CL-007

オープンソースソフトウェアにおける脆弱性修正期間の分析
Analysis of Vulnerability Fixing Time in Open Source Software

佐藤 将也†
Masaya Sato

1. はじめに

オープンソースソフトウェア（Open Source Software，以降，OSS）が広く普及している．OSS の利用者は，利用する OSS を選定する際に，機能や利用しやすさだけでなく，脆弱性への対応も考慮して導入するか否かを決定すると考えられる．OSS は，開発過程も公開されていることが多く，脆弱性への対処の過程も利用者から参照できる．脆弱性修正の過程は，利用者が OSS を選定する際の有効な情報になりうる．特に，脆弱性が修正されるまでの期間の傾向を把握することは，利用する OSS の選定において重要である．しかし，脆弱性は多数報告されており，修正期間の傾向を把握するのは難しい．また，特定の OSS に限定したとしても，脆弱性の修正過程を理解するには専門的な知見が求められる．このため，脆弱性の修正状況を簡易に把握できる指標があることが望ましい．そこで我々は，脆弱性修正期間と OSS の情報を関連付け，脆弱性修正期間の分析を行っている．本稿では，脆弱性への対処状況を把握するための予備調査として，OSS における脆弱性修正期間を収集し，傾向を分析した結果を報告する．調査結果より，収集した脆弱性のうち 82.3%が 4 週間以内に修正されていることを示した．また，脆弱性の深刻度と修正期間の関係を分析した結果，深刻度と修正期間に相関は見られなかった．

2. オープンソースソフトウェアと脆弱性修正期間

2.1 オープンソースソフトウェア

OSS は，ソースコードが公開され，再配布などがライセンスにより規定されたソフトウェアである[1]．OSS の例として，Linux や Firefox があり，世界中で広く利用されている．OSS 開発では，GitHub が利用されることが多い．GitHub は OSS 開発を支援するサービスであり，ソースコードの公開だけでなく，開発者や利用者が当該 OSS に関して議論するための Issues という機能を持つ．Issues では，Issue として機能追加や問題提起が行われ（オープンされ），議論が完了すると Issue がクローズされる．また，GitHub は Git を利用した共同開発のための機能を有している．Issues で議論された機能追加や問題提起がソフトウェアに取り込まれる際には，Commit によりソフトウェアへ反映される．

2.2 脆弱性修正期間

2.2.1 脆弱性情報

脆弱性とは，ソフトウェアにおいて設計や実装上の不備により，悪意ある利用者からの攻撃により，ソフトウェア本来の機能や性能を損うセキュリティ上の問題箇所である．ソフトウェアに脆弱性があると，利用者にとって不利益となることがあることがある．このため，脆弱性の早期発見と修正が OSS 利用者にとっては重要となる．

脆弱性は性質や攻撃に利用された際の影響などを考慮して，脆弱性情報として集約されている．脆弱性情報の 1 つとして NIST による National Vulnerability Database（以降，NVD）[2]がある．NVD による脆弱性情報には，脆弱性の識別子と深刻度が含まれている．脆弱性の識別子として，Common Vulnerabilities Exposure （以降，CVE）番号が用いられている．また，脆弱性の深刻度には，Common Vulnerability Scoring System (以降，CVSS)スコアが用いられている．CVSS スコアは，その脆弱性の深刻度を計算したものである．CVSS スコアには v2 と v3 があるが，本研究では CVSS v3 base score を用いる．CVSS v3 base score は，脆弱性の利用しやすさ（Exploitability）と影響（Impact）を基に点数が 0 点から 10 点までの範囲で小数点以下第 1 位まで計算される．なお，CVSS は，脆弱性の危険さではなく深刻度を測るために用いられる．

2.2.2 脆弱性の修正

脆弱性の修正では，まず，開発者，または利用者を含む第三者により脆弱性が発見され，開発者へ通知される．通知には個別のメールや団体などを仲介した通知も含まれる．次に，その脆弱性への修正方針や修正案が開発者間で議論され，具体的な修正内容が提案され，ソフトウェアに取り込まれることで修正が完了し，修正版を利用者に配布することで，利用者は脆弱性が修正されたソフトウェアを利用できる．

