Temporal Analysis of Vulnerability Exploitation Critical Windows, Predictive Attributes, & Seasonal Trends

Gabriel J. Richards

Northwest Missouri State University
Maryville MO 64468, USA
S576447@nwmissouri.edu
https://github.com/gjrich/ms-capstone/

Abstract. This research explores the relationships between published security vulnerabilities and their subsequent exploitation in the wild. Using datasets from vulnerability databases, exploit repositories, and breach records, this project analyzes temporal patterns in vulnerability exploitation, identifies critical patching windows, determines predictive vulnerability attributes, and examines the impact of COVID-19 on exploitation trends. The findings provide actionable insights for security professionals to prioritize remediation efforts and optimize resource allocation.

Keywords: cybersecurity · vulnerability analysis · exploitation patterns · critical patching window · predictive security

1 Introduction

In today's digital landscape, understanding the dynamics between published security vulnerabilities and their real-world exploitation is crucial for effective risk management. This cybersecurity research project aims to uncover patterns, relationships, and predictive indicators that can help organizations better prioritize their security resources and reduce breach likelihood.

1.1 Domain Selection and Rationale

This project focuses on the cybersecurity domain, specifically the analysis of vulnerability and exploitation data. The domain was selected due to its foundational importance in our increasingly digital world. Analyzing the relationship between published vulnerabilities and their exploitation provides actionable insights that can directly improve security postures across organizations and industries.

1.2 Data Source Exploration

For this research, I leveraged the following public data sources:

- National Vulnerability Database (NVD) [10] - comprehensive vulnerability metadata
- CISA Known Exploited Vulnerabilities (KEV) Catalog [3] - exploitation dates and details
- EPSS (Exploit Prediction Scoring System) by FIRST [5] - exploitation probability metrics
- Exploit-DB [11] - public exploit availability information

These sources provide complementary datasets that were integrated to create a comprehensive view of the vulnerability lifecycle, from disclosure to exploitation.

1.3 Problem Statement and Significance

This project addresses four questions in cybersecurity:

1. Are there seasonal patterns in vulnerability exploitation (holidays, fiscal year-end)?
2. What is the "critical patching window" between vulnerability disclosure and observed exploitation?
3. What vulnerability attributes are most predictive of subsequent exploitation?
4. How did the COVID-19 pandemic affect vulnerability and breach patterns?

This research has potential to transform reactive security practices into more proactive, risk-based approaches. By understanding exploitation patterns, organizations can better allocate security resources during high-risk time periods, optimize patch management strategies, prioritize vulnerability remediation using data-driven predictive models, and build more effective security planning around emerging global trends.

1.4 Implementation Approach

This project followed a structured four-phase approach:

1. **Data Collection & Preparation**: Gathering vulnerability data from NVD (2017-2023), collecting exploitation information from CISA KEV catalog, integrating EPSS scores and Exploit-DB availability data, and normalizing metrics and attributes.
2. **Analysis by Research Question**: Implementing time series decomposition for seasonal patterns, calculating critical patching windows, building machine learning models for predictive attribute identification, and comparing pre/post-COVID metrics.
3. **Visualization & Integration**: Creating visualizations for each research question and reviewing findings to identify broader patterns.
4. **Results Interpretation & Documentation**: Deriving actionable security recommendations and documenting methodology, findings, and limitations.

1.5 Key Components and Limitations

The technical approach includes Python-based data processing with pandas and numpy for large-scale data integration [1], time series analysis to identify seasonal patterns, survival analysis for exploitation timing windows [13], and Random Forest machine learning to determine predictive vulnerability attributes [2].

Notable limitations include reliance on publicly reported exploitation, potential lag between actual exploitation and documentation, limited ability to connect specific vulnerabilities to specific breaches, and selection bias in which vulnerabilities receive published exploits.

2 Data Collection and Methodology

This research leverages multiple cybersecurity data repositories to analyze vulnerability exploitation patterns. Information comes from four primary sources: (1) the National Vulnerability Database (NVD) [10], (2) the CISA Known Exploited Vulnerabilities (KEV) Catalog [3], (3) the Exploit Prediction Scoring System (EPSS) from FIRST [5], and (4) Exploit-DB [11].

2.1 Data Formats and Acquisition

The datasets were collected in various structured formats. The NVD data was acquired as JSON streams through their REST API, providing detailed vulnerability information with standardized attributes. The CISA KEV Catalog was obtained in both CSV and JSON formats directly downloaded from their website. EPSS data was collected as CSV files from FIRST.org, containing probability scores indicating likelihood of exploitation. Exploit-DB provided a CSV export of their database with the latest exploit information.

