memcached analysis

Rafilx

2022-06-19

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

## Loading required package: gridExtra

##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine

## Loading required package: viridisLite

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

R Markdown

Na dissertação, foi relatado que

- 3,1% das requisições de Memcached eram variações de `stats`
- 91,6% das requisições eram `set` ou `get`
- 5,1% das requisições eram malformadas (e.g., requisições HTTP ou SSDP) ou eram `flush_all` (para limpar chaves do cache)

É possível fazer essa análise por trimestre, ou ao menos analisar a incidência de `stats` e `set`+`get`

```
db <- dbConnect(RSQLite::SQLite(), dbname="../db/database-2022-05-11/mix_protocol.sqlite")


data_unfetch <-dbSendQuery(db, "
  SELECT ip, requests_per_attack, tempo_inicio, tempo_final, QUOTE(payload) as quote_payload, QUOTE(get_
         CAST(CAST(year AS text) || CAST(period AS text) as integer) as year_period, SUBSTR(payload,0,25
    FROM (
      SELECT *,  strftime(\"%Y\", tempo_inicio) as year, ((strftime(\"%m\", tempo_final) - 1) / 3) + 1 A
        FROM MEMCACHED_ANALYSIS
    )
")

data <- fetch(data_unfetch)

data_memcached_payload_types_unfetch <-dbSendQuery(db, "
  SELECT id, quantity, SUBSTR(payload,0,25) AS payload_limit
    FROM MEMCACHED_PAYLOAD_TYPES
")
```

```
## Warning: Closing open result set, pending rows
```

```
data_memcached_payload_types <- fetch(data_memcached_payload_types_unfetch)


dbDisconnect(db)
```

```
## Warning in connection_release(conn@ptr): There are 1 result in use. The
## connection will be released when they are closed
```

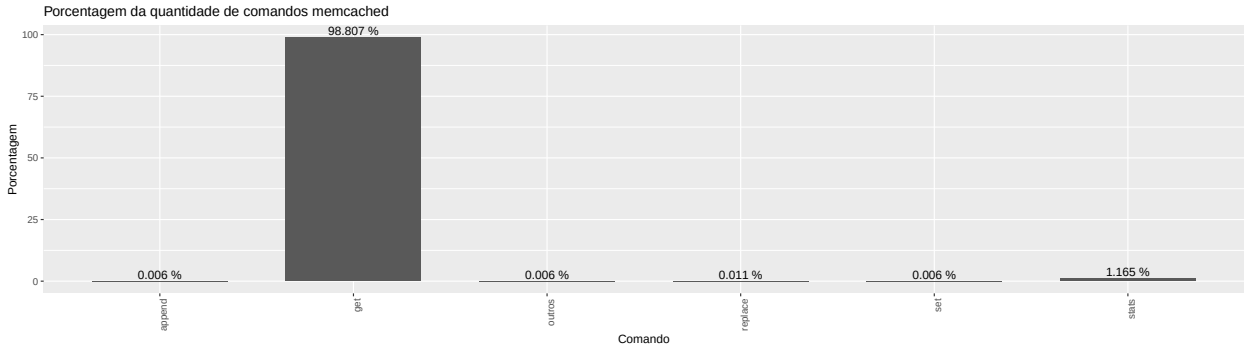
```
data['tempo_final_cast'] = as.POSIXct(data[['tempo_final']], format = "%Y-%m-%d %H:%M:%S")
data['tempo_inicio_cast'] = as.POSIXct(data[['tempo_inicio']], format = "%Y-%m-%d %H:%M:%S")
```

```
memcached_payload_types = data_memcached_payload_types %>%
  mutate(payload_str = toString(payload_limit)) %>%
  arrange(desc(quantity)) %>%
  select('quantity', 'payload_limit', 'id')

memcached_payload_types_quantity_percentage = memcached_payload_types %>%
  mutate(sum_quantity = sum(quantity)) %>%
  mutate(quantity_percentage = (quantity / sum_quantity) * 100)

memcached_payload_types_quantity_percentage %>%
  select('quantity', 'quantity_percentage', 'payload_limit') %>%
  arrange(desc(quantity)) %>%
  head(15)
```

```
##    quantity quantity_percentage payload_limit
## 1     17719           98.806669           get
## 2       209            1.165449         stats
## 3         2            0.011153       replace
## 4         1            0.005576           set
```

```
## 5          1             0.005576        append
## 6          1             0.005576        outros
## 7          0             0.000000           add
## 8          0             0.000000           cas
## 9          0             0.000000       prepend
## 10         0             0.000000     flush_all
```

```
memcached_payload_types_quantity_percentage %>%
 arrange(desc(quantity)) %>%
 filter(quantity > 0) %>%
 select('quantity_percentage', 'payload_limit') %>%
 ggplot( aes(x=payload_limit, y=quantity_percentage)) +
   geom_bar(stat="identity", width = 0.7, position="dodge") +
   geom_text(aes(label = paste(round(quantity_percentage, 3), "%"),  vjust = -0.25)) +
   scale_fill_viridis(discrete=TRUE, direction = -1) +
   theme(axis.text.x = element_text(angle = 90, vjust = 1, hjust=1)) +
   ylab("Porcentagem") +
   xlab("Comando") +
   ggtitle("Porcentagem da quantidade de comandos memcached")
```



