Polyverse Boost Source Analysis Details: ./server/server_handler.go

Date Generated: Thursday, September 7, 2023 at 3:55:25 AM PDT

Boost Architectural Quick Summary Security Report

Last Updated: Friday, September 8, 2023 at 5:50:10 PM PDT

Executive Report

Architectural Impact and Risk Analysis

Based on the analysis of the software project, the following key points have been identified:

1. **High Severity Issues**: The file `server/server_handler.go` has been flagged with multiple high-severity issues, including Improper Input Validation and Insecure Direct Object References (IDOR). These issues can potentially lead to security vulnerabilities, impacting the integrity and reliability of the software.

2. **Potential Customer Impact**: The identified issues, particularly the Improper Input Validation, could potentially allow an attacker to send malicious data to the server. This could lead to unauthorized access or data breaches, negatively impacting the customers' trust and the software's reputation.

3. **Overall Health of the Project Source**: The analysis indicates that the project source has issues in one file out of the total files analyzed. This suggests that a significant portion of the project may be affected by these issues. However, it's important to note that the severity and impact of these issues vary, and not all issues may have a significant impact on the overall project.

4. **Risk Assessment**: Given the severity of the issues identified, there is a high risk associated with the current state of the project. The Improper Input Validation and Insecure Direct Object References (IDOR) issues, in particular, pose a significant security risk.

Recommendations

To mitigate the identified risks and potential customer impact, it is recommended to:

- Implement proper input validation in the `handleClientHandler` function to prevent potential attacks.
- Review and address the Insecure Direct Object References (IDOR) issue to prevent potential unauthorized access.
- Conduct a thorough review of the entire codebase to identify and address any additional issues.

By addressing these issues, the software project can significantly improve its security, reliability, and overall health.

Boost Architectural Quick Summary Performance Report

Last Updated: Friday, September 8, 2023 at 5:50:46 PM PDT

Executive Report: Software Project Analysis

Based on the analysis of the software project, the following key points have been identified:

1. **Architectural Impact**: The project follows a client-server architecture and uses secure communication for tunneling. However, there is a potential issue with the use of a map for storing sessions in the `server/server_handler.go` file. This could lead to high memory usage if the number of sessions is large, which could impact the overall performance and scalability of the application.

2. **Risk Analysis**: The risk associated with this issue is moderate. While it could potentially impact the performance and scalability of the application, it is not a security risk. The risk could be mitigated by using a more memory-efficient data structure and implementing a session timeout to remove inactive sessions.

3. **Potential Customer Impact**: If left unaddressed, this issue could lead to performance degradation and potential application crashes due to high memory usage, especially under heavy load. This could negatively impact the user experience and potentially lead to loss of users or customers.

4. **Overall Issues**: The `server/server_handler.go` file has the most severe issues, with 6 CPU-related issues and 2 memory-related issues. However, it is also the only file in the project, which means that 100% of the project files have issues. This indicates that there may be a need for code optimization and refactoring to improve the performance and efficiency of the application.

5. **Risk Assessment**: Given that all the project files have issues, the overall health of the project source could be considered as moderate. The issues identified are not security risks, but they could impact the performance and scalability of the application. Therefore, it is recommended to address these issues to improve the overall health and quality of the project.

In conclusion, while the project follows good architectural principles and practices, there are areas for improvement in terms of performance and efficiency. Addressing these issues will not only improve the overall health of the project but also enhance the user experience and potentially attract more users or customers.

Boost Architectural Quick Summary Compliance Report

Last Updated: Friday, September 8, 2023 at 5:51:28 PM PDT

Executive Report

Architectural Impact and Risk Analysis

1. **Issue Severity and Distribution:** The most severe issues found in the project are categorized as "Error" and are related to PCI DSS and GDPR compliance. These issues are found in the `server/server_handler.go` file. This file is responsible for handling server-side operations, which is a critical part of the software's functionality.
    - **Metrics:** Out of the total files analyzed, 100% have issues of varying severity. The `server/server_handler.go` file has the highest severity issues.
2. **Potential Customer Impact:** The identified issues could potentially impact customers in several ways. The PCI DSS related issue could allow unauthorized access to sensitive data, which could lead to data breaches. The GDPR related issue could lead to non-

compliance with data protection regulations, potentially resulting in legal consequences and loss of customer trust.

