NIST Cybersecurity White Paper
NIST CSWP 41

Likely Exploited Vulnerabilities

A Proposed Metric for Vulnerability Exploitation Probability

Peter Mell*
Computer Security Division
Information Technology Laboratory
**Former NIST employee; all work for this publication was done while at NIST.*

Jonathan Spring
Cybersecurity Division
Cybersecurity and Infrastructure Security Agency

This publication is available free of charge from:
https://doi.org/10.6028/NIST.CSWP.41

May 19, 2025

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

Publication History
Approved by the NIST Editorial Review Board on 2025-04-18

How to Cite this NIST Technical Series Publication:
Mell P, Spring J (2025) Likely Exploited Vulnerabilities: A Proposed Metric for Vulnerability Exploitation Probability. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Cybersecurity White Paper (CSWP) NIST CSWP 41. https://doi.org/10.6028/NIST.CSWP.41

Author ORCID iDs
Peter Mell: 0000-0003-2938-897X
Jonathan Spring: 0000-0001-9356-219X

Contact Information
SecureByDesign@cisa.dhs.gov

Additional Information
Additional information about this publication is available at https://csrc.nist.gov/publications/cswp, including related content, potential updates, and document history.

Abstract

This work presents a proposed security metric to determine the likelihood that a vulnerability has been observed to be exploited. Only a small fraction of the tens of thousands of software and hardware vulnerabilities that are published every year will be exploited. Predicting which ones is important for the efficiency and cost effectiveness of enterprise vulnerability remediation efforts. Currently, such remediation efforts rely on the Exploit Prediction Scoring System (EPSS), which has known inaccurate values, and Known Exploited Vulnerability (KEV) lists, which may not be comprehensive. The proposed likelihood metric may augment EPSS remediation (correcting some inaccuracies) and KEV lists (enabling measurements of comprehensiveness). However, collaboration with industry is necessary to provide necessary performance measurements.

Keywords

Common Vulnerabilities and Exposures; CVE; Known Exploited Vulnerabilities; KEV; Exploit Prediction Scoring System; EPSS; exploit; Likely Exploited Vulnerabilities; LEV; metrology; prediction; scoring; security; vulnerability.

Table of Contents

List of Tables

List of Figures

1. Introduction

This work provides a novel security metric to determine the likelihood that a vulnerability has been observed to be exploited. Only a small fraction of the tens of thousands of software and hardware vulnerabilities that are published every year will be exploited. Predicting which ones is important for the efficiency and cost effectiveness of enterprise vulnerability remediation efforts.

One study shows that only 5 % of vulnerabilities have been observed to be exploited in the wild [2], while the monthly vulnerability remediation rate for companies is 16 % [3]. The remediation rate is so low because it is expensive for companies to address vulnerabilities. Security patches must be tested with enterprise software to ensure continuity of operations; some vulnerabilities require non-patch remediations that require human deployment.

This situation would not be a problem if the 16 % were to cover the 5 %, but metrology is lacking to accurately make that calculation. Thus, predicting which vulnerabilities will be exploited is critically important for the efficiency and cost-effectiveness of enterprise vulnerability remediation efforts.

One effort that has made great progress towards this goal is the Exploit Prediction Scoring System (EPSS). EPSS provides probabilities that a vulnerability will be observed to be exploited in the wild within the next 30 days [4]. However, its probabilities are known to be inaccurate for vulnerabilities that have been previously observed to be exploited; this is stated in the EPSS documentation itself [1]. Fortunately, the probabilities are not randomly inaccurate; they underestimate the true probability.

Another area that is foundational for vulnerability remediation prioritization are Known Exploited Vulnerability (KEV) lists. KEV lists are enumerations of vulnerabilities that have definitively been exploited in the past (e.g., [5] and [6]). However, such lists may not be comprehensive, and prior to this research, metrology did not exist to measure their coverage.

This work provides equations to measure probabilities that vulnerabilities have been observed to be exploited in the past. Lists of these probabilities for vulnerabilities are named Likely Exploited Vulnerabilities (LEV) lists. This name highlights such lists' probabilistic nature and to distinguish them from KEV lists (which they do not replace).

The LEV probabilities have at least four use cases:

1. measure the expected number and proportion of vulnerabilities, as identified by Common Vulnerability and Exposures (CVE) identifiers, that actors have exploited,
2. estimate the comprehensiveness of KEV lists,
3. augment KEV based vulnerability remediation prioritization by identifying higher-probability vulnerabilities that may be missing, and
4. augment EPSS based vulnerability remediation prioritization by identifying vulnerabilities that may be underscored.

LEV accuracy is dependent upon EPSS performance because LEV is calculated from historical EPSS scores. EPSS has published testing indicating that the latest version has a high performance in recall (i.e., coverage) and precision (i.e., efficiency), see Sec. 9. As EPSS performance improves over time (which it is has done significantly over the three versions), LEV performance will improve.

While the derivation of the LEV metric is mathematically sound, it inevitably has a margin of error which is currently unknown. Publicly available vulnerability exploitation data is unavailable but necessary to thoroughly test the performance of the LEV metric.

The EPSS performance results are not of a form that enables a direct calculation of LEV performance. Thus, while LEV performance is presumed to be high given the quality of its foundational EPSS data, performance results for the LEV probabilities themselves are not currently available. The National Institute of Standards and Technology (NIST) is seeking industry partners with relevant datasets to empirically measure the performance of LEV probabilities.

The rest of this work is organized as follows. Section 2 provides background on vulnerability enumerations, EPSS, and KEV lists. Section 3 discusses LEV probability use cases. Section 4 presents the LEV equations. Section 5 derives the equations. Section 6 provides example LEV output. Section 7 presents empirical results. Section 8 discusses KEV list comprehensiveness measurements and possible sources of error. Section 9 covers performance. Section 10 discusses the LEV implementation and Section 11 concludes. Appendix A contain excerpts from the EPSS documentation. Appendix B is a list of abbreviations and acronyms.

2. Background

This section contains background necessary to understand the LEV equations and approach. It introduces vulnerability enumeration lists, EPSS scoring, and discusses KEV lists.

2.1. Common Vulnerabilities and Exposures

The Common Vulnerabilities and Exposures (CVE) is an enumeration of known vulnerabilities in information technology products, primarily software but also hardware [7]. It is widely accepted as the most comprehensive list available and is used extensively within the information technology security community. The United States National Vulnerability Database (NVD) provides information and references for all CVEs [8].

2.2. Exploit Prediction Scoring System

EPSS provides a daily updated score for each CVE. These scores are estimated probabilities that each CVE will be observed to be exploited in the wild within the next 30 days. The EPSS scores are widely accepted by the security industry as shown by the scores being incorporated into 111 security products as of January 2025 [10]. However, security experts debate the pros and cons of using it (e.g., [11], [12]).

The scores are created using a machine learning model trained with several thousand inputs. Training data is provided by a small number of enterprise security monitoring companies, each of which has a large, deployed sensor net to observe exploitation activity for their enterprise customers.

Discussed more thoroughly in Sec. 5.1, EPSS was designed to not include past vulnerability exploitation as an input into its model. This makes the model blind to past exploitation, resulting in inaccurate scores (i.e., probabilities) for vulnerabilities that have been previously exploited. The scores will be too low. A vulnerability that is exploited in a 30-day period will NOT receive a bump up in EPSS score in the subsequent 30-day period. This is not an error as it enables the model to be predictive and have more accurate results for vulnerabilities that have not been previously exploited.

EPSS version 1 was first released 2021-4-14; it was updated to version 2 on 2022-2-04. A much more accurate version 3 was released on 2023-3-07. A research paper explaining EPSS is available at [4].

2.3. Known Exploited Vulnerability Lists

KEV lists identify vulnerabilities that are known to have been exploited in the past. Typically, once a vulnerability is added to a KEV list, it stays on the list permanently. Vulnerabilities not on a list have an unknown status relative to past exploitation.

A prominent KEV list is from the United States (US) Cybersecurity and Infrastructure Security Agency (CISA) [5]. For a vulnerability to be included on this list, it must have a CVE identifier, be relevant to IT systems used by the US government or critical infrastructure, and have an available remediation action (e.g., a patch or workaround). Government agencies are required to remediate these vulnerabilities in a stated time from (usually within two weeks) according to binding operational directive (BOD) 22-01, *Reducing the Significant Risk of Known Exploited Vulnerabilities* [9].