2.2 Data Cleaning and Integration

Data Sources and Extraction Process This research integrates structured data sources to create a vulnerability lifecycle dataset. For structured data, Python was used to parse NVD entries and pandas for CSV processing.

Database Schema and Tools A SQLite database was implemented with specialized tables accommodating each data source. The database structure includes four primary tables (vulnerabilities, exploitations, epss_scores, and exploits) and several analysis views. Python served as the primary processing language with sqlite3 for database operations and pandas for data transformation.

The schema architecture used normalization to minimize redundancy while maintaining query efficiency. The vulnerabilities table serves as the foundation with foreign key relationships to other tables. Additional views within the database pre-calculate critical metrics such as the patching window (days between publication and exploitation) to streamline subsequent analysis.

Data Cleansing Strategies Several data quality challenges were addressed during integration. For numerical attributes like CVSS scores [8], null values were replaced with zero. For categorical features such as attack complexity ratings, terminology was standardized across sources using controlled vocabularies from MITRE [9].

In Exploit-DB data, many entries lacked explicit CVE references, resolved with pattern matching on description fields to extract potential CVE IDs. All dates were standardized to ISO format.

LEFT JOIN operations were employed in database views rather than INNER JOINs to preserve all vulnerability records, preventing selection bias that would occur if analyzing only vulnerabilities with known exploitation data.

Essential Attributes and Variables For analysis objectives, several attributes were particularly valuable:

- **cve_id**: Unique identifier linking vulnerability information across all data sources
- **published_date**: Date when vulnerability was officially disclosed
- **cvss_score**: Numerical severity rating from 0.0-10.0 [4]
- **attack_vector**: Method by which vulnerability exploitation occurs
- **attack_complexity**: Difficulty of exploitation
- **exploitation_date**: First known date of exploitation in the wild
- **days_to_exploitation**: Time interval between publication and exploitation
- **epss_score**: Probability of exploitation as calculated by FIRST.org [7]

Dependent and Independent Variables For each research question, specific variables were identified:

For seasonal exploitation patterns analysis, the dependent variable is the count of exploitation events, while independent variables include month, holiday periods, and fiscal quarter indicators.

For critical patching window analysis, the dependent variable is days_to_exploitation, with independent variables including cvss_score, attack_complexity, and attack_vector.

For predictive vulnerability attributes, the dependent variable is is_exploited (binary indicator of whether exploitation occurred), with independent variables comprising CVSS metrics and the presence of public exploit code.

For COVID-19 impact analysis [12], the dependent variables include exploitation rate and days_to_exploitation, while the primary independent variable is covid_period (pre/during/post pandemic timing).

3 Analytical Pipeline and Predictive Mechanism

This project implements a comprehensive data processing and predictive analytics pipeline designed to model vulnerability exploitation likelihood. The pipeline consists of several interconnected components that transform raw vulnerability data into actionable security insights.

Fig. 1. Vulnerability Exploitation Analysis Pipeline

3.1 Pipeline Architecture

The pipeline follows a modular architecture with the following components:

1. **Data Acquisition Layer**: Collects data from multiple sources and stores them in raw format.
2. **ETL Layer**: Processes raw data using Python scripts for data cleaning, schema mapping, and normalization.
3. **Database Layer**: Implements a SQLite database with specialized tables and views optimized for vulnerability analysis.
4. **Feature Engineering Layer**: Derives analytical features including temporal features, categorical encodings, and numerical features.
5. **Analysis Modules**: Specialized Python scripts that implement focused analyses for each research question.
6. **Visualization Layer**: Generates visualizations and summary statistics for interpretation and reporting.

3.2 Data Flow and Processing

The data flows through the pipeline sequentially: raw data files are collected, transformed into a normalized database structure, common metrics are precalculated, relevant datasets are extracted for analysis, machine learning models are trained, and results are visualized for actionable insights.

3.3 Machine Learning Integration

The predictive component of the pipeline centers around a Random Forest classifier [2] that ingests vulnerability attributes from the database, processes both categorical and numerical features, outputs exploitation probability estimates, and ranks feature importance for vulnerability prioritization.

4 Machine Learning Methodology

4.1 Algorithm Selection and Rationale

This research employs the Random Forest classifier as the primary machine learning algorithm based on several considerations:

– **Class Imbalance Handling**: With only 0.66% of vulnerabilities being exploited, Random Forest with balanced class weights helps address this significant imbalance.