- Agrupamento realizado por período (trimestre) e "memcached_request_type" é o comando utilizado no ataque `["stats", "set", "get", "add", "cas", "replace", "append", "prepend", "flush_all", "outros"]`
- Somando a quantidade de requisições utilizadas por cada comando e período

```
data_grouped_period_command = data %>%
  mutate(year_period_int = year_period,
         year_period = as.factor(year_period),
         command = as.factor(memcached_request_type)) %>%
  group_by(year_period, command) %>%
  summarise(sum_requests_per_attack = sum(requests_per_attack),
            number_of_attacks = n(),
            tempo_inicio=min(tempo_inicio_cast),
            tempo_final=max(tempo_final_cast))
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

```
data_grouped_period_command_percentage = data_grouped_period_command %>%
  ungroup() %>%
  group_by(year_period) %>%
  summarise(command = command,
            number_of_attacks = number_of_attacks,
            tempo_inicio = tempo_inicio,
            tempo_final = tempo_final,
            sum_period_number_of_attacks = sum(number_of_attacks),
            sum_period_requests_per_attack = sum(sum_requests_per_attack),
            sum_requests_per_attack = sum_requests_per_attack) %>%
  mutate(number_of_attacks_percentage = (number_of_attacks / sum_period_number_of_attacks) * 100,
         number_of_requests_percentage = (sum_requests_per_attack / sum_period_requests_per_attack) * 1
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

```
minimum_percentage_as_others = 1
decimals_digits = 1

data_grouped_period_command_others_percentage = data_grouped_period_command_percentage %>%
  mutate(
    command = case_when(
      number_of_requests_percentage < minimum_percentage_as_others ~ "OUTROS",
      TRUE ~ as.character(command)
    )
  ) %>%
  group_by(year_period, command) %>%
  summarise(number_of_requests_percentage = sum(number_of_requests_percentage))
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