In December of 2024, there were 1228 vulnerabilities on the CISA KEV list. At the same time, there were 260k vulnerabilities on the CVE list. The coverage of the CVEs by the CISA KEV list is then 0.5 %. This is less than the previously cited statistic of 5 % of vulnerabilities having been exploited, but a reduced overall CVE coverage is expected due to the limited scope of the CISA KEV (i.e., its focus on U.S. government systems and critical infrastructure).

Security companies also produce KEV lists (e.g., [6][13]); these may have a wider scope.

3. Purpose and Uses

The LEV equation described in Sec. 4.1 provides a daily-updated probability for each CVE with the likelihood that the CVE has been observed to be exploited in the wild at some point in the past. These results can be used to measure the expected proportion of CVEs that have been exploited and the comprehensiveness of KEV lists. These results can also augment both KEV based and EPSS based vulnerability remediation prioritization.

3.1. Measurement of the Expected Proportion of Exploited CVEs

An important metric is to know the number of CVEs that have been observed to be exploited and the proportion. A lower bound on this can be determined by using the Expected_Exploited() equation below, which uses the LEV equation provided in Sec. 4.1. It provides the expected number of CVEs that have been observed to be exploited on a particular date. The Expected_Exploited_Proportion() equation follows that provides the proportion of CVEs that have been observed to be exploited.

Expected_Exploited() equation is a proxy for measuring the number of CVEs that have been exploited (as opposed to being 'observed' to be exploited). However, that metric cannot be calculated directly because we cannot measure what we cannot see and because that is the formulation used by EPSS, upon which LEV is based.

Table 1. Variables for the Expected_Exploited() Equation

Variable	Description
v	a vulnerability (e.g., a CVE)
d	a date without a time component (e.g., 2024-12-31)
d_0	the first date on which an EPSS score is available for the associated *v*
d_n	the date on which the calculation should be performed (usually the present day)
cves	the set of all CVE vulnerabilities in scope for the KEV list whose d_0 date is less than or equal to d_n

$$\text{Expected\_Exploited}() >= \sum\nolimits_{\forall v \in cves} \text{LEV}(v, d_0, d_n)$$

$$\text{Expected\_Exploited\_Proportion}() >= \frac{\text{Expected\_Exploited}()}{|cves|}$$

3.2. Measurement of the Comprehensiveness of KEV Lists

The LEV probabilities enable the measurement of the comprehensiveness of KEV lists. Previously, KEV maintainers had no metric to demonstrate how close their list was to including all relevant vulnerabilities.

The KEV_Exploited() equation provides a lower bound for the number of vulnerabilities that a KEV list should contain. In order to compute a correct measurement, it is necessary to use the set of CVEs that are within scope for the KEV list.

Table 2. Variables for the Expected_Exploited() Equation

Variable	Description
v	a vulnerability (e.g., a CVE)
D	a date without a time component (e.g., 2024-12-31)
d_0	the first date on which an EPSS score is available for the associated v
d_{kev}	the date representing the most recent update date for the KEV list
scoped_cves	the set of all CVE vulnerabilities in scope for the KEV list whose d_0 date is less than or equal to d_{kev}

$$\text{KEV\_Exploited}(d_{kev}) >= \sum_{\forall v \in scoped\_cves} \text{LEV}(v, d_0, d_{kev})$$

3.3. Augmenting KEV-Based Remediation Prioritization

KEV lists can be augmented by LEV lists. An LEV list is created by choosing a threshold probability and then including all vulnerabilities with an LEV probability greater than the threshold. LEV lists can identify higher-probability vulnerabilities potentially missing from a KEV list. This is important to validate the comprehensiveness of KEV lists (see Sec. 8).

One approach is to investigate vulnerabilities above the chosen LEV list threshold for inclusion in the KEV list. We find through the LEV that many vulnerabilities are low probability (see Sec. 8), thus the workload in performing this KEV validation is relatively small.

Another approach is to specially distinguish vulnerabilities above the chosen threshold as likely candidates. These vulnerabilities are then included in remediation activities along with the KEV list vulnerabilities (possibly with a lower priority). Again, the workload in doing so will be relatively small since the vast majority of vulnerabilities are low probability.

In the case that a KEV list pertinent to an organization's system is not available (e.g., due to cost constraints or licensing issues), an LEV list can be used as a non-comprehensive KEV list. Such a list would be focused solely on the smaller number of more probable vulnerabilities. Given that most vulnerabilities have extremely low probabilities, LEV lists using thresholds even as low as 20 % yield a manageable number of vulnerabilities upon which to focus remediation efforts. It should be emphasized that such a use of LEV, while helpful, will NOT produce a comprehensive KEV list. Such a use of LEV lists DOES NOT replace the need for a KEV list.

3.4. Augmenting EPSS-Based Remediation Prioritization

EPSS is used by organizations to prioritize the remediation of vulnerabilities along with patch testing and application. The goal is to remediate vulnerabilities that are most likely to be exploited within the next 30 days. One approach is to specify a probability threshold and remediate all vulnerabilities above that threshold (starting with the most probable).

However, as discussed in Sec. 2.2 and more thoroughly in Sec. 5.1, EPSS provides inaccurate scores for previously exploited vulnerabilities. It is also not currently possible to fix the inaccurate scores in EPSS. For this reason, EPSS should not be used alone when prioritizing enterprise vulnerability remediation.

A mathematically defensible solution is obtainable if the goal is changed to include remediation of previously exploited vulnerabilities and a comprehensive KEV list is available. To do this, change the EPSS scores to be 1.0 for all vulnerabilities on a KEV list. A comprehensive KEV list will cover all inaccurate EPSS scores, writing over the errors, and producing defensible results (albeit with the expanded goal).

Unfortunately, KEV lists may not be comprehensive and so this approach cannot guarantee to overwrite all incorrect EPSS scores. However, with the expanded goal, each EPSS score can also be reassigned to be the maximum of the LEV and revised-with-KEV EPSS probabilities. This covers then both the chance that the vulnerability was previously exploited (which in some cases is known to be 1.0 for vulnerabilities on a KEV list) and the chance that it will be exploited.

The addition of LEV probabilities is a practical solution that can overwrite remaining inaccurate EPSS scores. It DOES NOT guarantee to remove all EPSS errors (only a comprehensive KEV list does that, which is a property that can be measured using LEV).

The below Composite_Probability() equation formalizes this approach.

Table 3. Variables and Functions for the Composite_Probability() Equation

Variable/Function	**Description**
v	a vulnerability (e.g., a CVE)
d	a date without a time component (e.g., 2024-12-31)
d_0	the first date on which an EPSS score is available for the associated *v*
d_{kev}	the date representing the most recent update date for the KEV list
d_n	the date on which the calculation should be performed (usually the present day)
EPSS(*v*,*d_n*)	EPSS score for v on date *d_n*
KEV(*v*,*d_n*)	if *v* in KEV list on date *d_n*: 1.0 otherwise: 0
LEV(*v*,*d_0*,*d_n*)	Use the LEV equation from Section 4.1

$$\text{Composite\_Probability}(v, d_n) = \max\left(\text{EPSS}(v, d_n), \text{KEV}(v, d_n), \text{LEV}(v, d_0, d_n)\right)$$

4. Equations

The LEV equation measures the probability that a vulnerability has been observed to be exploited in the past. This section presents two variants of the LEV equations: LEV and LEV2.

4.1. LEV Equation

The first equation, LEV, requires less computational resources, both memory and CPU. It uses EPSS scores as predictors for 30-day windows, as they were designed. It is slightly more complicated than LEV2. LEV is the equation used for the experiments and discussions in this work.

