apresentacao

Rafilx

2022-07-25

```
##
## 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

##
## Attaching package: 'scales'

## The following object is masked from 'package:viridis':
##
##     viridis_pal
```

R Markdown

- Apresentação com os plots e conclusões mais que valem a pena estar em uma apresentação

Dados Gerais

Dados analisados

- Todos os protocolos foram agrupados em um único database para realizar a análise de dados

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

data_mix_protocol_unfetch <-dbSendQuery(db_mix_protocol, "
  SELECT *, CAST(CAST(year AS text) || CAST(period AS text) as integer) as year_period
    FROM (
      SELECT *,  strftime(\"%Y\", tempo_inicio) as year, ((strftime(\"%m\", tempo_final) - 1) / 3) + 1 
        FROM MIX_PROTOCOL
    )
   WHERE year_period >= 20183
")
data_mix_protocol <- fetch(data_mix_protocol_unfetch)

dbDisconnect(db_mix_protocol)
```

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

- Formatando os timestamp

```
data_mix_protocol['tempo_final_cast'] = as.POSIXct(data_mix_protocol[['tempo_final']], format = "%Y-%m-
data_mix_protocol['tempo_inicio_cast'] = as.POSIXct(data_mix_protocol[['tempo_inicio']], format = "%Y-%
```

- Total geral

```
data_mix_protocol %>%
  group_by() %>%
  summarise(
    period = max(tempo_final_cast) - min(tempo_inicio_cast),
    total_requests_per_attack = sum(requests_per_attack),
    total_number_of_attacks = n(),
    total_victim = n_distinct(vitima_ip),
    protocols = n_distinct(attack_protocol),
    inicio = min(tempo_inicio_cast),
    fim = max(tempo_final_cast)
)
```

```
## # A tibble: 1 x 7
##   period    total_requests_per_attack total_number_of_at~ total_victim protocols
##   <drtn>                        <dbl>               <int>        <int>     <int>
## 1 1325 days                47162285747             9350315      1812711         9
## # ... with 2 more variables: inicio <dttm>, fim <dttm>
```

- Agrupamento por trimestre

```
data_grouped_period = data_mix_protocol %>%
  mutate(year_period_int = year_period,
         vitima_ip = as.factor(vitima_ip),
         year_period = as.factor(year_period)) %>%
  group_by(year_period) %>%
  summarise(sum_requests_per_attack = sum(requests_per_attack),
            number_of_attacks = n(),
            count_victim = n_distinct(vitima_ip))

data_grouped_period
```

```
## # A tibble: 16 x 4
##    year_period sum_requests_per_attack number_of_attacks count_victim
##    <fct>                         <dbl>             <int>        <int>
##  1 20183                       8175815              1506         1101
##  2 20184                    1707228084            188121        53948
##  3 20191                    2863493225            210394       101634
##  4 20192                    5081561250            364605       164671
##  5 20193                    2167379981            438644       243381
##  6 20194                     563682319             42991        23195
##  7 20201                     881396316            251324       234910
##  8 20202                    2384195735            157261        96669
##  9 20203                    7991034726            372465       185919
## 10 20204                    4706112931            677197       292816
## 11 20211                    6077724098            383619       161817
## 12 20212                    4991995766           1118040       139140
## 13 20213                    1293643577           1205535        63061
## 14 20214                    3094726696           1251412       108706
## 15 20221                    2637695562           1929369        69823
## 16 20222                     712239666            757832        79494
```

- Com a tabela acima, não é possível observar a quantidade de novas vítimas por trimestres, apenas as vítimas distintas naquele trimestre
- Para pegar a quantidade de novas vítimas por trimestre, é necessário agrupar pelo ip da vítima e depois agrupar por trimestre

```
data_new_victim_period = data_mix_protocol %>%
  ungroup() %>%
  group_by(vitima_ip) %>%
  summarise(year_period = min(year_period)) %>%
  ungroup() %>%
  group_by(year_period) %>%
  summarise(new_victims = n_distinct(vitima_ip)) %>%
  mutate(year_period=as.factor(year_period))

data_new_victim_period
```

```
## # A tibble: 16 x 2
##    year_period new_victims
##    <fct>             <int>
##  1 20183              1101
##  2 20184             53745
```

```
##  3 20191             96704
##  4 20192            160390
##  5 20193            236232
##  6 20194             20958
##  7 20201            233617
##  8 20202             93707
##  9 20203            177846
## 10 20204            260435
## 11 20211            140865
## 12 20212            106356
## 13 20213             42652
## 14 20214             83255
## 15 20221             39316
## 16 20222             65532
```

- Para a apresentação, uma tabela dividido por período ficou muito grande, vou agrupar por ano para melhor visualização

```
data_grouped_year = data_mix_protocol %>%
  mutate(year_int = as.integer(year),
         vitima_ip = as.factor(vitima_ip),
         year = as.factor(year)) %>%
  group_by(year) %>%
  summarise(sum_requests_per_attack = sum(requests_per_attack),
            number_of_attacks = n(),
            count_victim = n_distinct(vitima_ip))

data_grouped_year
```

```
## # A tibble: 5 x 4
##   year  sum_requests_per_attack number_of_attacks count_victim
##   <fct>                   <dbl>             <int>        <int>
## 1 2018                1715403899            189627        54846
## 2 2019               10676116775           1056634       520952
## 3 2020               15962739708           1458247       775145
## 4 2021               15458090137           3958606       423354
## 5 2022                3349935228           2687201       139974
```

```
data_new_victim_year = data_mix_protocol %>%
  ungroup() %>%
  group_by(vitima_ip) %>%
  summarise(year = min(year)) %>%
  ungroup() %>%
  group_by(year) %>%
  summarise(new_victims = n_distinct(vitima_ip)) %>%
  mutate(year=as.factor(year))

data_new_victim_year
```

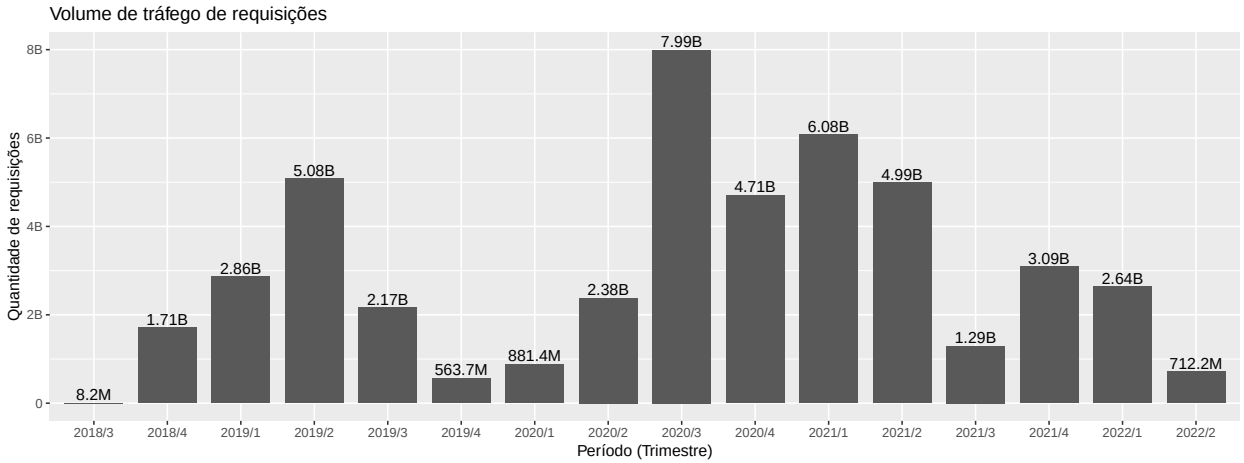
```
## # A tibble: 5 x 2
##   year  new_victims
##   <fct>       <int>
```

