NIST 2017 Level 1 Worked example

The NIST 2017 [59] policy allows a number of different authentication methods. We focus on subscriber chosen memorized secrets (typically referred to as 'passwords') which are the most common form of authentication [67].

D.3.1 Policy summary

General authentication rules

- Periodic re-authentication of subscriber sessions SHALL be preformed every 30 days.
- Access controls in place.
  - Specified by NIST 800-53 [120].
- The verifier SHALL limit consecutive failed authentication attempts on a single account to no more than 100.
- Communication between the claimant and verifier SHALL be via an authenticated protected channel.
- A CSP SHOULD bind at least two physical authenticators to the subscriber's credentials.

Authenticator 1: Memorized secrets

- Memorized secrets SHALL be at least 8 characters in length if chosen by the subscriber.
- Verifiers SHOULD permit subscriber-chosen memorized secrets at least 64 characters in length.
- No other complexity requirements for memorized secrets SHOULD be imposed.
- All printing ASCII characters as well as the space character SHOULD be acceptable in memorized secrets.
- Unicode characters SHOULD be accepted.
  - The verifier SHOULD apply the Normalization Process for Stabilized Strings using either the NFKC or NFKD normalization
- Truncation of the secret SHALL NOT be performed.

- Memorized secret verifiers SHALL NOT permit the subscriber to store a "hint" that is accessible to an unauthenticated claimant.
- Verifiers SHALL NOT prompt subscribers to use specific types of information (e.g., "What was the name of your first pet?") when choosing memorized secrets.
- When processing requests to establish and change memorized secrets, verifiers SHALL compare the prospective secrets against a list that contains values known to be commonly-used, expected, or compromised. For example, the list MAY include, but is not limited to:
  - Passwords obtained from previous breach corpuses.
  - Dictionary words.
  - Repetitive or sequential characters (e.g. 'aaaaaa', '1234abcd').
  - Context-specific words, such as the name of the service, the username, and derivatives thereof.
- Verifiers SHOULD offer guidance to the subscriber, such as a password-strength meter, to assist the user in choosing a strong memorized secret.
- Verifiers SHOULD NOT require memorized secrets to be changed arbitrarily.
- The verifier SHOULD offer an option to display the secret - rather than a series of dots or asterisks - until it is entered.
- Verifiers SHALL store memorized secrets in a form that is resistant to offline attacks. Memorized secrets SHALL be salted and hashed using a suitable one-way key derivation function.
- The salt SHALL be at least 32 bits in length and be chosen arbitrarily so as to minimize salt value collisions among stored hashes. Both the salt value and the resulting hash SHALL be stored for each subscriber using a memorized secret authenticator.
- The iteration count for the PBKDF2 function SHOULD be as large as verification server performance will allow, typically at least 10,000 iterations.
- Verifiers SHOULD perform an additional iteration of a key derivation function using a salt value that is secret and known only to the verifier. The secret salt value[4] SHALL be stored separately from the hashed memorized secrets.

[4] Often known as a "pepper".

D.3.2 Benefits: NIST 2017 Level 1 policy

The expected benefit of the NIST 2017 Level 1 example, will be measured by the expected losses when no policy is in place minus the expected losses when the NIST 2017 Level 1 policy is in place.

In Section D.3.2.1 we will compute the probability of compromise, and the expected Loss due to compromise, when the NIST 2017 Level 1 policy is in place. In Section D.3.2.3 we complete the same computation for when there is no policy in place. Finally, in Section D.3.2.5 these two results are compared to give the overall security benefit offered by the NIST 2017 Level 1 policy.

D.3.2.1 Quantifying attacks: NIST 2017 Level 1

We will now discuss the impact of the NIST 2017 Level 1 advice on each of the 11 cost categories that we identified in Chapter 2.

Assertion Manufacture or Modification *The attacker generates a false assertion or the attacker modifies an existing assertion.*

From Figure D.1, in the 2017 Verizon's Data Breach Investigations Report (DBIR) [169] we find that that a breach due to the variety privilege abuse occurs 128 times. This value can be used as an estimate for an assertion being modified or manufactured. The probability of this occurring is mitigated by the access controls specified by NIST 800-53 [120]. To model this we account for the percentage of users who have access to the assertion mechanisms. We take this value as 10% of those who would have access if this advice was not in place.

$$\begin{aligned} \text{P[assertion manu or mod]} &= (0.1)\left(\frac{(128)(0.72)}{4788}\right) \\ &= 0.00192481203007365 \end{aligned} \tag{D.2}$$

Physical Theft *A physical authenticator is stolen by an Attacker.*

For the NIST 2017 Level 1 policy there is no physical authenticator that can be stolen. However a laptop or PC could be taken. Verizon DBIR reports 39 breaches due to theft. Reauthentication is required once every 30 days. If the user is only required to login every 30 days, then in a year time frame, a login will only be required approximately 12 times. Therefore, if an attacker steals a computer, there are only approximately 12 days in the year when the device will be locked. The probability the computer is unlocked when stolen is $1 - \frac{12}{365} = 0.96712387$.

If the disc is not encrypted, rebooting and rewriting the stored secret will compromise the authentication. NIST Level 1 does not specify encrypting the computer disc. This will require theft and tampering.

$$\begin{aligned}
&\mathrm{P[theft]} = \qquad \text{(D.3)}\\
&\mathrm{P[stolen]P[tampering]} + \mathrm{P[stolen]P[unlocked]}\\
&= \left(\frac{(39)(0.72)}{4788}\right)\left(\frac{27}{4788}\right) + \left(\frac{(39)(0.72)}{4788}\right)(0.96712387)\\
&= 0.00570492567549227
\end{aligned}$$

Duplication *The subscriber's authenticator has been copied with or without their knowledge.*

Duplication of the memorized secret can occur when a subscriber willingly divulges the secret or when it has been copied without their knowledge after being recorded electronically or physically. Shay et al. [149] found that over a quarter of their respondents reported they had shared a password. Because we need the person the password was divulged to be the same as the attacker, we introduce the Verizon reported statistic that the vector in 108 of 3231 breaches is a partner. While this likely implies a business partner, we can generalize to someone who trust was placed in. The attack vector in 16 breaches was the victim work area. This is likely an upper bound, however it does account for copying a written password, copying a password from an electronic file and gaining access if the password is saved to the victim's browser. There is nothing in the 2017 NIST policy that mitigates the duplication of the password.

$$
\begin{aligned}
&\text{P[duplication]} = \text{P[divulge password]P[partner exploits]} \qquad \text{(D.4)}\\
&+ \text{P[recorded password compromised]}\\
&= (0.25)\left(\frac{(108)(0.72)}{3231}\right) + \left(\frac{(16)(0.72)}{3231}\right)\\
&= 0.00958217270195177
\end{aligned}
$$

Eavesdropping *The authenticator secret or authenticator output is revealed to the attacker as the subscriber is authenticating. Or an out-of-band secret is intercepted by the attacker by compromising the communication channel.*

Eavesdropping can take the form of shoulder surfing, keylogging, pass-the-hash attack, or the secret being intercepted by an attacker as it travels over a compromised communication channel.

Verizon DBIR 2017 gives the frequency of a breach resulting from physical surveillance as 21 in 4788. We take this for the frequency of a shoulder surfing compromise. This is also helped by the NIST 2017 policy allowing a toggle for displaying the password at entry.

A pass-the-hash attack should not be possible under the NIST hash-salt method since the hash is not being passed over the channel.

The NIST 2017 Level 1 guidelines require the use of approved encryption and an authenticated protected channel when requesting memorized secrets. This means the channel is secure from eavesdropping unless the encryption is broken. As with all the pieces of advice, we assume that the organization follows it to the fullest degree. Therefore the probability that an attacker compromises the encryption is negligibly small.

In reality, the likelihood that an attacker can break the encryption to access the data transmitted is dependent on the type of encryption used. In 2017, the paper [89] showed that 28 administrators struggled to set up HTTPS. At the end of a two hour lab study they found that 18.5% of their participants failed to set up a secure HTTPS connection. However, services such as Let's Encrypt have made this set-up easier [92].

We were unable to find a statistic for how often eavesdropping attacks take place. This is likely a reflection of the difficulty/impossibility of detecting an eavesdropper.

The NIST 2017 Level 1 policy does not mitigate a keylogger attack and we can take these values as the frequency reported in DBIR [169].

$$\begin{aligned}
&\text{P[eavesdropping]} = \\
&\text{P[physical surveillance]} + \text{P[keylogger breach]} \\
&= \left(\frac{(21)(0.72)}{4788}\right) + \left(\frac{(595)(0.72)}{4788}\right) \\
&= 0.0926315789473664
\end{aligned} \tag{D.5}$$

Offline Guessing attacks *The authenticator is exposed using analytical methods outside the authentication mechanism. E.g. A trial and error guessing attack against an offline dataset of passwords.*

For offline guessing to take place a dataset or subset of encrypted passwords must be leaked. In the case of the NIST 2017 Level 1 policy, the passwords will be individually hashed and salted before being stored and then an additional iteration of a key derivation function with a separately stored salt (often called a “pepper”) will take place against the whole dataset.

This means offline guessing for this policy can only occur if both the password database and the separately stored salt (or pepper) is leaked [99].

$$\begin{aligned}
&\text{P[user password cracked offline]} \\
&= \text{P[passwords leaked} \cup \text{pepper leaked]P[user password cracked]}
\end{aligned} \tag{D.6}$$

We saw already, in Section 5.4.1.1, that our benefits model has been adjusted to take into account the dependence of offline guessing on the dataset already being leaked before cracking can occur. But now we include the requirement for the pepper to also be leaked before cracking can occur.

It is worth noting that there are situations when brute force guessing will still be feasible even if the pepper is not leaked [40]. However, in these scenarios it is likely that other aspects of the NIST 2017 Level 1 requirements have not been adhered to. For example, salting individual passwords and not storing hints.

$$\begin{aligned}
&\mathbb{E}[\text{Benefits}] = \\
&\mathbb{E}[\text{Benefits}|(\text{no password leak} \cup \text{no salt leak})]\mathbb{P}[\text{no leak} \cup \text{no salt}] \\
&+ \mathbb{E}[\text{Benefits}|(\text{password leak} \cap \text{salt leak})]\mathbb{P}[(\text{password leak} \cap \text{salt leak})]
\end{aligned} \tag{D.7}$$

The 2017 Verizon's Data Breach Investigations Report [169] states that, from 4788 breaches, 49 breaches resulted in the capture of stored data.

We need two separately stored data files to be leaked. However we only need a breach to occur once:

$$\begin{aligned}
&\text{P[database of passwords is leaked} \cap \text{salt leaked]} \\
&= \left(\frac{(49)(0.72)}{4788}\right)\left(\frac{49}{4788}\right) \\
&= 0.00007540781779234\text{66}
\end{aligned} \tag{D.8}$$

We can see that the Level 1 NIST 2017 policy makes offline guessing very difficult for an attacker. Given that both the password dataset and the salt have been leaked, we look for the probability a password in that database is cracked.

Because the passwords are also individually salted, some of the attackers' computing power is consumed by combining each guess with all the possible salts. Therefore instead of the usual 10^{13} guesses we would allow an attacker, this time the attacker will make $10^{13}/\#\textit{unique salts}$. NIST 2017 requires that the salts are randomly generated for each user. Therefore $\#\textit{unique salts} = \#\textit{users}$.

There have been many suggestions by researchers for ways to model password guessing; Entropy [20] which has now been disproved [83], Zipf law [98] [171] [143] and Loss analysis in comparison to optimal rates [116].

However, the best way to determine how susceptible your password set is to cracking is to attempt password cracking or an equivalent such as Kelley at al.'s guess-number calculator [83]. Kelley et al.'s paper "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms", gives the result for the number of guesses required to crack passwords created under a variety of password composition policies. For example, for their BlacklistHard policy they required their study participants to create a password which contained "at least 8 characters," did "not contain a dictionary word" and did not exist in "a five-billion-word dictionary created using the algorithm outlined by Weir et al. [178]". One of their guessing experiments showed that they could guess 23% of the passwords in this dataset in 10^{13} guesses. Only 5.5% were guessed in 10^{10} guesses. We use this value as the probability an offline guessing attack successfully guesses a password since the NIST 2017 level 1 policy requires passwords to be created using similar constraints.

$$\begin{aligned}
&\text{P[user password is guessed in } (10^{13}/\#\textit{unique salts}) \text{ guesses]} \\
&= \text{P[user password is guessed in } \ 10^{10} \text{ guesses]} \\
&= 0.055
\end{aligned} \tag{D.9}$$

Side Channel Attack *The authenticator secret is exposed using physical characteristics of the authenticator.*

While side channel attacks are generally designed against cryptographic keys, there is at least one attack which will work against password [158]. The attack involves changing one character at a time to one that requires a longer processing time. Thus when the password is typed by the user the attacker can identify whether that character occurred in the password. There is some probability that the user will detect the changes occurring as their password will be rejected if the changed key does exist in their password. We set the probability that the attack goes ahead undetected at 0.8. The attacker will need to force the user to download malware onto their machine.

We presume that this attack would be targeted at an individual. Hackmageddon [128] finds that 15.2% of attacks are targeted. Of those 22.3% target individuals. Giving our overall probability of a user being targeted as: 3.3896%.

$$\begin{aligned}&\text{P[side channel attack]} = \qquad\qquad (\text{D.10})\\ &\text{P[targeted attack].P[not detected].P[malware]}\\ &= (0.033896)(0.8)\left(\frac{(33)(0.72)}{3231}\right) = 0.000199410451251986\end{aligned}$$

Phishing or Pharming *The authenticator output is captured by fooling the subscriber into thinking the attacker is a verifier or relying party.*

The NIST 2017 level 1 policy offers no mitigation for phishing or pharming, and therefore we consult the Verizon DBIR documents for the frequency breaches from such attacks occurring. We use the statistic provided for phishing as there is none available for pharming.

$$\begin{aligned}&\text{P[phishing} \cup \text{pharming]} = \qquad\qquad (\text{D.11})\\ &\text{P[phishing]} = \frac{(653)(0.72)}{4788} = 0.0981954887218051\end{aligned}$$

Social Engineering *The attacker establishes a level of trust with a subscriber in order to convince the subscriber to reveal their authenticator secret or authenticator output.*

The mitigation suggested by the NIST 2017 guidelines is to avoid using authenticators that present a risk of social engineering of third parties. Memorized secrets do not fall into this category and we have no mitigation at level 1 against social engineering. The Verizon DBIR reports that 39 breaches are of the variety social pretexting. That is, a person presents a false motive for requiring information.

$$\text{P[social engineering]} = \text{P[pretexting]} = \frac{(39)(0.72)}{4788} = 0.00586466165413429 \qquad (\text{D.12})$$

Online Guessing attacks *The attacker connects to the verifier online and attempts to guess a valid authenticator output in the context of that verifier.*

E.g. An attacker performs repeated login trials by guessing possible values for the password and username.

For online guessing, the attacker does not need to have access to the password dataset. However, a knowledge of both the username and password is necessary.

Therefore for an online guessing attack an attacker can either target individuals whose username they know or can easily find out, and then guess their password. Or alternatively, they can attempt to guess both the username and the password of any person.

In both cases online guessing is hindered by rate limiting or throttling. An organization can increase the time allowed between guessing attempts or stop all attempts after a threshold number of wrong guesses. The NIST 2017 guidelines allows at most 100 consecutive incorrect attempts. The counter will reset to zero every time there is a successful login. We assume a user will login once per working day so the counter will reset when this happens. This means in 1 year an attacker can try 99*261 guesses against a user's account. Kelley et al. show the percentage of passwords guessed in $99 * 261 = 25839$ guesses is approximately 1% [83].

Because the limit on wrong guesses is attached to a subscriber's account, an attacker can try many wrong usernames and they will not be tallied. Therefore we calculate the probability of guessing a username in 10^{13} guesses. In reality we hope that an organizations' administrator could prohibit such a large number of username guesses by blocking the IP address of the attacker, but this mechanism is not stated in the NIST 2017 policy. Because there are no blacklisting or composition requirements on the usernames for the NIST 2017 level 1 policy, we use Kelley et al.'s [83] guessing return for their Basic8survey password. A study of the strength of usernames could be an interesting future research project. 63% of these passwords could be guessed in 10^{13} guesses.