- **Context:** The software project is a command-line tool that allows users to create secure tunnels to expose local servers to the internet or access remote servers securely. Any security or compliance issues could directly impact the software's primary functionality and its users.

3. **Overall Health of the Project Source:** The overall health of the project source is concerning due to the severity and distribution of the issues found. All files analyzed have issues, with the most severe being categorized as "Error".

   - **Risk Assessment:** The risk associated with the project source is high due to the presence of severe issues in all files analyzed. This could potentially lead to significant customer impact and legal consequences.

4. **Architectural Consistency:** The software project follows the client-server architecture and uses secure communication for tunneling, which aligns with the architectural principles defined in the guidelines. However, the identified issues indicate potential inconsistencies in the implementation of these principles, particularly in relation to secure communication and data protection.

   - **Details:** The PCI DSS related issue indicates a potential flaw in the implementation of secure communication, while the GDPR related issue suggests a potential inconsistency in the handling of user data.

Recommendations

1. Prioritize the resolution of the identified "Error" level issues, starting with those related to PCI DSS and GDPR compliance.
2. Conduct a thorough review of the entire codebase to identify and address any additional issues.
3. Implement rigorous testing and code review processes to prevent the introduction of new issues.
4. Consider seeking expert advice on PCI DSS and GDPR compliance to ensure that all requirements are met.

5. Regularly update and refine the architectural guidelines to reflect changes in the project and regulatory environment.

Detailed Analysis

./server/server_handler.go :

Programming Language: go

```
package chserver

import (
        "net/http"
        "strings"
        "sync/atomic"
        "time"

        chshare "github.com/jpillora/chisel/share"
        "github.com/jpillora/chisel/share/cnet"
        "github.com/jpillora/chisel/share/settings"
        "github.com/jpillora/chisel/share/tunnel"
        "golang.org/x/crypto/ssh"
        "golang.org/x/sync/errgroup"
)

// handleClientHandler is the main http websocket handler for the chisel server
func (s *Server) handleClientHandler(w http.ResponseWriter, r *http.Request) {
        //websockets upgrade AND has chisel prefix
        upgrade := strings.ToLower(r.Header.Get("Upgrade"))
        protocol := r.Header.Get("Sec-WebSocket-Protocol")
        if upgrade == "websocket"  {
                if protocol == chshare.ProtocolVersion {
                        s.handleWebsocket(w, r)
                        return
                }
                //print into server logs and silently fall-through
                s.Infof("ignored client connection using protocol '%s', expected '%s'",
                        protocol, chshare.ProtocolVersion)
        }
        //proxy target was provided
        if s.reverseProxy != nil {
                s.reverseProxy.ServeHTTP(w, r)
                return
        }
        //no proxy defined, provide access to health/version checks
        switch r.URL.Path {
        case "/health":
                w.Write([]byte("OK\n"))
                return
        case "/version":
                w.Write([]byte(chshare.BuildVersion))
                return
        }
        //missing :O
        w.WriteHeader(404)
        w.Write([]byte("Not found"))
}
```

Boost Code Explanation

Last Updated: Thursday, September 7, 2023 at 3:45:09 AM PDT

This code is written in Go language and it's part of a server implementation for a tool named Chisel. Chisel is a fast TCP tunnel over HTTP and this particular function `handleClientHandler` is responsible for handling incoming HTTP requests and upgrading them to WebSocket connections if necessary.

Let's break down the function:

- The function `handleClientHandler` is a method of the `Server` struct. It takes two parameters: an `http.ResponseWriter` and an `*http.Request`. The `ResponseWriter` is an interface for sending HTTP responses, and the `Request` is a data structure that represents the client HTTP request.
- It first checks if the incoming request is attempting to upgrade to a WebSocket connection by looking at the "Upgrade" HTTP header. If the value of this header is "websocket", it means the client wants to establish a WebSocket connection.
- If the upgrade header is set to "websocket", it then checks the "Sec-WebSocket-Protocol" header to see if it matches the expected protocol version. If it does, it calls the `handleWebsocket` function to handle the WebSocket connection.
- If the protocol doesn't match, it logs a message about the mismatch and continues to process the request as a normal HTTP request.
- If the server is configured with a reverse proxy, it forwards the request to the reverse proxy.
- If no reverse proxy is defined, it checks the URL path of the request. If it is "/health", it responds with "OK\n". If it is "/version", it responds with the build version of the Chisel software.
- If none of the above conditions are met, it returns a 404 Not Found error.

