A Password-Policy Taxonomy: Usability Research in Support of Cyber-Security

Drafted by Kevin Killourhy

October 5, 2010

Abstract

Password policies—documents which regulate how users must create, manage, and change their passwords—can have complex and unforeseen consequences on organizational security. Since these policies regulate user behavior, users must be clear as to what is expected of them. Unfortunately, current policies are written in language that is often ambiguous. Vagary and discrepancies between the policies that govern users in their workplace and their personal life make it likely that policy requirements will be bypassed or ignored, thereby weakening security.

This work arose from a need to deal with ambiguities in current password policies. These ambiguities not only prevent researchers from comparing and contrasting policy statements, they also could cause users to misinterpret what is expected of them. To tackle ambiguity, we developed a formal language for stating what behavior is and is not allowed when creating, managing, and changing passwords. This formal language lends itself to policy analysis and visualization.

We collected a corpus of 41 password policies, translated them into the formal language, and analyzed them. Of the 41 policies we considered, no two were exactly the same. A user governed by multiple policies (e.g., one at work and one at the bank) has little chance of finding consistent policies. While preliminary, we feel that this approach is promising. Having these clear, unambiguous policy statements enables us to explore password policies in much greater detail, discuss the relative merits of different statements, and develop a sound set of best practices.

1 Introduction

As part of the Research and Development thrust of the recent Comprehensive National Cyber-Security Initiative, we undertook an exploration of the relationship between usability and security in password policies. In this domain, as in many areas of computer security, we find a mixture of security concerns and usability concerns. Adams and Sasse [1999] observed that when users feel that password rules are preventing them from accomplishing their work, they will work around them. For instance, if files cannot be easily shared among account owners, users will share an account, violating the password policy restriction on sharing. For a policy to ensure security, it must meet both theoretical security criteria and usability criteria.

Password policies are documents that specify how passwords must be created, stored, changed, and managed. Depending on the security concerns of the policy writer, these documents regulate everything from length and lifetime, to when and how a user can transmit passwords. For example, SANS Institute [2006] has shared a template password policy for use by policy writers. The template requires that user-account passwords be changed every six months, and it offers guidance on what constitutes a strong password (e.g., "at least fifteen alphanumeric characters"). The template

requires that users not reveal their password, not check "remember password" boxes in applications, and not store a written password in their office.

These policy statements are written with security concerns in mind. The restriction on users writing down their password is meant to protect the password, but it requires user "buy in" if that protection is to exist. Consequently, the role of the user cannot be discounted when developing a policy. Since some security requirements are all but unenforceable, security depends on a policy that can explain what is expected of a user and convince the user to comply.

Unfortunately, the security concerns that prompt the statements in a password policy are rarely understood by users. Typical policy requirements include a password with an 8 character minimum length (or 15 in the example above), comprised of mixed-case letters, numbers, and special characters, and which does not contain dictionary words or personal information. Cheswick [2010] has called these "eye of newt" policy statements in part because they sound like a magic formula to a user.

A typical user might be governed by multiple policies both at work and at home (e.g., to access corporate email servers, personal financial information, medical records, and e-commerce sites). Ambiguities in these policies, discrepancies between them, and the sheer number of different policies may cause confusion. As a result of this cognitive burden, users may choose weak passwords, write them down, or violate policies in other ways. As policy violation becomes an accepted matter of course, the security goals of the policy are not met. Consequently, overall security may be weakened.

2 Problem and approach

In a preliminary survey of password policies, we discovered an unexpected problem both for our research effort and for users: *ambiguity*. We were surprised at the frequency with which there was disagreement over what a policy meant. If we had difficulty agreeing, would users who are expected to follow these policies also have different interpretations? An example of the difficulty can be seen by examining a statement in the password policy of one Federal agency:

> *Passwords contain a combination of letters, numbers, and at least one special character.*

Different people have interpreted this statement to mean each of the following:

1. Users must create passwords with at least one letter, at least one number, and at least one special character (i.e., defining *a combination* to mean one or more of each).

2. Users must create passwords with at least one special character (i.e., defining *a combination* to mean zero or more of each, except special characters since *at least one* is explicitly required).

3. Users must create passwords with at least two letters, two numbers, and one special character (i.e., because *letters* and *numbers* are both plural).

All interpretations are reasonable, and as a result a reader cannot be sure whether his or her interpretation is correct.

Ambiguity is both a usability issue and a security issue. Users faced with such a policy cannot know which passwords are in compliance and which are not. Users who take the time to create a memorable, strong password only to have it rejected as non-compliant are unlikely to try as hard the second time. Researchers cannot accurately assess security when there is disagreement over the security properties. For instance, each of the three interpretations corresponds to a password space of different size; the strength of the password depends on the size of this space.

In this work, our aim is to resolve the problem of password ambiguity by developing methods and tools for studying and clarifying ambiguous policy statements. Only after policy statements have been refined and expressed clearly can we compare statements across policies, discuss best practices, and investigate the security and usability properties of different policy statements.

To that end, we created a policy-statement taxonomy, for organizing policy statements, discussing their properties, and comparing policies to each another. We collected a corpus of password policies, and studied them to understand the variety of policy rules and regulations that are made by employers, financial firms, and others who write password policies. Based on these insights, we developed a grammar for a formal language describing various kinds of policy statements. We translated the policies into this formal language, enabling us to conduct statistical analysis and visualize policy differences.

3 Background

Our effort exists within 30 years of research into password usability and password policies. Research studies have periodically tried to characterize the passwords that people choose [Morris and Thompson, 1979, Klein, 1990, Wu, 1999, Florêncio and Herley, 2007, Dell'Amico et al., 2010]. These works usually rely on a list of passwords obtained by a system-administrator, observed through a toolbar, or revealed through a security compromise. They provide empirically grounded insight into the actual passwords that people choose. While these studies do not consider password policies directly, they are related to our research on policy since users' password choices are governed in part by policy requirements. We hope that the present work may inform future studies of user behavior when choosing passwords.

Some researchers have attempted to examine the connection between password policy and user behavior already. Mannan and Oorschot [2007] surveyed users of online banks regarding their understanding of bank's security requirements. They found a disconnect between bank guidelines and user practice. For instance, bank password-change recommendations were on the order of a few months, yet the majority of users do not change their passwords within a year. Furnell [2007] surveyed 10 popular websites, examining the password-creation guidelines, the enforcement of password-composition restrictions, and the reset policy. Such guidance is an important facet in any password framework: convincing users of the importance of following password-policy regulations. Inglesant and Sasse [2010] surveyed 32 staff members at a research university and a financial-services organization about their password usage. They found that password policies which ignore human factors may result in unexpectedly poor security (e.g., shared and written-down passwords).

Many authors have observed the increasing burden of password management on users. Summers and Bosworth [2004] argue for user guidance and enforcement to prevent users from choosing easily guessable passwords. Spafford [2006] argued that the best practices shared by many modern password policies are actually artifacts based on out-of-date risk assessments. For instance, he traces requirements that passwords be changed monthly to the computing power of Department of Defense machines in the 1970s. He advocates updating these out-of-date best practices to modern, *sound* practices based on individual risk assessments. Farrell [2008] introspectively determined that he used over 61 passwords, 15 of which were committed to memory. He argues that policy writers must acknowledge the increasing burden of password management on their users.

Some research has attempted to make writing security policies easier. Xu et al. [2008] present a visualization of an access-control policies. Johnson et al. [2010] provide an extensive list of guidelines for anyone architecting a system for writing security and privacy policies.

A few researchers have focused specifically on writing password policies. Shay et al. [2007] present a language for expressing a password-policy scenario in a formal language. Using simulation,

they are able to estimating a measure of system harm resulting from the given scenario. Parkin et al. [2009] developed an ontological framework for reasoning about the security and usability costs of different policy decisions. These earlier works show the potential for password policies written in formal language. That effort partners well with the aim of the current work to bridge the gap between ambiguous, informal policy statements and clear, explicit, formal policy statements.

Perhaps the closest research efforts were those of Bonneau and Preibusch [2010] and Florêncio and Herley [2010]. Bonneau and Preibusch [2010] surveyed the empirical password policies of 150 websites. By creating accounts and systematically changing the password, they were able to infer the enforced lengths and complexity requirements. Florêncio and Herley [2010] surveyed length and complexity requirements for 75 websites according to their password policies. Their primary finding was that length and complexity requirements are not explained by the size of the assets protected but by the dependence on advertising dollars. In other words, sites that depend on users for revenue have weaker, more accommodating policies.

While our research overlaps that of Bonneau and Preibusch [2010] and Florêncio and Herley [2010]—including some of the same sites in our survey—the focus of our work differs in important ways. Foremost, our interest in capturing the diversity and ambiguities in password policies results in a different research approach. Our investigation moves beyond length and complexity requirements and into the numerous other requirements and guidance that password policies provide.

4 Creating a taxonomy of policy statements

Unlike length and complexity requirements, not all of the statements in a password policy can be easily captured by a vector of features. In this work, a different approach was taken. We tried to capture the wide variety of policy statements that govern user behavior in a flexible, extensible way, rather than focus on a few key features of policies.

Instead of a feature vector, we translate a password policy into a set of statements written with a common, unambiguous grammar. This grammar defines a taxonomic structure for organizing password-policy statements into similar groups and dissimilar classes. With this taxonomy, we can compare policies, visualize them, and quantify disagreement between them.

We create the taxonomy by first collecting a corpus of password policies. Then, we develop the grammar whereby the regulations of the password policies can be expressed as explicit expectations on user behavior when creating, changing, storing, or communicating passwords. Next, we use this grammar to translate the 41 password policies into a formal, well-defined language. Finally, we explore how policies, once represented in this language, lend themselves to comparison, visualization, and further analysis. Each of these steps in our method will be described in the following sections.

4.1 Collecting a policy corpus

We employed two methods for collecting password policies for study. We collected workplace policies by searching Federal Government websites. We collected personal-site policies by searching popular websites. The two approaches enabled us to collect 41 policies: 22 concerning workplace passwords, and 19 concerning personal-site passwords.

In an effort to collect a variety of workplace policies, we visited each of the Federal Departments and Agencies listed at `www.usa.gov`. On each site, we attempted to find the password policy, primarily by searching for "password policy" and "passphrase policy" in the site-wide search box, but also by navigating through links. To qualify, the policy document itself had to be obtained. In cases where only a memorandum or article describing some details of the policy were returned, we did not include the document in our corpus. Such partial descriptions may not include all

the details of the policy. Policies were also checked to ensure that they applied to employees of the organization (rather than outside users of a service provided by the organization). Policies judged to govern workplace behavior concerning passwords were added to the corpus. From the sites visited, we obtained 22 policies.

To collect a comparatively sized set of policies from sites that a user might visit in their personal life, we began with the corpus of online password policies assembled by Florêncio and Herley [2010]. Their paper explains the collection methodology in detail, but one of the aims was a representative collection. In their paper, they examined 77 policies in total and provided links to 56 of them. We visited all 56 links. If a link is broken or we cannot find a minimal set of policy rules, we excluded the policy from our corpus by necessity. For a policy to be in our corpus it must have, at a minimum, a requirement or recommendation about password length or a statement about which character sets are acceptable.

Following this procedure, we added 15 sites to our corpus. Because our corpus contained a paucity of financial-service websites, and because we felt that such policies would be important in our corpus (as they are personal sites that are still likely to have detailed password policies), we visited the top business and financial services websites, ranked by US traffic, according to `www.alexa.com` (in August 2010). We employed the same strategy to find a password policy governing users of these site as above. Following this procedure, we added four sites, for a total of 19 personal site policies and 41 policies overall.

4.2 Developing a taxonomic grammar

While reading and becoming familiar with the regulations and guidelines of the policies in the corpus, we made several attempts at organizing the policies and policy statements into rules. Our final product was a formal language in which policies were represented as a set of statements formed from a subset of English. However, in the process of arriving at this approach, we attempted two other approaches: feature vectors and qualitative coding.

Our first attempt at organizing password policies was to encode them as feature vectors. In casual conversation, IT staff members refer to password policies by a pair of numbers. For instance, switching from an 8/90 policy to a 12/90 policy means increasing the minimum length from 8 characters to 12, and keeping the 90 day lifetime. Earlier research, such as that by Florêncio and Herley [2010], effectively encoding password length, composition requirements, and lifetime in this manner. However, because our interest was in capturing other aspects of policy behavior (e.g., what word and letter combinations are forbidden, and whether users can use password vaults), this approach quickly became unwieldy.