Table 4. Variable and Function for the LEV and LEV2 Equations

Variable/Function	**Description**
v	a vulnerability (e.g., a CVE)
d	a date without a time component (e.g., 2024-12-31)
d_0	the first date on which an EPSS score is available for the associated *v*
d_n	the date on which the calculation should be performed (usually the present day)
epss(*v*,*d*)	the EPSS score for vulnerability *v* on date *d*
dates(*d_0*,*d_n*,*w*)	the set of dates to include d_0 and not exceeding d_n, formed by adding multiples of *w* to d_0
datediff(*d_i*,*d_j*)	the number of days between d_i and d_j inclusive
winsize(*d_i*,*d_n*,*w*)	datediff(d_i,d_n) ≥ *w*: *w* datediff(d_i,d_n) < *w*: datediff(d_i,d_n)
weight(*d_i*,*d_n*,*w*)	winsize(d_i,d_n,*w*)/w

$$\mathrm{LEV}(v, d_0, d_n) >= 1 - \prod_{\forall d_i \in \mathrm{dates}(d_0, d_n, 30)} (1 - \mathrm{epss}(v, d_i) \times \mathrm{weight}(d_i, d_n, 30))$$

4.2. LEV2 Equation

The LEV2 equation requires significantly more computational resources. It handles EPSS scores as covering only a single day by dividing them by 30 (instead of each score covering a 30-day window). This enables the equation to incorporate far more EPSS scores within the computation and increases the equation's responsiveness to changing scores (especially for newly released vulnerabilities). Comparing the effectiveness and properties of LEV and LEV2 is left for future work.

While the computational resources are greater for LEV2 than LEV, producing LEV2 measurements on a daily basis within a few hours for all vulnerabilities is well within the reach of a modern but ordinary laptop. The computational challenge for LEV2 is due to both to memory and disk speed requirements as well as processing power. There are nearing 300k vulnerabilities and EPSS publishes a daily file of EPSS scores for all vulnerabilities. Computing LEV probabilities for all vulnerabilities then can require using data from over a thousand of EPSS files. Even a single LEV2 computation may require EPSS scores from all EPSS score files (one per day). Caching all EPSS data in memory to enable faster calculations on all CVEs requires more than 16 GB (certainly achievable with modern computers). However, even with sufficient memory, the LEV2 computations take 30 times longer than with LEV due to using a window size of 1 instead of 30 (compute time is on the order of hours for all CVEs on a commodity business laptop).

The LEV2 equation uses the same variables and functions as the LEV equation (see Table 1).

$$\mathrm{LEV2}(v, d_0, d_n) >= 1 - \prod_{\forall d_i \in \mathrm{dates}(d_0, d_n, 1)} \left(1 - \frac{\mathrm{epss}(v, d_i)}{30}\right)$$

5. Equation Derivation

This section discusses how the LEV equations were derived. It first discusses how EPSS scores are inaccurate for previously exploited vulnerabilities. It then discusses how the EPSS probabilities for previously exploited vulnerabilities are still useful; they are lower bounds. The section then performs the derivation of the LEV equation starting from a textbook conditional probability equation and applying it to the problem of calculating a vulnerability's probability of past exploitation.

5.1. EPSS Scores as Pre-Threat Intelligence

The EPSS Frequently Asked Questions (FAQs) states that EPSS scores "should be thought of as pre-threat intelligence" [1]. Appendix A contains the complete quoted answer from the EPSS FAQ. It says that EPSS scores should be used when threat intelligence, indicating that a vulnerability is being exploited, does not exist. The EPSS FAQ has a question 'Everyone knows this vulnerability has been exploited, why doesn't EPSS score it at 100%?'. EPSS doesn't always score known exploited vulnerabilities at 100 % even when the maintainers know it is currently being exploited.

This is not a mistake or flaw in the EPSS design; the documentation states that this is necessary to create a predictive machine learning model. The EPSS maintainers **do** use the knowledge of when vulnerabilities are observed to be exploited in training the EPSS model. However, they **do not** provide such knowledge as input to the model. That data is just used to train the probability output [1].

The maintainers found that adding vulnerability observation knowledge as input to the model makes the model less predictive. The model then gives too much weight to past observations instead of indicators of future observation. This then trained the model to focus on a different but useful goal (the LEV goal), identifying past exploitation instead of future exploitation.

The documentation says, "if there is an absence of exploitation evidence, then EPSS can be used to estimate the probability it will be exploited" [1]. It also states that if there is evidence of exploitation, then EPSS should not be used: "if there is evidence that a vulnerability is being exploited, then that information should supersede anything EPSS has to say" [1].

5.2. EPSS Scores as Lower Bounds

While EPSS scores assume that a vulnerability has not been observed to be exploited in the past, what happens when that assumption is incorrect? In this case, future exploitation probabilities for the vulnerability (EPSS scores) will increase compared to the vulnerability not having been observed to be exploited. Past observations of exploitation encourage future observations of exploitation. There are five reasons for this:

1. A past observation demonstrates the utility for attackers in exploiting the vulnerability. They wouldn't exploit the vulnerability it if it wasn't useful to them.

2. A past observation demonstrates the ability of attackers to successfully weaponize the vulnerability.
3. A past observation demonstrates the ability of attackers to reach the vulnerability's attack surface in a real-world environment.
4. A past observation demonstrates the ability of security systems to detect attacks using that vulnerability.
5. A past observation provides momentum for its continued exploitation. Why would attackers suddenly stop using a useful exploit tool once it has been constructed and used? Over time, for a particular vulnerability, exploit usage probabilities will eventually decrease as the vulnerability ages and has increasing inapplicability. But, this effect is the same for and also occurs with never exploited vulnerabilities.

Thus, EPSS scores are lower bounds.

5.3. EPSS Scores as Conditional Probabilities

Conditional probability, written P(B|A), is a function P that calculates the probability of event B occurring given that event A has occurred. While not immediately obvious, the LEV equation uses conditional probabilities with the related textbook formula:

$$\mathrm{P}(\mathrm{A} \cap \mathrm{B}) = \mathrm{P}(\mathrm{A}) \cdot \mathrm{P}(\mathrm{B}|\mathrm{A}) \qquad (1)$$

Applying this to the LEV problem:

- Let P calculate the probability that a particular vulnerability was NOT observed to be exploited during a series of consecutive of time windows (representing a single uninterrupted period). All but the last window has a length of 30 days. The last window has a length from 1 to 30 days.
- Let B represent the event that the vulnerability was NOT observed to be exploited within the last window.
- Let A represent the event that the vulnerability was NOT observed to be exploited within all previous time windows. The start time for A is always 00:00 hours on the date on which the vulnerability received its first EPSS score.
- Note that the EPSS score for a window is the score from the first day of the window (since EPSS scores are designed to represent 30-day windows). This derivation uses the function EPSS(W) to represent the EPSS score on the first day of window W.

The conditional probability equation can be rewritten to show the windows explicitly.

$$\mathrm{P}(\mathrm{W}_{1..t}) = \mathrm{P}(\mathrm{W}_{1..t-1}) \cdot \mathrm{P}(\mathrm{W}_t|\mathrm{W}_{1..t-1}) \qquad (2)$$

- t is the number of windows (minimum of 2)
- 1..t represents a single continuous timespan from the start of window W_1 to the end of window W_t.

- $W_{1..t}$ is (A ∩ B) from equation 1.
- $W_{1..t-1}$ is A from equation 1.
- W_t is B from equation 1.

Consider the term $P(W_{1..t-1})$ from equation 2. It can be simplified by applying equation 1 iteratively (in this second iteration of equation 1, P(A ∩ B) is represented by $P(W_{1..t-1})$.

$$P(W_{1..t-1}) = P(W_{1..t-2}) \cdot P(W_{t-1}|W_{1..t-2}) \qquad (3)$$

The term $P(W_{1..t-2})$ from equation 3 can be further simplified in the same manner through using another iteration of equation 1 (in this third iteration of equation 1, P(A ∩ B) is represented by $P(W_{1..t-2})$.

$$P(W_{1..t-2}) = P(W_{1..t-3}) \cdot P(W_{t-2}|W_{1..t-3}) \qquad (4)$$

This term expansion approach can be applied iteratively to keep expanding the P(A) term from equation 1 until there is only one window left. At this point, the probability of a single 30-day window is one minus the EPSS score (since the event is that the vulnerability is NOT observed to be exploited), as shown in equation 5. This is also the result for $P(W_{1..t})$ for cases of t=1 (important to note since this derivation assumes that t>1).

$$P(W_1) = 1 - \mathrm{EPSS}(W_1) \qquad (5)$$

Equation 2 then simplifies to

$$P(W_{1..t}) = \left(1 - \mathrm{EPSS}(W_1)\right) * \prod_{i=2..t} P(W_i|W_{1..i-1}) \qquad (6)$$

The term $P(W_i|W_{1..i-1})$ from equation 6 can be simplified to $1 - \mathrm{EPSS}(W_i)$ if the vulnerability was not observed to be exploited in the previous windows. The discussion of EPSS as pre-threat intelligence from Sec. 5.1 makes this a reasonable assumption. However, Sec. 5.2 discusses how EPSS scores are lower bounds (even though it might be reasonable to consider them equalities). This changes the comparison operator for the equation from = to <= when simplifying the term $P(W_i|W_{1..i-1})$ to be $1 - \mathrm{EPSS}(W_i)$. Equation 7 shows this substitution.

$$P(W_{1..t}) <= \prod_{i=1..t} \left(1 - \mathrm{EPSS}(W_i)\right) \qquad (7)$$

Note how the first term in equation 6 was moved within the product operator in equation 7.

