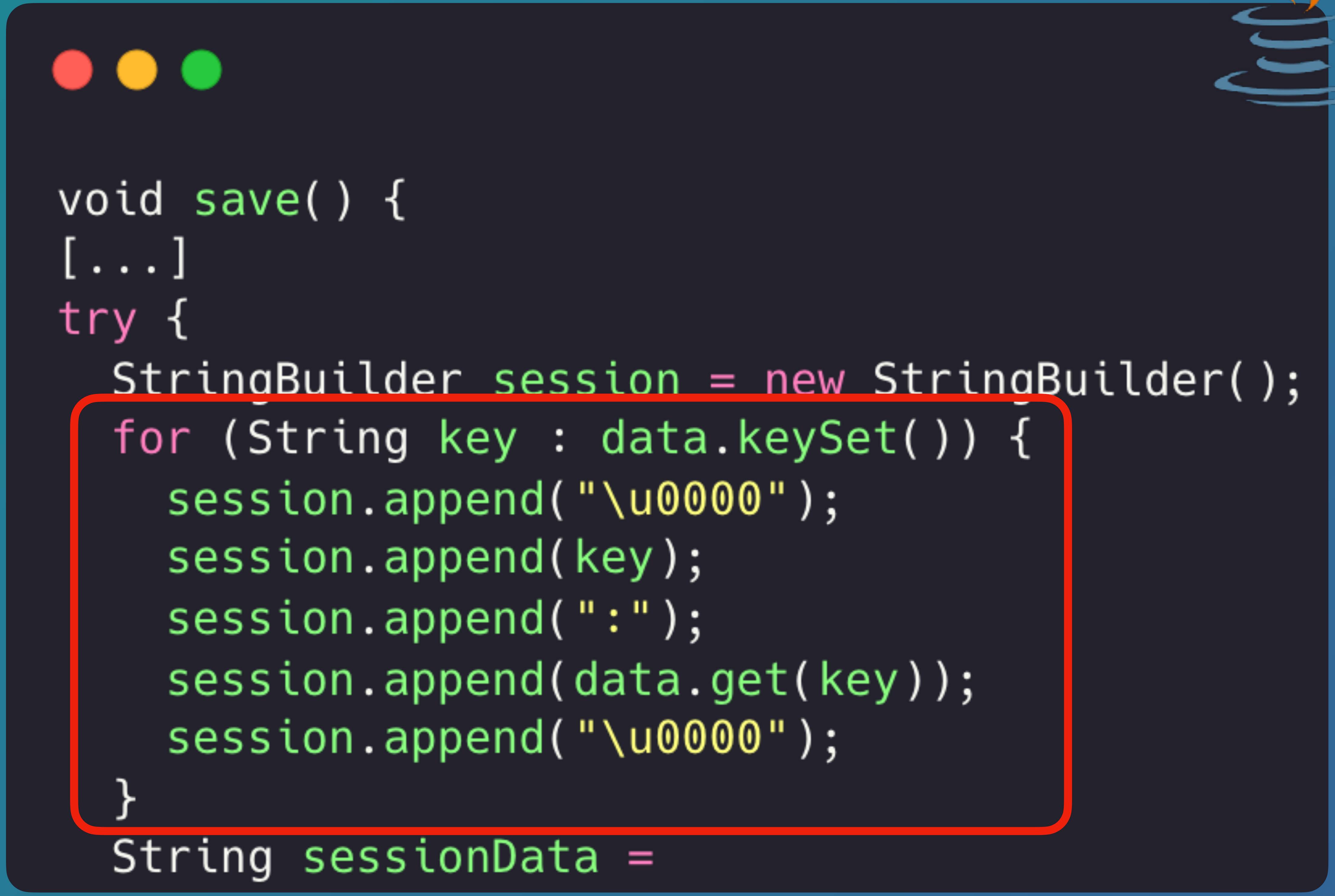


```
void save() {
[...]
try {
  StringBuilder session = new StringBuilder();
  for (String key : data.keySet()) {
    session.append("\u0000");
    session.append(key);
    session.append(":");
    session.append(data.get(key));
    session.append("\u0000");
  }
  String sessionData =
```

Session: {"key1": "value1", "key2": "value2"}

becomes "\x00key1:value1\x00\x00key2:value2\x00"

Play Session Injection

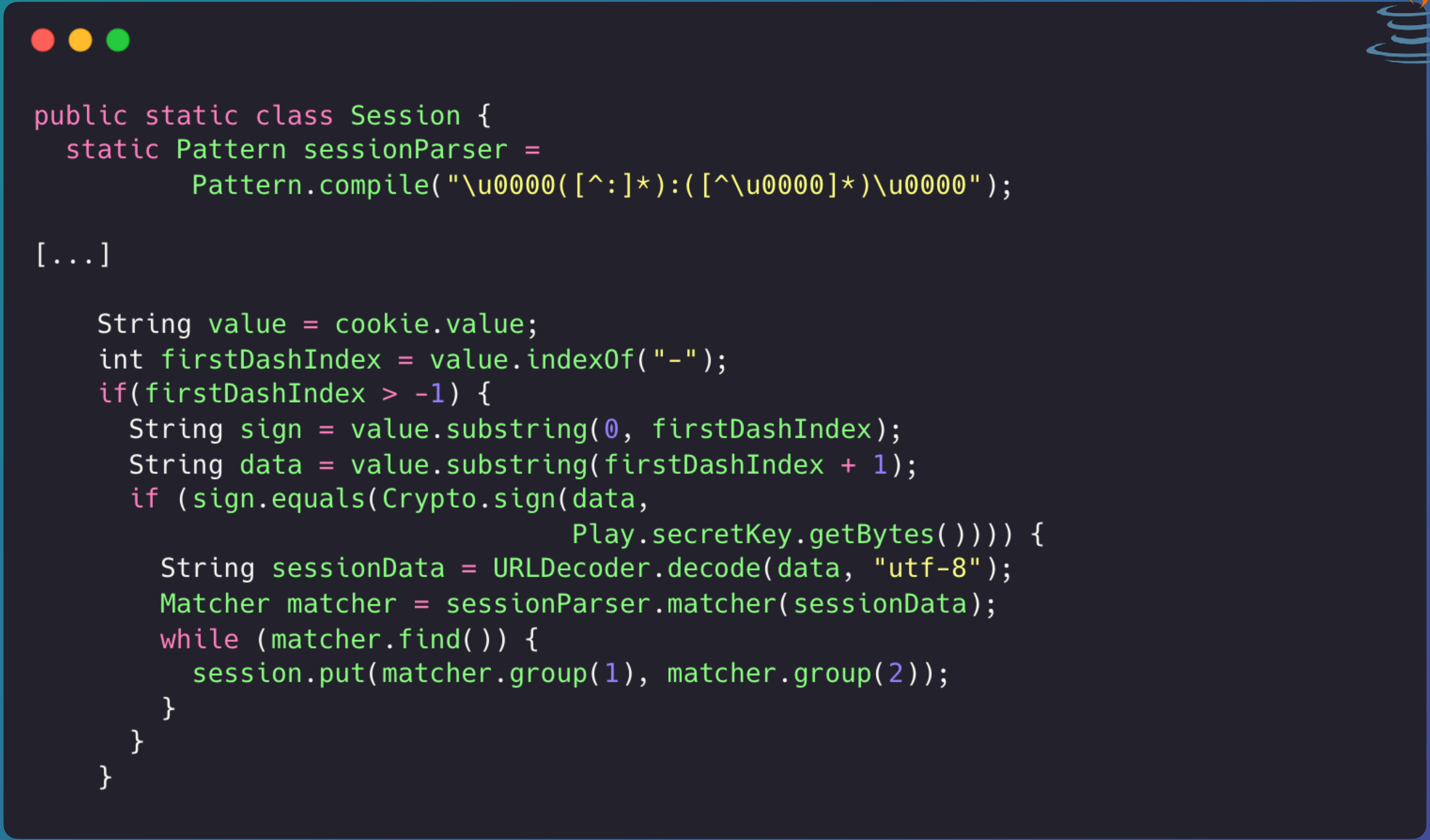


```
void save() {
[...]
try {
  StringBuilder session = new StringBuilder();
  for (String key : data.keySet()) {
    session.append("\u0000");
    session.append(key);
    session.append(":");
    session.append(data.get(key));
    session.append("\u0000");
  }
  String sessionData =
```

Session: {"username": "louis", "email": "louis@pentesterlab.com"}

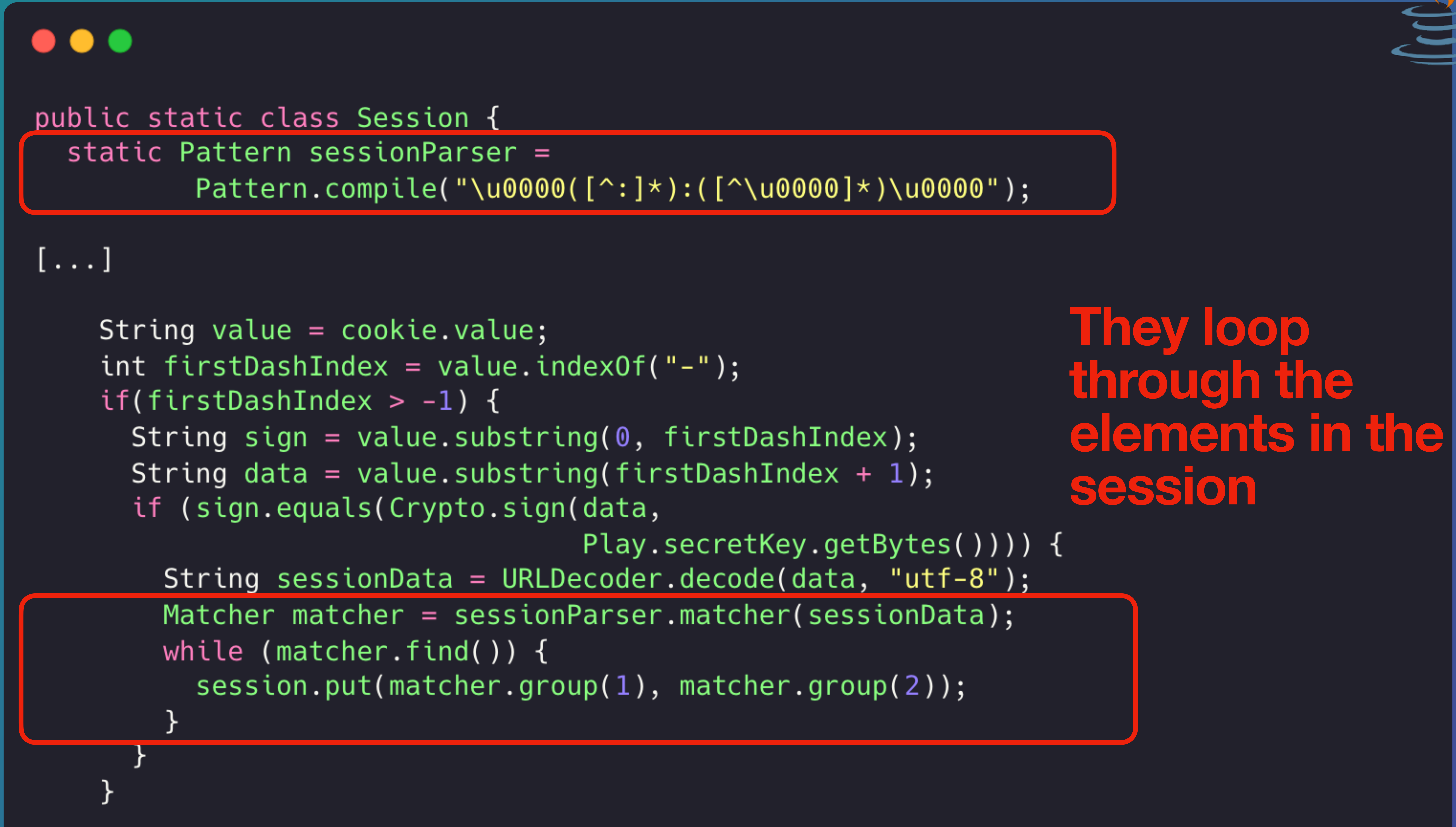
becomes "\x00username:louis\x00\x00email:louis@pentesterlab.com\x00"

Play Session Injection



```java
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                  Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

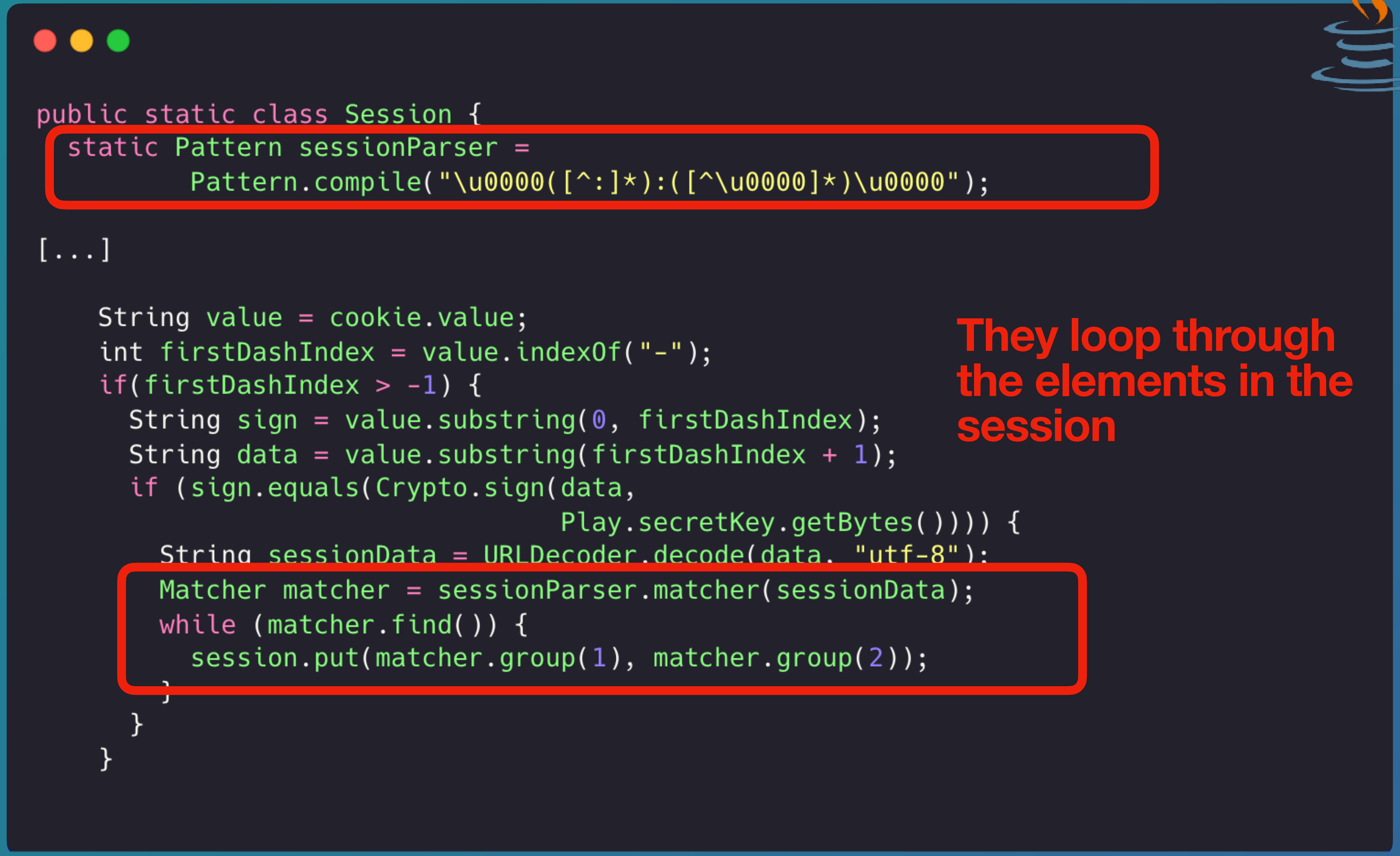
Play Session Injection



```
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                      Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

They loop through the elements in the session

Play Session Injection



```
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                  Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

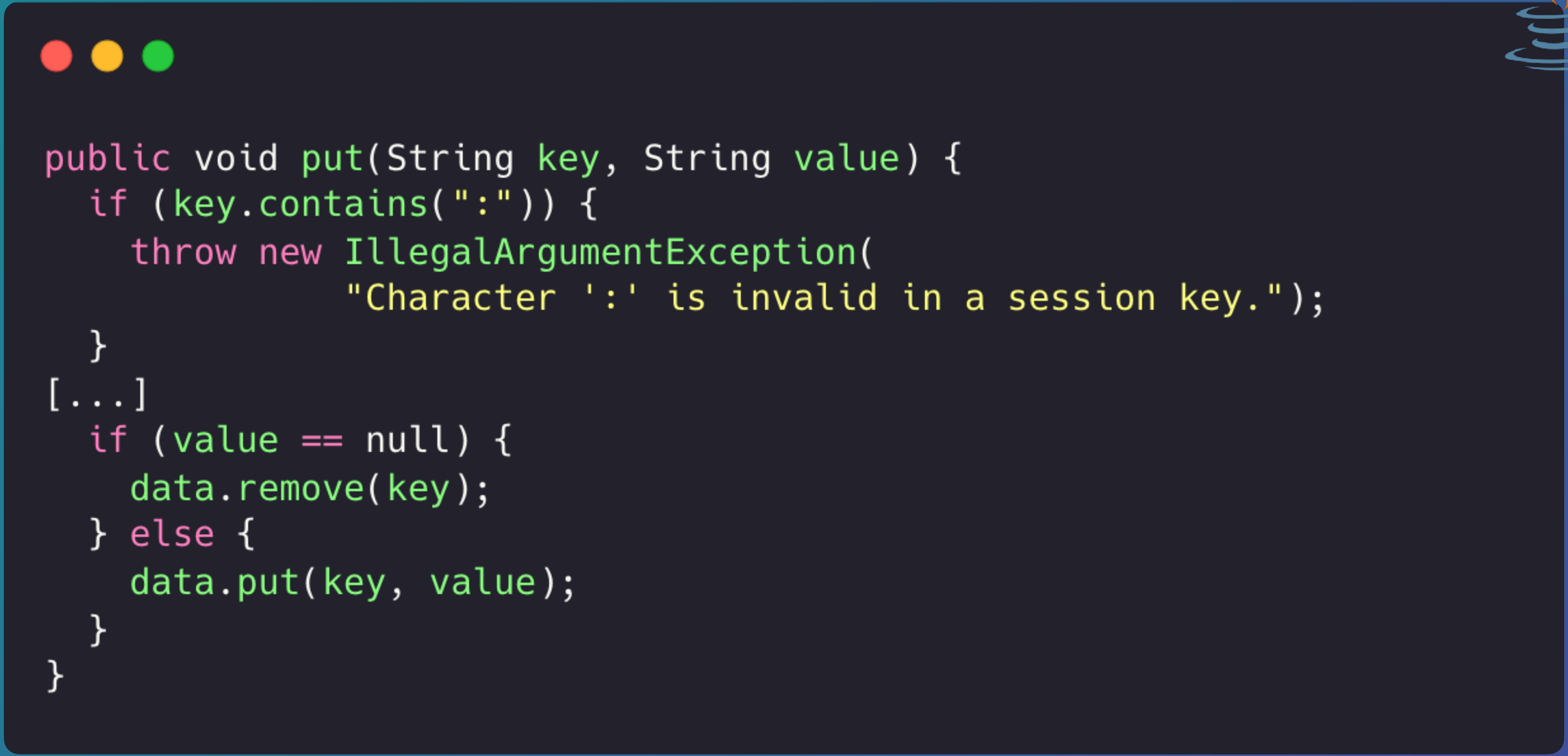
They loop through the elements in the session

"\x00key1:value1\x00\x00key2:value2\x00" becomes:

"\x00key1:value1\x00" => session.put("key1" , "value1")

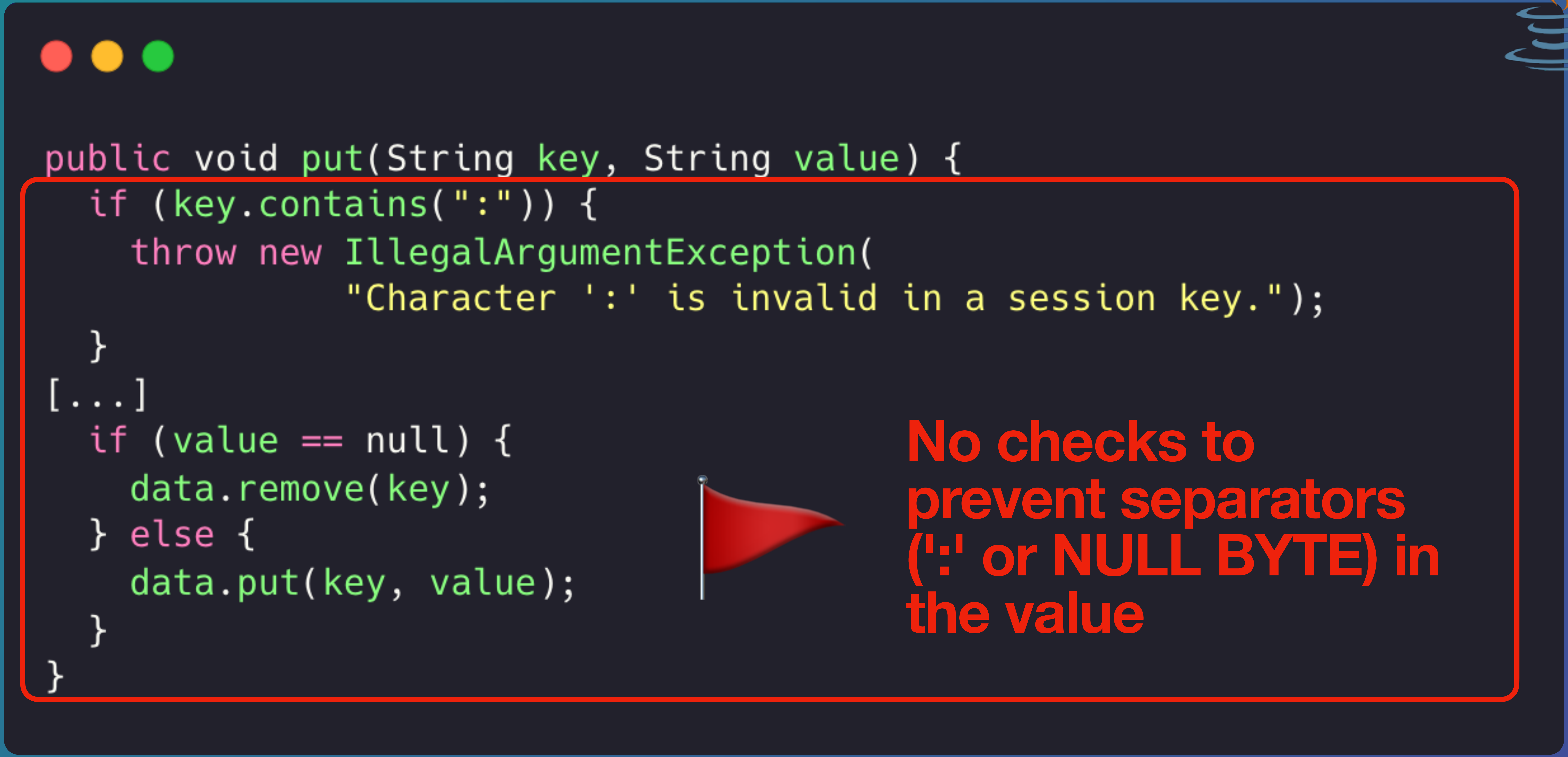
"\x00key2:value2\x00" => session.put("key2" , "value2")

Play Session Injection