We would like to know the probability that each of these two types of online guessing attacks take place. place. To estimate the probability of an attacker targeting a user whose username they know we use the Hackmageddon statistics from 2017 [128]. These tell us that an individual has a probability of 0.033896 of having a targeted attack against them. Verizon DBIR tells us that breaches due to brute force or use of stolen credentials occur 73 and 631 times respectively.

$$\begin{aligned} &\text{P[online guessing successful]} = \\ &\text{P[targeted attack].P[password in (99*261) guesses]} \\ &+ \text{P[brute force or using stolen creds]P[(username guessed in } 10^{13} \text{ guesses]} \\ &* \text{P[password guessed in (99*261)]} \\ &= (0.033896)\,(0.01)(0.72) + \left(\frac{(73+631)(0.72)}{4788}\right)(0.63)(0.01) \\ &= 0.000910998568420505 \end{aligned} \tag{D.13}$$

Endpoint compromise *Malicious code on the endpoint proxies remote access to a connected authenticator without the subscriber's consent or causes authentication to other than the intended verifier or compromises a multi-factor software cryptographic authenticator.*

The NIST Level 1 guidelines do not include mitigations for endpoint compromise. Therefore, finding a frequency in the Verizon report we use the statistic for the number of breaches which leverage a backdoor or C2 servers. C2, command and control servers are used by attackers to maintain communications with compromised systems within a target network.

$$\begin{aligned} \text{P[endpoint compromise]} &= \text{P[breach using backdoor or C2]} \\ &= \frac{(678)(0.72)}{4788} = 0.101954887218044 \end{aligned} \tag{D.14}$$

Unauthorized binding *An attacker is able to cause an authenticator under their control to be bound to a subscriber's account.*

The unauthorized binding of the authenticator to an attacker could take the form of an unauthorized password reset. In order for an attacker to successfully reset a subscriber's password they would need to already have access to their email account and be interested in targeting that individual since it is a time consuming attack. Alternatively they could reset a password by having physical access to the unencrypted computer disc and rebooting and resetting the password. We assume this would also have to be a targeted attack.

$$\begin{aligned}
&\mathrm{P}[\text{unauthorized binding}] = \Big(\mathrm{P}[\text{use stolen credentials}] \cup \mathrm{P}[\text{physical theft}]\Big) \qquad \text{(D.15)}\\
&* \mathrm{P}[\text{targeted attack}]\\
&= \left(\frac{(631)(0.72)}{4788} + \frac{(39)(0.72)}{4788}\right)(0.033896) = 0.00341508571428736
\end{aligned}$$

D.3.2.2 Calculation of NIST 2017 Level 1 Losses

Recall the following equation for computing the expected loss from password attacks:

$$\begin{aligned}
\mathbb{E}[\text{Loss}] = &\left(\mathbb{P}\left[\mathcal{N}\left(p_l, \frac{p_l(1-p_l)}{N}\right) > \alpha\right] L_{\text{system}} + Np_lL_1\right)\mathbb{P}[\text{leak}] \qquad \text{(D.16)}\\
+ &\left(\mathbb{P}\left[\mathcal{N}\left(p_{l'}, \frac{p_{l'}(1-p_{l'})}{N}\right) > \alpha\right] L_{\text{system}} + Np_{l'}L_1\right)\mathbb{P}[\text{leak}']
\end{aligned}$$

where $p_l = \mathbb{P}[\text{compromise}|\text{leak}]$, $p_{l'} = \mathbb{P}[\text{compromise}|\text{no leak}]$ and $\mathbb{P}[\text{leak}'] = \mathbb{P}[\text{no leak}]$.

So we calculate the probability of compromise separately for when a leak has occurred and when a leak has not occurred. P[offline guess] will equal zero for $p_{l'}$ but will equal 0.055 in p_l (Equation D.9). The probability of compromise is given in terms of each attack type:

$$\begin{aligned}
p &= 1 - \prod_a (1 - P_a)\\
&= 1 - (1 - \mathrm{P}[\text{assertion manu}])(1 - \mathrm{P}[\text{theft}])(1 - \mathrm{P}[\text{dup}]) \qquad \text{(D.17)}\\
&\quad (1 - \mathrm{P}[\text{eavesdrop}])(1 - \mathrm{P}[\text{offline guess}])(1 - \mathrm{P}[\text{side channel}])\\
&\quad (1 - \mathrm{P}[\text{phishing}])(1 - \mathrm{P}[\text{social eng}])(1 - \mathrm{P}[\text{online guess}])\\
&\quad (1 - \mathrm{P}[\text{endpoint}])(1 - \mathrm{P}[\text{unauth bind}])
\end{aligned}$$

For the NIST 2017 Level 1 policy this computation with and without a leak occurring is:

$$p_l = \mathbb{P}[\text{compromise}|\text{leak}] = 0.324538796275133 \tag{D.18}$$

$$p_{l'} = \mathbb{P}[\text{compromise}|\text{no leak}] = 0.285226239444585 \tag{D.19}$$

Now, taking the values defined in Section D.1: $Lsystem = \$10^7$, $L_1 = \$166$, and $N = 500$ users. We calculate the expected loss due to compromise for the organization when the NIST 2017 Level 1 policy is in place.

$$\mathbb{E}[\text{Loss}|\text{NIST 2017 L1 policy}] = \tag{D.20}$$

$$\begin{aligned}
&\Bigg(\mathbb{P}\left[\mathcal{N}\left(0.324538796275133, \frac{(0.324538796275133)(1-0.324538796275133)}{500}\right) > 0.5\right] \times \$10^7 \\
&+ (500)(0.324538796275133)(\$166)\Bigg) \times 0.0000754078177923466 \\
&+ \Bigg(\mathbb{P}\left[\mathcal{N}\left(0.285226239444585, \frac{(0.285226239444585)(1-0.285226239444585)}{500}\right) > 0.5\right] \times \$10^7 \\
&+ (500)(0.285226239444585)(\$166)\Bigg) \times (1-0.0000754078177923466) \\
&= 2.10466041986588 + 24527.6078713633 = 24529.7125317831
\end{aligned}$$

D.3.2.3 Quantifying attacks: No policy

In order to determine the 'benefit' of a password policy, we compare the losses with the policy in place to losses that would occur without the policy.

Thus for each type of attack the authentication policy mitigates, we require a statistic indicating how probable this attack would have been without the policy. We will be using the DBIR [169] statistics as the baseline values for each attack type. Below are the baseline values for each attack type. Explanations are offered where they were not covered in Section D.3.2.1.

Assertion Manufacture or Modification

$$\text{P[assertion manufacture or modification]} = \left(\frac{(128)(0.72)}{4788}\right) = 0.0192481203007521 \tag{D.21}$$

Physical Theft

$$\text{P[theft]} = \frac{(39)(0.72)}{4788} = 0.00586466165413429 \tag{D.22}$$

Duplication

$$\begin{aligned}\text{P[duplication]} &= \text{P[divulge.password]} * \text{P[partner.exploits]} \\ &+ \text{P[recorded.password.compromised]} \\ &= (0.25)\left(\frac{(108)(0.72)}{3231}\right) + \left(\frac{(16)(0.72)}{3231}\right) = 0.00958217270195177\end{aligned} \tag{D.23}$$

Eavesdropping Eavesdropping can take the form of shoulder surfing, keylogging, skimming, pass-the-hash attack, or the secret being intercepted by an attacker as it travels over a compromised communication channel.

$$\begin{aligned}\text{P[eavesdropping]} &= \text{P[network eavesdropping breach]} \\ &+ \text{P[keylogger breach]} + \text{P[physical surveillance breach]} \\ &= \left(\frac{(118)(0.72)}{3231}\right) + \left(\frac{(595)(0.72)}{4788}\right) + \left(\frac{(21)(0.72)}{4788}\right) = 0.118926843571323\end{aligned} \tag{D.24}$$

Offline Guessing attacks

$$\text{P[database of passwords is leaked]} = \left(\frac{(49)(0.72)}{4788}\right) = 0.00736842105263307 \tag{D.25}$$

With no policy in place the passwords will be stored in plaintext. Therefore once the password set has been leaked, the passwords will be visible to all.

$$\text{P[user password is guessed in } 10^{13} \text{ guesses]} = 1 \tag{D.26}$$

Side Channel Attack

$$\text{P[side channel attack]} = \text{P[targeted attack].P[not detected].P[malware]} \tag{D.27}$$

$$= (0.033896)(0.8)\left(\frac{(33)(0.72)}{3231}\right) = 0.000199410451253493 \tag{D.28}$$

Phishing or Pharming

$$\text{P[phishing} \cap \text{pharming]} = \text{P[phishing]} = \frac{(653)(0.72)}{4788} = 0.0981954887218045 \tag{D.29}$$

Social Engineering

$$\text{P[social engineering]} = \text{P[pretexting]} = \frac{(39)(0.72)}{4788} = 0.00586466165413534 \tag{D.30}$$

Online Guessing attacks It is reasonable to assume that any password with less than 8 characters can easily be guessed in an exhaustive search. In the Yahoo.com password set [15] (had no password composition restrictions) 22.428% of the passwords contained less than 8 characters. After 8 characters we know that 63% of the passwords can be guessed in 10^{13} [83]. This indicates that when no policy is in place, an attacker should be able to guess at least 71.29836% of the users' passwords in a year. Given that there are no restrictions on the username we use this same value to estimate the probability it is guessed by an attacker.

$$\begin{aligned}
&\text{P[online guessing successful]} = \qquad \text{(D.31)}\\
&\text{P[targeted attack].P[password guessed in } 10^{13} \text{ guesses]}\\
&+ \text{P[brute force or using stolen creds].P[username guessed in } 10^{13} \text{ guesses]}\\
&* \text{P[password guessed in } 10^{13}]\\
&= (0.033896)(0.7129836)(0.72) + \left(\frac{(73+631)(0.72)}{4788}\right)(0.7129836)(0.7129836)\\
&= 0.0712162867316309
\end{aligned}$$

Endpoint compromise

$$\text{P[endpoint compromise]} = \frac{(678)(0.72)}{4788} = 0.101954887218044 \tag{D.32}$$

Unauthorized binding

$$
\begin{aligned}
&\text{P[unauthorized binding]} = && \text{(D.33)}\\
&\Big(\text{P[use stolen credentials]} \cup \text{P[physical theft]}\Big)\text{P[targeted attack]}\\
&= \left(\frac{(631)(0.72)}{4788} + \frac{(39)(0.72)}{4788}\right)(0.033896) = 0.00341508571428736
\end{aligned}
$$

D.3.2.4 Calculation of Losses with No policy

$$p_l = \mathbb{P}[\text{compromise}|\text{leak}] = 1 \tag{D.34}$$

$$p_{l'} = \mathbb{P}[\text{compromise}|\text{no leak}] = 0.366081483597361 \tag{D.35}$$

$$
\begin{aligned}
&\mathbb{E}[\text{Loss}|\text{No policy}] = && \text{(D.36)}\\
&\left(\mathbb{P}\left[\mathcal{N}\left(1, \frac{(1)(1-1)}{500}\right) > 0.5\right] \times \$10^7 + (500)(1)(\$166)\right) \times 0.00736842105263307\\
&+ \left(\mathbb{P}\left[\mathcal{N}\left(0.366081483597361, \frac{(0.366081483597361)(1-0.366081483597361)}{500}\right) > 0.5\right]\right.\\
&\left.\times \$10^7 + (500)(0.366081483597361)(\$166)\right) \times (1 - 0.366081483597361)\\
&= \$74317.89 + \$31254.01 = \$105571.91
\end{aligned}
$$

D.3.2.5 Quantifying benefit of the NIST 2017 Level 1 policy

The expected benefit of the NIST 2017 Level 1 policy is measured by the expected losses when no policy is in place minus the expected losses when the NIST 2017 Level 1 policy is in place.

$$
\begin{aligned}
\mathbb{E}[\text{Benefits}] &= \mathbb{E}[\text{Loss without policy}] - \mathbb{E}[\text{Loss with policy}] && \text{(D.37)}\\
&= \$105571.90 - \$24529.71 && \text{(D.38)}\\
&= \$81042.19 && \text{(D.39)}
\end{aligned}
$$

This tells us that the overall benefit for this organization of implementing the NIST 2017 Level 1 policy is $81042.19.

D.3.3 Costs: NIST 2017 Level 1 policy

Table 5.3 defines the twelve categories of costs. To quantify each equation we must input values which will depend on the individual organization and the individual pieces of advice. As mentioned, the lowercase items in these equations are variables dependent on the advice and the capitals are constants known or set by the organization. We identify each cost, probability and number of repetitions with $\{i, j\}$ where i denotes the cost category $1 \dots 12$ and j denotes the piece of advice the cost belongs to.

In this section we will quantify the cost associated with each piece of advice in the NIST 2017 Level 1 policy. We use input from our user administrator surveys described in Section 2.6 as well as our own insights and analysis.

In order to keep the survey concise, we allowed respondents to *optionally* include the rationale behind their assigned costs. This means that in some cases we do not have a clear quantifiable idea of what the costs are. We therefore include an arbitrary scaling for the assigned costs where we do not know the exact values. These are described below. In all other cases we are able to give more accurate quantifications.

Minor user education costs that are periodic we assign as 1 hour of administrative time every 6 months and 15 minutes of every users' time every 6 months. For Major user education costs this becomes 1 hour every month for the organisation and 15 minutes every month for each users. Minor organisation time to implement is assigned as 1 day, Major is three weeks (120 hours), except in cases when we know it to be longer or shorter. Finally, minor help desk time is assigned as 15 minutes per user per year while major help desk time is assigned as 15 minutes per user every 3 months. These values were formulated using input from system administrators, developers and log data from our university's implementation of multi-factor authentication as well as from feedback we received from the administrator surveys (Section 2). These values can be inputted into the quantified model directly according to Table 2.5. Where more insightful values can be provided we have detailed these below.

j=1: Composition Enforcing these composition restrictions will take minor $C_{4,1}$*: Organizations' time to implement.* According to the above rules we assign the task 1 day. This time may involve the time to approve the use of the system and establish an effective blocklist tailored to the organisation. This cost will occur with probability 1 and will be repeated once in time frame T, $R_{4,1} = 1$.

Administrators in our study agreed that this advice does not require additional computing power for organisations. However they did say it would require user education. On the Organisation side of the table we assign 1 hour every 6 months for the organisation to provide this user education and on the User side in $C_{12,1}$ we assign 15 minutes every 6 months for the users to receive this information.

We would like to know how the probability of *forgetting* is affected by different composition policies. We could find no literature identifying the distribution of the probability of forgetting. We would like to know whether the probability is correlated with the guessability of the password. Some researchers have described forgettability in terms of the entropy of a password [148]. Komanduri et al. record the number of participants in each of their password composition groups who forgot their passwords [87]. However they found no significant difference in the rates across different conditions. They did find that participants in the stricter composition groups wrote down their passwords more frequently than those in the other groups. We believe the correlation between forgettability and strength would be an interesting study for researchers and would be very beneficial for further understanding user behavior.

For the purpose of our example we will use statistics sent to us by Saranga Komanduri [85, 86]. The probability that a user forgets a password that was chosen according to the rules of the BlacklistHard Policy is $P_{7,1} = 0.02044025$.

We are interested in the probability that the advice *makes it more difficult for a user to create a password.* Ur et al. found that of their 49 study participants, 43 (88%) said they had a well-defined process for creating passwords [164] and similarly in our user study, 90% of participants said that this advice would make it more difficult for them to create a password.

The probability of a conflict occurring can be estimated using a leaked or past password dataset:

$$P_8 = \frac{\text{Number of passwords which do not abide by the policy}}{\text{Total number of passwords in database}} \tag{D.40}$$

Or alternatively

$$P_8 = \text{average number of password rejections at creation} \tag{D.41}$$

This second equation can give a value greater than 1 but while this makes it no longer a probability it does give a much better estimate for the inconvenience to the user.

