Thou Shalt Discuss Security: Quantifying the Impacts of Instructions to RFC Authors

Justin Whitaker
North Carolina State University
jdwhitak@ncsu.edu

Sathvik Prasad
North Carolina State University
snprasad@ncsu.edu

Bradley Reaves
North Carolina State University
bgreaves@ncsu.edu

William Enck
North Carolina State University
whenck@ncsu.edu

ABSTRACT

The importance of secure development of new technologies is unquestioned, yet the best methods to achieve this goal are far from certain. A key issue is that while significant effort is given to evaluating the outcomes of development (e.g., security of a given project), it is far more difficult to determine what organizational practices result in secure projects. In this paper, we quantitatively examine efforts to improve the consideration of security in Requests for Comments (RFCs)— the design documents for the Internet and many related systems — through the mandates and guidelines issued to RFC authors. We begin by identifying six metrics that quantify the quantity and quality of security informative content. We then apply these metrics longitudinally over 8,437 documents and 49 years of development to determine whether guidance to RFC authors changed these security metrics in later documents. We find that even a simply worded — but effectively enforced — mandate to explicitly consider security created a significant effect in increased discussion and topic coverage of security content both in and outside of a mandated security considerations section. We find that later guidelines with more detailed advice on security also improve both volume and quality of security informative content in RFCs. Our work demonstrates that even modest amounts of guidance can correlate to significant improvements in security focus in RFCs, indicating a promising approach for other network standards bodies.

CCS CONCEPTS

• **Networks** → **Security protocols**; • **Security and privacy** → *Usability in security and privacy*.

KEYWORDS

Requests for Comments; Internet Standards; Network Security; Text Analysis

ACM Reference Format:
Justin Whitaker, Sathvik Prasad, Bradley Reaves, and William Enck. 2019. Thou Shalt Discuss Security: Quantifying the Impacts of Instructions to RFC Authors. In *5th Security Standardisation Research Workshop (SSR'19), November 11, 2019, London, United Kingdom.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3338500.3360332

SSR'19, November 11, 2019, London, United Kingdom

ACM ISBN 978-1-4503-6832-2/19/11…$15.00
https://doi.org/10.1145/3338500.3360332

1 INTRODUCTION

The security of networks and networked systems is of paramount importance, and it is commonly recognized that trustworthy systems do not occur by accident. Rather, security must be considered from the very beginning of a project in order to prevent major errors [57]. Accordingly, efforts like the Security Development Lifecycle provide extensive guidance on the process by which secure software should be designed and implemented to improve security.

While prior work has sought to understand how the software development process contributes to security of completed projects [3], to the best of our knowledge such questions have not been asked of network standards. This is a vast oversight. Not only are network standards substantial development undertakings in their own right, network protocols are not as easily patched or replaced and can be operational for decades. For example, secure extensions for DNS and BGP have existed for many years, but are rarely used [8, 23].

In this paper, we seek to understand how instructions to network standard authors affect network security through the lens of Request for Comments (RFC) documents. Requests for Comments are used to define new network protocols, enhance existing protocols, and codify best practices. Not only are these documents freely available, but they are developed transparently in public meetings and through public online venues (e.g., mailing lists). The availability of these documents — with the first dating back to 1969 — provides an unprecedented window through which to longitudinally analyze security consideration in network standard designs.

Unlike the secure development lifecycle, which establishes a sophisticated process model for organizations to follow, published guidance to RFC writers is limited. It primarily consists of two critical interventions — a mandate to include a distinct "security considerations" section and a set of guidelines for that section. As a result, it is unclear how much improvement in security coverage we should realistically expect from such minimal guidance. We seek to quantitatively evaluate the effects of this guidance by measuring the security informative content that appears in RFCs before and after the implementation of the mandate and guidelines.

We make the following contributions:

- **Define Quantitative Metrics:** We define six metrics to measure the security content and quality of RFCs. Among

our 16 key findings, we were surprised to learn that a simple mandate to discuss security in a dedicated section led to a substantial increase in both the size and quality of the mandated section and the amount of security discussion outside the mandated section. The later introduction of more detailed guidelines also had a significant positive effect.

- **Identify Security Informative Text:** We define "security informative" content, show that experts can label security informative content with high interrater reliability, and create and evaluate a machine learning classifier to identify security informative content.
- **Examine Development Process:** We analyze how security coverage changes over the course of developing RFC standards. We learn that successfully published draft standards tend to have longer security considerations sections, discuss a greater number of topics, and include more security discussion throughout the document. We also learn that roughly half of RFCs have a low amount of security content added initially that remains low, while the other half see a gradual increase in security content as the draft nears completion.

This paper focuses on a quantitative analysis of RFCs. We recognize that many factors — including organizational culture — also impact security outcomes, yet are beyond the visibility of researchers. Nevertheless, by performing the first security analyses of the network standards development, we take the first steps to better understanding this crucial issue.

The paper is organized as follows. In Section 2, we discuss the background of RFCs and their publishing organizations. Section 3 describes our metrics and other aspects of our methodology. In Section 4, we develop and evaluate a model that classifies security-informative paragraphs. Section 5 describes our analysis of security considerations sections, while Section 6 describes our analysis of all security informative content. In Section 7, we conduct case studies on draft standards, examples provided by the RFC 3552 "Security Considerations" section guidelines, and a known vulnerable RFC. Section 8 provides additional discussion, and Section 9 provides recommendations for future standards writers. Section 10 provides a discussion of related work, while Section 11 concludes.

2 BACKGROUND

Requests for Comments (**RFC**s) started as informal memos between Internet researchers. The first, *Host Software*, was published in April 1969 [10] — the same year ARPANET was established. RFCs have evolved from their nascent form to become structured communiques for relaying protocol standards and other information to the Internet technology community. They are technical documents used to specify new Internet protocols, describe or update existing protocols, and promote current best practices.

RFCs are numbered in their sequence of publication, though some RFC numbers have never been issued. They are authored, reviewed, and published by Internet researchers and engineers in streams for the Internet Architecture Board (**IAB**), Internet Engineering Task Force (**IETF**), and Internet Research Task Force (**IRTF**). There is also a stream for independent authors to submit RFCs and a legacy stream for older RFCs. RFCs follow a particular submission process depending on their stream, are then edited by the RFC Editor, and finally returned to the authors for final review before being published [2]. RFCs additionally have a status which is either "Proposed Standard," "Informational," "Unknown," "Experimental," "Best Current Practice," "Draft Standard," "Internet Standard," or "Historic." Most RFCs are Proposed Standards, followed by Informational. RFCs that specify protocol standards begin as Proposed Standards, are matured in a series of Draft Standards, and finally become Internet Standards. Revisions of Internet Standards start the process again as Proposed Standards.

RFCs contain natural language text as well as code examples and figures conveyed in ASCII art. Though they were originally very free-form, RFCs are now highly structured. There are style conventions which much be followed for language, punctuation, capitalization, citations, and abbreviations, as well as a prescribed structure [16]. Currently, RFCs must contain a first-page header, title, abstract, "Status of This Memo", copyright notice, table of contents, body, and "Author's Address" sections. The body must contain an "Introduction" section and a "Security Considerations" section. If applicable, RFCs are expected to contain "Requirements Language," "IANA Considerations," "Internationalization Considerations," and "References" sections as well.

The "Security Considerations" section (**SCS**) is the prime section for the security impacts of an RFC. The SCS is the designated area for authors to discuss security issues relevant to the RFC. The first RFC to include an SCS was RFC 1060 [47], published in 1990. The section ironically read:

> *Security issues are not discussed in this memo.* [47]

After SCSs first appeared with RFC 1060, they fast became a common feature of RFCs, but were included on a purely voluntary basis. SCSs were made mandatory by RFC 1543, *Instructions to RFC Authors* [42], which was published in October 1993. We refer to this RFC as the **mandate** in this paper. The SCS was one of many sections enumerated in the RFC to now be required. The mandate for "Security Considerations" sections, in its entirety, read:

> *All RFCs must contain a section near the end of the document that discusses the security considerations of the protocol or procedures that are the main topic of the RFC.* [42]

This was later considered insufficient guidance on how to discuss security. Although there was an updated *Instructions to RFC Authors* [43] in 1997, its instructions on SCSs remained the same.

