There are many ways to achieve this result. I’ll show one here and walk through the process starting with the data (this is the 2018 DMARC evaluation run):

library(hrbrthemes)

library(ggchicklet)

library(tidyverse)

f500\_dmarc <- read\_csv("<https://rud.is/dl/f500-industry-dmarc.csv.gz>", col\_types = "cc")

f500\_dmarc

## # A tibble: 500 x 2

## industry p

##

## 1 Retailing Reject

## 2 Technology None

## 3 Health Care Reject

## 4 Wholesalers None

## 5 Retailing Quarantine

## 6 Motor Vehicles & Parts None

## 7 Energy None

## 8 Wholesalers None

## 9 Retailing None

## 10 Telecommunications Quarantine

## # … with 490 more rows

The p column is the DMARC classification for each organization (org names have been withheld to protect the irresponsible) and comes from the p=… value in the DMARC DNS TXT record field. It has a limited set of values, so let’s enumerate them and assign some colors:

dmarc\_levels <- c("No DMARC", "None", "Quarantine", "Reject")

dmarc\_cols <- set\_names(c(ft\_cols$slate, "#a6dba0", "#5aae61", "#1b7837"), dmarc\_levels)

We want the aggregate value of each p, thus we need to do count counting:

(dmarc\_summary <- count(f500\_dmarc, industry, p))

## # A tibble: 63 x 3

## industry p n

##

## 1 Aerospace & Defense No DMARC 9

## 2 Aerospace & Defense None 3

## 3 Aerospace & Defense Quarantine 1

## 4 Apparel No DMARC 4

## 5 Apparel None 1

## 6 Business Services No DMARC 9

## 7 Business Services None 7

## 8 Business Services Reject 4

## 9 Chemicals No DMARC 12

## 10 Chemicals None 2

## # … with 53 more rows

We’re also going to want to sort the industries by those with the most DMARC (sorted bars/chicklets FTW!). We’ll need a factor for that, so let’s make one:

(dmarc\_summary %>%

filter(p != "No DMARC") %>% # we don't care abt this `p` value

count(industry, wt=n, sort=TRUE) -> industry\_levels)

## # A tibble: 21 x 2

## industry n

##

## 1 Financials 54

## 2 Technology 25

## 3 Health Care 24

## 4 Retailing 23

## 5 Wholesalers 16

## 6 Energy 12

## 7 Transportation 12

## 8 Business Services 11

## 9 Industrials 8

## 10 Food, Beverages & Tobacco 6

## # … with 11 more rows

Now, we can make the chart:

dmarc\_summary %>%

mutate(p = factor(p, levels = rev(dmarc\_levels))) %>%

mutate(industry = factor(industry, rev(industry\_levels$industry))) %>%

ggplot(aes(industry, n)) +

geom\_chicklet(aes(fill = p)) +

scale\_fill\_manual(name = NULL, values = dmarc\_cols) +

scale\_y\_continuous(expand = c(0,0), position = "right") +

coord\_flip() +

labs(

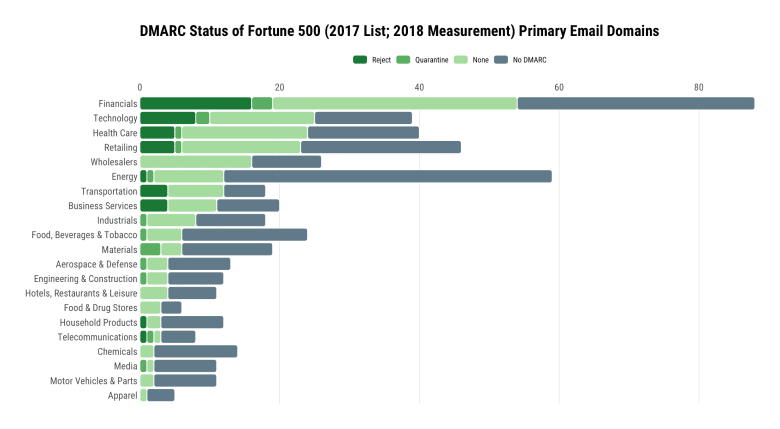
x = NULL, y = NULL,

title = "DMARC Status of Fortune 500 (2017 List; 2018 Measurement) Primary Email Domains"

) +

theme\_ipsum\_rc(grid = "X") +

theme(legend.position = "top")



