CONNECTEDNESS: A DIMENSION OF SECURITY BUG SEVERITY ASSESSMENT FOR MEASURING UNCERTAINTY

A PREPRINT

Chan Shue Long*
shue87655421@gmail.com

March 14, 2025

ABSTRACT

Current frameworks for evaluating security bug severity, such as the Common Vulnerability Scoring System (CVSS), prioritize the ratio of exploitability to impact. This paper suggests that the above approach measures the "known knowns" but inadequately addresses the "known unknowns" especially when there exist multiple possible exploit paths and side effects, which introduce significant uncertainty. This paper introduces the concept of connectedness, which measures how strongly a security bug is connected with different entities, thereby reflecting the uncertainty of impact and the exploit potential. This work highlights the critical but underappreciated role connectedness plays in severity assessments.

Keywords Attack Surface · Cyber Risk Quantification · Philosophy of Cybersecurity · Uncertainty

1 Introduction

The adage "we can not control what we can not measure." [Verendel, 2009] underscores the centrality of quantification in cybersecurity. Yet, as Knight and Jones [Knight and Jones, 2002] observed, the conflation of risk and uncertainty persists [2]. For instance, CVSS's exploitability and impact metrics collapse multi-faceted vulnerabilities into singular values, disregarding the uncertainty inherent in unexplored attack vectors and side effects.

Today's severity assessment frameworks usually prioritize two dimensions: (1) the likelihood of a vulnerability being exploited and (2) the severity of its impact. The state-of-the-art scoring system CVSS measures them correspondingly with "exploitability metrics" and "impact metrics" [Mell et al., 2022][3]), and they are only capable of capturing the risk of a single exploit path. When there exist multiple possible exploit paths and multiple possible side effects, CVSS fails to address this situation and forces the user to use a single value (for each sub-item in the metrics) to represent the exploitability of multiple possible exploit paths and the impact of multiple possible side effects - in such cases, it fails to see a security bug as a set of possible security events and measure the effect of the induced uncertainty. In such situations, uncertainty plays an important role in severity analysis; hence, we need a way to measure the impact of uncertainty.

We will argue that connectedness is a good measure of uncertainty. First, we need to establish the view: "security bug is a set of possible security events".

*homepage: https://katsuragicsl.github.io/

[2]The practical difference between risk and uncertainty is "that in the former, the distribution of the outcome in a group of instances is known (either through calculation a priori or from statistics of past experience), while in the case of uncertainty this is not true" [Knight and Jones, 2002]. We can hereby define risk as the "known knowns" and uncertainty as the "unknown unknowns" as suggested by Simpson [Simpson, 2024]. For example, a microservice having a 3 percent chance of melting down and being out of service is a risk; knowing that a feature of a given application is likely to be problematic without knowing its exact problems and how likely they occur, is an uncertainty.

[3]also see `https://www.first.org/cvss/v4.0/specification-document`

2 Security bug is a set of possible security events

A security bug can have multiple possible exploit paths and multiple possible side effects. Let us consider XSS as an example. In earlier days when XSS prevention techniques were not mature and feasible libraries were not abundant, people used custom filters to sanitize user input before reflecting it in the response, such as catching and removing dangerous elements in the input, HTML-encode the input, etc. However, these methods were not very effective, due to their bypasses of which many offensive security practitioners must be familiar with.

The main cause of the difficulties of the early XSS defense attempts faced is that there were many possible exploit paths, even if they were not known until someone published them: the user input reflected in the response body has connections with many different entities such as:

1. an HTML element
2. an attribute of an existing element [4]
3. the inside of existing JavaScript code
4. how HTML is parsed in different browsers and different libraries [5]

If we consider the impact of an XSS, we will also see multiple possible side effects, such as cookie stealing [6], phishing [7], cross-origin cookie leakage [8].