```
data_grouped_period_command_others_percentage %>%
  print(n=16)
```

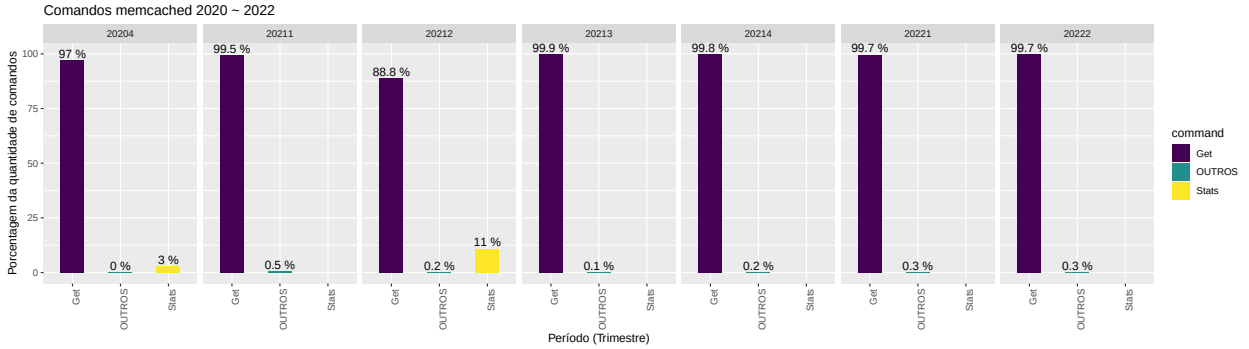
```
## # A tibble: 16 x 3
## # Groups:   year_period [7]
##    year_period command number_of_requests_percentage
##    <fct>       <chr>                           <dbl>
##  1 20204       Get                          97.0
##  2 20204       OUTROS                        0.00000656
##  3 20204       Stats                         3.00
##  4 20211       Get                          99.5
##  5 20211       OUTROS                        0.466
##  6 20212       Get                          88.8
##  7 20212       OUTROS                        0.220
##  8 20212       Stats                        11.0
##  9 20213       Get                          99.9
## 10 20213       OUTROS                        0.0763
## 11 20214       Get                          99.8
## 12 20214       OUTROS                        0.211
## 13 20221       Get                          99.7
```

```
## 14 20221        OUTROS                    0.308
## 15 20222        Get                      99.7
## 16 20222        OUTROS                    0.290
```

Porcentagem menores que 1 foram agrupadas como “OUTROS”

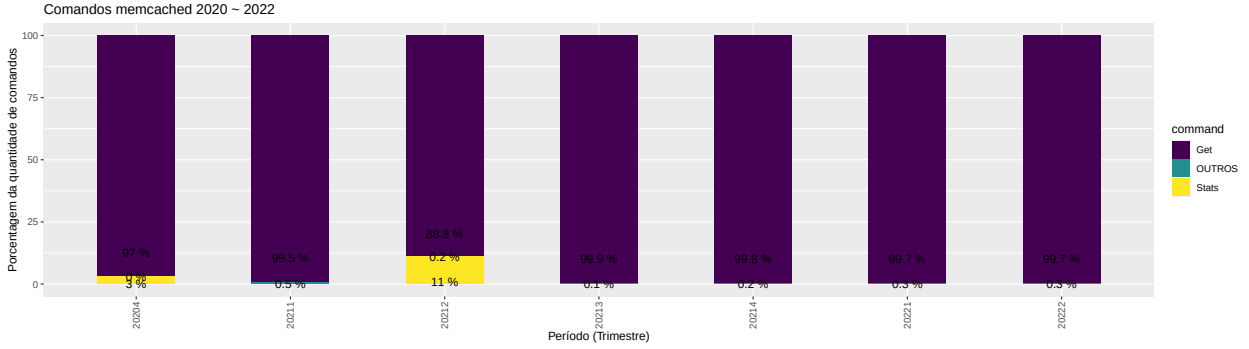
- Gráfico de barras 2020 ~ 2022