```
public void put(String key, String value) {
  if (key.contains(":")) {
    throw new IllegalArgumentException(
              "Character ':' is invalid in a session key.");
  }
[...]
  if (value == null) {
    data.remove(key);
  } else {
    data.put(key, value);
  }
}
```

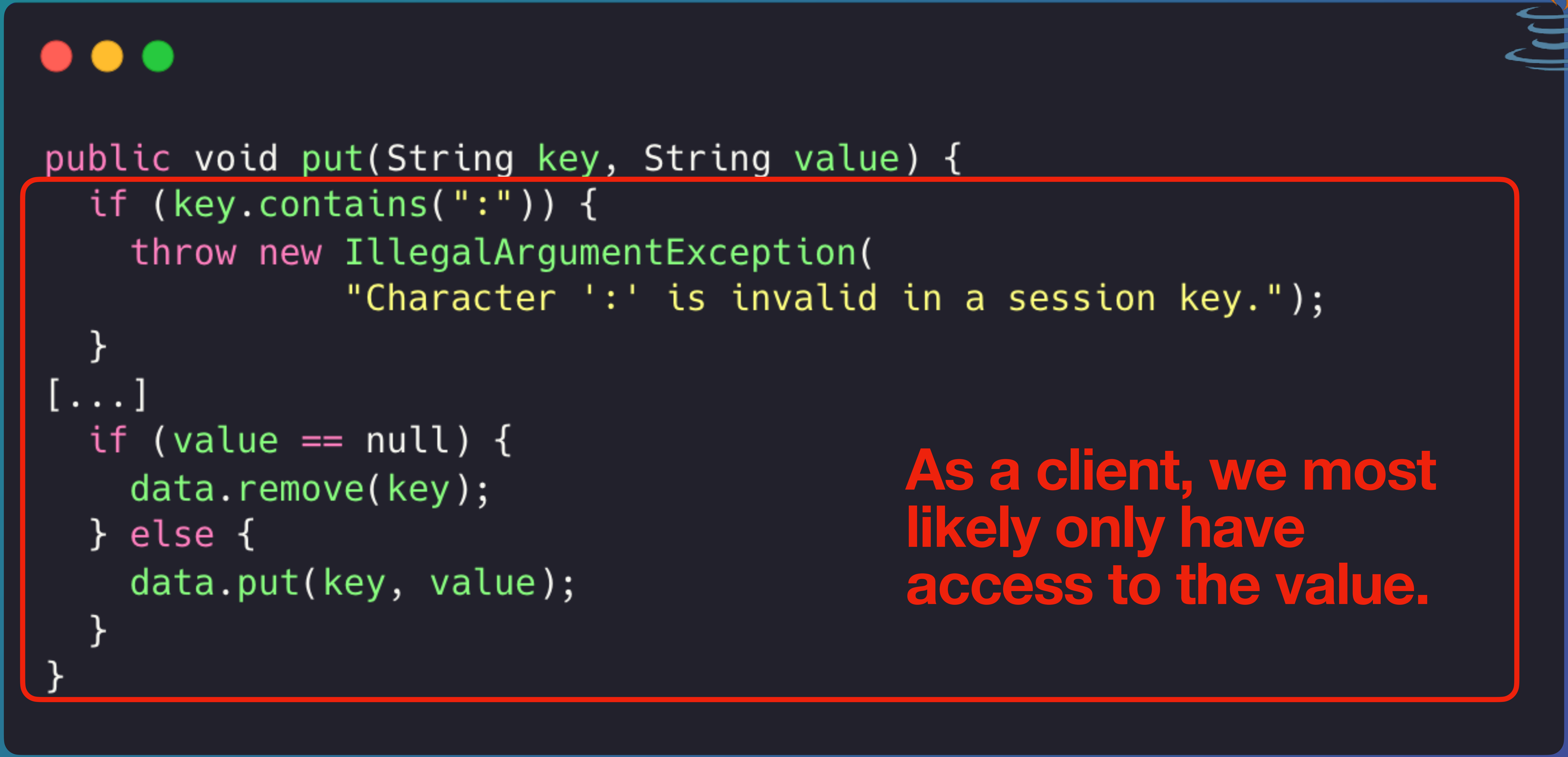
Play Session Injection



```
public void put(String key, String value) {
  if (key.contains(":")) {
    throw new IllegalArgumentException(
              "Character ':' is invalid in a session key.");
  }
[...]
  if (value == null) {
    data.remove(key);
  } else {
    data.put(key, value);
  }
}
```

No checks to prevent separators (':' or NULL BYTE) in the value

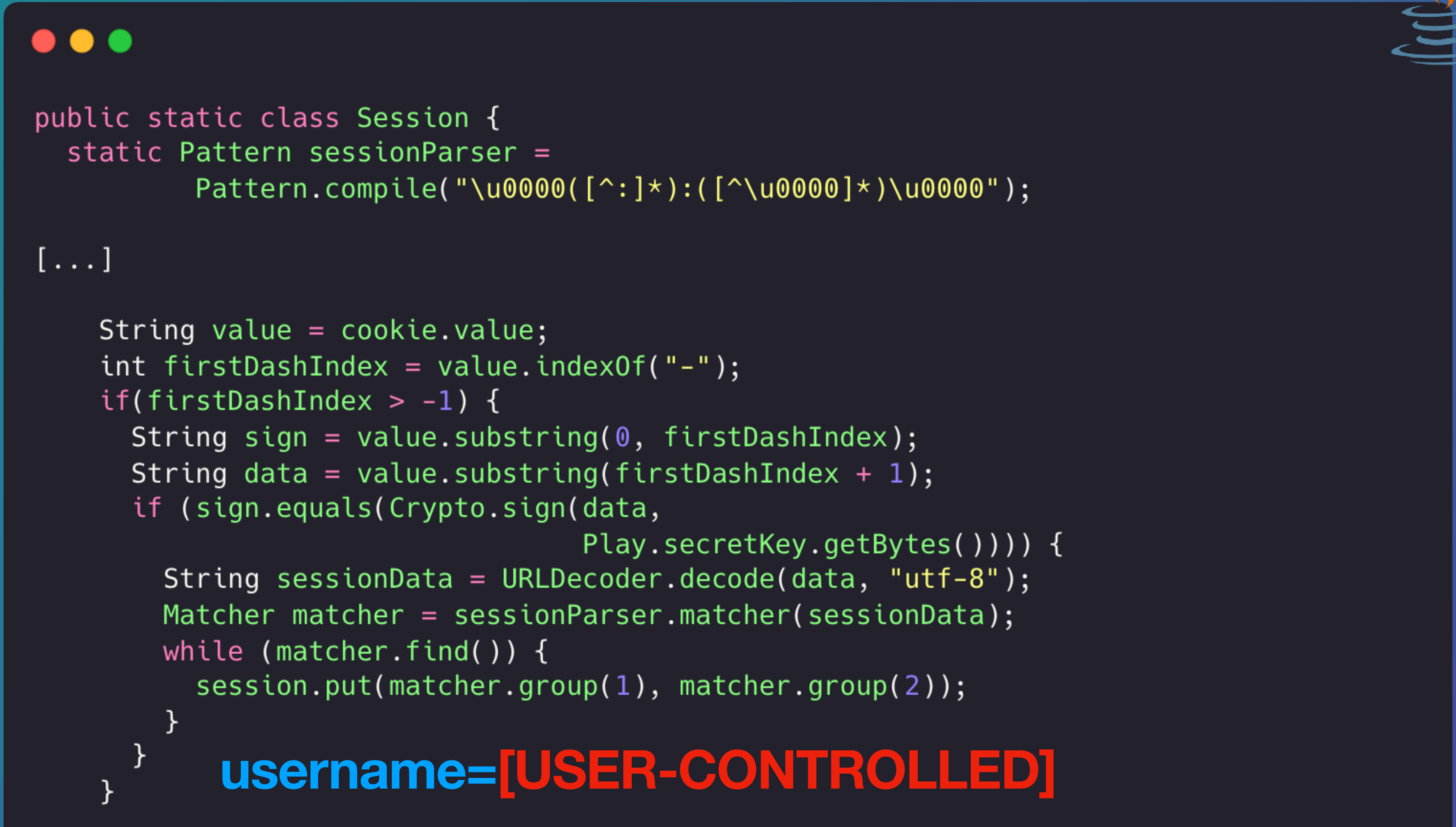
Play Session Injection



```
public void put(String key, String value) {
  if (key.contains(":")) {
    throw new IllegalArgumentException(
              "Character ':' is invalid in a session key.");
  }
[...]
  if (value == null) {
    data.remove(key);
  } else {
    data.put(key, value);
  }
}
```

As a client, we most likely only have access to the value.

Play Session Injection



```java
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                            Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

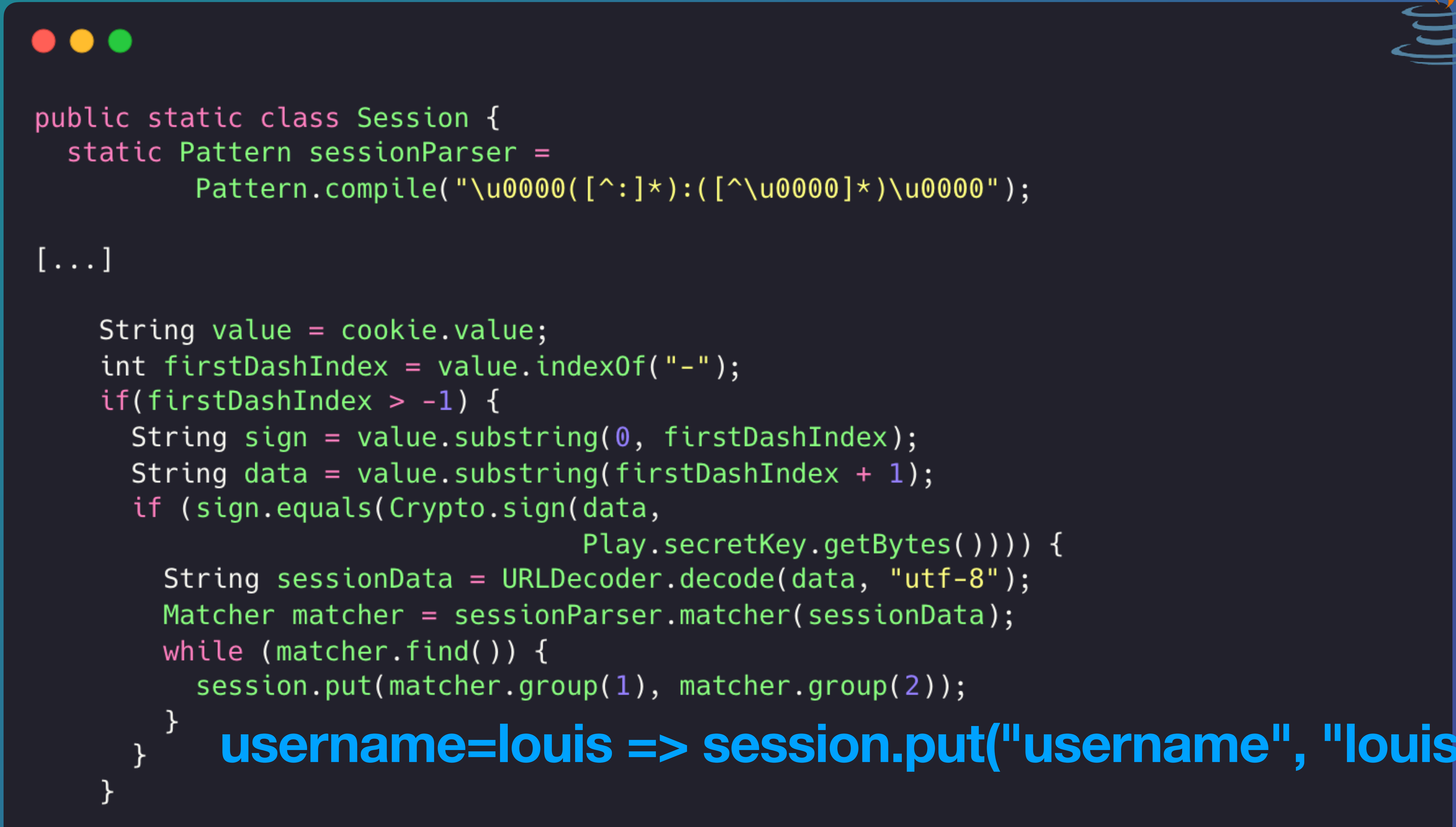
username=[USER-CONTROLLED]

Play Session Injection

```
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                  Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

username=louis

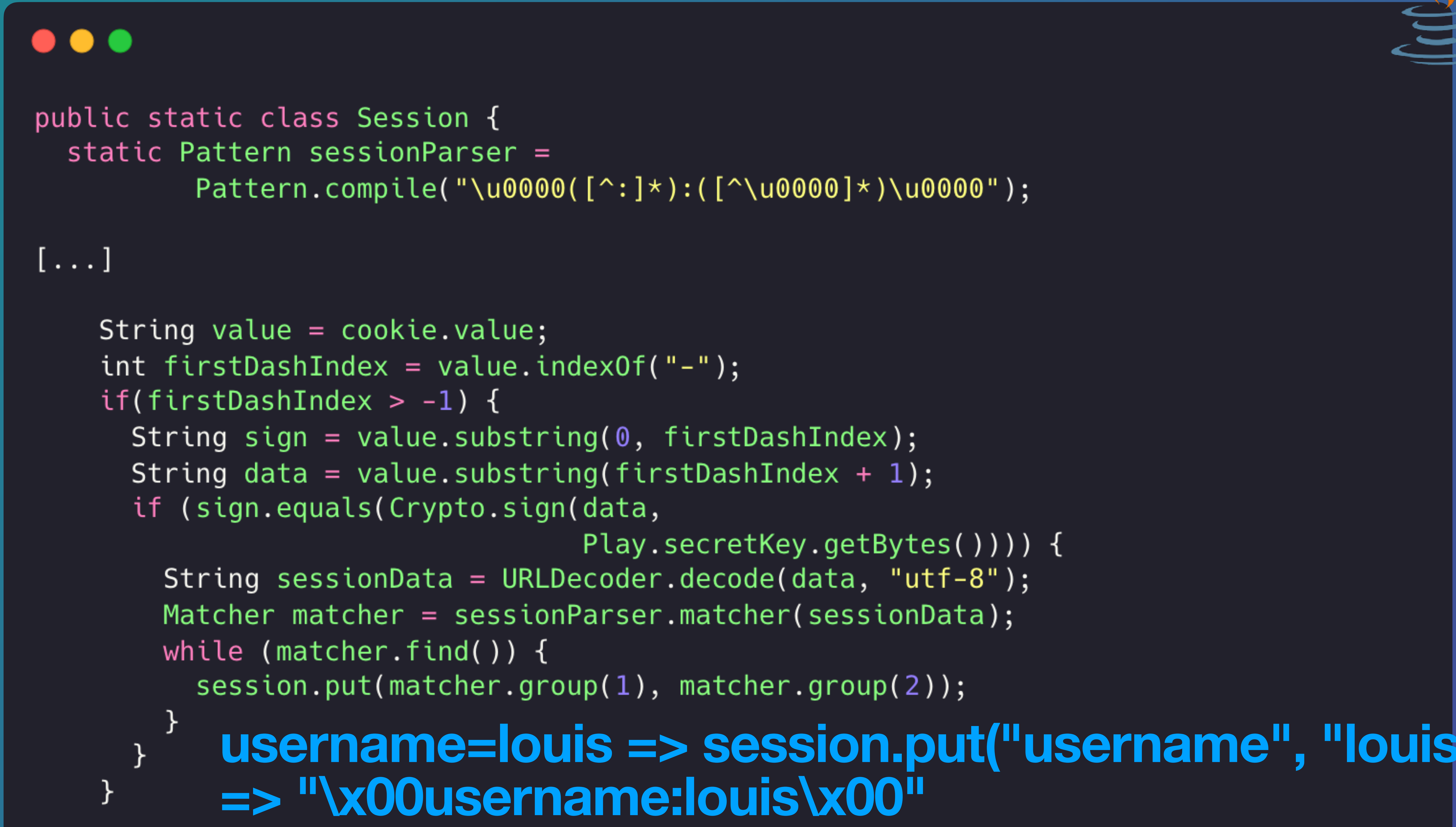
Play Session Injection



```
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                  Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

username=louis => session.put("username", "louis")