However, there is an unaddressed issue. What if the last window is less than 30 days? To account for this, the EPSS score is proportioned by the amount of time it covers (from 1 to 30 days). This modifies equation 7 as follows.

$$P(W_{1..t}) \leq \prod_{i=1..t} \left(1 - \left(\mathrm{EPSS}(W_1) * \frac{window\ size\ i}{30}\right)\right) \qquad (8)$$

Finally, LEV calculates the complement of P. LEV is for when a vulnerability has been observed to be exploited. P is a function that calculates the probability that a vulnerability has not been observed to be exploited.

$$\mathrm{LEV} >= 1 - \mathrm{P}(\mathrm{W}_{1..t}) \qquad (9)$$

The inequality flipped from <= to >= because of the subtracting $1-\mathrm{P}(\mathrm{W}_{1..t})$. Expanding this out yields the following.

$$\mathrm{LEV} \geq 1 - \prod_{i=1..t}\left(1 - \left(\mathrm{EPSS}(\mathrm{W}_1) * \frac{window\ size\ i}{30}\right)\right) \qquad (10)$$

This then is the derivation for the LEV equation.

The LEV equation in Sec. 4.1 appears slightly different because it is somewhat more formal. It takes as input the vulnerability being calculated and the desired date for the LEV calculation. It uses a weight() function to represent *window size i*/30. It also uses the dates() function to explicitly calculate on the EPSS score for the first day of each window. In equation 10, the EPSS() function that calculates an EPSS score for a window is then replaced with the lowercase epss() function from Sec. 4.1 that calculates for a particular vulnerability on a particular date.

6. Example Output

The LEV implementation outputs daily information on each CVE. This includes the overall past exploitation probability, but also includes additional supportive data to enable a person to understand a vulnerability's history with respect to exploitation probability.

For each CVE, the following data fields are provided:

1. CVE name
2. Publish date
3. Description
4. LEV probability (the probability of past observation of exploitation)
5. The peak (i.e., maximum) EPSS score among the evaluated 30-day windows
6. The date of the peak EPSS score
7. The EPSS scores for each of the 30-day windows
8. The dates for each window
9. The affected products using Common Platform Enumeration (CPE) values [15]

Below is example output for vulnerability CVE-2023-1730, produced on 2024-12-12. Each of the above listed data fields is provided. The EPSS scores started and remained at 0.00 for months after the initial vulnerability publication on 2023-05-02. They peaked at 0.16 on 2024-01-27 and then fell, hovering at about 0.05. The last EPSS score is shown as 0.06. However, the actual EPSS for that date (2024-11-22) is 0.08. It appears as 0.06 because, per the note on the bottom, the score needed to be adjusted because the last window was just 21 days (while EPSS is for a prediction on a 30-day window). While all the individual EPSS probabilities were less than or equal to 0.16, the overall past exploitation probability as calculated by the LEV equation was 0.70.

```
CVE-2023-1730, published 2023-05-02

The SupportCandy WordPress plugin before 3.1.5 does not validate and
escape user input before using ...

LEV probability: 0.70

Peak EPSS: 0.16, Peak date: 2024-01-27

EPSS scores:

0.00 0.00 0.00 0.05 0.06 0.06 0.06 0.10 0.13 0.16 0.08 0.05 0.05 0.04
0.05 0.05 0.05 0.05 0.05 0.06

EPSS dates:

2023-05-02 2023-06-01 2023-07-01 2023-07-31 2023-08-30 2023-09-29
2023-10-29 2023-11-28 2023-12-28 2024-01-27 2024-02-26 2024-03-27
2024-04-26 2024-05-26 2024-06-25 2024-07-25 2024-08-24 2024-09-23
2024-10-23 2024-11-22

Note: final EPSS score adjusted for window size of 21
```

```
('a', 'supportcandy', 'supportcandy')
```

Another example is CVE-2023-29373, output produced on 2025-1-22. This vulnerability is different in that its EPSS probability history eventually goes down to 0.00. It is 0.00 even without the adjustment due to the final window size being 19. This reduction to 0.00 is necessary and reasonable for older vulnerabilities to avoid the LEV probability eventually rising to 1.0. The vast majority of CVEs have low LEV probabilities, indicating that EPSS's eventual lowering of the probability down to 0.00 is the norm.

```
CVE-2023-29373, published 2023-06-14

Microsoft ODBC Driver Remote Code Execution Vulnerability...

LEV probability: 0.54350

Peak EPSS: 0.08, Peak date: 2024-03-10

EPSS window scores:

0.00 0.05 0.02 0.02 0.03 0.04 0.05 0.06 0.08 0.08 0.07 0.04 0.04 0.03
0.03 0.03 0.03 0.03 0.03 0.00

EPSS dates:

2023-06-14 2023-07-14 2023-08-13 2023-09-12 2023-10-12 2023-11-11
2023-12-11 2024-01-10 2024-02-09 2024-03-10 2024-04-09 2024-05-09
2024-06-08 2024-07-08 2024-08-07 2024-09-06 2024-10-06 2024-11-05
2024-12-05 2025-01-04

Note: final EPSS score adjusted for window size of 19

('o', 'microsoft', 'windows_server_2022')\n", "('o', 'microsoft',
'windows_11_22h2')\n", "('o', 'microsoft', 'windows_server_2016')\n",
"('o', 'microsoft', 'windows_10_1607')\n", "('o', 'microsoft',
'windows_10_21h2')\n", "('o', 'microsoft', 'windows_10_22h2')\n",
"('o', 'microsoft', 'windows_server_2019')\n", "('o', 'microsoft',
'windows_10_1809')\n", "('o', 'microsoft', 'windows_10_1507')\n",
"('o', 'microsoft', 'windows_server_2008')\n", "('o', 'microsoft',
'windows_11_21h2')\n", "('o', 'microsoft', 'windows_server_2012')
```

The EPSS scores vary over the lifetime of these two vulnerabilities. They both start with 0, rise some, hit a peak value, and then decline over time. This very typical for CVEs. There is often some minor oscillation in probabilities, but not usually large spikes up and down.

7. Empirical Results

This section contains empirical results from using the LEV equation. The distribution of LEV probabilities is presented. Empirical data on the proportion of vulnerabilities expected to be exploited is discussed. Example measurements of the comprehensiveness of KEV lists are provided. And measurements of recall are presented showing that LEV lists do not replace KEV lists.

7.1. LEV Distributions

Fig. 1 provides the cumulative distribution of LEV probabilities per year.