```
data_grouped_period_command_others_percentage %>%
  ggplot( aes(x=command, y=number_of_requests_percentage, fill=command)) +
    geom_bar(stat="identity", width = 0.5, position="dodge") +
    geom_text(aes(label = paste(round(number_of_requests_percentage, decimals_digits), "%"),  vjust = -0
    scale_fill_viridis(discrete=TRUE) +
    theme(axis.text.x = element_text(angle = 90, vjust = 1, hjust=1)) +
    facet_grid(~year_period) +
    ylab("Porcentagem da quantidade de comandos") +
    xlab("Período (Trimestre)") +
    ggtitle("Comandos memcached 2020 ~ 2022")
```



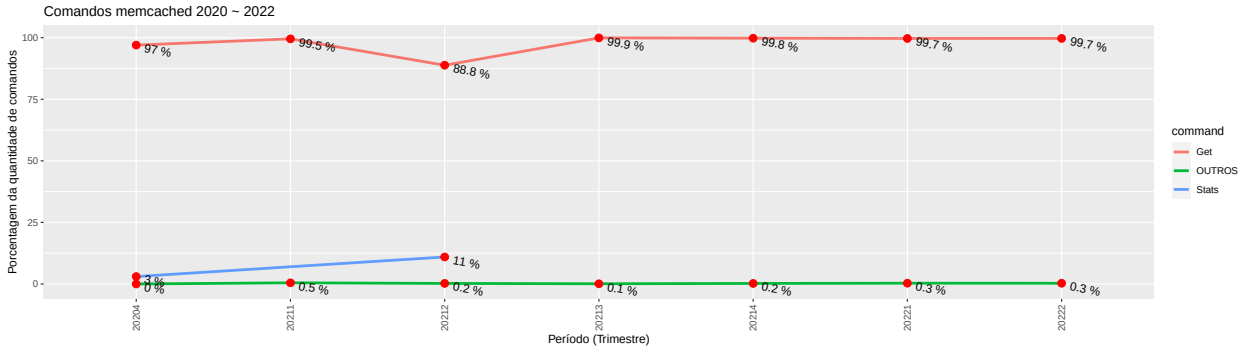
- Gráfico de barras empilhadas 2020 ~ 2022

```
data_grouped_period_command_others_percentage %>%
  ggplot( aes(x=year_period, y=number_of_requests_percentage, fill=command)) +
    geom_bar(stat="identity", width = 0.5) +
    geom_text(aes(label = paste(round(number_of_requests_percentage, decimals_digits), "%")), position =
    scale_fill_viridis(discrete=TRUE) +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
    ylab("Porcentagem da quantidade de comandos") +
    xlab("Período (Trimestre)") +
    ggtitle("Comandos memcached 2020 ~ 2022")
```



- Gráfico de linhas 2020 ~ 2022

```
data_grouped_period_command_others_percentage %>%
  ggplot( aes(x=year_period, y=number_of_requests_percentage, group=command)) +
    geom_line(size=1.2, aes(color=command)) +
    geom_point(color="red", size=3,  aes(color=command)) +
    geom_text(
      aes(label = paste(round(number_of_requests_percentage, decimals_digits), "%")),
      hjust = -0.03, nudge_x = 0.05, nudge_y = -1, angle = -10,
    ) +
    scale_fill_viridis(discrete=TRUE) +
    theme(
      axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
    ) +
    ylab("Porcentagem da quantidade de comandos") +
    xlab("Período (Trimestre)") +
    ggtitle("Comandos memcached 2020 ~ 2022")
```