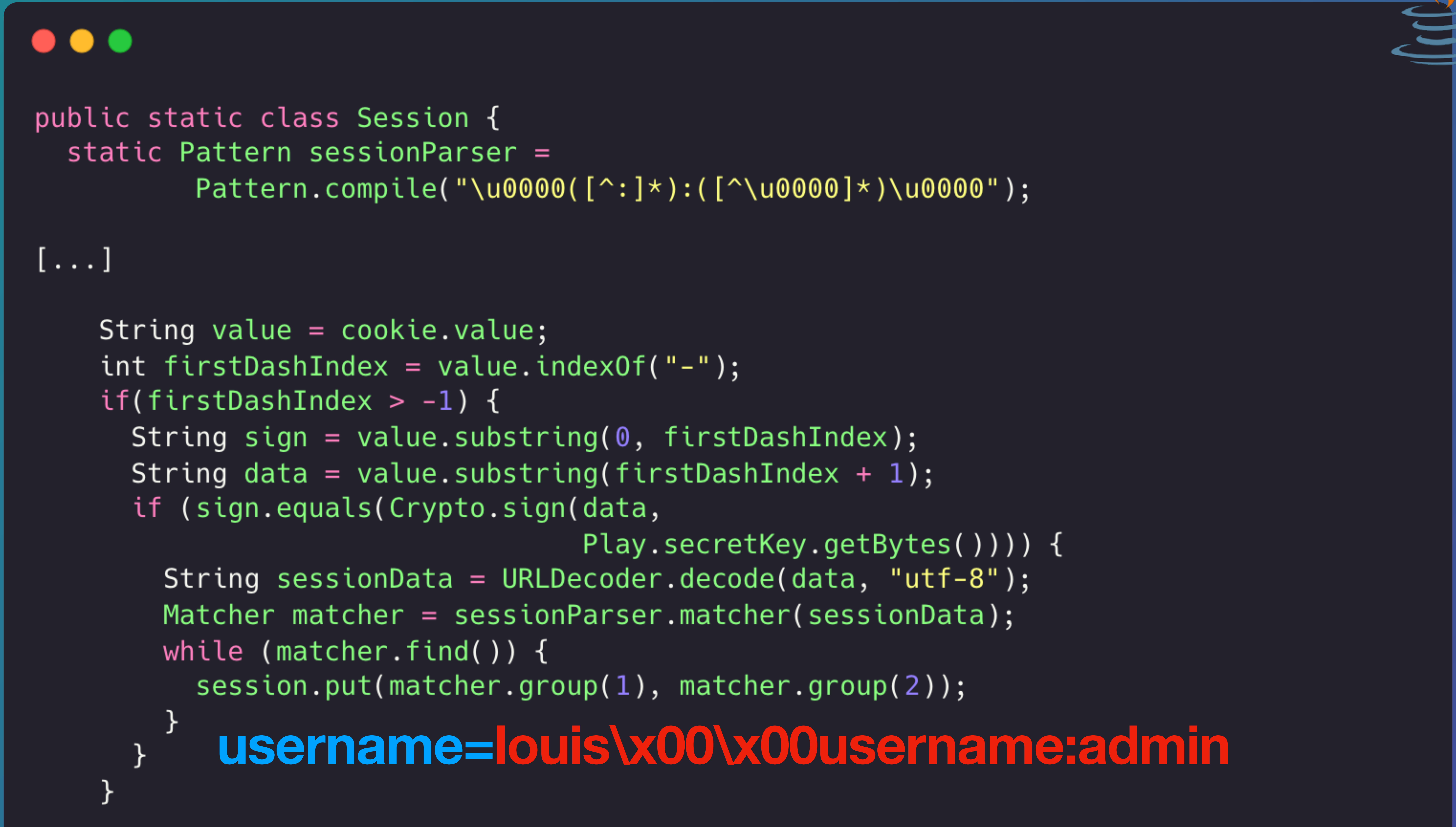
Play Session Injection



```
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                   Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

username=louis => session.put("username", "louis")
=> "\x00username:louis\x00"

Play Session Injection



```
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                  Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

username=louis\x00\x00username:admin

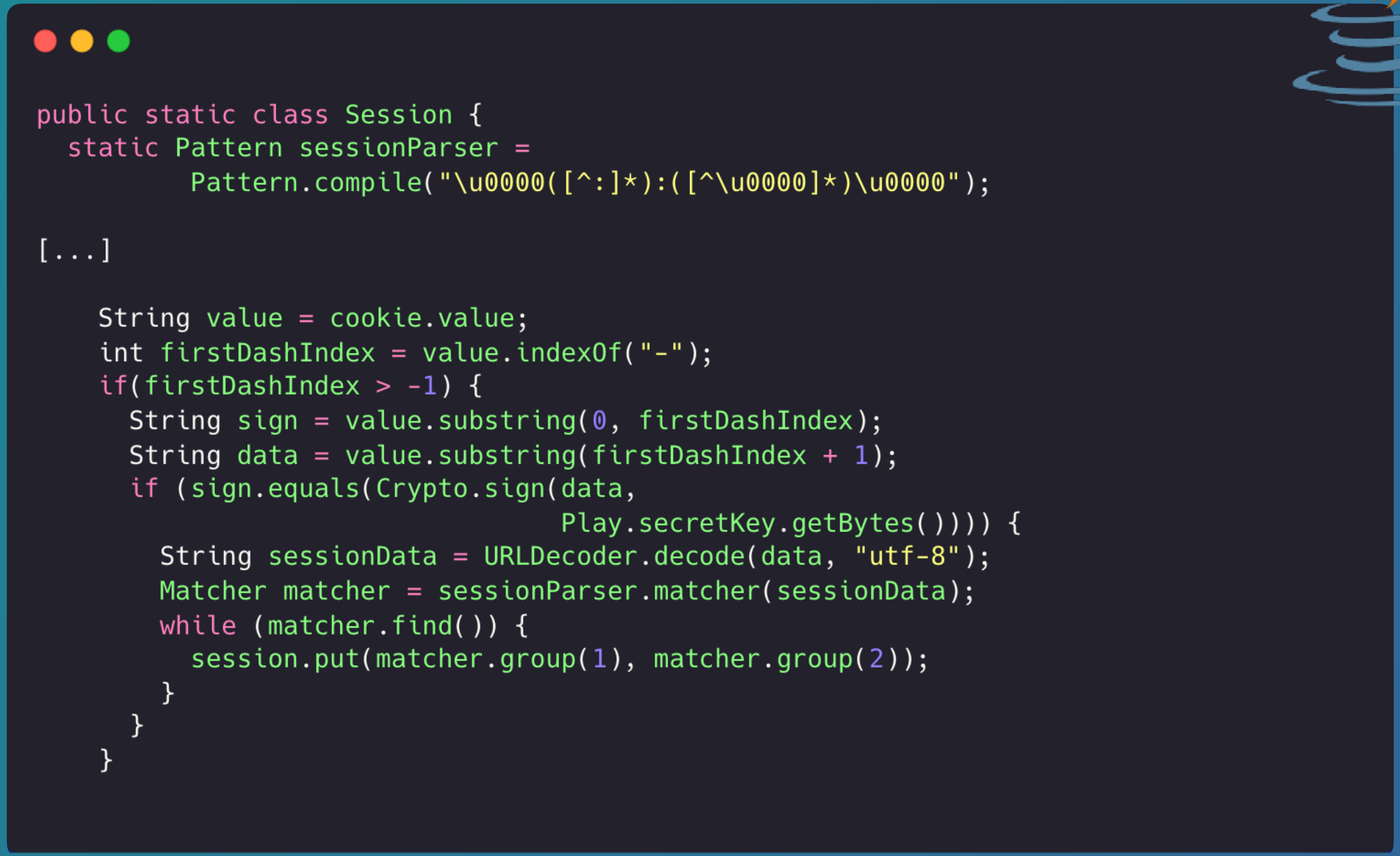
Play Session Injection

```java
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

username=louis\x00\x00username:admin

=> session.put("username", "louis\x00\x00username:admin")

Play Session Injection



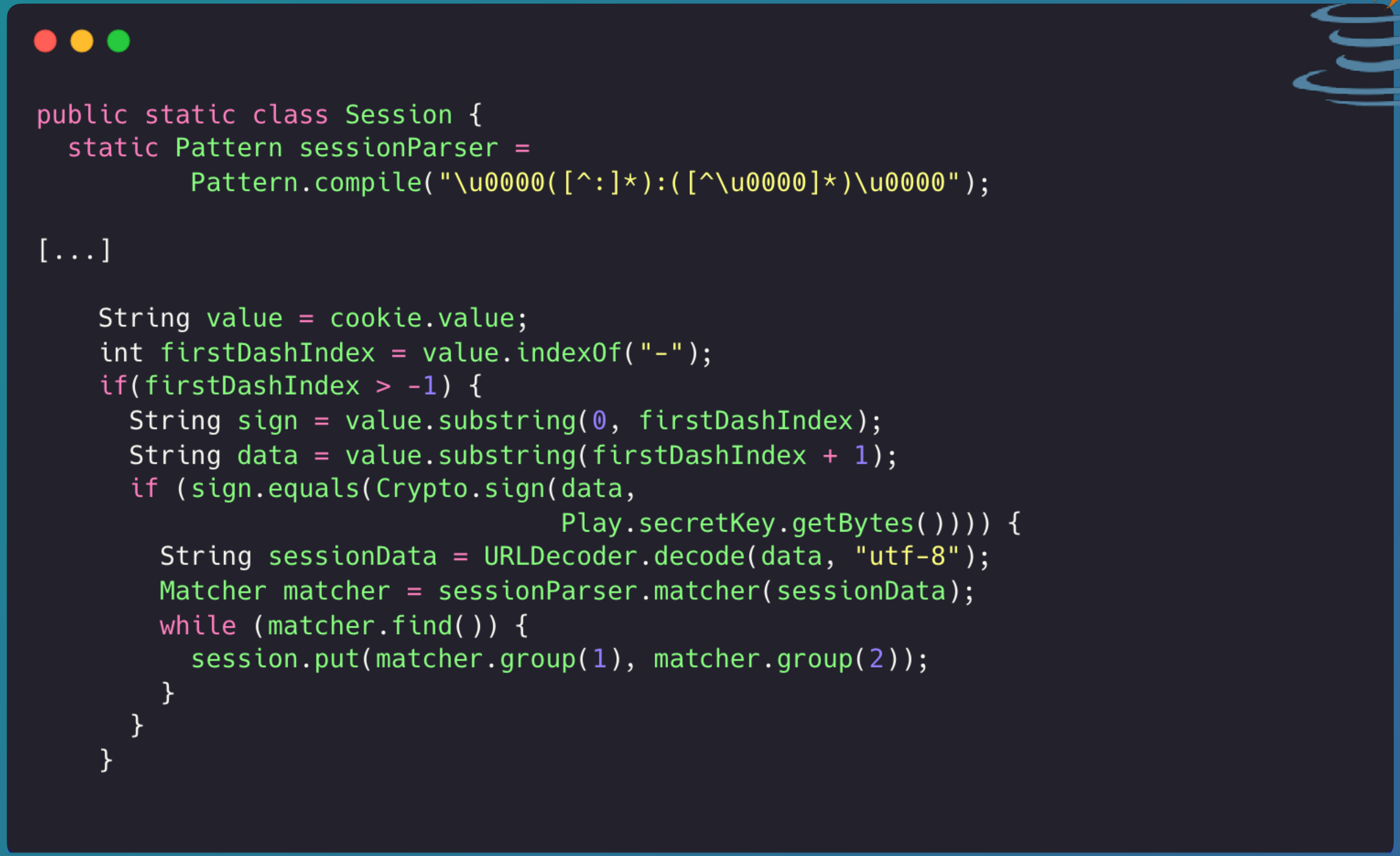
```
public static class Session {
  static Pattern sessionParser =
         Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                  Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

username=louis\x00\x00username:admin

=> session.put("username", "louis\x00\x00username:admin")

=> \x00username:louis\x00\x00username:admin\x00

Play Session Injection



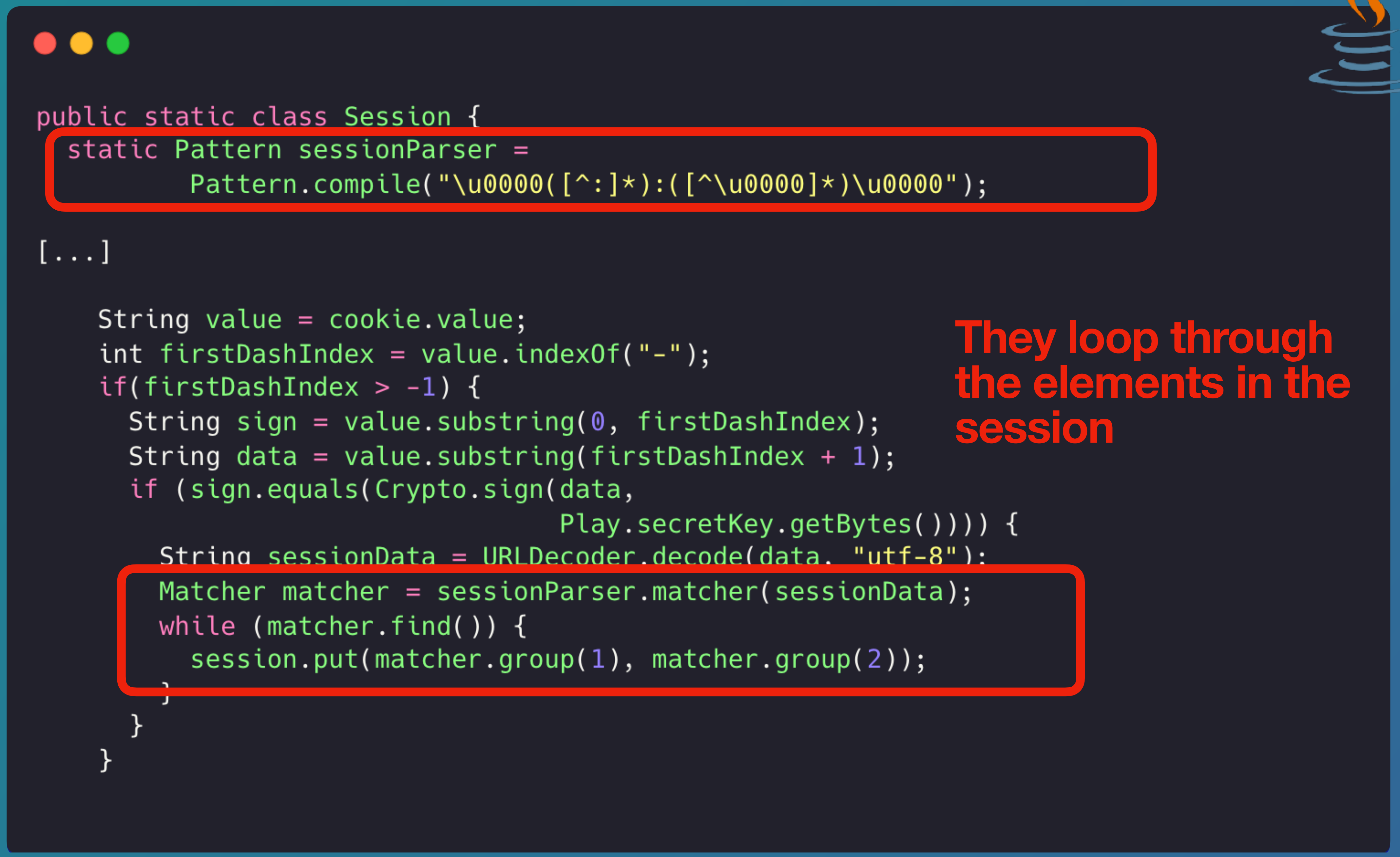
```java
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                  Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

username=louis\x00\x00username:admin

=> session.put("username", "louis\x00\x00username:admin")

=> \x00username:louis\x00\x00username:admin\x00

Play Session Injection



```
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                            Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

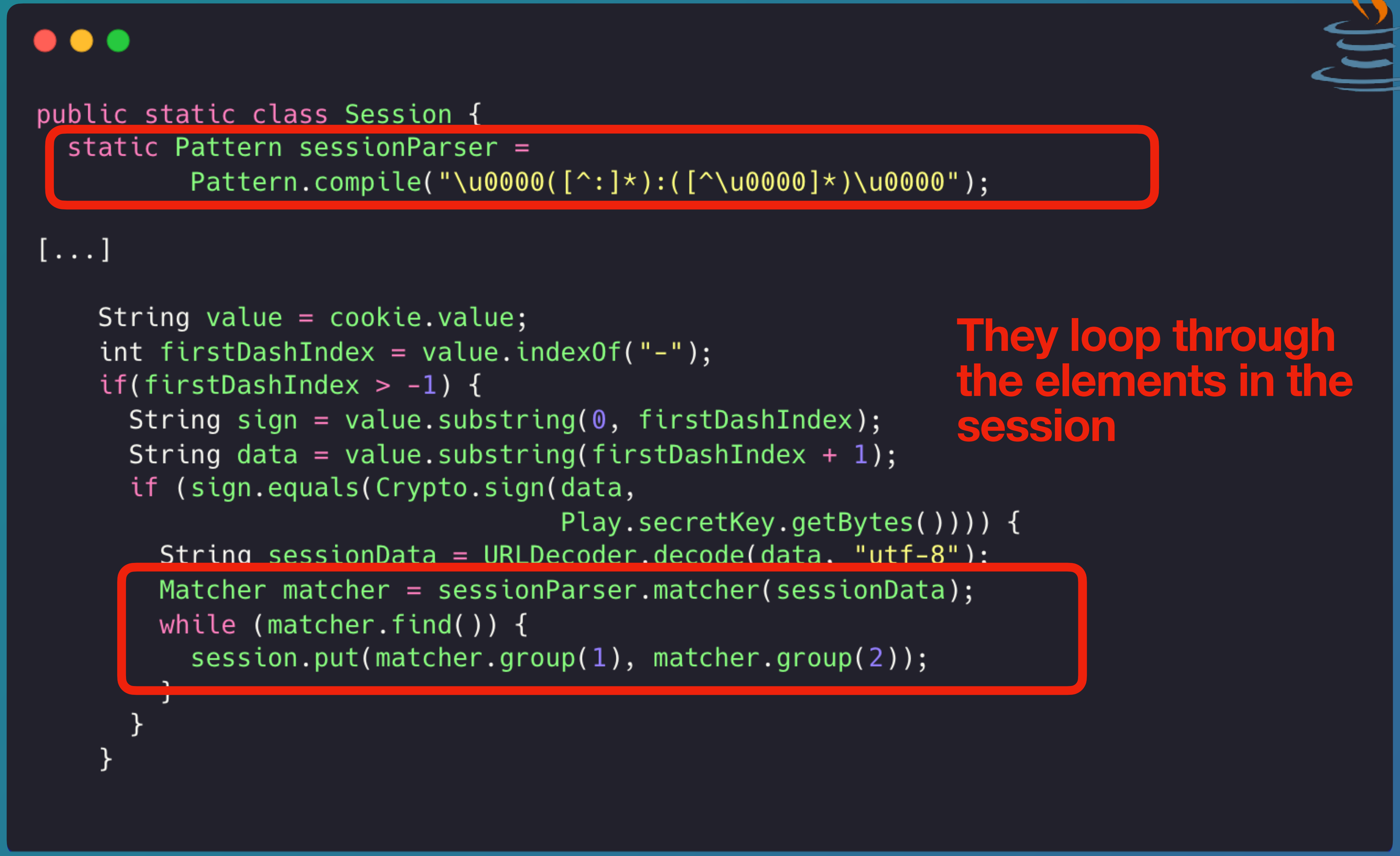
They loop through the elements in the session

\x00username:louis\x00\x00username:admin\x00 becomes

"\x00username:louis\x00" => session.put("username" , "louis")

"\x00username:admin\x00" => session.put("username" , "admin")

Play Session Injection



```java
public static class Session {
  static Pattern sessionParser =
          Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");

[...]

    String value = cookie.value;
    int firstDashIndex = value.indexOf("-");
    if(firstDashIndex > -1) {
      String sign = value.substring(0, firstDashIndex);
      String data = value.substring(firstDashIndex + 1);
      if (sign.equals(Crypto.sign(data,
                                  Play.secretKey.getBytes()))) {
        String sessionData = URLDecoder.decode(data, "utf-8");
        Matcher matcher = sessionParser.matcher(sessionData);
        while (matcher.find()) {
          session.put(matcher.group(1), matcher.group(2));
        }
      }
    }
```

They loop through the elements in the session

\x00username:louis\x00\x00username:admin\x00 becomes

"\x00username:louis\x00" => session.put("username" , "louis")

"\x00username:admin\x00" => session.put("username" , "admin") (OVERWRITE)

CVE ANALYSIS

Why?