OSS では，これらの脆弱性への対処過程が公開されていることがある．たとえば，脆弱性への対処について，GitHub の Issue により提案される場合がある．この場合，Issue で修正案について議論され，具体的な修正がソフトウェアに取り込まれる（Commit される）と対処完了となり，その際に Issue もクローズされる．実際に，NVD に収録されている脆弱性には，Issue への URL が参考情報として登録されている例もある．

2.2.3 脆弱性修正期間

脆弱性修正期間は，2.2.2 項で述べた脆弱性の修正に要した期間とする．ただし，脆弱性修正期間は集計や公開がされていない．このため，公開されている情報から脆弱性修正期間を見積もる必要がある．そこで，脆弱性への対処が開始された時点を修正開始とし，脆弱性へ対処するパッチがソフトウェアに取り込まれた時点を修正完了とし，修正開始から修正完了までの期間を脆弱性修正期間とする．

公開されている情報から修正開始と修正完了を取得するために，本研究では GitHub を用いる．GitHub を用いた OSS 開発では，脆弱性への対処が Issues で議論されることがある．また，NVD による脆弱性情報データベースには，脆弱性に対応する Issue の URL が登録されているものがある．そこで，脆弱性と対応した Issue の開始時点を修正開始とする．また，修正完了は修正がソフトウェアに取り込

†岡山県立大学　Okayama Prefectural University

表 1 取得した脆弱性情報の件数
（2022 年 10 月 18 日時点）

項目	件数
CVE 番号	25,127
GitHub の URL を持つもの	23,824
Issue と Commit を持つもの	1,184
CVSS V3 base score を持つもの	1,143
リポジトリ情報を取得できたもの	**1,116**
リポジトリ数	503

まれた際の Commit 時点とする．Issue における議論は，脆弱性修正の必要がない場合や，他の Issue に移行されることがあり，Issue の終了を修正完了とすることは難しい．一方で，Commit は，当該 OSS に取り込まれた記録なので，修正が完了したことを示す情報として適している．

以上より，脆弱性修正期間は，脆弱性に対応する Issue 開始から Commit 完了までの期間とする．なお，集計を簡単にするために，脆弱性修正期間は時間単位で分析し，分以下の粒度は切り捨てて用いる．

huntr [3]や snyk [4]において，脆弱性の報告日や修正の過程が公開されている．このため，修正開始や修正完了だけでなく，開発者への報告日まで利用した分析が可能である．ただし，GitHub の Issue や Commit を用いる方法よりも分析可能な脆弱性の件数が少なかったため，本調査では利用しないこととした．

3. 研究目的とデータ取得

3.1 目的

本研究では脆弱性情報を収集し，脆弱性修正期間の特徴を調査する．具体的には，以下の 2 項目について調査する．

(1) 脆弱性修正期間の分布
OSS において脆弱性の修正にどの程度の期間を要しているのかの傾向を分析する．これは，OSS における脆弱性修正期間の傾向を概観し，OSS 選択時の参考となる情報を提供するためである．

(2) CVSS スコアごとの脆弱性修正期間の分析
脆弱性修正期間と CVSS スコアの関係を分析する．特に，CVSS スコアの高い（深刻度の高い）脆弱性が早期に修正されているのか否かを分析する．

3.2 脆弱性情報の収集

脆弱性情報は NVD データベースを用いる．NVD では発見された脆弱性について CVE 番号ごとに脆弱性の種類や当該 OSS に対応するリポジトリを JSON 形式で公開している．本研究では 2022 年 10 月 18 日に NVD から取得した脆弱性情報（全期間）を用いた．取得した脆弱性情報の件数を**表 1** に示す．