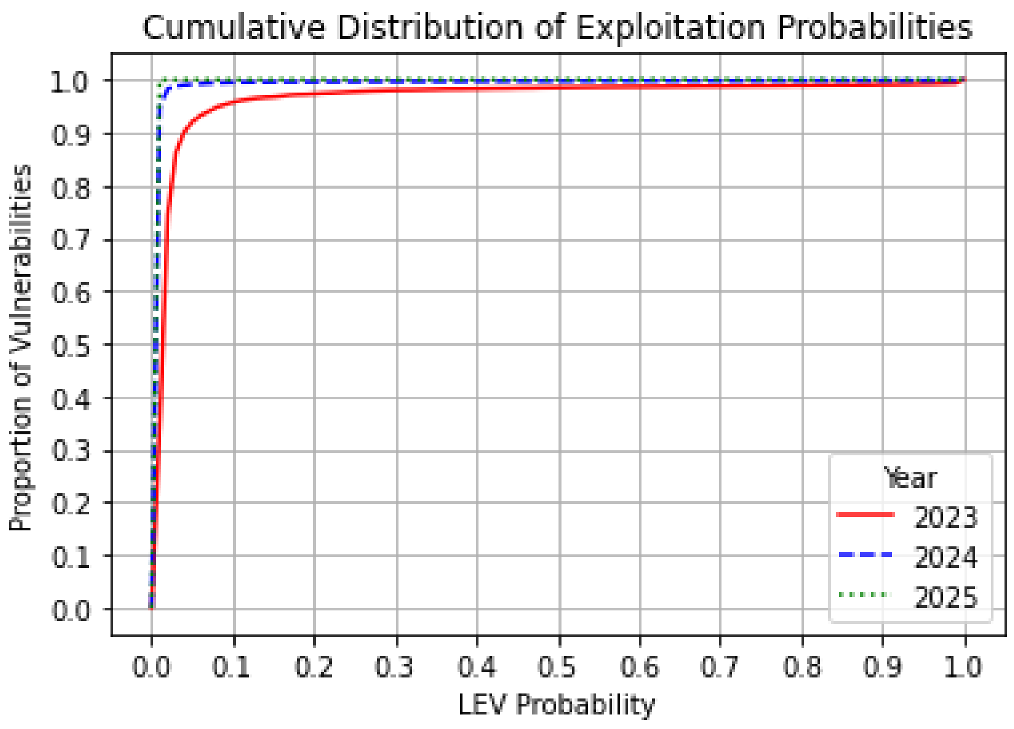

Fig. 1. Cumulative distribution of LEV probabilities (3/6/2025)

The curves for older years are skewed to the right because they have more EPSS scores comprising their LEV probability, thus promoting an increase in the LEV probabilities. Note that in the LEV equation, the addition of EPSS scores can only increase the overall probability of a vulnerability being exploited in the past. This makes sense because every new day provides a new possibility for a vulnerability to be exploited. Older vulnerabilities, on average, are more likely to be exploited.

Fig. 2 and Fig. 3 show the overall LEV probability distribution using a binning approach.

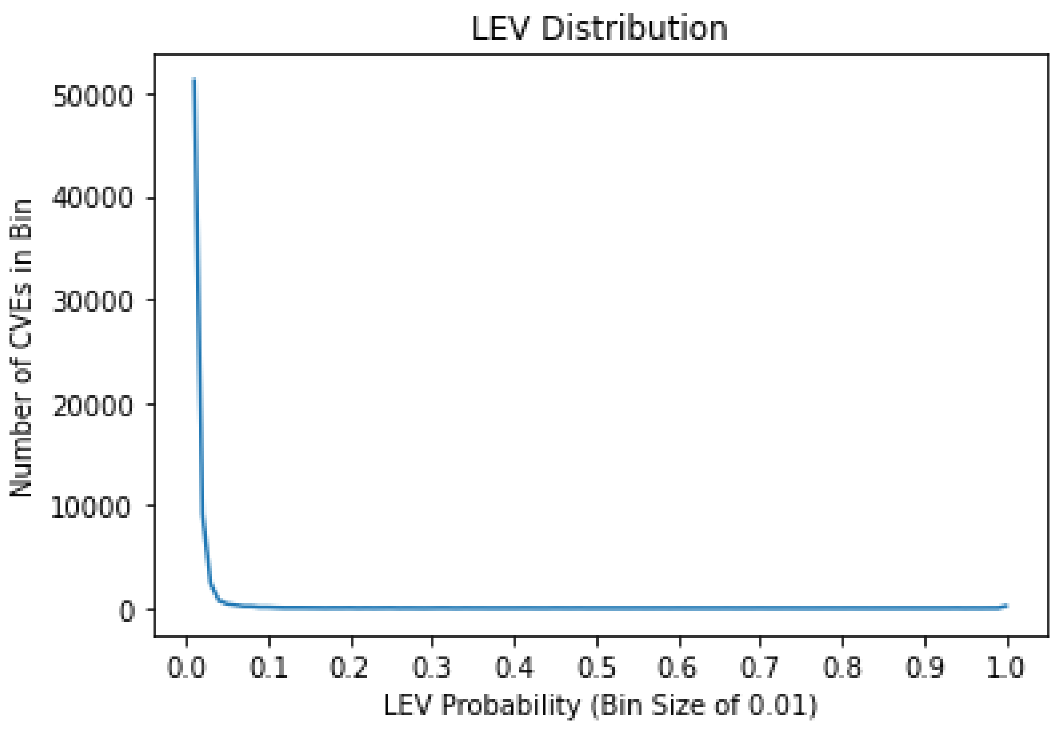

Fig. 2. LEV distribution using binning

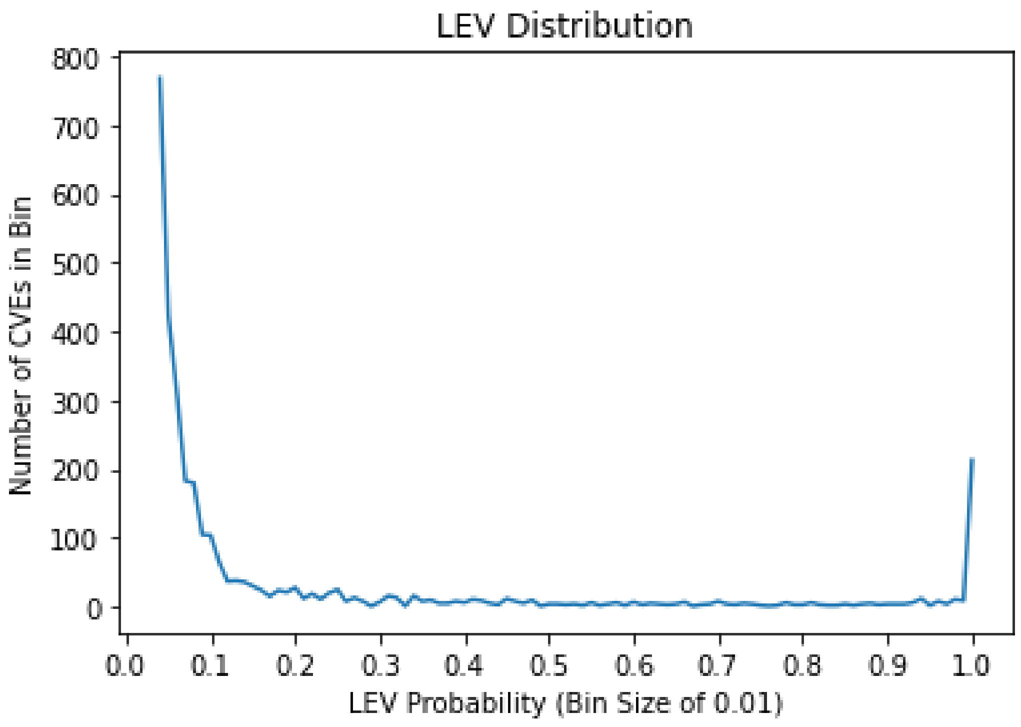

Fig. 3. LEV Distribution using binning (excluding bins of less than 0.03)

Fig. 2 provides the full distribution while Fig. 3 focuses excludes the most populated probability bins of 0.01 and 0.02 to enable better visual inspection of the rest of the distribution. It is easier to see in Fig. 3 the several hundred vulnerabilities with a probability of almost 1.0. Interestingly, many of these vulnerabilities are not included in tested KEV lists.

The majority of vulnerabilities have very low LEV probabilities. There are so many of them that, based on the mathematical definitions of probability, a significant number of low probability vulnerabilities will be observed to be exploited. This is one reason that LEV lists cannot replace KEV lists. LEV cannot identify which of the many low probability vulnerabilities will be exploited, it can only help compute how many of them are expected to be exploited. KEV lists identify the exact ones that have been exploited.

7.2. Proportion of CVEs Expected to be Exploited

The Expected_Exploited() equation from Sec. 3.1 can be used to compute the number of CVEs that are expected to be observed to be exploited. This number is compared against the total number of CVEs in the Expected_Exploited_Proportion() equation to determine the proportion that should be observed to be exploited.