Our second attempt at organizing passwords was to analyze password policies in the style of a social scientist trying to organize research data. Using a software package for qualitative-data-analysis, a researcher tagged sentences and sections in a password policy with a topic *code*. For instance, a line insisting that strong passwords have a minimum of 15 characters would be coded "Length," and a line stating that passwords must be alphanumeric would be coded "Character Content." We identified 9 top-level codes, in this way: Length, Lifetime, Lockout, Storage, Use, Character Set, Character Content, Content Features, and Forgotten. The plan was to subdivide each of these into different kinds of statements within each topic area. This approach also became unwieldy, as the distinctions between topic areas was subject for debate.

These failed attempts did build to our final, more successful, approach to studying password policies. Rather than trying to divide up the password policy into topic areas, as in the previous approach, we simply translated each policy statement into clear language that explicitly states what is expected of the user. For instance, "strong passwords are at least 15 characters and alphanumeric"

became two statements: (1) "users must create passwords that are at least 15 characters long" and (2) "users must create passwords where every character is a letter or a number." Note that the user is the subject of the sentence, making it clear that the policy applies to the user (compare "passwords must be stored as cryptographic hashes" which is probably, but not explicitly, regulating the behavior of a system administrator not a user).

After rewriting several policies, we discovered that some sentences were being reused. Re-reading what we had written, we saw that other sentences were saying the same thing but in different ways. By restricting ourselves to a grammar in which a policy statement could only be expressed in one way—and two different statements imposed two different requirements, even if only subtly—the idea of a formal grammar for expressing password policies arose.

The actual grammar of the formal language was specified using an extension of the Backus-Naur Form (EBNF). A parser was implemented as a Perl script using the package `Parse::RecDescent`. A document describing the language and the procedure for translating a policy into the language was written. This document is included as Appendix A, which includes both the EBNF syntax and a description of the semantics.

4.3 Translating policies to the formal language

A single researcher, who also created the language specification, performed the translation of the policies in the corpus.[1] Translations involved two steps. First, the translator identified which sentences, bullets, or policy statements were *within scope*. Then, he translated each statement to one or more statements in the formal language. Each of these steps is described in much greater detail in Appendix A, but the intuition is fairly simple.

Some statements were judged as beyond scope because this work is exploratory. We considered many kinds of policy statement to be too narrow for inclusion in the language at this time. For instance, most organizations forbid group accounts, but a few have detailed policies describing when and where group accounts with a shared password are allowed. Rather than delve into the details of these atypical policies, we considered policies involving group passwords to be beyond scope. If this research matures, one might easily extend the formal language to express group-account regulations.

Statements that were not beyond scope were translated into the language using the instructions laid out in Appendix A. The translated policy documents were checked with a language parser to ensure that the syntax of each statement is correct. In the end, all 41 policies in the corpus were translated into the policy language.

4.4 Analysis and visualization

The translated policy documents enable analysis and visualization that would not be possible with the raw documents. Specifically, since the formal grammar has the property that semantically equivalent statements are syntactically equal, we can analyze statement prevalence and frequency. Further, since the language has a hierarchical characteristic whereby similar statements start the same way, a tree-like visualization of password policies becomes possible.

Analysis with summary statistics

Some summary statistics provide a clear picture of the diversity of password policies. The 41 policies yielded a total of 449 statements in the formal policy language. The policy with the fewest

[1] In Section 5.3, we compare these translations to those of a second translator, but all the other analysis and figures are based on these primary translations.

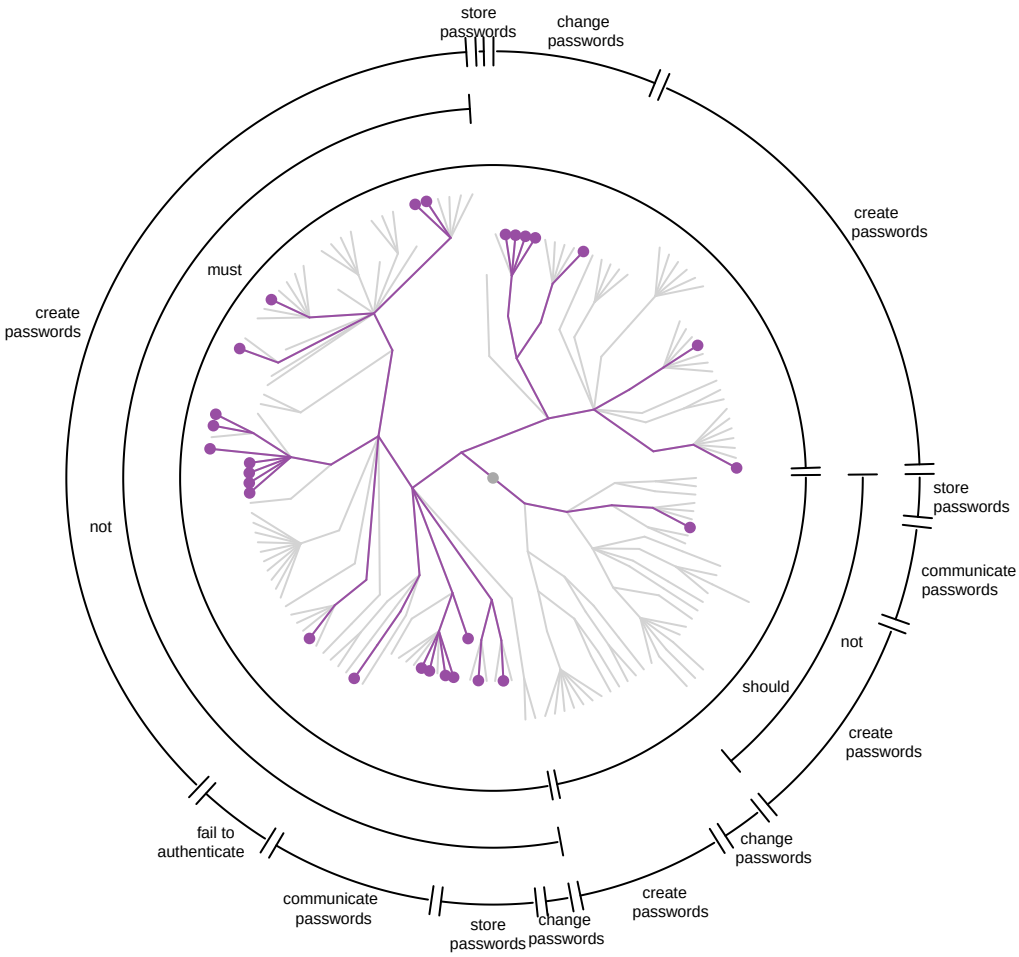


Figure 1: A tree-like visualization of the password policy statements in the corpus. Each path from the center to the outside corresponds to a unique statement of expectation about a user's behavior. The statements have been organized so that those sharing a common prefix inhabit the same region of the space. For instance, the 57 different statements which begin "Users must not create passwords ..." inhabit the region from 7–12 o'clock, while the 22 different statements which begin "Users should not ..." inhabit the region from 3–4 o'clock. The subset of statements made by a particular password policy can be represented as a subtree. One such subtree, comprised of the 28 statements in the NIST password policy, is depicted in purple.

statements had only one (Weather Channel Users: *Users must create passwords with length greater than or equal to 6 characters.* The policy with the most statements had 38 (Brookhaven National Lab). The median number of statements per policy was 9.

We should note that care should be taken in interpreting the number of statements in a policy as a measure of complexity or user burden. Not all statements are equal. Some may impose stricter constraints than others, and some rules may only be burdensome in combination with other rules. These summary statistics provide some preliminary quantitative characterizations of the diversity and complexity of password policies, but they are only preliminary.

Of the 449 total statements, there were 153 unique statements, meaning that there is some overlap across policies. The statement which appeared in the most policies was *Users must create passwords with length greater than or equal to 8 characters.* This statement appeared in 23 policies (56%). The next most prevalent statement appears in only 15 policies (37%): *Users must not communicate passwords to anyone.* Regarding the infrequent policies, 71 of the 153 statements (46%) appeared in only one policy. Additionally, of the 41 policies, no two policies shared the exact same set of statements (0%).

Visualization with policy trees

One powerful advantage of the taxonomy of password statements is a visualization based on the parse tree. Figure 1 shows an example of such a visualization. Each statement can be broken up into phrases based on which grammar rule matched the words. For instance, the policy statement "*Users / must // create passwords / with length / greater than or equal to / 8 characters*" has vertical bars delimiting the phrases. The double vertical bar indicates a blank phrase (contrasting with the phrase "*not*" which can appear instead of the blank).

We can visualize each policy statement as a series of line segments, one per phrase, which radiate out from a central point. If two policies start with the same phrases, they follow the same path. When the two policies use different phrases, the paths diverge. Since a policy is a collection of statements, we can view it as a tree.

Figure 1 shows a tree with 153 paths from the center of the circle to the outside, each corresponding to a different one of the 153 unique statements made in the policies. The darker paths, drawn in purple, correspond to policy statements made by the (now obsolete) NIST password policy. The lighter paths, drawn in grey, correspond to statements that are in other policies and not NIST's. The arcs around the outside of the figure demonstrate that different slices (or wedges) of the circle corresponds to different prefix phrases. The median number of phrases is 6 in a policy, with a minimum of 4 ("*Users / must / not / communicate passwords / except in an emergency*") and a maximum of 7 ("*Users / must // create passwords // in the set of / strings with / at least 2 unique characters*"). Note that, by convention, since all policies start with the phrase "*Users*," we do not count it when counting the number of phrases in a statement.

The policy tree visualizations for the 41 policies in our corpus are included in Appendix B. We believe these visualizations to be a useful measure of policy complexity. Policies with more statements produce denser trees. (Of course, we should repeat our earlier note of caution about equating the complexity or burden of different policy statements.) We also find these visualizations to be useful in comparing policies by placing trees for two policies side-by-side. Since policy statements correspond to the same path relative to the side-by-side centers, it is possible to spot paths that are shared and paths that are not. It is also possible to compare regions of coverage (e.g., which policies have many strict "*must*" regulations and which policies have more suggestive "*should*" guidelines).

Beyond the pictures, we also created a password-policy explorer in the R statistical programming

environment. The policy is displayed as a tree like that in Figure 1, and a user can click on individual nodes in the tree to see the corresponding policy statement or prefix. This interactive aspect to the visualization has made it easy to use the policy pictures to get a broad sense of the similarities and differences between two policies, and then to drill down into individual statements where the policies agree or disagree.

5 Explorations and preliminary experiments

To highlight the potential of this taxonomy of password-policy statements we conducted several exploratory experiments. First, we compared policies that have a known relationship to see how that relationship reflects in the visualization. Second, we compared the workplace policies to the personal-site policies to see whether workplace policies are stricter (as would be expected based on earlier results by Florêncio and Herley [2010]). Finally, to investigate the repeatability of the translation process and the problem of different password-policy interpretation readers, we compared the translations of a second translator to those presented above.

5.1 Comparing Policies: NIST, the Census Bureau, DOC, and the Marines

The password-policy visualizations make it easy to explore and compare different password policies. Figure 2 shows four policies in a 2×2 tiling. The policies on the top row are from NIST and the Census Bureau. The policies on the bottom line are from the Department of Commerce (DoC) and the Marine Corps Enterprise Network (MCEN). Note that NIST and the Census Bureau are both organizations within the Department of Commerce.

Given their department-level connection, we might expect to see some similarities between the NIST and Census policies. We can observe similarities, and also some differences. For instance, between 12 o'clock and 1 o'clock are a set of restrictions on when passwords must be immediately changed. Specifically, the prefix is "*Users must change passwords immediately if*" and there are five different completions, corresponding to the path's split into a five-tined fork. Neither NIST nor Census include the first completion: *sent unencrypted* (i.e., a requirement that passwords be changed immediately if sent in clear text). Both NIST and Census include the next three completions: *found non-compliant*, *directed by management*, and *compromised*. NIST also includes the final completion while Census does not: *shared*. NIST's policy allows passwords to be communicated to others in an emergency, but this statement tries to ensure that shared passwords are changed as soon as possible.

The similarities between the two policies suggests a common source. The two policies do share many of the same statements with the password policy for the Department of Commerce, shown in the lower left of Figure 2. Note that the DoC policy shares the three regulations on immediate password change: *found non-compliant*, *directed by management*, and *compromised*. However, there are some discrepancies between the department-level policy and those of the organizations in the department. For instance, in the 3 o'clock position, both NIST and Census have the same path while DoC has a slightly different one (i.e., NIST and Census have a path that angles downward in the last phrase while DoC has a path that angles slightly upward. This corresponds to an 8 character minimum for NIST and Census and a 12 character minimum for DoC.[2]

In contrast to the NIST, Census, and DoC policies which have dozens of statements, the Marine (MCEN) password policy in the lower right is much more sparse. It is comprised of only 10

[2]During the time our corpus was collected, the Department of Commerce and all the organizations within it were undergoing a move from an 8 character to a 12 character minimum. As a result, the various organizational password policies reflect different stages of the transition.

statements. Note that this policy says nothing about the conditions under which passwords must be immediately changed. Also note, in the 6 o'clock position, the MCEN policy has a rule that is in none of the other three: *Users must not change passwords before 7 days.* The MCEN policy is one of a few that set a limit on how often a user can change his or her password (presumably to prevent them from returning too quickly to an old password). This exploration of the common and the unusual across policies promotes discussion of what statements should be in a policy.