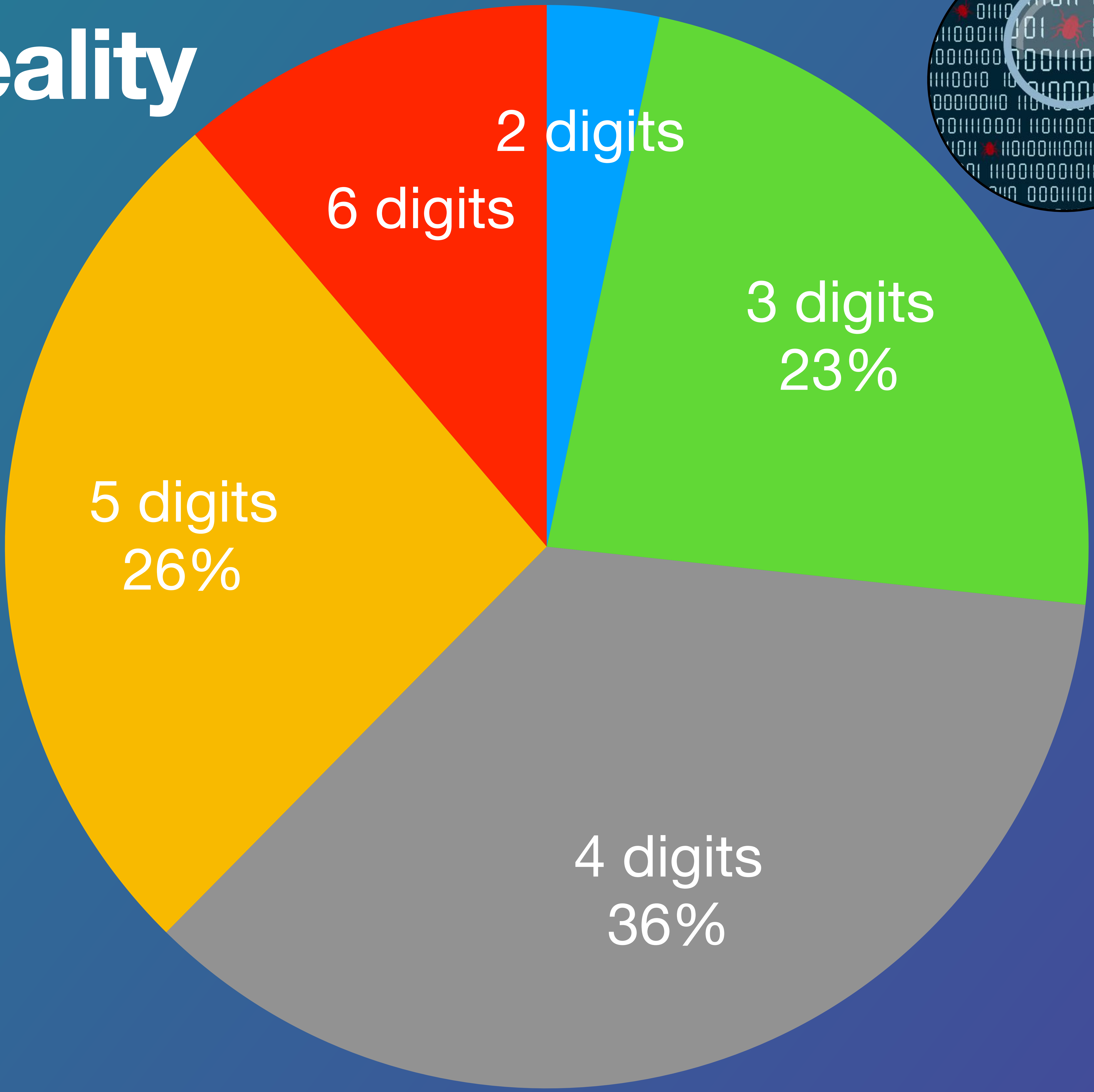
- Deliberate practice
- Learning new patterns
- Learning how to fix issues
- Find incomplete patches

Analysing CVE (Code Review)

- Read the advisory!
- Clone the repository
- Find the tag for the vulnerable and fixed versions
- Extract a patch/diff
- Analyse the patch/diff:
    - What does the vulnerable code look like?
    - What does the fix look like?
    - Is this properly fixed?
    - More vulnerabilities in the same area?

Expectations vs Reality

575 CVEs (diff based on vulnerable version versus patched version):



Expectations vs Reality

[VULNERABLE] 👩‍💻 🪲 [PATCHED]

[VULNERABLE] 👩‍💻 🪲 👩🏾‍💻 [PATCHED]

[VULNERABLE] 👩‍💻 👨🏻‍💻 🪲 👨🏽‍💻 [PATCHED]

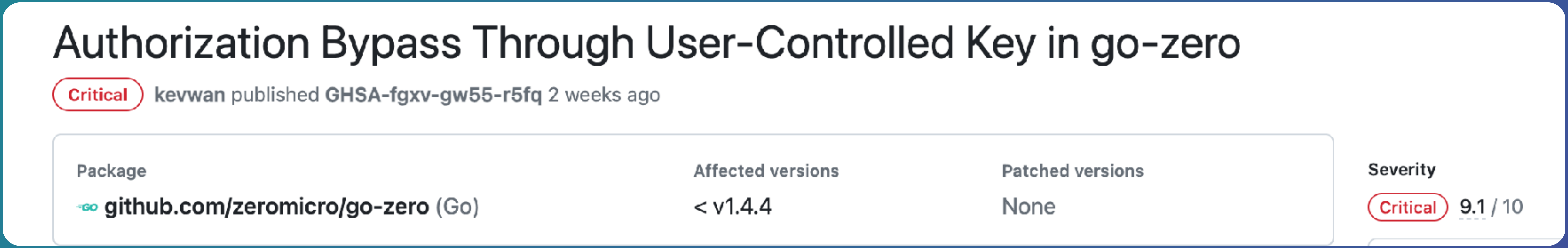
[VULNERABLE] 👩‍💻 👨🏻‍💻 🪲 👨🏽‍💻 👩🏾‍💻 [PATCHED]

[VULNERABLE] 👩‍💻 👨🏻‍💻 🪲 👨🏽‍💻 👩🏾‍💻 👨🏿‍💻 [PATCHED]

Methodology

Analysing CVE (Code Review)

https://github.com/zeromicro/go-zero/security/advisories/GHSA-fgxv-gw55-r5fq



Authorization Bypass Through User-Controlled Key in go-zero

Critical **kevwan** published **GHSA-fgxv-gw55-r5fq** 2 weeks ago

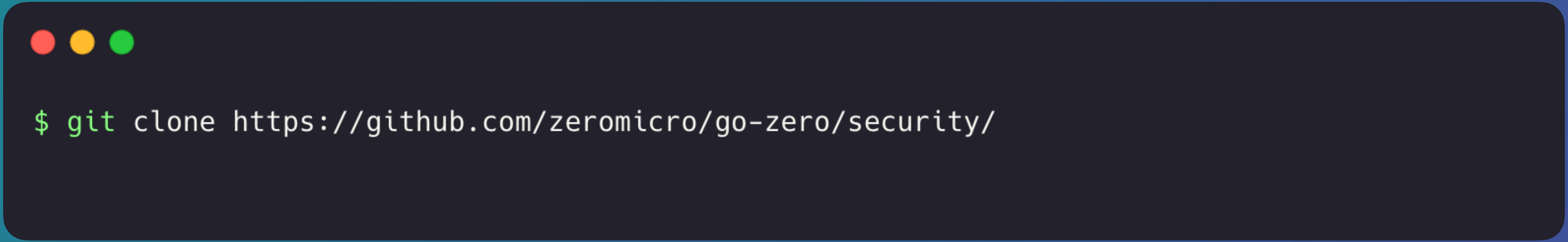
Package	Affected versions	Patched versions
github.com/zeromicro/go-zero (Go)	< v1.4.4	None

Severity

Critical 9.1 / 10

Analysing CVE (Code Review)

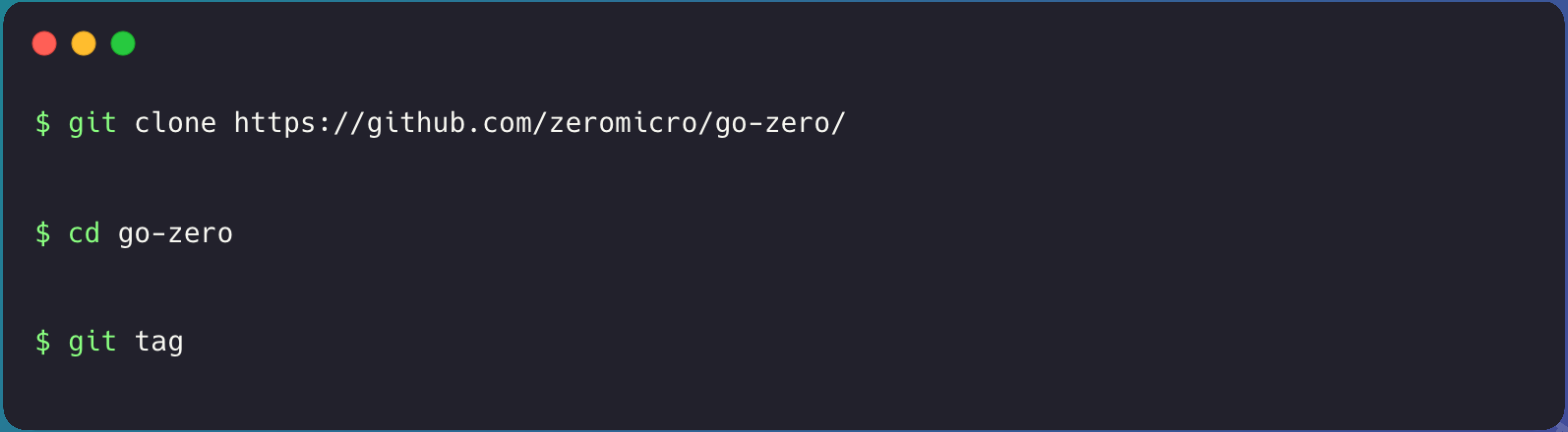
https://github.com/zeromicro/go-zero/security/advisories/GHSA-fgxv-gw55-r5fq



```
$ git clone https://github.com/zeromicro/go-zero/security/
```

Analysing CVE (Code Review)

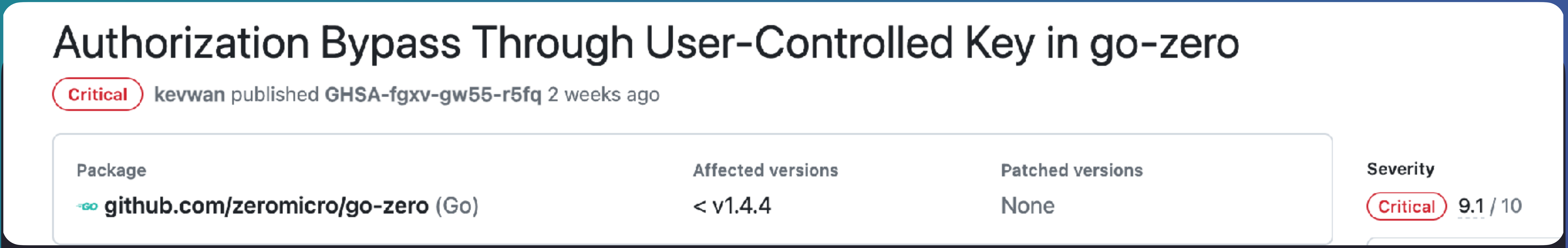
https://github.com/zeromicro/go-zero/security/advisories/GHSA-fgxv-gw55-r5fq



```
$ git clone https://github.com/zeromicro/go-zero/

$ cd go-zero

$ git tag
```

Analysing CVE (Code Review)

https://github.com/zeromicro/go-zero/security/advisories/GHSA-fgxv-gw55-r5fq

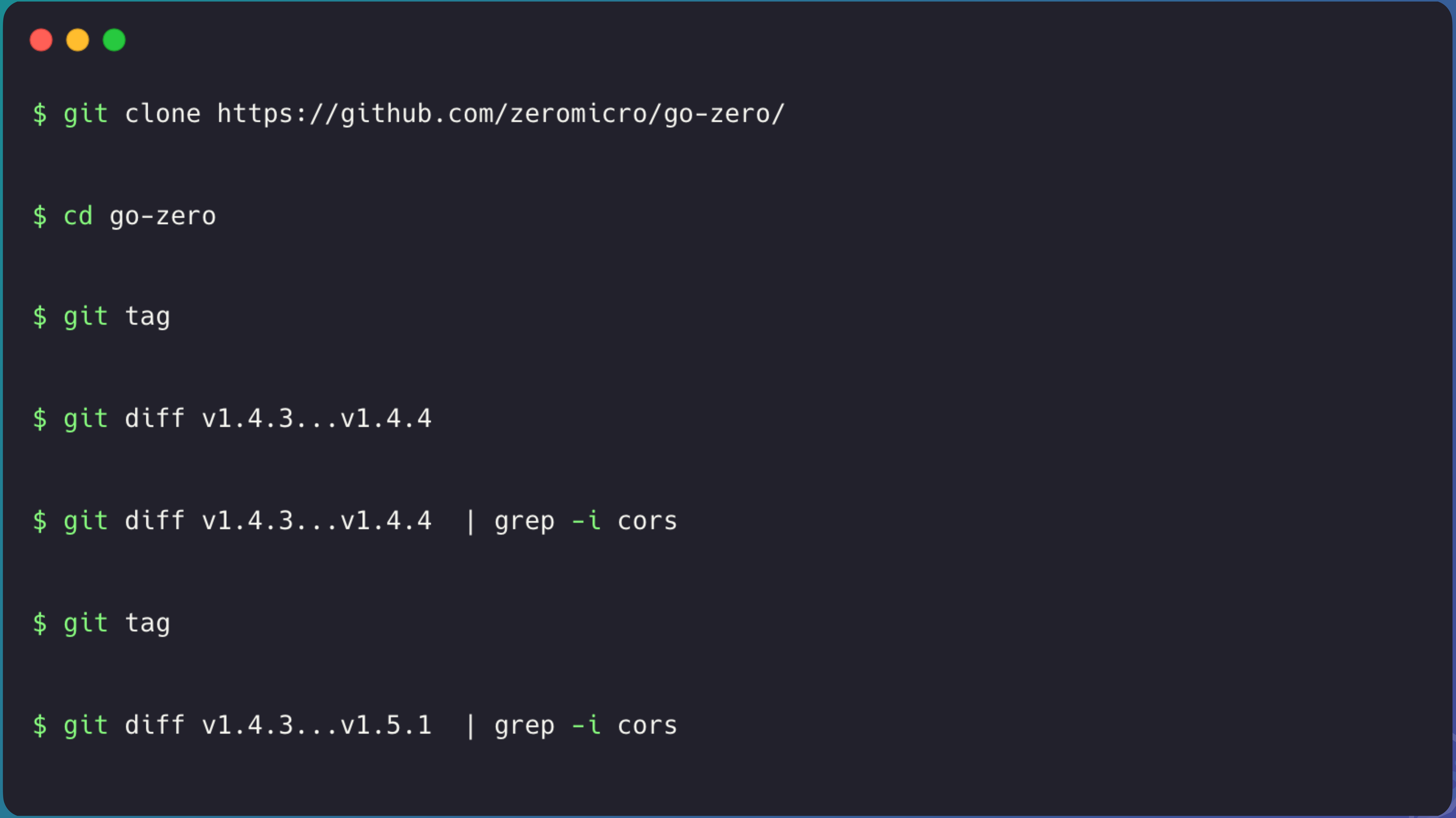


Authorization Bypass Through User-Controlled Key in go-zero

Critical **kevwan** published **GHSA-fgxv-gw55-r5fq** 2 weeks ago

Package	Affected versions	Patched versions	Severity
github.com/zeromicro/go-zero (Go)	< v1.4.4	None	Critical 9.1 / 10

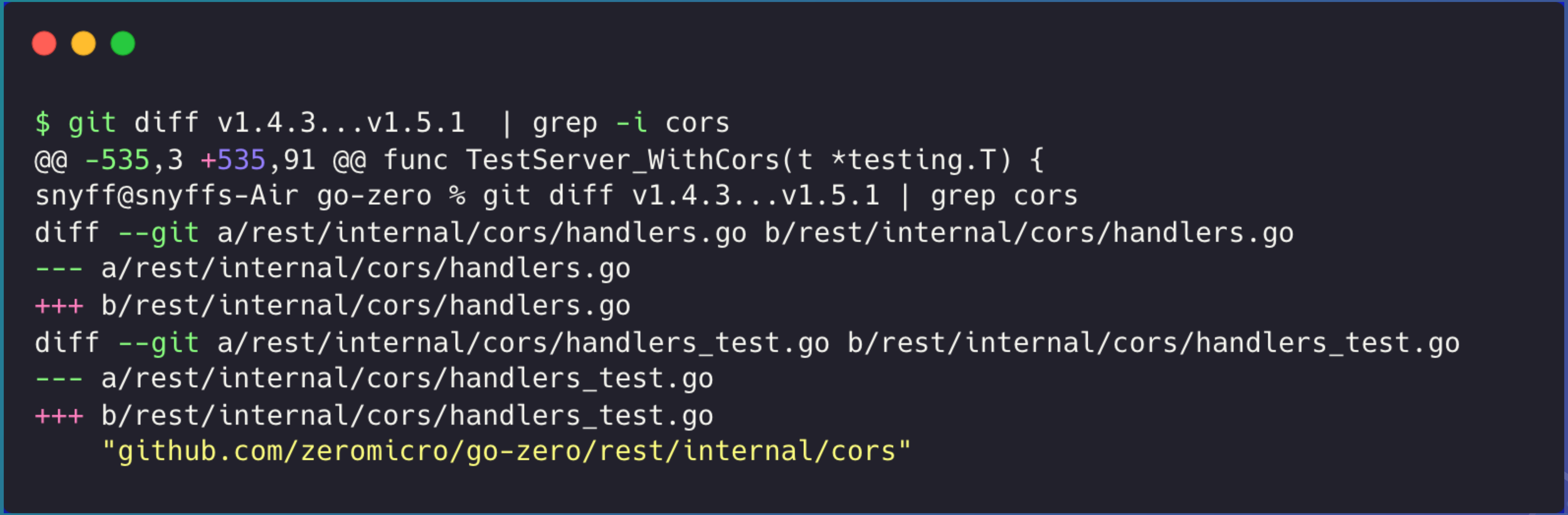
```
$ cd go-zero

$ git tag

$ git diff v1.4.3...v1.4.4
```



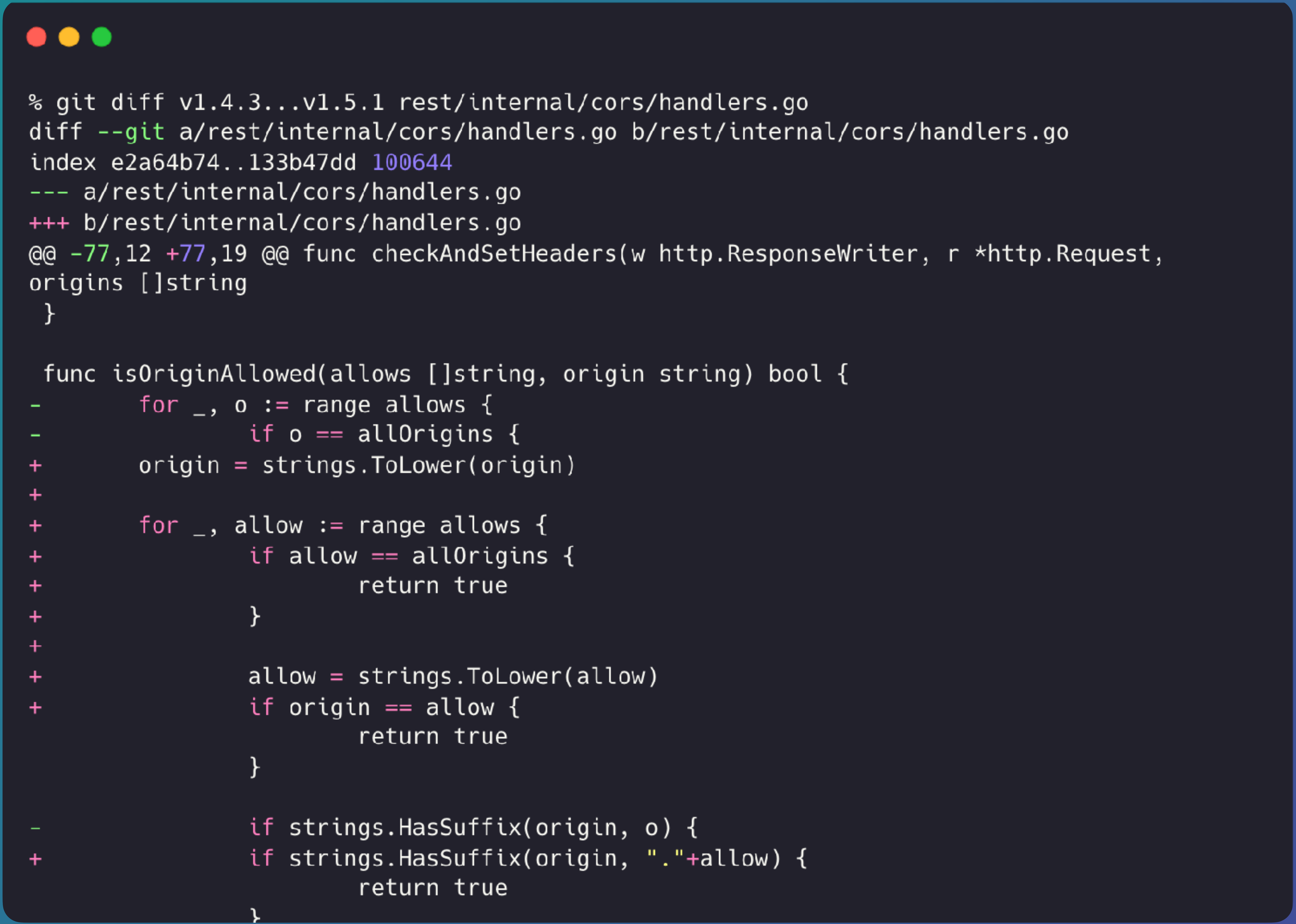
```
$ git clone https://github.com/zeromicro/go-zero/

$ cd go-zero

$ git tag

$ git diff v1.4.3...v1.4.4

$ git diff v1.4.3...v1.4.4  | grep -i cors

$ git tag

$ git diff v1.4.3...v1.5.1  | grep -i cors
```