```
## 1 2018         54846
## 2 2019        514284
## 3 2020        765605
## 4 2021        373128
## 5 2022        104848
```

Apresentar os resultados com números ao invés de porcentagens

- Total de requisições por trimestre

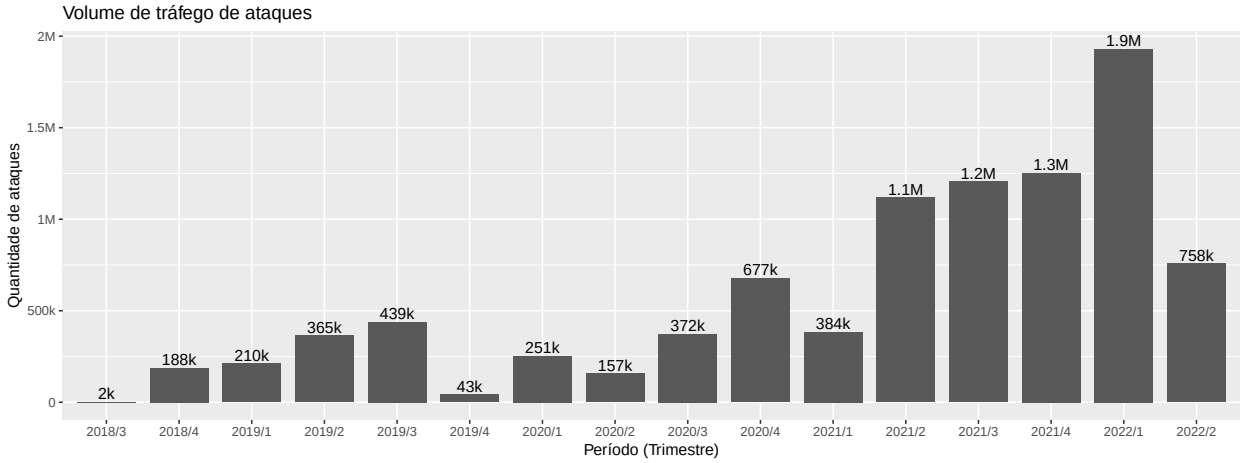
```
data_grouped_period %>%
  mutate(year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/")) %>%
  ggplot( aes(x=year_period, y=sum_requests_per_attack)) +
  geom_bar(stat="identity", width = 0.8, position="dodge") +
  geom_text(aes(label = addUnits(sum_requests_per_attack),  vjust = -0.25)) +
  scale_fill_viridis(discrete=TRUE) +
  scale_y_continuous(labels = addUnits) +
  ylab("Quantidade de requisições") +
  xlab("Período (Trimestre)") +
  ggtitle("Volume de tráfego de requisições")
```



- O terceiro trimestre de 2020 foi o que teve o maior volume de requisições, alcançando a marca de 7.99 Bilhões de requisições realizadas

- Total de ataques por trimestre

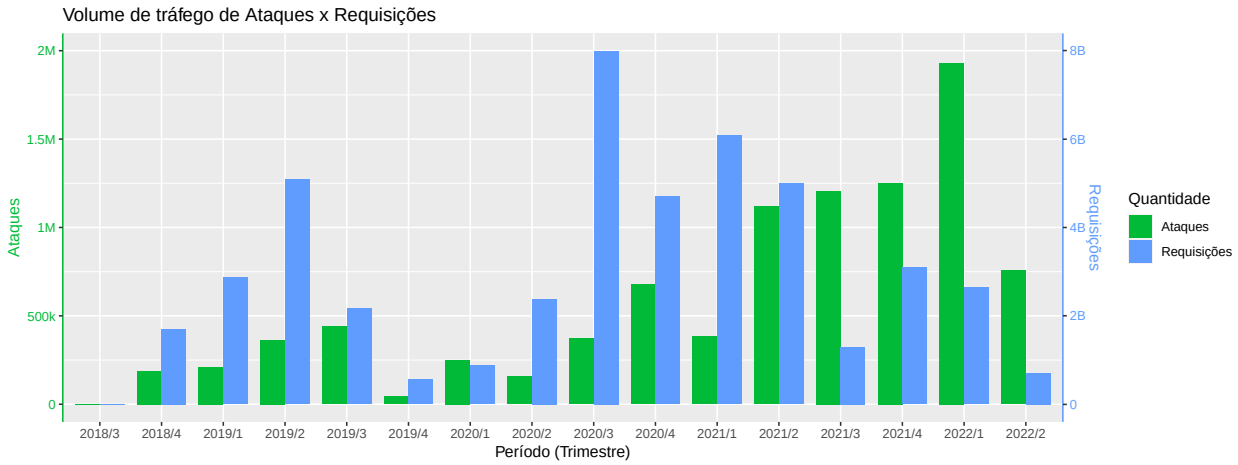
```
data_grouped_period %>%
  mutate(year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/")) %>%
  ggplot( aes(x=year_period, y=number_of_attacks)) +
  geom_bar(stat="identity", width = 0.8, position="dodge") +
  geom_text(aes(label = addUnits(number_of_attacks),  vjust = -0.25)) +
  scale_fill_viridis(discrete=TRUE) +
  scale_y_continuous(labels = addUnits) +
  ylab("Quantidade de ataques") +
  xlab("Período (Trimestre)") +
  ggtitle("Volume de tráfego de ataques")
```



- O trimestre com maior número de requisições foi o 2020.3 com 7.99 Bilhões de requisições em apenas 372 mil ataques, representando que uma maior quantidade de requisições não significam maior número de ataques, mas sim que tiveram ataques com muitas requisições O trimestre com maior número de ataques foi o de 2022.1 com 1.9 Milhões de ataques registrados para apenas 2.64 Bilhões de requisições.

- Juntar dois gráficos de Volume de Ataques e de Requisições
- Total de ataques por trimestre

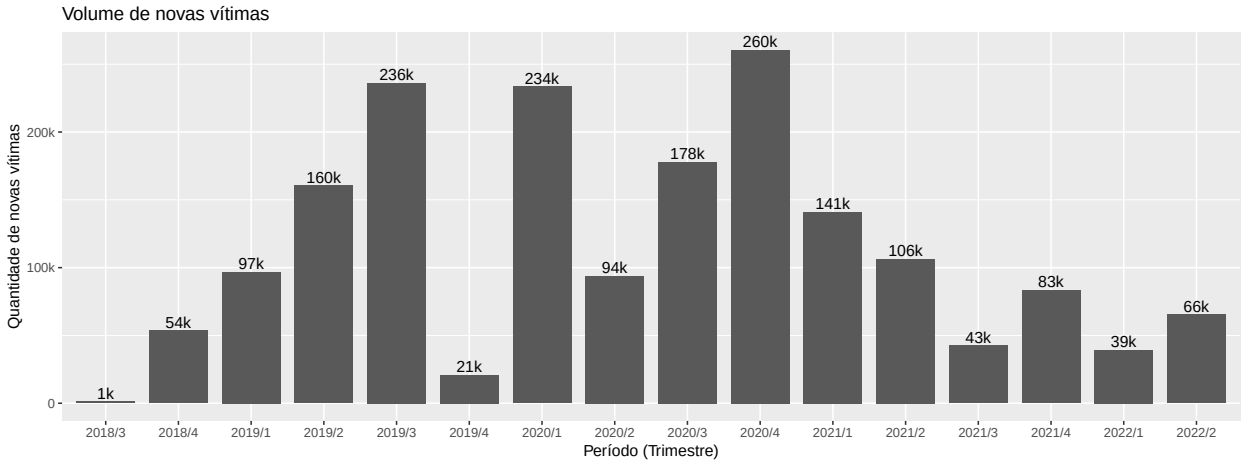
```
scale = 4000
color_requests = "#609cfe" # light blue
color_attacks = "#01bb38" # green
size_text = 14

data_grouped_period %>%
  mutate(sum_requests_per_attack = sum_requests_per_attack/scale) %>%
  select("Requisições"="sum_requests_per_attack", Ataques="number_of_attacks", year_period) %>%
  gather("Ataques", "Requisições", -year_period, key="Quantidade", value = "number") %>%
  mutate(year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/")) %>%
  ggplot( aes(x=year_period, y=number, fill=Quantidade)) +
  geom_bar(stat="identity", width = 0.8, position="dodge") +
  scale_fill_manual(values = c(color_attacks, color_requests)) +
  scale_y_continuous(labels = addUnits, sec.axis = sec_axis(~ . * scale, name = "Requisições", labels =
  theme(
    axis.line.y.right = element_line(color = color_requests),
    axis.text.y.right = element_text(color = color_requests),
    axis.title.y.right = element_text(color = color_requests),
    axis.line.y.left = element_line(color = color_attacks),
    axis.text.y.left = element_text(color = color_attacks),
    axis.title.y.left = element_text(color = color_attacks),
  ) +
  ylab("Ataques") +
  xlab("Período (Trimestre)") +
  ggtitle("Volume de tráfego de Ataques x Requisições")
```



Vítimas novas

- Será verificado o aparecimento e desaparecimento de ip de vítimas durante os períodos
- Total de novas vítimas que surgiram em cada trimestre
  - é contado as vítimas distintas, então a vítima de ip "52.233.175.59" que aparece no trimestre 20204 e 20211 será contado como um somente no primeiro momento em que apareceu (20204)