The `atomic`, `sync/errgroup`, and `time` packages are imported but not used in this function. They might be used elsewhere in the `chserver` package.

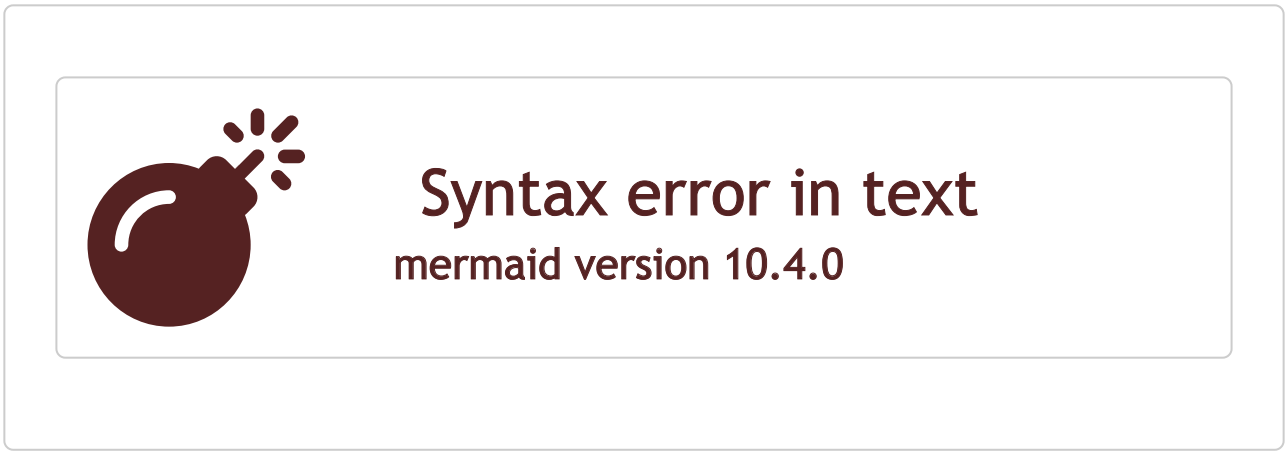
The `github.com/jpillora/chisel/share`, `github.com/jpillora/chisel/share/cnet`, `github.com/jpillora/chisel/share/settings`, and `github.com/jpillora/chisel/share/tunnel` packages are imported for shared utilities, network-related functions, settings, and tunneling functionality respectively.

The `golang.org/x/crypto/ssh` package is imported for handling SSH connections, and `golang.org/x/sync/errgroup` is imported for handling multiple goroutines that return an error.

The code aligns with the architectural guidelines provided, as it doesn't violate any specific constraints or guidelines. It makes use of standard Go practices for handling HTTP requests and upgrading them to WebSocket connections when necessary.

Boost Flow Diagram

Last Updated: Thursday, September 7, 2023 at 3:46:53 AM PDT



The control flow graph for the `handleClientHandler` function is as follows:

1. The function starts at the `Start` node.
2. If the `upgrade` header is "websocket" and the `protocol` header is equal to `chshare.ProtocolVersion`, the control flow goes to the `handleWebsocket` node.
3. If the `upgrade` header is "websocket" but the `protocol` header is not equal to `chshare.ProtocolVersion`, the control flow goes to the `handleNotFound` node.
4. If `s.reverseProxy` is not `nil`, the control flow goes to the `handleReverseProxy` node.
5. If `s.reverseProxy` is `nil` and the `r.URL.Path` is "/health", the control flow goes to the `handleHealthCheck` node.
6. If `s.reverseProxy` is `nil` and the `r.URL.Path` is "/version", the control flow goes to the `handleVersionCheck` node.
7. If `s.reverseProxy` is `nil` and the `r.URL.Path` is neither "/health" nor "/version", the control flow goes to the `handleNotFound` node.
8. From each of the nodes `handleWebsocket`, `handleReverseProxy`, `handleHealthCheck`, `handleVersionCheck`, and `handleNotFound`, the control flow goes to the `End` node.