On 2025-01-16, there were 66 125 CVE vulnerabilities with EPSS version 3 scores. Using the Expected_Exploited() equation, it was computed that 1006 CVEs were expected to have been observed to be exploited. This is a proportion of 1.5 %, much lower than the 5 % figure cited in [2].

An explanation for this is that this work’s dataset uses newer vulnerabilities since it only includes those CVEs for which EPSS version 3 scores are available for their entire history (those CVEs published since 2023-03-07). Newer vulnerabilities have less time to be exploited and thus a lower LEV probability.

Rerunning the experiment with both EPSS version 2 and 3 covered CVEs (those published since 2022-02-04) yields a proportion of 4.7 % (in line with the research from [2]). Including EPSS version 1 covered CVEs (those published since 2021-04-14) yields a proportion of 8.7 %. Rerunning the experiment with all CVEs (some going back to 1980) yields a proportion of 14.6 %. While this demonstrates a growing proportion of vulnerability exploitation over time, the older data contains a higher error rate. EPSS version 2 is less accurate than version 3; version 1 is less accurate than version 2. The vulnerabilities published prior to 2021-04-14 (the EPSS version 1 start date) are missing many EPSS scores.

Table 5. Proportion of expected exploited on 2025-01-16

Start Date	Number of CVEs	Expected Exploited	Proportion
2023-03-07	66 125	1006	0.015
2022-02-04	94 440	4410	0.047
2021-04-14	112 754	9791	0.087
All dates	273 979	39 925	0.146

These older vulnerabilities identified by LEV as likely being exploited are still important. According to [14], 66 % percent of exploits attack vulnerabilities that are at least five years old; 37 % percent target those that are at least ten years old.

It was hypothesized that the proportion would approach 1.0 as older vulnerabilities were included. This is because each added day presents a new opportunity for exploitation; each new EPSS score can only push an LEV probability upwards. This does not happen, however, because EPSS apparently correctly accounts for a lack of applicability of many older CVEs by calculating their probabilities to be almost zero.

7.3. LEV Recall of KEV Lists

The statistical measurement of ‘recall’, applied to LEV, measures the coverage of KEV lists by LEV lists of differing threshold values. As discussed previously in Sec. 3.3, an LEV list can be created by choosing a threshold probability and then including all vulnerabilities with an LEV probability greater than the threshold.

Fig. 4 provides the recall measurements for an LEV list relative to its coverage of a KEV list.

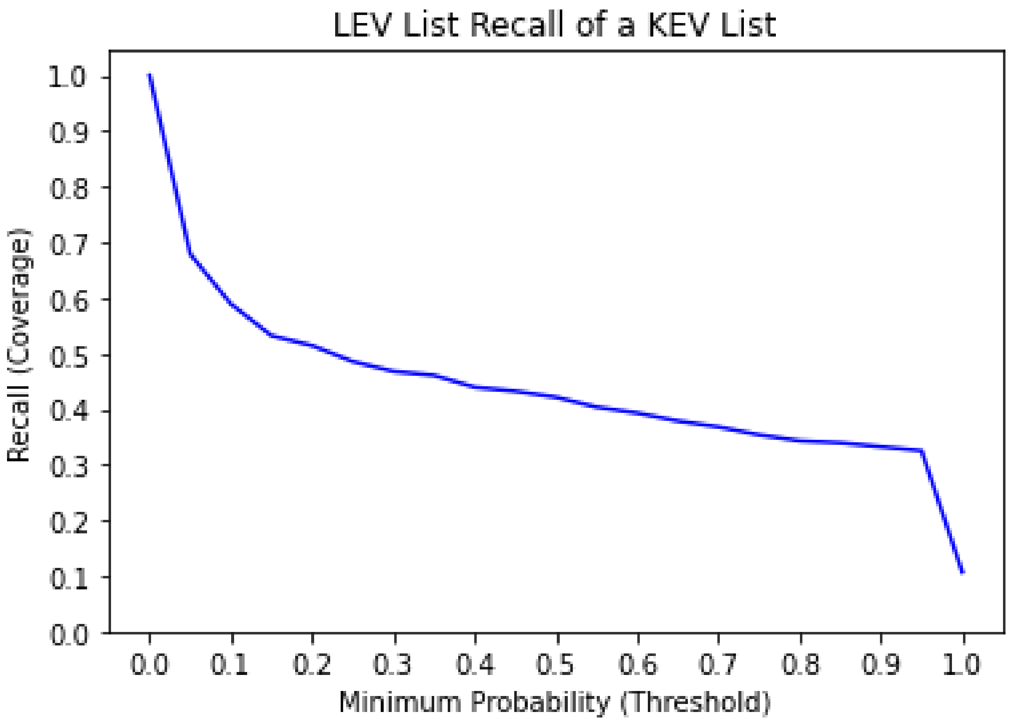

Fig. 4. Recall measurements for LEV and CISA KEV

Even at a threshold of 0.1, the LEV list does not provide more than 60 % coverage. This indicates that LEV lists should not be used as a replacement for KEV lists. It does not necessarily show a deficiency in LEV. It demonstrates that there are a large number of low probability vulnerabilities, and given the large number, many of them will actually be exploited despite their low probability.

8. KEV List Comprehensiveness Measurements and Potential Sources of Error

As discussed in Sec. 3.2, LEV results can be used to measure the comprehensiveness of a KEV list; a lower bound for the expected number of known exploited vulnerabilities can be measured.

LEV cannot always be used to identify which vulnerabilities are missing from a KEV list. For example, Sec. 7.1 shows that the vast majority of vulnerabilities have a very low LEV probability; there are so many of them that a large number will be vulnerable, but the LEV probabilities cannot distinguish between them.

LEV can be used to identify missing vulnerabilities when the LEV vulnerabilities are high. When the LEV equation (10) for a specific product vulnerability (that is, CVE ID) is high, we expect the CVE ID to be present on the CISA KEV list (see Sec. 3.2). When a vulnerability has a high LEV but is not on the KEV, there is some sort of mismatch that should be understood. We generally use 0.9999 as a “high” LEV, but the threshold is simply a measure of confidence for the LEV results. Using a threshold as low as 0.1 can yield a practical number of candidate vulnerabilities to review for KEV inclusion since most CVEs have an LEV less than this.

When a vulnerability has a high LEV probability (using whatever threshold is convenient for ‘high’) and it is not on the KEV list under analysis, one can consider what kind of error kept the vulnerability off the KEV list.

The source of error changes how we identify it and how we respond to it. We identify each kind of error through a different process; therefore, identifying errors is an iterative process of ruling out each kind. We also respond to each kind of error with a different recommended response for vulnerabilities in that kind. Therefore, determining the kind of error is important for understanding whether the error should lead to a change in the KEV, the change in our analysis, change in the source data, or is merely the result of the nature of probabilities.

The sources of error we discuss in this section are as follows:

- KEV scope choice – these are exploited but are not in the scope of the KEV; this is not truly an error but a mismatch in scope between the source data and the exploited vulnerability list.
- Probability error – the LEV equation (10) makes some mistake or invalid assumption. For example, since (10) takes in multiple scores per probability calculation, it could amplify small dependent errors if the equation incorrectly assumes independence.
- Analysis error – the LEV value is high, but in fact should not be. The error occurs somewhere before the LEV calculation (that is, in data collection or analysis).
- KEV visibility error – these should be on the KEV, but they are not because the list owner does not know about the exploitation
- Calibration error – the EPSS score rank orders vulnerabilities correctly but the probabilities are all shifted too high, making the LEV an overestimate. That is, the LEV value for the vulnerability should not actually be above the analysis threshold for being ‘high’.

- Chance – while the vulnerability correctly has a high likelihood of exploitation, it is not in fact exploited.

In general, the errors should be ruled out in the order presented here.

8.1. KEV policy choice

Summary: The vulnerability is exploited but is not in the scope of the KEV.

How to identify: KEV lists may have an explicit scope, an implicit scope, or both. For example, the CISA KEV states it is "for purposes of safeguarding federal information and information systems."[1] While this scope is explicit, it is not granular at the level of products. We identify the implicit scope of the CISA KEV based on the product information for vulnerabilities listed on the KEV to February 2025.

How to respond: Adjust the analysis defined in Sec. 3.2 to exclude this vulnerability. The result is not truly an error but a mismatch in scope between the source data and the exploited vulnerability list. Alternatively, the KEV list maintainer may decide to adjust the scope of the list.