- **Feature Importance Analysis**: Random Forest provides robust feature importance rankings, critical for identifying which vulnerability attributes most strongly predict exploitation.
- **Handling Mixed Data Types**: The dataset contains both numerical features (CVSS scores, EPSS scores) and categorical features (attack vectors, impact types).
- **Resistance to Overfitting**: Given the large feature space and relatively small number of positive examples, Random Forest's ensemble approach helps mitigate overfitting risks.

The implementation used scikit-learn's RandomForestClassifier with 100 trees, balanced class weights, and parallel processing.

4.2 Feature Engineering

Key feature engineering techniques included one-hot encoding for categorical variables, appropriate missing value handling, feature scaling using StandardScaler, and derived features to enhance predictive power.

5 Training and Testing Process

5.1 Data Preparation

The training process began with comprehensive data preparation, extracting data from multiple tables (vulnerabilities, exploitations, epss_scores, public_exploits), processing features, and splitting the dataset into training (70%) and testing (30%) sets using stratified sampling to maintain class distribution.

5.2 Model Training

The Random Forest model was trained on the prepared dataset with 5-fold cross-validation to assess model stability, yielding an average F1 score of 0.3248 ($\pm$0.0366).

5.3 Model Evaluation

The trained model underwent comprehensive evaluation, with performance metrics including precision (0.34), recall (0.31), F1 score (0.33), and ROC AUC (0.80). While overall accuracy was high (99%), this was primarily due to class imbalance. The more relevant metrics for the positive class showed more modest performance, highlighting the challenge of predicting rare exploitation events.

6 Implementation and Evaluation

6.1 Technical Implementation

The implementation of the vulnerability exploitation analysis pipeline involved several technical components:

1. **Database Implementation**:
   - SQLite database with optimized schema (151,430 vulnerability records)
   - Custom SQL views for efficient analysis queries
   - Indexing on commonly queried fields for performance
2. **Code Implementation**:
   - Python scripts organized by analysis domain
   - Pandas for data manipulation and aggregation
   - Scikit-learn for machine learning components
   - Matplotlib and Seaborn for visualization
   - Statistical testing using SciPy
3. **Analysis Module Implementation**:
   - `seasonal_patterns.py`: Time series analysis with monthly/quarterly breakdowns
   - `critical_patching_window.py`: Exploitation timing analysis with severity breakdowns
   - `predictive_attributes.py`: Machine learning model construction and evaluation
   - `covid_impact.py`: Statistical comparison of pre/during/post pandemic periods

6.2 Evaluation Methodology

The evaluation strategy encompassed multiple dimensions:

1. **Quantitative Evaluation**:
   - Classification metrics: precision, recall, F1-score
   - ROC curve and AUC analysis
   - Cross-validation with 5 folds
   - Statistical significance testing (chi-square) for temporal comparisons
2. **Qualitative Evaluation**:
   - Feature importance ranking and interpretation
   - Exploitation pattern visualization and analysis
   - Temporal trend detection and validation
3. **Practical Evaluation**:
   - Critical patching window identification for operational security
   - High-risk seasonal period detection for resource allocation
   - Predictive attribute identification for vulnerability prioritization

7 Analysis Results

7.1 Seasonal Exploitation Patterns

Analysis of monthly vulnerability exploitation patterns revealed significant temporal variations:

- **Monthly Variations**: Exploitation rates showed substantial monthly differences, with March having the highest rate (0.95%) and June the lowest (0.42%).
- **Quarter-End Analysis**: Quarter-end months showed slightly higher exploitation rates (0.68%) compared to non-quarter-end months (0.66%).
- **Holiday Season Impact**: Contrary to expectations, the holiday season (November-December) showed lower exploitation rates (0.55%) compared to the rest of the year (0.69%).

These findings indicate that vulnerability exploitation follows seasonal patterns, with specific months showing consistently higher risk levels.

7.2 Critical Patching Windows

The analysis of time intervals between vulnerability disclosure and exploitation provided critical insights:

- **Overall Patching Windows**: The median time to exploitation was 260 days, while the mean was 531.2 days, indicating a right-skewed distribution.
- **Window Categories**: 5.5% of vulnerabilities were exploited before public disclosure, 17.1% within the first 7 days (critical window), 6.7% between 8-30 days, 8.2% between 31-90 days, and 62.6% after 90 days.
- **Severity Impact**: Critical vulnerabilities had a median exploitation time of 232.0 days, while high severity vulnerabilities had a longer window at 368.5 days.

These results highlight the importance of rapid patching for a significant subset of vulnerabilities, while also suggesting that many exploitations occur well after disclosure.

