Cybersecurity case study, Part 2

425/625

2023-11-06

In part 1 you extract data about vulnerabilities and whether or not they have been exploited. Suppose your supervisor has never seen the data and wants you to give a brief summary of relationships among the vulnerability characteristics and whether or not they have been exploited. Your supervisor needs a quick summary for a meeting this afternoon. Perform some data exploration and analysis to help your supervisor understand the data and prepare for the meeting.

Deliverables

Upload a PDF to Gradescope. Include a bullet-point summary at the top of the PDF that summarizes your main takeaways. Including supporting code and results below that.

```
## set your working directory
# setwd("~/Desktop/Year_2_Sem_1/SDS625/Case Studies/425-625-Fall-2023-main/cyber/R")
```

Data Exploration - read the file

```
cve_exploits <- readRDS("../data/cve.with.exploits.rds")
head(cve_exploits)
```

```
##              id     sourceIdentifier  published lastModified vulnStatus
## 1 CVE-1999-0236        cve@mitre.org 1997-01-01   2022-08-17   Modified
## 2 CVE-2004-1865        cve@mitre.org 2004-03-26   2020-12-08   Analyzed
## 3 CVE-2008-1447 secure@microsoft.com 2008-07-08   2020-03-24   Analyzed
## 4 CVE-2008-2812  secalert@redhat.com 2008-07-09   2023-02-13   Modified
## 5 CVE-2008-2931  secalert@redhat.com 2008-07-09   2023-02-13   Modified
## 6 CVE-2008-3275  secalert@redhat.com 2008-08-12   2023-02-13   Modified
##         source    type                                vectorString
## 1 nvd@nist.gov Primary CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N
## 2 nvd@nist.gov Primary CVSS:3.1/AV:N/AC:L/PR:H/UI:R/S:C/C:L/I:L/A:N
## 3 nvd@nist.gov Primary CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:C/C:N/I:H/A:N
## 4 nvd@nist.gov Primary CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
## 5 nvd@nist.gov Primary CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
## 6 nvd@nist.gov Primary CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H
##   attackVector attackComplexity privilegesRequired userInteraction     scope
## 1      NETWORK              LOW               NONE            NONE UNCHANGED
## 2      NETWORK              LOW               HIGH        REQUIRED   CHANGED
## 3      NETWORK             HIGH               NONE            NONE   CHANGED
## 4        LOCAL              LOW                LOW            NONE UNCHANGED
## 5        LOCAL              LOW                LOW            NONE UNCHANGED
## 6        LOCAL              LOW                LOW            NONE UNCHANGED
##   confidentialityImpact integrityImpact availabilityImpact baseScore
## 1                  HIGH            NONE               NONE       7.5
## 2                   LOW             LOW               NONE       4.8
```

```
