5. Assessing UID Technologies

Immediately after HUD introduced a UID in the Universal Data Elements ("Dataset"), many Planning Offices and Shelters began exploring technologies to construct, maintain, and use UIDs. The goal of this section is to describe how to assess plans and technologies in terms of their ability to perform an unduplicated accounting while protecting privacy. This section itemizes what should be the content of the assessment and what problems it should address.

This section and the next examine initial UID technologies in the absence of more recent regulation ("VAWA"). VAWA, as discussed in Section 7. subsequently rendered most of these technologies unacceptable. Section 8 through Section 12, introduces and tests PrivaMix as a solution that meets the higher privacy standards imposed by VAWA, but these two sections remain useful in characterizing the space of what makes a solution acceptable..

In this writing, a "Proposed Solution" is a UID technology bundled with an accompanying set of policies and practices for the construction, maintenance and use of a UID technology for Clients of Shelters in a HMIS. The entire package, UID technology, policies and practices, bundled together, is the subject of the assessment.

The overall problem for which UIDs have been introduced is easy to understand. It is termed the "HMIS Unduplicated Count Problem" and is stated below.

The HMIS Unduplicated Count Problem.
Given a set of Clients, a set of Shelters, and a Planning Office, where Clients visit Shelters, and Shelters report Dataset to the Planning Office on the Clients that visit, how should information about Clients be reported to Shelters and to the Planning Office such that the Planning Office can identify distinct visits of Clients across Shelters but not the identities of the Clients?

In order to determine whether a Proposed Solution is a sufficient solution to the HMIS Unduplicated Count Problem, an assessment must be done that demonstrates that the Proposed Solution remains useful for HMIS purposes while still being minimally invasive to privacy. Framed this way, the HMIS Unduplicated Count Problem is an optimization problem.[9] On the one hand, a Proposed Solution should provide an accurate accounting of distinct Client visits. On the other hand, a Proposed Solution should protect the privacy of Clients. The sufficiency of a Proposed Solution is based on performance guarantees that can be made. Specifically, a performance guarantee that the Proposed Solution has a minimal risk of re-identification when the solution is considered with other publicly and readily available information and techniques is termed a "Compliance Statement" in this writing. Similarly, a performance guarantee that the Proposed Solution provides a reasonably accurate unduplicated accounting of client visit patterns to shelters within the regional setting it is to be deployed is a "Warranty" in this writing. An assessment of a Proposed Solution is done by providing Compliance and Warranty statements.

The next subsections provide more information about Warranty and Compliance statements. But first, the notion of "source information" and "de-duplication instrument" are introduced.

9 Viewing the HMIS Unduplicated Count Problem as an optimization of privacy and utility is deemed unacceptable by VAWA, which happened after efforts to construct UID technologies had begun.

5.1 Basic terms

A UID technology involves transforming some source information collected from the Client at a Shelter, into a UID. The ideal is to have a UID uniquely associated with a Client such that no two UIDs relate to the same Client, and a Client has only one UID. Resulting UIDs are used by the Planning Office to identify the same Clients across Shelter visits by matching UIDs or by using a "de-duplication" instrument. These terms are further described in the next subsections.

5.1.1. Source information

Source information is something a Client holds or knows that forms the basis of the Client's UID. Common examples of source information are name, date of birth, and Social Security number. The source information is not the same as the UID, but instead is used as the basis for a method (or algorithm) that computes a UID from it. For example, an algorithm for constructing a UID could involve concatenating the Client's date of birth with the first 4 letters of the Client's first name. For example, Alice with birthdate 9/12/1960 would have UID "09121960ALIC."

In some cases, the source information may rely solely on volunteered verbal information from the Client. This is termed "non-verifiable" source information. Client information is just accepted as stated and is not checked against other credentials.