We also found it interesting that, despite their differences, the NIST, Census, and DoC policies do look similar, when compared with a policy from another unrelated organizations in the Federal government. In this regard, the visualization seems useful as it does enable a viewer to notice the resemblance of policies to one another.

5.2 Comparing Policy Groups: Workplace and Personal-Site

In addition to comparisons of individual policies, we can compare sets of policies by comparing the statements which appear in each set. For instance, one might want to compare workplace and personal-site policies to see whether workplace policies tend to be more strict. Florêncio and Herley [2010] found that sites which draw ad-revenue have more lenient policies, and sites which do not need to entice their users to join (i.e., workplaces) are stricter.

A statistical analysis of the number of statements in each policy reveals that the median for workplace policies was 10 statements, and the median for personal-site policies was 7. As mentioned, the number of statements alone is a problematic measure of strictness. Moving beyond summary statistics, Figure 3 visually contrasts the workplace policy tree and the personal-site policy tree. The workplace tree is comprised of the union of all the workplace statements, and the personal-site tree is comprised of the personal-site statements.

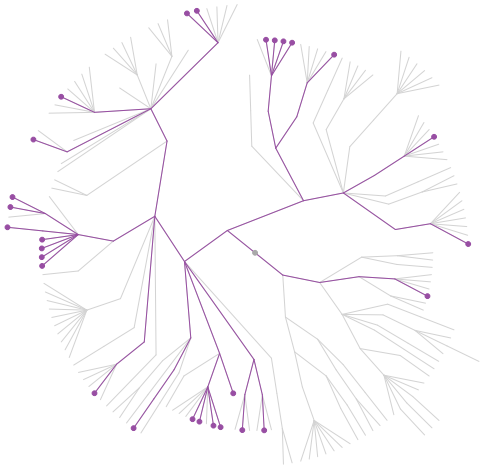
Both trees are dense, suggesting a diversity of policy statements both in the workplace and at personal sites. However, the figure reveals that personal-site policies use more *should* statements (i.e., those begin *Users should*), while workplace policies use more *must* statements (i.e., those beginning *Users must*). The *should* statements are represented by the lower right quarter of each policy-tree visualization, from 3 o'clock to 6 o'clock, and the remainder are *must* statements.

Note that the partitioning of policies into workplace and personal-site is only one possible partitioning out of many. Since the personal-site policies are comprised of sites ranging from banks to sports news, we expect that a subdivision of the personal-site policies would be informative. One might imagine that banks and other financial services companies would have more restrictive policies as security is more of a concern. The formal-language representation and visualization lends itself to such further partitioning, exploration, and analysis.

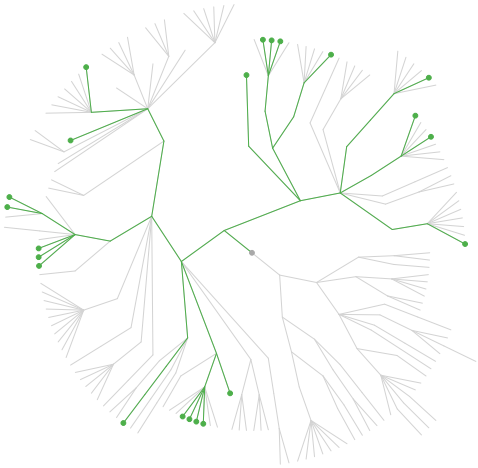
5.3 Comparing Translations: Reliability and Interpretation

Because the analysis above is based on the translations of a single researcher, a natural question is whether translations from another translator would be the same. To answer this question, we designed a small pilot experiment in which a second person translated a subset of policies, and the two translations were compared. In this section, we describe this pilot experiment, the somewhat ambiguous results, and what improvements might be taken in the future.

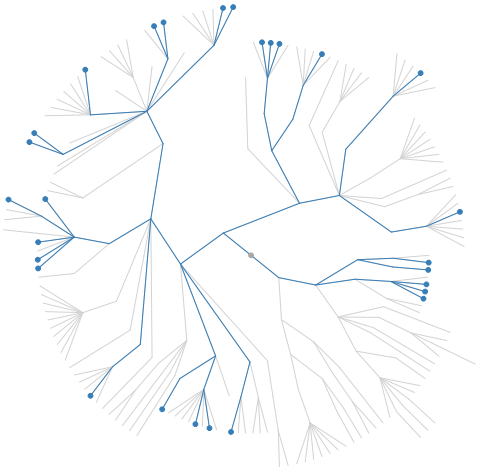
A research assistant working with our group was recruited to act as a second translator. He was briefed on our interest in studying password policies and our specific approach of translating them into a formal grammar. We provided him with the translation instructions reprinted in Appendix A, and went over those instructions with him verbally. Prior to this effort, our translator was not familiar with the EBNF notation or regular expressions in the instructions, and they were explained by the experimenter.



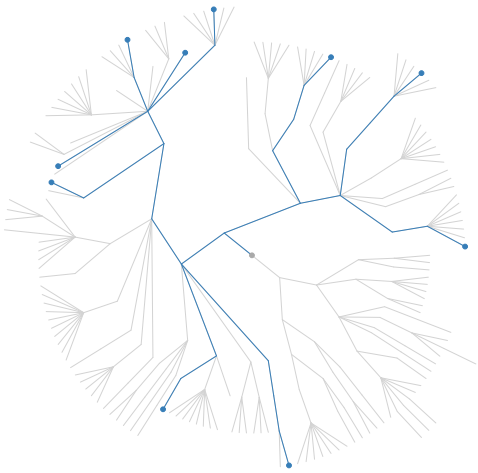
NIST



Census Bureau

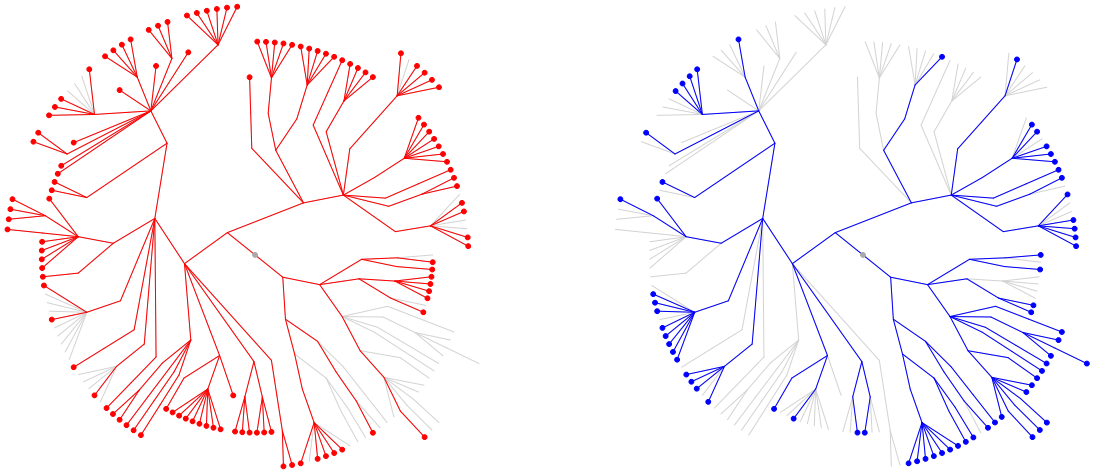


Department of Commerce



Marines (MCEN)

Figure 2: Panels comparing four different workplace password policies.



Workplace Policies Personal-Site Policies

Figure 3: Panels comparing and contrasting the statements seen in workplace and personal-site password policies.

We marked which lines of the policies were within scope before giving them to the translator. Discriminating between within-scope and out-of-scope statements was done simply so that we would have a workable, small language for this initial study. We did not feel that exploring the repeatability of that process was as important as exploring the repeatability of translating within-scope statements into the formal language.

After this round of instructions, we presented the second translator with six password policies; five were randomly selected from the corpus and one was the NIST policy, specifically selected because of our familiarity with it. The translator was given a week to perform the translations among his other duties. During this time, he asked a few clarifying questions but performed largely independently.

The translator was not told that his translations would be compared with those of the original translator, only that some of the policies had already been translated. Our intent in avoiding this topic was to prevent an impression of a "right answer" against which his answers would be evaluated. We wanted him to interpret the policies independently, insofar as the constraints of the formal language allowed.

Upon receiving the translations from the second translator, we performed an extensive analysis, the details of which are in Appendix C. Due to logistical constraints, we were not able to make a parser available to the translator during his translation. Consequently, not all of his translated statements were legal according to the grammar. We fixed small syntactic errors where possible, but when we could not map the translator's statement to an unambiguous legal statement in the formal language, we dropped it as untranslated. (In a few other cases, the translator himself was uncertain about a translation and so did not submit one.)

Having made these corrections to the translations, our next step in the analysis was to compare translations across translators. This procedure too is extensively documented in Appendix C, with the results shown in Table 1. The results are organized by policy and the results of the comparison. For instance, the 2 in the first row and column means that, when translating the

	Amazon Customer	Ames Laboratory	Maui High Performance Computing Center	National Institute of Standards and Technology	TD Bank Financial Group Customer	USDA Biosafety Level-3 Facilities	Total
Same	2	7	4	11	3	3	30
Untranslated	1	1		7		1	10
Missing	1		1	8	1	2	13
Must/Should		1	4				5
Whole/Substring		1		2		1	4
Other Disagreement			3	3			6
Totals	4	10	12	31	4	7	68

Table 1: Breakdown of how many statements between the translators were the same, how many differed, and how they differed. Each column corresponds to a different password policy. The first row (Same) counts how many of the statements were the same in both translations. The remaining rows count how many statements differed and classifies them based on how they different. Marginal counts of the number of statements per policy and per class are provided at the ends of the columns and rows respectively.

Amazon Customer policy, the two translations agreed exactly on 2 statements. The 4 in the fourth row and third column means that, when translating the MHPCC policy, the translators disagreed only on whether a rule was a *must* rule or a *should* rule 4 times.

This table offers some interesting insight. The first row (Same) counts the number of statements that are translated exactly the same by both translators. A total of 30 statements (44%) were translated exactly the same. As inter-rater reliability statistics go, this score seems quite low. In a typical binary-classification problem, an inter-rater reliability of 50% is equivalent to random guessing. However, our translators are not performing binary classification, and the total number of possible statements (i.e., classes) is well over 100. As such, we are unsure how negatively to interpret this number, and must look at where the disagreements arose.

The second row (Untranslated) breaks down the number of statements that were included in one translation and untranslated in the other. These include statements that were not translated and statements whose translations were dropped for being ill-formed and ambiguous. Since there were 10 untranslated statements (15%), the large majority from the NIST password policy, a lot of the disagreement in translation occurred because the translator did not know how to correctly phrase

the translation. Consequently, better instruction and access to the parser for testing translations might improve the inter-rater reliability.

The third row (Missing) breaks down the number of statements that were in one translation but not in the other. One might ask how this row differs from the second row. Missing statements occur when a line in the original policy is expanded to multiple statements in the formal language. For instance, the line "do not use dictionary words or names" might be translated as: (a) *Users must not create passwords in the set of dictionary words.* (b) *Users must not create passwords in the set of proper nouns.* If both translators include statement (a) but only one translator includes statement (b), we classify it as Missing rather than Untranslated. The translator did not necessarily struggle with the translation, he simply decided the Missing statement was not necessary.

The fourth row (Must/Should) breaks down the number of statements where the only disagreement between translators is whether the statement is a *must* or a *should*. Often the source password policy is ambiguous about this distinction. The 5 statements (7%) in this class may represent legitimately different interpretations of whether a policy is discouraging or prohibiting behavior.

The fifth row (Whole/Substring) tallies the number of statements where the translators disagree only in whether a string is prohibited as a whole password or as a substring of a password. For instance, one translator might say *Users must not create passwords in the set of dictionary words*, and the other says *Users must not create passwords* ***with a substring*** *in the set of dictionary words.* This pair of translations represents a Whole/Substring disagreement. There were 4 statements (6%) where this was the difference in the translations. Again, these disagreements may represent ambiguity in the source password policies.