## 3                  NONE            HIGH               NONE       6.8
## 4                  HIGH            HIGH               HIGH       7.8
## 5                  HIGH            HIGH               HIGH       7.8
## 6                  NONE            NONE               HIGH       5.5
##   baseSeverity exploitabilityScore impactScore weaknesses.source
## 1         HIGH                 3.9         3.6      nvd@nist.gov
## 2       MEDIUM                 1.7         2.7      nvd@nist.gov
## 3       MEDIUM                 2.2         4.0      nvd@nist.gov
## 4         HIGH                 1.8         5.9      nvd@nist.gov
## 5         HIGH                 1.8         5.9      nvd@nist.gov
## 6       MEDIUM                 1.8         3.6      nvd@nist.gov
##   weaknesses.type   value exploited
## 1         Primary CWE-200         1
## 2         Primary  CWE-79         0
## 3         Primary CWE-331         1
## 4         Primary CWE-476         0
## 5         Primary CWE-269         0
## 6         Primary CWE-120         0
```

```
str(cve_exploits)
```

```
## 'data.frame':    99272 obs. of  24 variables:
##  $ id                    : chr  "CVE-1999-0236" "CVE-2004-1865" "CVE-2008-1447" "CVE-2008-2812" ...
##  $ sourceIdentifier      : chr  "cve@mitre.org" "cve@mitre.org" "secure@microsoft.com" "secalert@redha
##  $ published             : POSIXct, format: "1997-01-01" "2004-03-26" ...
##  $ lastModified          : POSIXct, format: "2022-08-17" "2020-12-08" ...
##  $ vulnStatus            : chr  "Modified" "Analyzed" "Analyzed" "Modified" ...
##  $ source                : chr  "nvd@nist.gov" "nvd@nist.gov" "nvd@nist.gov" "nvd@nist.gov" ...
##  $ type                  : chr  "Primary" "Primary" "Primary" "Primary" ...
##  $ vectorString          : chr  "CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N" "CVSS:3.1/AV:N/AC:L/PR
##  $ attackVector          : chr  "NETWORK" "NETWORK" "NETWORK" "LOCAL" ...
##  $ attackComplexity      : chr  "LOW" "LOW" "HIGH" "LOW" ...
##  $ privilegesRequired    : chr  "NONE" "HIGH" "NONE" "LOW" ...
##  $ userInteraction       : chr  "NONE" "REQUIRED" "NONE" "NONE" ...
##  $ scope                 : chr  "UNCHANGED" "CHANGED" "CHANGED" "UNCHANGED" ...
##  $ confidentialityImpact: chr  "HIGH" "LOW" "NONE" "HIGH" ...
##  $ integrityImpact       : chr  "NONE" "LOW" "HIGH" "HIGH" ...
##  $ availabilityImpact    : chr  "NONE" "NONE" "NONE" "HIGH" ...
##  $ baseScore             : num  7.5 4.8 6.8 7.8 7.8 5.5 5.5 8.8 7.8 7.8 ...
##  $ baseSeverity          : chr  "HIGH" "MEDIUM" "MEDIUM" "HIGH" ...
##  $ exploitabilityScore   : num  3.9 1.7 2.2 1.8 1.8 1.8 1.8 2.8 1.8 1.8 ...
##  $ impactScore           : num  3.6 2.7 4 5.9 5.9 3.6 3.6 5.9 5.9 5.9 ...
##  $ weaknesses.source     : chr  "nvd@nist.gov" "nvd@nist.gov" "nvd@nist.gov" "nvd@nist.gov" ...
##  $ weaknesses.type       : chr  "Primary" "Primary" "Primary" "Primary" ...
##  $ value                 : chr  "CWE-200" "CWE-79" "CWE-331" "CWE-476" ...
##  $ exploited             : num  1 0 1 0 0 0 0 1 1 0 ...
```