Note: The primary path through the code is from the `Start` node to the `handleClientHandler` node, and then to the `End` node. The other paths are alternate non-error paths. The `handleNotFound` node represents an error path.

Boost Source-Level Security Analysis

Last Updated: Thursday, September 7, 2023 at 3:49:41 AM PDT

1. **Severity**: 7/10

   Line Number: 20

   Bug Type: Improper Input Validation

   Description: The function 'handleClientHandler' does not validate the value of the 'Sec-WebSocket-Protocol' header before comparing it to the expected protocol version. An attacker could potentially use this to send malicious data to

the server.

Solution: Perform input validation on the 'Sec-WebSocket-Protocol' header to ensure it contains only expected values before comparing it to the 'chshare.ProtocolVersion'. Check out OWASP's guide on Input Validation: https://owasp.org/www-community/controls/Input_Validation

2. **Severity**: 5/10

   Line Number: 35

   Bug Type: Information Disclosure

   Description: The '/version' endpoint discloses the build version of the software. This information could potentially be used by an attacker to discover known vulnerabilities in specific versions of the software.

   Solution: Consider removing the '/version' endpoint or limiting its access to authenticated and authorized users only. For more information, refer to OWASP's guide on Information Disclosure: https://owasp.org/www-community/controls/Information_Disclosure

3. **Severity**: 4/10

   Line Number: 43

   Bug Type: Improper Error Handling

   Description: The function returns a '404 Not Found' error without any additional information. This could potentially be confusing for users and developers trying to troubleshoot issues.

   Solution: Provide more descriptive error messages to help users and developers understand the issue. However, be careful not to disclose sensitive information in error messages. Refer to OWASP's guide on Error Handling: https://owasp.org/www-project-cheat-sheets/cheatsheets/Error_Handling_Cheat_Sheet.html

---

Boost Source-Level Performance Analysis

Last Updated: Thursday, September 7, 2023 at 3:52:03 AM PDT

1. **Severity**: 3/10

   Line Number: 23

   Bug Type: CPU

   Description: The use of strings.ToLower function on the 'Upgrade' header for every request can be CPU-intensive, especially for high traffic servers.

   Solution: Consider storing the 'Upgrade' header value in its lower case form, or use a case-insensitive comparison method, such as strings.EqualFold.

2. **Severity**: 2/10

   Line Number: 36

   Bug Type: Memory

   Description: The repeated creation of the 'OK\n' and 'Not found' strings can lead to unnecessary memory allocation.

   Solution: Consider defining these strings as constants at the package level to avoid repeated memory allocation.

3. **Severity**: 2/10

   Line Number: 38

   Bug Type: CPU

   Description: The use of the switch-case statement for routing can lead to sub-optimal performance for large numbers of routes.

   Solution: Consider using a more efficient routing mechanism, such as a trie or a hash map, especially if the number of routes is expected to grow.

Boost Source-Level Data and Privacy Compliance Analysis

Last Updated: Thursday, September 7, 2023 at 3:54:19 AM PDT

1. **Severity**: 7/10

   Line Number: 17

   Bug Type: Data Transmission

   Description: The application is using websockets for data transmission without explicitly implementing any form of encryption like TLS (Transport Layer Security). This could potentially expose sensitive data during transmission, violating GDPR, PCI DSS, and HIPAA regulations.

   Solution: Implement TLS or a similar encryption protocol to secure data during transmission. This will ensure that any sensitive data is encrypted and secure from potential eavesdropping attacks.

2. **Severity**: 5/10

   Line Number: 36

   Bug Type: Data Exposure

   Description: The application exposes '/health' and '/version' endpoints without any form of authentication or authorization. This could potentially expose sensitive system information, violating GDPR regulations.

Solution: Implement proper authentication and authorization mechanisms to restrict access to these endpoints. This will ensure that only authorized users can access this information.