The final row (Other Disagreements) is a catch-all for the 6 statements (9%) that disagreed in some other way. These statements include situations where one translator translated a line as forbidding *addresses and other locations* in passwords and the other forbid *personally identifying information.* They also include two cases where the second translator unintentionally exercised the grammar to very nearly find two different, equivalent ways to express the same statement. We may wish to modify grammar and instructions to deflect such attempts.

A further observation from this pilot is that many of the statements classified as Missing or Other Disagreements concern a few strings that describe what can and cannot occur in a password: *addresses and other locations*, *birthdays and other dates*, *personally identifying information*, *proper nouns*, and *dictionary words.* These sets of strings include some overlap, and most would agree that none of them should be part of a password. For instance, if a policy forbids birthdays and family names, it seems more likely that the policy writer forgot to forbid addresses than that he or she has intentionally, tacitly allowed them. Consequently, we might refine the grammar and improve inter-rater reliability by coming up with a single descriptor that captures all of these strings.

6 Summary and Future Work

This work arose from a need to deal with ambiguities in current password policies. These ambiguities not only prevent researchers from comparing and contrasting policy statements, they also could cause users to misinterpret what is expected of them. To tackle ambiguity, we developed a formal language for stating what behavior is and is not allowed when creating, managing, and changing passwords. This formal language lends itself to policy analysis and visualization.

We have used this language to begin to explore the diversity and complexity of password policies by examining a corpus of 41 password policies collected from a variety of workplaces and personal sites. These policies regulate not just what passwords are legal when they are created but where they can be stored, how they can be communicated, and when they must be changed. Of the 41

policies we considered, no two exactly agreed on user behavior regarding passwords.

Having these clear, unambiguous policy statements will enable us to study their properties. What quantifiable security benefits are provided by a statement or set of statements? What usability issues to the introduce (e.g., by requiring user's to remember their passwords)? These usability issues can also be quantified and studied empirically (e.g., whether a policy statement significantly increases the number of forgotten-password incidents).

References

Anne Adams and Martina Angela Sasse. Users are not the enemy. *Communcations of the ACM*, 42(12):41–46, 1999.

Joseph Bonneau and Sören Preibusch. The password thicket: Technical and market failures in human authentication on the web. In *Proceedings of the 9th Workshop on the Economics of Information Security (WEIS-2010)*, pages ??–??, June 7–8,2010, Cambridge, MA, 2010. ACM Press.

Bill Cheswick. Rethinking passwords. Presentation at the Solaris Security Summit, Baltimore, MD, 2010. `http://www.cheswick.com/ches/talks/baltimore.pdf`, accessed Oct. 2010.

Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. Password strength: An empirical analysis. In *Proceedings of 29th IEEE Conference on Computer Communication (INFOCOM-2010)*, pages 1–9, Mar 14–19, 2010, San Diego, CA, 2010. IEEE Press.

Stephen Farrell. Password policy purgatory. *IEEE Internet Computing*, 12(5):84–87, 2008.

Dinei Florêncio and Cormac Herley. A large-scale study of web password habits. In *Proceedings of the 16th International World Wide Web Conference (WWW-2007)*, pages 657–666, May 8–12, 2006, Banff, Alberta, 2007.

Dinei Florêncio and Cormac Herley. Where do security policies come from? In *Proceedings of the 6th Symposium on Usable Privacy and Security (SOUPS-2010)*, July 14–16, 2010, Redmond, WA, 2010. ACM Press, New York, NY.

Steven Furnell. An assessment of website password practices. *Computers & Security*, 26:445–451, 2007.

Philip G. Inglesant and M. Angela Sasse. The true cost of unusable password policies: Password use in the wild. In *Proceedings of the 28th International Conference on Human Factors in Computer Systems*, pages 383–392, Apr 10–15, 2010, Atlanta, GA, 2010. ACM Press, New York, NY.

Maritza Johnson, John Karat, Klare-Marie Karat, and Keith Grueneberg. Optimizing a policy authoring framework for security and privacy policies. In *Proceedings of the 6th Symposium on Usable Privacy and Security (SOUPS-2010)*, July 14–16, 2010, Redmond, WA, 2010. ACM Press, New York, NY.

Daniel V. Klein. Foiling the cracker; a survey of, and improvements to unix password security. In *Proceedings of the 2nd USENIX Security Symposium*, pages 5–14, Aug 27–28, Portland, OR, 1990. USENIX.

Mohammad Mannan and Paul C. Van Oorschot. Security and usability: The gap in real-world online banking. In *Proceedings of the New Security Paradigms Workshop (NSPW-2007)*, pages 1–14, Sep 18–21, 2007, North Conway, NH, 2007. ACM Press, New York, NY.

Robert Morris and Ken Thompson. Password security: A case history. *Communications of the ACM*, 22(11):594–597, 1979.

Simon E. Parkin, Aad van Moorsel, and Robert Coles. An information security ontology incorporating human-behavioural implications. In *Proceedings of the 2nd International Conference on Security of Information and Networks (SIN-2009)*, pages 46–55, Oct 6–10, 2009, Famagusta, North Cyprus, 2009. ACM Press, New York, NY.

SANS Institute. SANS password policy, 2006. `http://www.sans.org/security-resources/policies/Password_Policy.pdf`, accessed Oct. 2010.

Richard J. K. Shay, Abhilasha Bhargav-Spantzel, and Elisa Bertino. Password policy simulation and analysis. In *Proceedings of the ACM Workshop on Digital Identity Management*, pages 1–10, Nov 2, 2007, Fairfax, VA, 2007. ACM Press, New York, NY.

Gene Spafford. Security myths and passwords, 2006. `http://www.cerias.purdue.edu/site/blog/post/password-change-myths/`, accessed Oct. 2010.

Wayne C. Summers and Edward Bosworth. Password policy: The good, the bad, and the ugly. In *Proceedings of the Winter International Symposium on Information and Communication Technologies*, pages 1–6, Jan 5–8, 2004, Cancun, Mexico, 2004. Trinity College, Dublin.

Thomas Wu. A real-world analysis of Kerberos password security. In *Proceedings of the ISOC Symposium on Network and Distributed System Security (NDSS-1999)*, San Diego, CA, 1999. Internet Society.

Wenjuan Xu, Mohamed Shehab, and Gail-Joon Ahn. Visualization based policy analysis: Case study in SELinux. In *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies (SACMAT-2008)*, pages 165–174, June 11–13, 2008, Estes Park, Colorado, 2008.

A Instructions for Translating Policies

A password policy may seem formal in the sense that it is written in a legalistic language, giving the impression of a binding contract. However, such policies are *informal* in the logical sense that the policy statements are not written in a clear, unambiguous form. For the Password-Policy Taxonomy project, we use a formal language to explicitly capture what is expected of the user. This document explains the procedure that we use to translate the statements in the informal language of standard password policies to the formal language for further study.

Because this work is preliminary, we do not translate statements within every topic covered by any password policy. Instead, we restrict our attention to a subset of topics, acknowledging the remainder as presently beyond scope. In Section A.1, we lay out the criteria used to differentiate between topics within and out of scope. We also describe the procedure for marking up policy documents to identify within-scope statements.

Once a statement is identified as within scope, the translation procedure is akin to writing statements in a programming language. In fact, we describe the syntax of the formal language in Extended Backus-Naur Form (EBNF) notation, typically used in programming-language specifications. The grammar of the language is presented in Section A.2 along with a description of the conventions we used when translating ambiguous, informal statements into the formal language.

In Section A.3, we present an example policy and translate it into the formal language. A reader simply trying to understand the general concept of this translation process and not the details might want to skim that section first. It should provide one with a better picture of the process before delving into the details presented in Sections A.1 and A.2.

A.1 Identifying Within-Scope Statements

Upon initial inspection, similar statements are made by all password policies. Policies contain length and character-set requirements, restrictions on how passwords are stored and whether they can be communicated, and warnings about what will befall a user who incorrectly enters a password too many times. While individual requirements, restrictions, and consequences may differ slightly, the impression is that all policies cover largely similar territory.

However, after examining many policies, apparent differences emerge in the topics covered by different policies. Some policies regulate applications' and system-administrators' behavior as well as users'. Some policies discourage passwords entirely, recommending either a security token or long passphrases with dramatically different composition requirements. Some policies lay out multiple sub-policies, each pertaining to passwords for different kinds of accounts (e.g., shared, normal-user, privileged-user, or system accounts).

Rather than attempt to capture the full diversity of the topics covered in password policies, we identified a subset of topics that commonly appear among many different policies. We ensured that the subset covers those topics which most closely regulate user behavior. The subset was found to be small enough as not to overwhelm our study with minutia, but large enough that the analysis provides useful insight.

A.1.1 Out-of-scope topics

The following topics were judged to be presently beyond scope:

1. **Passphrases, PINs, and Alternatives:** Some policy writers use *passphrase* and *password* synonymously, while others define a *passphrase* to be longer than a password (e.g., 15 or more characters), created by assembling a sequence of words (not characters). This latter

notion of a passphrase is certainly worth study, especially as several security experts have started to recommend them, but our focus is restricted to passwords for now. Likewise, personal identification numbers (PINs) for personal digital assistants (e.g., Blackberries) are beyond scope. More generally, alternatives to passwords are not yet included in the formal language. Some policies explicitly discourage password use in favor of one-time passwords or public keys. Such statements are ignored.

2. **Non-standard Accounts and Access:** A single policy can regulate behavior for several different kinds of accounts. For instance, a policy might specify one length and lifetime for general users, a stricter length and lifetime for users with elevated privileges (e.g., those who handle personally identifying information), and require documented permission from management to maintain a group account with a shared password. For simplicity, we restrict our focus to one account per policy: the general-user account. When it is unclear which of multiple accounts corresponds to that of a general user, we select the least-privileged, non-guest account.

3. **Non-User Agents:** A password policy might regulate the behavior of many parties, not just the user. For instance, system administrators might be required to change passwords or disable accounts when users are terminated or transfered to another group. Applications may be required to obscure passwords when they are typed rather than displaying them on the screen. While formalized password-policy statements could have many applications, they were formulated to help us express the expectations and responsibilities placed on users. Consequently, regulations of the behavior of these non-user agents are beyond scope.

4. **Infrequent Topics:** Some policy topics are judged infrequent, either in the sense that they are covered in very few policies, or they regulate behavior which arises rarely (in our judgement). The consequences of forgotten passwords and expired accounts fall into this category. Many policies do not cover these topics explicitly so we postpone their study to future work. Likewise, initial-password procedures are beyond scope (e.g., how new accounts are assigned passwords and when they must be changed). Statements about such procedures regulate behavior that is only relevant once per account, and they typically involve the activity of non-user agents.

   Confusion may arise because password lockout is within scope. One might argue that the number of incorrect password guesses and the consequences of guessing incorrectly are either infrequent topics or involve too many non-user agents. We recognize that there are shades of gray in determining what is within scope, and we decided that, despite its complexity, the study of various policies' lockout requirements would be worth exploring even at this early stage.

5. **Platitudes and Generic Advice:** Likewise, we do not translate general tips and tricks for creating passwords into the formal language. Choosing a password that is easy to remember but hard to guess is easier said than done. Guidance about creating acronyms from phrases or converting letters to numbers may help users choose good passwords but they are suggestions not regulations. Only tips with explicit recommendations or prohibitions are translated (e.g., using mixed-case letters or avoiding proper nouns).

The formal password-policy language could be extended to express statements about these out-of-scope topics (as should become clear in Section A.2). To do so for the present work seemed premature. If the current work proves useful, such extensions might be warranted.

A.1.2 Marking within-scope statements

For this work, each password-policy document has been stored as a PDF (Portable Document Format). The name of the organization, date on which the document or webpage was obtained, URL from which it was accessed, and name of the PDF are recorded. Each document was viewed and processed using a PDF viewing application that allows the insertion of text annotations. The translator examines the document to find sentences, bullets, or other spans of text judged to be a within-scope statement regulating users' behavior when creating or managing passwords. To identify the statements in a policy document that should be translated, the translator inserts a parenthetical number in the document margin, starting at (1) and incrementing with each identified span of text. To make them stand out, we color these marks in red. These parenthetical indices will be used to link translated statements back to the source document.

A.2 Translating into the Formal Language

Having identified those policy statements that are within scope in Section 1, this section describes how these statements are translated into the formal password-policy language. The password-policy topics captured by the formal language are those which govern user behavior in password creation and use. These expectations on user behavior are expressed as explicit statements about what a user *must*, *must not*, *should*, and *should not* do.

As a formal language there are absolute rules (called a grammar) regarding how policy statements must be phrased. In the same way that a misplaced word or punctuation mark might cause a syntax error in a programming language, an incorrectly phrased policy statement will violate the rules of the policy grammar. This grammar has been specified in a concise, unambiguous notation called Extended Backus-Naur Form (EBNF). The notation is commonly used to describe the syntax of programming languages. We adopt it because tools exist to parse and automatically check the correctness of a document written with an EBNF-specified grammar.