```
data_grouped_period_payload = data %>%
  mutate(year_period = as.factor(year_period),
         payload = as.factor(quote_payload)) %>%
  group_by(year_period, payload) %>%
  summarise(sum_requests_per_attack = sum(requests_per_attack),
            number_of_payloads_by_period = n())
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

```
data_grouped_period_payload_percentage = data_grouped_period_payload %>%
  ungroup() %>%
  group_by(year_period) %>%
  summarise(payload = payload,
            number_of_payloads_by_period = number_of_payloads_by_period,
            sum_number_of_payloads_by_period = sum(number_of_payloads_by_period),
            sum_period_requests_per_attack = sum(sum_requests_per_attack),
            sum_requests_per_attack = sum_requests_per_attack) %>%
  mutate(number_of_payloads_percentage = (number_of_payloads_by_period / sum_number_of_payloads_by_peri
         number_of_requests_percentage = (sum_requests_per_attack / sum_period_requests_per_attack) * 1
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

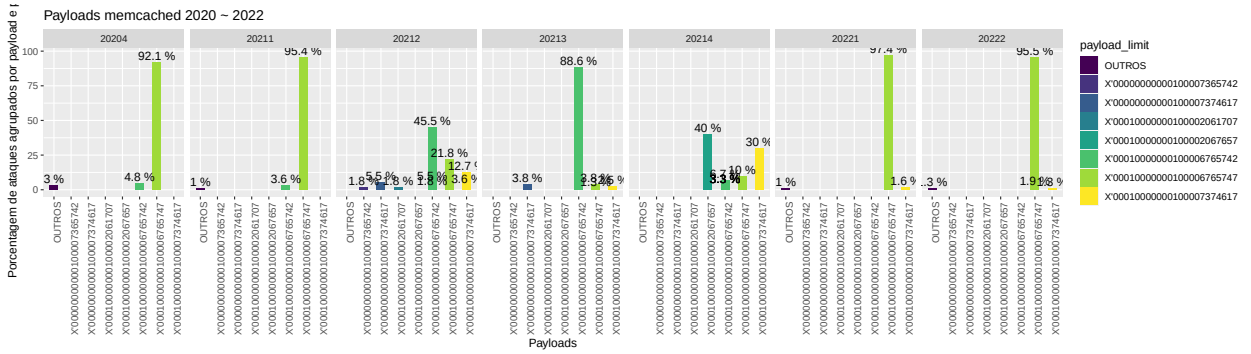
```
data_grouped_period_payload_others_percentage = data_grouped_period_payload_percentage %>%
  mutate(
    payload = case_when(
      number_of_payloads_percentage < minimum_percentage_as_others ~ "OUTROS",
      TRUE ~ as.character(payload)
    )
  ) %>%
  group_by(year_period, payload) %>%
  summarise(
    number_of_payloads_percentage = sum(number_of_payloads_percentage))
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

Porcentagem menores que 1 foram agrupadas como "OUTROS" payload foi limitado a mostrar os 25 primeiros characteres

- Esses dados podem ser analisados como
  - year_period = ano e trimestre
  - payload = o payload utilizado no ataque, aqueles payloads pouco utilizados > 1 foi agrupado como "OUTROS"
  - number_of_payloads_percentage = porcentagem da quantidade de ataques que utilizaram o mesmo payload naquele período nesse caso cada trimestre (year_period) tem um somatório de 100% do campo (number_of_payloads_percentage)
- Gráfico de barras 2020 ~ 2022

```
data_grouped_period_payload_others_percentage %>%
  mutate(payload_limit=substr(payload, 0, 25)) %>%
  ggplot( aes(x=payload_limit, y=number_of_payloads_percentage, fill=payload_limit)) +
    geom_bar(stat="identity", width = 0.5, position="dodge") +
    geom_text(aes(label = paste(round(number_of_payloads_percentage, decimals_digits), "%"),  vjust = -0
    scale_fill_viridis(discrete=TRUE) +
    theme(axis.text.x = element_text(angle = 90, vjust = 1, hjust=1)) +
    facet_grid(~year_period) +
    ylab("Porcentagem de ataques agrupados por payload e periodo") +
    xlab("Payloads") +
    ggtitle("Payloads memcached 2020 ~ 2022")
```



```
data_grouped_period_payload_others_percentage %>%