Ten years after RFC 1543, in July 2003, the second milestone RFC regarding SCSs was published. This was RFC 3552, *Guidelines for Writing RFC Text on Security Considerations* [46]. We refer to this RFC as the **guidelines**. These were written in response to a perceived deficiency in the security discussion of prior RFCs. RFC 3552 provided additional instructions on writing SCSs. It urged authors to conduct thorough threat modeling prior to writing their RFCs, and specifically required authors to discuss eavesdropping, replay, message insertion, deletion, modification, man-in-the-middle, and denial-of-service attacks. If any of these attacks were out of scope, it required authors to describe why. The guidelines required an assessment of authentication methods, assumptions, and lower level services required by the protocol.

RFC 3552 also gave a crash course in security. It discussed the goals of security (confidentiality, integrity, and availability) and

additional properties such as non-repudiation. The RFC gave an overview of the Internet's threat model, including active and passive adversaries. It specified types of attacks, such as man-in-the-middle attacks, and described security mechanisms, technologies, and protocols. This thorough treatment made the RFC 43 pages in total.

Lastly, the guidelines provided two SCSs that were deemed exemplary. The first was a hypothetical further revision to the SMTP update in RFC 2821 The revision added inline notes and supplemented the SMTP update's communication security discussion. The second was from the Virtual Router Redundancy Protocol standard in RFC 2338 with added inline notes. These example SCSs will be the subject of a case study in section 7. RFCs are the cornerstone of the development of Internet technologies. Because of this, it is important that they include high quality security discussion. In subsequent sections, we will analyze the effects of time, the RFC 1543 SCS mandate, and the RFC 3552 SCS guidelines on metrics quantifying security discussion in RFCs.

3 METHODOLOGY

In this paper, we seek to characterize both the security content of RFCs and the effect of the mandate and guideline RFCs on security content in RFCs. In this section, we begin by defining the metrics we use to quantitatively measure this security content. To aid this effort, we also provide a novel definition of "security informative" text. We then describe the RFC data we use in our study, and conclude with an overview of the statistical procedures we use to characterize our findings. The following section describes the creation and evaluation of a classifier to identify security informative paragraphs, and later sections describe our findings.

3.1 Security Metrics

We aim to analyze what factors impact an SCS being included in an RFC, how long RFCs are, the breadth of their discussion, and how much security discussion occurs in them. We also want to identify how much text in both the SCS and the remainder of the RFC is security discussion. To quantify security discussion in RFCs, we have developed six metrics:

SCS Presence measures whether an RFC has a clearly labeled SCS. This metric is binary.

SCS Word Count measures the length of SCS.

SCS Topic Coverage measures how many of 10 topics identified in the SCS Guidelines are discussed in the SCS based on the presence of keywords.

SIP Word Count measures the word count of all security informative (SI) content in the RFC, regardless of whether it is inside the SCS. We discuss our definition of "security informative" in detail later in this section.

Compartmentalization indicates the ratio of SI content within the SCS to the total amount of SI content. For example, a compartmentalization of 40% indicates that 40% of all SI content is inside the SCS, with 60% of SI content falling outside the SCS.

Density measures how much of the SCS content is actually security informative. Density is the ratio of SI content in the SCS to total content in the SCS. For example, a density of 80% indicates that 20% of the content in the SCS is *not* SI, while the remaining 80% is SI.

The latter three metrics rely on a determination of whether text is "security informative." In this paper, we make this determination on a paragraph level and refer specifically to security informative paragraphs (SIPs). We use a machine learning classifier, discussed in Section 4, to label RFC paragraphs as security informative. Because of the importance of context, lower levels of granularity (e.g., sentence-level) are more difficult to obtain (and would require solving decades-old problems in NLP), and are ultimately no more useful for our purposes than paragraph-level metrics.

Our definition of "security informative" text is:

> *Text that a security expert would recognize as being intended to alert the reader to a potential security issue or concern.*

Discussion of privacy, security protocols, security tools such as firewalls, adversaries, threat models, insecure use cases or configurations, authentication, and authorization are all security informative. Because of this, there will inherently be more SIPs in an RFC which discusses a security topic. We acknowledge that this definition is necessarily subjective, and we chose this definition after rejecting several alternatives, including a definition of "security relevant" text. Because any functionality or implementation could have security implications (whether obvious or not at time of creation), this latter definition was too imprecise and broad to be useful.

In Section 4, we discuss the results of asking two security experts to independently code paragraphs as SI or not-SI. We show that such coding has high inter-rater reliability, providing confidence that our definition is meaningful and reproducible. To better provide the reader with insights into what is or is not SI, we provide three example paragraphs from this analysis.

An example SIP which both labelers agreed upon is:

> *However, it should be noted that an attacker that has some knowledge, such as of MAC addresses commonly used in DHCP client identification data, may be able to discover the client's DHCP identify by using a brute-force attack. Even without any additional knowledge, the number of unknown bits used in computing the hash is typically only 48 to 80.* [54]

This is clearly a SIP because it is discussing an attack. An example of a non-SIP, which both labelers agreed upon, is:

> *The values 1-47 are reserved for algorithms for which an RFC has been approved for publication. The values 48-63 are reserved for private use amongst cooperating systems. The values 64-255 are reserved for future expansion.* [41]

This is clearly not a SIP because it is not discussing any security topics. An edge case the two labelers disagreed on is:

> *One may notice that many documents that explain the DNS and that are intended for a wide audience incorrectly describe the resolution process as using QNAME minimization (e.g., by showing a request going to the root, with just the TLD in the query). As a result, these documents may confuse readers that use them for privacy analysis.* [55]

This was an edge case the two labelers disagreed on. The first labeler argued it was security informative because the authors are considering confusion around privacy information. The second labeler argued it was not security informative because it does not itself include any privacy discussion. The final label was decided by a third labeler was SIP for the same reason as the first labeler.

3.2 Data Sources

In this paper, we examine two datasets: the set of all published RFCs and the set of all RFC draft standards.

The RFC dataset is composed of the text and associated metadata of 8,437 RFCs up to RFC 8453,[1] obtained from the official RFC repository [2]. We verified that the "missing" 16 RFCs were never actually issued. All RFCs are in plain-text, ASCII format.

The set of draft standards was also acquired from the official RFC repository [2] on February 7, 2019. This set includes both historic drafts and in-progress drafts. Not all RFCs are standards, but all RFC standards begin as draft standards. A published RFC may have numerous unpublished draft revisions until it is published as an RFC. RFCs do not change after publication, and changes to an accepted standard are only made through external errata or a subsequent standard.

In our analysis, we associate 125,946 draft documents to 33,706 draft standards by matching documents to the unique draft standard identifier (e.g., `draft-ietf-drums-smtpupd`, which became RFC 2821 revising SMTP). The IETF website [1] provides a link from RFC to its pre-publication draft, and we use this information to identify which drafts became standards. We identified 6,831 published draft standards and 26,875 non-published draft standards. Note that non-published drafts may still be accepted at a later+ point. Draft sets that are not published often include a final document consisting of a short description of the fate of the draft standard, such as being rejected or merged into another project. To prevent these status updates from affecting our results, we ignore the last document in a draft document set if it contains less than three paragraphs.

3.3 Statistics Preliminaries

We use several statistical measures to quantify the effects of time, the mandate, and the guidelines on our metrics. To assess the mandate's impact, we compare RFCs published before the mandate to those published between the mandate and guidelines. To evaluate the guideline's impact, we compare RFCs published between the mandate and the guidelines to those published after the guidelines. We do this to prevent the impacts of the mandate from influencing the measurements of the guideline's impacts and vice versa.

The first statistic is Spearman's rank correlation coefficient, ρ. This measures the monotonicity of a relationship between two variables. A positive value indicates a positive relationship, and a negative value indicates a negative relationship. The absolute value of ρ indicates the degree of monotonicity. A value of zero indicates no relationship, and an absolute value of one indicates a perfectly monotonic relationship. We use Spearman's ρ over the Pearson correlation because Spearman's ρ does not assume linearity. We use it to quantify the relationship our metrics have with time. As in common in other fields[35], we interpret an absolute value of ρ equal to or greater than 0.9 to indicate "very high correlation," than 0.7 to indicate "high correlation," than 0.5 to indicate "moderate correlation," than 0.3 to indicate "low correlation," and less than 0.3 to indicate "negligible correlation."

We also use t-tests to measure if the difference between the means of two populations is significant. If the p-value of a t-test is less than α, we have identified a statistically significant difference between the two populations. We use Cohen's d to quantify the effect size, or magnitude of the differences between two populations given there is a statistically significant difference. We follow the interpretation guidelines provided by Sawilowsky [48], which describes a d of 0.01 as "very small," 0.20 as "small," 0.50 as "medium," 0.80 as "large," 1.20 as "very large," and 2.0 as "huge."