The likelihood that these composition rules *makes it more difficult for a user to create a password* can also be taken from the statistics for Komanduri's BlacklistHard policy. They report users making an average of 0.98 attempts before successfully creating a password which satisfied the BlacklistHard restrictions. $P_{8,1} = 0.98$. The time to choose a new password under this composition policy is 85.4 seconds; $C_{8,1}(85.4)$. We also include the basic cost *need to pick a new password* since a new password will need to be chosen as a result of a new policy being put in place; $C_{9,1}(85.4)$. This is repeated once for each user.

The help desk cost of a user forgetting their password is covered under the cost of forgetting function. Recall the cost function for forgetting is:

$$\begin{aligned} C_7() :& \text{Cost of forgetting} = \text{P[abandon|forget]} * (\text{\$ORG PROFIT PER USER}) \\ &+ C_1(\text{TIME ADMIN RESET}) + C_{11}(\text{TIME ADMIN RESET}) \end{aligned}$$

j=2: Input The costs associated with allowing the full input types by the subscriber falls with the organization authenticating. The system must be able to accept all ASCII and Unicode characters. In our administrator survey, administrators were asked about the costs of allowing all ASCII characters. Administrators said there was a minor cost associated with this. In order to accept both ASCII and UNICODE characters we expect there to be a major implementation cost. Accepting all UNICODE characters introduce complications because different keyboards in different locales will potentially give you different encoding. What works on one computer may not work on another computer. However, its implementation is certainly possible and involves applying a normalization process for stabilized strings. NIST suggests using either the NFKC or NFKD normalization [59].

Password hint Not allowing a hint allows for the same forgetability normally associated with a password policy; $P_{7,1} = 0.02044025$. However, this is already accounted for in $j = 1$ *composition rules* so there are no other costs associated with this piece of advice.

Specific knowledge based questions Using specific information questions for authentication allow for a high success rate for both brute force guessing and targeted attacks. The information required to answer the questions can often be found very easily through online searches or can be known by a peer. There is no real cost associated with not allowing these question-answer authentication methods since an alternative, email or other, can still be used for password recovery. Therefore we leave it out of our calculations.

j=3: Password strength meter Ur et al. [163] present research relating to study participants' reaction to password meters. They found that "the majority of participants who saw the most stringent meters changed their mind partway into password creation, erasing what they had typed and creating a different password". From Table 3 in Ur's paper we find that the percentage of participants who changed their password while entering it when there was no meter was 14.4% and the number of participants who changed their password when there was a meter was an average of 32% over all the different stringency password meters. The difference of $32\% - 14\% = 18\%$ is what we take as the probability a password meter "inconveniences a user's personal system for password creation", $P_{8,3} = 0.18$, though, notably, in the study, participants voluntarily chose to change their password. Ur et al. also report that between 27% and 40% of participants in the four stringent password meter conditions found the meter annoying. This is in comparison to 13% of the lenient/baseline meter participants. We use the low 13% as the annoyance of using a password meter, $P_{11,3} = 0.13$. To assign a cost to the inconvenience of the password meter we take the difference between the average time taken by those using the password meter ($33.6s$) and those creating passwords with no meter ($19.9s$). $C_{11,3} = 33.6s - 19.9s = 13.7seconds$. Including a password strength meter will take organisation time to implement and will also involve user education.

j=4: Toggle to display entered secret Administrators in our survey indicated that there was minor cost to the organisation involved to implement this. We therefore assign it 1 day for set-up, $C_{4,4}$(1 day). It also involved user education. This toggle helps users who would have made a typo in their password on entry. Therefore we take the costs to be 'negative' to show that the advice is offering an improvement in these areas. To measure how likely a typo is we take Komanduri's statistics [85] which record an average of 0.08 confirmation failures for participants using the BlacklistHard policy. So when the secret is able to be toggled visible a user no longer needs to wait for the authentication process to reject the password before realizing it must have been typed wrong. So the probability is $P_{2,5}$=$P_{3,5}$=0.08 per login that a user saves 3 seconds of authentication time $-C_{2,5}(3)$.

j=5: Hash and salt passwords Administrators said this was a minor implementation cost. We have assigned 1 day of organizations time to setting up a system which will hash and salt a password according to the requirements of the NIST 2017 policy. There is an additional 2 seconds of computing time needed for the organization during the authentication procedure.

The disadvantage of hashing and salting passwords over using reversible encryption is that if the password is forgotten it must be created anew. The user cannot be simply reminded of their old password as the organization should not have access to it. We therefore determine that a user will have to choose a new password with the same probability that they forget their password with $P_{9,5}$ = the probability of forgetting = 0.02044. The cost $C_9(85.4)$ will therefore occur with probability $P_{9,5}$ for each user.

Password expiry Not requiring expiry has no costs associated with it. If expiry was enforced multiple costs would need to be reoccurring. For example, all costs associated with composition of the password would reoccur every time the password needs to be recreated.

j=6: Throttling The policy states that:

- The verifier SHALL limit consecutive failed authentication attempts on a single account to no more than 100.

While throttling protects against online guessing it means that an adversary can intentionally lock users out of their accounts [51]. The probability of an adversary seeking to leverage throttling to conduct a denial of service attack depends on the organization. We were unable to find any statistics on its occurrence despite it being an easily measurable attack. We suspect this type of attack would very rarely be used to target an organization, but it is easily used against an individual. To calculate the probability of this we look at the probability of a breach due to a brute force guessing attack [169] and the probability of a targeted attack [128]. The probability affects each user.

$$P_{2,4} = \mathrm{P}[\text{targeted attack}\cup \text{ brute force attacker}] = \tag{D.42}$$
$$(0.033896)\left(\frac{(73)(0.72)}{4788}\right) = 0.00037209142$$

Though it is not stated in the policy, we assume the lockout time at 100 consecutive incorrect guesses is 12 hours or until an administrator can unlock the account. We calculate the cost as 12 hours of the users' time $C_{11,6}$(12 hours), and assign the minor help desk, user education and implementation costs as specified by respondents in our administrator survey (Section 2).

j=7: Access to password files should be restricted In our administrator survey we found that this advice incurred minor help desk costs periodically, a small amount of user education periodically and that it is a minor cost to implement.

j=8: Access controls should be applied to access to particular features or systems System wide access controls incur major help desk/user support time and minor user education. It also takes major periodic time to implement. we see this as an initial major set up cost and then a minor per user cost so that each user can be set up with the access controls applicable to them.

Administrators assigned a major help desk cost to users and users assigned a minor increase in their time or effort. We therefore assume these two are related and add in a 15 minute user time or effort for each user as a result of this policy.

In Table D.2 we reference this information using a second set of rows as we have multiple inputs to some cost categories.

j=9: Two physical authenticators The policy states that:

- The CSP SHALL bind at least one, and SHOULD bind at least two, physical (something you have) authenticators to the subscriber's online identity, in addition to a memorized secret or one or more biometrics.
  - While all identifying information is self-asserted at Level 1, preservation of online material or an online reputation makes it undesirable to lose control of an account due to the loss of an authenticator. The second authenticator makes it possible to securely recover from an authenticator loss. For this reason, a CSP SHOULD bind at least two physical authenticators to the subscriber's credential at Level 1 as well.

At creation, either a user will need to provide the physical authenticators or the organisation will need to. We assume the organisation provides the devices. We assign 15 minutes of the organization's time per user to distribute the two physical authenticators. We also estimate each authenticator will cost the organization $6 and the postage of these authenticators will cost $0.50.

We estimate the probability a user loses one of the authenticators as 0.01. In this instance the organization will need to send a replacement.

We also assign a minor user education for teaching users the purpose of these physical authenticators. There is a help desk cost should a user lose an authenticator.

j=10: Authenticated protected channel Setting up an authenticated channel could at one time have high costs for the organization in both time and money. Now free and efficient certificates and pre-written code is available to make the task accessible to any organization or administrator. Administrators marked this as a minor cost under the advice statement "don't transmit in cleartext". We assign 1 day for the implementation. Administrators also marked it as having user education and help desk time.

j=11: Re-authentication This requirement affects the amount of times each user must authenticate. We have generally assumed that the user will login daily. Here however we assign a cost to the number of mandatory logins. The user must re-authenticate every 30 days. So in a 1 year time frame, the number of mandatory logins is:

$$\#logins = \left(\frac{T}{re\text{-}authentication\ requirement}\right)(\#users) = \frac{365}{30} = 12 \qquad \text{(D.43)}$$

We assign 20 minutes organization time to automate this. Using Komanduri's statistics we estimate that the authentication process takes 31.85 seconds for the Blacklist Hard policy.

D.3.3.1 Calculation of NIST 2017 Level 1 costs

$$\mathbb{E}[Costs] = \sum_{\substack{i=12\\j=11}} P_{i,j}.C_{i,j}.R_{i,j} \qquad \text{(D.44)}$$

Table D.2 shows the costs associated with each of the $j = 11$ pieces of advice in NIST 2017 Level 1 policy.

Figure D.2: Costs of implementing the NIST 2017 Level 1 password advice.

		Organisation costs					User costs						
		$C_1(t)$: Increased help desk/user support time	$C_2(t_o)$: User education provided	$C_3(r_o)$: Additional organization resources	$C_4(t_o)$: Organization time taken to implement	$C_5(tc_o)$: Increased organization computing power needed	$C_6(t_u)$: Makes it more difficult to create a password	C_7: Increased risk of forgetting	$C_8(r_u)$: Additional user resources needed	$C_9(t_u)$: Need to pick a new password	$C_{10}(tc_u)$: Increased user computing power need	$C_{11}(t_u)$: User time and inconvenience	$C_{12}(t_u)$: User education time required
j=1: Composition	$p(i,1)$		1		1		0.98	0.02044		1			1
	$C(i,1)$		C_2(1hr)		C_4(1 day)		C_6(85.4 secs)	C_7		C_9(85.4 secs)			C_{12}(15mins)
	$R(i,1)$		2		1		#users	#users		#users			2*#users
j=2: Input	$p(i,2)$	1	1		1								1
	$C(i,2)$	C_1(15mins)	C_2(1hr)		C_4(3 week)								C_{12}(15mins)
	$R(i,2)$	#users	2		1								2*#users
j=3:Password strength meter	$p(i,3)$		1		1		0.18					0.13	1
	$C(i,3)$		C_2(1hr)		C_4(1 day)		C_6(85.4 secs)					C_{11}(13.7 secs)	C_{12}(15mins)
	$R(i,3)$		2		1		#users					#users	2*#users
j=4: Toggle to display entered secret	$p(i,4)$		1		1							0.08	1
	$C(i,4)$		C_2(1hr)		C_4(1 day)							$-C_{11}$(3 secs)	C_{12}(15mins)
	$R(i,4)$		2		1							#logins	2*#users
j=5: Hash and salt passwords	$p(i,5)$				1	1				0.02044			
	$C(i,5)$				C_4(1 day)	C_5(2 secs)				C_9(85.4 secs)			
	$R(i,5)$				1	#logins				#users			
j=6: Throttling	$p(i,6)$	1	1		1							0.000071576	1
	$C(i,6)$	C_1(15mins)	C_2(1hr)		C_4(1 day)							C_{11}(12hrs)	C_{12}(15mins)
	$R(i,6)$	#users	2		1							#users	2*#users
j=7: Access to password files	$p(i,7)$	1	1		1								
	$C(i,7)$	C_1(15mins)	C_2(1hr)		C_4(1 day)								
	$R(i,7)$	#users	2		1								
j=8: Access controls applied to features & systems	$p(i,8)$	1	1		1							1	1
	$C(i,8)$	C_1(15mins)	C_2(1hr)		C_4(3 weeks)							C_{11}(15 mins)	C_{12}(15mins)
	$R(i,8)$	4*#users	2		1							#users	2*#users
j=8: continued	$p(i,8)$				1								
	$C(i,8)$				C_4(15mins)								
	$R(i,8)$				#users								
j=9: Two physical authenticators	$p(i,9)$	0.01	1	1	1							0.01	
	$C(i,9)$	C_1(15mins)	C_2(1hr)	C_3(\$12.50)	C_4(15mins)							C_{11}(15 mins)	
	$R(i,9)$	#users	2	#users	#users							#users	
j=9 continued	$p(i,9)$			0.01	0.01								
	$C(i,9)$			C_3(\$6.50)	C_4(15mins)								
	$R(i,9)$			#users	#users								
j=10: Authenticated protected channel	$p(i,7)$	1	1		1								1
	$C(i,7)$	C_1(15mins)	C_2(1hr)		C_4(1 day)								C_{12}(15mins)
	$R(i,7)$	#users	12		1								12*#users
j=11: Reauthentication	$p(i,8)$				1							1	
	$C(i,8)$				C_4(1 day)							C_{11}(31.85 secs)	
	$R(i,8)$				1							12*(#users)	

We take the constant values defined in Section D.2 for the organization and first fill these into the cost categories. For example, \$ADMIN WAGES = $\$18/(60*60) = \$0.005/second$.

The cost category equations then become:
$C_1(t_0) = supportTimeTaken * 0.005$
$C_2(t_o) = EducatorTime * 0.005$
$C_3(r_o) = \$orgResources$
$C_4(t_o) = timeToImplement * 0.005$
$C_5(t_{c_o}) = orgCompTime * 0.01$
$C_6(timeChoosePwd) = (10/100)*15 + C_{11}(timeChoosePwd) + (15/100)*C_7()$
$C_7() = (10/100) * 15 + C_1(900) + C_{11}(900)$
$C_8(r_u) = 1 * \$userResources$
$C_9(timeChoosePwd) = (20/100)*15 + C_{11}(timeChoosePwd) + C_5(2) + (20/100)* C_7()$
$C_{10}(t_{c_u}) = 1 * userCompTime * 0.01$
$C_{11}(t_u) = 1 * userTime * 0.005$
$C_{12}(t_u) = 1 * EducationTime * 0.005$

Now for each piece of advice j we can fill the input values from the table into these cost categories. Finally, we multiply each $C_{i,j}$ by the corresponding $P_{i,j}$ and $R_{i,j}$.