脆弱性情報をもとに脆弱性修正期間を決定するためには，脆弱性に対応するリポジトリを特定できる必要がある．そこで，NVD から取得した脆弱性情報のうち，リポジトリとして GitHub の URL が登録されている 23,824 件を抽出した．また，脆弱性修正期間を分析するためには，当該 CVE に対応する修正開始（Issue）と修正完了（Commit）を取得可能な 1,184 件を抽出した．このうち，脆弱性の性質を示す指標である CVSS v3 base score を持ち，分析のために必要なリポジトリ情報を取得できた 1,116 件を対象に分析する．なお，分析対象とした 1,116 件に対応するリポジトリ数は 503 件であり，1 リポジトリに複数の CVE 番号が割り当てられている場合もある．

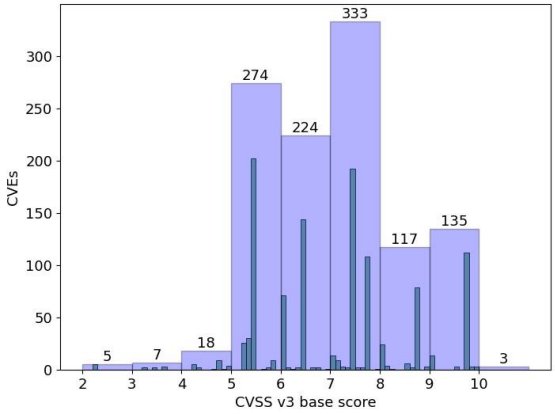

図 1 CVSS スコアごとの CVE 番号数

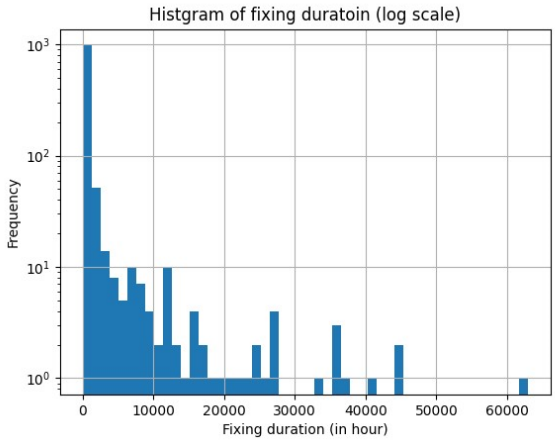

図 2 脆弱性修正期間の分布

CVSS スコアごとの CVE 番号の分布を**図 1** に示す．CVSS スコアは小数点以下 1 桁まで計算されており，グラフでは 0.1 ごとに集計したものと小数点以下を切り捨てて集計したものを重ねて表示した．以降では計算を簡単にするために，小数点以下を切り捨てた CVSS v3 base score を用いた．図 1 より，本調査で用いる脆弱性は CVSS スコア 5 から 8 の範囲のものが多く含まれる．また，CVSS スコアが 4 以下と 10 のものは極端に少ない．

4. 脆弱性修正期間の分析

4.1 脆弱性修正期間の分布

脆弱性修正期間の分布について調査した．修正期間は，全期間，1 日単位，1 時間単位の 3 段階に分けて分析する．調査対象の全脆弱性について，脆弱性修正期間のヒストグラムを**図 2** に示す．図 2 の縦軸は対数スケールで表記している．図 2 より，脆弱性修正期間は短いものが非常に多いことが分かる．ただし，脆弱性修正期間が 10,000 時間（416 日）以上のものも多く存在する．