Explaining the concept behind this:

1.The Base Score in CVSS represents the severity of a vulnerability based on its original characteristics. It provides a numerical value that indicates the potential impact and exploitability of a vulnerability. The Base Score is independent of any specific environment and reflects the vulnerability's worst-case scenario. 2. Impact score allows you to find the highly-exploitable CVEs with the biggest impact on your company's Risk Index. 3. The exploitability score represents how easily the vulnerability is accessed by intruder, the complexity of the attack, and the number of times an attacker must authenticate to successfully exploit a vulnerability. Lower

base scores indicate less severe vulnerabilities, lower impact scores have less significant impact on security, lower exploitability score means it's harder for attacker to exploit the data. Hence we want vulnerabilities with lower scores in all categories.

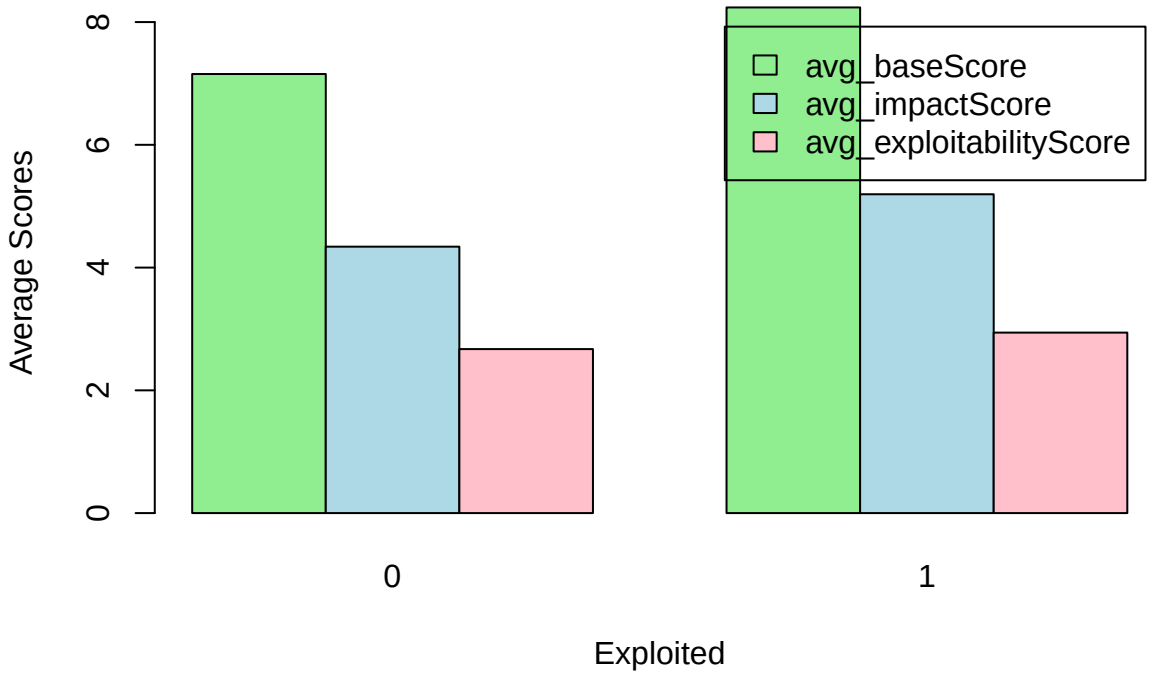
Create summary

```
#colnames(cve_exploits)

summary_exploited <- cve_exploits %>%
  group_by(exploited) %>%
  summarise(
    avg_baseScore = mean(baseScore, na.rm = TRUE),
    avg_impactScore = mean(impactScore, na.rm = TRUE),
    avg_exploitabilityScore = mean(exploitabilityScore, na.rm = TRUE)
  )


summary_exploited
```

```
## # A tibble: 2 x 4
##   exploited avg_baseScore avg_impactScore avg_exploitabilityScore
##       <dbl>         <dbl>           <dbl>                   <dbl>
## 1         0          7.15            4.34                    2.67
## 2         1          8.24            5.19                    2.94
```

```
summary_exploited_df <- as.data.frame(summary_exploited)
```

```
# Create a barplot
barplot(
  t(summary_exploited_df[, -1]),  # Transpose the data, excluding the 'exploited' column
  beside = TRUE, # side by side
  col = c("lightgreen", "lightblue", "pink"),
  names.arg = summary_exploited_df$exploited,
  xlab = "Exploited",
  ylab = "Average Scores",
  main = "Comparison of Average Scores by Exploited Status",
  legend.text = rownames(t(summary_exploited_df[, -1]))
)
```




```
avg_basescore_diff <- summary_exploited_df$avg_baseScore[2] - summary_exploited_df$avg_baseScore[1]
avg_impactscore_diff <- summary_exploited_df$avg_impactScore[2] - summary_exploited_df$avg_impactScore[
avg_explotabilityscore_diff <- summary_exploited_df$avg_exploitabilityScore[2] - summary_exploited_df$a
```

```
avg_basescore_diff
```

```
## [1] 1.085072
```

```
avg_impactscore_diff
```

```
## [1] 0.8540082
```

```
avg_explotabilityscore_diff
```

```
## [1] 0.2684032
```