$$
\begin{aligned}
\mathbb{E}[Costs]_{j=1} &= ((1)(60 \ast 60 \ast 0.005)(2)) + ((1)(60 \ast 60 \ast 8)(0.005)(1)) \\
&\quad + ((0.98)((10/100 \ast 15) + (1 \ast 85.4 \ast 0.005) + (15/100)((10/100) \ast 15 + (900 \ast 0.005) \\
&\quad + (1 \ast 900 \ast 0.005)))(500)) + ((0.02044)((10/100) \ast 15 + (900 \ast 0.005) \\
&\quad + (1 \ast 900 \ast 0.005))(500)) + ((1)((20/100) \ast 15 + (1 \ast 85.4 \ast 0.005) + (2 \ast 0.01) \\
&\quad + (20/100) \ast ((10/100) \ast 15 + 900 \ast 0.005 + (1 \ast 900 \ast 0.005)))(500)) \\
&\quad + ((1)(1 \ast 900 \ast 0.005)(2 \ast 500)) = 9276.79 \\
\mathbb{E}[Costs]_{j=2} &= ((1)(900 \ast 0.005)(500)) + ((1)(60 \ast 60 \ast 0.005)(2)) \\
&\quad + (1 \ast (60 \ast 60 \ast 8 \ast 5 \ast 3 \ast 0.005) \ast 1) + ((1)(1 \ast 900 \ast 0.005)(2 \ast 500)) = 8946 \\
\mathbb{E}[Costs]_{j=3} &= ((1)(60 \ast 60 \ast 0.005)(2)) + ((1)(60 \ast 60 \ast 8)(0.005)(1)) \\
&\quad + ((0.18)((10/100) \ast 15 + (1 \ast 85.4 \ast 0.005) + (15/100)((10/100) \ast 15 + (900 \ast 0.005) \\
&\quad + (1 \ast 900 \ast 0.005)))(500)) + ((0.13)(1 \ast 13.7 \ast 0.005) \ast 500) \\
&\quad + ((1)(1 \ast 900 \ast 0.005)(2 \ast 500)) = 4999.6325 \\
\mathbb{E}[Costs]_{j=4} &= ((1)(60 \ast 60 \ast 0.005)(2)) + ((1)(60 \ast 60 \ast 8)(0.005)(1)) \\
&\quad + ((0.08)(1 \ast 3 \ast 0.005) \ast 130500) + ((1)(1 \ast 900 \ast 0.005)(2 \ast 500)) = 4836.6 \\
\mathbb{E}[Costs]_{j=5} &= ((1)(60 \ast 60 \ast 8)(0.005)(1)) + ((1)(2 \ast 0.01)(130500)) + ((0.02044)((20/100) \ast (15) \\
&\quad + (1)(85.4)(0.005) + ((2)(0.01)) + (20/100) \ast ((10/100) \ast 15 + 900 \ast 0.005 \\
&\quad + (1 \ast 900 \ast 0.005)))(500)) = 2810.69034 \\
\mathbb{E}[Costs]_{j=6} &= ((1)(900 \ast 0.005)(1)) + ((1)(60 \ast 60 \ast 0.005)(2)) + ((1)(60 \ast 60 \ast 8)(0.005)(1)) \\
&\quad + ((0.000071576)(1 \ast 60 \ast 60 \ast 12 \ast 0.005)(500)) + ((1)(1 \ast 900 \ast 0.005)(2 \ast 500)) \\
&\quad = 6937.730208 \\
\mathbb{E}[Costs]_{j=7} &= ((1)(900 \ast 0.005)(1)) + ((1)(60 \ast 60 \ast 0.005)(2)) + ((1)(60 \ast 60 \ast 8)(0.005)(1)) \\
&\quad = 2430 \\
\mathbb{E}[Costs]_{j=8} &= ((1)(900 \ast 0.005)(4 \ast 500)) + ((1)(60 \ast 60 \ast 0.005)(2)) \\
&\quad + (1 \ast (60 \ast 60 \ast 8 \ast 5 \ast 3 \ast 0.005) \ast 1) + (1(1 \ast 900 \ast 0.005) \ast 500) \\
&\quad + ((1)(1 \ast 900 \ast 0.005)(2 \ast 500)) + ((1)(900 \ast 0.005)(500)) = 20196 \\
\mathbb{E}[Costs]_{j=9} &= ((0.01)(900 \ast 0.005)(500)) + ((1)(60 \ast 60 \ast 0.005)(2)) \\
&\quad + ((1)(12.5)(500)) + ((1)(900 \ast 0.005)(500)) + ((0.01)(1 \ast 900 \ast 0.005)(500)) \\
&\quad + ((0.1)(6.5)(500)) + ((0.01)(1 \ast 900 \ast 0.005)(500)) = 8636 \\
\mathbb{E}[Costs]_{j=10} &= ((1)(900 \ast 0.005)(1)) + ((1)(60 \ast 60 \ast 0.005)(12)) + ((1)(60 \ast 60 \ast 8)(0.005)(1)) \\
&\quad + ((1)(1 \ast 900 \ast 0.005)(12 \ast 500)) = 29610 \\
\mathbb{E}[Costs]_{j=11} &= ((1)(60 \ast 60 \ast 8)(0.005)(1)) + ((1)(1 \ast 31.85 \ast 0.005)(500)) = 1099.5
\end{aligned}
$$

$$\mathbb{E}[Costs] = \sum_{\substack{i=12\\j=11}} P_{i,j}.C_{i,j}.R_{i,j} = \$99{,}778.94 \tag{D.45}$$

D.3.4 Value of the NIST 2017 Level 1 policy

$$\mathbb{E}[Value] = \mathbb{E}[Benefits] - \mathbb{E}[Costs] \tag{D.46}$$

$$\mathbb{E}[Value] = \$81{,}042.19 - \$99{,}778.94 = -\$18{,}736.75 \tag{D.47}$$

NIST 2003 Level 1 Worked example

We now look at the NIST 2003 Level 1 policy and highlight the costs and benefits of this policy. We will not repeat our explanations already described for the NIST 2017 policy so this will be a briefer account.

One of the suggestions by the NIST 2003 policy [20] for authentication is a challenge response protocol. This involves a subscriber creating a password and sending it to the organization. When the subscriber wants to login the organization sends them a challenge, for example ‘xY253N’. The subscriber then hashes their password with this value, $\mathcal{H}[challenge, password]$, and sends it back to the verifier. The verifier hashes their stored value of the password with the same challenge. If the results match the subscriber is successfully authenticated. This is the method we will consider when looking at their authentication rules.

D.3.5 Policy summary

General authentication rules

- Plaintext passwords or secrets shall not be transmitted across a network.
- This level does not require cryptographic methods that block offline analysis by eavesdroppers.
- There is no requirement at this level to use approved cryptographic techniques.
- Long term shared secrets may be revealed to verifiers.

- There is no stipulation about the revocation or lifetime of credentials at Level 1.
- Files of shared secrets used by verifiers shall be protected by discretionary access controls that limit access to administrators and only those applications that require access.

Authenticator: Memorized secrets

- There are no min-entropy requirements for Level 1.
- The probability of success of a targeted online password guessing attack by an attacker who has no a priori knowledge of the password, but knows the user name of the target, shall not exceed 2^{-10} (1 in 1024), over the life of the password.
  - Lock the password for 1 minute after 3 incorrect guesses.
- Shared secret files shall not contain the plaintext password.
  - Typically they contain a one-way hash or “inversion” of the password.

D.3.6 Benefits: NIST 2003 Level 1 policy

Benefits, as mentioned, are affected by the probability of successful attack. Looking at the pieces of advice from the NIST 2003 policy we can see which of the identified attacks the policy protects against.

D.3.6.1 Quantifying attacks: NIST 2003 Level 1

Assertion Manufacture or Modification

- Files of shared secrets used by verifiers shall be protected by discretionary access controls that limit access to administrators and only those applications that require access.

Discretionary access controls limit the number of people who have the authority to manipulate assertions. If 10% of the people in an organization have access to the authentication mechanism then we model this by taking this fraction of the probability of a breach.

$$\begin{aligned}\text{P[assertion manufacture or modification]} &= \left(\frac{(128)(0.72)}{4788}\right)(0.1) \qquad \text{(D.48)}\\ &= 0.00192481203007365\end{aligned}$$

Physical theft No protection

$$\text{P[theft]} = \frac{(39)(0.72)}{4788} = 0.00586466165413429 \qquad \text{(D.49)}$$

Duplication No protection

$$\begin{aligned}\text{P[duplication]} &= \text{P[divulge password]*P[partner exploits]} \qquad \text{(D.50)}\\ &+ \text{P[recorded password compromised]}\\ &= (0.25)\left(\frac{(108)(0.72)}{3231}\right) + \left(\frac{(16)(0.72)}{3231}\right)\\ &= 0.00958217270195177\end{aligned}$$

Eavesdropping

- Plaintext passwords or secrets shall not be transmitted across a network.
- This level does not require cryptographic methods that block offline analysis by eavesdroppers.
- There is no requirement at this level to use approved cryptographic techniques.
- Long term shared secrets may be revealed to verifiers.

The NIST 2003 policy recommends a challenge response protocol whereby the challenge is sent to the subscriber, the subscriber hashes their password with this challenge ($\mathcal{H}$[secret,challenge]) and sends the hashed version back to the verifier. The verifier/organization hashes their stored value of the users' password with the same challenge and verifies that both hashes match. An eavesdropper on the connection can; see the plaintext password when it is sent to the subscriber at creation, or decrypt the hash sent back to the verifier by the user.

We were unable to find a statistic for how often eavesdropping attacks take place. This is likely a reflection of the difficulty/impossibility of detecting an eavesdropper. For lack of a better option we take the probability of a remote eavesdropper attack as the statistic in the Verizon DBIR for breaches from the vector LAN access.

To find the probability an attacker sees the plaintext password sent to the verifier at creation we say that of the number of logins by a user in time frame T, one is at creation. So the probability, given an eavesdropper is on a users' connection, that they are on the creation connection is 1/#logins. For our organization, the #logins per user in time frame T is one per working day; #logins = 261. So the probability of an eavesdropper on the line at creation is: P[eavesdropper sees plaintext at creation] = P[eavesdropper(stat:LAN access)].P[connection is at creation] = P[eavesdropper(stat:LAN access)](1/261).

The probability an attacker decrypts the hash sent back to the verifier by the user is the probability of an offline guessing attack being successful by the probability the password was eavesdropped. We estimate that 72.07408% of the passwords can be guessed in 10^{13} guesses (described further below).

A shoulder surfing eavesdropping attack is slightly more difficult for the NIST 2003 policy than the NIST 2017 policy. Because a password can not be toggled as visible, a physical eavesdropper must be able to see and record the keys typed by the user on their keyboard. For this reason, we include physical access to the victim's work area as a requirement for a physical eavesdropping attack. Keylogger attacks are still not mitigated. A pass-the-hash attack should still not be possible.

$$
\begin{aligned}
&\text{P[eavesdropping]} = \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(D.51)}\\
&\text{P[eavesdropping attempted(stat:LAN access)].P[connection is at creation]}\\
&+\text{P[eavesdropping attempted(stat:LAN access)].P[offline guessing]}\\
&+\text{P[keylogger breach]} + \text{P[physical surveillance breach} \cup \text{access to victim work area]}\\
&= \left(\frac{(118)(0.72)}{3231}\right)\left(\frac{1}{261} + 0.64\right) + \left(\frac{(595)(0.72)}{4788}\right) + \left(\frac{(21)(0.72)}{4788}\right)\left(\frac{16}{4788}\right)\\
&= 0.106413954407714
\end{aligned}
\tag{D.52}
$$

Offline guessing attack

- Shared secret files shall not contain the plaintext password.
  - Typically they contain a one-way hash or “inversion” of the password.
- There are no min-entropy requirements for Level 1.

For offline guessing to take place first a dataset of passwords must be leaked. This time there is no requirement for a global salt so this does not also need to have been leaked.

$$\text{P[database of passwords is leaked]} = \frac{(49)(0.72)}{4788} = 0.00736842105258628 \tag{D.53}$$

The passwords in this case are stored as a one-way hash or “inversion” and can therefore be reversed with a key. Attackers can either steal the key with the dataset, or brute force guess the key. Brute forcing the key should only be possible if the key space is small. In some cases, such as the Adobe leak [40] even when passwords are cracked, the key is never discovered. Because the NIST policy sets no requirements on the size of the key we take the probability it is compromised as 0.01 [64].

The passwords can be guessed directly using brute force especially since salting is not enabled. There are no composition or minimum entropy requirements on the password. As with the no policy set up (Section D.3.2.3), we assume passwords less than 8 characters can be brute force guesses and for those above 8 characters, we take Kelley et al.'s value for percentage of passwords cracked when they were made using their basic8survey password rules. This gives 72.07408% of passwords guessed in 10^{13} guesses.

$$\begin{aligned}&\text{P[user password revealed]} = \\ &\text{P[key leaked]} + \text{P[key brute forced]} + \text{P[user password in } 10^{13} \text{ guesses]} \\ &= \frac{49}{4788} + 0.01 + 0.7207408 = 0.740974718128655\end{aligned} \tag{D.54}$$

Side Channel Attack No protection.

As we discussed in NIST 2017 Level 1, there are side channel attacks which exist for compromising passwords.

$$\begin{aligned}\text{P[side channel attack]} =& \text{P[targeted attack].P[not detected].P[malware]} \\ =&(0.033896)(0.8)\left(\frac{(33)(0.72)}{3231}\right) = 0.000199410451251986\end{aligned} \tag{D.55}$$

Phishing or Pharming No protection

$$\text{P[phishing} \cup \text{pharming]} = \text{P[phishing]} = \frac{(653)(0.72)}{4788} = 0.0981954887218051 \tag{D.56}$$

Social Engineering No protection

$$\text{P[social engineering]} = \text{P[pretexting]} = \frac{(39)(0.72)}{4788} = 0.00586466165413429 \tag{D.57}$$

Online guessing

- There are no min-entropy requirements for Level 1.

- Lock the password for 1 minute after 3 incorrect guesses.

Locking the account for 1 minute after 3 incorrect guesses means that an attacker can make no more than $525600 * 3 = 1576800 \approx 10^6$ guesses in the year time frame.

There is no minimum entropy requirement for Level 1 passwords. In the NIST 2003 policy, this means that there is no minimum length. Again, we assume passwords less than 8 characters long can easily be brute force guessed (Section D.3.2.3).

We use [83] to estimate the probability of guessing the users' password eight characters and above in length in 1576800 guesses. We take their basic8survey composed passwords, of which 14% are guessed in 1576800 guesses. Combined with the simplicity of guessing the shorter passwords, this gives a probability of compromise of 0.3328808.

Because the passwords are not salted, once a password is cracked all other users' who have used the same password will also be compromised.

$$\text{P[online guessing successful]} = \text{P[targeted attack].P[password in 1576800 guesses]} \tag{D.58}$$

$$+ \text{P[(username guessed in } 10^{13} \text{ guesses]P[password guessed in 1576800]}$$

$$= (0.033896)(0.3328808)(0.72) + \left(\frac{(73+631)(0.72)}{4788}\right)(0.7207408)(0.3328808)$$

$$= 0.0335231274432347$$

Endpoint Compromise No protection

$$\text{P[endpoint compromise]} \tag{D.59}$$

$$= \text{P[breach using backdoor or C2]} = \frac{(678)(0.72)}{4788} = 0.101954887218044$$

Unauthorized Binding No protection

$$
\begin{aligned}
&\mathrm{P}[\text{unauthorized binding}] = && \text{(D.60)}\\
&\Big(\mathrm{P}[\text{use stolen credentials}] \cup \mathrm{P}[\text{physical theft}]\Big)\mathrm{P}[\text{targeted attack}]\\
&= \left(\frac{(631)(0.72)}{4788} + \frac{(39)(0.72)}{4788}\right)(0.033896) = 0.00341508571428736
\end{aligned}
$$

D.3.6.2 Quantifying benefit of the NIST 2003 Level 1 policy

We quantify benefits by splitting the equation to look at the scenario when the password dataset has been leaked and when it has not (Equation D.16). For the NIST 2003 Level 1 policy this gives the following probabilities of compromise:

$$p_l = \mathbb{P}[\text{compromise}|\text{leak}] = 0.823647757902002 \tag{D.61}$$

$$p_{l'} = \mathbb{P}[\text{compromise}|\text{no leak}] = 0.319169770518423 \tag{D.62}$$

Now, taking the values defined in Section D.1: $Lsystem = \$10^7$, $L_1 = \$166$, and $N = 500$ users. We calculate the expected loss due to compromise for the organization when the NIST 2003 Level 1 policy is in place.

$$
\begin{aligned}
&\mathbb{E}[\text{Loss}|\text{NIST 2003 L1 policy}] = && \text{(D.63)}\\
&\left(\mathbb{P}\left[\mathcal{N}\left(0.8236477579, \frac{(0.8236477579)(1-0.8236477579)}{500}\right) > 0.5\right] \times \$10^7\right.\\
&\left. + (500)(0.8236477579)(\$166)\right) \times 0.001417397660819\\
&+ \left(\mathbb{P}\left[\mathcal{N}\left(0.3191697705, \frac{(0.3191697705)(1-0.3191697705)}{500}\right) > 0.5\right] \times \$10^7\right.\\
&\left. + (500)(0.3191697705)(\$166)\right) \times (1 - 0.001417397660819)\\
&= \$74206.1431050767 + \$27246.3697187516 = \$101452.512823828
\end{aligned}
$$

Taking the value for expected losses with no policy from Section D.3.2.3, our computation of the NIST 2003 L1 benefits is:

$$\begin{aligned}\mathbb{E}[\text{Benefits}] &= \mathbb{E}[\text{Loss}|\text{No policy}] - \mathbb{E}[\text{Loss}|\text{NIST 2003 L1 policy}] \\ &= \$105571.91 - \$101452.51 = \$4119.39\end{aligned} \tag{D.64}$$

D.3.7 Costs: NIST 2003 Level 1 policy

In this section we will quantify the costs associated with each piece of advice in the NIST 2003 policy

Composition

- There are no min-entropy requirements for Level 1.