Throughout this section, the formal rules of the EBNF grammar are presented alongside descriptions of what the rules mean. Readers unfamiliar with EBNF notation should still find the syntax of the rules fairly intuitive. First, we describe the overall structure of policies and statements. Then, we discuss each of the different kinds of statement in separate subsections.

A.2.1 Structure of a password policy

Formally, a password policy is a collection of statements about behavior:

```
1 policy: policy_stmt(s)
```

This rule formally defines a policy to be a collection of one or more policy statements (`policy_stmt`'s). The parenthetical `(s)` denotes one or more policy statements.

Each policy statement is decomposed into the following parts:

```
1 policy_stmt: rule_idx "Users" modal_verb pwd_stmt '.'
```

A policy statement is divided into a rule index (`rule_idx`), and a sentence with "`Users`" as the subject, a modal verb phrase (`modal_verb`), a statement about passwords (`pwd_stmt`), and a period ("`.`") to terminate the sentence.

The rule index is defined by the following rule:

```
1 rule_idx: /\d+\w*/ ":"
```

The index of the rule is intended to connect the formal policy statement with a location in the informal policy document for later reference. It is encoded as a number followed by zero or more letters and then a colon. The expression `/\d+\w*/` is a Perl regular expression denoting one or more digits followed by zero or more letters. For instance, "`1:`" and "`2a:`" are valid rule indices while "a2:" and "`1`" (with no colon) are not.

The numeric portion of the rule index is intended to correspond with the number marked (according to Section A.1.2) next to the informal statement being translated. The alphabetic portion is intended to differentiate multiple formal statements that correspond to the same informal statement.

The substance of a policy statement is the sentence that follows the rule index. The subject of the sentence is "`Users`," and the subject is followed by one of four modal verb phrases according to the following rule:

```
1 modal_verb: "must not"
2       | "should not"
3       | "must"
4       | "should"
```

The rule uses a vertical bar (`|`) to denote a choice among four options. Each option has its standard meaning. For instance, a sentence beginning "`Users must not`" is prohibiting behavior, and a sentence beginning "`Users should`" is recommending behavior.

Sometimes it can be tricky to determine whether an informally written policy is forbidding an activity (`must not`) or merely discouraging it (`should not`). When deciding, we look for strong statements (e.g., "Never use a dictionary word") as evidence of a requirement. However, if such statements are wrapped in weaker wording (e.g., "Consider the following tips for creating a strong password:"), we view the statement as a recommendation. Note that sometimes a policy will present tips for creating a strong password and later clarify that these tips must be followed. Careful reading is often required.

After the modal verb comes the verb phrase that explicitly defines the behavior regulated by the statement:

```
1 pwd_stmt: change_stmt
2       | commun_stmt
3       | create_stmt
4       | store_stmt
5       | failauth_stmt
```

As with `modal_verb`, the vertical bars denote a choice. In this password statement, the choice is between five different topics: (1) changing a password, (2) communicating a password, (3) creating a password, (4) storing a password, and (5) failing to authenticate. Each of these five kinds of statement have different structure and are described in separate sections.

A.2.2 Rules on password lifetimes and expiry

Statements that regulate when users must change their password have the following form:

```
1 change_stmt: "change passwords" time_crit
2       | "change passwords" time_crit "if" event_name
```

The change statement (`change_stmt`) is either an absolute or a conditional statement. The absolute statement requires only a time criteria (`time_crit`) as defined by the following rule:

```
1 time_crit: "immediately"
2       | "before" day_name
```

In the password policies we have examined, passwords must either be changed immediately or before a certain number of days have passed. The number of days must be specified according to the `day_name` rule:

```
day_name: /1 day/
     | /\d+\s+days/
```

As an example of the use of the absolute change statement, if an informal policy stated, "Passwords must be changed every 90 days," we would translate that as

```
1: Users must change passwords before 90 days.
```

The conditional statement is used to express the fact that, if a certain event (`event_name`) takes place, the password must be changed according to a specified time criteria. The kinds of events described in password policies are captured by the following rule:

```
event_name: "compromised"
     | "directed by management"
     | "found non-compliant"
     | "sent unencrypted"
     | "shared"
```

An informal policy statement might not correspond exactly to one of these events, but the match should be close to one. For instance, an informal policy might state, "Passwords that have been compromised or are suspected of being compromised must be changed right away." We would translate such a statement into the formal statment

```
2: Users must change passwords immediately if compromised.
```

A.2.3 Rules on password communication and transmission

Statements concerning whether users talk about or transmit their passwords (e.g., over the Internet) are defined by the following rule:

```
commun_stmt: "communicate passwords" "to" person_name
     | "communicate passwords" "by" media_name
     | "communicate passwords" "except in an emergency"
```

In accordance with this rule, communications regulations take one of three forms.

The first form governs to whom a user can communicate a password. Only two sets of people (`person_name`) appear in the password policies we examined:

```
person_name: "a third party"
     | "anyone"
```

Fir instance, an informal statement such as "Do not share your password with friends, family, secretaries, etc." would be translated to the statement

```
3: Users must not communicate passwords to anyone.
```

The second form governs the transmission medium (`media_name`) by which a user can or cannot communicate passwords:

```
media_name: "any means"
     | "any network without encryption"
     | "any network"
     | "email without encryption"
     | "email"
     | "mail without encryption"
     | "mail accompanied by the user ID"
     | "mail"
     | "phone mail"
     | "phone"
     | "Internet or wide-area network without encryption"
     | "Internet or wide-area network"
     | "local-area network without encryption"
     | "local-area network"
```

As in other cases, a translator may need to make a judgement about which of the choices is most appropriate in a given translation. For instance, we would translate the informal statement, "Passwords on the Internet must be encrypted in compliance with FIPS 140-2," as

```
4: Users must not communicate passwords by Internet or wide-area network
   without encryption.
```

The third form exists as a special case. Several policies make an exception to an overall ban on communicating passwords in the event of an emergency or other extraordinary circumstances. Such exceptions are translated into the formal language as:

```
5: Users must not communicate passwords except in an emergency.
```

A.2.4 Rules on password creation and composition

The majority of statements about passwords govern their length, the characters that must (and must not) appear, and other choices that are made when the user creates the password. Statements about password creation are defined by the following rule:

```
1 create_stmt: "create passwords" "with length" charlen_cmp
2      | "create passwords" /with a character in the first \d+ characters/ charset_crit
3      | "create passwords" "with a character" charset_crit
4      | "create passwords" /with \d+ or more characters/ charset_crit
5      | "create passwords" "with all characters" charset_crit
6      | "create passwords" "with an internal character" charset_crit
7      | "create passwords" "with a first or last character" charset_crit
8      | "create passwords" "with a substring" stringset_crit
9      | "create passwords" stringset_crit
```

These nine different kinds of password-creation requirement basically fall into one of three classes: length (line 1), character set (lines 2–7), and string set (lines 8–9). We consider each class separately.

(Rules on password length) Formal statements about password length include the phrase "`create passwords with length`" followed by a character length comparison (`charlen_cmp`) defined:

```
1 charlen_cmp: "greater than or equal to" char_length
```

The only comparison is an inequality, requiring a character length (`char_length`) defined:

```
1 char_length: /\d+\s+characters/
```

Any number followed by spaces followed by the word "`characters`" is a valid length. Comparisons are expressed with inequalities of the single form "`greater than or equal to`" because all other forms of inequality and equality can be derived from it. This restriction prevents two different formal statements from expressing equivalent expectations about user behavior. For instance, the informal statement "Passwords must be between 8 and 20 characters" can only be translated as

```
6a: Users must create passwords with length greater than or equal to 8
   characters.
6b: Users must not create passwords with length greater than or equal to
   21 characters.
```

Note that one informal statement may need to be translated as multiple formal statements. We denote these different formal statements by assigning each one a different letter in the rule index (i.e., $6a$ and $6b$).

(Rules on password character sets) Statements about password character sets regulate which characters are allowed (or recommended) at which positions within a password. The six different character-set choices (lines 2–7 in the `create_stmt` definition) lay out six different sets of positions that are regulated. For instance, consider line 2 which regulates `/a character in the first \d+ characters/`, and again, the pattern `\d+` means any number is valid. This choice would be used to capture a statement requiring that certain kinds of characters be in the first 7 positions in a password. Such statements appear in some policies.

All six choices of creation statement which regulate character sets include a character set criteria (`charset_crit`) defined as follows:

```
1 charset_crit: "in the set of" charset_name
```

The only criteria concerns membership in a character set (`charset_name`), defined as:

```
1 charset_name: basic_charset(s /,/)
```

All character sets that we examined were combinations of one or more basic types of character set (`basic_charset`), which we specify as a comma-separated list. These basic character sets are:

```
1  basic_charset: "upper-case letters"
2       | "lower-case letters"
3       | "letters (unspec)"
4       | "numbers
5       | /these special characters: .+ /
6       | "special characters (unspec)"
7       | "punctuation"
8       | "control or non-printable characters (unspec)"
9       | "whitespace"
10      | "all other characters (unspec)"
11      | "those not used in the previous password"
12      | "those used in the previous password"
```

The characters contained in each basic character sets are often self evident. In some cases, where the actual set of characters is ambiguous, we include the "`(unspec)`" tag in the name. The tag makes the under-specified nature of the character set explicit. Statements involving special characters are often under-specified, as in "special characters (e.g., , $, %)." Except for these examples, the characters in the set are not specified. In those cases where a full set of special characters

is provided, the set can be captured using line 5 of the `basic_charset` rule. Note that the list of special characters must be separated on both sides by whitespace. For instance, the informal statement "Passwords must have one or more letters (upper or lower-case), digits (which can't be in the first or last position), and the following special characters: `!@#$%^&*()`" would be translated as

```
7a: Users must create passwords with a character in the set of upper-case
   letters, lower-case letters.
7b: Users must create passwords with an internal character in the set of
   numbers.
7c: Users must create passwords with a character in the set of these special
   characters: !@#$%^&*() .
```

Note that, by convention, when a character set includes multiple basic character sets separated by a comma, we order them in the same order as listed above (e.g., "upper-case letters" comes before "lower-case letters").

Regarding translations involving special characters, we consider the terms "special character," "symbol," and "punctuation" to be equivalent, *except* when punctuation is explicitly differentiated from other kinds of non-alpha-numeric characters. In such cases, and only in such cases, is the punctuation character set used. For instance a requirement for "numbers, punctuation, or all other characters" would be translated as "`numbers, punctuation, all other characters (unspec)`."

(Rules on password string sets) Some policies regulate not just characters in the password but whole sequences of characters (or strings). As in line 8 of the `create_stmt` rule, these statements can involve any substring of the whole password or, as in line 9, they can involve only the whole password. Both kinds of statement about sets of strings require a string-set criteria (`stringset_crit`), defined as

```
1 stringset_crit: "in the set of" stringset_name
2       | "equal to" string_name
```

We have seen two kinds of criteria, equality and set membership. The equality choice requires a string name (`string_name`):

```
1 string_name: "the user ID"
2       | "their name"
```

Only two strings have been regulated, the user ID and the user's name. For instance, the informal policy statement, "Do not include your user ID in your password" would be translated as

```
8: Users must not create passwords with substrings equal to their user ID.
```

More commonly, we see set-membership regulations. These statements involve a set of strings (`stringset_name`) defined by the following rule:

```
1 stringset_name: "addresses or other locations"
2       | "birthdays or other dates"
3       | "dictionary words" "followed by a number"
4       | "dictionary words" "in reverse"
5       | "dictionary words" "with numbers substituted for letters"
6       | "dictionary words"
7               "preceded or followed by a number or special character (unspec)"
8       | "dictionary words"
```

```
9       | "incremental changes to existing passwords (unspec)"
10      | "otherwise forbidden content" "concatenated"
11      | "otherwise forbidden content" "in reverse"
12      | "otherwise forbidden content" "preceded or followed by a number"
13      | "passwords" "to an outside system"
14      | "passwords" "to any other system"
15      | "proper nouns"
16      | "personally identifying information"
17      | "strings with" /a character repeated \d+ or more times consecutively/
18      | "strings with" /a character repeated \d+ or more times/
19      | "strings with" /a run of \d+ or more consecutive characters in sequence/
20      | "strings with" /at least \d+ unique characters/
21      | "strings with" /characters from \d+ of these \d+ sets:/ charset_name
22      | "strings with" "word or number patterns (unspec)"
23      | "their last" /\d+ passwords/
24      | "their last" /\d+ years of passwords/
25      | /those passwords used \d+ times in the last \d+ years/
26      | "vendor default passwords"
```

Many of the string sets are self explanatory. Rather than splitting hairs, any prohibition against using names—family, pet, sports teams, or fantasy character—are grouped under the set of `proper nouns`. Likewise, different policies forbid words from different dictionaries, but the current grammar ignores such nuances and groups all such sets as `dictionary words`. For instance, the informal statement "Your password cannot be a word in English, Japanese, or Klingon, either forward or in reverse" would be translated as

```
9a: Users must not create passwords in the set of dictionary words.
9b: Users must not create passwords in the set of dictionary words in
   reverse.
```

We have found that these sets satisfactorily capture the sets that appear in the password policies that we have examined. In the future, as new policies are examined, this rule might need to be expanded to allow more choices.