We select an initial α of 0.01 to determine the significance of our statistical tests. Our p-values must lie below α to be considered statistically significant. We additionally use Bonferroni correction to control family-wise error rate. Because we conduct three null hypothesis tests for each metric, we correct our α to 0.0033.

4 SECURITY-INFORMATIVE PARAGRAPHS CLASSIFIER

In this section, we create a classifier for security-informative paragraphs (SIPs) of RFCs to enable our SIP word count, compartmentalization, and density metrics. We pre-process RFC text into paragraphs which we partition into training and testing sets. We leverage a paragraph's presence in an SCS to automatically create noisy labels for the training set. We then design and apply a classifier which uses term frequency-inverse document frequency (TF-IDF) [45] vectorization, singular-value decomposition (SVD), and logistic regression. A large set of labeled training data is required to optimize the weights of our logistic regression model. To reduce the amount of manual labeling required for our training set, we labeled SCS paragraphs as SIP and others as non-SIP. We used regular expressions to identify paragraphs in SCSs. We hypothesize — and later verify — that SIPs occur both inside and outside SCSs, but occur more frequently within SCSs. This means that the labels of our training data are noisy, but provide meaningful information for learning to classify SIPs due to the sheer size of the data set. Because we know our training data labels are not always correct, the training set performance of our model is not a focus of our performance evaluation. We will instead evaluate the accuracy of our model on the manually labeled test set.

Can experts reliably identify security-informative content?: Before we further describe our classifier design and evaluation, we must first ensure that our definition of "security-informative" is meaningful. We do this by manually labeling 1,100 paragraphs, which we also use to evaluate our classifier's performance. These 1,100 paragraphs were randomly selected from the set of paragraphs present in RFCs, and were not used in the training of our classifier. Two raters (both paper authors) manually code our test set into three classes: "SIP", "non-SIP", and "malformed". "Malformed" paragraphs were affected by parsing errors or failures of the ASCII-art removal heuristic. We chose to label 1,100 paragraphs so as to have roughly 1,000 paragraphs labeled as either "SIP" or "non-SIP."
Finding 1: *Security-informative text can consistently be manually*

[1] *Framework for Abstraction and Control of TE Networks (ACTN)*, published August 2018 [7]

identified by experts. Cohen's kappa inter-rater agreement measure [18] quantifies the agreement of raters on labeling categorical data. A low kappa means raters frequently disagreed, while a high kappa means raters often agreed. We achieved a kappa of 0.742 across three classes on our test set, which may be interpreted as significant agreement[17]. This confirms our choice of definition for "security informative" and leads to our finding that security-informative text can be consistently manually identified by experts.

Data Pre-Processing: Because RFCs are unstructured text, preprocessing is required to remove irrelevant information and segment them into paragraphs. To remove irrelevant text which is not intended to inform the reader about a standard or convey other central information, we use regular expressions to remove the "Table of Contents," "Acknowledgements," "References," "Authors' Addresses," "Status of This Memo," and "Copyright" sections. An additional complicating factor was the inclusion of ASCII-art figures in RFCs. We created a heuristic to remove paragraphs containing a ratio of characters commonly used in ASCII-art (such as +, |, _, and others) above a selected threshold of 0.05. The character set and threshold were selected manually to minimize the number of ASCII-art figures included in the data set without excluding legitimate paragraphs. Finally, we removed the headers and footers on each page, and removed empty paragraphs caused by many consecutive newlines. We break text into paragraphs at consecutive double newline characters, which is the structure used in RFCs to delineate paragraphs. However, manual review revealed it was often the case that there are single sentences isolated this way. Therefore, we merged single-sentence paragraphs into the subsequent paragraph if doing so does not span a section boundary. Our training set was composed of the 127,551 paragraphs not included in our testing set.

Classifier Design: Our classifier uses TF-IDF vectorization followed by SVD, which is then input into a logistic regression model. We use TF-IDF to transform a sequence of characters into a vector that represents the importance of terms to a document. TF-IDF captures the relative frequency of a term in a document adjusted for how frequent the term is in the corpus. We removed uninformative terms by requiring a minimum document frequency threshold of 0.5% and removing stopwords. The result of our this TF-IDF vectorization is a vector of size $|V|$, where V is the selected set of terms from the corpus. The dimensionality of these vectors are reduced from 889 terms to 100 components with truncated SVD to reduce overfitting and improve computational efficiency before being input into the model.

We selected binary logistic regression as our classifier because of its robustness against overfitting. We used balanced class weighting, which weighs the error penalty of a class inversely proportional to its size, because of the disproportionate ratio of non-SCS paragraphs to SCS paragraphs. We also used balanced class weighting because we anticipated that our training data would contain many samples falsely labeled as non-SIP by our SCS heuristic.

Evaluation: We evaluated our classifier qualitatively and quantitatively. By inverting the logistic regression model's coefficient vector with training data's SVD transformation, we associated the classifier's input weights to terms in the TF-IDF vocabulary. We found the terms weighted heavily by the model were indeed more security-informative.

A third rater labeled conflicting data points in our test set to decide the final classes. The result was that 104 of test samples were labeled malformed, 221 were SIP, and 793 were non-SIP. We quantitatively evaluated our model on the test set and achieved an accuracy of 82%, with a recall of 56% and precision of 59%. We note that these accuracies are well within expected ranges for difficult natural language processing problems [27, 59]. Our classifier's performance is standard for this domain of text analysis, and we will use it to apply our SIP word count, compartmentalization, and density metrics to RFCs in subsequent sections.

5 SECURITY CONSIDERATIONS SECTIONS METRICS

In this section, we describe the SCS presence, word count, and topic coverage metrics in greater detail. We investigate how they were affected by time, the RFC 1543 mandate, and the RFC 3552 guidelines. Because we conduct three null hypothesis tests per metric in this section, we correct our initial α of 0.01 to 0.0033.

5.1 Presence

We first investigate the presence of SCSs in RFCs, which we measure by identifying "Security Considerations" section headings.

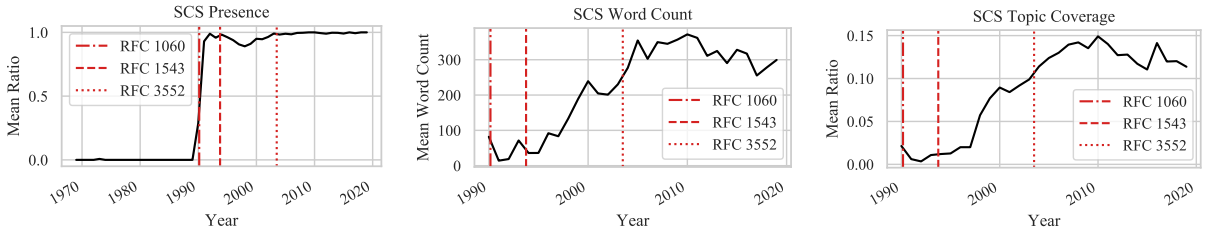
Finding 2: *Since the mandate and guidelines, SCSs are nearly always present.* Figure 1a shows the changes in the yearly mean SCS presence ratio in RFCs. Year of publication and SCS presence have a ρ of 0.917 ($p < 0.0001$), meaning that the rates of SCS presence have a very high positive correlation with time. The SCS presence ratio more than tripled from 27% to 94% after the mandate, with a significant t-test ($p < 0.0001$) and a Cohen's d effect size of 1.89. The average SCS presence ratio rose from 94% to 99% after the guidelines, with a significant t-test ($p < 0.0001$) and a Cohen's d effect size of 0.31. Although both had a statistically significant effect on SCS presence, the mandate had a huge effect and the guidelines had a small one.

Finding 3: *SCS presence increased dramatically in the year before the mandate.* The rates of including SCSs in RFCs rose from 0% to over 90% in a single year before the mandate. This may have two explanations: one is that the mandate formalized an existing consensus to include SCSs in RFCs. The other is that writers were aware of the pending mandate and responded before its official publication. Because SCSs are present in virtually all RFCs since the guidelines, if there is an absence of important security content, it is not due to the SCS being omitted.

5.2 SCS Word Count

We approximate the amount of security discussion in SCSs by the number of words they contain. We separate the SCS from the rest of the RFC and count the number of words in the section. This metric considers only RFCs that contain an SCS because its purpose is to identify changes in how SCSs are written, not whether they are present.

Finding 4: *Despite its lack of detailed guidance, SCS word count grew four-fold only after the mandate's publication. It nearly doubled after the guidelines.* Figure 1b shows the changes in the yearly mean SCS word count in RFCs. Year of publication and SCS word count



(a) SCS Presence ratio rapidly increased before the section was mandated.