Analysing CVE (Code Review)

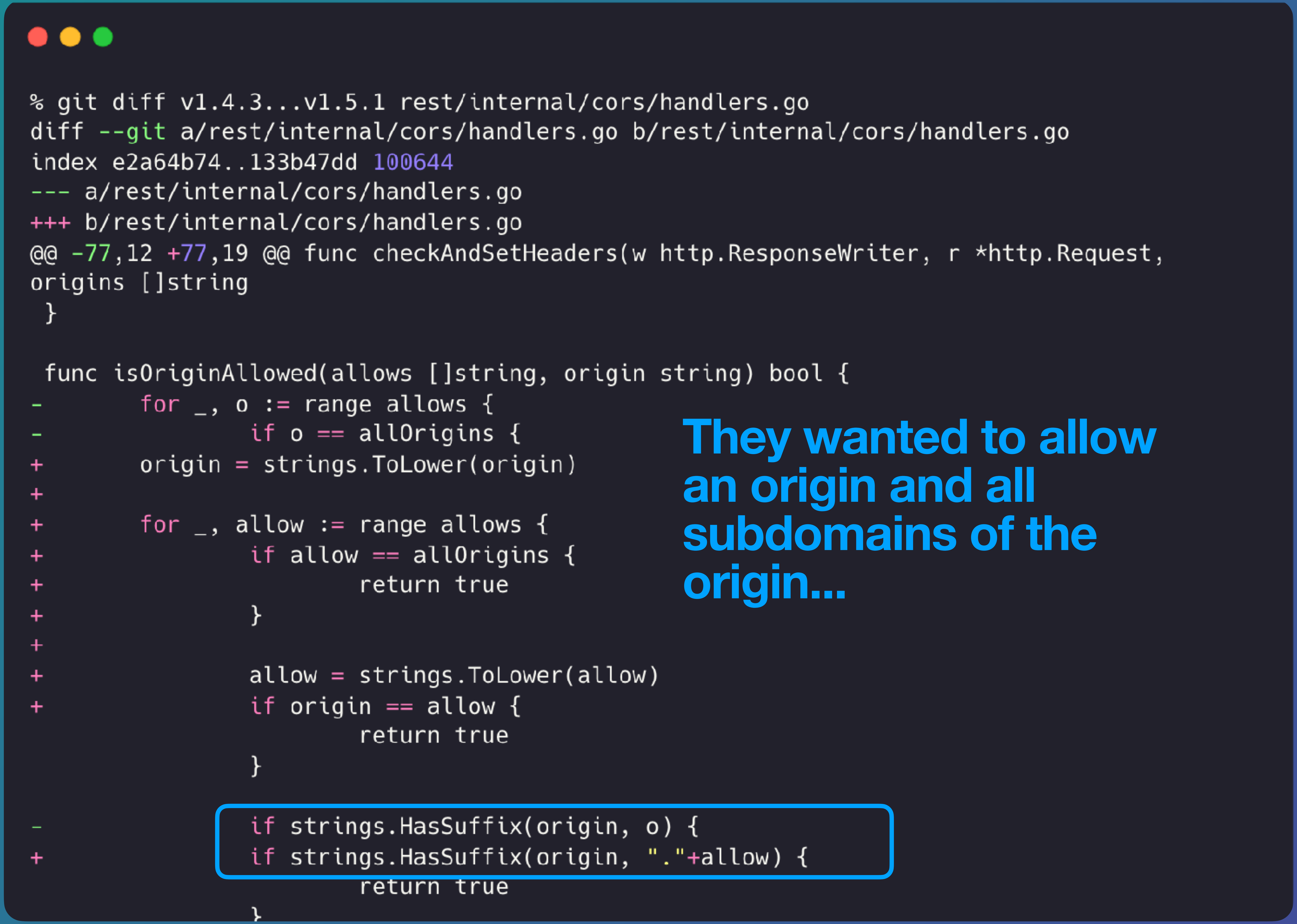
https://github.com/zeromicro/go-zero/security/advisories/GHSA-fgxv-gw55-r5fq



```
$ git diff v1.4.3...v1.5.1  | grep -i cors
@@ -535,3 +535,91 @@ func TestServer_WithCors(t *testing.T) {
snyff@snyffs-Air go-zero % git diff v1.4.3...v1.5.1 | grep cors
diff --git a/rest/internal/cors/handlers.go b/rest/internal/cors/handlers.go
--- a/rest/internal/cors/handlers.go
+++ b/rest/internal/cors/handlers.go
diff --git a/rest/internal/cors/handlers_test.go b/rest/internal/cors/handlers_test.go
--- a/rest/internal/cors/handlers_test.go
+++ b/rest/internal/cors/handlers_test.go
    "github.com/zeromicro/go-zero/rest/internal/cors"
```

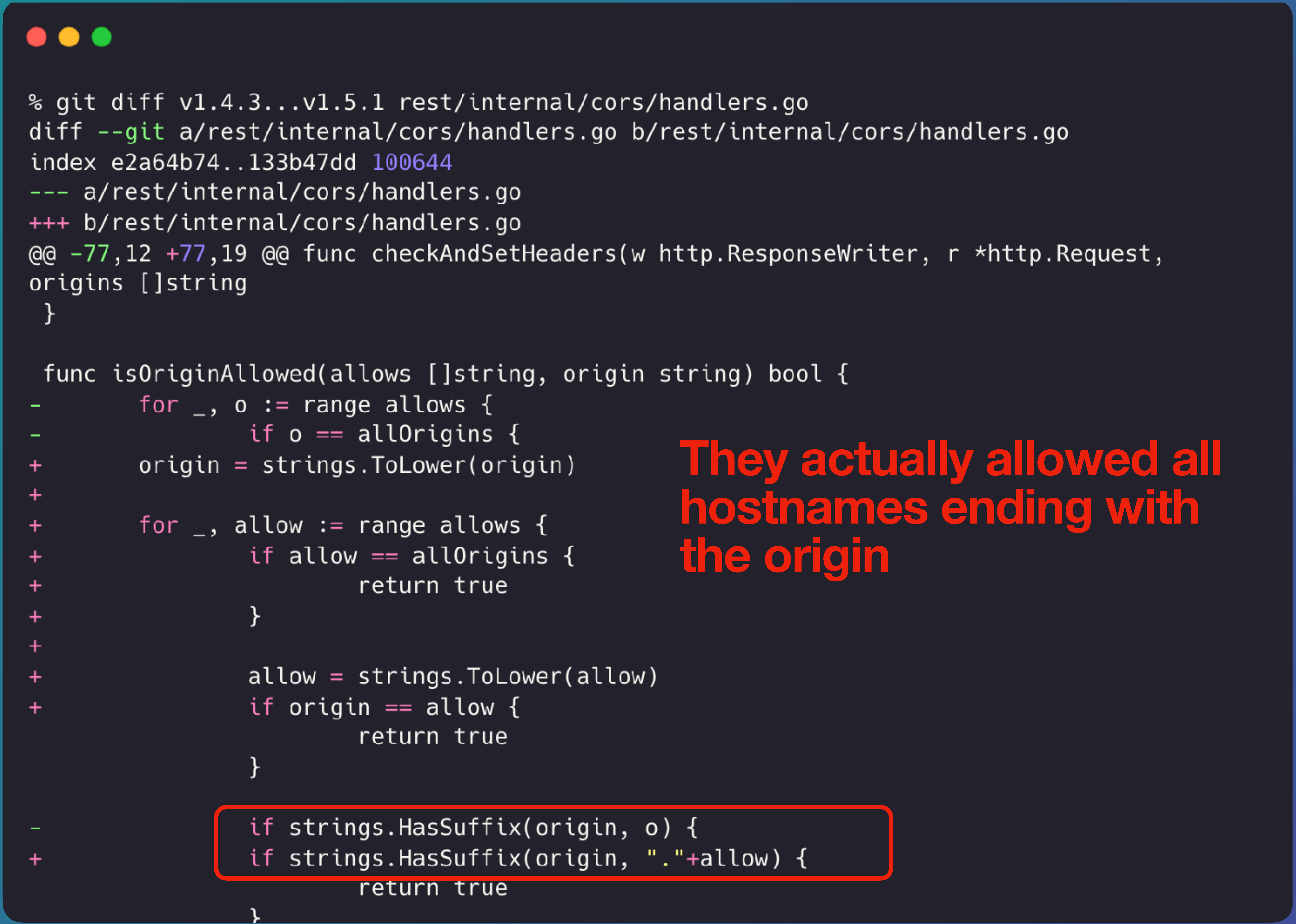


```
% git diff v1.4.3...v1.5.1 rest/internal/cors/handlers.go
diff --git a/rest/internal/cors/handlers.go b/rest/internal/cors/handlers.go
index e2a64b74..133b47dd 100644
--- a/rest/internal/cors/handlers.go
+++ b/rest/internal/cors/handlers.go
@@ -77,12 +77,19 @@ func checkAndSetHeaders(w http.ResponseWriter, r *http.Request,
origins []string
 }

 func isOriginAllowed(allows []string, origin string) bool {
-       for _, o := range allows {
-               if o == allOrigins {
+       origin = strings.ToLower(origin)
+
+       for _, allow := range allows {
+               if allow == allOrigins {
+                       return true
+               }
+
+               allow = strings.ToLower(allow)
+               if origin == allow {
                        return true
                }

-               if strings.HasSuffix(origin, o) {
+               if strings.HasSuffix(origin, "."+allow) {
                        return true
                }
```



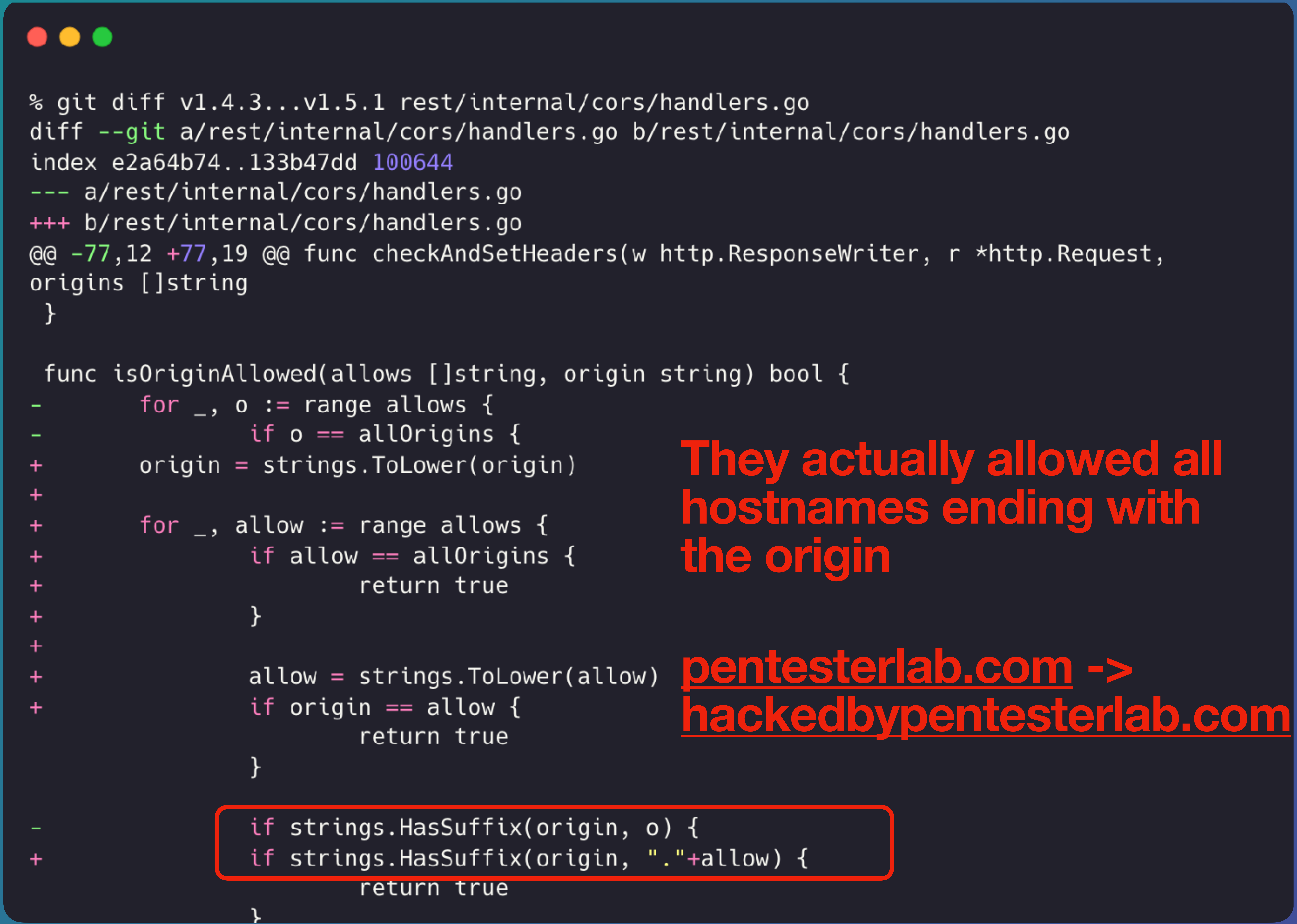
```
% git diff v1.4.3...v1.5.1 rest/internal/cors/handlers.go
diff --git a/rest/internal/cors/handlers.go b/rest/internal/cors/handlers.go
index e2a64b74..133b47dd 100644
--- a/rest/internal/cors/handlers.go
+++ b/rest/internal/cors/handlers.go
@@ -77,12 +77,19 @@ func checkAndSetHeaders(w http.ResponseWriter, r *http.Request,
origins []string
 }

 func isOriginAllowed(allows []string, origin string) bool {
-   for _, o := range allows {
-       if o == allOrigins {
+   origin = strings.ToLower(origin)
+
+   for _, allow := range allows {
+       if allow == allOrigins {
+           return true
+       }
+
+       allow = strings.ToLower(allow)
+       if origin == allow {
            return true
        }

-       if strings.HasSuffix(origin, o) {
+       if strings.HasSuffix(origin, "."+allow) {
            return true
        }
```

They wanted to allow an origin and all subdomains of the origin...



```
% git diff v1.4.3...v1.5.1 rest/internal/cors/handlers.go
diff --git a/rest/internal/cors/handlers.go b/rest/internal/cors/handlers.go
index e2a64b74..133b47dd 100644
--- a/rest/internal/cors/handlers.go
+++ b/rest/internal/cors/handlers.go
@@ -77,12 +77,19 @@ func checkAndSetHeaders(w http.ResponseWriter, r *http.Request,
origins []string
 }

 func isOriginAllowed(allows []string, origin string) bool {
-       for _, o := range allows {
-               if o == allOrigins {
+       origin = strings.ToLower(origin)
+
+       for _, allow := range allows {
+               if allow == allOrigins {
+                       return true
+               }
+
+               allow = strings.ToLower(allow)
+               if origin == allow {
                        return true
                }

-               if strings.HasSuffix(origin, o) {
+               if strings.HasSuffix(origin, "."+allow) {
                        return true
                }
```

They actually allowed all hostnames ending with the origin

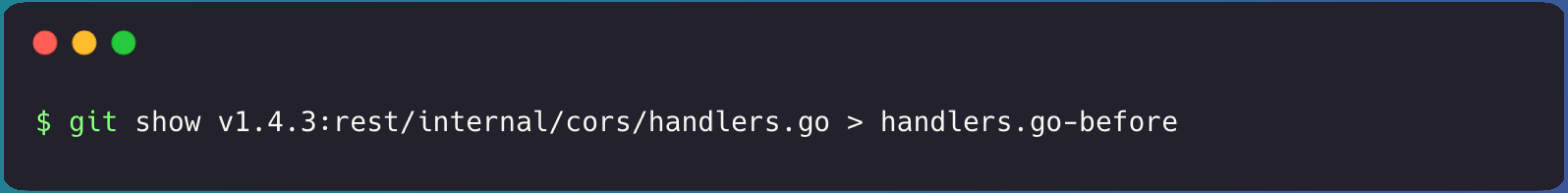


```
% git diff v1.4.3...v1.5.1 rest/internal/cors/handlers.go
diff --git a/rest/internal/cors/handlers.go b/rest/internal/cors/handlers.go
index e2a64b74..133b47dd 100644
--- a/rest/internal/cors/handlers.go
+++ b/rest/internal/cors/handlers.go
@@ -77,12 +77,19 @@ func checkAndSetHeaders(w http.ResponseWriter, r *http.Request,
origins []string
 }

 func isOriginAllowed(allows []string, origin string) bool {
-       for _, o := range allows {
-               if o == allOrigins {
+       origin = strings.ToLower(origin)
+
+       for _, allow := range allows {
+               if allow == allOrigins {
+                       return true
+               }
+
+               allow = strings.ToLower(allow)
+               if origin == allow {
                        return true
                }

-               if strings.HasSuffix(origin, o) {
+               if strings.HasSuffix(origin, "."+allow) {
                        return true
                }
```

They actually allowed all hostnames ending with the origin

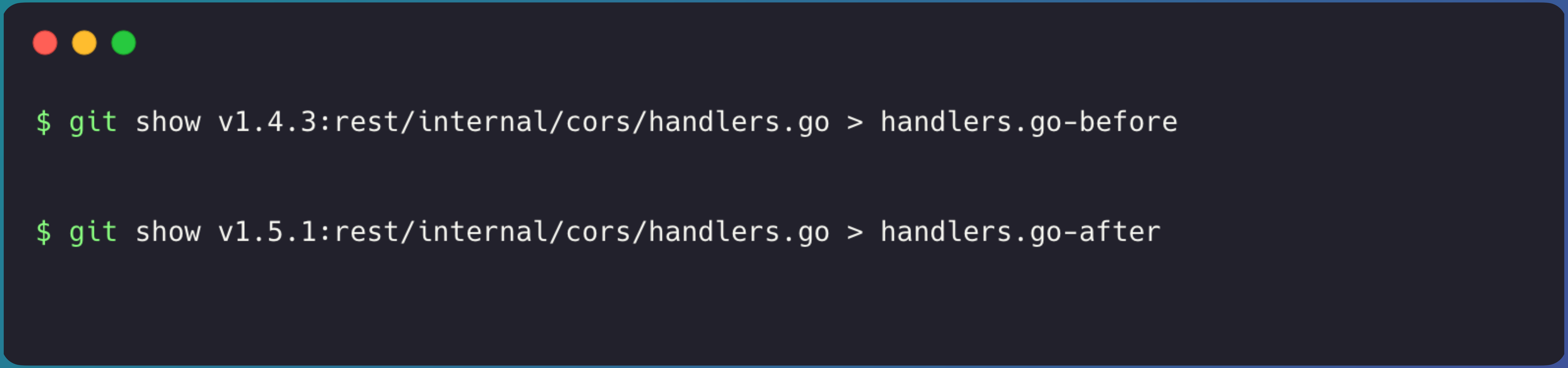
pentesterlab.com -> hackedbypentesterlab.com

Analysing CVE (Code Review)



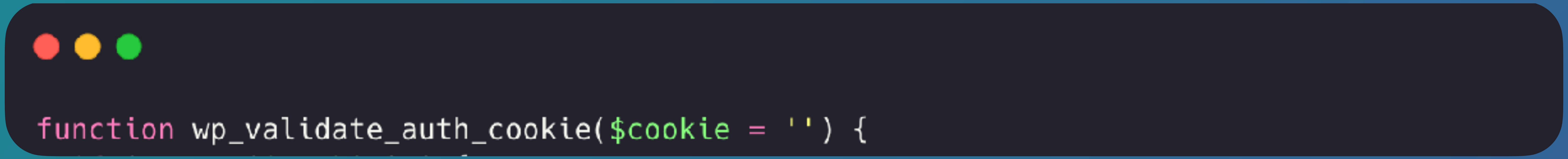
```
$ git show v1.4.3:rest/internal/cors/handlers.go > handlers.go-before
```

Analysing CVE (Code Review)