A.2.5 Rules on writing and storing passwords

Statements that regulate where and how users can store their passwords have the following form:

```
1  store_stmt: "store passwords" "in writing" loc_name
2       | "store passwords" "online" loc_name
```

Two kinds of storage are regulated: written and online. Both cases require a location name (`loc_name`) to explain the regulation. The location name is defined as follows:

```
1  loc_name: "anywhere"
2       | "in a secure location"
3       | "in an insecure location"
4       | "in automated scripts"
5       | "in clear text or weakly encrypted"
6       | "in clear text in an insecure location"
7       | "on outside systems"
```

Some policies forbid writing a password anywhere; others forbid writing them in clear text; and still others forbid writing them in clear text and storing that clear-text password in an insecure location. Each of these different regulations is expressed as a different location (`loc_name`).

Once again, a translator must decide which of the choices in the formal grammar most closely correspond to the informal statement. For instance, consider the statements "You should never write your password in a place where others might find them," and "Passwords must be stored at a protection level no lower than that of the information protected by the password." We would translate both of these as

```
10a. Users must not store passwords in writing in an insecure location.
10b. Users must not store passwords online in an insecure location.
```

Prohibitions on using "remember password" features of web browsers or other applications fall under the "`in automated scripts`" choice. The umbrella location "`in clear text or weakly encrypted`" is meant to include restrictions on "clear text" with no mention of weak encryption.

A.2.6 Rules on failed authentications and lockout

Password-lockout statements can be viewed from a user's perspective as an expectation (i.e., a limit on the number of incorrect authentication attempts) and a consequence for failing to meet that expectation (e.g., lockout for 15 minutes). Failure-to-authenticate statements are formally coded using the following grammar rule:

```
1 failauth_stmt: "fail to authenticate" rep_name "to avoid" lockout_name
```

They require a repetition name (`rep_name`) which is define as:

```
1 rep_name: /\d+ times in a \d+ \w+ interval/
2       | /\d+ times/
```

The policies we examined specify lockout either as a threshold number of incorrect logins (assumed consecutive) or a threshold number in a given time interval.

The consequences of the lockout for the user (`lockout_name`) are defined as follows:

```
1 lockout_name: /administrative unlock or a \d+ \w+ lockout/
2       | "administrative unlock"
3       | "a lockout of unspecified duration"
4       | /a \d+ \w+ lockout/
```

The user is locked out of the account until either some time period (specified or not) has passed or a system administrator performs some action to unlock the account.

For instance, if a policy states, "Lockout occurs after 3 attempts; Users must present their ID at the help desk to have their account unlocked and their password reset," we would translate it as

```
11: Users must not fail to authenticate 3 times to avoid administrative
   unlock.
```

As with all the translation rules, some amount of careful reading and minor approximation may be necessary when rendering the ambiguities and vagueries of informal rules in an explicit and formal language.

A.3 Example

To illustrate the process of translating an informal password policy into the formal policy language, we present an illustrative example. As we go through the steps of the translation, we explain our rationale.

A.3.1 Identifying within-scope statements

The following policy was obtained from NASA's Entry Descent Landing Repository (and modified slightly for the sake of example). It is not one of the policies in our corpus, but it has many of the same structures, making it illustrative:

```
1        Users shall adhere to the following password protections:
2        - Upon first login, the user account password shall be changed
3   (1)  - Users shall not share account passwords
4   (2)  - Passwords shall be between 8 and 31 characters in length, where
5          supported by the operating system
6   (3)  - Passwords shall contain at least one character each from at
7          least three of the following sets of characters: uppercase letters,
8          lowercase letters, numbers, special characters, where supported by the
9          operating system.
10  (4)  - Passwords shall not be a dictionary word.
11  (5)  - Passwords shall not be either wholly or predominantly composed
12         of the following: The user's ID, owner's name, birth date, Social
13         Security Number, family member or pet names, names spelled backwards,
14         or other personal information about the user.
15  (6)  - Passwords shall not be the name of a vendor, product,
16         contractor, project, division,  section or group.
17  (7)  - Passwords shall not be repetitive or a keyboard pattern.
18  (8)  - Passwords shall not be the name of an automobile, sports team,
19         athlete, or other popular cultural symbols.
20  (9)  - Passwords shall not be any of the precluded categories with
21         numbers appended or prepended.
22  (10) - A user account for system access shall have used a minimum of 24
23         passwords before a password can be reused.
24  (11) Note that User account passwords will expire after 90 days. Two
25       notices will be sent via email to the user alerting them to the
26       upcoming expiration. If the password is allowed to expire, you
27  (12) must contact the EDLR Curator to enable the account. Passwords
28       shall not be reused before 2 years have elapsed.
```

The first step in translating this policy is to identify which statements are within scope, as discussed in Section A.1. Line 1 does not concern a specific behavior and so is beyond scope. Line 2 concerns initial passwords which are beyond scope per the *Infrequent Topics* item in Section A.1. Line 3 is the first specific within-scope statement, concerning password communication (though there is some ambiguity as will be discussed below). Since it is within scope, it is marked with a (1). All of the remaining bullets in lines 4–23, are also within scope, so they are marked with an incrementing sequence of numbers in the margin. The first sentence in line 24 is within scope (a change requirement), while the second sentence that begins on that line is not (regulating application behavior). The sentence beginning on line 26 concerns the consequence of an expired password and is beyond scope (again as in *Infrequent Topics*). The final sentence, beginning on line 27, is within scope (concerning password creation). In total, 12 statements were judged to be within scope and marked. In the example policy above, the within-scope statements have been enumerated in red, in the left margin, just as a translator would do when reading the PDF version of the policy.

A.3.2 Translating into the Formal Language

Statement (1) states that users must not share passwords. Note an apparent ambiguity. Some translators might interpret this statement to mean that users should not tell others what their

passwords are. Other translators might fairly interpret the statement to mean that users must not use the same password across multiple accounts. With no way to resolve this ambiguity to our complete satisfaction, we chose the former interpretation, and created the following translation:

```
1: Users must not communicate passwords to anyone.
```

Statement (2) regulates user behavior when creating passwords. Specifically, the statement sets a minimum and a maximum length. Note that the final clause concerns non-user agents (i.e., the operating system), so it is beyond scope. We arrive at this translation:

```
2a: Users must create passwords with length greater than or equal to 8
   characters.
2b: Users must not create passwords with length greater than or equal to
   32 characters.
```

Statement (3) is directly translated into another password-creation requirement:

```
3: Users must create passwords in the set of strings with characters from 3
   of these 4 sets: upper-case letters, lower-case letters, numbers, special
   characters (unspec).
```

Note that the character sets are listed in the same order they appear in the `basic_charset` definition. Since the set of special characters is not defined, we use the *(unspec)* tag.

Statement (4) is translated as:

```
4: Users must not create passwords in the set of dictionary words.
```

Statement (5) restricts the password content in a variety of ways, at a level of specificity we do not include in our formal language. For instance, "pet names" is not among the string sets. However, all the restrictions are part of the more general set of *personally identifying information*, and so we decide that the spirit of the statement can be translated as:

```
5: Users must not create passwords with substrings in the set of
   personally identifying information.
```

Note that we interpret the specifying phrase "not [...] predominantly composed of" to mean "`must not create passwords with substrings in the set of`."

Statement (6) also provides a level of specificity that we do not capture in our rules. Since all of the forbidden things are proper nouns, we translate it as:

```
6: Users must not create passwords in the set of proper nouns.
```

Statement (7) is directly translated as:

```
7: Users must not create passwords in the set of strings with word or number
   patterns (unspec).
```

Because there are many different kinds of patterns and repetitions and the statement does not unambiguously explain what patterns are disallowed, the *(unspec)* tag is used.

Statement (8) provides additional proper names which are forbidden, and so we translate it as a duplicate of statement 6:

```
8: Users must not create passwords in the set of proper nouns.
```

By convention, we may or may not translate duplicate rules. If we marked the duplicate statement as within scope, we create the duplicate translation because every marked statement must have a corresponding translation. However, during markup, if we see that a statement is repeated several times, we have the option of not marking every instance.

Statement (9) can be directly translated as:

```
9: Users must not create passwords in the set of otherwise forbidden
   content preceded or followed by a number.
```

Statement (10) translates as:

```
10: Users must not create passwords in the set of their last 24
   passwords.
```

Statement (11) translates as:

```
11: Users must change passwords before 90 days.
```

Statement (12) translates as:

```
12: Users must not create passwords in the set of their last 2
   years of passwords.
```

To understand how these translations conform to the rules of the formal grammar, we suggest that new translators take each of the translated rules and develop a parse tree. Identify the `modal_verb` and the `pwd_phrase`; determine which kind of statement it is (e.g., `create_stmt` or `change_stmt`); subdivide those statements into their components (e.g., `event_name`s and `stringset_name`s). By checking that each formal statement complies with the grammar, one might develop an intuition about how to compose such statements.

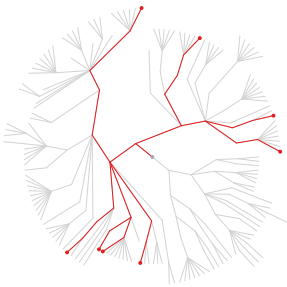
A.4 Summary

In this document, we have described how we translated informal password policies into a formal language specifying the expectations on a user. Our intent was to describe this process in such a way that a new translator might use the description to develop this skill. The first step is to read the policy, and to identify and mark statements which are within scope. The second step is to consider each of the marked statements and compose one or more statements written in the formal policy language that express the same expectation on user behavior. This second step is guided by the syntax of the language (i.e., which formal statements are permissible and which are not), and by a discussion of our thought processes when we choose among the permissible statements.

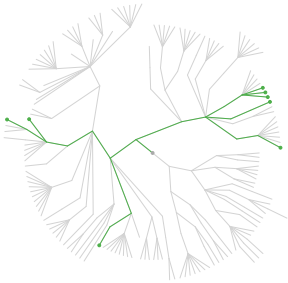
Note that the current grammar rules and the translation procedures are still under development. Because developing a grammar involves some degree of authorship preference, we would not claim that this is the only or the best grammar. Instead, we have tried to remove ambiguity in using the grammar, so that others who might wish to use it are able to do so as consistently as possible. Extensions and modifications are certainly welcome, but we hope that the formal language and these translation instructions prove useful in their current form.

B Policy Trees for Corpus

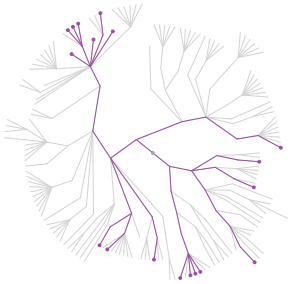
The following 22 password-policy trees were obtained by searching for employee password policies throughout Federal Government websites.