(b) Later SCSs dwarf early ones by word count.

(c) Early SCSs had poor topic coverage compared to recent ones.

Figure 1: SCS presence, word count, and topic coverage findings

have a ρ of 0.779 ($p < 0.0001$). This indicates SCS word count grows longer with time. Mean SCS word count grew four-fold after the mandate, increasing from 38.91 to 169.73 after the mandate with a significant t-test ($p < 0.0001$) and a Cohen's d effect size of 0.507. It nearly doubled after the guidelines from 169.731 to 324.356 with a significant t-test ($p < 0.0001$) and a Cohen's d effect size of 0.366. Both the mandate and guidelines had a medium effect size on increasing the SCS word count.

5.3 Topic Coverage

The third SCS metric we analyzed is topic coverage. In this analysis, we aim to quantify the breadth of topics discussed in an SCS when present. We extracted a set of topics from the RFC 3552 SCS Guidelines' overview of security and associated a list of keywords from the SCS Guidelines with respect to each topic. We used the guidelines RFC as a source of topics because the guidelines are what authors are expected to follow when discussing security in RFCs. Measuring the inclusion rates of security topics put forward by the guidelines enables us to assess the extent to which the guidelines succeeded in broadening the security discussion in SCSs.

Topics were determined by the headings of sections in the SCS Guidelines. We identified initial keywords by using terms highlighted with all-capitalized letters in the sections and terms included in sub-section headings. We added the full names for acronyms and, because the SCS Guidelines were written in 2003, we also included modern equivalents for dated terms. An example of this is adding the modern term "HTTPS" for what was then referred to as "S-HTTP." We did this to prevent false negatives in newer RFCs that mention the same technologies or techniques discussed in the guidelines but with recent equivalents.

We pre-process the RFCs to make all characters lowercase, replace punctuation marks with spaces, and replace consecutive whitespace characters with a single space. If a term is contained in the set of terms used by an SCS, then we consider the topic the term is associated with to be covered. The topic coverage ratio for an RFC is the number of topics covered by its SCS out of the total number of ten topics.

Finding 5: *Simply mandating authors to include SCSs coincided with an eight-fold increase in topic coverage. Topic coverage further doubled after the guidelines.* Figure 1c shows the changes in the yearly mean topic coverage ratio. Year of publication and topic coverage ratio have a ρ of 0.785 ($p < 0.0001$). This indicates that topic coverage has a high positive correlation with date of publication and is steadily increasing with time. The mean topic coverage ratio rose nearly eight-fold from 0.009 to 0.068 after the mandate, with a significant t-test ($p < 0.0001$) and a Cohen's d effect size of 0.692. The mean topic coverage ratio nearly doubled from 0.068 to 0.130 after the guidelines, with a significant t-test ($p < 0.0001$) and a Cohen's d effect size of 0.441. Both the mandate and guidelines had a medium effect size on increasing topic coverage. This indicates that despite the mandate lacking guidance on which topics should be included in RFCs, authors may have been influenced by the mandate to discuss additional topics compared to before. It also indicates that the guidelines were successful in promoting the discussion of a broad range of security topics in SCSs.

5.4 Discussion

SCS presence increased to over 90% before the section was mandated. Interestingly, other metrics such as SCS word count and topic coverage improved only after the mandate. This may show that the mandate had a positive effect on security discussion. All metrics increased further after the guidelines. However, it is possible that the increase in SCS word count is solely due to the inclusion of non-SI content. The metrics in Section 6 will investigate this possibility.

6 SECURITY-INFORMATIVE PARAGRAPHS METRICS

The metrics defined in Section 5 are limited in their ability to quantify security discussion in RFCs because such discussion may occur outside of SCSs, and SCSs may contain non-SI paragraphs. In this section, we use the classifier described in Section 4 to implement SIP word count, compartmentalization, and density. These are important for assessing the total amount of security discussion in an RFC, how much of that discuss is isolated in the SCS, and how much of the SCS is security discussion. Because we conduct three null hypothesis tests per metric in this section, we correct our initial α of 0.01 to 0.0033.

6.1 SIP Word Count

The first metric we investigate is SIP word count, which is the total number of words in the paragraphs that are classified as SI by our model. But, not all paragraphs from SCSs are SI and not all SIPs are from an SCS. Because of this, SIP word count is a more accurate proxy for the total amount of text devoted to security discussion throughout an RFC than the previous SCS word count metric.

Finding 6: *SI content in RFCs has rapidly increased over time.* Figure 2 shows the changes in the yearly mean SIP word count. Year of

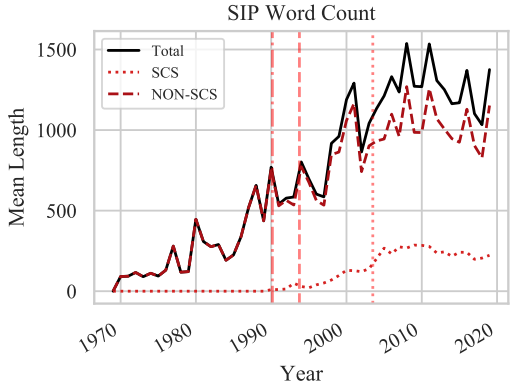


Figure 2: Yearly mean SIP word count. Security discussion from both inside and outside SCSs are increasing over time.

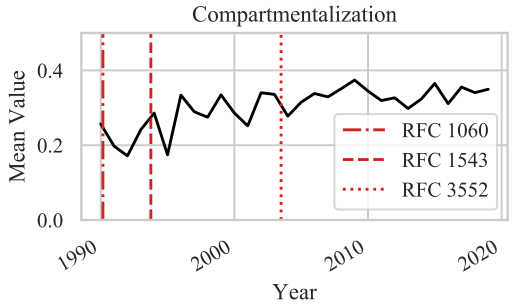


Figure 3: Yearly mean compartmentalization is increasing over time due to authors including more content in the SCS.

publication and mean annual SIP word count have a very high positive correlation with a ρ of 0.929 ($p < 0.0001$). The figure also contrasts SIP word count for text from SCSs and all other sections. Security discussion in SCSs steadily increased after the mandate and stabilized soon after the mandate at around 250 words on average.
Finding 7: *SI content nearly tripled after the mandate, and increased further after the guidelines. Particularly, even SI content outside of the SCS increased after the mandate.* The mean SIP word count nearly tripled from 318.784 to 952.187 after the SCS mandate, with a significant t-test ($p < 0.0001$) and a medium Cohen's d effect size of 0.47. The mean SIP word count rose 34% from 952.187 to 1277.204 after the SCS guidelines, with a significant t-test ($p < 0.0001$) and a small Cohen's d effect size of 0.18.

6.2 Compartmentalization

The second metric we investigate is compartmentalization, the ratio of SIPs in an RFC which are from its SCS. A compartmentalization of 1.0 means that all security discussion in that RFC occurs in its SCS, while a value of 0.5 means that half of security discussion occurs in the SCS. Because this metric relies on the presence of an SCS, we exclude RFCs not containing an SCS from this analysis. High compartmentalization may indicate good structure in an RFC, where security discussion occurs in the designated section and is thus easy for readers to locate. It also enables us to identify if authors relocate security discussion to or from the SCS.

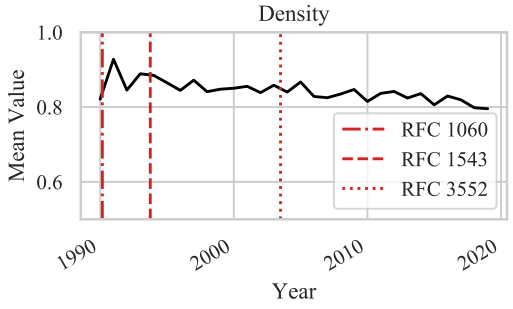


Figure 4: Yearly mean density is decreasing over time.