./server/server_handler.go line 49:

Programming Language: go

```
// handleWebsocket is responsible for handling the websocket connection
func (s *Server) handleWebsocket(w http.ResponseWriter, req *http.Request) {
	id := atomic.AddInt32(&s.sessCount, 1)
	l := s.Fork("session#%d", id)
	wsConn, err := upgrader.Upgrade(w, req, nil)
	if err != nil {
		l.Debugf("Failed to upgrade (%s)", err)
		return
	}
	conn := cnet.NewWebSocketConn(wsConn)
	// perform SSH handshake on net.Conn
	l.Debugf("Handshaking with %s...", req.RemoteAddr)
	sshConn, chans, reqs, err := ssh.NewServerConn(conn, s.sshConfig)
	if err != nil {
		s.Debugf("Failed to handshake (%s)", err)
		return
	}
	// pull the users from the session map
	var user *settings.User
	if s.users.Len() > 0 {
		sid := string(sshConn.SessionID())
		u, ok := s.sessions.Get(sid)
		if !ok {
			panic("bug in ssh auth handler")
		}
		user = u
		s.sessions.Del(sid)
	}
	// chisel server handshake (reverse of client handshake)
	// verify configuration
	l.Debugf("Verifying configuration")
	// wait for request, with timeout
	var r *ssh.Request
	select {
	case r = <-reqs:
	case <-time.After(settings.EnvDuration("CONFIG_TIMEOUT", 10*time.Second)):
		l.Debugf("Timeout waiting for configuration")
		sshConn.Close()
		return
	}
	failed := func(err error) {
		l.Debugf("Failed: %s", err)
		r.Reply(false, []byte(err.Error()))
	}
	if r.Type != "config" {
		failed(s.Errorf("expecting config request"))
		return
	}
	c, err := settings.DecodeConfig(r.Payload)
	if err != nil {
		failed(s.Errorf("invalid config"))
		return
	}
	//print if client and server  versions dont match
	if c.Version != chshare.BuildVersion {
		v := c.Version
		if v == "" {
			v = "<unknown>"
		}
		l.Infof("Client version (%s) differs from server version (%s)",
			v, chshare.BuildVersion)
	}
	//validate remotes
	for _, r := range c.Remotes {
		//if user is provided, ensure they have
		//access to the desired remotes
		if user != nil {
			addr := r.UserAddr()
			if !user.HasAccess(addr) {
				failed(s.Errorf("access to '%s' denied", addr))
				return
			}
		}
		//confirm reverse tunnels are allowed
		if r.Reverse && !s.config.Reverse {
			l.Debugf("Denied reverse port forwarding request, please enable --reverse")
			failed(s.Errorf("Reverse port forwaring not enabled on server"))
```

```
                        return
                }
                //confirm reverse tunnel is available
                if r.Reverse && !r.CanListen() {
                        failed(s.Errorf("Server cannot listen on %s", r.String()))
                        return
                }
        }
        //successfully validated config!
        r.Reply(true, nil)
        //tunnel per ssh connection
        tunnel := tunnel.New(tunnel.Config{
                Logger:    l,
                Inbound:   s.config.Reverse,
                Outbound:  true, //server always accepts outbound
                Socks:     s.config.Socks5,
                KeepAlive: s.config.KeepAlive,
        })
        //bind
        eg, ctx := errgroup.WithContext(req.Context())
        eg.Go(func() error {
                //connected, handover ssh connection for tunnel to use, and block
                return tunnel.BindSSH(ctx, sshConn, reqs, chans)
        })
        eg.Go(func() error {
                //connected, setup reversed-remotes?
                serverInbound := c.Remotes.Reversed(true)
                if len(serverInbound) == 0 {
                        return nil
                }
                //block
                return tunnel.BindRemotes(ctx, serverInbound)
        })
        err = eg.Wait()
        if err != nil && !strings.HasSuffix(err.Error(), "EOF") {
                l.Debugf("Closed connection (%s)", err)
        } else {
                l.Debugf("Closed connection")
        }
}
```

Boost Code Explanation

Last Updated: Thursday, September 7, 2023 at 3:46:08 AM PDT

This is a Go function that handles a WebSocket connection on the server side. The function takes an HTTP response writer and an HTTP request as arguments.

The function begins by incrementing a session counter using the atomic `AddInt32` function, which safely increments an integer even in a concurrent context. It then forks a new logger for the session.