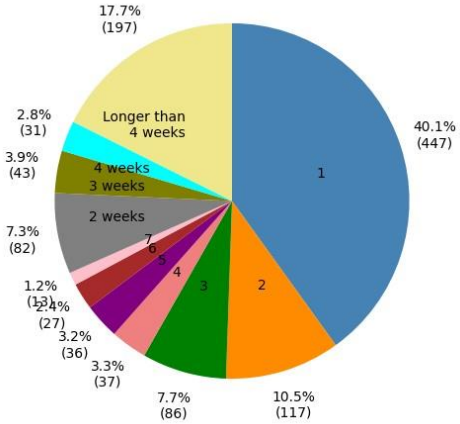

図 3 1 日（24 時間）単位の脆弱性修正期間の割合

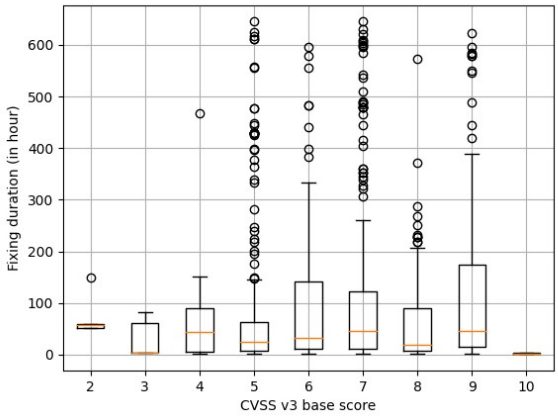

図 5 CVSS スコアごとの脆弱性修正期間
（1 時間以上 672 時間未満，919 件）

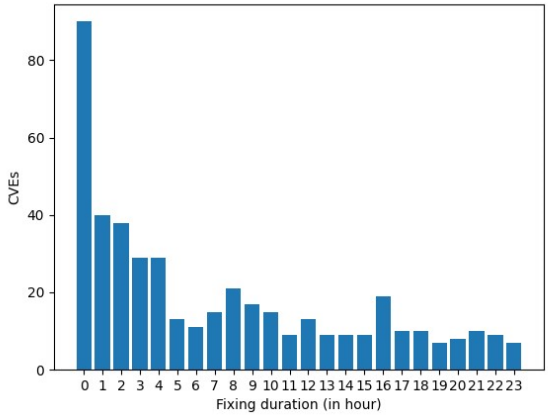

図 4 24 時間以内に修正された脆弱性における脆弱性修正期間の 1 時間単位の分布

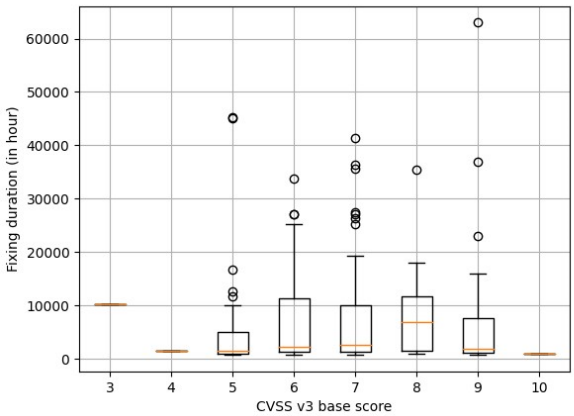

図 6 CVSS スコアごとの脆弱性修正期間
（672 時間以上，197 件）

以上より，脆弱性修正期間のばらつきは非常に大きく，脆弱性修正期間全体の傾向を把握することは難しい．そこで，脆弱性修正期間が比較的短いものに焦点を絞り分析を進める．1 日単位で脆弱性修正期間を集計した結果を**図 3** に示す．修正に 7 日より長い期間を要しているものについては 1 週間単位で分類し，4 週間を超えるものはまとめて集計した．また，割合の下に件数を表示した．図 3 より，調査した脆弱性の 40.1%は 1 日（24 時間）以内に，68.4%が 7 日（168 時間）以内に修正されていることが分かる．また，日を追うごとに修正件数も少しずつ減少していることが分かる．一方，31.6%は修正に 7 日より長い期間を要し，17.7%は 4 週間以内には修正されないことが分かる．

次に，24 時間以内に修正された脆弱性について，1 時間単位で集計した結果を**図 4** に示す．この調査により，90 件の脆弱性が 0 時間（1 時間以内）で修正されていることが分かった．これらの脆弱性について個別に調査した結果，脆弱性修正が非常に容易などの理由で 1 時間以内に修正された場合と，脆弱性修正の記録を残すために Issue が作成される場合が存在した．記録を残すために Issue が作成された場合，Issue の作成直後（数分以内）に Commit が適用されていた．このため，脆弱性修正期間が 0 時間の脆弱性は，脆弱性修正期間の分析には適さないといえる．