```
data_new_victim_period %>%
  mutate(year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/")) %>%
  ggplot( aes(x=year_period, y=new_victims)) +
  geom_bar(stat="identity", width = 0.8, position="dodge") +
  geom_text(aes(label = addUnits(new_victims),  vjust = -0.25)) +
  scale_fill_viridis(discrete=TRUE) +
  scale_y_continuous(labels = addUnits) +
  # theme(axis.text.x=element_text(angle=60, hjust=1)) +
  # scale_fill_manual(values=safe_colorblind_palette) +
  # scale_y_log10(breaks = trans_breaks("log10", function(x) 10^x), labels = trans_format("log10", math
  ylab("Quantidade de novas vítimas") +
  xlab("Período (Trimestre)") +
  ggtitle("Volume de novas vítimas")
```



- Existe um número significativo de novas vítimas em todos os trimestres, com aumentos expressivos no terceiro trimestre de 2020 em que o protocolo CoAP foi adicionado o que pode apresentar que no terceiro trimestre de 2020 o honeypot foi encontrado por scans e entre o terceiro e quarto trimestre ele iniciou sendo incorporado a listas de refletores disponíveis a ser usado por ferramentas de ataque ou booters

Após mostrar os resultados gerais, agora irá agrupar também por protocolo

- Agrupamento realizado por período (trimestre) e "attack_protocol" é o protocolo utilizado no ataque `["chargen", "cldap", "coap", "dns", "memcached", "ntp", "qotd", "ssdp", "steam_games", "outros"]`
- Somando a quantidade de requisições utilizadas por cada protocolo e período

```
data_grouped_period_protocol = data_mix_protocol %>%
  mutate(year_period_int = year_period,
         year_period = as.factor(year_period),
         attack_protocol = as.factor(attack_protocol)) %>%
  group_by(year_period, attack_protocol) %>%
  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_protocol_percentage = data_grouped_period_protocol %>%
  ungroup() %>%
  group_by(year_period) %>%
  summarise(attack_protocol = attack_protocol,
            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) * 10
```

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

```
data_grouped_period_protocol_others_percentage = data_grouped_period_protocol_percentage %>%
  mutate(
    attack_protocol = case_when(
      attack_protocol == "NTP" |
       attack_protocol == "MEMCACHED" | attack_protocol == "CHARGEN" ~ as.character(attack_protocol),
      # attack_protocol == "NTP" | number_of_requests_percentage < minimum_percentage_as_others  ~ "OUT
      number_of_requests_percentage < minimum_percentage_as_others  ~ "OUTROS",
      TRUE ~ as.character(attack_protocol)
    )
  ) %>%
```

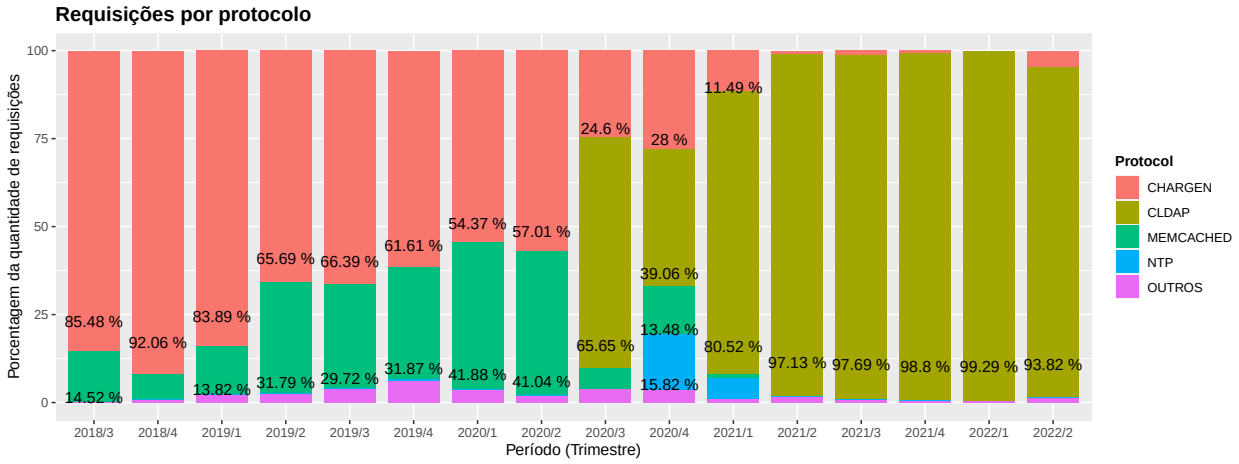
```
  group_by(year_period, attack_protocol) %>%
  summarise(number_of_requests_percentage = sum(number_of_requests_percentage))
```

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

- Gráfico de linhas representando a porcentagem de requisições por protocolo > Protocolos com menos de 5% São agrupados como "Outros"

```
# data_grouped_period_protocol_others_percentage %>%
#   ggplot( aes(x=year_period, y=number_of_requests_percentage, group=attack_protocol)) +
#     labs(color = "Protocol") +
#     geom_line(size=1.2, aes(color=attack_protocol)) +
#     geom_point(size=2,  aes(color=attack_protocol)) +
#     # geom_text_repel(
#     #  aes(label = paste(round(number_of_requests_percentage, decimals_digits), "%"), color=attack_pr
#     #  # xlim = c(NA, Inf),
#     #  # ylim = c(-Inf, Inf)
#     #   nudge_x = -0.4, direction = "y", hjust = "right"
#     # ) +
#     scale_fill_viridis(discrete=TRUE) +
#     theme(legend.title= element_text(size=10, face="bold"),
#           plot.title = element_text(size=14, face="bold")) +
#     # theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
#     ylab("Porcentagem da quantidade de requisições") +
#     xlab("Período (Trimestre)") +
#     ggtitle("Requisições por protocolo")

# data_grouped_period_protocol_percentage %>%
#   # mutate(attack_protocol = as.factor(attack_protocol)) %>%
#   # rename("Protocol" = attack_protocol) %>%
#   ggplot( aes(x=year_period, y=number_of_requests_percentage)) +
#     # labs(color = "Protocol") +
#     # geom_bar(stat="identity", width = 0.5) +
#     # geom_text_repel(
#     #   aes(label = paste(round(number_of_requests_percentage, 2), "%")),
#     #   position = position_stack(vjust = 0.1)
#     # ) +
#     facet_wrap(~attack_protocol) +
#     theme(legend.title= element_text(size=10, face="bold"),
#           plot.title = element_text(size=14, face="bold")) +
#     # scale_fill_viridis(discrete=TRUE) +
#     ylab("Porcentagem da quantidade de requisições") +
#     xlab("Período (Trimestre)") +
#     ggtitle("Requisições por protocolo")

# Não pode morrer, plotar completo inicio ao fim, CHARGEN, MEMCACHED e talvez NTP
# justificativa do porque eles morreram do nada....
# NTP adicionar no periodo 2018.3

data_grouped_period_protocol_others_percentage %>%
  rename("Protocol" = attack_protocol) %>%
  mutate(year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/")) %>%
```