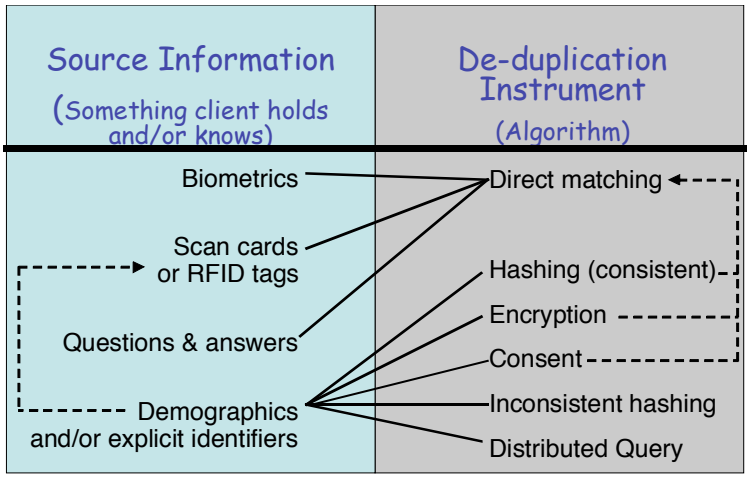
An interesting example of non-verifiable source information for UIDs is realized by allowing a Client to makeup her own UID (e.g., "100678") or by constructing a UID based on Client answers to simple questions like "your favorite color, song, and ice cream" or "which picture most resembles your first love." As long as the Client answers consistently across multiple Shelter visits, the UID will be associated with the Client. As long as the questions tend to evoke unique answers from each Client and Clients answer the same way on each visit, then the resulting UID will be uniquely associated with a Client.

"Verifiable" source information is something provided by the Client that can be confirmed. Examples include a driver's license or a fingerprint.

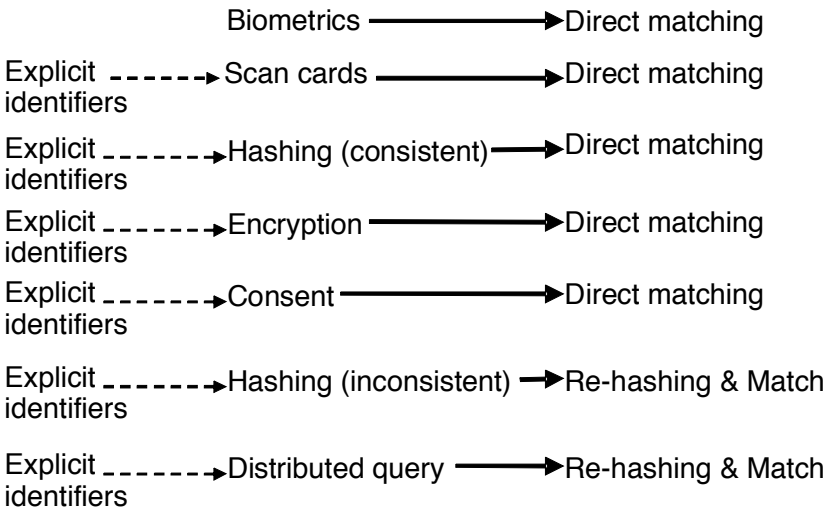
5.1.2. De-duplication instrument

A set of algorithms that describe how to construct a UID from source information and how to use UIDs to match Clients are collectively termed a "UID technology." Algorithms that construct UIDs may be as simple as concatenating parts of Client demographics, as demonstrated above, or more complicated as computing a unique value for a Client. Algorithms that match (or "de-duplicate") UIDs can be as simple as comparing two numbers, or as complicated as computing probabilistic matches.

Figure 15 (a) includes UID technologies already being considered. Source information includes biometrics, scan cards, question-and-answer, and the use of demographics and explicit identifiers. De-duplication instruments include directly matching (or linking) assigned, hashed, or encrypted values. Inconsistent hashing and distributed query are de-duplication instruments that do not simply match constructed UIDs. The UID technology termed "consent" merely checks whether Client permission was given. Each of these categories of UID technologies will be further described when they are assessed in Section 6. Figure 15 (b) shows some sample ways these are combined.



(a)



(b)

Figure 15. UID Technologies, assessed in Section 6, are broken down by source information and de-duplication instrument (a). Sample ways source and de-duplication instruments combine are shown in (b).