  mutate(payload_limit=substr(payload, 0, 80)) %>%
  select('year_period', 'percentage'='number_of_payloads_percentage', 'payload_limit') %>%
  print(n=35)
```

```
## # A tibble: 35 x 3
## # Groups:   year_period [7]
##    year_period percentage payload_limit
##    <fct>            <dbl> <chr>
##  1 20204             3.04  OUTROS
##  2 20204             4.82  X'00010000000100006765742068656961 6E0D0A'
##  3 20204            92.1   X'00010000000100006765747320702068206 50A'
##  4 20211             0.982 OUTROS
##  5 20211             3.58  X'00010000000100006765742068656961 6E0D0A'
##  6 20211            95.4   X'00010000000100006765747320702068206 50A'
##  7 20212             1.82  X'00000000000100007365742064632030203020313237332 06E6~
##  8 20212             5.45  X'0000000000010000737461747 30D0A'
##  9 20212             1.82  X'00010000000100002061707065 6E64206674636120302038363~
## 10 20212            45.5   X'00010000000100006765742056636F30570D0A6765742056636~
## 11 20212             5.45  X'00010000000100006765742068656961 6E0D0A'
## 12 20212             1.82  X'00010000000100006765742079756E78690D0A'
## 13 20212            21.8   X'00010000000100006765747320702068206 50A'
## 14 20212            12.7   X'0001000000010000737461747 30D0A'
## 15 20212             3.64  X'0001000000010000737461747 30D0A00'
## 16 20213             3.80  X'0000000000010000737461747 30D0A'
## 17 20213            88.6   X'00010000000100006765742056636F30570D0A6765742056636~
## 18 20213             1.27  X'0001000000010000676574732 00A'
## 19 20213             3.80  X'00010000000100006765747320702068206 50A'
## 20 20213             2.53  X'0001000000010000737461747 30D0A'
## 21 20214            40     X'00010000000100002067657473206875206831206875612 0687~
## 22 20214             6.67  X'00010000000100006765742031313932376534616265343 5613~
## 23 20214             3.33  X'00010000000100006765742038346165633239323939393 1323~
## 24 20214             3.33  X'00010000000100006765742039323965633332636430376 4346~
## 25 20214             3.33  X'00010000000100006765742056636F30570D0A6765742056636~
## 26 20214             3.33  X'00010000000100006765742068656961 6E0D0A'
## 27 20214            10     X'00010000000100006765747320702068206 50A'
## 28 20214            30     X'0001000000010000737461747 30D0A'
## 29 20221             0.971 OUTROS
## 30 20221            97.4   X'00010000000100006765747320702068206 50A'
## 31 20221             1.62  X'0001000000010000737461747 30D0A00'
## 32 20222             1.29  OUTROS
## 33 20222             1.94  X'0001000000010000676574732 00A'
## 34 20222            95.5   X'00010000000100006765747320702068206 50A'
## 35 20222             1.29  X'0001000000010000737461747 30D0A00'
```