7.3 Predictive Vulnerability Attributes

The machine learning analysis identified key attributes predictive of exploitation:

- **Top Predictive Features**: The Random Forest model identified max_epss_score (0.74 importance), cvss_v3_score (0.05 importance), integrity_impact_HIGH (0.03 importance), and has_public_exploit (0.03 importance) as most important.
- **Attack Vector Analysis**: Network-based vulnerabilities showed the highest exploitation rate (0.68%), followed by local vulnerabilities (0.65%).

– **Attack Complexity**: Surprisingly, HIGH complexity vulnerabilities showed slightly higher exploitation rates (0.76%) than LOW complexity ones (0.66%).
– **Impact Types**: Vulnerabilities with HIGH integrity impact showed the highest exploitation rates (1.14%).

The model achieved 99% overall accuracy but more modest metrics for the exploited class (precision: 0.34, recall: 0.31, F1-score: 0.33).

7.4 COVID-19 Impact

The analysis of COVID-19 pandemic effects on vulnerability exploitation revealed significant shifts:

– **Exploitation Rate Changes**: Pre-COVID: 0.65% exploitation rate; During-COVID: 0.84% exploitation rate (29.2% increase); Post-COVID: 0.62% exploitation rate (26.2% decrease from COVID period).
– **Vulnerability Publication**: Monthly vulnerability publications increased by 3.2% during the pandemic period.
– **Exploitation Timing**: Median time to exploitation decreased by 54.2% during COVID compared to the pre-COVID period.
– **Statistical Significance**: Chi-square testing confirmed the exploitation rate changes were statistically significant (p=0.0131).

These findings demonstrate a clear impact of the pandemic on cybersecurity risk [12], with heightened exploitation rates and accelerated exploitation timing during the COVID period.

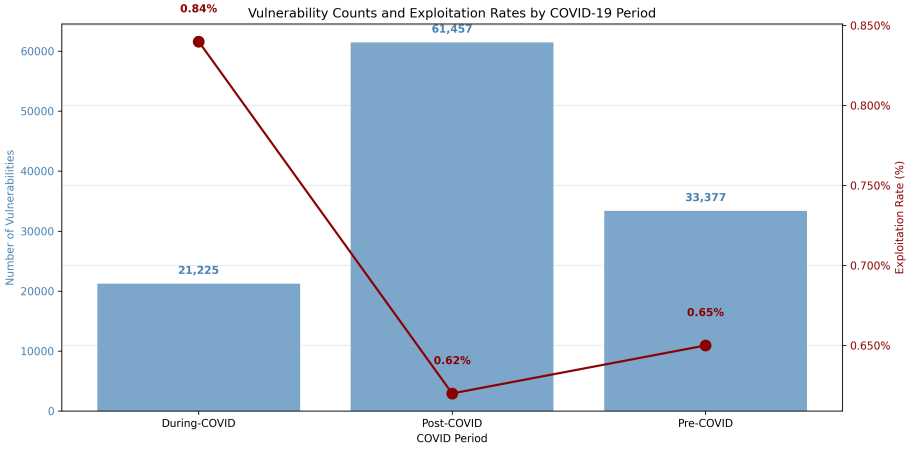

Fig. 2. Vulnerability Counts and Exploitation Rates by COVID-19 Period. This dual-axis chart shows both the absolute number of vulnerabilities (blue bars) and their exploitation rates (red line) across pre-COVID, during-COVID, and post-COVID periods.