The NIST 2003 policy has no composition requirements and therefore incurs no costs relating to it.

j=1: Throttling

- Lock out the claimant for a minute after three successive failed authentication attempts.

One scheme suggested by the NIST 2003 Level 1 policy is to lock out the claimant for a minute after three successive failed authentication attempts.

Brostoff and Sasse [18] show that 31% of their study participants who would have been able to login with unlimited number of attempts, failed to login successfully within 3 attempts.

Brostoff and Sasse also found that approximately 7% of these failed logins led to password reminder requests: $P_{(7,1)} = 0.31 * 0.07 = 0.0217$

Because it only requires 3 guesses, both a brute force attacker or a peer could also lock the account.

$$\begin{aligned} P_{2,1} &= \text{P[targeted.attack]}\,(\text{P[brute force.attacker]} + \text{P[partner]}) \\ &= (0.033896)\left(\frac{(73)(0.72)}{4788} + \frac{(108)(0.72)}{3231}\right) = 0.00118786145 \end{aligned} \tag{D.65}$$

The 60 second inconvenience to the user will occur whether an outsider or the user themselves have exceeded the 3 guess limit: $0.31 + 0.00118786145 = 0.31118786145$.

Administrators in our survey also indicated that it required minor held desk time, user education and time to implement.

Password expiry

- There is no stipulation about the revocation or lifetime of credentials at Level 1.

Not requiring expiry has no costs associated with it.

j=2: Password storage

- Shared secret files shall not contain the plaintext password.
  - Typically they contain a one-way hash or "inversion" of the password.

We have assigned 1 day of organizations time to setting up a system which will create a one-way hash or "inversion" for each user password. There is an additional 0.5 seconds of computing time needed for the organization to store the password "inversion" at creation.

j=3: Authenticated protected channel

- Plaintext passwords or secrets shall not be transmitted across a network.

The NIST 2003 level 1 policy recommends the use of a challenge-response protocol. The password is hashed with the challenge before being sent across the network. This requires an additional 0.5 second of user computing time to preform the hash on the password and challenge. It will take 1 second of the organizations computing time to generate a unique challenge for the authentication and hash this with their stored users' password. We assign 1 day to the organization to set this process up and also include the need for user education and help desk time as per the "Don't transmit in cleartext" advice (Table 2.5).

Cryptographic methods

- This level does not require cryptographic methods that block offline analysis by eavesdroppers.
- There is no requirement at this level to use approved cryptographic techniques.

These incur no user or administrator direct costs.

Shared secrets revealed to verifier

- Long term shared secrets may be revealed to verifiers.

In the challenge-response protocol this refers to when the secret is initially created by the user, it may be sent to the verifier for their storage. This is a key component of the challenge-response system and while it has security impacts it incurs no additional costs.

j=4: Access to password files

- Files of shared secrets used by verifiers shall be protected by discretionary access controls that limit access to administrators and only those applications that require access.

D.3.7.1 Calculation of NIST 2003 Level 1 costs

Table D.3 shows the costs associated with pieces of advice in NIST 2003 Level 1.

Figure D.3: Costs of implementing the NIST 2003 Level 1 password advice.

		Organisation costs					User costs						
		$C_1(t_o)$: Increased help desk/user support time	$C_2(t_o, t_u)$: User education required	$C_3(r_o)$: Additional organization resources	$C_4(t_o)$: Organization time taken to implement	$C_5(tc_o)$: Increased organization computing power needed	$C_6(t_u)$: Makes it more difficult to create a password	C_7: Increased risk of forgetting	$C_8(r_u)$: Additional user resources needed	$C_9(t_u)$: Need to pick a new password	$C_{10}(tc_u)$: Increased user computing power need	$C_{11}(t_u)$: User time and inconvenience	$C_{12}(t_u)$: User education time required
j=1: Throttling	*p(i,1)*		1		1			0.0217				0.31	1
	C(i,1)		C_2(1hr)		C_4(1 day)			C_7				C_{11}(60 secs)	C_{12}(15mins)
	R(i,1)		2		1			#users				#logins	2*#users
j=2: Password storage	*p(i,2)*				1	1							
	C(i,2)				C_4(1 day)	C_5(0.5 secs)							
	R(i,2)				1	#users							
j=3: Authenticated protected channel	*p(i,3)*	1	1		1	1	0.18				1	0.13	1
	C(i,3)	C_1(15mins)	C_2(1hr)		C_4(1 day)	C_5(1 secs)	C_6(85.4)				C_{10}(0.5)	C_{11}(13.7)	C_{12}(15mins)
	R(i,3)	1	2		1	#logins	#users				#logins	#users	2*#users
j=4: Access to password files	*p(i,4)*	1	1		1								
	C(i,4)	C_1(15mins)	C_2(1hr)		C_4(1 day)								
	R(i,4)	#users	2		1								

Using this table and the values defined in Section Section D.2 we have the following results for the cost of the NIST 2003 L1 policy:

$$\mathbb{E}[Costs] = \sum_{\substack{i=12\\j=4}} P_{c_{i,j}}.C_{i,j}.R_{i,j} = \$26{,}468.56 \tag{D.66}$$

D.3.8 Value of the NIST 2003 Level 1 policy

$$\mathbb{E}[Value] = \mathbb{E}[Benefits] - \mathbb{E}[Costs] \tag{D.67}$$

$$\mathbb{E}[Value] = \$4{,}119.39 - \$26{,}468.56 = -\$22{,}349.16 \tag{D.68}$$

NIST 2017 Level 3 Worked example

Level 3 is the highest authentication assurance level in the NIST 2017 policy [59]. It requires multi-factor authentication. To satisfy Level 3 we have chosen to authenticate using a Single-Factor Cryptographic device plus a memorized secret. A single factor cryptographic device is a hardware device that performs cryptographic operations using protected cryptographic key(s) and provides the authenticator output via direct connection with the user endpoint. NIST describes the authenticator as operating by signing a challenge nonce presented through a direct computer interface (e.g. a USB port). For our evaluation we assume such a device will take the form of a USB.

The rules for using a memorized secret for authentication are the same as for the NIST 2017 Level 1 policy (Section D.3.0.4). Therefore we will list the additional rules which apply to the use of the single-factor cryptographic device.

D.3.9 Policy summary

General authentication rules

- Periodic re-authentication of subscriber sessions SHALL be preformed after 12 hours or 15 minutes inactivity; SHALL use both authentication factors.

- Communication between the claimant and verifier SHALL be via an authenticated protected channel.
- Access controls in place.
  - Specified by NIST 800-53 [120].
- Store the expected authenticator output in hashed form.
- A verifier impersonation resistant authentication protocol SHALL establish an authenticated protected channel with the verifier.
- The CSP SHALL bind at least one, and SHOULD bind at least two, physical (something you have) authenticators to the subscriber's online identity, in addition to a memorized secret or one or more biometrics.
  - Binding of multiple authenticators is preferred in order to recover from the loss or theft of the subscriber's primary authenticator.
  - One example of a verifier impersonation resistant authentication protocol is client authenticated TLS, because the client signs the authenticator output along with earlier messages from the protocol that are unique to the particular TLS connection being negotiated.

Authenticator 1: Memorized secrets

- Same as NIST 2017 Level 1 policy (Section D.3.0.4)

Authenticator 2: Single factor cryptographic device

- The single factor cryptographic device authenticators encapsulate one or more secret keys unique to the device that SHALL NOT be exportable (i.e., cannot be removed from the device).
- The challenge nonce SHALL be at least 64 bits in length, and SHALL either be unique over the authenticator's lifetime or statistically unique.
- Tamper detection and response should be in place for covers and doors [48].
- Single-factor cryptographic device authenticators SHOULD require a physical input (e.g., the pressing of a button) in order to operate.
- The secret key and its algorithm SHALL provide at least the minimum security length specified in the latest revision of SP 800-131A (112 bits as of the date of this publication).

- Use a cryptographic authenticator that requires the verifier store a public key corresponding to a private key held by the authenticator.

D.3.10 Benefits: NIST 2017 Level 3 policy

The NIST 2017 Level 3 policy requires two factor authentication and Client authenticated TLS. Therefore the probability of successfully compromising a user is marked by the probability of compromising the single factor cryptographic device, the memorized secret and the client certificate and private key. A compromise, for example, could result from: the password (pwd) compromised via social engineering, the cryptographic device (USB) stolen and the certificate and private key ($cTLS$) gained via malware. To model this, we quantify the probabilities of compromise separately for each of the two factors and also for the client authenticated TLS.

$$p_{pwd} = 1 - \prod_a (1 - \mathrm{P}_{pwd,a}) \tag{D.69}$$

$$p_{USB} = 1 - \prod_a (1 - \mathrm{P}_{USB,a}) \tag{D.70}$$

$$p_{cTLS} = 1 - \prod_a (1 - \mathrm{P}_{cTLS,a}) \tag{D.71}$$

where p is the probability of the $pwd/USB/cTLS$ being compromised and P_a is the probability of compromise due to each individual attack type a.

Once we have calculated p_{pwd}, p_{USB} and p_{cTLS} we can calculate the probability a user is compromised, p, by:

$$p = p_{pwd} * p_{USB} * p_{cTLS} \tag{D.72}$$

As for the previous policies, we must again calculate this p when there has been a leak of the password file and when there has not: $\mathbb{P}$[compromise|no leak] and $\mathbb{P}$[compromise|database leak]

D.3.10.1 Quantifying attacks: NIST 2017 Level 3

Assertion Manufacture or Modification Assertion manufacture or modification can only for implemented by a privileged user. These, in our example, allow access to 10% of the administrators. The risk of this attack occurring is mitigated for all three factors by having access controls in place.

$$\begin{aligned}&\mathrm{P}_{pwd,cTLS,USB}[\text{assertion manufacture or modification}] = \\ &\left(\frac{(128)(0.72)}{4788}\right)(0.1) = 0.00192481203007519\end{aligned} \tag{D.73}$$

Theft

- Periodic re-authentication of subscriber sessions SHALL be preformed after 12 hours or 15 minutes inactivity; SHALL use both authentication factors.

If the computer is unlocked when it is stolen then the certificate and private key will be compromised. Given a typical 8 hour working day and a 5 day working week, the computer will lock at lunch time and at the end of the working day. Meaning that the computer is locked for approximately 16.75 hours 5 days a week and all day during the weekend. The probability that an attacker stealing a laptop finds it unlocked is $1 - \left((\frac{5}{7})(\frac{16.75}{24}) + \frac{2}{7}\right) = 0.21577380952$.

$$\begin{aligned}&\mathrm{P}_{pwd,cTLS}[\text{theft}] = \\ &\mathrm{P}[\text{stolen}]*\mathrm{P}[\text{tampering}] + \mathrm{P}[\text{stolen}]*\mathrm{P}[\text{unlocked}] \\ &= \left(\frac{(39)(0.72)}{4788}\right)\left(\frac{27}{4788}\right) + \left(\frac{(39)(0.72)}{4788}\right)(0.21577380952) \\ &= 0.0012985117869626\end{aligned} \tag{D.74}$$

The USB is susceptible to theft.

$$\mathrm{P}_{USB}[\text{theft}] = \frac{(39)(0.72)}{4788} = 0.00586466165413534 \tag{D.75}$$

Duplication

The Secret key on the USB is not exportable.

$$\text{P}_{USB}[\text{duplication}] = 0 \tag{D.76}$$

Password is only duplicable if revealed.

$$\text{P}_{pwd}[\text{duplication}] = \text{P}[\text{divulge password}]*\text{P}[\text{partner exploits}] \tag{D.77}$$

$$+ \text{P}[\text{recorded password compromised}]$$

$$= (0.25)\left(\frac{(108)(0.72)}{3231}\right) + \left(\frac{(16)(0.72)}{3231}\right) = 0.00958217270194984$$

Theft of the certificate's private key must be a targeted attack as each individual user will have their own stored private key and certificate. We assume that if the private key is captured then the certificate will also be taken with little extra effort. The value for a targeted attack comes from [128].

$$\text{P}_{cTLS}[\text{duplication}] = \text{P}[\text{targeted attack}].\text{P}[\text{private key} \cap \text{certificate leaked}] \tag{D.78}$$

$$= 0.033896\left(\frac{(49)(0.72)}{4788}\right) = 0.00024976$$

Eavesdropping

The communication of the password over the network is encrypted.

$$\text{P}_{pwd}[\text{eavesdropping}] = \text{P}[\text{physical surveillance}] + \text{P}[\text{keylogger breach}] \tag{D.79}$$

$$= \left(\frac{(21)(0.72)}{4788}\right) + \left(\frac{(595)(0.72)}{4788}\right)$$

$$= 0.0926315789473684$$

The USB cryptographic device and client authenticated TLS are not susceptible to eavesdropping attacks. They use public private key pairs and therefore the private key is never transmitted.

$$\text{P}_{USB,cTLS}[\text{eavesdropping}] = 0 \tag{D.80}$$

Offline Guessing attacks The password hash and salt can be cracked offline if it is leaked by the organization. The USB and client authenticated TLS public keys stored on the organizations' system could be leaked and a brute force attack against the public keys to identify the private key is possible. However, the probability of success is negligibly low for a well implemented asymmetric cryptographic algorithm.

$$\begin{aligned}
&\text{P[database of passwords is leaked} \cup \text{salt leaked]} && \text{(D.81)}\\
&= \left(\frac{(49)(0.72)}{4788}\right)\left(\frac{49}{4788}\right) = 0.0000754078177900894 &&\\
&P_{pwd}\text{[user password is guessed in } 10^{13}/\#users \text{ guesses]} = 0.055 && \text{(D.82)}\\
&P_{USB,cTLS}\text{[brute force guess private key]=0} && \text{(D.83)}
\end{aligned}$$

Side Channel Attack Side channel attacks are possible against all three security methods: password, TLS and USB cryptographic device. However, measuring the probability of them occurring is a difficult problem. We make an attempt at rough estimates here but due to the lack of sufficient data these numbers are not reliable.