8.2. Probability error

Summary: The LEV equation (10) makes some mistake or invalid assumption. For example, since (10) takes in multiple scores per probability calculation, it could amplify small dependent errors if the equation incorrectly assumes independence.

How to identify: Peer review of the equation, its formulation, and justification for each mathematical step. Ideally, consult several peer experts with distinct areas of expertise to cover different types of formalism error.

How to respond: Adjust the equation to account for any concerns identified in the formal specification. Re-run the analysis and see if the vulnerability still has a high LEV value.

8.3. Analysis error

Summary: The LEV value is high, but in fact should not be. The error occurs somewhere before the LEV calculation (that is, in data collection or analysis).

How to identify: The EPSS Special interest group (SIG) provides transparency on types of errors that may occur in data collection and analysis (see https://www.first.org/epss/faq and https://www.first.org/epss/model). For example, the source EPSS data is based on exploitation activity, defined as "is evidence that exploitation of a vulnerability was attempted, not that it was successful against a vulnerable target."[2] If the vulnerability has a high LEV because many scanners are attempting to exploit it, but all in-scope systems have been hardened or patched,

[1] BOD 22-01: Reducing the Significant Risk of Known Exploited Vulnerabilities. CISA. 2022. https://www.cisa.gov/news-events/directives/bod-22-01-reducing-significant-risk-known-exploited-vulnerabilities
[2] https://www.first.org/epss/model. Accessed Feb 19, 2025.

there will be exploitation activity without actual active exploitation. Determining if a vulnerability falls into this space between, for example, exploitation activity and actual active exploitation, requires expert review of the known information about the vulnerability.

How to respond: The vulnerability should not be added to the KEV, though it may deserve regular review to make sure that does not change.

8.4. KEV visibility error

Summary: These should be on the KEV, but they are not because the list owner does not know about the exploitation.

How to identify: Search for evidence of exploitation. Searching may include directing incident responders to identify how exploitation of the vulnerability would be visible in logs of different kinds (system logs, network intrusion detection systems, host-based intrusion detection systems, etc.), as well as checking with vendors, security companies, and the community as to whether they have similar evidence. If evidence of exploitation is found, then the error is a KEV visibility error.

How to respond: Add the CVE to the KEV list.

8.5. Calibration error

Summary: The EPSS score rank orders vulnerabilities correctly but the probabilities are all shifted too high, making the LEV an overestimate. That is, the LEV value for the vulnerability should not actually be above the analysis threshold for being ‘high’.

How to identify: The previous four sources of error have been ruled out, but the number of vulnerabilities with high LEVs is much larger than would be expected based on chance. For example, if the LEV threshold is 0.9999, but much more than 0.0001 of the vulnerabilities analyzed have a high LEV but no other known error.

How to respond: Adjust the EPSS scores at some point in the LEV analysis to shift them all down by a proportion related to the number of excess vulnerabilities identified.

8.6. Chance

Summary: While the vulnerability correctly has a high likelihood of exploitation, it is not in fact exploited.

How to identify: All other sources of error have been ruled out.

How to respond: Do not add vulnerability to the KEV list. No adjustment to the LEV calculation or analysis pipeline is made. Everything is working as expected, this result is just what it means to be working with probabilities.

9. Performance

LEV performance is based upon EPSS performance since LEV is computed using EPSS scores. Below is provided data showing that EPSS version 3 performance is high with respect to recall and precision. While this indicates that LEV performance should also be high, it is not possible to use this data to make claims about LEV performance.

NIST is seeking industry partners to empirically test the LEV probabilities. For this, datasets are needed that consist of a set of CVEs and data on when, if at all, each CVE was first observed to be exploited. Since in practice only 5 % of CVEs are exploited [2], it is important that the dataset contains many CVEs that have never been observed to have been exploited (ideally in that proportion).

Shown in Figure 1 from [4], EPSS performance is much greater for version 3 than for versions 1 and 2. For this reason, the empirical work of this paper is limited to using EPSS version 3 scores (CVEs published after 2023-3-07).

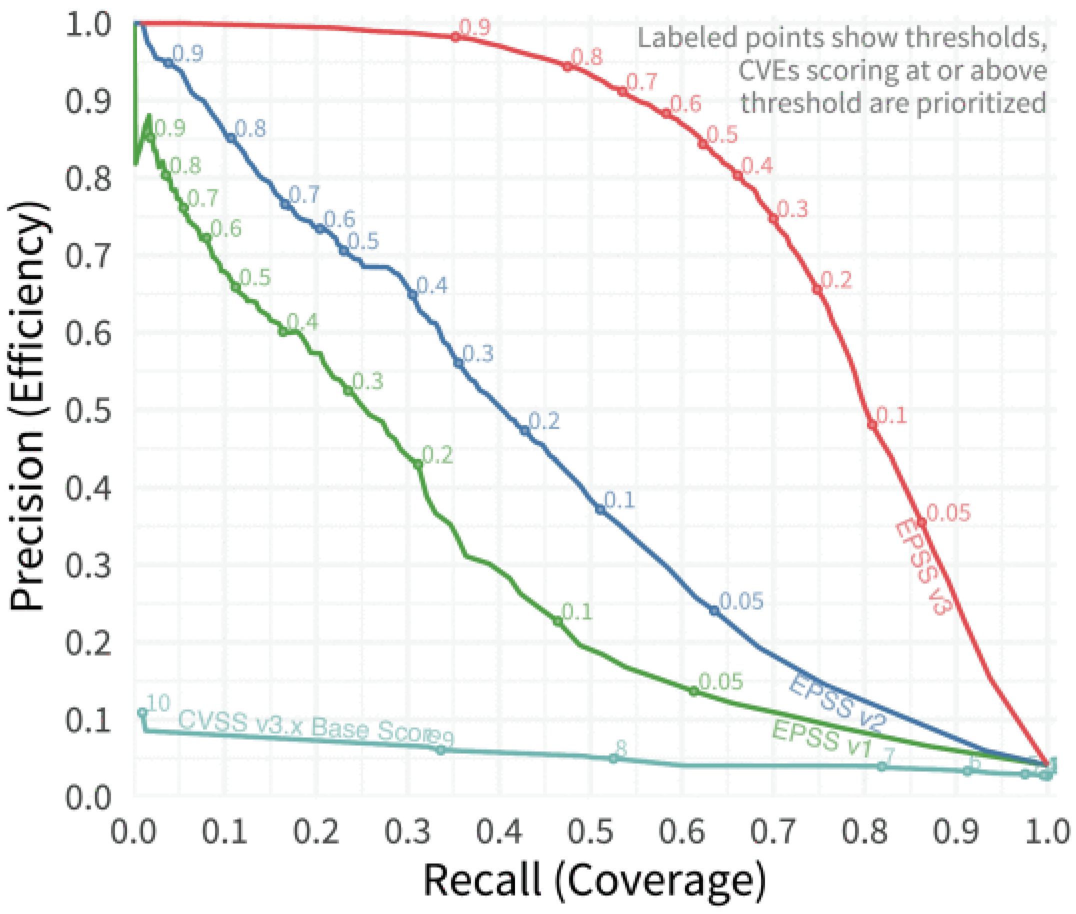

Fig. 5. EPSS Performance, graph copied from [4]

In Fig. 5, the top red line represents EPSS v3. The labelled dots represent sets of CVEs with EPSS scores of at least that dot's value. The x-axis, recall, shows the coverage of this set compared to the CVEs that are observed to be exploited during a 30-day period. The y-axis, precision, represents the proportion of the identified CVE set are observed to be exploited during a 30-day period.

10. Implementation and Availability

The LEV equation has been implemented using Python; it downloads data from several resources prior to computing LEV probabilities: NVD, CISA KEV, and EPSS.

10.1. NVD Download

The code downloads the NVD database to retrieve the complete list of CVEs, their publish dates, and descriptions. NVD can only be downloaded by requesting an NVD application programming interface (API) key from NIST (https://nvd.nist.gov/developers/request-an-api-key). The API key username and password must be entered into the source code to enable NVD downloads. It can take a couple of hours to download the database for the first time; after that the code downloads just the changes and new CVE additions.

10.2. CISA KEV Download

The CISA KEV list is downloaded as a small CSV (comma-separated-values) file.