```
  ggplot( aes(x=year_period, y=number_of_requests_percentage, fill=Protocol)) +
    labs(color = "Protocol") +
    geom_bar(stat="identity", width = 0.8) +
    # geom_text_repel(
    #   aes(label = paste(round(number_of_requests_percentage, 2), "%")),
    #   position = position_stack(vjust = 0.1)
    # ) +
    geom_text(
      aes(label = labelBiggerThan9(number_of_requests_percentage)), position = position_stack(vjust = 0
      ) +
    theme(legend.title= element_text(size=10, face="bold"),
          plot.title = element_text(size=14, face="bold")) +
    # scale_fill_viridis(discrete=TRUE, option = "C") +
    ylab("Porcentagem da quantidade de requisições") +
    xlab("Período (Trimestre)") +
    ggtitle("Requisições por protocolo")
```



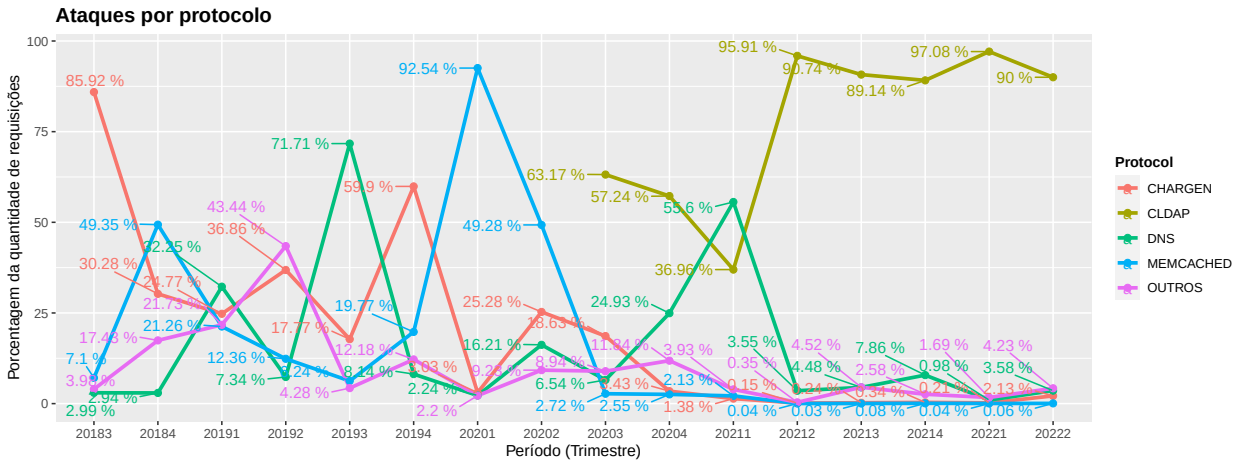
- Pré CLDAP vs Pós CLDAP (predominante)
  - Pré é predominante de CHARGEN e MEMCACHED
  - Curiosamente NTP aparece em 2020.4
- Agrupa como outros

```
data_grouped_period_protocol_others_attacks_percentage = data_grouped_period_protocol_percentage %>%
  mutate(
    attack_protocol = case_when(
      #attack_protocol == "NTP" |
       attack_protocol == "MEMCACHED" | attack_protocol == "CHARGEN" | attack_protocol == "DNS" ~ as.cha
      # attack_protocol == "NTP" | number_of_attacks_percentage < minimum_percentage_as_others  ~ "OUTR
      attack_protocol == "SSDP" | number_of_attacks_percentage < 30  ~ "OUTROS",
      TRUE ~ as.character(attack_protocol)
    )
  ) %>%
  group_by(year_period, attack_protocol) %>%
  summarise(number_of_attacks_percentage = sum(number_of_attacks_percentage))
```

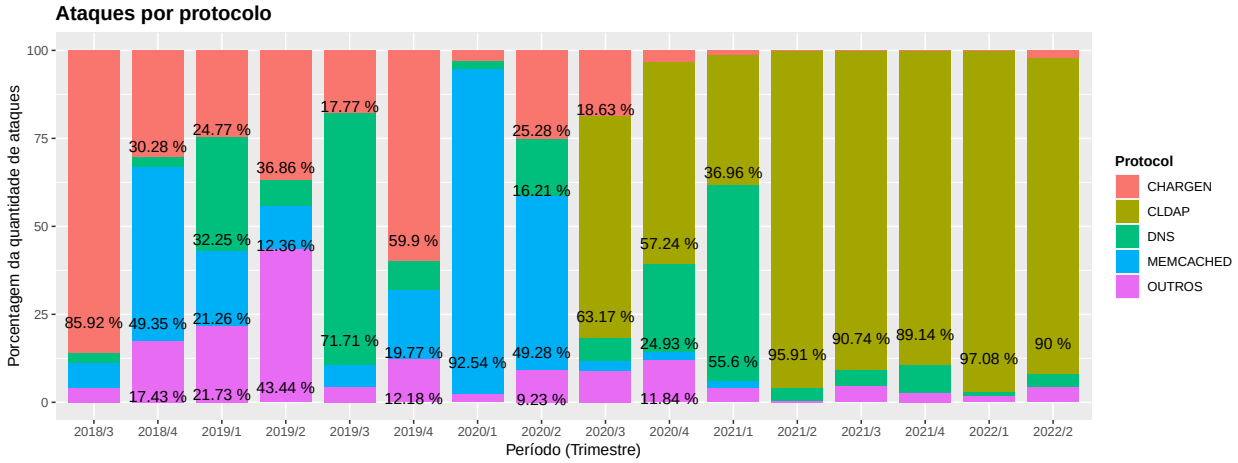
```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

- Gráfico de linhas representando a porcentagem de ataques por protocolo

```
data_grouped_period_protocol_others_attacks_percentage %>%
  ggplot( aes(x=year_period, y=number_of_attacks_percentage, group=attack_protocol)) +
    labs(color = "Protocol") +
    geom_line(size=1.2, aes(color=attack_protocol)) +
    geom_point(size=2,  aes(color=attack_protocol)) +
    geom_text_repel(
     aes(label = paste(round(number_of_attacks_percentage, decimals_digits), "%"), color=attack_protocol
     # xlim = c(NA, Inf),
     # ylim = c(-Inf, Inf)
      nudge_x = -0.4, direction = "y", hjust = "right"
    ) +
    scale_fill_viridis(discrete=TRUE) +
    theme(legend.title= element_text(size=10, face="bold"),
          plot.title = element_text(size=14, face="bold")) +
    # theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
    ylab("Porcentagem da quantidade de requisições") +
    xlab("Período (Trimestre)") +
    ggtitle("Ataques por protocolo")
```




```
data_grouped_period_protocol_others_attacks_percentage %>%
  rename("Protocol" = attack_protocol) %>%
  mutate(year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/")) %>%
  ggplot( aes(x=year_period, y=number_of_attacks_percentage, fill=Protocol)) +
    labs(color = "Protocol") +
    geom_bar(stat="identity", width = 0.8) +
    geom_text(
      aes(label = labelBiggerThan9(number_of_attacks_percentage)), position = position_stack(vjust = 0.
      ) +
    theme(legend.title= element_text(size=10, face="bold"),
          plot.title = element_text(size=14, face="bold")) +
    # scale_fill_viridis(discrete=TRUE) +
    ylab("Porcentagem da quantidade de ataques") +
    xlab("Período (Trimestre)") +
    ggtitle("Ataques por protocolo")
```



- Antes de entrar no específico do DNS, algo que chamou a atenção é a grande quantidade de CLDAP que é um protocolo que foi adicionado posteriormente ao honeypot MP-H - Então uma tabela sem os dados referentes aos protocolos adicionados posteriormente (CLDAP, COAP)

- Para a apresentação, uma tabela dividido por período ficou muito grande, vou agrupar por ano para melhor visualização

```
data_mix_protocol_rm_cldap = data_mix_protocol %>%
  filter(!attack_protocol %in% c("CLDAP", "COAP"))

data_mix_protocol_grouped_year_rm_cldap = data_mix_protocol_rm_cldap %>%
  mutate(year_int = as.integer(year),
         vitima_ip = as.factor(vitima_ip),
         year = as.factor(year)) %>%
  group_by(year) %>%
  summarise(sum_requests_per_attack = sum(requests_per_attack),
            number_of_attacks = n(),
            count_victim = n_distinct(vitima_ip))

data_mix_protocol_new_victim_year_rm_cldap = data_mix_protocol_rm_cldap %>%
  ungroup() %>%
  group_by(vitima_ip) %>%
  summarise(year = min(year)) %>%
  ungroup() %>%
  group_by(year) %>%
  summarise(new_victims = n_distinct(vitima_ip)) %>%
  mutate(year=as.factor(year))

data_mix_protocol_grouped_year_rm_cldap_general = data_mix_protocol_grouped_year_rm_cldap %>%
  inner_join(data_mix_protocol_new_victim_year_rm_cldap)
```

```
## Joining, by = "year"
```

```
data_mix_protocol_grouped_year_rm_cldap_general
```

```
## # A tibble: 5 x 5
##   year  sum_requests_per_attack number_of_attacks count_victim new_victims
```

```
##   <fct>                  <dbl>             <int>        <int>       <int>
## 1 2018                1715403899            189627        54846       54846
## 2 2019               10676116775           1056634       520952      514284
## 3 2020                8874882967            833948       478915      473123
## 4 2021                1393067376            530825       129389      122200
## 5 2022                  62609720            131629        30419       24918
```

```
data_mix_protocol_rm_cldap %>%
  ungroup() %>%
  summarise(
    period = max(tempo_final_cast) - min(tempo_inicio_cast),
    total_requests_per_attack = sum(requests_per_attack),
    total_number_of_attacks = n(),
    total_victim = n_distinct(vitima_ip),
    protocols = n_distinct(attack_protocol),
    inicio = min(tempo_inicio_cast),
    fim = max(tempo_final_cast)
  )
```

```
##      period total_requests_per_attack total_number_of_attacks total_victim
## 1 1325 days               22722080737                 2742663      1189371
##   protocols              inicio                 fim
## 1         7 2018-09-24 17:48:09 2022-05-11 14:41:27
```