In Figure 15 (a), the solid line linkages between source information and de-duplication instruments show combinations of source information currently under consideration by some Planning Offices. Notice that biometrics, scan cards, question-and-answer, and demographic source information use direct matching to determine whether two UIDs match. Hashing, encryption, consent, inconsistent hashing, and distributed query all use demographics and/or explicit identifiers (e.g., Social Security number) as source information. The dashed lines in Figure 15 show secondary relationships. Demographics and explicit identifiers may be stored on scan cards. Hashed and encrypted values use direct matching for de-duplication. Consent also uses direct matching on demographics and/or explicit identifiers for de-duplication.

Assessing a UID technology involves producing Warranty and Compliance statements. Each of these are further described below.

5.2 Warranty statement (utility)

Given a Proposed Solution to the HMIS Unduplicated Count Problem, a Warranty shows that a reasonably accurate unduplicated accounting of records from Shelter Datasets is possible by the Planning Office. Below are fundamental issues that should be addressed by a Warranty.

The Warranty should demonstrate how de-duplication is done in the general case and identify the Proposed Solution's overall performance. Measures of accuracy should be included and cases that inflate or deflate the overall accounting should be addressed.

The behavior of the Proposed Solution using non-verifiable source information and verifiable source information should be examined. Particular attention should be given to the behavior of the Proposed Solution if Clients provide bad source information, such as purposeful name misspellings, wrong information, plausible differences in the information, or no information in part. Finally, consider the extent that the Proposed Solution can instill client confidence. This is particularly important when using non-verifiable source information because in these cases the system relies significantly on the cooperation of the Client.

Figure 16 lists considerations for Warranty statements.

WARRANTY (UTILITY) STATEMENT

Non-Verifiable source information	*If a UID is based on non-verifiable source information provided by the Client that is not truthful or is inconsistently used, what happens?*
Verifiable source information	*Can problems occur if the UID is based on verifiable source information? What if the information is not correct?*
Client confidence and trustworthiness	*The more Clients (and those who regularly intake Clients) trust the overall system and are encouraged to provide truthful information, the more likely Clients will actually provide truthful information. How trustworthy is the UID likely to be perceived by Clients (as well as by those who regularly intake Clients)? How would a lack of trust effect overall performance?*
Inflated accounting	*What are the circumstances under which de-duplication is likely to inflate the accounting? What are the circumstances in which a known Client is is not recognized (even if this does not actually inflate the count)? Explain the circumstances that generates these false negatives.*
Deflated accounting	*What are the circumstances under which de-duplication is likely to deflate the accounting? What are the circumstances in which a known Client is considered to be a different Client (even if this does not actually deflate the count)? Explain the circumstances that generates these false positives.*
Handling bad or missing input	*What is the effect of bad, incomplete, or missing source information on performance? How are these cases handled? (Note: "bad" information refers to accidental typing or other input mistakes.)*

Figure 16. Warranty Statements should seek to answer these questions.

5.3 Compliance statement (privacy)

Given a Proposed Solution to the HMIS Unduplicated Count Problem and publicly and readily available data and techniques, a Compliance Statement shows that the number of Clients who may be re-identified from the records in a Shelter Dataset is minimal. Below are fundamental issues that should be addressed by a Compliance Statement.

Consider any vulnerabilities the intimate stalker may exploit. Refer to Section 3.4.

Consider the ability to link Datasets, which include UIDs, to other available information in an attempt to re-identify Clients. Refer to Section 4.

"Dictionary attacks" should be considered. The idea of a dictionary attack is to generate UIDs for all possible source values and then see which results match UIDs stored in Dataset. Because the source that produced the UID is known, the information about the Client becomes known. Dictionary attacks assume the attacker has access to the UID technology and knowledge of what source information is used.

Here is an example of a dictionary attack. Assume a UID technology uses encryption to compute a number from the Client's Social Security number. Even without knowing how encryption works (which will be discussed in Section 6.3), one can use a dictionary attack to learn the source information that generates a UID. Figure 17 shows an encryption method that when given a Social Security number produces a UID.

