

[New issue](#)[Jump to bottom](#)

# Sending 2000, 4000 or 6000 characters to Discord panics in util.PartitionMessage (index out of range) #240

✓ Closed justinsteven opened this issue on May 21 · 0 comments · Fixed by #242

justinsteven commented on May 21

Contributor

## What happens

Sending exactly 2000, 4000 or 6000 characters to Discord panics

## What should happen

Sending any number of characters to Discord should succeed

## Demo

```
package main

import (
    "fmt"
    t "github.com/containrrr/shoutrrr/pkg/types"
    "strings"
)

// BEGIN taken from shoutrrr 0.5.3 util and discord packages

var limits = t.MessageLimit{
    ChunkSize:      2000,
    TotalChunkSize: 6000,
    ChunkCount:     10,
}

const maxSearchRunes = 100

// Min returns the smallest of a and b
```

```

func Min(a int, b int) int {
    if a < b {
        return a
    }
    return b
}

// PartitionMessage splits a string into chunks that is at most chunkSize runes, it will search the 1
// for a whitespace to make the split appear nicer. It will keep adding chunks until it reaches maxCo
// the total amount of runes in the chunks reach maxTotal.
// The chunks are returned together with the number of omitted runes (that did not fit into the chunk
func PartitionMessage(input string, limits t.MessageLimit, distance int) (items []t.MessageItem, omit
    runes := []rune(input)
    chunkOffset := 0
    maxTotal := Min(len(runes), limits.TotalChunkSize)
    maxCount := limits.ChunkCount - 1

    for i := 0; i < maxCount; i++ {
        // If no suitable split point is found, use the chunkSize
        chunkEnd := chunkOffset + limits.ChunkSize
        // ... and start next chunk directly after this one
        nextChunkStart := chunkEnd
        if chunkEnd > maxTotal {
            // The chunk is smaller than the limit, no need to search
            chunkEnd = maxTotal
            nextChunkStart = maxTotal
        } else {
            for r := 0; r < distance; r++ {
                rp := chunkEnd - r
                if runes[rp] == '\n' || runes[rp] == ' ' {
                    // Suitable split point found
                    chunkEnd = rp
                    // Since the split is on a whitespace, skip it in the next ch
                    nextChunkStart = chunkEnd + 1
                    break
                }
            }
        }

        items = append(items, t.MessageItem{
            Text: string(runes[chunkOffset:chunkEnd]),
        })

        chunkOffset = nextChunkStart
        if chunkOffset >= maxTotal {
            break
        }
    }

    return items, len(runes) - chunkOffset
}

// END taken from shoutrrr 0.5.3

func fuzz(length int) {
    defer func() {

```

```

        if r := recover(); r != nil {
            fmt.Printf("Recovered panic when partitioning message of length %d: %s\n", le
        }
    }()
    PartitionMessage(strings.Repeat("A", length), limits, maxSearchRunes)
}

func main() {
    for i := 0; i <= 20000; i++ {
        fuzz(i)
    }
}

```

```

% go run shoutrrr_partitionmessage_fuzz.go
Recovered panic when partitioning message of length 2000: runtime error: index out of range [2000]
with length 2000
Recovered panic when partitioning message of length 4000: runtime error: index out of range [4000]
with length 4000
Recovered panic when partitioning message of length 6000: runtime error: index out of range [6000]
with length 6000

```

## Notes

There is also a report of `util.PartitionMessage` panicking with what seems to be a message of 3990 characters - see [projectdiscovery/notify#130 \(review\)](#). I haven't been able to reproduce this crash. It may or may not be related.

This is a potential DoS vulnerability. If an attacker can cause a consumer of shoutrrr to attempt to send a Discord message of a precise length, the consumer will panic, rendering the service unavailable. Without a published [security policy](#) I don't have a way of discretely reporting this. May I suggest you publish a policy :)

  justinsteven mentioned this issue on May 21

**Allow -bulk to work with stdin** [projectdiscovery/notify#130](#)

 Merged

  piksel mentioned this issue on May 21

**discord message size fixes** [#242](#)

 Merged

 piksel closed this as completed in [#242](#) on May 21

---

  justinsteven mentioned this issue on May 21

**Sending 1999, 3999 or 5999 characters to Discord panics in util.PartitionMessage (index out of range) #244**

 Closed

  piksel mentioned this issue on May 22

**fix: text partitioning logic #245**

 Merged

 justinsteven added a commit to justinsteven/notify that referenced this issue on May 22

 Bump shoutrrr for long message fixes ...

5ac3908

 ehsandeep added a commit to projectdiscovery/notify that referenced this issue on May 30

 Allow -bulk to work with stdin ([#130](#)) ...

✓ 74fff6e

  GoVulnBot mentioned this issue on Jul 15

**x/vulndb: potential Go vuln in github.com/containrrr/shoutrrr: CVE-2022-25891**  
golang/vulndb#528

 Closed

  jba mentioned this issue on Jul 15

**x/vulndb: potential Go vuln in github.com/containrrr/shoutrrr: CVE-2022-25891** jba/nested-modules#380

 Open

  AdamKorcz mentioned this issue on Jul 25

**pkg/util: Add fuzzer and ClusterfuzzLite #262**

 Open

No one assigned

---

Labels

None yet

---

Projects

None yet

---


Milestone

No milestone

---

Development

Successfully merging a pull request may close this issue.

 discord message size fixes  
containrrr/shoutrrr

---

1 participant