DNS

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

data_dns_unfetch <-dbSendQuery(db_dns, "
  SELECT *, CAST(CAST(year AS text) || CAST(period AS text) as integer) as year_period
    FROM DNS_ANALYSIS
        JOIN DNS_ANALYSIS_QUESTION
          ON DNS_ANALYSIS.id = DNS_ANALYSIS_QUESTION.dns_analysis_id
     WHERE QTYPE != 0
")
data_dns <- fetch(data_dns_unfetch)

dbDisconnect(db_dns)
```

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

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

```
data_dns %>%
  group_by() %>%
  summarise(period = max(tempo_final_cast) - min(tempo_inicio_cast),
            total_requests_per_attack = sum(requests_per_attack),
```

```
            total_number_of_attacks = n(),
            inicio = min(tempo_inicio_cast),
            fim = max(tempo_final_cast)
            )
```

```
## # A tibble: 1 x 5
##   period     total_requests_per_attack total_number_of_atta~ inicio
##   <drtn>                         <int>                 <int> <dttm>
## 1 558.9 days                  89983623                587860 2020-10-29 16:15:05
## # ... with 1 more variable: fim <dttm>
```

```
data_dns %>%
  group_by(year_period) %>%
  summarise(
            total_requests_per_attack = sum(requests_per_attack),
            total_number_of_attacks = n(),
            total_victim = n_distinct(ip),
            )
```

```
## # A tibble: 7 x 4
##   year_period total_requests_per_attack total_number_of_attacks total_victim
##         <int>                     <int>                   <int>        <int>
## 1       20204                  22039213                  136983        23996
## 2       20211                  40049327                  213052        66819
## 3       20212                  15808087                   39611        17328
## 4       20213                   1278717                   53934         7600
## 5       20214                   6563759                   98317        12727
## 6       20221                   3505991                   18869         6841
## 7       20222                    738529                   27094         2815
```

```
data_dns_new_victim_year = data_dns %>%
  ungroup() %>%
  group_by(ip) %>%
  summarise(year_period = min(year_period)) %>%
  ungroup() %>%
  group_by(year_period) %>%
  summarise(new_victims = n_distinct(ip)) %>%
  mutate(year_period=as.factor(year_period))

data_dns_new_victim_year
```

```
## # A tibble: 7 x 2
##   year_period new_victims
##   <fct>             <int>
## 1 20204             23996
## 2 20211             65731
## 3 20212             16400
## 4 20213              7055
## 5 20214             11986
## 6 20221              5358
## 7 20222              1996
```

```
data_dns_grouped = data_dns %>%
  group_by(qname, qtype) %>%
  summarise(tempo_inicio=min(tempo_inicio_cast), tempo_final=max(tempo_final_cast),
            sum_requests_per_attack=sum(requests_per_attack), number_of_attacks=n()) %>%
  mutate(tempo_diff_secs = as.numeric(tempo_final - tempo_inicio, units="secs"),
         tempo_diff = tempo_final - tempo_inicio,
         avg_requests_per_attack=trunc(sum_requests_per_attack/number_of_attacks)) %>%
  arrange(desc(tempo_diff_secs))
```

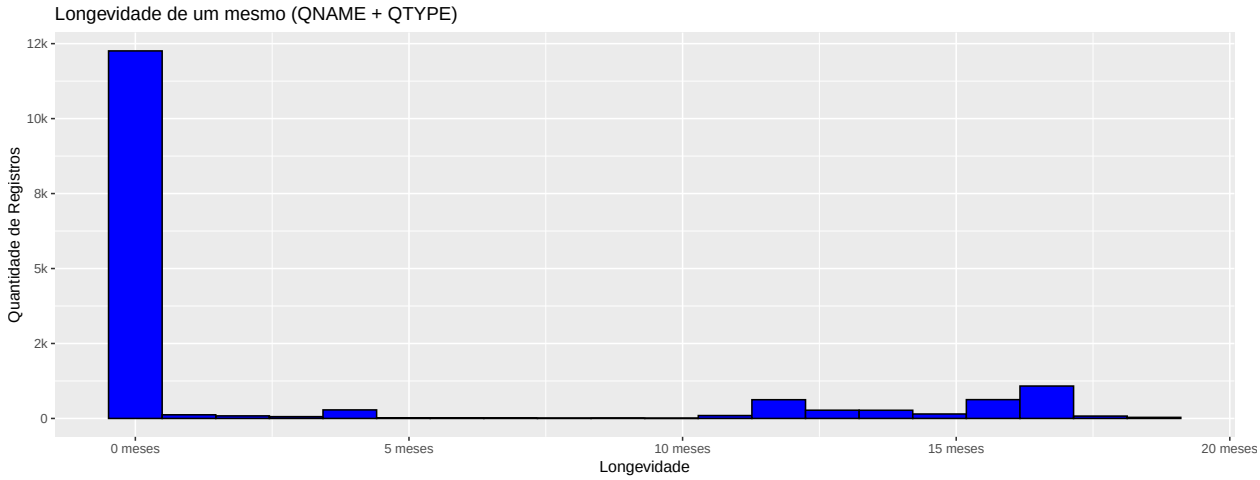
```
## `summarise()` has grouped output by 'qname'. You can override using the
## `.groups` argument.
```

```
data_dns_grouped
```

```
## # A tibble: 16,125 x 9
## # Groups:   qname [15,305]
##    qname          qtype tempo_inicio        tempo_final         sum_requests_pe~
##    <chr>          <chr> <dttm>              <dttm>                         <int>
##  1 VERSION.BIND.  TXT   2020-10-30 02:39:27 2022-05-11 13:05:22             6160
##  2 peacecorps.go~ ANY   2020-10-31 14:28:23 2022-05-11 05:22:49         72734156
##  3 200-19-107-23~ A     2020-10-30 11:15:36 2022-05-09 21:23:22              148
##  4 version.bind.  TXT   2020-11-01 04:46:10 2022-05-11 12:13:52              833
##  5 a.gtld-server~ A     2020-10-30 02:55:07 2022-05-09 06:50:12               89
##  6 com.           ANY   2020-10-31 11:38:32 2022-05-09 16:38:23              511
##  7 238.107.19.20~ PTR   2020-11-01 23:58:38 2022-05-09 04:56:13              842
##  8 dns-test.rese~ TXT   2020-11-04 06:49:48 2022-05-11 09:58:47               87
##  9 mopa.ae.       MX    2020-11-04 14:39:23 2022-05-11 13:52:05             6096
## 10 szgmc.gov.ae.  MX    2020-11-03 14:47:10 2022-05-10 13:29:54              889
## # ... with 16,115 more rows, and 4 more variables: number_of_attacks <int>,
## #   tempo_diff_secs <dbl>, tempo_diff <drtn>, avg_requests_per_attack <dbl>
```

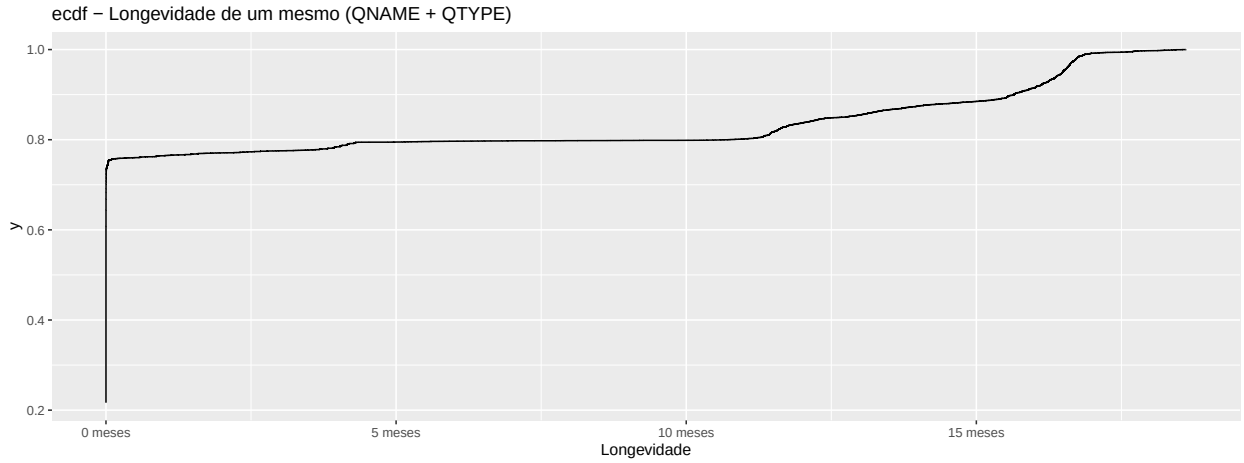
- Dessa forma é buscado apresentar quanto tempo um QNAME em conjunto com o QTYPE, permanece ativo, ex o qname "peacecorps.gov." do qtype "ANY" teve o primeiro ataque registrado em 31/10/2020 e o ultimo ataque com o mesmo qname e qtype em 11/05/2022 totalizando uma longevidade de aproximadamente um ano e meio (48092066 secs)