Dados do payload get

- São o resto do payload após o "get"

```
data_get_payload = data %>%
  filter(memcached_request_type %in% c("Get")) %>%
```

```
  select(year_period, requests_per_attack, quote_payload_get)

data_grouped_period_payload_get = data_get_payload %>%
  mutate(year_period = as.factor(year_period),
         payload_get = as.factor(quote_payload_get)) %>%
  group_by(year_period, payload_get) %>%
  summarise(sum_requests_per_attack = sum(requests_per_attack),
            number_of_payloads_get_by_period = n())
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

```
data_grouped_period_payload_get_percentage = data_grouped_period_payload_get %>%
  ungroup() %>%
  group_by(year_period) %>%
  summarise(payload_get = payload_get,
            number_of_payloads_get_by_period = number_of_payloads_get_by_period,
            sum_number_of_payloads_get_by_period = sum(number_of_payloads_get_by_period),
            sum_period_requests_per_attack = sum(sum_requests_per_attack),
            sum_requests_per_attack = sum_requests_per_attack) %>%
  mutate(number_of_payloads_get_percentage = (number_of_payloads_get_by_period / sum_number_of_payloads
         number_of_requests_percentage = (sum_requests_per_attack / sum_period_requests_per_attack) * 1
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

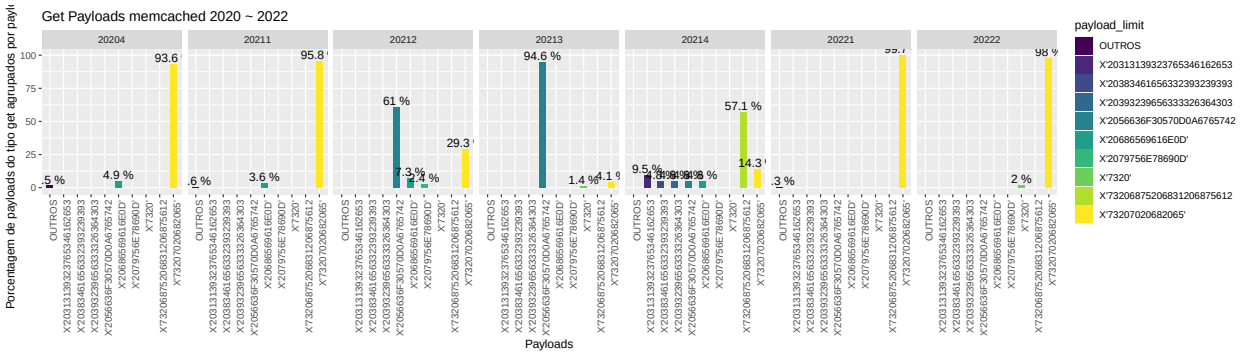
```
data_grouped_period_payload_get_others_percentage = data_grouped_period_payload_get_percentage %>%
  mutate(
    payload_get = case_when(
      number_of_payloads_get_percentage < minimum_percentage_as_others ~ "OUTROS",
      TRUE ~ as.character(payload_get)
    )
  ) %>%
  group_by(year_period, payload_get) %>%
  summarise(
    number_of_payloads_get_percentage = sum(number_of_payloads_get_percentage))
```

```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

- Esse gráfico não ficou bacana, mas da uma noção
- Nesses dados, foi pego somente os registros que eram do tipo "get" e os payloads após o byte 'get', que tecnicamente representa o que foi a key buscada pelo comando get
- Gráfico de barras 2020 ~ 2022

```
data_grouped_period_payload_get_others_percentage %>%
  mutate(payload_limit=substr(payload_get, 0, 25)) %>%
  ggplot( aes(x=payload_limit, y=number_of_payloads_get_percentage, fill=payload_limit)) +
    geom_bar(stat="identity", width = 0.5, position="dodge") +
    geom_text(aes(label = paste(round(number_of_payloads_get_percentage, decimals_digits), "%"),  vjust
    scale_fill_viridis(discrete=TRUE) +
```

```
    theme(axis.text.x = element_text(angle = 90, vjust = 1, hjust=1)) +
    facet_grid(~year_period) +
    ylab("Porcentagem de payloads do tipo get agrupados por payload e periodo") +
    xlab("Payloads") +
    ggtitle("Get Payloads memcached 2020 ~ 2022")
```




```
data_grouped_period_payload_get_others_percentage %>%
  mutate(payload_limit=substr(payload_get, 0, 60)) %>%
  select(year_period, percentage=number_of_payloads_get_percentage, payload_limit) %>%
  print(n=24)
```

```
## # A tibble: 24 x 3
## # Groups:   year_period [7]
##    year_period percentage payload_limit
##    <fct>            <dbl> <chr>
##  1 20204             1.52  OUTROS
##  2 20204             4.89  X'20686569616E0D'
##  3 20204            93.6   X'73207020682065'
##  4 20211             0.635 OUTROS
##  5 20211             3.59  X'20686569616E0D'
##  6 20211            95.8   X'73207020682065'
##  7 20212            61.0   X'2056636F30570D0A6765742056636F30570D0A6765742056636~
##  8 20212             7.32  X'20686569616E0D'
##  9 20212             2.44  X'2079756E78690D'
## 10 20212            29.3   X'73207020682065'
## 11 20213            94.6   X'2056636F30570D0A6765742056636F30570D0A6765742056636~
## 12 20213             1.35  X'7320'
## 13 20213             4.05  X'73207020682065'
## 14 20214             9.52  X'203131393237653461626534356131373462336633623562386~
## 15 20214             4.76  X'203834616563323932393939313233343862353835616432643~
## 16 20214             4.76  X'203932396563333263643037643466306461303835363263316~
## 17 20214             4.76  X'2056636F30570D0A6765742056636F30570D0A6765742056636~
## 18 20214             4.76  X'20686569616E0D'
## 19 20214            57.1   X'732068752068312068756120687562206875632068756420687~
## 20 20214            14.3   X'73207020682065'
## 21 20221             0.331 OUTROS
## 22 20221            99.7   X'73207020682065'
## 23 20222             1.99  X'7320'
## 24 20222            98.0   X'73207020682065'
```