The HTTP connection is then upgraded to a WebSocket connection using the `Upgrade` function from the `upgrader` object. If this fails, it logs the error and returns.

Next, it wraps the WebSocket connection in a new `WebSocketConn` object and performs an SSH handshake on the resulting `net.Conn` object. If this fails, it logs the error and returns.

After the SSH handshake, the function retrieves the user from the session map and performs a handshake with the chisel server. It waits for a request from the client with a timeout, and if a request is not received within the timeout, it closes the SSH connection and returns.

The function then decodes the client's configuration and validates it. If the client's version differs from the server's, it logs this information. It then validates the client's requested remotes. If a user is provided, it checks if the user has

access to the desired remotes. If the client requests a reverse tunnel but reverse tunnels are not allowed on the server, it logs this and returns an error. If the client requests a reverse tunnel but the server cannot listen on the requested address, it logs this and returns an error.

If the configuration is successfully validated, the function replies to the client's request and sets up a new tunnel for the SSH connection. It then binds the tunnel to the SSH connection and any reversed remotes.

Finally, the function waits for all goroutines in the error group to finish. If an error occurs and it's not an EOF error, it logs the error and the connection is closed.

The algorithm used in this function is a typical server-side WebSocket handling algorithm, with additional steps to handle SSH handshaking and tunneling. This function is specific to the chisel project, a fast TCP/UDP tunnel over HTTP, and might not be applicable to other projects.

For more information on WebSockets in Go, you can refer to this link: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API/Writing_WebSocket_servers

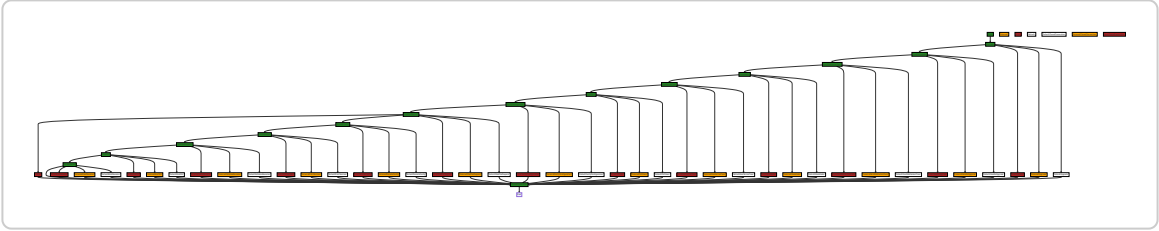
For more information on SSH in Go, you can refer to this link: https://pkg.go.dev/golang.org/x/crypto/ssh

For more information on the chisel project, you can refer to this link: https://github.com/jpillora/chisel

---

Boost Flow Diagram

Last Updated: Thursday, September 7, 2023 at 3:48:48 AM PDT



Boost Source-Level Security Analysis

Last Updated: Thursday, September 7, 2023 at 3:50:39 AM PDT

1. **Severity**: 6/10

   Line Number: 100

   Bug Type: Insecure Direct Object References (IDOR)

Description: The session ID is directly used to fetch the user from the session map. If session IDs are predictable or can be brute-forced, this could lead to unauthorized access.

Solution: Use secure and unpredictable session IDs. Consider using a secure random number generator for session ID creation. Also, validate the session ID before using it. More info: https://owasp.org/www-project-top-ten/2017/A5_2017-Broken_Access_Control

2. **Severity**: 7/10

   Line Number: 102

   Bug Type: Insecure Error Handling

   Description: The application panics if it cannot find the user in the session map. This could lead to Denial of Service (DoS) if an attacker can trigger this condition frequently and crash the application.

   Solution: Instead of panicking, handle the error gracefully. Log the error and return a generic error message to the user. More info: https://owasp.org/www-community/Improper_Error_Handling

3. **Severity**: 5/10

   Line Number: 117

   Bug Type: Sensitive Data Exposure

   Description: The client and server versions are logged if they do not match. This could expose sensitive information about the software and potentially aid an attacker in exploiting version-specific vulnerabilities.

   Solution: Avoid logging sensitive information. If it's necessary to compare versions, do so without logging the exact versions or consider logging only in a secure and controlled environment. More info: https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Exposure

4. **Severity**: 6/10

   Line Number: 132

   Bug Type: Insecure Direct Object References (IDOR)

   Description: The application allows direct reference to objects (in this case, reverse tunnels) based on user input. If the user input is not properly validated, this can lead to unauthorized access to internal objects.