Hence, one should not see XSS as a single event, but as a set of possible events including:

1. the user input creates a dangerous HTML entity
2. the user input creates an entity with a dangerous attribute
3. the user input creates a dangerous attribute in an existing entity
4. the user input creates an entity which confuses the HTML parsing logic

3 Connectedness

3.1 The more uncertainty, the more potential risk

With hindsight, even if we lived in the early 2000s, we should have expected many possible ways to exploit XSS utilizing different objects related to the logic of how a webpage is rendered and how HTML is parsed. For example, even if we do not know that it is possible to exploit it by utilizing event handlers when the input is reflected in the `href` attribute of a `<a>` tag, we should still see it as a possible way. We do not know the actual way to exploit, but we do know that the behavior "user input getting reflected in the response" is connected with all components related to how a webpage is served, in particular the `<a>` tag and its `href` attribute; we just do not know exactly how and how likely it allows an exploit - this is the "known unknown". Once we find out an actual exploit, it becomes a "known known".

The more "known unknowns", the more uncertainty we have, and the more we should expect that some of them will one day turn into "known knowns". Equivalently, if a risk is a potential negative security event, we can see "uncertainty" as a potential risk.

3.2 Definition of connectedness

We hereby introduce the concept of **connectedness**: given a behavior (a system's behavior of interest: including security bugs), connectedness is a quantity describing the intensity of the behavior with other entities. The intensity consists of the number of entities connected with the given behavior and the strength of their connection. The more and stronger connections a given behavior has, the higher its connectedness. The strength of a connection is determined by how closely the given behavior is related to the entity.

[4] such as the href attribute of an existing a tag

[5] for example see https://portswigger.net/research/bypassing-dompurify-again-with-mutation-xss

[6] which is usually the main concern.

[7] by disguising a phishing attempt as a genuine content served by a trusted domain, through its url.

[8] in case the cookie of the another site is scoped to parent domain, and the vulnerable site is another subdomain. This is hypothetical, but still a possible side effect, although not a strong one.

In the case of XSS, the behavior of interest is that the user input pollutes the response and causes unexpected JavaScript execution. The entities connected with this behavior consist of HTML elements, attributes of existing elements, the inside of existing JavaScript code and the parser differentials 2. The first three have strong connections with XSS as they would be directly affected by the user input - they are what would be served to the users; the last one has a weaker connection since its relation with XSS is more indirect.

To further illustrate the concept of connectedness, we study two examples in the next section.

3.3 Examples

1. Input reflection

   Rated as informational usually (for example by Tenable[9]). An input reflection can be seen as "a (reflected) XSS that did not make it". It connects to everything that a reflected XSS connects to, hence it has the same known unknowns, except that none of them has turned into an actual risk. However, as discussed in 3.1, there are multiple potential risks.

2. Missing referrer policy

   Rated as informational usually (for example by Tenable[10]). By default the policy will be `strict-origin-when-cross-origin` when the referrer policy header is not set. Comparing to the above issue, the possible exploit paths and the possible side effects are much lesser - obviously it depends on the threat. For example we assume that "leaking" the referrer to the same origin is not problematic; we also do not consider browser errors in managing the same origin policy (since in that case we could not do much about it as a webapp engineer; and much worse things would happen). If we assume that "leaking" the referrer to the same origin is a problem, for example path A and path B are served by different microservices, and for some reasons we do not want the microservices know which requests are directed from another microservice, then this referrer leak would be a problem. In usual settings, there are not many ways one can manipulate this issue. Hence its level of connectedness is comparatively lower than the above issue.[11]

Now we could see the difference made by adding connectedness into the analysis of severity: in the CVSS framework, both are usually treated as "informational" issues, inducing more or less the same level of risk.

However, they have very different levels of connectedness. If a security practitioner had to prioritize the triage/fix of one over that of the other, the input reflection issue should be prioritized for it has more potential risk. [12]

By taking connectedness into account, one could differ security bugs with seemingly equal importance.

3.4 Limitations

The first limitation of applying connectedness on severity analysis is that whether an entity is considered as connected to the security bug in concern depends on the threat model of the system. If our threat model excluded a threat scenario, those entities that have meaningful interactions with the security bug only in that threat scenario will not be considered as connected with the security bug. Hence different threat models give different outcomes in the analysis of connectedness of the same security bug. See the footnote of example 2 in section 3.2.