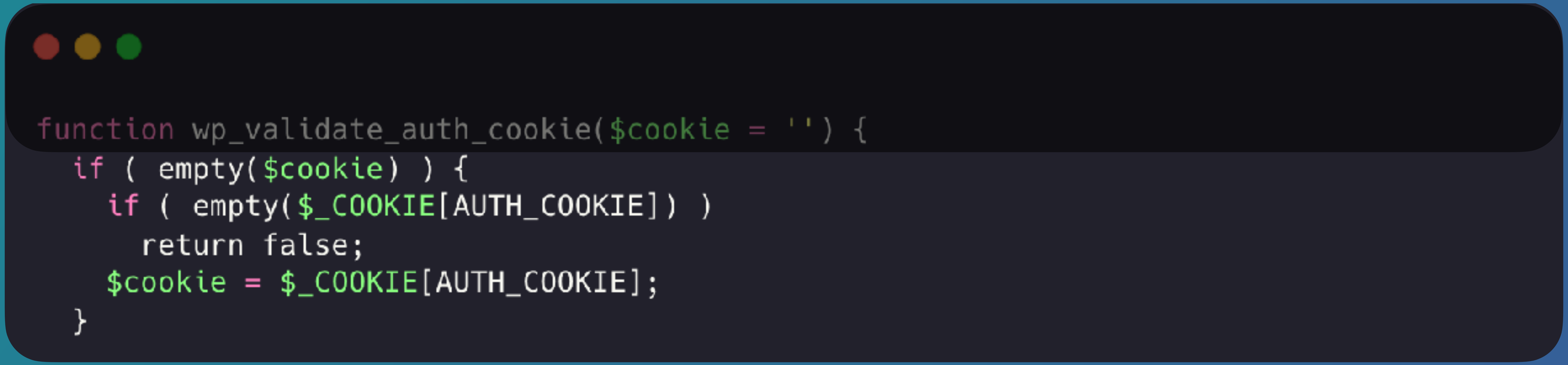


```
$ git show v1.4.3:rest/internal/cors/handlers.go > handlers.go-before

$ git show v1.5.1:rest/internal/cors/handlers.go > handlers.go-after
```

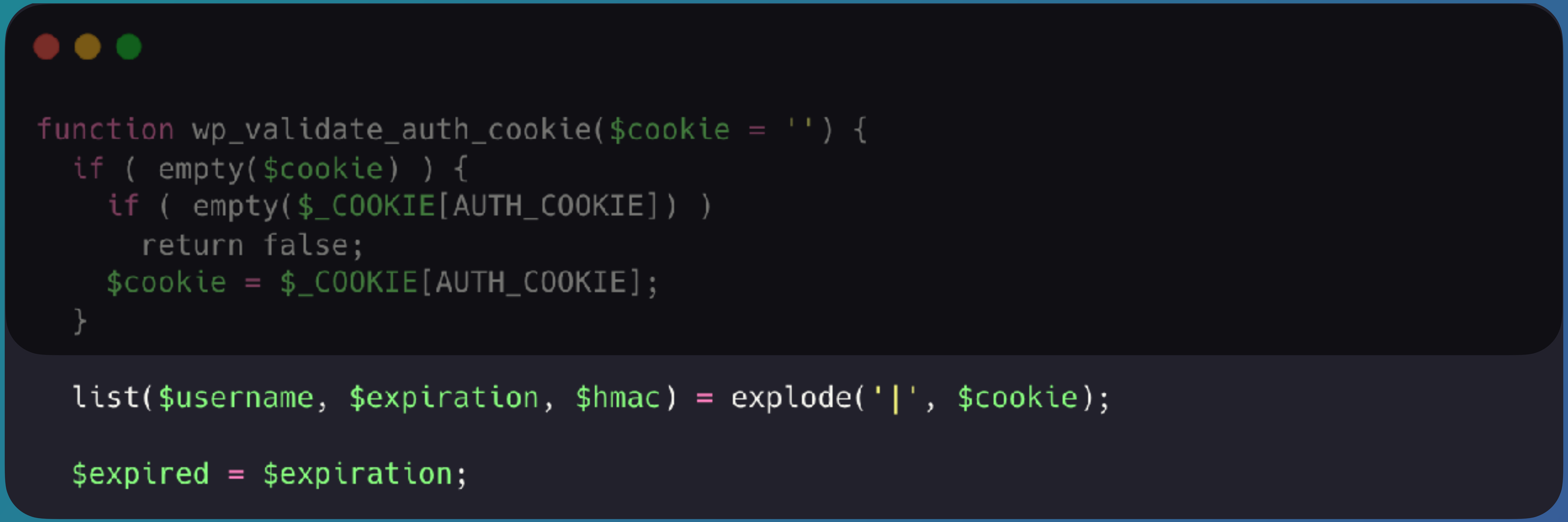
CVE-2008-1930



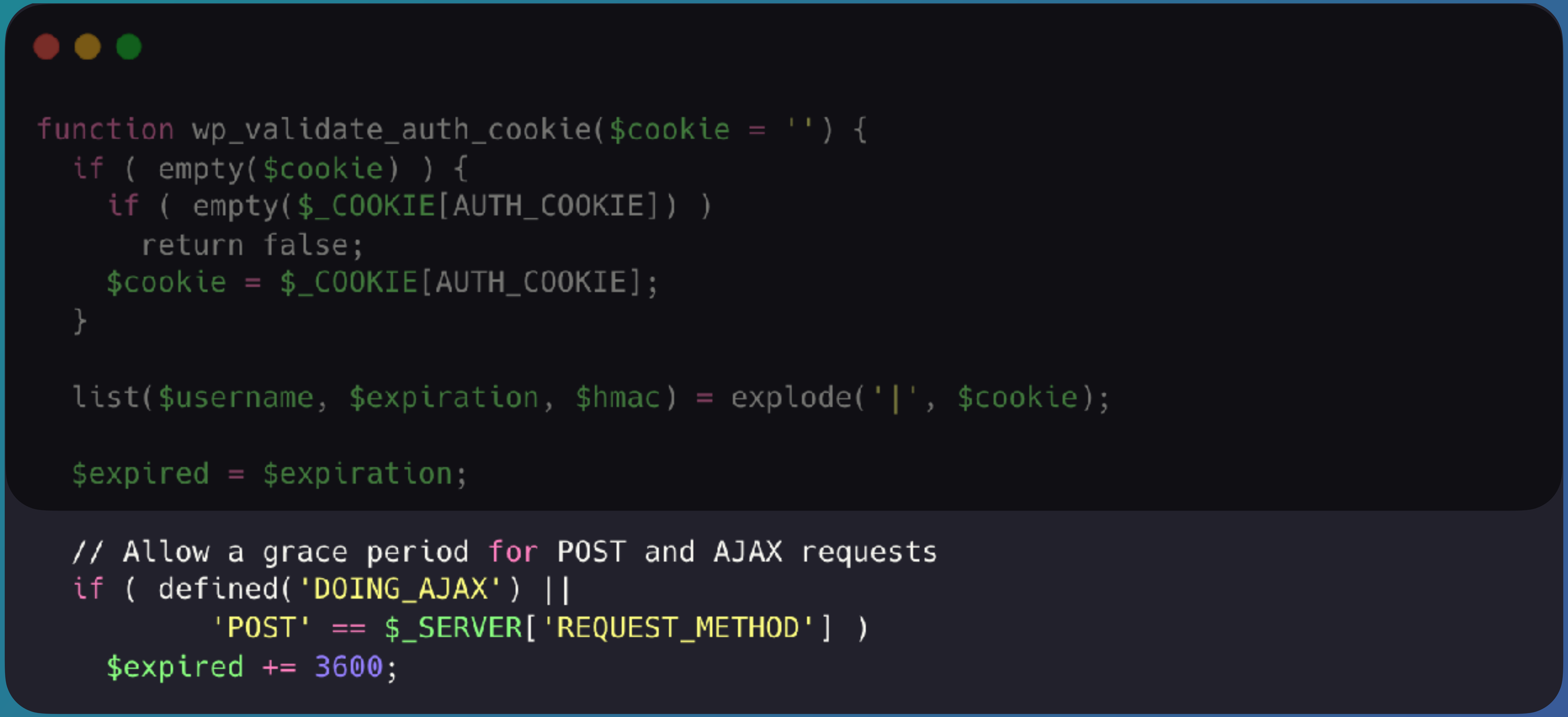
```
function wp_validate_auth_cookie($cookie = '') {
```



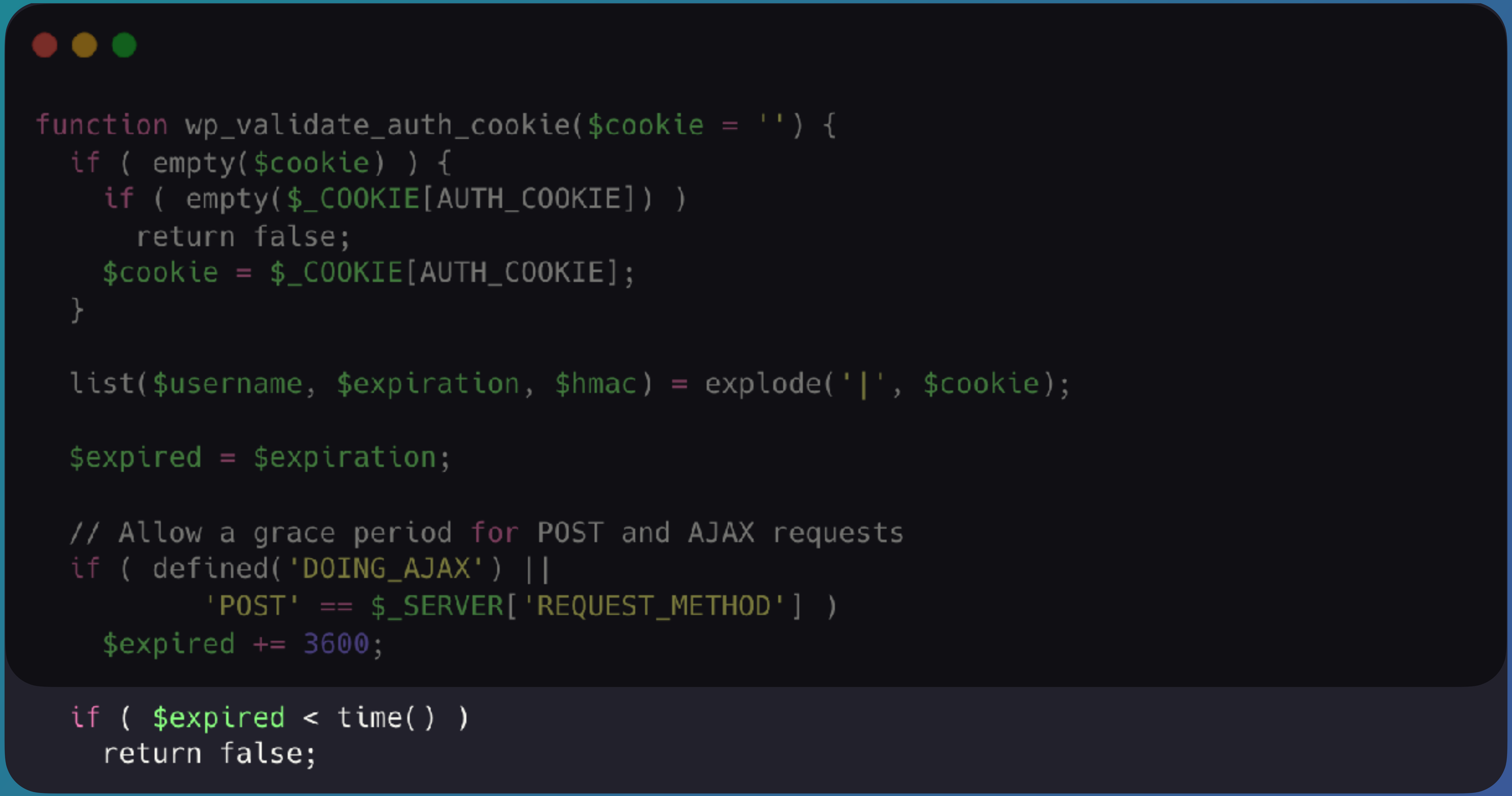
```php
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }
```



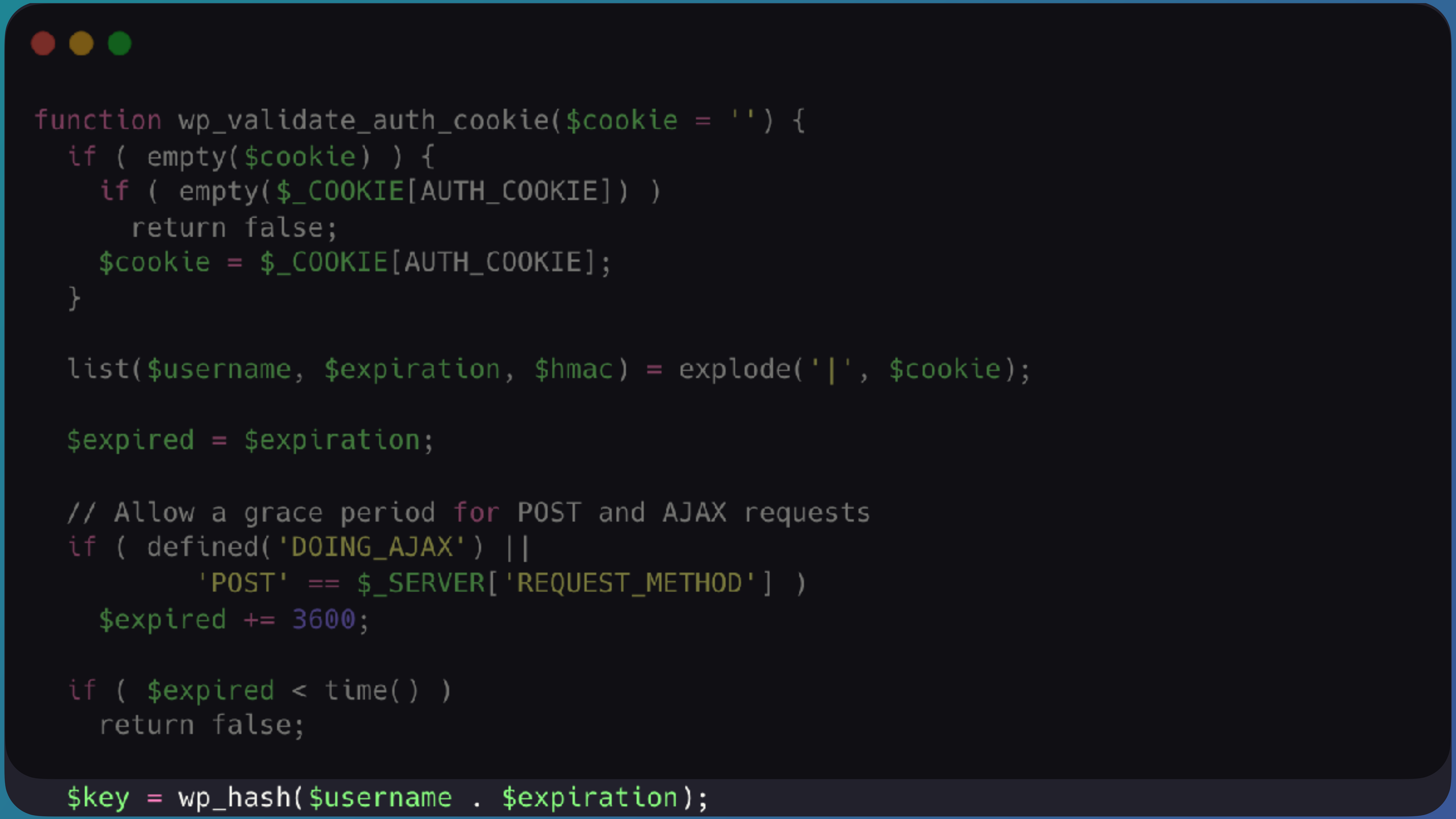
```php
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;
```



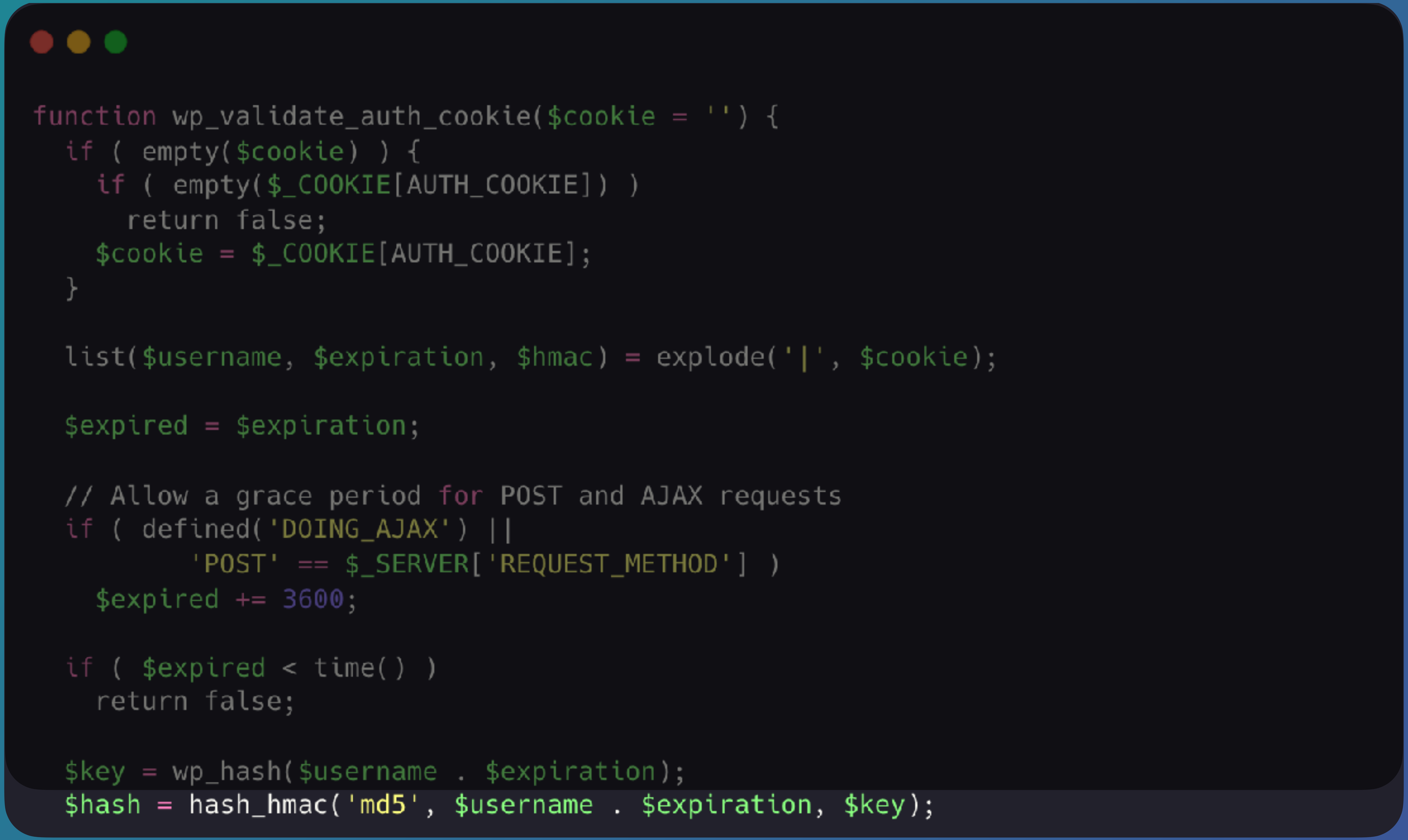
```
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;
```