We notice and confirm the following: * Generally the scores for base, impact, and exploitability are lower when said ID is not exploited * This is a general correlation to keep in mind- the lower the score, the less likely it will be exploited

Here are the differences amongst the type of scores: 1. base score- 1.085072 2. impact score- 0.8540082 3. exploitability score- 0.2684032

Studying the number of exploited vs not-exploited ID's

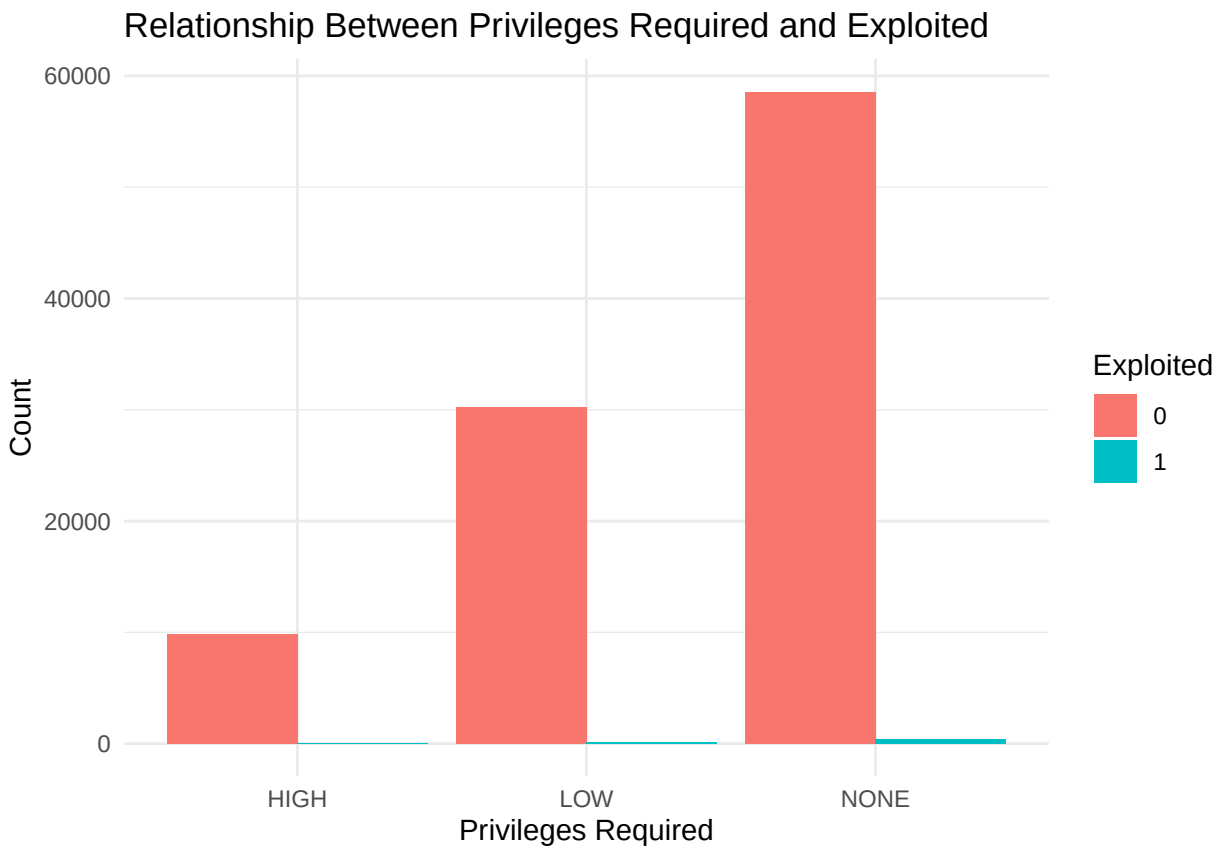
```
fraction_exploited <- cve_exploits %>%
  summarize(Fraction_Exploited = mean(exploited))

fraction_exploited
```

```
##   Fraction_Exploited
```

```
## 1         0.006245467
```

- Note: Only 0.625% of the ID's are exploited
- In the graph below, let's try to visualize the number of exploited vs non-exploited data and let's normalize it since such a small portion of the data is exploited anyways.