Finding 8: *Most security discussion occurs outside of SCSs.* Figure 3 shows the changes in yearly mean compartmentalization. Date of publication and compartmentalization have a high positive correlation with a ρ of 0.704 ($p < 0.0001$). This means that compartmentalization is increasing over time. Surprisingly, the majority of security discussion occurs outside of SCSs. Mean compartmentalization rose 21.9% from 0.237 to 0.289 after the mandate, with an insignificant t-test ($p = 0.006$). Mean compartmentalization rose 15.6% from 0.289 to 0.334 after the guidelines, with a significant t-test ($p < 0.0001$) and a small Cohen's d effect size of 0.152. This means that compartmentalization had a small increase after the guidelines, and not after the mandate.
Finding 9: *Increased compartmentalization of security discussion in the SCS is due to the addition of security discussion, not the relocation of content from elsewhere.* Security discussion is occurring more in the SCS. One explanation of why compartmentalization is increasing over time might be that the accompanying increase in SCS length is due to authors moving more security discussion to the SCS, instead of being due solely to an increase in the amount of security discussion. However, Figure 2 shows both SCS and non-SCS SIP word counts have a very high positive correlation with year of publication ($p < 0.0001$), and non-SCS SIP word counts did not decrease as compartmentalization increased. This means the rise in compartmentalization is due to the addition of SI content to the SCS, not the relocation of content from elsewhere.

6.3 Density

The final metric we present in this section is density. Density is the ratio of paragraphs in an SCS which are SI. An RFC with a density of 1.0 has an SCS which contains only SI paragraphs. High density may be a positive metric in RFCs because their SCS stays focused on its purpose, which is to convey SI content. Additionally, this metric is useful because it can capture the amount of non-security discussion in SCSs. Because this metric relies on the presence of an SCS, we exclude RFCs not containing an SCS from this analysis.
Finding 10: *Non-SI content in SCSs is slightly growing over time.* Figure 4 shows the changes in yearly mean density. Date of publication and density have a high negative correlation with a ρ of -0.709 ($p = 0.00001$). This means that density is decreasing over time. Mean density fell 3.6% from 0.882 to 0.851 after the mandate, with an insignificant t-test (p = 0.016). Mean density fell a further 2.7% from 0.851 to 0.829 after the guidelines, with a significant t-test (p = 0.00015) and a small Cohen's d effect size of 0.106. Density

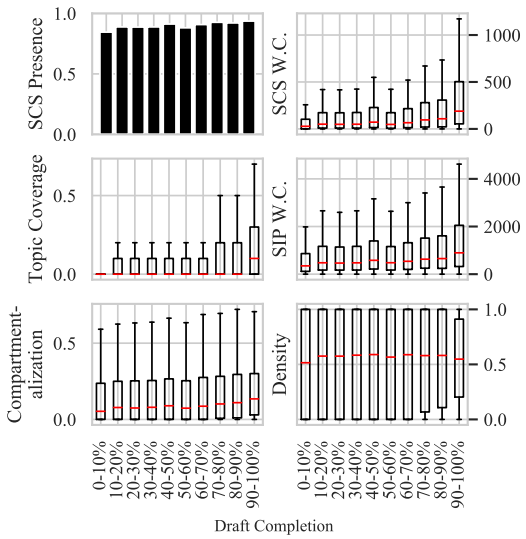


Figure 5: Draft metrics by percentage of completion. The upper 50th percentiles tends to increase for some metrics towards the final drafts.

slightly decreased after the guidelines, and not after the mandate.

Finding 11: *Longer SCSs tend to have proportionally more non-SI discussion than shorter SCSs.* We were curious about what caused this drop in density and hypothesized that it may be due to an increase in SCS length. Density and SCS word count have a ρ of -0.523 ($p < 0.0001$), which indicates that there is a moderate negative correlation between density and SCS word count.

6.4 Discussion

In this section, we used the classifier developed in Section 4 to investigate these metrics which quantify security discussion in more nuanced ways than we were capable of in Section 5. We found SCS length is increasing both due to an increased amount of SI and non-SI content. In the next section, we apply the metrics developed in Sections 5 and 6 to conduct several case studies.

7 CASE STUDIES

We conduct three case studies with the metrics developed in Sections 5 and 6. These investigate draft standards, the example SCSs in the guidelines, and a vulnerable RFC.

7.1 Draft Standards

We now apply our metrics to draft standards obtained from the IETF to investigate whether they are correlated with a draft standard being published, and to see how authors discuss security throughout the draft standard process. To assess how metrics change as draft standards approach completion, we place drafts documents into bins for each 10% of completion. For example, if a document is the first of two drafts for a standard, it will be placed in the 50-60% bin. This is required because there may be an arbitrary number of draft documents for a particular draft standard.

Finding 12: *All our metrics increase as draft standards approach completion.* Table 1 shows the values of ρ and p for our metrics,

Table 1: All metrics have a statistically significant high or very high positive correlation with draft completion.

Metric	ρ	p
SCS Presence	0.806	0.005
SCS Word Count	0.939	0.00005
Topic Coverage	0.855	0.0016
SIP Word Count	0.855	0.0016
Compartmentalization	0.806	0.0049
Density	0.794	0.006

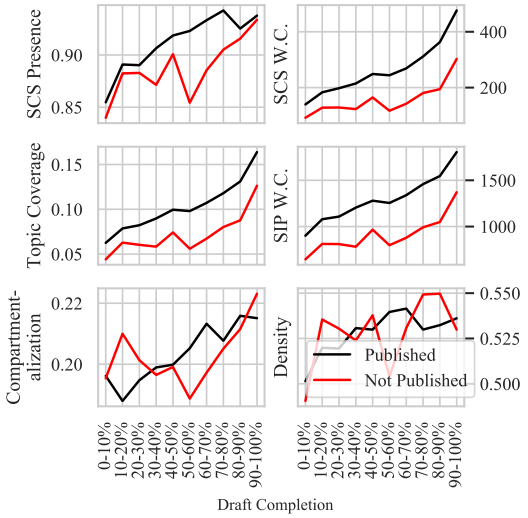


Figure 6: Mean SCS presence, SCS word count, topic coverage, and SIP word count tend to be higher throughout all drafts in standards which are eventually published.

which have statistically significant high correlation with completion.

Finding 13: *There is a split distribution in draft standard metrics by completion: For about half, security discussion is included in the first draft and changes little. For the rest, it increases as drafts approach completion.* Figure 5 shows the distribution of our metrics by draft completion. For SCS and SIP word count, the median value changes little along draft completion, but the upper 50th percentile increases greatly. A similar effect can be observed in topic coverage, although the median value increases slightly for the final bin.

Finding 14: *Some authors may be quickly conforming to IETF norms in the final drafts.* Figure 5 shows that the quartiles for density remain far apart until completion nears, at which point they noticeable tighten together. The upper 50th percentile for SCS word count and topic coverage also increase quickly. If there is an expectation regarding the amount of security discussion in the SCS, this may cause drafts with low density to add it in the final drafts and for density to increase. If there is an expected length of SCSs, this may cause drafts with dense SCSs to add filler content in the final drafts and for density to decrease.

Finding 15: *SCS presence, SCS word count, and SIP word count tend to be higher throughout the draft process for accepted draft standards.* Figure 6 shows the mean values for our metrics for accepted and

rejected draft standard by completion. We see that although the values increase for both classes, accepted drafts consistently have a greater SCS presence ratio, SCS word count, and SIP word count throughout the draft process. This trend does not hold true for compartmentalization and density.

We used the IETF's draft standards to identify how authors approach security in the draft standard process, and differences between the qualities of accepted and non-accepted drafts. We found a stark contrast in approaches where some authors include security discussion in the first draft, after which it remains largely unchanged, and that others gradually add security discussion along drafts. Neither approach is alone tied to a draft being accepted or not. Instead, all metrics tend to increase along draft completion for accepted and rejected draft standards, but that SCS presence ratio, SCS word count, SIP word count, and SCS topic coverage are consistently higher in accepted draft standards throughout the draft writing process. We expect that this is due to RFCs with high values for those metrics tending to be more fully developed than those with low values for them.

7.2 RFC 3552 SCSs

In this section, we investigate the metrics of the two example SCSs provided to RFC authors by the guidelines. We do this to gain insight on which metrics were being emphasized by the IETF at the time the guidelines were published. The first example given by the guidelines was a retrospective rework of RFC 2821, *Simple Mail Transfer Protocol*. RFC 2821 revised RFC 821 by adding a previously absent SCS. The retrospective SMTP SCS put forth in the guidelines added further discussion on communication security topics. The second example was from RFC 2338, *Virtual Router Redundancy Protocol*, to show a high quality existing SCS. This one was not significantly altered in the guidelines.

In total, we investigate the metrics of four actual or synthetic RFCs. The first two are the original SMTP standard, RFC 821, and the revised SMTP standard, RFC 2821. We directly use these RFCs without modification. The third RFC is the retrospective improvement of RFC 2821 included in the guidelines. Because only the SCS is provided in the guidelines, we manually replaced the SCS of RFC 2821 with the retrospective SCS given in the guidelines to create a synthetic RFC before applying our metrics. For our last document, we use RFC 2338 without modification because its SCS was not significantly altered in the guidelines.