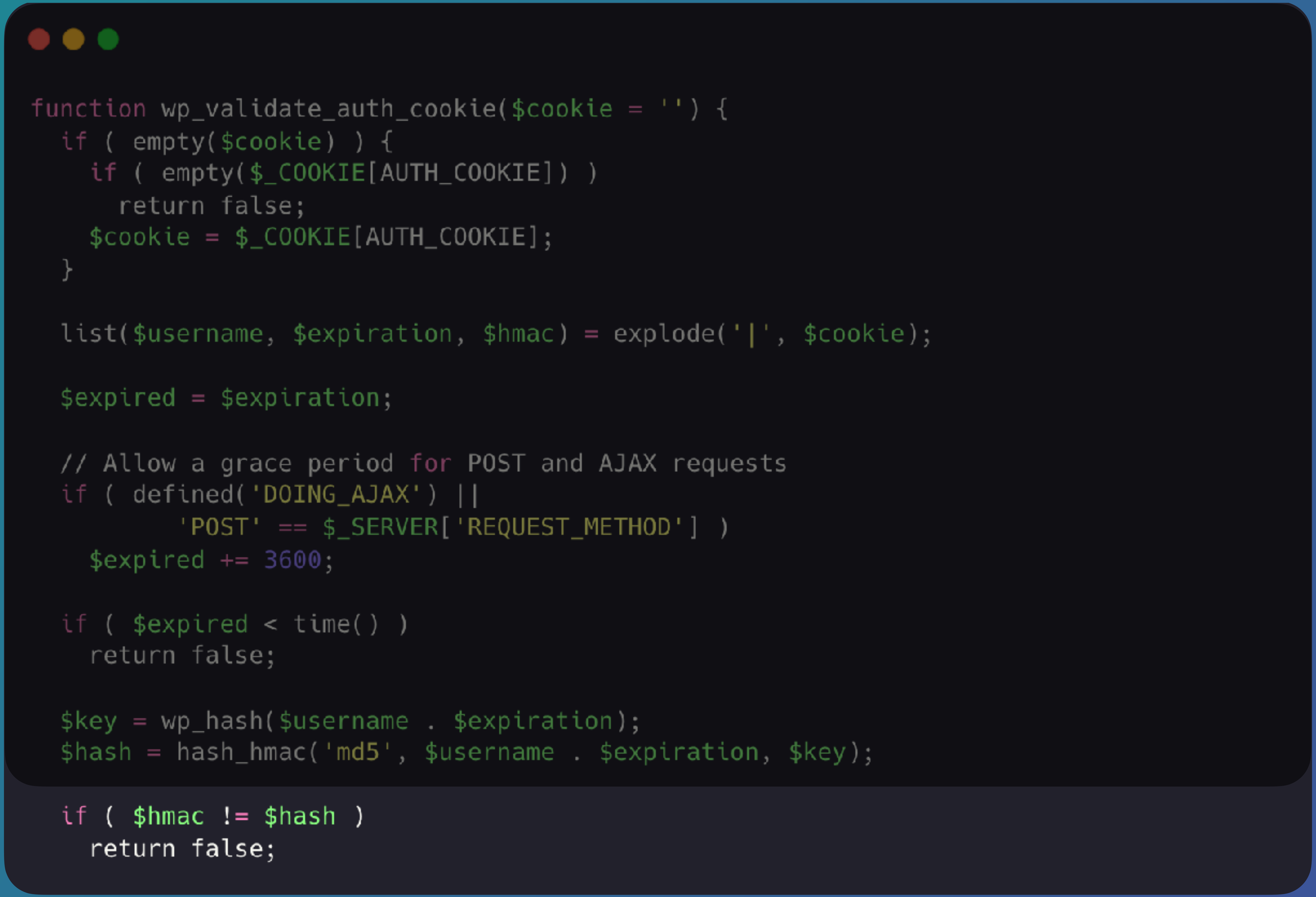
```php
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;

  if ( $expired < time() )
    return false;
```



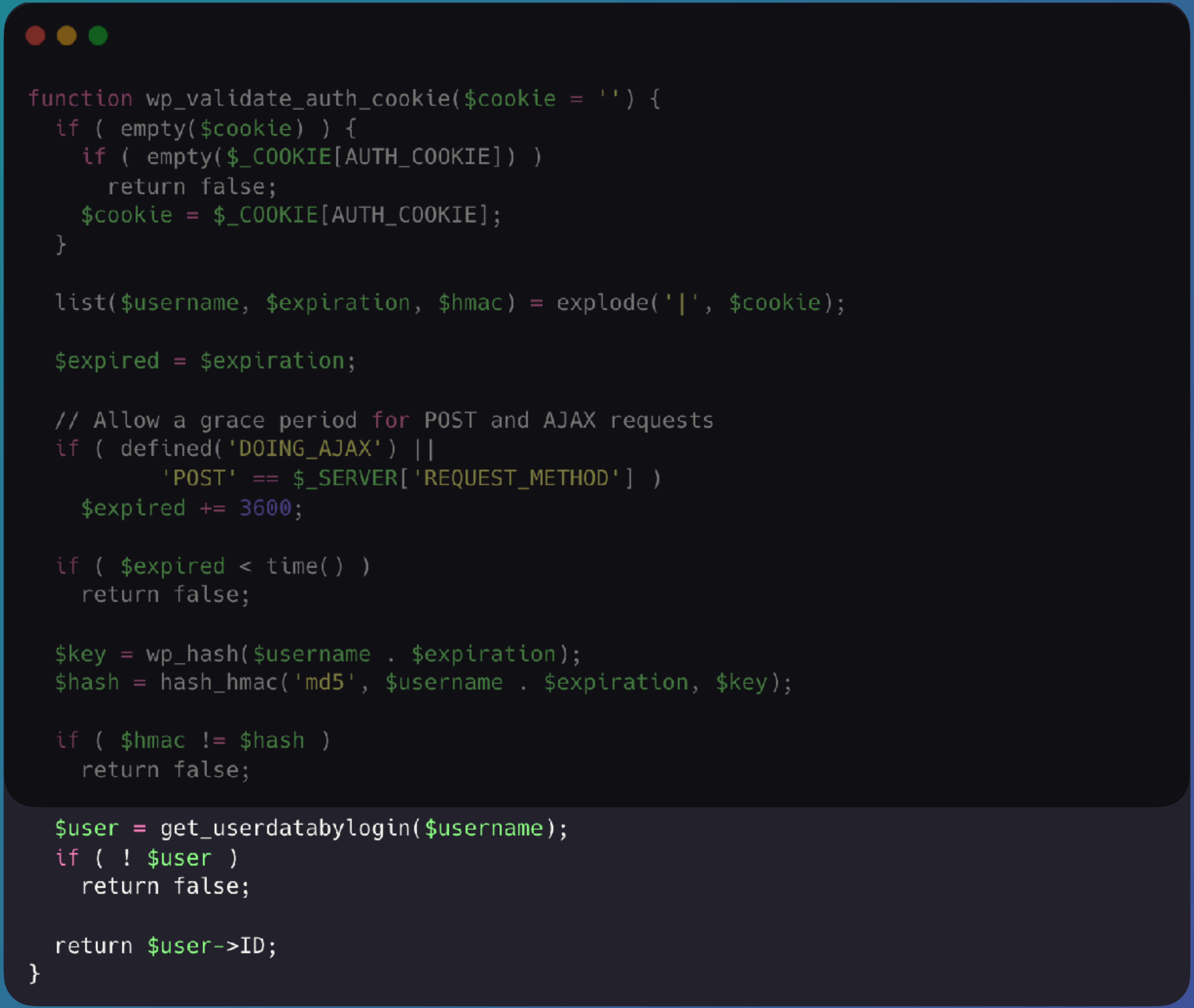
```php
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;

  if ( $expired < time() )
    return false;

  $key = wp_hash($username . $expiration);
```



```
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;

  if ( $expired < time() )
    return false;

  $key = wp_hash($username . $expiration);
  $hash = hash_hmac('md5', $username . $expiration, $key);
```



```php
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;

  if ( $expired < time() )
    return false;

  $key = wp_hash($username . $expiration);
  $hash = hash_hmac('md5', $username . $expiration, $key);

  if ( $hmac != $hash )
    return false;
```

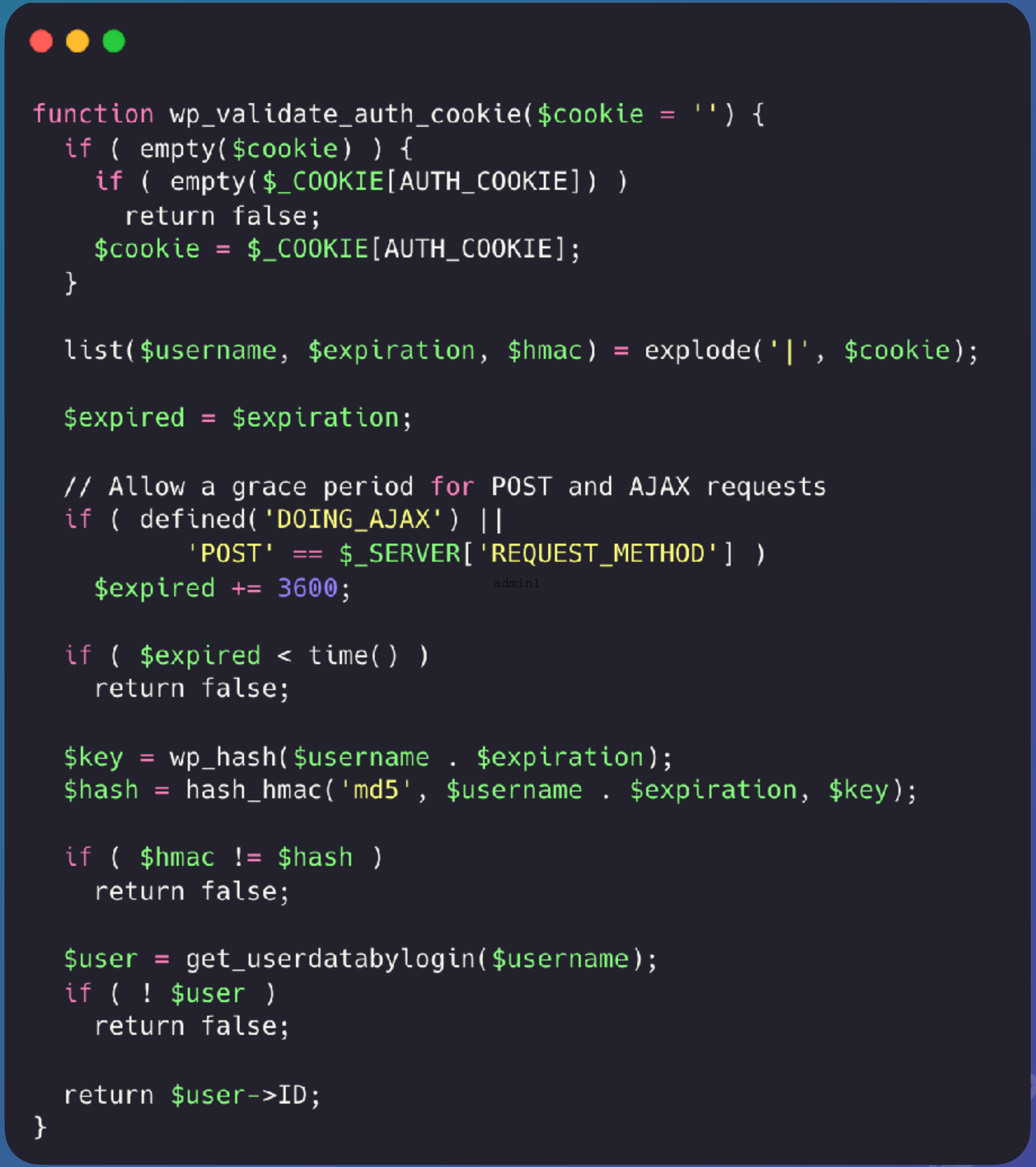


```php
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;

  if ( $expired < time() )
    return false;

  $key = wp_hash($username . $expiration);
  $hash = hash_hmac('md5', $username . $expiration, $key);

  if ( $hmac != $hash )
    return false;

  $user = get_userdatabylogin($username);
  if ( ! $user )
    return false;

  return $user->ID;
}
```

admin:1353464343:16849b89783b5918a41bbd29a3c4bbf6

admin
1353464343
16849b89783b5918a41bbd29a3c4bbf6

hmac(admin1353464343)



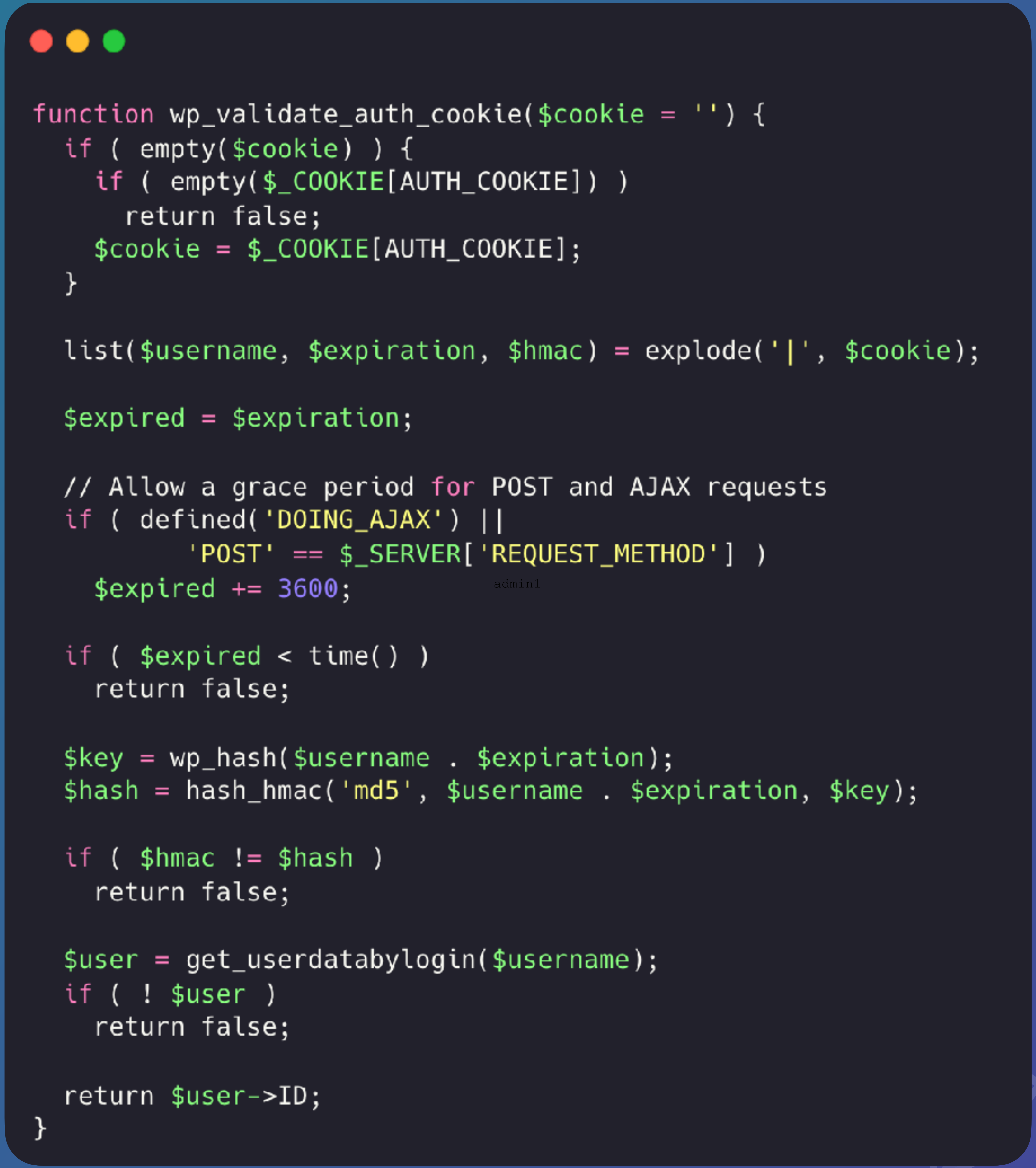
```
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;

  if ( $expired < time() )
    return false;

  $key = wp_hash($username . $expiration);
  $hash = hash_hmac('md5', $username . $expiration, $key);

  if ( $hmac != $hash )
    return false;

  $user = get_userdatabylogin($username);
  if ( ! $user )
    return false;

  return $user->ID;
}
```

admin1

admin1:1353464343:1ba7d82099dd6119781b54ecf8b79259

admin1
1353464343
1ba7d82099dd6119781b54ecf8b79259

hmac(admin11353464343)



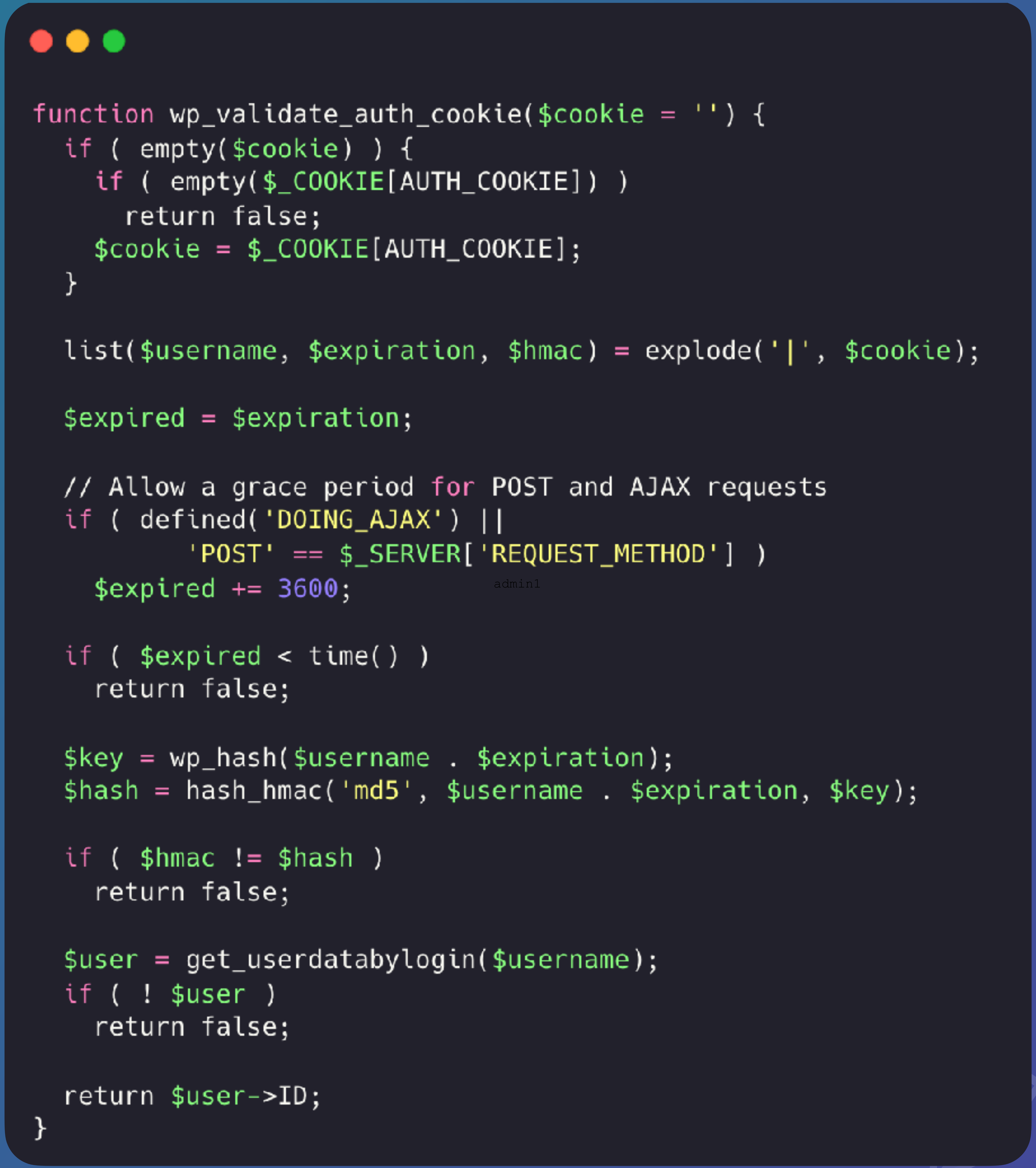
```
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;            admin1

  if ( $expired < time() )
    return false;

  $key = wp_hash($username . $expiration);
  $hash = hash_hmac('md5', $username . $expiration, $key);

  if ( $hmac != $hash )
    return false;

  $user = get_userdatabylogin($username);
  if ( ! $user )
    return false;

  return $user->ID;
}
```

admin**1**:1353464343:1ba7d82099dd6119781b54ecf8b79259
admin:11353464343:1ba7d82099dd6119781b54ecf8b79259

admin
11353464343
1ba7d82099dd6119781b54ecf8b79259

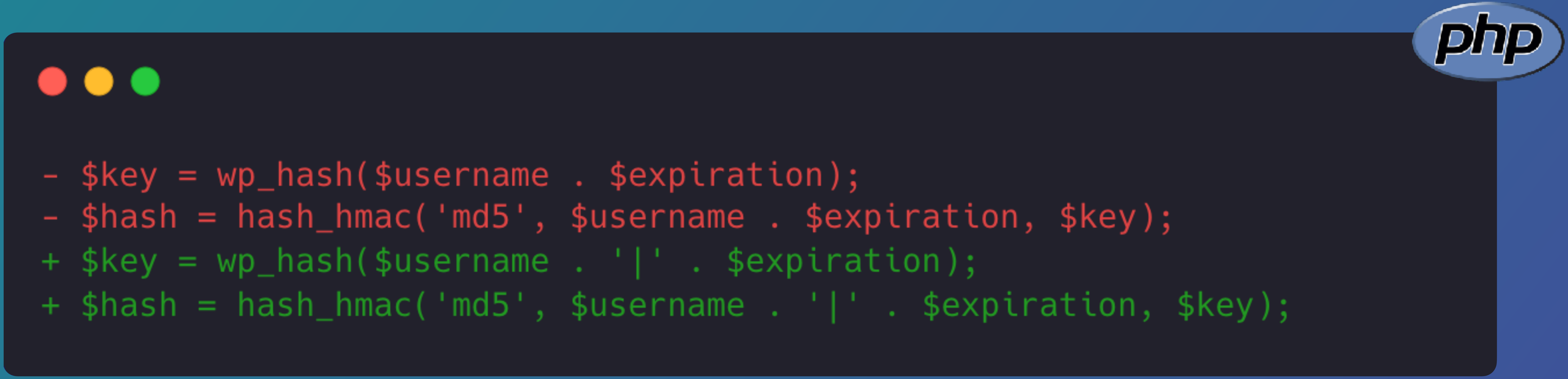
hmac(admin11353464343)

hmac(admin11353464343)



```php
function wp_validate_auth_cookie($cookie = '') {
  if ( empty($cookie) ) {
    if ( empty($_COOKIE[AUTH_COOKIE]) )
      return false;
    $cookie = $_COOKIE[AUTH_COOKIE];
  }

  list($username, $expiration, $hmac) = explode('|', $cookie);

  $expired = $expiration;

  // Allow a grace period for POST and AJAX requests
  if ( defined('DOING_AJAX') ||
          'POST' == $_SERVER['REQUEST_METHOD'] )
    $expired += 3600;

  if ( $expired < time() )
    return false;

  $key = wp_hash($username . $expiration);
  $hash = hash_hmac('md5', $username . $expiration, $key);

  if ( $hmac != $hash )
    return false;

  $user = get_userdatabylogin($username);
  if ( ! $user )
    return false;

  return $user->ID;
}
```

The Fix



```
- $key = wp_hash($username . $expiration);
- $hash = hash_hmac('md5', $username . $expiration, $key);
+ $key = wp_hash($username . '|' . $expiration);
+ $hash = hash_hmac('md5', $username . '|' . $expiration, $key);
```

Lesson learned:
Always include a delimiter between values when signing data.