While side channel attacks are generally designed against cryptographic keys, there is at least one attack which will work against passwords [158]. This attack was described in Section D.3.0.4 and applies again here.

$$\begin{aligned}
&\text{P}_{pwd}\text{[side channel attack]} = && \text{(D.84)}\\
&\text{P[targeted attack].P[not detected].P[malware]} &&\\
&= (0.033896)(0.8)\left(\frac{(33)(0.72)}{3231}\right) = 0.000199410451253482 &&
\end{aligned}$$

The NIST guidelines state that the USB device must be resistant to power and timing analysis attacks. Other attacks are possible though, particularly those that can be carried out with physical access [96].

$$\begin{aligned}
&\text{P}_{USB}\text{[side channel attack]} = && \text{(D.85)}\\
&\text{P[targeted].P[physical access].P[exploit vulnerabilities]} &&\\
&= (0.033896)\left(\frac{(53)(0.72)}{3231}\right)\left(\frac{6}{4788}\right) = 0.00000050166865632045 &&
\end{aligned}$$

Encryption for TLS using the private key is susceptible to a side channel attack over the network [2]. Side channel attacks to determine the private key can also be done via physical access or malware.

$$\begin{aligned}
&\mathrm{P}_{cTLS}[\text{side channel attack}] = \qquad \text{(D.86)}\\
&\mathrm{P}[\text{targeted attack}].\,(\mathrm{P}[\text{physical access}] \ \cup\ \mathrm{P}[\text{malware}])\\
&\cup\,(\mathrm{P}[\text{eavesdropper(stat:LAN access)}])\,(\mathrm{P}[\text{exploit vulnerabilities}])\\
&= (0.033896)\left(\frac{(53)(0.72)}{3231} + \frac{(33)(0.72)}{3231}\right) + \left(\frac{(118)(0.72)}{3231}\right)\left(\frac{6}{4788}\right)\\
&= 0.000682546111239098
\end{aligned}$$

Phishing or Pharming A password will only be able to be phished if the client authenticated TLS connection is compromised first.

$$P_{pwd}[\text{phishing} \cap \text{pharming}] = \qquad \text{(D.87)}$$

$$\mathrm{P}[\text{phishing}] = \frac{(653)(0.72)}{4788} = 0.0981954887218045$$

$$P_{USB}[\text{phishing} \cap \text{pharming}] = 0 \qquad \text{(D.88)}$$

It is not reasonable to assume that the private key of the USB or client authenticated TLS can be compromised via phishing. Since it should never be transmitted. However we do consider it to be possible to compromise it via social engineering.

$$P_{cTLS}[\text{phishing} \cap \text{pharming}] = 0 \qquad \text{(D.89)}$$

Social Engineering The mitigation suggested by the NIST 2017 guidelines are to avoid using authenticators that present a risk of social engineering of third parties. The password alone does not fall into this category.

$$P_{pwd}[\text{social engineering}] = \mathrm{P}[\text{pretexting}] = \frac{(39)(0.72)}{4788} = 0.00586466165413534 \qquad \text{(D.90)}$$

We consider that a person masquerading as an IT assistant could relatively easily convince a user to send them the files which include the certificate and private key. It is slightly more difficult than revealing a password as the certificate and key cannot be easily stated over the phone. The Verizon statistics tell us that phones are the attack vector 5 of 3231 times.

$$P_{cTLS}[\text{social engineering}] = \text{P[pretexting]\&P[not phone]} \tag{D.91}$$
$$= \left(\frac{(39)(0.72)}{4788}\right)\left(1 - \frac{5}{3231}\right) = 0.00585558604031134$$

Convincing a user to part with a USB decide would be more difficult than either of the above. Since the user would not have the use of the device once they part with it as it cannot be duplicated there would be an inconvenience factor for the user in this. If we take the P[pretexting] to get someone's password as requiring 1 minute of the users' time. Then for a USB we set the time as 10 minutes for someone with physical proximity to the user and 3 working days (1440 minutes) for a remote attacker.

$$P_{USB}[\text{social engineering}] = \tag{D.92}$$
$$\left(\frac{\text{P[pretexting]}}{10}\right)(\text{P[partner exploits]}) + \left(\frac{\text{P[pretexting]}}{1440}\right)(\text{P[remote attacker]})$$
$$= \left(\frac{(39)(0.72)}{(4788)(10)}\right)\left(\frac{108}{4788}\right) + \left(\frac{(39)(0.72)}{(4788)(1440)}\right)\left(1 - \frac{108}{4788}\right)$$
$$= 0.000017209376827148$$

Online Guessing attacks

$$P_{pwd}[\text{online guessing successful}] \tag{D.93}$$
$$= \text{P[targeted attack].P[password in (99*365) guesses]}$$
$$+ \text{ P[brute force guessing].P[(username guessed in } 10^{13} \text{ guesses]}$$
$$* \text{ P[password guessed in (99*365)]}$$
$$= (0.033896)\,(0.01)(0.72) + \left(\frac{(73 + 631)(0.72)}{4788}\right)(0.63)(0.01)$$
$$= 0.000910998568420505$$
$$P_{USB,cTLS}[\text{online guessing successful}] = 0 \tag{D.94}$$

Endpoint compromise

$$P_{pwd}[\text{endpoint compromise}] = \frac{(678)(0.72)}{4788} = 0.101954887218044 \tag{D.95}$$

Because the private key never leaves the USB device, a compromised endpoint should not be able able to compromise the device.

$$P_{USB}[\text{endpoint compromise}] = 0 \tag{D.96}$$

A compromised endpoint could compromise the private key and certificate.

$$P_{cTLS}[\text{endpoint compromise}] = \frac{(678)(0.72)}{4788} = 0.101954887218044 \tag{D.97}$$

Unauthorized binding

$$P_{pwd}[\text{unauthorized binding}] = \text{P}[\text{use stolen credentials}].\text{P}[\text{targeted attack}] \tag{D.98}$$

$$= \left(\frac{(631)(0.72)}{4788}\right)(0.033896) = 0.00321629714285714$$

To compromise the USB or client TLS the attacker would need access to the database and need to have a USB, and certificate and private key created. This would need to be a targeted attack.

$$P_{USB,cTLS}[\text{unauthorized binding}] = \left(\frac{(49)(0.72)}{4788}\right)(0.1)(0.033896) \tag{D.99}$$

$$= 0.000024976$$

D.3.10.2 Calculation of NIST 2017 Level 3 losses

We quantify benefits by splitting the equation to look at the scenario when the password dataset has been leaked and when it has not (Equation D.16). For the NIST 2003 Level 3 policy this gives use the following probabilities of compromise:

$$\mathbb{P}_{pwd}[\text{compromise}|\text{leak}] = 1 - \prod_a (1 - \mathrm{P}_{pwd,a}) = 0.3214100261 \tag{D.100}$$
$$\mathbb{P}_{_{USB}}[\text{compromise}|\text{leak}] = 1 - \prod_a (1 - \mathrm{P}_{USB,a}) = 0.0078205399 \tag{D.101}$$
$$\mathbb{P}_{_{cTLS}}[\text{compromise}|\text{leak}] = 1 - \prod_a (1 - \mathrm{P}_{cTLS,a}) = 0.1109406998 \tag{D.102}$$

$$\mathbb{P}_{pwd}[\text{compromise}|\text{no leak}] = 1 - \prod_a (1 - \mathrm{P}_{pwd,a}) = 0.2819153715 \tag{D.103}$$
$$\mathbb{P}_{_{USB}}[\text{compromise}|\text{no leak}] = 1 - \prod_a (1 - \mathrm{P}_{USB,a}) = 0.00782053993 \tag{D.104}$$
$$\mathbb{P}_{_{cTLS}}[\text{compromise}|\text{no leak}] = 1 - \prod_a (1 - \mathrm{P}_{cTLS,a}) = 0.1109406998 \tag{D.105}$$

$$\begin{aligned} p_l &= \mathbb{P}[\text{compromise}|\text{leak}] \qquad\qquad (\text{D.106})\\ &= \mathbb{P}_{pwd}[\text{compromise}|\text{leak}] * \mathbb{P}_{_{USB}}[\text{compromise}|\text{leak}] * \mathbb{P}_{_{cTLS}}[\text{compromise}|\text{leak}]\\ &= 0.0002788605356\\ p_{l'} &= \mathbb{P}[\text{compromise}|\text{no leak}] \qquad\qquad (\text{D.107})\\ &= \mathbb{P}_{pwd}[\text{compromise}|\text{no leak}] * \mathbb{P}_{_{USB}}[\text{compromise}|\text{no leak}] * \mathbb{P}_{_{cTLS}}[\text{compromise}|\text{no leak}]\\ &= 0.0002445943347 \end{aligned}$$

Now, taking the values defined in Section D.1: $Lsystem = \$10^7$, $L_1 = \$166$, and $N = 500$ users, we calculate the expected loss due to compromise for the organization when the NIST 2017 Level 3 policy is in place.

$$\begin{aligned} &\mathbb{E}[\text{Loss}|\text{NIST 2003 L1 policy}] = \qquad\qquad (\text{D.108})\\ &\Big(\mathbb{P}\left[\mathcal{N}\left(0.0002788605356, \frac{(0.0002788605356)(1-0.0002788605356)}{500}\right) > 0.5\right] \times \$10^7\\ &+ (500)(0.0002788605356)(\$166)\Big) \times 0.0000754078177900894\\ &+ \Big(\mathbb{P}\left[\mathcal{N}\left(0.0002445943347, \frac{(0.0002445943347)(1-0.0002445943347)}{500}\right) > 0.5\right] \times \$10^7\\ &+ (500)(0.0002445943347)(\$166)\Big) \times (1 - 0.0000754078177900894)\\ &= \$0.00180843073953175 + \$21.0335265280851 = \$21.0353349588246 \end{aligned}$$

D.3.10.3 Quantifying benefit of the NIST 2017 Level 3 policy

Taking the value for expected losses with no policy from Section D.3.2.3, our computation of the NIST 2017 L3 benefits is:

$$\begin{aligned}\mathbb{E}[\text{Benefits}] &= \mathbb{E}[\text{Loss}|\text{No policy}] - \mathbb{E}[\text{Loss}|\text{NIST 2017 L3 policy}] \qquad (D.109)\\ &= \$105571.91 - \$21.04 = \$105550.87\end{aligned}$$

D.3.11 Costs: NIST 2017 Level 3 policy

In this section we will quantify the costs associated with the pieces of advice in the NIST 2017 policy. Because the advice relating to Memorized Secrets is the same as for the Level 1 policy we will not repeat the details of the calculations here. They are included in Table D.2 as $j = 1 \ldots 5$. The advice relating to Rate limiting, $j = 6$, Access controls, $j = 7$ & $j = 8$, are also the same as for NIST 2017 Level 1. Therefore, these will also not be re-discussed.

j=9: Client authenticated protected channel

- A verifier impersonation resistant authentication protocol SHALL establish an authenticated protected channel with the verifier.
  - One example of a verifier impersonation resistant authentication protocol is client authenticated TLS, because the client signs the authenticator output along with earlier messages from the protocol that are unique to the particular TLS connection being negotiated.

The organization must set up a system to allow Client authenticated TLS to be used by all of it's users. We suggest that this is a major implementation cost: 3 working weeks of the organisation's time. The organization will need to distribute a certificate to each of it's users. Options for this include physically handing certificates out, posting them, or allowing them to be downloaded from a website. Because we have decided the users in this company are external, we opt for the latter option. This can be done using open source certificate software. The user has the non trivial task of downloading and setting up the certificate. We estimate that on average 15 minutes help desk time will be required for users to help them if they run into difficulties with this task. Major user education will also be required.

The user needs 30 minutes to set up the certificate when they create an account. They will also need to set up a new certificate every time it expires. We say that in our year long time frame a user will need to install a certificate twice.

It will take 3 seconds for the organization to generate a public private key pair for the user's certificate. It will also take approximately 0.1 seconds extra to authenticate each user at login.

If the user's computer is lost, broken or stolen they will need to authenticate themselves in another way and generate a new certificate. We estimate that this process would take an hour for the user and will require another 15 minutes help desk time. It will happen with the Verizon probability of a physical theft $P_{9,9} = \frac{39}{4788} = 0.081453634$.

An inconvenience of the client authentication is that it is linked to a device. Google has reported that 6 in 10 users switch between devices while online shopping [159]. This statistic could be applied to tell us that each user will be inconvenienced by this lack of portability with a probability of 0.6. We estimate the inconvenience as 10 second per login.

In Table D.4 we reference this information using a second set of rows as we have multiple inputs to some cost categories.

j=10: Reauthentication

- Periodic re-authentication of subscriber sessions SHALL be preformed after 12 hours or 15 minutes inactivity; SHALL use both authentication factors.

Komanduri's statistics [85] tell us that with this password policy a user will take 31.85 seconds to authenticate. We double this when a USB must be used as well.

If we estimate that a user moves away from their computer, for more than 15 minutes, three times per day. Then, given 261 working days in a year, each user will need to authenticate 783 times in time frame T.

j=11: Single factor cryptographic device

- The single factor cryptographic device authenticators encapsulate one or more secret keys unique to the device that SHALL NOT be exportable (i.e., cannot be removed from the device).
- The challenge nonce SHALL be at least 64 bits in length, and SHALL either be unique over the authenticator's lifetime or statistically unique.
- Tamper detection and response should be in place for covers and doors [48].
- Single-factor cryptographic device authenticators SHOULD require a physical input (e.g., the pressing of a button) in order to operate.
- The secret key and its algorithm SHALL provide at least the minimum security length specified in the latest revision of SP 800-131A (112 bits as of the date of this publication).
- Use a cryptographic authenticator that requires the verifier store a public key corresponding to a private key held by the authenticator.
- A CSP SHOULD bind at least two physical authenticators (something you have) to the subscriber's credentials.
  - Binding of multiple authenticators is preferred in order to recover from the loss or theft of the subscriber's primary authenticator.

The different pieces of advice relating to the use of single factor cryptographic devices overlap largely with each other. Therefore we will discuss them all together. This means that in Table D.4 we will have multiple costs in each category under $j = 11$.

The type of single factor cryptographic device we are considering is a dedicated USB authenticator. We allow three working weeks of organization time to set up this method of authentication. This allows for finding a supplier, briefing customers and integrating the protocols into their software. We assign 15 minutes of the organization's time per user for distribution of the USBs. Also if a user loses the device the organization will need to have an alternative method for identifying and authenticating the user and distribute a new USB to them. `Mozy.ie` in 2012 [107] found that 70% of people have lost a data storage device. They also found that the average person loses 1.24 items a year and that the more frequently you carry an item, the more likely you are to lose it. We will say that 2.5% of users will lose their USB device in the year long time frame. The probability that they lose the second physical authenticator bound to their account is: 0.01. So losing either of these will be is: $0.01 + 0.025 = 0.035$. This is the probability that the organization will need to send a replacement.

The user will need to wait approximately 15 working hours for their device to arrive when they create their account. If they lose their two physical authentication devices they will also need to wait for a replacement to arrive; probability is $0.01 \times 0.025 = 0.00025$. To demonstrate this the probability value is 1 for the certainty that at creation they will need to have the device sent to them, plus 0.00025 to represent the probability of being locked out while a replacement device is issued. $P_{11,11} = 1 + 0.00025$. At each login the user will need to take the time to press a button or insert the USB in order to authenticate. We assign an average of 31.85 seconds to this; the same amount of time as to enter the password. An inconvenience for the user is that it is now necessary to carry this authenticator around with them. Pixie Technology Inc. [131] found that 28% of people misplace their car keys at least once a week. Since car keys are another form of physical authenticator we can equate the likelihood of misplacing them to the likelihood of misplacing the USB device. So with $P_{11,11} = 0.28$, it will take 10 minutes for the user to find their USB, $C_{11,11} = C_{11}(10mins)$, $R_{11,11} = \#logins/\#weeks = \#logins/52$.

The organization will use additional computing power. For each login, they will need to create a statistically random 64 bit nonce, they will also then need to verify that the signature did originate from the user matching the saved public key. $C_{5,11} = C_5(3secs)$

We presume that the organization purchases the USB devices from a dedicated supplier before distributing them to their users. We estimate each device costing \$6 and the organization will need to purchase two of them per user to satisfy the advice to distribute backup authenticators. If a user loses their

USB device, it will need to be replaced. We use the same probability of loss of 0.035% as before. We also assign \$0.50 per user for the cost of delivery each time a USB needs to be posted.

Administrators also assigned major user education and help desk costs to this advice.

D.3.11.1 Calculation of NIST 2017 Level 3 costs

Table D.4 shows the costs associated with the NIST 2017 Level 3 advice. Note that because $j = 1, \ldots, 8$ are included in Table D.2 under the NIST 2017 Level 1 advice, we do not repeat them here.

Figure D.4: Costs of implementing the NIST 2017 Level 3 password advice.