The second limitation is that it is less objective to determine the strength of a connection, compared to current metrics, such as user interaction and privileges required, etc.

In defence of the first limitation, the author considers it a common problem of all severity analysis frameworks: security practitioners have to consider the threat model of the system in concern in order to accurately assess the severity of a given security bug. For example, if the system does not concern itself with the referrer being passed between different paths of the same origin, there is no point in considering its impact and possible exploits.

In defence of the second limitation, the author considers that in practice it is subjective enough for security practitioners to come to a consensus on the connectedness of a security bug, once they listed out and discussed all the connected

[9] `https://www.tenable.com/plugins/was/114135`

[10] `https://www.tenable.com/plugins/was/98527`

[11] there are some possible side effects that were not mentioned, for example one could imagine that if there exists a bug in the server which causes DoS when a long referrer header is received, then the missing referrer policy might increasing an epsilon of chance of such DoS happens. But the connection between the missing referrer policy issue and this possible behavior is very weak (since the chance is small and such DoS could have been triggered by other means more efficiently), so it does not contribute much to the level of connectedness. I hope the reader is convinced that overall the input reflection issue still has higher level of connectedness.

[12] The author also believes that the majority of security practitioners would do the same in such situation.

entities. Indeed, it is possible for a security practitioner to miss certain entities and mis-evaluate the connectedness of a security bug, but this type of errors is in the category of "unknown unknowns"; also by discussions with other security practitioners they should realize what they missed. "Unknown unknowns" in severity analysis should not and could not be tackled with the methods of evaluation [13], but with something in a higher level of the abstraction ladder, such as continuous threat modeling, to (hopefully) reduce the "unknown unknowns".

4 Conclusion

In this paper, the author defined the concept of connectedness and showed how to apply it in the analysis of security bugs. The author also showed how it can differ security bugs that seemingly have the same severity in traditional frameworks.

There are limitations in the application of this concept: it depends on the threat model of the system. Also, while it helps us to evaluate the "known unknowns", it does not help us to tackle the "unknown unknowns": our analysis of connectedness on a given security bug could also be faulty - it is possible for us to miss significant exploit paths and side effects, and mistakenly believe that a particular security bug is of low connectedness.

However, it does not hurt the fact that connectedness helps assess the "known unknowns" and should be included in the severity analysis of security bugs.

What we should take away from this paper is that uncertainty can strongly affect the severity of a security bug, and we should include connectedness as a measure of uncertainty, in order to achieve more accurate estimations of the severity of security bugs.

5 Acknowledgement

The author would like to thank Wong Wai Tuck for his comments on an early draft of this paper.

References

Vilhelm Verendel. Quantified security is a weak hypothesis: a critical survey of results and assumptions. In *Proceedings of the 2009 Workshop on New Security Paradigms Workshop*, NSPW '09, page 37–50, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605588452. doi:10.1145/1719030.1719036. URL https://doi.org/10.1145/1719030.1719036.

F.H. Knight and D.E. Jones. *Risk, Uncertainty and Profit*. Warriors (Washington, D.C.). Beard Books, 2002. ISBN 9781587981265. URL https://books.google.com.hk/books?id=Im2dnQAACAAJ.

Andrew Simpson. Into the unknown: the need to reframe risk analysis. *Journal of Cybersecurity*, 10(1):tyae022, 11 2024. ISSN 2057-2085. doi:10.1093/cybsec/tyae022. URL https://doi.org/10.1093/cybsec/tyae022.

Peter Mell, Jonathan Spring, Dave Dugal, Srividya Ananthakrishna, Francesco Casotto, Troy Fridley, Christopher Ganas, Arkadeep Kundu, Phillip Nordwall, Vijayamurugan Pushpanathan, Daniel Sommerfeld, Matt Tesauro, and Christopher Turner. Measuring the common vulnerability scoring system base score equation, 2022-11-15 05:11:00 2022. URL https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=935413.

[13] since no matter what evaluation method one uses, by definition there will always be "unknown unknowns".