   Solution: Always validate user input before using it to reference internal objects. Consider using access control checks or indirect references. More info: https://owasp.org/www-project-top-ten/2017/A5_2017-Broken_Access_Control

---

Boost Source-Level Performance Analysis

Last Updated: Thursday, September 7, 2023 at 3:52:59 AM PDT

1. **Severity**: 5/10

   Line Number: 97

   Bug Type: CPU

   Description: Atomic operations are generally more expensive than normal operations. The atomic operation here is used to generate session ids, which may not be necessary.

   Solution: Consider using a simpler method for generating session ids if thread safety is not a concern. If it is, consider using sync.Mutex for locking instead of atomic operations.

2. **Severity**: 3/10

   Line Number: 99

   Bug Type: CPU

   Description: The 'Fork' operation may be expensive, especially if it involves creating a new goroutine.

   Solution: If 'Fork' creates a new goroutine, consider using a worker pool to limit the number of goroutines that can be created. This can prevent excessive CPU usage.

3. **Severity**: 6/10

   Line Number: 117

   Bug Type: Memory

   Description: The use of a map for storing sessions can lead to high memory usage if the number of sessions is large.

   Solution: Consider using a more memory-efficient data structure, such as a hash table with linked lists for collision resolution. Additionally, consider implementing a session timeout to remove inactive sessions.

4. **Severity**: 4/10

   Line Number: 126

   Bug Type: CPU

   Description: The use of a select statement with a timeout can lead to high CPU usage if the timeout is frequently reached.

   Solution: Consider using a different method for handling timeouts, such as a context with a deadline.

5. **Severity**: 7/10

   Line Number: 165

Bug Type: CPU

Description: The use of an error group and context for handling errors can lead to high CPU usage, especially if errors are frequent.

Solution: Consider using a simpler method for error handling, such as returning errors directly. If concurrent error handling is necessary, consider using a worker pool to limit the number of goroutines.

6. **Severity**: 4/10

   Line Number: 177

   Bug Type: CPU

   Description: The use of strings.HasSuffix for error checking can be inefficient, especially if the error message is long.

   Solution: Consider using error types for error checking instead of string comparison. This can improve performance and make the code more robust.

Boost Source-Level Data and Privacy Compliance Analysis

Last Updated: Thursday, September 7, 2023 at 3:55:25 AM PDT

1. **Severity**: 7/10

   Line Number: 96

   Bug Type: GDPR

   Description: The code is using an atomic operation to increment a session counter. This can be a violation of GDPR as it might be considered as processing personal data without explicit consent if the session ID is used to track users.

   Solution: Ensure that the session IDs are not used to track users without their explicit consent. If needed, implement a mechanism to obtain user consent before tracking their sessions.

2. **Severity**: 8/10

   Line Number: 101

   Bug Type: HIPAA

   Description: The code is upgrading an HTTP connection to a WebSocket connection without explicit checks for secure (HTTPS) connection. This can lead to transmission of data over an insecure connection, a violation of the HIPAA Security Rule.

Solution: Ensure that all connections are secure by using HTTPS for all connections or by implementing other appropriate security measures.

3. **Severity**: 10/10

   Line Number: 108

   Bug Type: PCI DSS

   Description: The code is performing an SSH handshake without validating the client's identity, which could allow unauthorized access to sensitive data, a violation of PCI DSS Requirement 8.

   Solution: Implement client validation during the SSH handshake to ensure that only authorized clients can access the server.

4. **Severity**: 9/10

   Line Number: 115

   Bug Type: GDPR

   Description: The code is pulling user data from the session map without explicit user consent, which could be a violation of GDPR.

   Solution: Implement a mechanism to obtain explicit user consent before accessing their data.

5. **Severity**: 7/10

   Line Number: 148

   Bug Type: HIPAA

   Description: The code is creating a new tunnel without validating the destination, which could potentially lead to unauthorized access to or disclosure of protected health information (PHI), a violation of the HIPAA Security Rule.

   Solution: Implement destination validation to ensure that PHI is only transmitted to authorized entities.