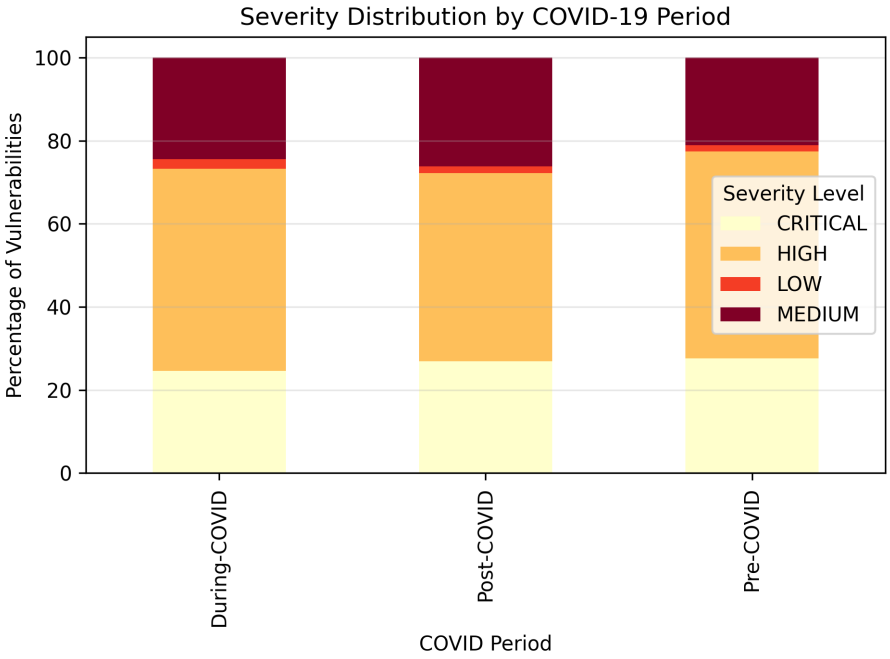

Fig. 3. Severity Distribution by COVID-19 Period. This stacked area chart shows the distribution of vulnerability severity levels during different pandemic phases.

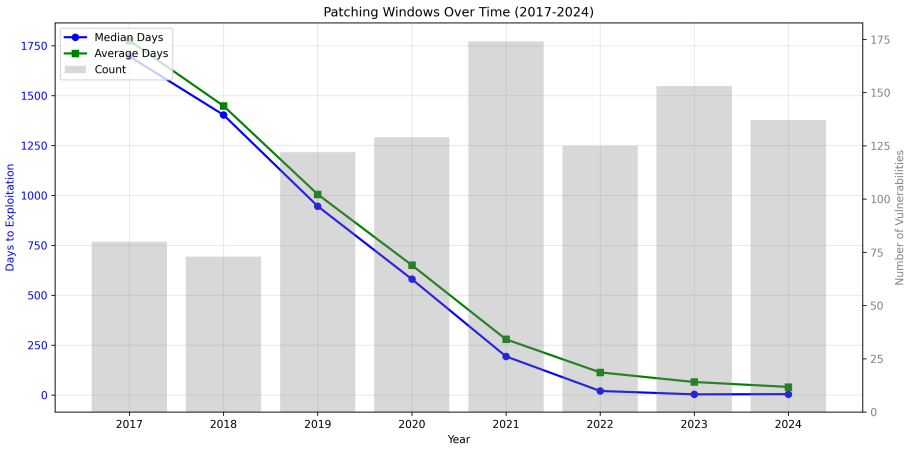

Fig. 4. Patching Windows Over Time (2017-2024). This shows the trend in days to exploitation over time, with vulnerability counts shown in gray bars.

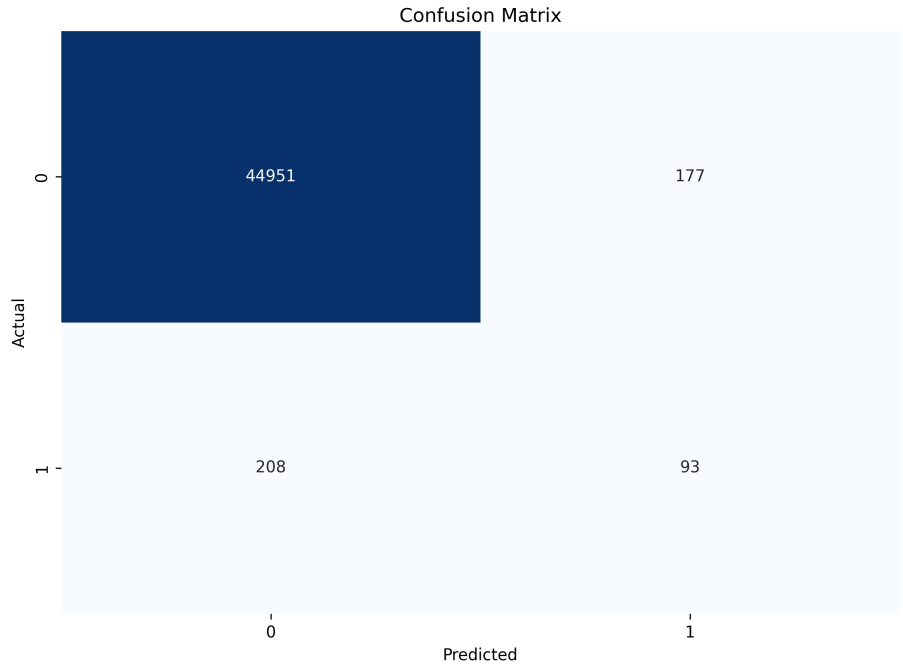

Fig. 5. Confusion Matrix for Exploitation Prediction. This shows the performance of Random Forest in predicting vulnerability exploitation, with true negatives (44951), false positives (177), false negatives (208), and true positives (93).