4.2 CVSS スコアごとの脆弱性修正期間の分析

CVSS スコアごとに CVE を分類し，脆弱性修正期間を分析する．この際，脆弱性修正期間が 0 の脆弱性は，分析には適さないため除外した．なお，脆弱性修正期間が 0 の CVE について CVSS スコアごとに集計したところ，図 1 と同様の傾向となり，CVSS スコアによる違いは見られなかった．以降の分析では，脆弱性修正期間が 1 時間以上 671 時間（4 週間）以内のもの（919 件）と 672 時間以上のもの（197 件）に分けて分析する．これは，脆弱性修正期間が 672 時間以上の脆弱性は修正期間が極端に長期化しているものが多いため，グラフから傾向を把握するのが困難になるためである．

CVSS スコアごとの脆弱性修正期間の箱ひげ図を**図 5** と**図 6** に示す．まず修正期間 1 時間以上 671 時間以内の脆弱性について分析する．図 5 より，CVSS スコアが 2, 3, 4, および 10 の脆弱性は，100 時間以内に修正される傾向があることが分かる．一方，CVSS スコアが 5 から 9 の脆弱性は，第 3 四分位数までは 200 時間以内に修正されているが，第 3 四分位数よりも大きい点が多く存在することが分

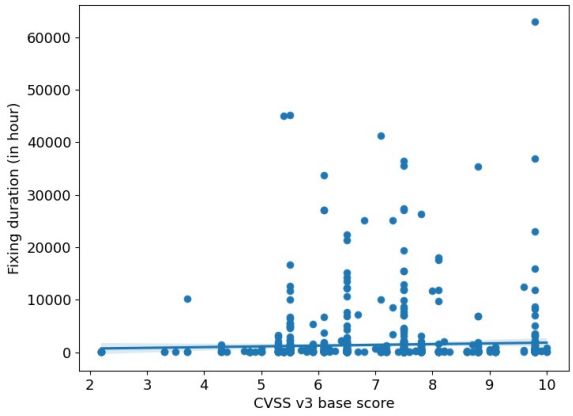

図 7 CVSS スコアと脆弱性修正期間の散布図

かる．これは図 1 に示すように CVSS スコア 5 から 9 の脆弱性が多く存在するため，外れ値も多く含まれたためだと推察できる．次に修正期間が 672 時間以上の脆弱性について分析する．図 6 より，CVSS スコアによる違いは，図 5 と同様の傾向にあるといえる．

図 5 と図 6 の結果より，以下が考察できる．CVSS スコアは，脆弱性修正期間にあまり影響がないと推察できる．CVSS スコアは脆弱性の深刻度を評価しているため，CVSS スコアが高いほど，利用者への影響は大きいと予測できる．このため，開発者にとって早期の対処が要求される可能性が高い．しかし，図 5 と図 6 の結果から，CVSS スコアが高くとも，脆弱性修正期間は短くなっていないことが分かる．図 7 に示すように CVSS スコアと脆弱性修正期間の散布図を見ても相関はなく，相関係数は 0.04 であった．以上より，CVSS スコアと脆弱性修正期間に相関はほとんどないといえる．

5. 関連研究

GitHub リポジトリの分析による脆弱性発見方法について研究されている[5]．この研究では GitHub リポジトリにおける Commit と脆弱性情報を対応付け，脆弱性を持つ可能性がある Commit を低い誤検知率で発見している．この研究ではコードの分析による脆弱性の発見を目的としているが，我々の研究ではコードは用いずリポジトリ情報のみから修正期間を分析している点で分析方法が異なる．コードを用いることでより詳細な分析が可能となるが，プログラミング言語の違いを分析する必要がある．一方，コードを分析に用いない場合は，プログラミング言語の違いを考慮する必要がなく，簡易な分析に適しているといえる．