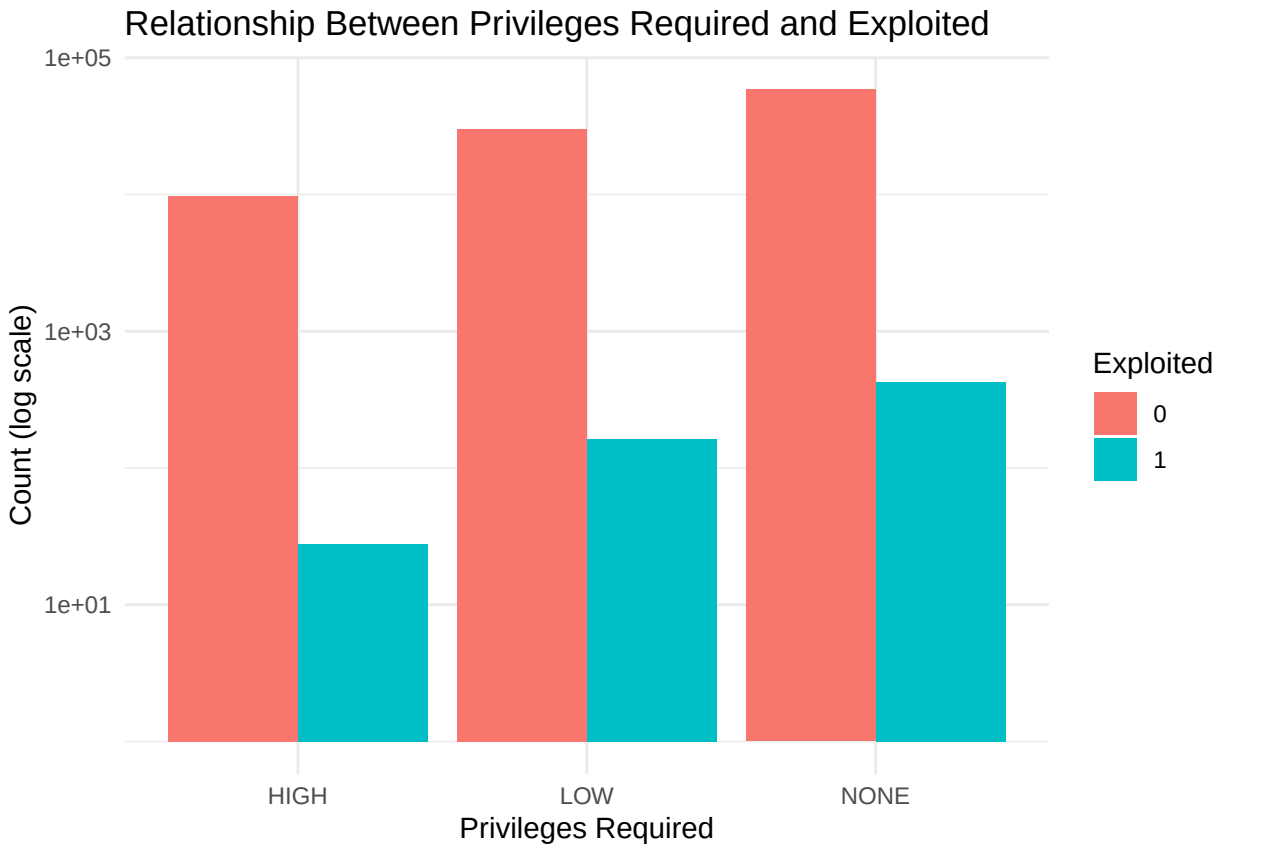
```
# Create a barplot to visualize the relationship
ggplot(cve_exploits, aes(x = privilegesRequired, fill = factor(exploited))) +
  geom_bar(position = "dodge") +
  labs(
    x = "Privileges Required",
    y = "Count",
    fill = "Exploited"
  ) +
  ggtitle("Relationship Between Privileges Required and Exploited") +
  theme_minimal()
```