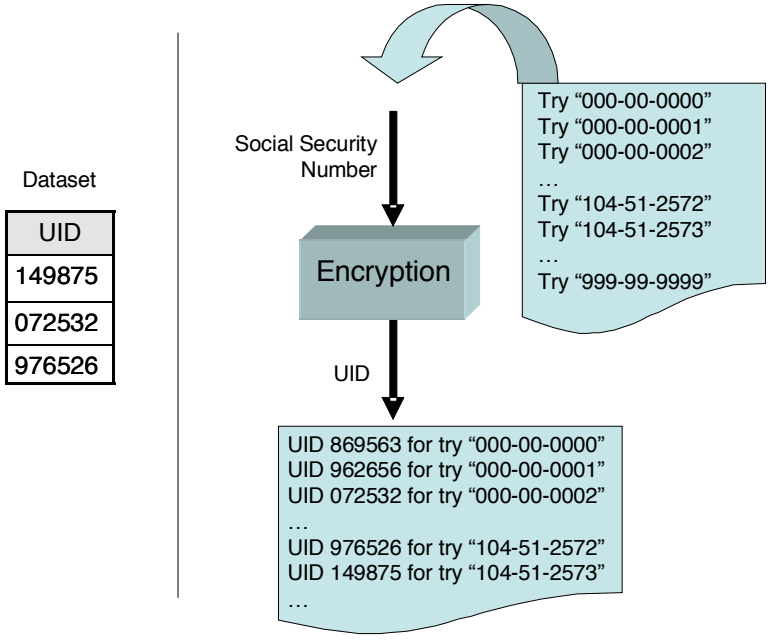


Figure 17. Example of a dictionary attack. Given a Dataset having UIDs 149875, 072532, and 976526, the knowledge that UIDs are encryptions of Social Security numbers, and access to the encryption function, a dictionary attack allows the UIDs to be learned by trying all possible Social Security numbers and seeing which Social Security numbers encrypt to the observed UIDs. In the example above, the Social Security number 104-51-2573 encrypts to 149875.

Suppose the Dataset contains the UIDs: 149875 and 072532. We can use a dictionary attack to learn the Clients' Social Security numbers that produced those UIDs by trying all possible 9-digit values and seeing which 9-digit Social Security numbers produce the UIDs that appear in the Dataset. As noted in Figure 17, the Social Security number "104-51-2572" produced UID 149875 and the Social Security number "000-00-0002" produced the UID 072532, so the Clients have the Social Security numbers "104-51-2572" and "000-00-0002," respectively.

A dictionary attack can be combined with linking to re-identify Clients by name. Assume a UID technology encrypts a combination of a Client's date of birth and gender to produce a UID. These same values also appear in the voter list (see Figure 10). So, computing UIDs for every voter in the voter list allows us to match UIDs in the Voter list to UIDs in the Dataset to re-identify Clients by name.

Of course, a dictionary attack can take a long time to compute. The example below reports the time needed for a dictionary attack to exhaust all possibilities using larger numbers on today's computers. To exhaust all possible 9-digit Social Security numbers (which requires 30 bits of storage) takes about 4 seconds. But to exhaust 36-bit numbers, which can store up to 11 digits, takes about 8 minutes. Using larger numbers requires more bits to store values; and, as the number of bits increases, the amount of time needed to test all possible values grows exponentially. Clearly, it is advantageous to use large numbers, as appropriate, in order to reduce the success of a dictionary attack.

Example (Exhausting Large Numbers)

A simple program that counts from 0 to the largest integer that can be stored in x bits simulates a dictionary attack on x-bit numbers because it exhausts all possible values. Timing the execution of this program gives an estimate of the minimum time needed for a dictionary attack based on numbers requiring x-bits of storage. It ran under Java 1.5 on a 2003 vintage machine (Dell PrecisionTM 650 workstation having an Intel® Xeon™ Processor at 3.06 GHz, 512KB L2 cache, and 4GB RAM). Counting all possible Social Security numbers (0 to 999,999,999) took 4 seconds and used 30 bit numbers. Figure 18 shows the results of counting all integer values from 0 to the largest number that could be stored in 28 to 47 bits. Resulting times ranged from 1 second to 1021463 seconds (or 12 days), respectively.