脆弱性修正期間に着目した研究[6][7]が行われている．文献[6]では，GitHub におけるセキュリティバグの報告から修正までの期間に着目して分析されている．修正期間に着目している点で我々の研究と観点が似ているが，個別の事例に踏み込んだ分析をしているため分析対象の脆弱性が少なく，全体の傾向を分析できない．文献[7]では，脆弱性を持つパッケージがリリースされるまでの過程を分析し，修正が即座にリリースされないことや，リリースされても十分に行き渡るまでに長い期間を要することが報告されている．この研究は修正後の分析を行っているが，我々の研究は修正されるまでの過程に着目している点が異なる．

パッケージごとの依存関係や脆弱性への対処状況を把握するためのサービスが提供されている[8]．これはパッケージ選定の際の参考情報を提供するという観点で我々の研究と類似している．我々の研究は，個別のパッケージではなく，脆弱性情報を取得可能な OSS 全体の傾向を分析している点が異なる．

6. おわりに

オープンソースソフトウェアにおける脆弱性修正期間の分析結果を述べた．脆弱性修正期間には，修正開始から修正完了までの時間数を用いた．修正開始と修正完了を Github における Issue 開始と Commit として調査した．調査対象とした脆弱性における脆弱性修正期間について，40.1%が 24 時間以内に，82.3%が 4 週間以内に修正されていることを示した．

また，脆弱性の深刻度を表した CVSS v3 base score ごとに脆弱性を分類して脆弱性修正期間を分析した．分析の結果，脆弱性の深刻度による脆弱性修正期間への影響はほとんど観察できなかった．ただし，深刻度が低い脆弱性は，深刻度が高い脆弱性よりも早期に修正される傾向があることを示した．また，深刻度が高い脆弱性であっても，早期に修正されるとは限らないことを示した．

残された課題として，脆弱性修正期間が 4 週間を超える脆弱性について，脆弱性修正期間が長大化した要因の調査がある．また，脆弱性修正期間が 0 時間の脆弱性が 90 件観測され，これらは分析に用いることができなかった．このため，より正確に脆弱性修正期間を見積もる方法を検討する必要がある．さらに，脆弱性を種類ごとに分類し分析を進める．

謝辞

本研究を進めるにあたり情報収集に協力くださった岡山県立大学情報工学部の齋藤勇征氏に感謝します．

参考文献

[1] Open Source Initiative: The Open Source Definition (online), available from ⟨https://opensource.org/osd/⟩ (accessed 2023-03-15).
[2] NIST: National Vulnerability Database (online), available from ⟨https://nvd.nist.gov/⟩ (accessed 2023-06-14).
[3] 418sec: huntr (online), available from ⟨https://huntr.dev/⟩ (accessed 2023-06-15).
[4] Snyk: Snyk Vulnerability Database (online), available from ⟨https://security.snyk.io/⟩ (accessed 2023-06-15).
[5] Perl, H., Dechand, S., Smith, M., Arp, D., Yamaguchi, F., Rieck, K., Fahl, S. and Acar, Y.: Vccfinder: Finding potential vulnerabilities in open-source projects to assist code audits, Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 426–437 (2015).
[6] 中野大扉，亀井靖高，Hindle, A.，佐藤亮介，鵜林尚靖：GitHub におけるセキュリティバグ報告と修正の調査，第 81 回全国大会講演論文集，Vol. 2019, No. 1, pp. 287–288 (2019).
[7] Chinthanet, B., Kula, R. G., McIntosh, S., Ishio, T., Ihara, A. and Matsumoto, K.: Lags in the release, adoption, and propagation of npm vulnerability fixes, Empirical Software Engineering, Vol. 26, pp. 1–28 (2021).
[8] Google Inc.: Open Source Insights (online), available from ⟨https://deps.dev/⟩ (accessed 2023-03-19)..