CONCLUSI🧐N

Assumptions!

Developers, yours, ...

Dum

“All important targets require substantial initial investments before discovering and consistently discovering vulnerabilities.”

- Silvio Cesare

If you create software, you may have as many CVEs as you want
And don't forget to add them all to your CV!
imgflip.com

Keeping in touch

https://pentesterlab.com/

https://pentesterlab.com/live-training/

louis@pentesterlab.com

@pentesterlab.com and @snyff.pentesterlab.com

https://www.linkedin.com/company/pentesterlab/

https://www.linkedin.com/in/snyff/

@PentesterLab and @snyff

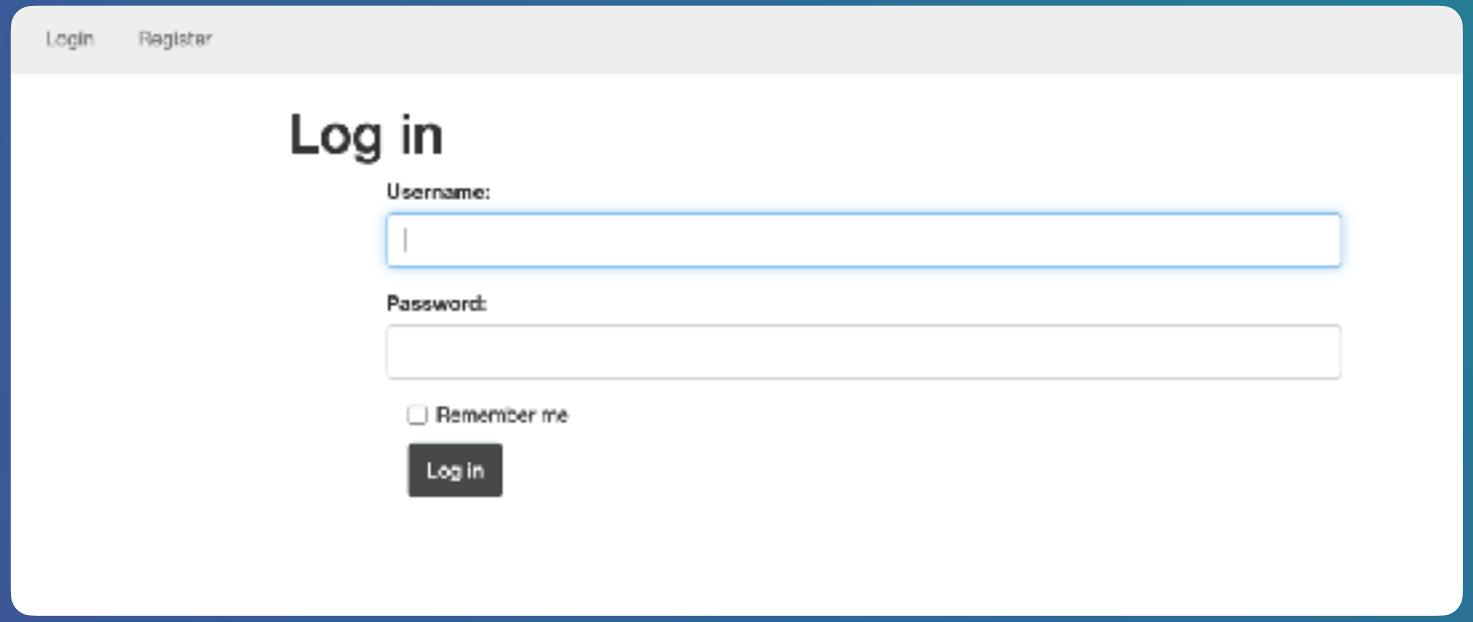
Conclusion

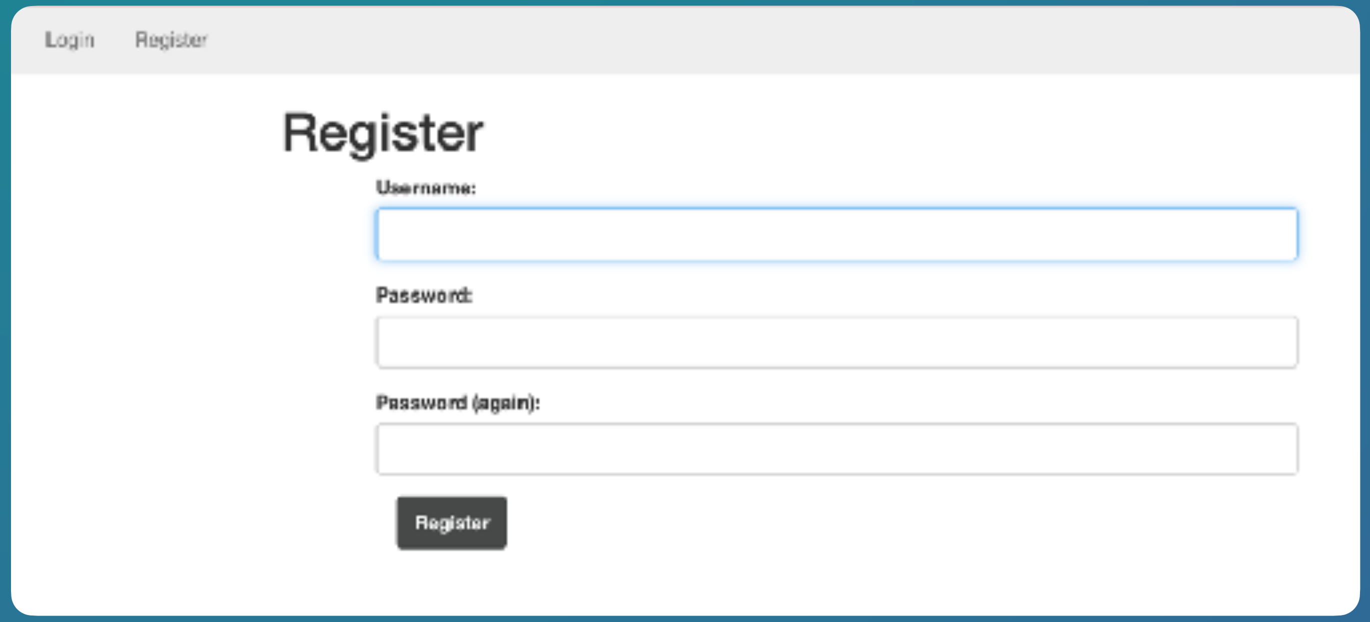
- Practice makes perfect
- There are still ****PLENTY**** of bugs to be found
- Keep notes!
- Now it's time to review some code!

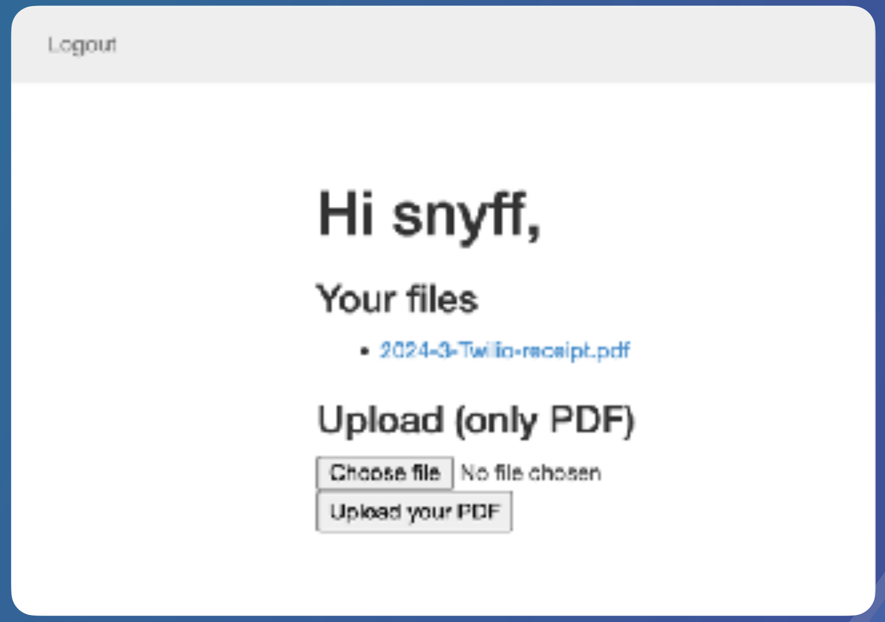
Hands-On 🫶

- One application written in both Golang and PHP:
  - PHP: https://github.com/PentesterLab/codereview-php
  - Golang: https://github.com/PentesterLab/codereview-golang

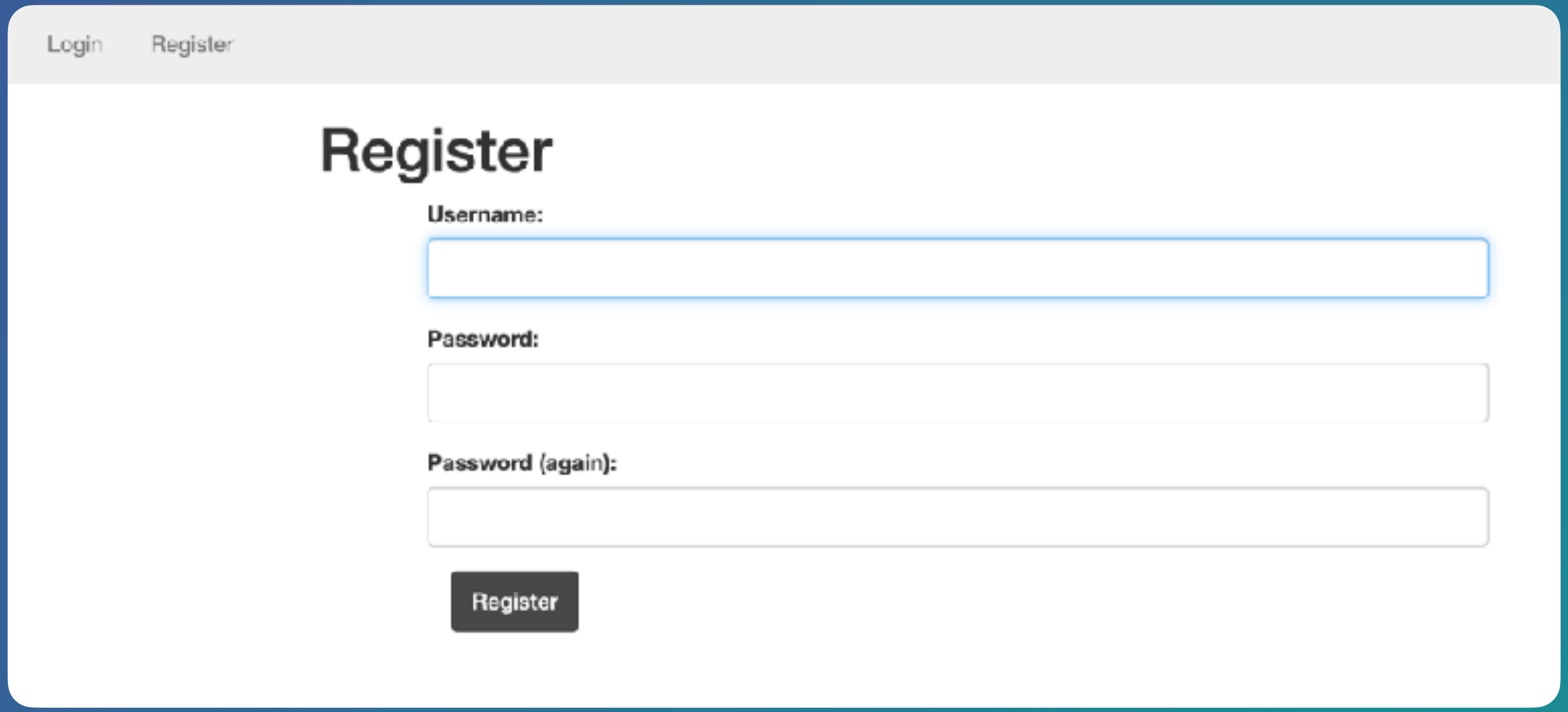
- A lot of vulnerabilities...







Hands-On 🫶: Code Review!


Login Register

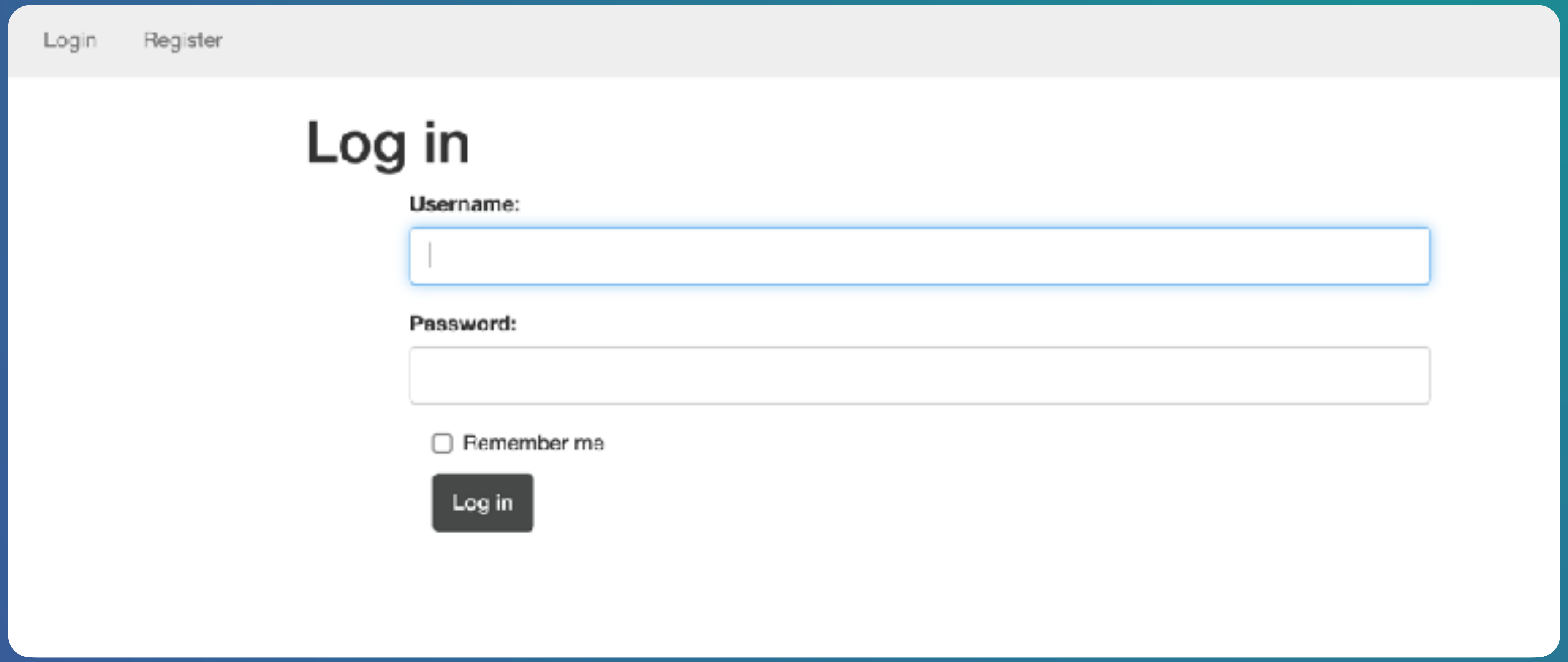
Register

Username:

Password:

Password (again):

Register


Login Register

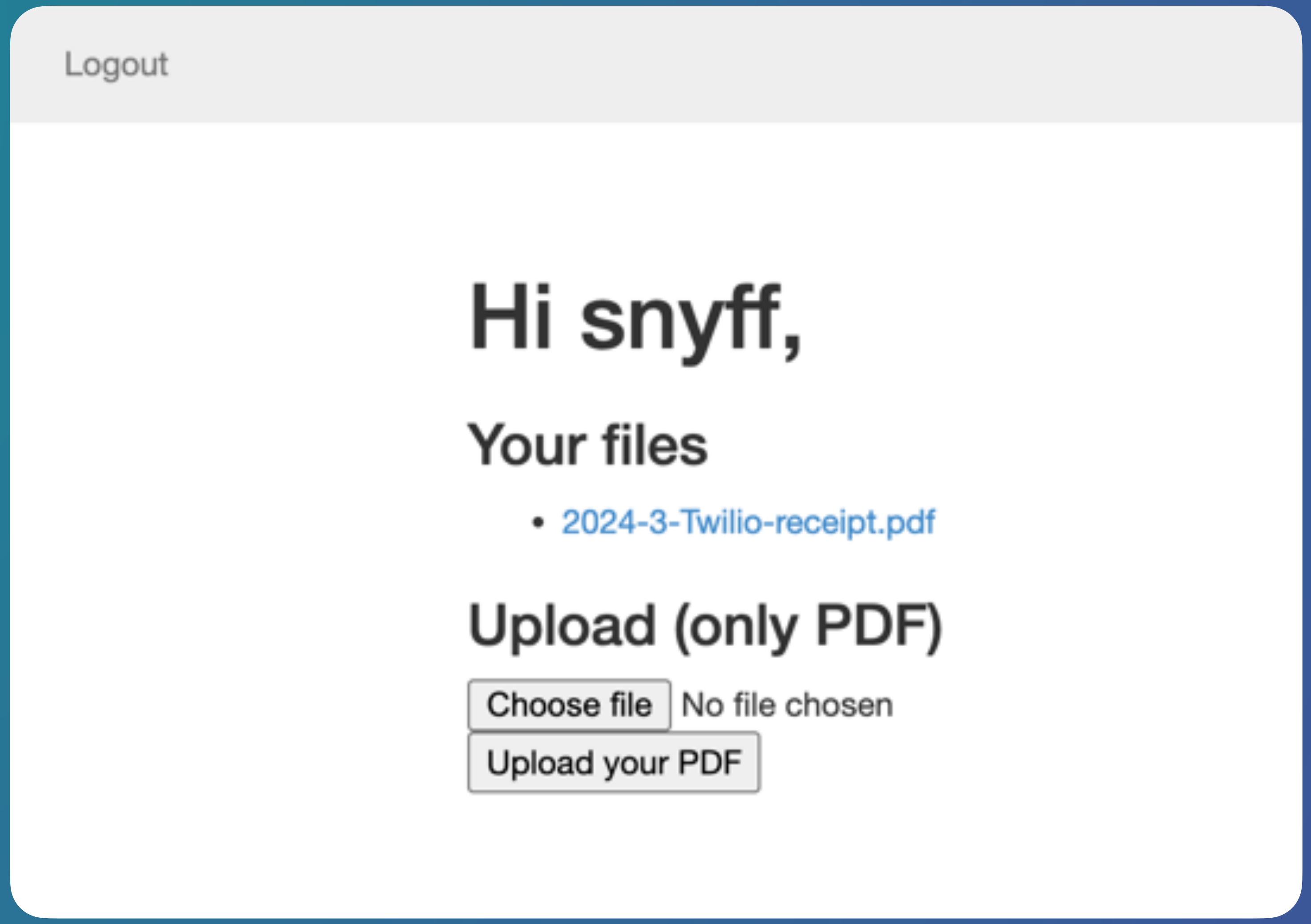
Log in

Username:

Password:

☐ Remember me

Log in


Logout

Hi snyff,

Your files

- 2024-3-Twilio-receipt.pdf

Upload (only PDF)

Choose file No file chosen

Upload your PDF