		Organisation costs					User costs						
		$C_1(t_o)$: Increased help desk/user support time	$C_2(t_o, t_u)$: User education required	$C_3(r_o)$: Additional organization resources	$C_4(t_o)$: Organization time taken to implement	$C_5(tc_o)$: Increased organization computing power needed	$C_6(t_u)$: Makes it more difficult to create a password	C_7: Increased risk of forgetting	$C_8(r_u)$: Additional user resources needed	$C_9(t_u)$: Need to pick a new password	$C_{10}(tc_u)$: Increased user computing power need	$C_{11}(t_u)$: User time and inconvenience	$C_{12}(t_u)$: User education time required
j=9: Client authenticated protected channel	*p(i,9)*	1	1		1	1					1	1	1
	C(i,9)	C_1(15 mins)	C_2(1hr)		C_4(3 weeks)	C_5(3 secs)					C_{10}(0.5 sec)	C_{11}(30mins)	C_{12}(15mins)
	R(i,9)	2*#users	12		1	2*#users					#logins	2*#users	12*#users
j=9 continued	*p(i,9)*	0.081453634				1						0.6	
	C(i,9)	C_1(15 mins)				C_5(0.5 sec)						C_{11}(10 secs)	
	R(i,9)	#users				#logins						#logins	
j=9 continued	*p(i,9)*											0.081453634	
	C(i,9)											C_9(1hr)	
	R(i,9)											#users	
j=10: Reauthentication	*p(i,10)*				1							1	
	C(i,10)				C_4(1 day)							C_{11}(63.7)	
	R(i,10)				1							783*#users	
j=11: Single factor cryptographic device	*p(i,11)*	1	1	1	1	1						1.00025	1
	C(i,11)	C_1(15mins)	C_2(1hr)	$12.50	C_4(3 weeks)	C_5(3 secs)						C_{11}(15hrs)	C_{12}(15mins)
	R(i,11)	4*#users	12	#users	1	#logins						#users	12*#users
j=11 continued	*p(i,11)*			0.035	1							0.28	
	C(i,11)			$6.50	C_4(15 mins)							C_{11}(10mins)	
	R(i,11)			#users	#users							#login/52	
j=11 continued	*p(i,11)*											1	
	C(i,11)											C_{11}(31.85 secs)	
	R(i,11)											#logins	

Using this table and the values defined in Section D.2 we have the following results for the cost of the NIST 2017 L3 policy:

$$\mathbb{E}[Costs] = \sum_{\substack{i=12\\j=11}} P_{c_{i,j}}.C_{i,j}.R_{i,j} = \$438{,}787.67 \tag{D.110}$$

D.3.12 Value of the NIST 2017 Level 3 policy

$$\mathbb{E}[Value] = \mathbb{E}[Benefits] - \mathbb{E}[Costs] \tag{D.111}$$

$$\mathbb{E}[Value] = \$105{,}550.87 - \$438{,}787.67 = -\$333{,}236.80 \tag{D.112}$$

NIST 2003 Level 4 Worked example

Level 4 is the highest authentication policy in the NIST 2003 standards. It requires authentication using a hard cryptographic token and a password.

The NIST 2017 Level 3 policy gave the option of using a password to “unlock” the hard cryptographic token and then authenticating with the verifier using the cryptographic key. Or using a password and also the cryptographic key to authenticate with the verifier. These are similar to the two authentication options described in NIST 2017 Level 4: A multi-factor cryptographic device, or a single factor cryptographic device used in conjunction with a memorized secret. To maintain consistency with the resources used in our NIST 2017 Level 3 analysis, we will assume both the hard cryptographic device and the password are communicated to the verifier during authentication.

The language used in the two standards documents differ, but we consider the “hard cryptographic token” in NIST 2003 to be identical to the “cryptographic devices” described in NIST 2017.

D.3.13 Policy summary

General authentication rules

- Remote registration is limited to Levels 1 through 3.

- At this level sensitive data transfers shall be cryptographically authenticated using keys bound to the authentication process.
  - Either public key or symmetric key technology may be used.
  - Example implementation: Client authenticated TLS.
- Re-authentication shall be required after not more than 24 hours from the initial authentication.
- Files of shared authentication secrets shall be protected by discretionary access controls that limit access to administrators and only those applications that require access.

Authenticator 1: NIST 2003 Level 2 requirements for Memorized secrets

- Shared secret files shall not contain the plaintext passwords or secret; two alternative methods may be used to protect the shared secret:
  1. Passwords may be concatenated to a salt and/or username and then hashed with an Approved algorithm.
  2. Store shared secrets in encrypted form using Approved encryption algorithms and modes and decrypt the needed secret only when immediately required for authentication.

  - To contrast the NIST 2017 policy, we choose the encryption option.
- Password authentication systems can make targeted password guessing impractical by:
  - Requiring use of high-entropy passwords. NIST offers a number of examples to satisfy the Level 2 password requirements. The most familiar one to us is:
    - ∗ Require a minimum of 8 character passwords, selected by subscribers from an alphabet of 94 printable characters,
    - ∗ Require subscribers to include at least one upper case letter, one lower case letter, one number and one special character, and;
    - ∗ Use a dictionary to prevent subscribers from including common words and prevented permutations of the username as a password.
  - Limiting the number of unsuccessful authentication attempts, or by controlling the rate at which attempts can be carried out.

    * Consider a system that required passwords to be changed every two years and limited trials by locking an account for 24 hours after 6 successive failed authentication attempts.

Authenticator 2: Hard token - a hardware device that contains a protected cryptographic key.

- Authentication is accomplished by proving possession of the device and control of the key. Hard tokens shall:
  - require the entry of a password or a biometric to activate the authentication key; or must also use a password in a secure authentication protocol, to establish two factor authentication.
  - Authentication keys may not be exportable.
  - Tamper detection and response should be in place for covers and doors [48].

D.3.14 Benefits: NIST 2003 Level 4 policy

The NIST 2003 Level 4 policy recommends two factor authentication and Client authentication TLS. Therefore, similar to the NIST 2017 Level 3 policy, the probability of successfully compromising a user is marked by the probability of compromising the single factor cryptographic device, the memorized secret and the client certificate and private key.

Therefore, we will require this computation again:

$$p_{pwd} = 1 - \prod_a (1 - \mathrm{P}_{pwd,a}) \tag{D.113}$$

$$p_{USB} = 1 - \prod_a (1 - \mathrm{P}_{USB,a}) \tag{D.114}$$

$$p_{cTLS} = 1 - \prod_a (1 - \mathrm{P}_{cTLS,a}) \tag{D.115}$$

where p is the probability of the $pwd/USB/cTLS$ being compromised. P_a is the probability of compromise due to each individual attack type a. As for all other policies, we must again calculate this p when there has been a leak of the password file and when there has not:$\mathbb{P}[\text{compromise}|\text{no leak}]$ and $\mathbb{P}[\text{compromise}|\text{database leak}]$

D.3.14.1 Quantifying attacks: NIST 2003 Level 4

Assertion Manufacture or Modification The risk of this attack occurring is mitigated for all three factors by having access controls in place.

$$\begin{aligned}&\mathrm{P}_{pwd,cTLS,USB}[\text{assertion manufacture or modification}] \qquad \text{(D.116)}\\ &= \left(\frac{(128)(0.72)}{4788}\right)(0.1) = 0.00192481203007519\end{aligned}$$

Theft

- Re-authentication shall be required after not more than 24 hours from the initial authentication.

The password and the certificate and private key will be compromised if the computer is unlocked when it is stolen.

Given a user logs in each day, working a 5 day working week, there is a $\left(\frac{6}{7}\right) = 0.85714285714$ probability that the computer is unlocked when it is stolen.

$$\begin{aligned}&\mathrm{P}_{pwd,cTLS}[\text{theft}] \qquad \text{(D.117)}\\ &= \mathrm{P}[\text{stolen}]*\mathrm{P}[\text{tampering}] + \mathrm{P}[\text{stolen}]*\mathrm{P}[\text{unlocked}]\\ &= \left(\frac{(39)(0.72)}{4788}\right)\left(\frac{27}{4788}\right) + \left(\frac{(39)(0.72)}{4788}\right)(0.85714285714)\\ &= 0.00505992424668823\end{aligned}$$

The USB is susceptible to theft.

$$\mathrm{P}_{USB}[\text{theft}] = \frac{(39)(0.72)}{4788} = 0.00586466165413534 \qquad \text{(D.118)}$$

Duplication

The Secret key on the USB is not exportable and we assume the issuer of the USBs is trustworthy.

$$\mathrm{P}_{USB}[\text{duplication}] = 0 \qquad \text{(D.119)}$$

Password is only duplicable if revealed.

$$\begin{aligned}
&\mathrm{P}_{pwd}[\text{duplication}] \qquad \text{(D.120)}\\
&= \mathrm{P}[\text{divulge password}].\mathrm{P}[\text{partner exploits}] + \mathrm{P}[\text{recorded password compromised}]\\
&= (0.25)\left(\frac{(108)(0.72)}{3231}\right) + \left(\frac{(16)(0.72)}{3231}\right) = 0.00958217270195177
\end{aligned}$$

$$\begin{aligned}
&\mathrm{P}_{cTLS}[\text{duplication}] \qquad \text{(D.121)}\\
&= \mathrm{P}[\text{targeted attack}].\mathrm{P}[\text{private key} \cap \text{certificate leaked}]\\
&= 0.033896\left(\frac{(49)(0.72)}{4788}\right) = 0.000249759999999422
\end{aligned}$$

D.3.14.2 Eavesdropping

- At this level sensitive data transfers shall be cryptographically authenticated using keys bound to the authentication process

The communication of the password over the network is encrypted.

$$\begin{aligned}
&\mathrm{P}_{pwd}[\text{eavesdropping}] = \mathrm{P}[\text{physical surveillance}] + \mathrm{P}[\text{keylogger breach}] \qquad \text{(D.122)}\\
&= \left(\frac{(21)(0.72)}{4788}\right) + \left(\frac{(595)(0.72)}{4788}\right) = 0.0926315789473664
\end{aligned}$$

The USB cryptographic device and client authenticated TLS are not susceptible to eavesdropping attacks. They use public private key pairs and therefore the private key is never transmitted.

$$\mathrm{P}_{USB,cTLS}[\text{eavesdropping}] = 0 \qquad \text{(D.123)}$$

D.3.14.3 Offline Guessing attacks

- Store shared secrets in encrypted form using approved encryption algorithms and modes and decrypt the needed secret only when immediately required for authentication.

- Require a minimum of 8 character passwords, selected by subscribers from an alphabet of 94 printable characters,

- Require subscribers to include at least one upper case letter, one lower case letter, one number and one special character, and;

- Use a dictionary to prevent subscribers from including common words and prevented permutations of the username as a password.

First we have the probability of a leak:

$$\text{P[database of passwords is leaked]} = \frac{(49)(0.72)}{4788} = 0.00736842105263307 \quad \text{(D.124)}$$

Attackers can either steal the key with the dataset, or brute force guess the key.

Because of the complex password composition requirements we take Kelley et al.'s value for percentage of passwords cracked when they were made using their comprehensive8 password rules (22% of passwords guessed in 10^{13} guesses). This asked for passwords with "at least 8 characters including an uppercase and lowercase letter, a symbol, and a digit. It may not contain a dictionary word".

$$\begin{aligned} & P_{pwd}[\text{user password revealed}|\text{leak}] && \text{(D.125)} \\ & = \text{P[key leaked]} + \text{P[key brute forced]} + \text{P[user password in } 10^{13} \text{ guesses]} \\ & = \frac{49}{4788} + 0.01 + 0.22 = 0.240233918128655 \end{aligned}$$

The USB and client authenticated TLS public key stored on the organizations' system could be leaked and a brute force attack against the public keys to identify the private key is possible. However the probability of success is negligibly low for a well implemented asymmetric cryptographic algorithm.

$$P_{USB,cTLS}[\text{brute force guess private key}]{=}0 \quad \text{(D.126)}$$

Side Channel Attack

- No protection

Side channel attacks are possible against all three security methods; password, TLS and USB cryptographic device.

$$\mathrm{P}_{pwd}[\text{side channel attack}] = \mathrm{P}[\text{targeted attack}].\mathrm{P}[\text{not detected}].\mathrm{P}[\text{malware}] \tag{D.127}$$

$$= (0.033896)(0.8)\left(\frac{(33)(0.72)}{3231}\right) = 0.000199410451251986$$

There is no stipulation in the NIST 2003 guidelines for protection against power or timing analysis attacks. Side channel attacks can be carried out either with physical access or via malware. We still assume that a side channel attack is going to be targeted.

$$\mathrm{P}_{USB}[\text{side channel attack}] \tag{D.128}$$

$$= \mathrm{P}[\text{targeted}].\left(\mathrm{P}[\text{physical access}] \ \cup \ \mathrm{P}[\text{malware}]\right)$$

$$= (0.033896)\left(\frac{(53)(0.72)}{3231} + \frac{(33)(0.72)}{3231}\right) = 0.000649594651809964$$

Encryption for TLS using the private key is susceptible to a side channel attack over the network [2]. Side channel attacks to determine the private key can also be done via physical access or malware.

$$\mathrm{P}_{cTLS}[\text{side channel attack}] \tag{D.129}$$

$$= \mathrm{P}[\text{targeted attack}].\left(\mathrm{P}[\text{physical access}] \ \cup \ \mathrm{P}[\text{malware}]\right)$$

$$\cup \left(\mathrm{P}[\text{eavesdropper(stat:LAN.access)}]\right)\left(\mathrm{P}[\text{exploit vulnerabilities}]\right)$$

$$= (0.033896)\left(\frac{(53)(0.72)}{3231} + \frac{(33)(0.72)}{3231}\right) + \left(\frac{(118)(0.72)}{3231}\right)\left(\frac{6}{4788}\right)$$

$$= 0.000682546111241588$$

Phishing or Pharming A password will only be able to be phished if the client authenticated TLS connection is compromised first.

$$P_{pwd}[\text{phishing} \cap \text{pharming}] = \tag{D.130}$$

$$\mathrm{P}[\text{phishing}] = \frac{(653)(0.72)}{4788} = 0.0981954887218051$$

$$P_{USB,cTLS}[\text{phishing} \cap \text{pharming}] = 0 \tag{D.131}$$

Social Engineering

$$P_{pwd}[\text{social engineering}] \tag{D.132}$$
$$= \text{P[pretexting]} = \frac{(39)(0.72)}{4788} = 0.00586466165413429$$

We consider that a person masquerading as an IT assistant could relatively easily convince a user to send them the files which include the certificate and private key.

$$P_{cTLS}[\text{social engineering}] = \text{P[pretexting]\&P[not phone]} \tag{D.133}$$
$$= \left(\frac{(39)(0.72)}{4788}\right)\left(1 - \frac{5}{3231}\right) = 0.00585558604031134$$

As with NIST 2017 Level 3, we set the time a USB needs to be taken from the user for 10 minutes for someone with physical proximity to the user and 3 working days (1440 minutes) for a remote attacker.

$$P_{USB}[\text{social engineering}] \tag{D.134}$$
$$= \left(\frac{\text{P[pretexting]}}{10}\right)(\text{P[partner exploits]}) + \left(\frac{\text{P[pretexting]}}{1440}\right)(\text{P[remote attacker]}) \tag{D.135}$$
$$= \left(\frac{(39)(0.72)}{(4788)(10)}\right)\left(\frac{108}{4788}\right) + \left(\frac{(39)(0.72)}{(4788)(1440)}\right)\left(1 - \frac{108}{4788}\right) = 0.000017209376827148 \tag{D.136}$$

Online Guessing attacks

- Require a minimum of 8 character passwords, selected by subscribers from an alphabet of 94 printable characters,
- Require subscribers to include at least one upper case letter, one lower case letter, one number and one special character, and;
- Use a dictionary to prevent subscribers from including common words and prevented permutations of the username as a password.
- Limit trials by locking an account for 24 hours after 6 successive failed authentication attempts.