10.3. EPSS Download and Data

EPSS publishes the EPSS scores for all CVEs as zipped CSV files. There is one file per day, and unfortunately there are a few missing days. The LEV code uses the EPSS scores from the next available day when a day is missing. The LEV code must download all historical EPSS daily files to use the probability history of a CVE to compute its LEV probability. This takes less than an hour the first time, but then the daily EPSS files are cached locally since they don't change.

The LEV code can use all EPSS scores from EPSS's initiation in 2021. However, the LEV empirical studies in this work use only EPSS version 3 to take advantage of the increased performance of this version. Thus, full high quality EPSS data is only available for CVEs published on and after 2023-3-07.

It is even possible to calculate LEV probabilities for CVEs published prior to EPSS version 1. For example, LEV probabilities can be calculated for CVEs published in the 1990s. The code will use the EPSS probabilities available for each CVE. Missing EPSS probabilities decrease the LEV probability, due to the nature of the LEV equation. Since LEV is a greater than or equal to inequality, a lowered LEV probability (due to missing EPSS data) is still correct. It will just have a smaller lower bound than if all of the EPSS probabilities were available.

11. Conclusion

This work describes a proposed security metric to determine the likelihood that a vulnerability has been observed to be exploited.

The LEV equations provide the probability that a vulnerability has been observed to be exploited in the wild. These probabilities can be used to measure the expected proportion of CVEs that have been exploited. They can measure the comprehensiveness of KEV lists. They also can augment remediation prioritization using EPSS and/or KEV lists by identifying higher-probability vulnerabilities that may be underscored in EPSS or missing from a KEV list.

This is important because it has been shown empirically that KEV lists are not comprehensive relative to the total set of vulnerabilities. Also, EPSS is, by design, inaccurate for vulnerabilities that have been previously observed to be exploited.

All four of these use cases are critically important because organizations rely upon both KEV lists and EPSS scores to focus limited resources on prioritizing vulnerability remediation and patch management.

However, it must be emphasized that, while the derivation of the LEV metric is mathematically sound, it inevitably has a margin of error which is currently unknown. Publicly available vulnerability exploitation data is unavailable but necessary to thoroughly test the performance of the LEV metric.

References

[1] FIRST (2024) EPSS Frequently Asked Questions: Everyone knows this vulnerability has been exploited, why doesn't EPSS score it at 100%? Available at https://www.first.org/epss/faq

[2] Jacobs J, Romanosky S, Adjerid I, Baker W (2020) Improving vulnerability remediation through better exploit prediction. *Journal of Cybersecurity* 6(1) (2020):1-12. https://doi.org/10.1093/cybsec/tyaa015

[3] Cyentia Institute, Kenna Security (2022) Prioritization to Prediction Volume 8: Measuring and Minimizing Exploitability. Available at https://library.cyentia.com/report/report_008756.html

[4] Jacobs J, Romanosky S, Suciu O, Edwards B, Sarabi A (2023) Enhancing Vulnerability prioritization: Data-driven exploit predictions with community-driven insights. *2023 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (IEEE Computer Society, Delft, The Netherlands), pp 194-206. https://doi.ieeecomputersociety.org/10.1109/EuroSPW59978.2023.00027

[5] CISA (2024) Known Exploited Vulnerabilities Catalog. Available at https://www.cisa.gov/known-exploited-vulnerabilities-catalog

[6] VulnCheck (2024) Exploit & Vulnerability Intelligence. Available at https://vulncheck.com/product/exploit-intelligence

[7] MITRE (2024) CVE. Available at https://cve.mitre.org

[8] NIST (2024) National Vulnerability Database. Available at https://nvd.nist.gov

[9] CISA (2021) BOD 22-01: Reducing the Significant Risk of Known Exploited Vulnerabilities. Available at https://www.cisa.gov/news-events/directives/bod-22-01-reducing-significant-risk-known-exploited-vulnerabilities

[10] FIRST (2024) Who is using EPSS? Available at https://www.first.org/epss/who_is_using

[11] Spring J (2022) Probably Don't Rely on EPSS Yet. Available at https://insights.sei.cmu.edu/blog/probably-dont-rely-on-epss-yet

[12] Deploy Securely (2022) Why you probably should use the EPSS. Available at https://blog.stackaware.com/p/why-you-probably-should-use-epss

[13] Tenable (2025) Documentation: Vulnerability Information. Available at https://docs.tenable.com/vulnerability-management/Content/vulnerability-intelligence/vulnerability-information.htm

[14] Romanosky S (2022) Exploit Prediction Scoring System: Changing our approach to vulnerability prioritization. Available at https://www.first.org/resources/papers/conf2023/FIRSTCON23-TLP-CLEAR-Romanosky-and-Jacobs-An-Introduction-to-EPSS.pdf

[15] NIST (2025) Official Common Platform Enumeration (CPE) Dictionary. Available at https://nvd.nist.gov/products

Appendix A. EPSS Documentation

The following text is an excerpt from the EPSS Frequently Asked Questions (FAQ) website, copied in January of 2025. It explains that EPSS scores have errors and why they have them.

"**Everyone knows this vulnerability has been exploited, why doesn't EPSS score it at 100%?**

EPSS should be thought of as pre-threat intelligence. Our detection systems, and by extension our threat intelligence, only report what they are programmed to look for, and then only what they can observe. For any given time period, vulnerability feeds can only describe a vulnerability in one of two ways, either it is *known to be actively exploited* or *we don't know* because we haven't observed exploitation, and absence of evidence is not the evidence of absence. If there is evidence that a vulnerability is being exploited, then that information should supersede anything EPSS has to say, because again, EPSS is pre-threat intel. If there is an absence of exploitation evidence, then EPSS can be used to estimate the probability it will be exploited. In other words, EPSS works best just below the obvious.

Having set that groundwork, there are two ways we have considered infusing threat intelligence information into EPSS. The first is after the model generates a probability we could modify the output if we see evidence of exploitation. The challenge is just how we would modify it. Just because we saw evidence of exploitation at some point in the past does not guarantee exploitation in the future. Modifying the score up to 100% would certainly overstate the exploitability of some vulnerabilities since not all vulnerabilities are 100% exploited in the future. This pushes us into the territory of producing a "magic number" that doesn't have much basis in science and will move EPSS away from being purely data-driven.

The second way we could infuse threat intelligence is by using them in an autoregressive model (explicitly including one or more variables for previous exploitation activity). Right now, exploitation activity is the target variable and we use things like "is this remotely exploitable?" and "would this allow code execution?" as the features in the model to help us understand just how exploitable a vulnerability is expected to be. In simpler terms, currently exploitation activity is only on the left side of the equation and everything else about the vulnerability is on the right side. We use the right side of the equation to predict the left side (exploitation activity). By putting in an autoregressive variable for exploitation, we would include something like "how often was this vulnerability exploited in the last 7 days?" on the right side of the equation (the features or independent variables of the model).

We have tried this approach with several variations. But remember that vulnerabilities are reported in one of two states, either it is exploited or we don't know if it's exploited. The challenge is the model assumes the two states are binary and opposite: exploited or not. The outcome is that the model learns to rate the vulnerabilities where we do not know if exploitation is occurring very low. Previous exploitation becomes such a strong indicator that the other variables become much less important. The outcome is that EPSS would become *reactionary* rather than *predictive*. Any vulnerability without observed and recorded exploitation activity would receive a low probability of future exploitation. This goes counter to our goal with EPSS to be pre-threat intel, and to distinguish between vulnerabilities that are likely versus unlikely to be exploited.

The net result is EPSS is pre-threat intel and works bets just below the obvious. It should be treated as a strength that it factors in the attributes and threat environment for vulnerabilities rather than just looking up the CVE ID for past exploitations. If you have evidence of active exploitations, leverage that, for everything else, you have EPSS."

Appendix B. List of Abbreviations and Acronyms

CISA
Cybersecurity and Infrastructure Security Agency

CPE
Common Platform Enumeration

CVE
Common Vulnerabilities and Exposures

CVSS
Common Vulnerabilities Scoring System

EPSS
Exploit Prediction Scoring System

KEV
Known Exploited Vulnerabilities

LEV
Likely Exploited Vulnerabilities

NVD
National Vulnerability Database