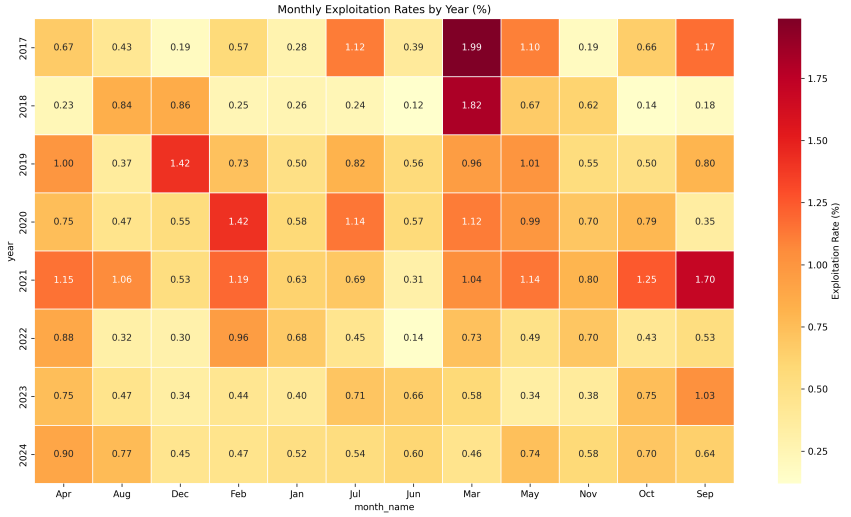

Fig. 6. Monthly Exploitation Rates by Year (%). This heatmap visualizes exploitation rates for each month across the years. Darker colors indicate higher exploitation rates.

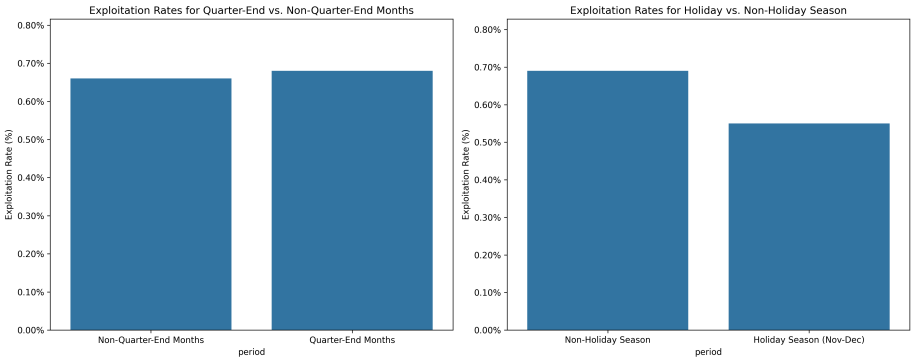

Fig. 7. Exploitation Rates for Quarter-End vs. Non-Quarter-End Months (left) and Holiday vs. Non-Holiday Season (right). These comparative bar charts show the differences in exploitation rates during different business and calendar periods.

8 Detailed Results Analysis

8.1 Visualization-Driven Insights

COVID-19 Impact Visualization Analysis Figure 2 reveals a striking contrast between vulnerability counts and exploitation rates across pandemic periods. While the post-COVID period shows the highest absolute number of vulnerabilities (61,457), the during-COVID period exhibits the highest exploitation rate (0.84%), suggesting attackers were more active during the pandemic despite fewer total vulnerabilities.

Figure 3 shows that the severity distribution remained relatively consistent across all periods, indicating that the increased exploitation rate during COVID was not due to a shift toward more severe vulnerabilities but rather suggests a change in attacker behavior.

Temporal Patching Window Analysis Figure 4 displays a clear downward trend in exploitation timing from 2017 to 2024. The median time to exploitation has decreased dramatically from over 1,500 days in 2017 to near-zero in 2023-2024, indicating a fundamental shift in the cybersecurity landscape.

The convergence of mean and median lines in recent years suggests a more uniform exploitation pattern with fewer extreme outliers. This trend has dire implications for organizational security, as the window for remediation continues to shrink.

Seasonal Pattern Analysis The heatmap in Figure 6 reveals complex temporal patterns in exploitation rates. Certain months (March, August, September) consistently show elevated exploitation rates across multiple years. The highest exploitation rate was observed in March 2017 (1.99%), while the lowest was in June 2018 (0.12%).