Now let's log normalize so we can better see the comparisons:

```
# Create a barplot to visualize the relationship with log-transformed counts
ggplot(cve_exploits, aes(x = privilegesRequired, fill = factor(exploited))) +
  geom_bar(position = "dodge") +
  labs(
    x = "Privileges Required",
    y = "Count (log scale)",
    fill = "Exploited"
  ) +
  ggtitle("Relationship Between Privileges Required and Exploited") +
  theme_minimal() +
```

```
  scale_y_log10()  # Apply log scale to the y-axis
```




```
exploited_fraction <- cve_exploits %>%
  group_by(privilegesRequired) %>%
  summarize(exploited_fraction = mean(exploited)) %>%
  arrange(desc(exploited_fraction))

exploited_fraction
```

```
## # A tibble: 3 x 2
##   privilegesRequired exploited_fraction
##   <chr>                           <dbl>
## 1 NONE                          0.00725
## 2 LOW                           0.00539
## 3 HIGH                          0.00285
```

We see that overall, most of the vulnerabilities happen to have no privileges. In fact the total priviledges are in descending order from no privileges to low privileges to high privileges. However, the lowest portion of exploited data is with the category of high privileges with 0.28% after the log scale normalization and the highest portion of exploited data is with no priviledges with 0.72% It's a slight increase- so that may be interesting for the company to conduct a risk analysis.

Creating a model

Let's try creating a regression:

```
# Select the columns to include as predictors (independent variables)
selected_columns <- c(
```

```
  "baseScore", "impactScore", "exploitabilityScore"
)

# Create a new dataframe with selected columns and prepare for modellign
selected_data <- cve_exploits %>% select(exploited, all_of(selected_columns))
selected_data <- selected_data %>%
  mutate_if(is.factor, as.integer)

# Split the data into training and testing sets
set.seed(123)  # For reproducibility
train_indices <- sample(1:nrow(selected_data), 0.7 * nrow(selected_data))
train_data <- selected_data[train_indices, ]
test_data <- selected_data[-train_indices, ]

# Create the regression model
model <- glm(exploited ~ ., data = train_data, family = binomial)
model
```

```
##
## Call:  glm(formula = exploited ~ ., family = binomial, data = train_data)
##
## Coefficients:
##         (Intercept)            baseScore          impactScore
##             -8.3704               0.2009               0.2809
## exploitabilityScore
##              0.1234
##
## Degrees of Freedom: 69489 Total (i.e. Null);  69486 Residual
## Null Deviance:       5078
## Residual Deviance: 4883  AIC: 4891
```

```
predictions <- predict(model, newdata = test_data, type = "response")

# Convert predicted probabilities to binary (0 or 1)
predicted_labels <- ifelse(predictions >= 0.5, 1, 0)

# Evaluate the model's performance
accuracy <- mean(predicted_labels == test_data$exploited)
cat("Accuracy on the test set:", accuracy, "\n")
```

```
## Accuracy on the test set: 0.9931166
```

Analysis **Base Score**: This coefficient is 0.2009. For every unit increase in the Base Score, the predicted probability of a vulnerability being exploited increases by 0.2009 units. **Impact Score**: This coefficient is 0.2809.For every unit increase in the Impact Score, the predicted probability of exploitation goes up by 0.2809 units. **Exploitability Score**: This coefficient is 0.1234. If the Exploitability Score goes up by one unit, the predicted probability of exploitation increases by 0.1234 units.

As the any of these scores goes up, the risk of exploitation also goes up by their corresponding values, meaning higher risk and vulnerability. This is also a strong model since our accuracy on the test set is: 99%.

Feel free to follow up if you have any questions regarding the analysis.