```
data_dns_grouped %>%
  mutate(tempo_diff_secs = tempo_diff_secs / months) %>%
  ggplot(aes(x= tempo_diff_secs)) +
  geom_histogram(bins = 20, fill='blue', color ='black') +
  ggtitle("Longevidade de um mesmo (QNAME + QTYPE)") +
  xlab("Longevidade") +
  ylab("Quantidade de Registros") +
  scale_y_continuous(labels = addUnits) +
  scale_x_continuous(labels = addUnitMeses) +
  theme(plot.margin = margin(r = 15))
```




```
# verificar para colocar a cada 3 ou 4 meses
```

```
data_dns_grouped %>%
  mutate(tempo_diff_secs = tempo_diff_secs / months) %>%
  ggplot(aes(x= tempo_diff_secs)) +
  stat_ecdf(geom = "step", pad = FALSE) +
  ggtitle("ecdf - Longevidade de um mesmo (QNAME + QTYPE)") +
  scale_x_continuous(labels = addUnitMeses) +
  xlab("Longevidade")
```

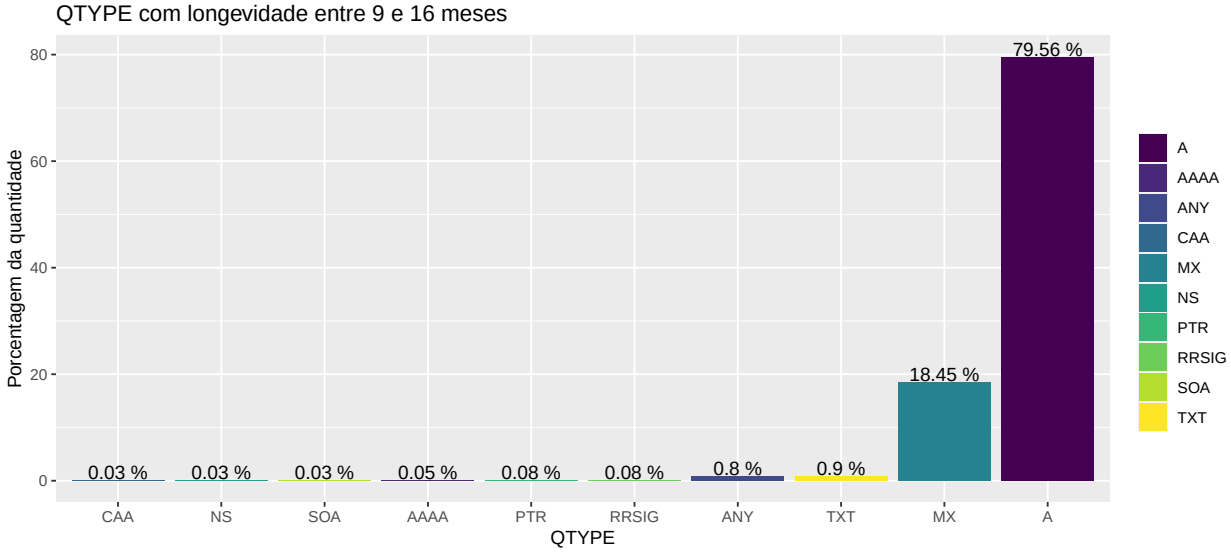



```
percentage_76_secs = quantile(data_dns_grouped$tempo_diff_secs, c(.76))[[1]]

data_dns_bigger_than_76 = data_dns_grouped %>%
  filter(tempo_diff_secs > percentage_76_secs) %>%
  ungroup() %>%
  group_by(qtype) %>%
  summarise(qtype_quantity = n()) %>%
  arrange(desc(qtype_quantity))


sum_qtype_quantity = sum(data_dns_bigger_than_76$qtype_quantity)
data_dns_bigger_than_76_percentage = data_dns_bigger_than_76 %>%
```

```
  mutate(qtype_quantity_percentage = (qtype_quantity / sum_qtype_quantity) * 100)

data_dns_bigger_than_76_percentage
```

```
## # A tibble: 10 x 3
##    qtype qtype_quantity qtype_quantity_percentage
##    <chr>          <int>                     <dbl>
##  1 A               3079                   79.6
##  2 MX               714                   18.4
##  3 TXT               35                    0.904
##  4 ANY               31                    0.801
##  5 PTR                3                    0.0775
##  6 RRSIG              3                    0.0775
##  7 AAAA               2                    0.0517
##  8 CAA                1                    0.0258
##  9 NS                 1                    0.0258
## 10 SOA                1                    0.0258
```

```
data_dns_bigger_than_76_percentage %>%
 ggplot( aes(x=reorder(qtype, +qtype_quantity_percentage), y=qtype_quantity_percentage, fill=qtype)) +
    geom_bar(stat="identity", position="dodge") +
    scale_fill_viridis(discrete=TRUE, name="") +
    geom_text(aes(label = paste(round(qtype_quantity_percentage, 2), "%")), vjust = -0.10, ) +
    ylab("Porcentagem da quantidade") +
    xlab("QTYPE") +
    ggtitle("QTYPE com longevidade entre 9 e 16 meses")
```

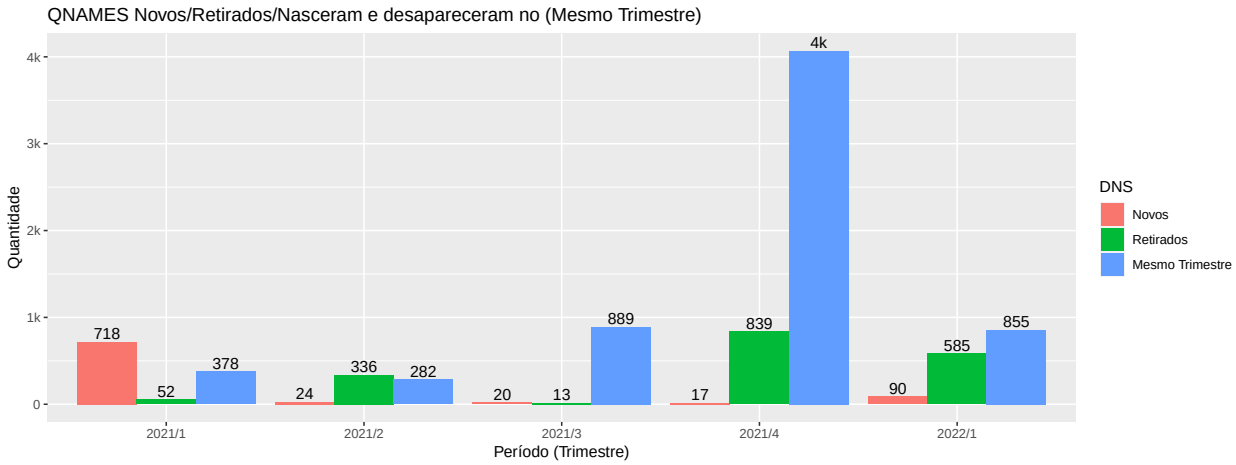