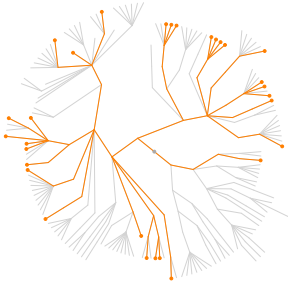
Agency for Healthcare Research and Quality



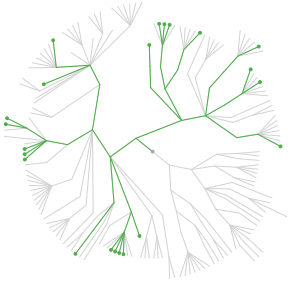
Ames Laboratory



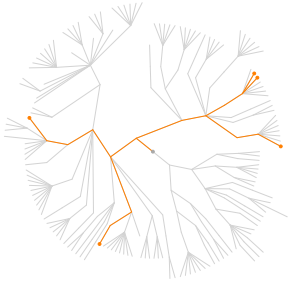
Argonne National Laboratory



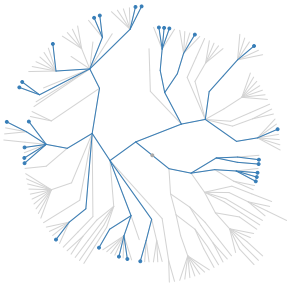
Brookhaven National Laboratory



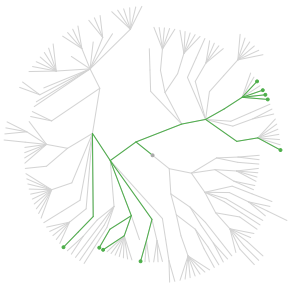
Census Bureau



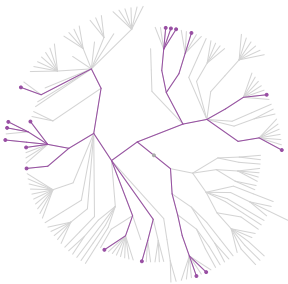
Department of Homeland Security



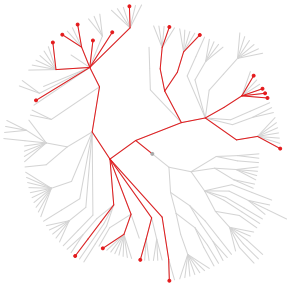
Department of Commerce



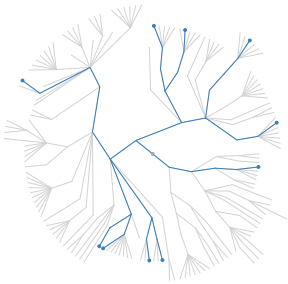
Department of Defense



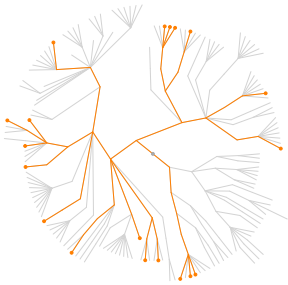
Department of Energy



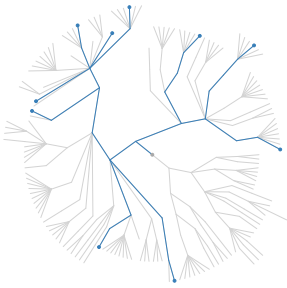
Federal Deposit Insurance Corporation



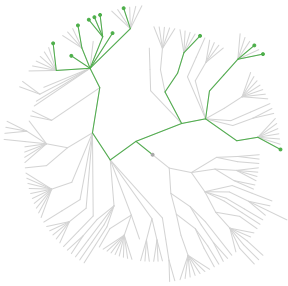
Fermi National Accelerator Laboratory



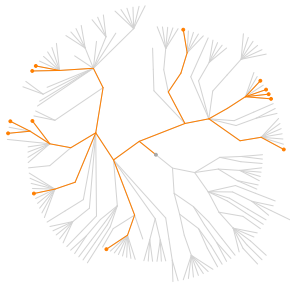
Lawrence Berkeley National Laboratory



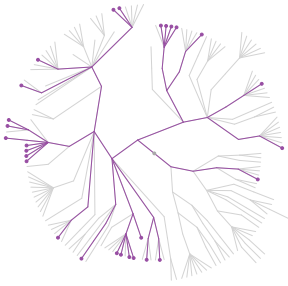
Marine Corps Enterprise Network



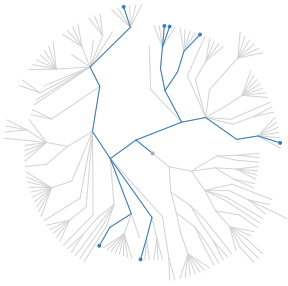
Maui High Performance Computing Center



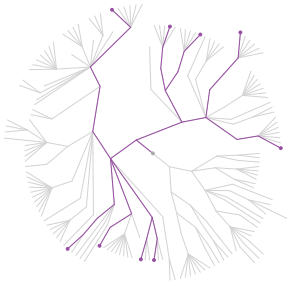
National Aeronautics and Space Administration



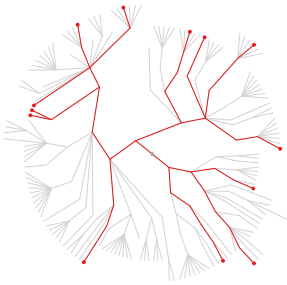
National Institute of Standards and Technology



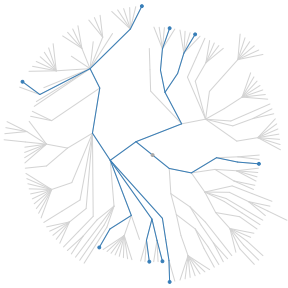
Peace Corps



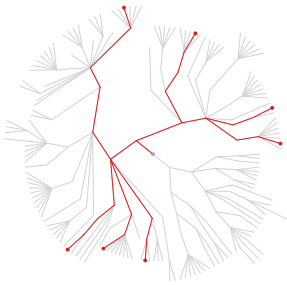
Small Business Administration



SLAC National Accelerator Laboratory



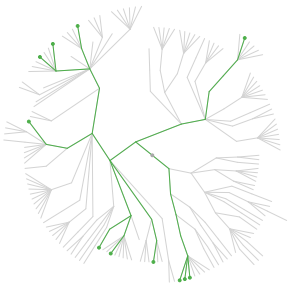
Department of State



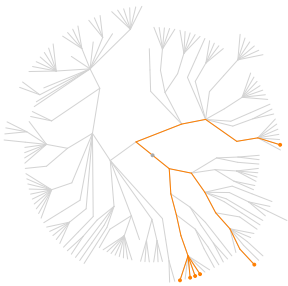
USDA IBM and IBM-compatible Mainframes

USDA Biosafety Level-3 Facilities

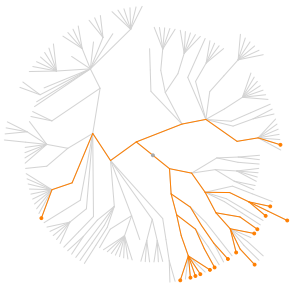
The following 19 password-policy trees were obtained by by searching for the password policies of popular or high-traffic websites with which a typical user might have an account.



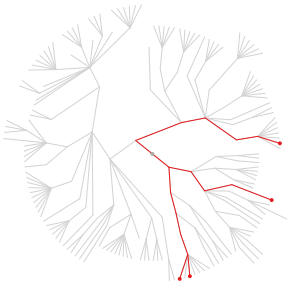
Google User



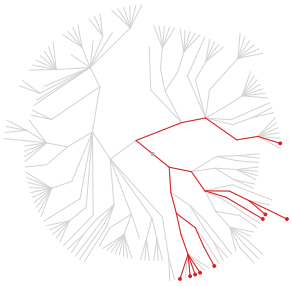
Facebook User



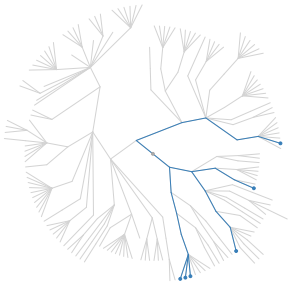
Yahoo User



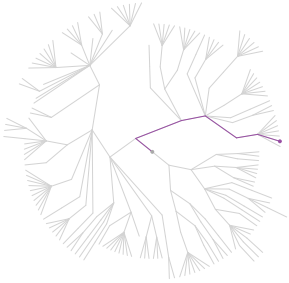
Youtube User



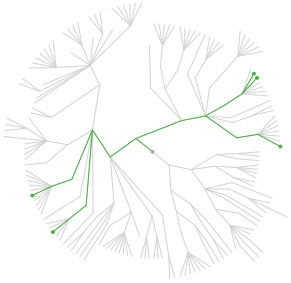
Windows Live User



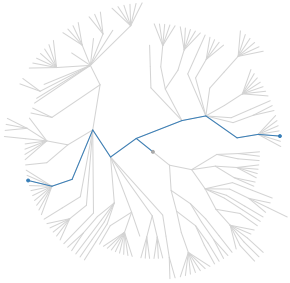
Amazon Customer



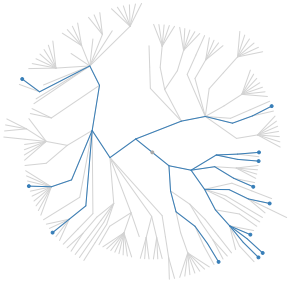
Weather Channel User



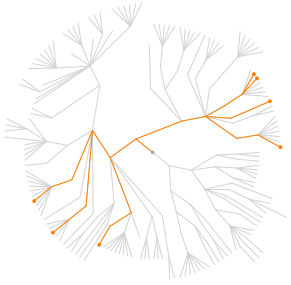
RockYou User



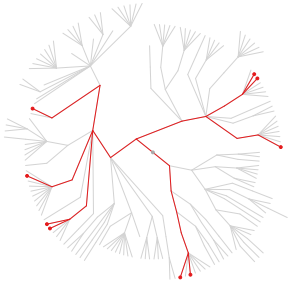
CBS Sports User



Vanguard Customer

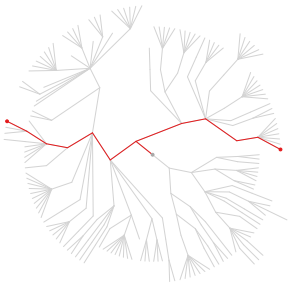


Schwab Customer

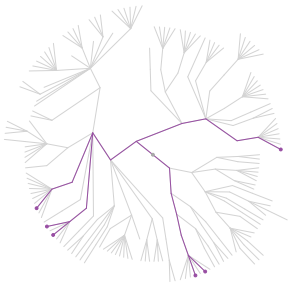


Bank of America Customer

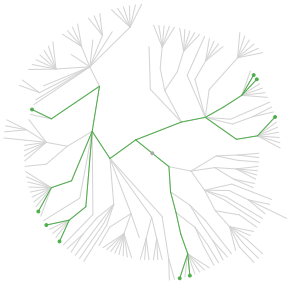
Citibank Customer



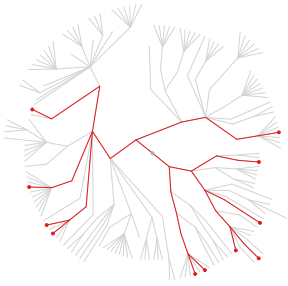
PayPal Customer



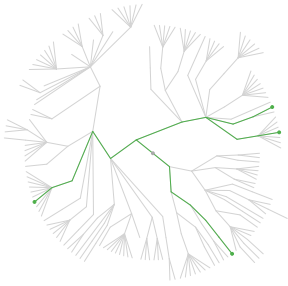
U.S. Bank Customer



Wachovia Online Services Customer



Discover Card Customer



TD Bank Financial Group Customer

HSBC Customer

C Error Analysis of the Replicated Translations

Note: This section is included from personal notes. Because of the informal tone and naming our second translator, it would need to be heavily edited before any wider distribution. I include it in the meantime in the hopes that the additional detail provides some insight.

In this section, I document the procedure that I followed after I received Michael's translations of the six policies. First, I processed each of his translations with the parser. Translations that generated parsing errors were accommodated (i.e., by fixing small syntactic errors or dropping ambiguous translations). Second, I compared each of his translations to each of mine. In each instance where our translations disagreed, I attempted to classify the type of disagreement (e.g., the two translations disagreed over whether a statement was a *must* or a *should* statement). These two steps are described in detail below.

C.1 Parsing the Replicated Translations

When I received the translations back from Michael, the first step was to put them into a format where I could use the Perl-based parser to convert them to policy trees for analysis. I split the Word document containing the six policies into six separate "raw" translation files. I call these raw translations because they are a first attempt at a correct translation, but some statements are not actually legal statements in the formal grammar. In order to perform the intended analysis, I needed to understand what errors were causing the parser to fail and to correct them insofar as that was possible.

In fact, if we were to re-run an experiment to compare translations, I would recommend that all translators be able to "check" whether their translation is legal. If they could be given access to the parser, which would throw a syntax error if a translation contains malformed statements, I think that would help achieve consistency among translations. One might argue that there is little to be learned by comparing translations which are not even legal according to the grammar, but we persist so as to see what comes of it.

Once the six raw translations were created, I went through each one, changing it as needed to create legal translations. I kept the following notes about what errors I found and what changes I made to the raw translations to make them legal.

Amazon Customer

- The raw translation of statement 1 is not legal: *1. Users must create passwords with letters (unspec).* First, let us dispense with the rather trivial use of a period (.) to delimit the rule index rather than a colon (:) as required. I make this replacement everywhere without further comment about it.

  More interesting is the way the password statement violates the grammar. It is interesting because the translation is exactly the kind of ambiguous statement that the formal language tries to avoid. The phrase *create passwords* must be followed with one of 9 phrases that serve to clarify exactly what part of the password is to be regulated by the statement (e.g., *with a character* or *with all characters*). A character set such as *letters (unspec)*) cannot apply, vaguely, to the password. In the formal grammar, it must apply to a set of characters, and those characters must be specified. It is somewhat encouraging for the formal grammar that it would have disallowed a vague statement like this one. In order to create a legal translation,

I removed this statement, because there was no unambiguous way to correct it to make it legal.