The metrics of these documents are shown in Table 2. Because the original SMTP standard had no SCS, the only applicable metric is SIP word count. Its SIP word count was higher than the mean, but the lowest of any of the RFCs investigated in this section. The SMTP revision's SIP word count jumped dramatically with more than five times the amount of security discussion compared to its predecessor. The revision's compartmentalization was low, however, indicating that much of this added discussion did not occur within the SCS. The only topic its SCS matched was authentication. Although the SCS word count was high, the density was very low. Low compartmentalization combined with very low density indicates that the SCS did not contain much more security content than other parts of the RFC. This may have been a factor in its low SCS topic coverage compared with the remaining two RFCs we investigate in this section.

The additions to RFC 2821's SCS by the guidelines in our synthetic RFC more than doubled the word count of the SCS and greatly increased the SCS topic coverage. The SIP word count, compartmentalization, and density in the document increased. Compartmentalization rose substantially from the 37.7th percentile to the 61.6th percentile. Density rose much less and remained much lower than the mean. This may indicate that density was not an emphasized in the guidelines as other metrics such as compartmentalization.

The RFC chosen by the guidelines with a positive example of an SCSs was RFC 2338, *Virtual Router Redundancy Protocol* (VRRP). It had high topic coverage and SCS and SIP word counts compared to the population. Compartmentalization and density were not much higher than the mean. This indicates other factors were considered more important for an SCS to be high quality.

Finding 16: *There is evidence the guidelines promoted high SCS word count, SCS topic coverage, and SIP word count over density and compartmentalization.* In this section, we applied our metrics to several SCSs described in the guidelines. We found that the retrospective SMTP SCS did not feature a substantial increase in density compared to other metrics, and that it remained low compared to other SCSs. The VRRP example was above the mean for every metric, but its density and compartmentalization were not as high compared to the other metrics. This may have been a factor in density continuing to decrease after the guidelines or in compartmentalization not featuring a strong positive trend over time. We found that even though it had a high SCS and SIP word count, the SMTP revision in RFC 2821 had low topic coverage, which may have been due to low density and compartmentalization. The ideal SCSs put forth by the guidelines had higher SCS word count, SCS topic coverage, and SIP word count than average, but their density and compartmentalization varied more.

7.3 Known Vulnerable RFC

Our final case study examines RFC 8342, *Network Management Datastore Architecture (NMDA)*[6]. A recently published paper showed that this standard enabled a denial of service attack by failing to specify which entity should clear inactive or timed out flow rules from the controller's configuration datastore [13]. In its brief, 115 word-long "Security Considerations" section , the standard incorrectly asserted the datastore architecture it defined had no security impacts. Its SCS did not mention denial-of-service. The RFC's SCS word count was in the 51st percentile, its SIP word count was in the 61st percentile, compartmentalization was in the 41st percentile, density was in the 46th percentile, and SCS topic coverage was 0.0. The SCS word count, SIP word count, and topic coverage are lower than the positive SCS examples provided by the guidelines, but most are close to the average for the entire population of RFCs. The expected metrics for an RFC must be known to identify whether its actual metrics fall short. Because so many RFCs do not require serious discussion of security, not all can be held to the same standard. More context is required to automatically detect whether a particular RFC lacks in security discussion that it requires, which is a source of future work in this area.

Table 2: Values and percentiles of the metrics for RFCs related to the examples given in the SCS Guidelines.

Standard	SCS Word Count	SCS Topic Coverage	SIP Word Count	Compartmentalization	Density
SMTP Original	- / -	- / -	768 / 63.8%	- / -	- / -
SMTP Revision	1,256 / 96.4%	0.1 / 53.2%	4,167 / 95.5%	0.150 / 37.7%	0.497 / 9.3%
SMTP Retrospective	2,638 / 99.4%	0.6 / 98.5%	5,499 / 97.4%	0.307 / 61.6%	0.606 / 15.7%
VRRP	416 / 81.9%	0.2 / 72.3%	1,639 / 81.6%	0.250 / 54.2%	0.983 / 58.5%

8 DISCUSSION

In this section, we answer questions raised by our investigation and address potential concerns the reader may have with our methods.

Should our metrics be treated normatively? We do not prescribe criteria for RFCs, and our metrics should not necessarily be treated as normative. Some standards describe protocols with fewer security concerns than most, and so do not require as much text to fully discuss them. About 50% of draft standards did not experience a lift in metric values throughout the draft process. This trend was not more noticeable in non-accepted drafts than accepted ones. A deeper, more qualitative investigation is required to verify if standards are unfairly rejected due to expectations by the IETF for RFCs with a particular length SCS and amount of security content despite mitigating circumstances.

An interesting example of why our metrics are descriptive, not prescriptive, is compartmentalization. It is not clear whether a high compartmentalization is generally good or bad. One view is that highly compartmentalized security discussion results in a document where security discussion is consolidated in the relevant section, and as such it is easy to quickly find and read all of the security impacts of the RFC. A countering view is that low compartmentalization is to be expected, and arguably desirable, because authors are discussing security in the context of features they impact. This may reduce the likelihood of readers missing important security information of functions because they were moved to the SCS. Compartmentalization tends to be low, so there is evidence that this latter position is the current norm.

Is the security culture of RFC authors changing? We found evidence that security content in RFCs is increasing. Assuming this is due to increasing expectations by the publisher, we claim that the security culture of RFC-publishing organizations is improving. Our approach has limits because we are trying to infer security culture and practices from published artifacts. We do not conduct interviews with members of the IAB, IETF, and IRTF to probe their perceptions, and doing so would only provide insight on the current culture — not where it evolved from.

Are authors of standards taking security seriously enough? Roughly half of authors add substantial security discussion towards the final drafts of their standard. This raises the question of whether the engineers are taking security seriously in the early development of the protocol. Drafts convey incomplete protocols being developed, and authors cannot be faulted for not considering security issues for protocol features that have not been developed yet. As such, we do not claim that authors are failing to take security seriously because security discussion is added towards its final drafts.

Is security content truly increasing? Our metrics have provided evidence that the security discussion in RFCs is growing with time. However, it is possible that information is being repeated in the body and the SCS or that irrelevant security discussion is being included in the SCS to meet culturally normative standards. These aspects are not measured with our methods because they require parsing and reasoning about semantics at a fine level of granularity. This requires a more sophisticated analysis which we leave to future work.

Do RFCs produce more trustworthy standards? This work focuses on the effects that security considerations (and the mandate and guidelines) have had on standards *text*, but the ultimate goal is more trustworthy products. Unfortunately, investigating whether higher-quality SCSs result in more secure products is an interesting question that would be very difficult (if not impossible) to satisfactorily evaluate. Of course, any such analysis would face the issue that demonstrating security is much harder than demonstrating its absence. Important metrics to evaluate would be rate of design vulnerabilities, implementation vulnerabilities, and frequency of configuration errors. Such metrics are difficult to obtain, especially on a per-standard basis. These metrics would of course be confounded by the quality of implementations (independent of the standard in question).

9 RECOMMENDATIONS

In this section, we highlight recommendations and lessons learned for future standards writers and standards bodies.

Recommendation 1: *Standards bodies should mandate security considerations sections.* To the best of our knowledge, no other standards body mandates content similar to a security considerations section. The lack of such sections has had a demonstrable negative impact on the security of deployed systems as well as the ability of analysts and researchers to fully understand the security implications of standards. For example, in recent work analyzing the security of the new 5G authentication protocol [5, 9], authors claimed as a substantial contribution that — after reading hundreds of pages of standards documents — they were actually able to succinctly describe the *apparent* security goals of the 5G AKA cryptographic authentication protocol. The lack of explicit identification of security goals for such an important, security-sensitive protocol is troubling. In contrast, our work demonstrated that mandates and guidelines corresponded with substantial improvements in security content in RFCs. Thus, we strongly recommend that standards bodies that create standards for computing mandate security considerations for all future standards. These bodies include the IEEE, the ITU, the 3GPP, as well as industry consortia like CableLabs.

Recommendation 2: *Individual standard authors should voluntarily include discrete security considerations sections in their documents.* While it is clear from our analysis on RFCs that even short mandates and guidelines can have a significant effect, we note that a mandate is not necessary for standards writers to begin to create separate security considerations sections. In fact, as we noted in Section 5, the mandate was established after a number of RFCs voluntarily

began including SCSs. Given that organizations may not quickly move to establish their own mandates, we recommend that authors begin incorporating security considerations sections regardless of the existence of a mandate.