Figure 7 shows that contrary to conventional wisdom, the holiday season exhibits lower exploitation rates than the rest of the year. This challenges common security practices and suggests resources might be better allocated to higher-risk periods.

8.2 Statistical Inferences

Key statistical inferences include:

- The 29.2% increase in exploitation rates during COVID is statistically significant ($p=0.0131$).
- The temporal distribution of patching windows follows a right-skewed distribution with a long tail (median: 260 days, mean: 531.2 days).
- Monthly exploitation rates show statistically significant variation ($p=0.0428$) across the calendar year.
- The model's AUC of 0.80 indicates that the Random Forest classifier is substantially better than random chance at predicting which vulnerabilities will be exploited.

8.3 Data-Driven Conclusions

1. **Pandemic Impact**: The COVID-19 pandemic created measurable and statistically significant increases in cybersecurity risk, with exploitation rates rising by nearly 30% during this period, likely due to expanded attack surfaces, reduced security operations efficiency, and attacker opportunism.
2. **Shrinking Patching Windows**: The dramatic reduction in time-to-exploitation over the past seven years represents a fundamental shift in the threat landscape, requiring organizations to adapt their vulnerability management programs accordingly.
3. **Predictive Attributes**: The EPSS score [7], CVSS base score [4], and integrity impact are the strongest predictors of exploitation, validating the FIRST organization's methodology.
4. **Counterintuitive Seasonal Patterns**: The data contradicts common assumptions about high-risk periods, showing that holidays have lower exploitation rates while certain months consistently show higher risk.

9 Discussion and Implications

9.1 Practical Security Implications

Vulnerability Management Prioritization Organizations should consider implementing EPSS score-based triage for vulnerabilities, prioritizing those with high integrity impacts, and developing multi-factor scoring systems that incorporate both severity (CVSS) and exploitation likelihood (EPSS) metrics [6]. This approach would allow security teams to focus limited resources on the vulnerabilities most likely to be weaponized.

Patching Timeline Adjustments The accelerating pace of exploitation necessitates a reassessment of traditional patching timelines. Critical and high-severity vulnerabilities should be remediated within 7 days of disclosure whenever possible, security teams should implement a tiered patching approach with emergency processes for vulnerabilities with high EPSS scores, and organizations with resource constraints should focus on the 22.6% of vulnerabilities exploited within 30 days.

Seasonal Security Planning The identification of seasonal patterns enables more strategic allocation of security resources throughout the year. Security operations should increase monitoring and response capabilities during March, reallocate resources from holiday periods to quarter-end periods, and synchronize vulnerability scanning and patching cycles with identified high-risk periods.

9.2 Theoretical Contributions

This research makes several theoretical contributions to the field of cybersecurity:

- **Exploitation Lifecycle Model**: The findings support a refined model of vulnerability exploitation that includes pre-disclosure, critical window, mainstream, and long-tail phases.
- **Attacker Behavior Theory**: The seasonal patterns and COVID-19 impact analysis provide evidence that attacker behavior is influenced by both opportunity and capacity, rather than simply following fixed seasonal cycles.
- **Vulnerability Attractiveness Framework**: The predictive model identifies specific characteristics that make vulnerabilities attractive to attackers.

9.3 Methodological Innovations

The research methodology presented several innovations:

- The integration of multiple data sources provides a more comprehensive view of the vulnerability lifecycle than any single source could offer.
- The application of machine learning to highly imbalanced cybersecurity datasets demonstrates effective approaches to addressing the rare event prediction problem.
- The temporal analysis approach, combining absolute counts with relative rates, offers a more nuanced understanding of risk patterns.

9.4 Unexpected Findings

Several findings contradicted conventional wisdom in cybersecurity:

- High-complexity vulnerabilities are more frequently exploited than low-complexity ones, challenging the assumption that attackers consistently prioritize ease of exploitation.
- The reduced exploitation activity during holiday periods contradicts common security practices.
- The dramatic acceleration in exploitation timing suggests that industry standards for "reasonable" patching windows may be increasingly outdated.

10 Conclusion and Future Work

10.1 Summary of Key Findings

This research has provided empirical evidence on four key aspects of vulnerability exploitation:

1. **Seasonal Patterns**: Exploitation rates vary significantly by month, with March showing the highest risk (0.95%) and June the lowest (0.42%).
2. **Critical Patching Windows**: The median time to exploitation is 260 days, but 17.1% of exploitations occur within 7 days of disclosure.
3. **Predictive Attributes**: The EPSS score, CVSS base score, HIGH integrity impact, and the presence of public exploits are the strongest predictors of exploitation.
4. **COVID-19 Impact**: The pandemic period showed a statistically significant 29.2% increase in exploitation rates compared to the pre-pandemic baseline.