```
data_dns_quarters = data_dns_grouped %>%
  mutate(year_period_inicio=paste(year(tempo_inicio), quarters(tempo_inicio), sep = "/"),
         year_period_fim=paste(year(tempo_final), quarters(tempo_final), sep = "/"))

data_dns_different_quarters = data_dns_quarters %>%
```

```
  filter(year_period_inicio != year_period_fim)

data_dns_quarters_new = data_dns_different_quarters %>%
  ungroup() %>%
  count(year_period_inicio, name = "Novos")

data_dns_quarters_removed = data_dns_different_quarters %>%
  ungroup() %>%
  count(year_period_fim, name = "Retirados")

data_dns_diff_quarters_new_removed = data_dns_quarters_new %>%
  inner_join(data_dns_quarters_removed, by = c("year_period_inicio" = "year_period_fim")) %>%
  mutate(year_period =  as.factor(year_period_inicio) ) %>%
  select('year_period', 'Novos', 'Retirados') %>%
  arrange(year_period)


data_dns_same_quarters = data_dns_quarters %>%
  filter(year_period_inicio == year_period_fim) %>%
  ungroup() %>%
  count(year_period_inicio, name = "Mesmo Trimestre") %>%
    mutate(year_period =  as.factor(year_period_inicio) )

data_quarters_new_removed_same = data_dns_diff_quarters_new_removed %>%
  inner_join(data_dns_same_quarters, by = c("year_period" = "year_period")) %>%
  mutate(year_period =  as.factor(year_period_inicio) ) %>%
  select('year_period', 'Novos', 'Retirados', 'Mesmo Trimestre') %>%
  arrange(year_period)


data_quarters_new_removed_same.st = stack(data_quarters_new_removed_same)
```

```
## Warning in stack.data.frame(data_quarters_new_removed_same): non-vector columns
## will be ignored
```

```
data_quarters_new_removed_same.st$year_period = rep(data_quarters_new_removed_same$year_period, 3)


data_quarters_new_removed_same.st %>%
  mutate(
    year_period = paste(substr(year_period, 0, 4), substr(year_period, 7, 7), sep = "/"),
    DNS = ind,
  ) %>%
  ggplot( aes(x=year_period, y=values, fill=DNS)) +
  geom_bar(stat="identity", position=position_dodge()) +
  ggtitle("QNAMES Novos/Retirados/Nasceram e desapareceram no (Mesmo Trimestre)") +
  geom_text(aes(label = addUnits(values)), vjust=-0.25, position = position_dodge(0.9)) +
  scale_y_continuous(labels = addUnits) +
  xlab("Período (Trimestre)") +
  ylab("Quantidade")
```



- Após o query id é possível ver a predominancia de ANY em ataques DNS

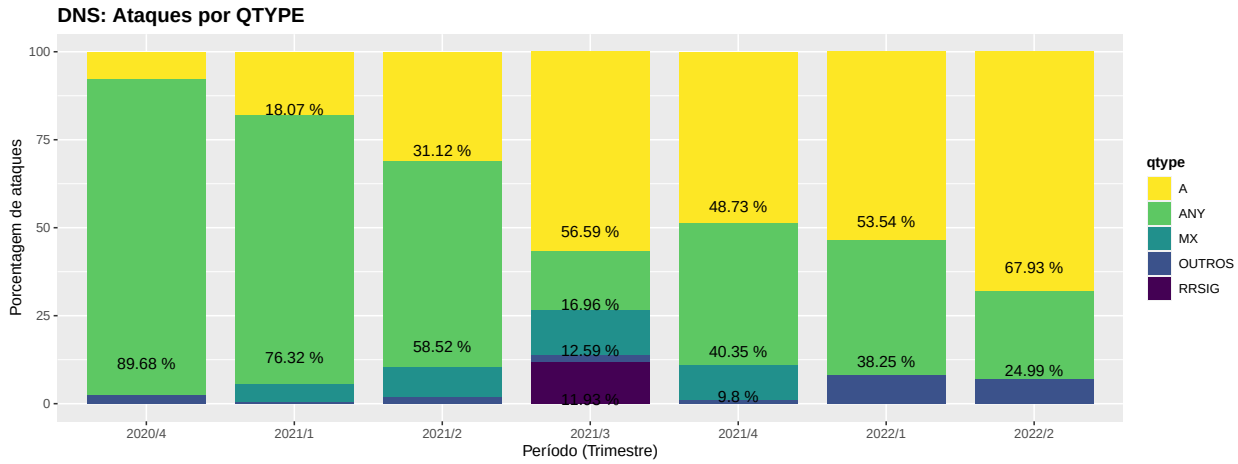
```
data_dns_qtype_percentage = data_dns %>%
  group_by(year_period, qtype) %>%
  summarise(
    ataques = n(),
    requests = sum(requests_per_attack),
  ) %>%
  ungroup() %>%
  group_by(year_period) %>%
  summarise(
    trimestre_ataques = sum(ataques),
    trimestre_requests = sum(requests),
    ataques = ataques,
    qtype = qtype,
    ataques_percentage = (ataques / trimestre_ataques) * 100,
    requests_percentage = (requests / trimestre_requests) * 100,
  )
```

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

```
data_dns_qtype_ataques_others_percentage = data_dns_qtype_percentage %>%
  mutate(
    qtype = case_when(
      ataques_percentage < minimum_percentage_as_others  ~ "OUTROS",
      TRUE ~ as.character(qtype)
    )
  ) %>%
  group_by(year_period, qtype) %>%
  summarise(ataques_percentage = sum(ataques_percentage))
```

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

```
data_dns_qtype_ataques_others_percentage %>%
  mutate(year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/")) %>%
  ggplot( aes(x=year_period, y=ataques_percentage, fill=qtype)) +
    geom_bar(stat="identity", width = 0.8) +
    geom_text(
      aes(label = labelBiggerThan9(ataques_percentage)), position = position_stack(vjust = 0.1)
      ) +
    theme(legend.title= element_text(size=10, face="bold"),
          plot.title = element_text(size=14, face="bold")) +
    scale_fill_viridis(discrete=TRUE, direction = -1) +
    ylab("Porcentagem de ataques") +
    xlab("Período (Trimestre)") +
    ggtitle("DNS: Ataques por QTYPE")
```

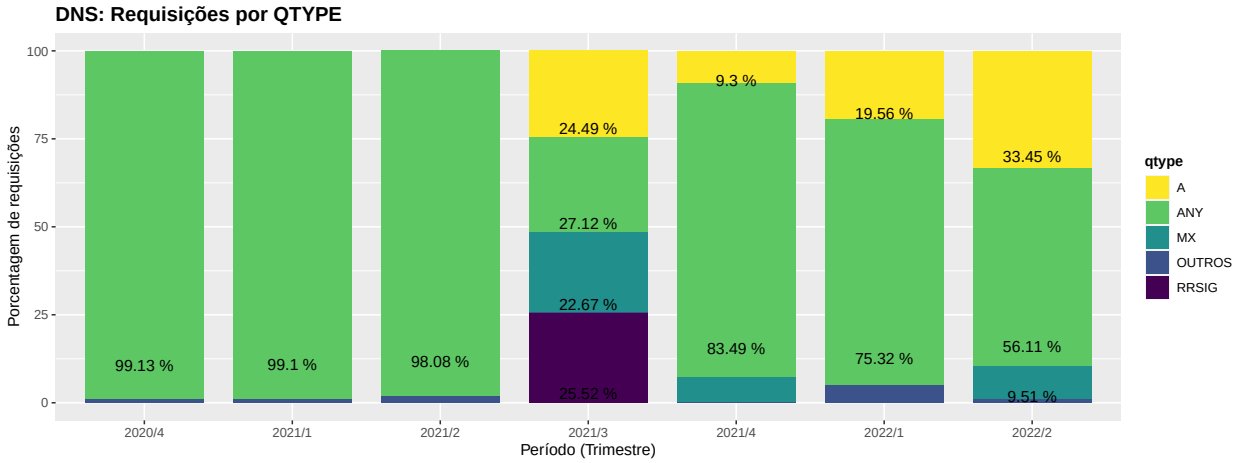