- The raw translation of statement 2 was not legal, but by changing *create password* to *create passwords*, I was able to make it legal.
- The raw translations of statements 3a and 3b are legal, and I left them alone.
- The raw translation of statement 4 is not legal, having the same problem as statement 1. Character sets such as *numbers, punctuation, all other characters (unsepec)*[sic] must apply to a specific set of password characters, and they do not in this translation. I had to drop the statement.
- The raw translation of statement 5 is not legal, but by changing *create password* to *create passwords*, I made it so.
- After these changes and corrections, the policy was parsed successfully.

Ames Laboratory:

- The raw translations of statements 1–6 are legal, and I left them alone.
- The raw translation of statement 7 is not legal. By changing the phrase *with the substrings* to *with a substring*, I made it so.
- The raw translations of statements 8–10 are legal, and I left them alone. After that small correction to statement 7, the policy was parsed successfully.

Maui High Performance Computing Center:

- The raw translations of statements 1–4 are legal, and I left them alone.
- The raw translation of statement 5 has an asterisk at the end of it (associated with an explanation later in the Word document about what he was trying to express with the rule). After removing the asterisk, the statement parses successfully.
- The raw translation of statement 6 is not legal because *word or number patterns* is not a valid set name, but *word or number patterns (unspec)* is. With that change, the statement parses successfully.
- The raw translation of statement 7 is not legal because *strings with otherwise forbidden content in reverse* is not a valid set name. The valid set is simply *otherwise forbidden content in reverse*. After I deleted the extraneous *strings with*, the statement parses successfully.
- The raw translation of statement 8 is not legal for the same reason. Again, by deleting the extraneous *strings with*, the statement parses successfully.
- The raw translation of statement 9 is legal, and I left it alone.
- The raw translation of statement 10 is not legal, but by changing *uppercase* to *upper-case*, and *lowercase* to *lower-case*, it was made legal.
- The raw translation of statement 11 is legal, and I left it alone. With these changes and corrections, the translation parsed successfully.

National Institute of Standards and Technology:

- The raw translation of statement 1 is legal, and I left it alone.
- The raw translation of statement 2 is not legal. A specific character set requires that the characters be listed one right after the other with no spaces, words or punctuation in between. It also must end with a space (so that the period ending the statement is not confused with the character set). By changing *;!,$,%,; and *.* to *!$%* ., this statement parsed successfully.
- The raw translation of statement 3 is not legal. The concatenation of phrases *create passwords strings* is missing a set phrase, and so I changed it to *create passwords in the set of strings*. Additionally, *a character repeated more than 4 times* is not a legal phrase but is semantically equivalent to the legal phrase *a character repeated 5 or more times*. With these changes, the statement parsed successfully.
- The raw translations of statements 4–19b are legal, and I left them alone.
- There is only a partial raw translation of statement 20: *Users must (couldn't find)*. Since Michael wasn't able to translate the statement into the formal language, I removed the statement.
- The raw translations of statements 21a and 21b are legal, and I left them alone. With these changes and corrections, the statements all parsed successfully.

TD Bank Financial Group Customer:

- The raw translations of statements 1 and 2a are legal, and I left them alone.
- The raw translation of statement 2b is not legal because *greater than 8 characters* is not a legal character-length statement. It is semantically equivalent to the legal statement *greater than or equal to 9 characters*. I changed the statement, and it parsed successfully.
- The raw translation of statement 2c is not legal. The character sets (i.e., *numbers, letters (unspec)*) must appear as part of a statement about characters, not the password as a whole. Because it is not possible to unambiguously correct the statement to make it legal, I dropped it.
- After these changes, the translation parsed successfully.

USDA Biosafety Level-3 Facilities:

- The raw translations of statements 1a–1c are legal, and I left them alone.
- The raw translation of statement 1d is not legal. By changing *with substrings* to *with a substring* and *their user ID* to *the user ID*, I was able to make it legal.
- The raw translations of statement 2 is legal, and I left it alone.
- The raw translation of statement 3 is not legal. The phrase *store passwords* must be followed by either *in writing* or *online*, and they must be followed by a location name. The translation just has the location name (i.e., *in an insecure location*) without specifying whether the storage is online or written. Because there is no way to unambiguously decide whether the statement refers to written or online storage, I dropped the statement from the translation.

- The raw translation of statement 4 is legal, and I left it alone. With these changes and corrections, the translation parsed successfully.

The following observations stem from this analysis: (1) Simple little syntactic errors, like using plurals where the rules expect a singular and vice versa are easily fixed by requiring that the translations be actually parsed. (2) The most frequent big issue, which resulted in statements that had to be dropped, concerned how to specify different things about the characters of a password. It's a tricky subject, and the rules are written very carefully. It's possible that making translators parse their statements would call attention to these subtleties, but more instruction might help as well. For instance, the "practice policies" that we give them during training could be chosen to include examples of the different kinds of statements about the characters in a password.

C.2 Comparison of the Parsed Translations

Once both Michael's and my translations were made legal according to the rules of the formal language, I compared the two translations. I tried to match statements in one translation to statements in the other by rule index. In a few cases, where the rule indices did not match up but the statements did, I matched them by context.

The statements in both translations were compared and, based on the comparison, classified into one of six categories:

Same: The statements matched across translation, or disagreed only in the rule index.

Untranslated: The statements were not translated by one of the translators (or were ambiguously translated and thus dropped).

Missing: The statements were included by one translator but not by the other. Admittedly the difference between this class and the Untranslated class is subtle, but I wanted to separate statements that were hard to translate from statements that one translator saw and the other did not. The former group falls in the Untranslated class, while the latter in the Missing class. In general, if a whole rule number is missing, I classified it as Untranslated. If a rule has been translated by both translators, but into different subparts, I classified it as Missing. For instance, if one translation does not include a translation for rule 7, I mark it Untranslated, but if both translations include rule 7a and 7b but only one has 7c, I mark it as Missing.

Must/Should: The statements differ only in whether it is a *must* statement or a *should* statement.

Whole/Substring: The statements are *create passwords* statements that differ only in that one contains the phrase *with substrings* and the other applies to the whole password.

Other Disagreement: The statements differ in some other way, not covered by the other class definitions.

The comparison and classification was performed for each of the six replicated translations. The following tables present the findings, with aggregate statistics about the total number of statements and classifications at the end.

Amazon:

Kevin's Rule Index	Michael's Rule Index	Notes	Class
1		Michael's translation did not parse.	(Untranslated)
2	2		(Same)
	3a	Kevin didn't include a regulation about dictionary words.	(Missing)
3	3b		(Same)

Ames:

Kevin's Rule Index	Michael's Rule Index	Notes	Class
1	1	Kevin said must; Michael said should about a prohibition on the communication of passwords.	(Must/Should)
2	2		(Same)
3	3		(Same)
4	4		(Same)
5	5		(Same)
6	6		(Same)
7	7		(Same)
8	8	Kevin said substring; Michael said whole-password regarding the inclusion of dictionary words.	(Whole/Substring)
9	9		(Same)
	10	Kevin didn't include a translation for this rule (oops).	(Untranslated)

MHPCC:

Kevin's Rule Index	Michael's Rule Index	Notes	Class
1	1		(Same)
2	2	Kevin said must; Michael said should about a prohibition on dictionary words.	(Must/Should)
3	4	Aside from the rule index, the two are the same.	(Same)
4	3	Kevin said must not include addresses; Michael said should not include PII.	(Other Disagreement)
5	5	Kevin said a password must contain 2 characters; Michael said it must not contain 1 character repeated 8 or more times. Interestingly, our two statements are nearly equivalent, but his doesn't quite capture the border case of a password with 8 of one letter and 1 of another (which is okay by my reading of the policy).	(Other Disagreement)
6	6	Kevin said must; Michael said should in a prohibition on word or number patterns.	(Must/Should)
7a	7	Kevin said must; Michael said should regarding a prohibition on reversed forbidden content.	(Must/Should)
7b		Kevin included a rule about forbidden content concatenated; Michael did not.	(Missing)
8	8	Kevin said must; Michael said should about a prohibition on otherwise forbidden content with a number tacked on.	(Must/Should)
9	9		(Same)
10	10	Kevin said special characters were required; Michael said punctuation.	(Other Disagreement)
11	11		(Same)

NIST:

Kevin's Rule Index	Michael's Rule Index	Notes	Class
1	1		(Same)
2	2	Kevin said *a character in the set of numbers, special characters (unspec)*; Michael said *strings with characters from 1 of these 2 sets: numbers, these special characters: !$%*.* Interestingly, these are nearly equivalent. If *with substrings* were added to Michael's translation, the two probably would be functionally equivalent. To clear up the problem of having two different, functionally equivalent statements in the formal language, perhaps we should add a convention that a k-of-n pattern only be used when $k > 1$.	(Other Disagreement)
3	3		(Same)
4	4		(Same)
5a		Kevin included a prohibition on vendor-default passwords; Michael did not.	(Missing)
5b		Kevin included a prohibition on proper names; Michael did not.	(Missing)
5c	5a	Kevin said substring; Michael said whole password in a prohibition on dictionary words.	(Whole/Substring)
5d	5b	Kevin said substring; Michael said whole password in a prohibition on reversed dictionary words.	(Whole/Substring)
5e		Kevin included prohibition on birthdays; Michael did not.	(Missing)
5f		Kevin included prohibition on addresses; Michael did not.	(Missing)
5g		Kevin included prohibitions on word or number patterns; Michael did not.	(Missing)
	5c	Michael included prohibitions on personally identifying information; Kevin did not, though in retrospect it seems like Kevin's 5e–5g could easily be collapsed to Michael's 5c. Maybe a restriction on PII is the level of granularity we want, not an itemized list of kinds of PII.	(Missing)
	5d	Michael included prohibitions on passwords to other systems; Kevin did not.	(Missing)
6a	6		(Same)
6b		Kevin included a requirement about changing shared passwords; Michael did not.	(Missing)
7		Kevin included a rule about storing passwords securely; Michael did not translate the rule.	(Untranslated)
8		Kevin included a rule about storing passwords in online scripts; Michael did not translate the rule.	(Untranslated)
9		Kevin included a rule about encrypting passwords sent by email; Michael did not translate the rule.	(Untranslated)
10		Kevin included a rule about discussing passwords in voicemail; Michael did not translate the rule.	(Untranslated)
11		Kevin included a rule about separating user ID and password in communications; Michael did not translate the rule.	(Untranslated)

12	5d	Kevin said passwords must not match those to *an outside system*; Michael said passwords must not match those to *any other system*.	(Other Disagreement)
13a		Kevin said passwords should not be communicated locally without encryption; Michael did not translate the rule.	(Untranslated)
13b		Kevin said passwords must not be communicated over the Internet without encryption; Michael did not translate the rule.	(Untranslated)
14	14		(Same)
15	15		(Same)
16	16		(Same)
17	17		(Same)
18	18		(Same)
19	19b		(Same)
20	19a		(Same)
21	21a/b	Kevin included a failed-authentication consequence of *administrative unlock or a 3 minute lockout*; Michael included two separate rules, with the conjoined consequence of *administrative unlock* ***and*** *a lockout of unspecified duration.* Putting the disagreement over the duration of the lockout, the more interesting issue is the interpretation of multiple rules as an **and**, whereas a single rule allows a translator to express **or**. Perhaps this subtlety could be explained better.	(Other Disagreement)

TD Bank:

Kevin's Rule Index	Michael's Rule Index	Notes	Class
1	1		(Same)
2a	2a		(Same)
2b	2b		(Same)
2c		Kevin included a requirement that letters and numbers only be used; Michael did not.	(Missing)

USDA Biosafety:

Kevin's Rule Index	Michael's Rule Index	Notes	Class
1a	1a		(Same)
1b	1d	Kevin prohibited a password equal to a user ID; Michael prohibited a substring of a password equal to a user ID.	(Whole/Substring)
	1b	Michael prohibited proper nouns; Kevin did not.	(Missing)
	1c	Michael prohibited PII; Kevin did not.	(Missing)
2	2		(Same)
3		Kevin included a rule about storing passwords in writing in an insecure location; Michael's translation was dropped as unable to be parsed.	(Untranslated)
4	4		(Same)

Results Table:

	Amazon	Ames	MHPCC	NIST	TD Bank	USDA Biosafety	Total
Same	2	7	4	11	3	3	30
Untranslated	1	1		7		1	10
Missing	1		1	8	1	2	13
Must/Should		1	4				5
Whole/Substring		1		2		1	4
Other Disagreement			3	3			6
Totals	4	10	12	31	4	7	68