■

bits	seconds
28	1
29	3
30	7
31	15
32	31
33	62
34	124
35	249
36	499
37	998
38	1996
39	3993
40	7986
41	15963
42	31926
43	63888
44	127725
45	255463
46	510774
47	1021463

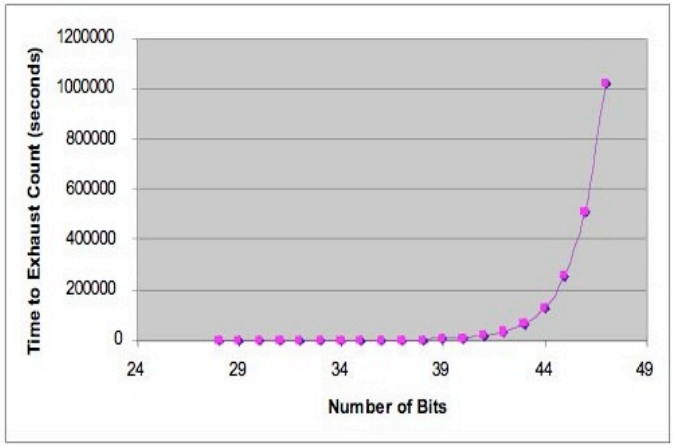


Figure 18. Experimental results of time needed to exhaust 28 to 47 bit numbers. Time is the number of seconds needed to count from 0 to the maximum value stored in 28 to 47 bits on today's computers.

Figure 19 shows an equation that characterizes the values reported in Figure 18 and predicts the time needed for larger values which consume more bits. The variable x is the number of bits for the maximum value and y is the number of seconds needed to count from 0 to the maximum value. (The correlation coefficient R^2=0.9989 provides a measure of fitness based on a linear regression of the log of the actual and predicted values.) There are some interesting surprises. To exhaust 46-bit numbers capable of storing a concatenation of a typical person's Social Security number, month, day and year of birth, and gender in 15-digits takes about 6 days. To exhaust 64-bit numbers, which can store up to 20 digits, takes about 89 centuries!

$y=2.08^{x-28}$	28 bit	1 second
	30 bit	4 seconds
	35 bit	2.8 minutes
	40 bit	1.8 hours
	45 bit	3 days
	50 bit	3.8 months
	55 bit	12.3 years
	60 bit	4.8 centuries
	64 bit	89.4 centuries
	65 bit	186 centuries
	100 bit	25,220,489,437,291 centuries
	128 bit	20,301,442,123,378,100,000,000 centuries

Figure 19. Predicted time to exhaust x bit numbers. On left is an equation that characterizes the values reported in Figure 18. The variable x is the number of bits and y is the number of seconds needed to count from 0 to the maximum integer value that can be stored in x bits. On the right, are predictions of the time needed to exhaust all integers able to be stored in x bits.

One way to improve the wait time further is for multiple computers to work on different ranges of values at the same time thereby dividing the overall time across the machines. Figure 19 reports that exhausting 55-bit values would take one machine about 12.3 years. Conversely, if 3000 machines worked collectively in parallel, they would need no more than 36 minutes to exhaust all 55-bit values. While using thousands of machines may seem impossible, a single spammer reportedly used more than 2000 machines to send spam. The machines were idle on the Internet and physically located at different sites around the world, so no one realized that the spammer had compromised their operating systems and was running his own programs. Similarly, an attacker could co-opt thousands of machines to perform a dictionary attack.

The system using a UID technology should maintain client secrets even if the Planning Office mounts a dictionary attack. If the Planning Office has direct access to its own copy of the hash function, it can mount a dictionary attack in order to learn private information. Alternatively, the Planning Office can pad the values submitted to shelters with values of its own choosing in an attempt to learn private information. These kinds of accesses must not allow the Planning Office to learn private client information.

Beyond the intimate stalker threat, linking attacks, and dictionary attacks, an assessment should also examine to what extent the algorithm for producing the UID can be "reverse engineered." For example, given the following list of UIDs: 09121960ALIC, 10251974JANE, …, one can conclude that the UID is constructed by concatenating the month, day and year of birth with the first 4 letters of the first name. In this example, observing the UIDs revealed the method for constructing the UIDs. Given a Client's name and date of birth, the Client can be found in the Dataset.

A Compliance Statement should also identify any new legal or technical privacy risks that may be introduced based on the existence of the Proposed Solution's UID. This is considered "exposure."

Here is an example. If a Proposed Solution uses fingerprints as the source information for UIDs in such a way that a UID database is a fingerprint database, then the existence of the resulting database of Client fingerprints may be useful to law-enforcement. The existence of the database's usefulness to third parties poses new privacy concerns for Clients and thereby, increases exposure.