An attacker can make 6 guesses every 24 hours. Therefore in a year, the attacker can make 2190 guesses against each valid account in the year time frame. According to Kelley et al., because of the complex password composition restrictions none of the users' passwords are successfully cracked with this number of guesses [83].

$$P_{pwd}, USB, cTLS[\text{online guessing successful}] = 0 \tag{D.137}$$

Endpoint compromise

$$P_{pwd}[\text{endpoint compromise}] = \frac{(678)(0.72)}{4788} = 0.1019548887218044 \tag{D.138}$$

Because the private key never leaves the USB device, a compromised endpoint should not be able to compromise the device.

$$P_{USB}[\text{endpoint compromise}] = 0 \tag{D.139}$$

A compromised endpoint could compromise the private key and certificate.

$$P_{cTLS}[\text{endpoint compromise}] = 0.1019548887218044 \tag{D.140}$$

$$\tag{D.141}$$

Unauthorized binding The unauthorized binding of the authenticator to an attacker could take the form of an unauthorized password reset. In order for an attacker to successful reset a subscribers password they would need to already have access to their email account and be interested in targeting that individual since it is a time consuming attack.

$$P_{pwd}[\text{unauthorized binding}] = \text{P[use stolen credentials].P[targeted attack]} \tag{D.142}$$

$$= \left(\frac{(631)(0.72)}{4788}\right)(0.033896) = 0.00321629714285458 \tag{D.143}$$

To compromise the USB or client TLS the attacker would need access to the database and need to have a USB, and certificate and private key created. This would need to be a targeted attack.

$$P_{USB,cTLS}[\text{unauthorized binding}] = \left(\frac{(49)(0.72)}{4788}\right)(0.1)(0.033896)$$
$$= 0.0000249759999994224$$

D.3.14.4 Calculation of NIST 2003 Level 4 losses

We quantify benefits by splitting the equation to look at the scenario when the password dataset has been leaked and when it has not (Equation D.16). For the NIST 2003 Level 3 policy we have the following probabilities of compromise:

$$\mathbb{P}_{pwd}[\text{compromise}|\text{leak}] = 1 - \prod_a (1 - \mathrm{P}_{pwd,a}) = 0.4559828624 \tag{D.144}$$

$$\mathbb{P}_{USB}[\text{compromise}|\text{leak}] = 1 - \prod_a (1 - \mathrm{P}_{USB,a}) = 0.0084645569 \tag{D.145}$$

$$\mathbb{P}_{cTLS}[\text{compromise}|\text{leak}] = 1 - \prod_a (1 - \mathrm{P}_{cTLS,a}) = 0.1142891665 \tag{D.146}$$

$$\mathbb{P}_{pwd}[\text{compromise}|\text{no leak}] = 1 - \prod_a (1 - \mathrm{P}_{pwd,a}) = 0.2839675914 \tag{D.147}$$

$$\mathbb{P}_{USB}[\text{compromise}|\text{no leak}] = 1 - \prod_a (1 - \mathrm{P}_{USB,a}) = 0.0084645569 \tag{D.148}$$

$$\mathbb{P}_{cTLS}[\text{compromise}|\text{no leak}] = 1 - \prod_a (1 - \mathrm{P}_{cTLS,a}) = 0.1142891665 \tag{D.149}$$

$$\begin{aligned} p_l &= \mathbb{P}[\text{compromise}|\text{leak}] && \text{(D.150)} \\ &= \mathbb{P}_{pwd}[\text{compromise}|\text{leak}] * \mathbb{P}_{USB}[\text{compromise}|\text{leak}] * \mathbb{P}_{cTLS}[\text{compromise}|\text{leak}] \\ &= 0.0004411210827 \\ p_{l'} &= \mathbb{P}[\text{compromise}|\text{no leak}] && \text{(D.151)} \\ &= \mathbb{P}_{pwd}[\text{compromise}|\text{no leak}] * \mathbb{P}_{USB}[\text{compromise}|\text{no leak}] * \mathbb{P}_{cTLS}[\text{compromise}|\text{no leak}] \\ &= 0.0002747122791 \end{aligned}$$

Now, taking the values defined in Section D.1: $Lsystem = \$10^7$, $L_1 = \$166$, and $N = 500$ users, we calculate the expected loss due to compromise for the organization when the NIST 2017 Level 3 policy is in place.

$$\begin{aligned}
&\mathbb{E}[\text{Loss}|\text{NIST 2003 L1 policy}] = \qquad\qquad\qquad\qquad \text{(D.152)}\\
&\Bigg(\mathbb{P}\left[\mathcal{N}\left(0.00044112108, \frac{(0.00044112108)(1-0.00044112108)}{500}\right) > 0.5\right] \times \$10^7\\
&+ (500)(0.00044112108)(\$166)\Bigg) \times 0.00736842105263307\\
&+ \Bigg(\mathbb{P}\left[\mathcal{N}\left(0.0002747122791, \frac{(0.0002747122791)(1-0.0002747122791)}{500}\right) > 0.5\right] \times \$10^7\\
&+ (500)(0.0002747122791)(\$166)\Bigg) \times (1-0.00736842105263307)\\
&= \$0.279531466191297 + \$23.4511752646053 = \$23.7307067307966
\end{aligned}$$

D.3.14.5 Quantifying benefit of the NIST 2003 Level 4 policy

Taking the value for expected losses with no policy from Section D.3.2.3, our computation of the NIST 2017 L3 benefits is:

$$\begin{aligned}
\mathbb{E}[\text{Benefits}] &= \mathbb{E}[\text{Loss}|\text{No policy}] - \mathbb{E}[\text{Loss}|\text{NIST 2017 L3 policy}] \qquad \text{(D.153)}\\
&= \$20993.78 - \$0.18 = \$20993.60
\end{aligned}$$

D.3.15 Costs: NIST 2003 Level 4 policy

In this section we will quantify the cost associated with the pieces of advice in the NIST 2003 Level 4 policy.

j=1: In-person registration

- Remote registration is limited to Levels 1 through 3.

At creation, a user must bring two IDs and proof of address to the organization in person. We assign 15 minutes per user for the organization help desk and 6 hours for the user. We also assign organisation resources for the storage of this information as 20 cent per user.

j=2: Client authenticated protected channel

- At this level sensitive data transfers shall be cryptographically authenticated using keys bound to the authentication process.

- Either public key or symmetric key technology may be used.
- Example implementation: Client authenticated TLS.

This has very similar costs to NIST 2017 Level 3. The difference is that the user can be handed a physical certificate when they register in person. So this will not be an addition cost at creation for the user. However certificates expire so the user will need to return to the organization to update it or update it themselves at least once a year. We assume the update is done remotely by the users. We assign 30 minutes for an average user to update their certificate and 15 minutes organization help desk time to guide the user through the process.

The organization of course also has the time it will take to set up the client authenticated procedures in the first place, $C_{4,2} = C_4(43200)$

If the user loses their computer/laptop, P[computer lost] $= \frac{39}{4788} = 0.081453634$, they will need to return to the organization to get a new certificate. This will take 6 hours of the user's time and another 15 minutes of the organization's time.

It will take 3 seconds for the organization to generate a public private key pair for the user's certificate; at creation and at expiry $P_{5,2} = 1$, $R_{5,2} = 2*\#users$.It will also take approximately 0.1 seconds extra to authenticate each user at login. As before, there is a cost for the user associated with needing to always use the same device at login.

j=3: Reauthentication

- Reauthentication shall be required after not more than 24 hours from the initial authentication.

Komanduri's statistics [85] tells us that with the comprehensive8 password policy a user will take 104.86 seconds to authenticate. We add on 31.85 seconds when a USB must be used as well. A user will need to authenticate each working day during the year.

j=4: Access to password file

- Files of shared secrets used by verifiers shall be protected by discretionary access controls that limit access to administrators and only those applications that require access.

This advice incurs minor help desk costs periodically, a small amount of user education periodically and it is a minor cost to implement.

j=5: Password storage

- Shared secret files shall not contain the plaintext passwords or secret
    - Store shared secrets in encrypted form using Approved encryption algorithms and modes and decrypt the needed secret only when immediately required for authentication.

We have assigned 1 day of organizations time to setting up a system which will encrypt the passwords. There is an additional 1 second of computing time needed for the organization to decrypt and re-encrypt the password at each user login.

j=6: Composition

- Require a minimum of 8 character passwords, selected by subscribers from an alphabet of 94 printable characters,
- Require subscribers to include at least one upper case letter, one lower case letter, one number and one special character, and;
- Use a dictionary to prevent subscribers from including common words and prevented permutations of the username as a password.

Setting up the composition rules is a minor implementation cost to the orgnaisation. Kelley et al. recorded that users will take 242.56 seconds to create passwords of this type [85].

Kelley et al. also provide statistics that allowed us the calculate the probability of forgetting this type of password as: $P_{7,5} = 0.02834645669$. The 'Probability' that this advice makes it more difficult to create a password can also be taken from these statistics '$P'_{7,6} = 2.12$.

We assign 1 hour every 6 months to the organisation for user education and similarly on the User side, we assign 15 minutes every 6 months for the users. The help desk cost of a user forgetting their password is covered under the cost of forgetting function.

j=7: Rate limiting

- Limit trials by locking an account for 24 hours after 6 successive failed authentication attempts.

Brostoff and Sasse [18] show that 31% of their study participants failed to login successfully within 3 attempts whereas only 7% failed within 10 attempts.

If we assume there is a linear relationship between the number of attempts allowed and the failure rate then we can extrapolate that 6 attempts will result in a failure rate of 21%.

Brostoff and Sasse also found that approximately 7% of these failed logins led to password reminder requests: $P_{(7,1)} = 0.21 * 0.07 = 0.0147$

We calculate the probability and cost of an attacker targeting a user to lock their account the same as we did for the NIST 2003 Level 1 policy.

$$\begin{aligned}\text{P[malicious lockout]} &= \text{P[targeted.attack]}\,(\text{P[brute force.attacker]} + \text{P[partner]}) \qquad \text{(D.154)}\\ &= (0.033896)\left(\frac{(73)(0.72)}{4788} + \frac{(108)(0.72)}{3231}\right) = 0.00118786145\end{aligned}$$

Therefore, the total probability that the user's account is locked for 24 hours is: $0.21 + 0.00118786145 = 0.21118786145$.

There is no stipulation in the advice that an administrator can unlock the account.

j=8: Password expiry

- Requires passwords to be changed every two years

This means that within our time frame approximately half the users' passwords will expire. It is a minor cost for organisations to implement. It involved major user education and major help desk time. The help desk time will be linked to users not changing passwords on-time and becoming locked out of systems and difficulties involved with password changes. This is in additional to the inevitable help desk time needed when users forget their new password which is covered under the cost *Increased risk of forgetting.* The additional help desk time will occur with each password expiry deadline within timeframe T.

With each password expiration, a user is unable to do their work until the have reset their password. A 2014 study by NIST [26] estimated that, they can spend up to 12.4 hours per year changing passwords on a 90-day expiration schedule, or 18.6 hours changing passwords on a 60-day expiration schedule. If we take these statistics and assume they are linear, then for expiry every two year, this will take 1hr 33 minutes out of a work day for 9 accounts. If we assume this policy applies to just one account rather than nine, then this tells us that it can take 10 minutes in our time frame T for users to reset their password.

The other inconvenience for the user is the repetition of the password creation policy and the increased risk of forgetting a newly created passwords. In this way we take the user costs identified as part of the composition requirement and reapply them here.

j=9: Single factor cryptographic device

- Require the entry of a password or a biometric to activate the authentication key; or must also use a password in a secure authentication protocol, to establish two factor authentication.
- Authentication keys may not be exportable.
- Tamper detection and response should be in place for covers and doors [48].

The type of single factor cryptographic device we are considering is a dedicated USB authenticator. We allow 3 weeks for the organization to set up the method of authentication. This allows for finding a supplier, briefing customers and integrating the protocols into their software. Users may collect the USBs at registration.

Also if a user loses the device ($P = 0.025$) they need to return to the organization, identify themselves and receive a new authenticator. We set this as 15 minutes of the organizations time and 6 hours of the user's time.

At each login the user will need to take the time to press a button or insert the USB in order to authenticate. We assign 31.85 seconds to this; the same amount of time as to enter a basic password. An inconvenience for the user is that it is now necessary to carry this authenticator around with them. Pixie Technology Inc. [131] found that 28% of their study participants admitted to misplacing their carkeys at least once a week. We equate this to the likelihood of misplacing the USB device. So with $P_{11,9} = 0.28$, $C_{11,9} = C_2(10mins)$, $R_{11,9} = \#logins/\#weeks = \#logins/52$.

The organization will use additional computing power. Though there is no requirement for the security length of the nonce or keys $C_{5,9} = C_5(2)$

A USB device will need to be purchased for each user at creation and also if the user loses the device. We estimate each device costing \$6.

Administrators also assigned major user education and help desk costs to this advice.

D.3.15.1 Calculation of NIST 2003 Level 4 costs

Table D.5 shows the costs associated with the NIST 2003 Level 4 advice.

Figure D.5: Costs of implementing the NIST 2003 Level 4 password advice.

		Organisation costs					User costs						
		$C_1(t_o)$: Increased help desk/user support time	$C_2(t_o, t_u)$: User education required	$C_3(r_o)$: Additional organization resources	$C_4(t_o)$: Organization time taken to implement	$C_5(tc_o)$: Increased organization computing power needed	$C_6(t_u)$: Makes it more difficult to create a password	C_7: Increased risk of forgetting	$C_8(r_u)$: Additional user resources needed	$C_9(t_u)$: Need to pick a new password	$C_{10}(tc_u)$: Increased user computing power need	$C_{11}(t_u)$: User time and inconvenience	$C_{12}(t_u)$: User education time required
j=1: In-person registration	*p(i,1)*	1		1								1	
	C(i,1)	C_1(15mins)		C_3($0.20)								C_{11}(6hrs)	
	R(i,1)	#users		#users								#users	
j=2: Client authenticated protected channel	*p(i,9)*	1	1		1	1					1	1	1
	C(i,9)	C_1(15 mins)	C_2(1hr)		C_4(3 weeks)	C_5(3 secs)					C_{10}(0.5 sec)	C_{11}(30mins)	C_{12}(15mins)
	R(i,9)	#users	12		1	2*#users					#logins	#users	12*#users
j=2 continued	*p(i,9)*	0.081453634				1						0.6	
	C(i,9)	C_1(15 mins)				C_5(0.5 sec)						C_{11}(10 secs)	
	R(i,9)	#users				#logins						#logins	
j=2 continued	*p(i,9)*											0.081453634	
	C(i,9)											C_9(1hr)	
	R(i,9)											#users	
j=3: Reauthentication	*p(i,3)*				1							1	
	C(i,3)				C_4(1 day)							C_{11}(136.71 secs)	
	R(i,3)				1							261*(#users)	
j=4: Access to password files	*p(i,4)*	1	1		1								
	C(i,4)	C_1(15mins)	C_2(1hr)		C_4(1 day)								
	R(i,4)	#users	2		1								
j=5: Password storage	*p(i,5)*				1	1							
	C(i,5)				C_4(1 day)	C_5(1 sec)							
	R(i,5)				1	#logins							
j=6: Composition	*p(i,6)*		1		1		2.12	0.0283465		1			1
	C(i,6)		C_2(1hr)		C_4(1 day)		C_6(242.56 secs)	C_7		C_9(242.56 secs)			C_{12}(15 mins)
	R(i,6)		2		1		#users	#users		#users			2*#users
j=7: Rate limiting	*p(i,7)*		1		1			0.0147				0.21	1
	C(i,7)		C_2(1hr)		C_4(1 day)			C_7				C_{11}(24 hours)	C_{12}(15mins)
	R(i,7)		2		1			#users				#logins	2*#users
j=8: Password expiry	*p(i,8)*	1	1		1		2.12	0.0283465		1		1	1
	C(i,8)	C_1(15mins)	C_2(1hr)		C_4(1 day)		C_6(242.56 secs)	C_7		C_9(242.56 secs)		C_{11}(10mins)	C_{12}(15mins)
	R(i,8)	#users	12		1		0.5*#users	0.5*#users		0.5*#users		#users	12*#users
j=9: Single factor cryptographic device	*p(i,9)*	0.025	1	1	1	1						0.025	1
	C(i,9)	C_1(15mins)	C_2(1hr)	$6	C_4(3 weeks)	C_5(2 secs)						C_{11}(6hrs)	C_{12}(15mins)
	R(i,9)	#users	12	#users	1	#logins						#users	12*#users
j=9 continued	*p(i,9)*	1		0.025								1	
	C(i,9)	C_1(15mins)		$6								C_{11}(31.85 secs)	
	R(i,9)	4*#users		#users								#logins	
j=9 continued	*p(i,10)*											0.28	
	C(i,10)											C_{11}(10mins)	
	R(i,10)											#login/52	