```
data_dns_qtype_requests_others_percentage = data_dns_qtype_percentage %>%
  mutate(
    qtype = case_when(
      requests_percentage < minimum_percentage_as_others  ~ "OUTROS",
      TRUE ~ as.character(qtype)
    )
  ) %>%
  group_by(year_period, qtype) %>%
  summarise(requests_percentage = sum(requests_percentage))
```

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

```
data_dns_qtype_requests_others_percentage %>%
  mutate(year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/")) %>%
  ggplot( aes(x=year_period, y=requests_percentage, fill=qtype)) +
    geom_bar(stat="identity", width = 0.8) +
    geom_text(
      aes(label = labelBiggerThan9(requests_percentage)), position = position_stack(vjust = 0.1)
      ) +
    theme(legend.title= element_text(size=10, face="bold"),
```

```
        plot.title = element_text(size=14, face="bold")) +
    scale_fill_viridis(discrete=TRUE, direction = -1) +
    ylab("Porcentagem de requisições") +
    xlab("Período (Trimestre)") +
    ggtitle("DNS: Requisições por QTYPE")
```



NTP

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

data_ntp_unfetch <-dbSendQuery(db_ntp, "
  SELECT *, CAST(CAST(year AS text) || CAST(period AS text) as integer) as year_period
    FROM NTP_ANALYSIS
")
data_ntp <- fetch(data_ntp_unfetch)

dbDisconnect(db_ntp)
```

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

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

data_ntp_grouped_period_ntp_type = data_ntp %>%
  mutate(year_period = as.factor(year_period)) %>%
  group_by(year_period, ntp_type) %>%
  summarise(sum_requests_per_attack = sum(requests_per_attack), number_of_attacks = n())
```

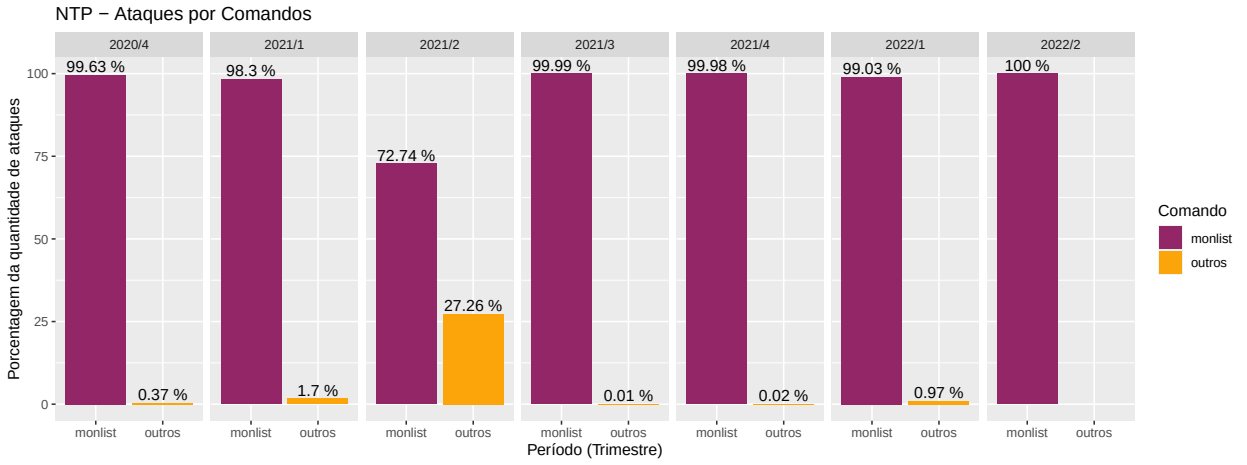
```
## `summarise()` has grouped output by 'year_period'. You can override using the
## `.groups` argument.
```

```
data_ntp_grouped_period_ntp_type_percentage = data_ntp_grouped_period_ntp_type %>%
  group_by(year_period) %>%
  summarise(ntp_type = ntp_type, number_of_attacks = number_of_attacks,
            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) * 10
```

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

- Gráfico de barras empilhadas apresentando a porcentagem da quantidade de ataques em cada "ntp_type" por período

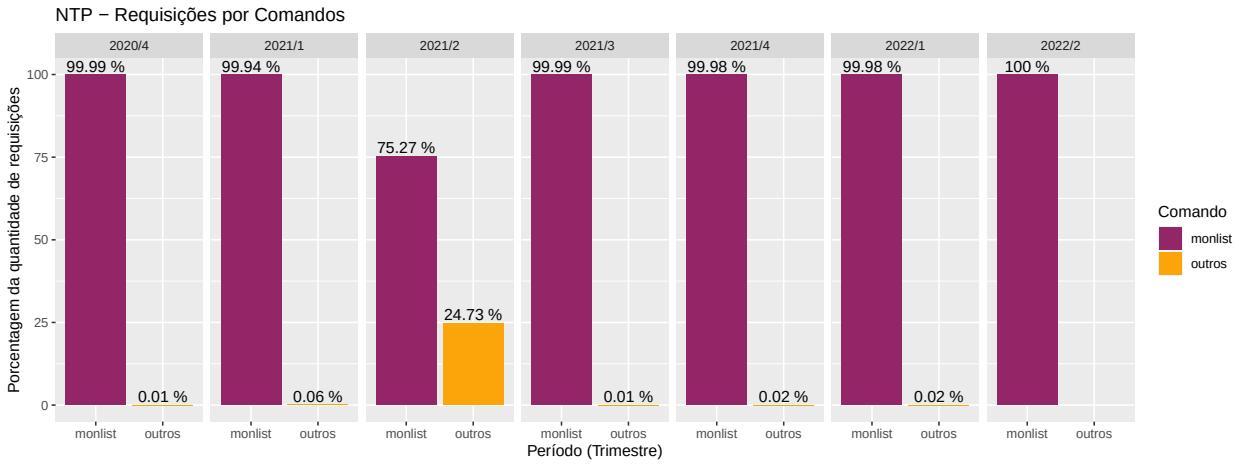
```
data_ntp_grouped_period_ntp_type_percentage %>%
  mutate(
    year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/"),
    Comando = ntp_type,
  ) %>%
  ggplot( aes(x=Comando, y=number_of_attacks_percentage, fill=Comando)) +
    geom_bar(stat="identity", position="dodge") +
    geom_text(aes(label = paste(round(number_of_attacks_percentage, 2), "%"),  vjust = -0.25)) +
    scale_fill_viridis(discrete=TRUE, option="inferno", begin = 0.8, end = 0.4, direction = -1) +
    facet_grid(~year_period) +
    ylab("Porcentagem da quantidade de ataques") +
    xlab("Período (Trimestre)") +
    ggtitle("NTP - Ataques por Comandos")
```



- Gráfico de barras apresentando a porcentagem da quantidade de requisições em cada "ntp_type" por período

```
data_ntp_grouped_period_ntp_type_percentage %>%
  mutate(
    year_period = paste(substr(year_period, 0, 4), substr(year_period, 5, 5), sep = "/"),
    Comando = ntp_type,
```

```
  ) %>%
  ggplot( aes(x=Comando, y=number_of_requests_percentage, fill=Comando)) +
    geom_bar(stat="identity", position="dodge") +
    geom_text(aes(label = paste(round(number_of_requests_percentage, 2), "%"),  vjust = -0.25)) +
    scale_fill_viridis(discrete=TRUE, option="inferno", begin = 0.8, end = 0.4, direction = -1) +
    facet_grid(~year_period) +
    ylab("Porcentagem da quantidade de requisições") +
    xlab("Período (Trimestre)") +
    ggtitle("NTP - Requisições por Comandos")
```



Memcached

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


data_memcached_payload_types_unfetch <-dbSendQuery(db_memcached, "
  SELECT id, quantity, SUBSTR(payload,0,25) AS payload_limit
    FROM MEMCACHED_PAYLOAD_TYPES
")
data_memcached_payload_types <- fetch(data_memcached_payload_types_unfetch)


dbDisconnect(db_memcached)
```

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

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

```r
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
```

```r
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) +
    ylab("Porcentagem") +
    xlab("Comando") +
    ggtitle("Memacached: Ataques por Comando")
```