Figure 20 lists considerations for Compliance statements.

COMPLIANCE (PRIVACY) STATEMENT

Intimate Stalker	*What vulnerabilities exist for the intimate stalker?*
Re-identification: Linking	*What vulnerabilities exist for re-identification of UIDs (and Dataset) using data linkage on UIDs? What is the identifiability of the Dataset?*
Re-identification: Dictionary Attack	*What vulnerabilities exist for re-identification of UIDs (and Dataset) using a dictionary attack on UIDs?*
Re-identification: Reversal	*What is involved in reverse engineering the UID construction method?*
Exposure	*What legal or technical risks or liabilities may be introduced based on the existence of the resulting database or UID technology?*

System Trust *The overall system consists of intakers, who enter Client information, insiders, who have access to Client information for a variety of meritorious reasons, and the Shelters and Planning Offices themselves. Which parties are heavily trusted?*

Figure 20. Compliance Statements should seek to answer these questions.

5.4 Other factors

There are many other factors that may contribute to a decision of which UID technology to use that are not part of the assessment. Among these are trust and economics. Where trust is placed differs among Proposed Solutions. Some solutions put more trust in the Shelters (e.g. distributed query), in the Clients (e.g., UID technologies using non-verifiable source information), or in the Planning Offices (e.g. consent).

Another key factor can be the economics of constructing, installing and maintaining the system. Some states are constructing systems for administrative oversight of social programs, so weaving HMIS requirements into those systems can be cost-effective, but doing so, may dictate the use of a particular UID technology.

Another factor can be available technical expertise.

While all these kinds of factors are important to the decision-making process, they are excluded from demonstrating the worthiness of the Proposed Solution. Warranty and Compliance statements demonstrate utility and privacy protection independent of these concerns.

5.5 Putting the pieces together

The goal of this section is to provide guidance on what should constitutes an assessment. The goal of the next section is to provide some overall assessments of 8 categories of technologies.

In summary, an assessment is a thorough review and analysis of a Proposed Solution that should be completed before a Proposed Solution is put to real-world use. Assessing a Proposed Technology requires a confluence of technology, policy, and sometimes law. Groups of people lacking the proper expertise or not focused on key issues pertinent to Warranty and Compliance issues can lead to poor assessments. The goal of this section and the next is to help those engaged in this process to ask themselves the right questions and to identify the right kinds of expertise needed. Along these lines, the following two recommendations are made.

Recommendation #11: Given a Proposed Solution, a person skilled in statistical, computational and/or legal principles, as appropriate, should certify in writing that the Proposed Solution has a minimal risk of re-identification when the solution is considered with other publicly and readily available information and techniques. Such writing should address vulnerabilities for inappropriate re-identifications by various categories of insiders. This is termed a "Compliance Statement" and should be made available for inspection.

Recommendation #12: Given a Proposed Solution, a person skilled in statistical and/or computational principles, as appropriate, should certify in writing that the Proposed Solution provides a reasonably accurate unduplicated accounting of client visit patterns to shelters within the regional setting it is to be deployed. Such writing should include possible false match and missed match rates. This statement is termed a "Warranty" and should be made available for inspection.

5.6 Privacy, not computer security

One word about computer security before continuing. This writing relates to data privacy concerns and not to computer security issues. It is assumed that any Proposed Solution operates in a computational environment having adequate computer security to authenticate users, limit access, combat intrusions and prevent eavesdropping. This writing does not address computer hacking, break-ins, viruses, or unauthorized computer users, because such issues appear to be adequately addressed with commercial computer security solutions. For general reference, see Pfleeger [21].

Instead, this writing addresses ways to limit authorized users from doing unauthorized tasks with available data. For example, the intimate stalker either has access to the Dataset already or obtains assistance from someone with access. Linking Dataset to other available information in order to re-identify Clients can only be done by someone with access to the Dataset. If someone does break-into the computer system and gains access to Dataset to attempt these things, the safeguards described in this writing will thwart their efforts. Described in this manner, these safeguards provide some privacy protection even in the face of a computer security breach. But more generally, these safeguards thwart unwanted activities by most of those who work with Dataset regularly.