Recommendation 3: *SCS Guidelines should be periodically updated to reflect advances in security knowledge.* In Section 5 we noted that for our topic coverage analysis we augmented the list of recommended topics in the guidelines to include modern alternatives (e.g., "HTTPS" instead of "S-HTTP"). While the guidelines do not claim to attempt to exhaustively enumerate all security issues, many current, important issues are not present. This limits the usefulness of the guidelines. Some of the important topics not mentioned are: modern side channel attacks (e.g., Spectre and Meltdown), key reinstallation attacks [58], reflected and amplified DoS attack techniques, network middleboxes, DNS hijacking, and multi-factor authentication.

10 RELATED WORK

Since we communicate in natural language, there is a wealth of text like RFCs from which security researchers extract information. Researchers have used mobile app descriptions to identify when an application requests more permissions than necessary [40].Descriptions have also been compared against behavior to detect misleading descriptions or malicious software [24]. Researchers disambiguated privacy policies with topic modeling [53], and summarized them with deep learning [26] and data mining [62].

Analyzing natural language software engineering artifacts has also provided information on the process by which software is developed. Morrison et al. applied text mining to such artifacts to identify if security practices are followed by development teams [34]. Machine learning has helped extract bug reports and feature requests [31] and identify users' rationale for their feedback [30]. Researchers have also mined social media for software feedback [25] and user requirements [28].

RFCs are similar to software engineering requirements because they specify the requirements which must be met by implementations of protocols. A wealth of research has been conducted in automatically identifying ambiguity in software [15, 38, 61], and regulatory [32] requirements, including detecting domain-specific ambiguities [14] and ambiguities arising in multilingual environments [11]. In addition to ambiguity, other metrics have been used to assess the quality of requirements. Quality-assessment tools have identified linguistic defects [22], incompleteness or inconsistency [37], template-conformance [4], and errors in use cases [56]. Singh et al. applied similar techniques to natural-language reviews of these requirements [51].

We use machine learning to classify security informative paragraphs to measure security discussion in RFCs, but predicting vulnerable code and software components has been another prominent application of machine learning. Researchers have used static metrics such as static analysis alerts [21] and software metrics [49, 52] to predict vulnerable software components. These static metrics have been used to detect risky applications on Android [44], and have detected vulnerabilities in Windows Vista with high precision but low recall [63]. Researchers have also used dynamic metrics [50] such as non-security failure alerts [19] to detect vulnerable components. Project metrics have been shown to have strong predictive power in identifying vulnerable code [33].

Researchers have used text analysis to identify security-relevant information in natural language text, but not on network protocol standards like RFCs. Examples are identifying bug reports with security impacts [20] and temporal constraints in API usage [39]. NLP has helped identify security content in software requirements to extract security policies [60], security requirements [12, 36], and mandatory log events [29]. Our research is the first to apply text analysis to network protocol standards to quantify security discussion.

11 CONCLUSION

We investigated the impacts the instructions to RFC authors by RFC 1543 & 3552 had on security discussion in RFCs. Because RFCs are unstructured natural language documents, we used text analysis and machine learning techniques to develop metrics that quantify how they discuss security. We used the "Security Considerations" section of RFCs and a security-informative paragraph classifier to create six metrics that describe how much security discussion is in an RFC, what it discusses, and where it takes place. In addition to the several case studies we conducted, we found that even a simple mandate to include "Security Considerations" sections had profound positive effects on how authors discussed security throughout the entire RFC, increasing both the amount and quality of security discussion. Our work shows that even minimal guidance can correspond to significant improvements in security-relevant outcomes.

ACKNOWLEDGMENTS

We would like to thank Sarah Reaves for her insights on the statistical analysis. We would also like to thank our anonymous reviewers for their helpful comments. This material is based upon work supported by the National Science Foundation under grant numbers CNS-1849994 and CNS-1513690. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] 2019. Internet Engineering Task Force. https://www.ietf.org/.
[2] 2019. RFC Editor. https://www.rfc-editor.org.
[3] Y. Acar, C. Stransky, D. Wermke, C. Weir, M. L. Mazurek, and S. Fahl. 2017. Developers Need Support, Too: A Survey of Security Advice for Software Developers. In *2017 IEEE Cybersecurity Development (SecDev)*. 22–26.
[4] Chetan Arora, Mehrdad Sabetzadeh, Lionel C. Briand, and Frank Zimmer. 2015. Automated Checking of Conformance to Requirements Templates Using Natural Language Processing. *IEEE Transactions on Software Engineering* 41 (2015), 944–968.
[5] David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, and Vincent Stettler. 2018. A Formal Analysis of 5G Authentication. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18)*. ACM, New York, NY, USA, 1383–1396. https://doi.org/10.1145/3243734.3243846
[6] M. Bjorklund, J. Schoenwaelder, P. Shafer, K. Watsen, and R. Wilton. 2018. *Network Management Datastore Architecture (NMDA).* RFC 8342. RFC Editor.
[7] D Ceccarelli and Y Lee. 2018. *Framework for Abstraction and Control of TE Networks (ACTN).* RFC 8453. RFC Editor.
[8] Taejoong Chung, Roland van Rijswijk-Deij, Balakrishnan Chandrasekaran, David Choffnes, Dave Levin, Bruce M Maggs, Alan Mislove, and Christo Wilson. 2017. A Longitudinal, End-to-End View of the DNSSEC Ecosystem. In *26th USENIX Security Symposium*. 1307–1322.
[9] Cas Cremers and Martin Dehnel-Wild. 2019. Component-Based Formal Analysis of 5G-AKA: Channel Assumptions and Session Confusion. In *Proceedings 2019*