Doh! We rly want them to be 100% width. Thankfully, {ggplot2} has a position\_fill() we can use instead of position\_dodge():

dmarc\_summary %>%

mutate(p = factor(p, levels = rev(dmarc\_levels))) %>%

mutate(industry = factor(industry, rev(industry\_levels$industry))) %>%

ggplot(aes(industry, n)) +

geom\_chicklet(aes(fill = p), position = position\_fill()) +

scale\_fill\_manual(name = NULL, values = dmarc\_cols) +

scale\_y\_continuous(expand = c(0,0), position = "right") +

coord\_flip() +

labs(

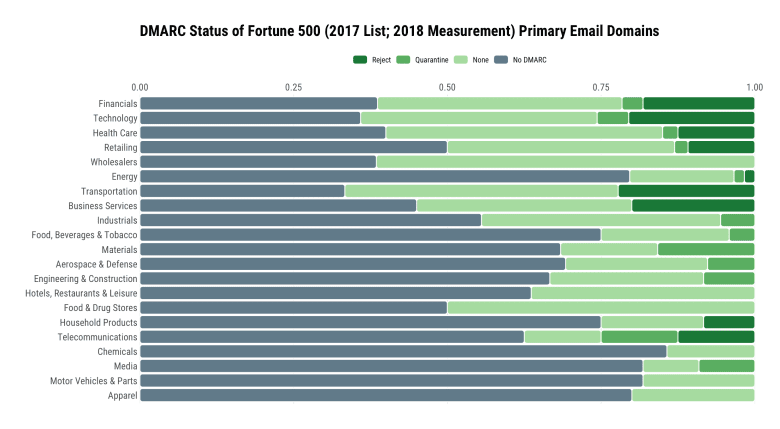
x = NULL, y = NULL,

title = "DMARC Status of Fortune 500 (2017 List; 2018 Measurement) Primary Email Domains"

) +

theme\_ipsum\_rc(grid = "X") +

theme(legend.position = "top")



Doh! Even though we forgot to use reverse = TRUE in the call to position\_fill() everything is out of order. *Kinda*. It’s in the order we told it to be in, but that’s not right b/c we need it ordered by the in-industry percentages. If each industry had the same number of organizations, there would not have been an issue. Unfortunately, the folks who make up these lists care not about our time. Let’s re-compute the industry factor by computing the percents:

(dmarc\_summary %>%

group\_by(industry) %>%

mutate(pct = n/sum(n)) %>%

ungroup() %>%

filter(p != "No DMARC") %>%

count(industry, wt=pct, sort=TRUE) -> industry\_levels)

## # A tibble: 21 x 2

## industry n

##

## 1 Transportation 0.667

## 2 Technology 0.641

## 3 Wholesalers 0.615

## 4 Financials 0.614

## 5 Health Care 0.6

## 6 Business Services 0.55

## 7 Food & Drug Stores 0.5

## 8 Retailing 0.5

## 9 Industrials 0.444

## 10 Telecommunications 0.375

## # … with 11 more rows

Now, we can go back to using position\_fill() as before:

dmarc\_summary %>%

mutate(p = factor(p, levels = rev(dmarc\_levels))) %>%

mutate(industry = factor(industry, rev(industry\_levels$industry))) %>%

ggplot(aes(industry, n)) +

geom\_chicklet(aes(fill = p), position = position\_fill(reverse = TRUE)) +

scale\_fill\_manual(name = NULL, values = dmarc\_cols) +

scale\_y\_percent(expand = c(0, 0.001), position = "right") +

coord\_flip() +

labs(

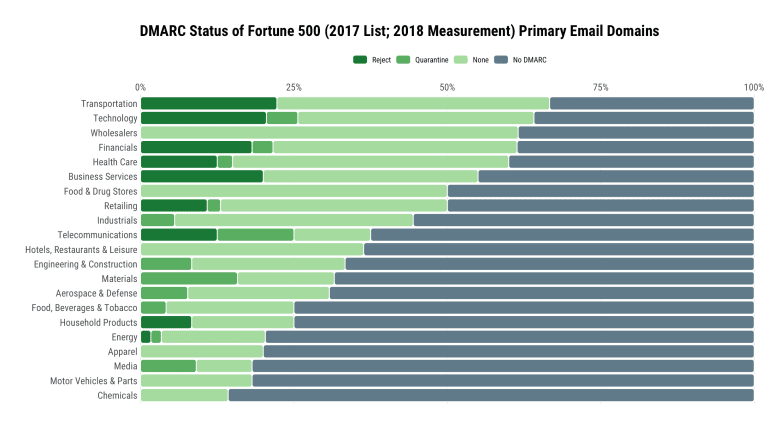
x = NULL, y = NULL,

title = "DMARC Status of Fortune 500 (2017 List; 2018 Measurement) Primary Email Domains"

) +

theme\_ipsum\_rc(grid = "X") +

theme(legend.position = "top")



**FIN**

As noted, this is one way to handle this situation. I’m not super happy with the final visualization here as it doesn’t have the counts next to the industry labels and I like to have the ordering by both count and more secure configuration (so, conditional on higher prevalence of Quarantine or Reject when there are ties). That is an exercise left to the reader