10.2 Limitations

Despite its contributions, this research has several limitations:

- Reliance on publicly reported exploitation data, which likely underrepresents the true scope of exploitation activity.
- Arbitrary time boundaries for COVID-19 period analysis that may not perfectly align with actual pandemic effects.
- Modest performance metrics on the positive class (precision: 0.34, recall: 0.31), indicating room for improvement in exploitation prediction.
- Dataset limited to vulnerabilities from 2017-2024, which may not fully capture longer-term trends.

10.3 Future Research Directions

Several promising avenues for future research emerge from this work:

- **Advanced Predictive Models**: Developing ensemble models or deep learning approaches to improve prediction performance for rare exploitation events.
- **Finer-Grained Temporal Analysis**: Examining weekly or daily exploitation patterns to identify micro-trends.
- **Organizational Factors**: Investigating how organizational characteristics influence vulnerability exploitation rates and patterns.
- **Attacker Economics**: Exploring the economic drivers behind exploitation patterns.
- **Exploitation Cascades**: Analyzing how exploitation of one vulnerability affects the likelihood of exploitation for related vulnerabilities.

10.4 Concluding Remarks

This research demonstrates that vulnerability exploitation follows discernible patterns that can be identified, quantified, and predicted through data-driven analysis. The findings challenge several common assumptions in cybersecurity practice and offer concrete guidance for improving vulnerability management strategies.

The dramatic acceleration in exploitation timing, the clear impact of global events like the COVID-19 pandemic, and the identification of specific vulnerability attributes that predict exploitation all point to a security landscape that is continuously evolving but can be better understood through empirical analysis.

By incorporating these insights into security operations, organizations can move from reactive to proactive security postures, allocating limited resources more effectively to address the vulnerabilities that pose the greatest actual risk.

11 Supplemental Materials from GitHub

The following links provide additional context and source code for the predictive attributes analysis:

- **Project Repository:** https://github.com/gjrich/ms-capstone/
- **Script generating the results:** https://github.com/gjrich/ms-capstone/blob/master/problems/predictive_attributes.py
- **Basic results (log file):** https://github.com/gjrich/ms-capstone/blob/master/problems/predictive_attributes.txt
- **Images detailing the analysis results:** https://github.com/gjrich/ms-capstone/tree/master/problems/analysis_results/predictive

These materials complement the discussion in Section 6, providing both the raw output and visualizations of the model's performance.

References

1. Allodi, L., Massacci, F.: A preliminary analysis of vulnerability scores for attacks in wild: The ekits and sym datasets. In: Proceedings of the 2012 ACM Workshop on Building analysis datasets and gathering experience returns for security. pp. 17–24 (2012)
2. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
3. Cybersecurity and Infrastructure Security Agency: Known exploited vulnerabilities catalog. CISA (2023), https://www.cisa.gov/known-exploited-vulnerabilities-catalog
4. FIRST.org: Common vulnerability scoring system v3.1: Specification document. Forum of Incident Response and Security Teams (2019), https://www.first.org/cvss/v3.1/specification-document
5. Forum of Incident Response and Security Teams: Exploit prediction scoring system (epss). FIRST.org (2023), https://www.first.org/epss/
6. Hubbard, D.W., Seiersen, R.: How to measure anything in cybersecurity risk. Wiley (2016)
7. Jacobs, J., Romanosky, S., Adjerid, I., Baker, W.: Exploit prediction scoring system (epss). In: 2019 APWG Symposium on Electronic Crime Research (eCrime). pp. 1–12. IEEE (2019)
8. Mell, P., Scarfone, K., Romanosky, S.: Common vulnerability scoring system. IEEE Security & Privacy **4**(6), 85–89 (2006)
9. MITRE Corporation: Common vulnerabilities and exposures (cve). MITRE (2023), https://cve.mitre.org/
10. National Institute of Standards and Technology: National vulnerability database. NIST Information Technology Laboratory (2023), https://nvd.nist.gov/
11. Offensive Security: Exploit database. Exploit-DB (2023), https://www.exploit-db.com/

12. Pranggono, B., Arabo, A.: Covid-19's impact on cybersecurity. Information Technology-New Generations pp. 349–351 (2021)
13. Spring, J.M., Falcarin, P., Garrido, A., Serna, J.: Time to exploit a vulnerability in the wild: Modeling with imperfect data. Computing **103**, 649–662 (2021)