Network and Distributed System Security Symposium. Internet Society, San Diego, CA. https://doi.org/10.14722/ndss.2019.23394
[10] Steve Crocker. 1969. *Host Software.* RFC 1. RFC Editor.
[11] Breno Dantas Cruz, Bargav Jayaraman, Anurag Dwarakanath, and Collin McMillan. 2017. Detecting Vague Words & Phrases in Requirements Documents in a Multilingual Environment. *2017 IEEE 25th International Requirements Engineering Conference (RE)* (2017), 233–242.
[12] Alex Dekhtyar and Vivian Fong. 2017. RE Data Challenge: Requirements Identification with Word2Vec and TensorFlow. *2017 IEEE 25th International Requirements Engineering Conference (RE)* (2017), 484–489.
[13] Vaibhav Hemant Dixit, Adam Doupé, Yan Shoshitaishvili, Ziming Zhao, and Gail-Joon Ahn. 2018. AIM-SDN: Attacking Information Mismanagement in SDN-datastores. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 664–676.
[14] Alessio Ferrari, Beatrice Donati, and Stefania Gnesi. 2017. Detecting Domain-Specific Ambiguities: An NLP Approach Based on Wikipedia Crawling and Word Embeddings. *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)* (2017), 393–399.
[15] Alessio Ferrari, Giuseppe Lipari, Stefania Gnesi, and Giorgio Oronzo Spagnolo. 2014. Pragmatic ambiguity detection in natural language requirements. *2014 IEEE 1st International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)* (2014), 1–8.
[16] H Flanagan and S Ginoza. 2014. *RFC Style Guide.* RFC 7322. RFC Editor. https://www.rfc-editor.org/rfc/rfc7322.txt
[17] Deen Freelon. 2010. Intercoder Reliability Calculation as a Web Service.
[18] Deen Freelon. 2013. ReCal OIR : Ordinal , Interval , and Ratio Intercoder Reliability as a Web Service.
[19] Michael Gegick, Pete Rotella, and Laurie A. Williams. 2009. Toward Non-security Failures as a Predictor of Security Faults and Failures. In *ESSoS*.
[20] Michael Gegick, Pete Rotella, and Tao Xie. 2010. Identifying security bug reports via text mining: An industrial case study. *2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010)* (2010), 11–20.
[21] Michael Gegick and Laurie L. Williams. 2007. Toward the Use of Automated Static Analysis Alerts for Early Identification of Vulnerability- and Attack-prone Components. *Second International Conference on Internet Monitoring and Protection (ICIMP 2007)* (2007), 18–18.
[22] Stefania Gnesi, Giuseppe Lami, and Gianluca Trentanni. 2005. An automatic tool for the analysis of natural language requirements. *Comput. Syst. Sci. Eng.* 20 (2005).
[23] Sharon Goldberg. 2014. Why is it taking so long to secure internet routing? *Commun. ACM* 57, 10 (2014), 56–63.
[24] Alessandra Gorla, Ilaria Tavecchia, Florian Gross, and Andreas Zeller. 2014. Checking app behavior against app descriptions. In *Proceedings of the 36th International Conference on Software Engineering*. ACM, 1025–1035.
[25] Emitza Guzman, Rana Mohammed A. Alkadhi, and Norbert Seyff. 2016. A Needle in a Haystack: What Do Twitter Users Say about Software? *2016 IEEE 24th International Requirements Engineering Conference (RE)* (2016), 96–105.
[26] Hamza Harkous, Kassem Fawaz, Rémi Lebret, Florian Schaub, Kang G. Shin, and Karl Aberer. 2018. Polisis: Automated Analysis and Presentation of Privacy Policies Using Deep Learning. In *USENIX Security Symposium*.
[27] Ghaith Husari, Ehab Al-Shaer, Mohiuddin Ahmed, Bill Chu, and Xi Niu. 2017. TTPDrill: Automatic and Accurate Extraction of Threat Actions from Unstructured Text of CTI Sources. In *Proceedings of the 33rd Annual Computer Security Applications Conference (ACSAC 2017)*. ACM, New York, NY, USA, 103–115.
[28] Georgi M. Kanchev, Pradeep K. Murukannaiah, Amit K. Chopra, and Peter Sawyer. 2017. Canary: Extracting Requirements-Related Information from Online Discussions. *2017 IEEE 25th International Requirements Engineering Conference (RE)* (2017), 31–40.
[29] Jason King, Rahul Pandita, and Laurie A. Williams. 2015. Enabling forensics by proposing heuristics to identify mandatory log events. In *HotSoS*.
[30] Zijad Kurtanovic and Walid Maalej. 2017. Mining User Rationale from Software Reviews. *2017 IEEE 25th International Requirements Engineering Conference (RE)* (2017), 61–70.
[31] Walid Maalej and Hadeer Nabil. 2015. Bug report, feature request, or simply praise? On automatically classifying app reviews. *2015 IEEE 23rd International Requirements Engineering Conference (RE)* (2015), 116–125.
[32] Aaron K. Massey, Richard L. Rutledge, Annie I. Antón, and Peter P. Swire. 2014. Identifying and classifying ambiguity for regulatory requirements. *2014 IEEE 22nd International Requirements Engineering Conference (RE)* (2014), 83–92.
[33] Nadia Patricia Da Silva Medeiros, Naghmeh Ivaki, Pedro Costa, and Marco Vieira. 2017. Software Metrics as Indicators of Security Vulnerabilities. *IEEE 28th International Symposium on Software Reliability Engineering* (2017), 216–227.
[34] Patrick Morrison, Benjamin A H Smith, and Laurie A. Williams. 2017. Measuring Security Practice Use: A Case Study at IBM. *2017 IEEE/ACM 5th International Workshop on Conducting Empirical Studies in Industry (CESI)* (2017), 16–22.
[35] Mavuto Mukaka. 2012. Statistics corner: A guide to appropriate use of correlation coefficient in medical research. *Malawi medical journal : the journal of Medical Association of Malawi* 24 3 (2012), 69–71.
[36] Nuthan Munaiah, Andrew Meneely, and Pradeep K. Murukannaiah. 2017. A Domain-Independent Model for Identifying Security Requirements. *2017 IEEE 25th International Requirements Engineering Conference (RE)* (2017), 506–511.
[37] Tuong Huan Nguyen, John C. Grundy, and Mohamed Almorsy. 2014. GUITAR: An ontology-based automated requirements analysis tool. *2014 IEEE 22nd International Requirements Engineering Conference (RE)* (2014), 315–316.
[38] Olga Ormandjieva, Ishrar Hussain, and Leila Kosseim. 2007. Toward a text classification system for the quality assessment of software requirements written in natural language. In *SOQUA*.
[39] Rahul Pandita, Kunal Taneja, Laurie A. Williams, and Teresa Tung. 2016. ICON: Inferring Temporal Constraints from Natural Language API Descriptions. *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)* (2016), 378–388.
[40] Rahul Pandita, Xusheng Xiao, Wei Yang, William Enck, and Tao Xie. 2013. WHYPER: Towards Automating Risk Assessment of Mobile Applications.. In *USENIX Security Symposium*, Vol. 2013.
[41] D. Piper. 1998. *The Internet IP Security Domain of Interpretation for ISAKMP.* RFC 2407. RFC Editor.
[42] J Postel. 1993. *Instructions to RFC Authors.* RFC 1543. RFC Editor. https://www.rfc-editor.org/rfc/rfc1543.txt
[43] J Postel and J Reynolds. 1997. *Instructions to RFC Authors.* RFC 2223. RFC Editor.
[44] Akond Rahman, Priysha Pradhan, Asif Partho, and Laurie A. Williams. 2017. Predicting Android Application Security and Privacy Risk with Static Code Metrics. *2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft)* (2017), 149–153.
[45] A Rajarman and J Ullman. 2011. *"Mining of Massive Datasets".* 1–17 pages.
[46] E Rescorla. 2033. *Guidelines for Writing RFC Text on Security Considerations.* RFC 3552. RFC Editor. https://www.rfc-editor.org/rfc/rfc3552.txt
[47] J Reynolds and J Postel. 1990. *Assigned Numbers.* RFC 1060. RFC Editor. https://www.rfc-editor.org/rfc/rfc1060.txt
[48] Shlomo S Sawilowsky. 2009. New effect size rules of thumb. (2009).
[49] Yonghee Shin, Andrew Meneely, Laurie A. Williams, and Jason A. Osborne. 2011. Evaluating Complexity, Code Churn, and Developer Activity Metrics as Indicators of Software Vulnerabilities. *IEEE Transactions on Software Engineering* 37 (2011), 772–787.
[50] Yonghee Shin and Laurie A. Williams. 2011. An initial study on the use of execution complexity metrics as indicators of software vulnerabilities. In *SESS@ICSE*.
[51] Maninder Singh. 2018. Automated Validation of Requirement Reviews: A Machine Learning Approach. *2018 IEEE 26th International Requirements Engineering Conference (RE)* (2018), 460–465.
[52] Ben H. Smith and Laurie A. Williams. 2011. Using SQL Hotspots in a Prioritization Heuristic for Detecting All Types of Web Application Vulnerabilities. *2011 Fourth IEEE International Conference on Software Testing, Verification and Validation* (2011), 220–229.
[53] John W. Stamey and Ryan A. Rossi. 2009. Automatically identifying relations in privacy policies. In *SIGDOC*.
[54] M. Stapp, T. Lemon, and A. Gustafsson. 2006. *A DNS Resource Record (RR) for Encoding Dynamic Host Configuration Protocol (DHCP) Information (DHCID RR).* RFC 4701. RFC Editor.
[55] M. Stiemerling, J. Quittek, and T. Taylor. 2005. *Middlebox Communications (MIDCOM) Protocol Semantics.* RFC 3989. RFC Editor.
[56] Saurabh Tiwari and Mayank Laddha. 2017. UCAnalyzer: A Tool to Analyze Use Case Textual Descriptions. *2017 IEEE 25th International Requirements Engineering Conference (RE)* (2017), 448–449.
[57] Alexander van den Berghe, Koen Yskout, Riccardo Scandariato, and Wouter Joosen. 2018. A Lingua Franca for Security by Design. *2018 IEEE Cybersecurity Development (SecDev)* (2018), 69–76.
[58] Mathy Vanhoef and Frank Piessens. 2017. Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*. ACM, New York, NY, USA, 1313–1328. https://doi.org/10.1145/3133956.3134027
[59] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*.
[60] Xusheng Xiao, Amit M. Paradkar, Suresh Thummalapenta, and Tao Xie. 2012. Automated extraction of security policies from natural-language software documents. In *SIGSOFT FSE*.
[61] Hui Yang, Anne N. De Roeck, Vincenzo Gervasi, Alistair Willis, and Bashar Nuseibeh. 2011. Analysing anaphoric ambiguity in natural language requirements. *Requirements Engineering* 16 (2011), 163–189.
[62] Razieh Nokhbeh Zaeem, Rachel L. German, and K. Suzanne Barber. 2018. PrivacyCheck: Automatic Summarization of Privacy Policies Using Data Mining. *ACM Trans. Internet Techn.* 18 (2018), 53:1–53:18.
[63] Thomas Zimmermann, Nachiappan Nagappan, and Laurie A. Williams. 2010. Searching for a Needle in a Haystack: Predicting Security Vulnerabilities for Windows Vista. *2010 Third International Conference on Software Testing, Verification and Validation* (2010), 421–428.