8. PrivaMix, a UID Technology for VAWA

Notwithstanding the stringent re-identification standard of VAWA, this section introduces a provable privacy-preserving UID technology ("PrivaMix") for gathering service utilization patterns of domestic violence shelter clients while guaranteeing the privacy of shelter clients.

8.1 The PrivaMix approach

PrivaMix combines a form of inconsistent hashing (Section 6.7) with distributed query (Section 6.8). The same client gets different UIDs at different Shelters and can get the same UID at the same Shelter. Inconsistently assigning UIDs across Shelters in this way thwarts the linking and dictionary attack pitfalls noted earlier (in Section 4). In order for the Planning Office to then compute the unduplicated accounting of Clients across Shelters, a distributed network of Shelter machines run computations on each other's UIDs to relate which UIDs relate to the same Clients. This is done without identifying Client information to the Shelters or Planning Office. Described in this way, there are three major phases: (1) inconsistent assignment of UIDs to Clients; (2) delivery of visit information to the Planning Office; and, (3) the de-duplication of UIDs by Shelters. The next subsections further describe the approach taken in each of these phases.

8.1.1. Inconsistent assignment of UIDs

Shelters share the "PrivaMix function," which is a strong one-way function (Section 6.2) used to assign inconsistent UIDs across Shelters and reporting periods. Each Shelter customizes the PrivaMix function by using it with a privately held value the Shelter selects. This is typically a large value (perhaps 512 bits or larger) and usually selected randomly and unknown to Shelter personnel. The PrivaMix function combines the Shelter's private value with a Client's source information to generate a unique UID for the Client at the Shelter. Because different Shelters have different private values, the same Client will have different UIDs at different Shelters. Together, the Shelter's private value and the Client's source information combine to determine the Client's UID.

The Client's source information can be any appropriate information specific to the client (see Section 5). For discussion in this section, references to Client source information are to the Client's Social Security number (SSN), but using other source information is possible[12] without loss of performance or protection.

Because the PrivaMix function is strong, it is infeasible to reverse the process and learn the Client's source information (e.g., SSN) from a UID. Because each Shelter customizes its PrivaMix function by randomly selecting a secretly held large random value, the Planning Office cannot feasibly exhaust all possible combinations, thereby thwarting a dictionary attack by the Planning Office (Section 5.3). Because the UIDs are not associated with any other data, linking on UIDs is not possible.

12 In fact, *date of birth* and *part of the first name* were used as source information in the real-world experiment reported in Section 10.

For improved privacy protection, Shelters can select different private values at each HMIS reporting session, thereby thwarting the ability to link HMIS data across reporting periods if such linking is undesired.

8.1.2. Delivery of Visit Information to the Planning Office

Once Shelters generate UIDs for Clients, they forward Client visit information along with UIDs to the Planning Office for de-duplication using a secure means (e.g., overnight delivery of a CD or over a secure Internet connection). Each record in the Dataset relates to a Client at the Shelter and includes the Client's UID. At the conclusion of this step, the Planning Office has the visit information it needs, but does not know which clients across Shelters may be the same. The Planning Office cannot de-duplicate the visits without additional processing by the Shelters.

8.1.3. De-duplication of UIDs by Shelters

In order to determine which UIDs relate to the same clients, a network of Shelter machines perform a computation on each other's UIDs. Each Shelter applies the PrivaMix function on the combination of its private value and the UIDs from the other Shelters; we term this "mixing." After all Shelters finish mixing, those results, or "complete mixes," will only be the same for those UIDs whose original Client source information was the same. These UIDs refer to the same Client.

To participate in mixing, each Shelter and the Planning Office has a computer on a reasonably secure network we term the "PrivaMix Network." The "PrivaMix Protocol" dictates communications between Shelter machines and the Planning Office machine. The purpose of the communication is have each Shelter mix (apply their customized PrivaMix function) the UIDs of all the other Shelters and keep track of which mixed results match which original UIDs. It also important that each Shelter only mix a UID once. When the protocol concludes, the UIDs across Shelters that relate to the same Client will have the same multi-mixed value, so the Planning Office can identify which records belong to the same Clients.

There are many possible functions that can serve as a PrivaMix function. While detailed requirements for a PrivaMix function appear in Section 8.3, one property is essential. A PrivaMix function must have the "commutative property" [28] in order for the Planning Office to identify which UIDs across Shelters relate to the same Client. The idea of the commutative property is that if the original source information is the same, the multi-mixed UIDs will be the same, regardless of the order in which the mixing occurs.

Before looking at two example, the following recommendations relate to using PrivaMix as a UID technology.

Recommendation #20: When using PrivaMix as a UID technology, care should be taken to avoid multiple Shelters from having the same private value. The Shelter's private value customizes the PrivaMix function to the Shelter. If multiple Shelters inadvertently have the same private value, then those Shelters assign exactly the same UIDs to the same clients. In most uses of PrivaMix, the UIDs will only be used for one-time mixing. In these cases, it is okay if Shelters inadvertently select the same private value though the likelihood of such should be rare.

Recommendation #21: When using PrivaMix as a UID technology, if the visit data is transmitted to the Planning Office over the PrivaMix network of Shelter and Planning Office machines, then appropriate computer security standards for the storage of Client information should be enforced because these machines contain Client source and visit information.

Example.
Here is an example with three Clients visiting two Shelters using integer multiplication instead of a strong one-way PrivaMix function. Because integer multiplication can be easily reversed using division, it is not "strong" and therefore cannot be used as a PrivaMix function. However, integer multiplication is commutative, so this example provides an overall demonstration of mixing.

In the first step, Shelters assign inconsistent UIDs to Clients. Figure49 provides a summary. Figure 50(a) shows three distinct Clients visiting two Shelters. A personal number appears with each Client. This is the numeric representation of the Client's source information.[13] Clients have source information 3, 7, and 11. The Client having source information 3 appears at both Shelters. The Clients having source information 7 and 11 appear at Shelter 1 and Shelter 2, respectively.

The Shelters have private values. Shelter 1's private value is 13. Shelter 2's private value is 23. In this example, integer multiplication is the function used. So, each Shelter multiplies its private value by its Client's source information to assign a UID to its Client. Figure 50(b) shows that Shelter 1 assigns the Client whose source information is 3, the UID 39 because 3 multiplied by 13 is 39. Similarly, the Client whose source information is 13 gets UID 91. Shelter 2 assigns the Client whose source information is 3, the UID 69 because 3 multiplied by 23 is 69. Similarly, the Client whose source information is 11 gets UID 253. Notice that the Client whose source information is 3 gets UID 39 at Shelter 1 and UID 253 at Shelter 2.

In the second step, Shelters forward visit information to the Planning Office. This information has a record for each Client and the Client is denoted by the assigned UID. Figure 51 depicts the flow of information from the Shelters to the Planning Office. Shelter 1 forwards two records, one for a Client with UID 39 and another for a Client with UID 91. Similarly, Shelter 2 forwards two records, one for a Client with UID 69 and another for a Client with UID 253. The Planning Office stores the UIDs from each Shelter, along with the other visit information. In Figure 51, the other visit information appears as "UDE," representing the other Universal Data Elements (see Section 3.5). At this time, the Planning Office knows there are four visits, but does not know how many Clients account for the four visits.

13 As discussed earlier in Section 5.1.1, a Client's source information may be a Social Security number, name, or a combination of other values specific to the Client. Whether the source information is a number or text, the computer represents the information as a number. Therefore, in this section, it is proper to think of the Client's source information as a numeric value, even though its print notation may include letters and symbols.

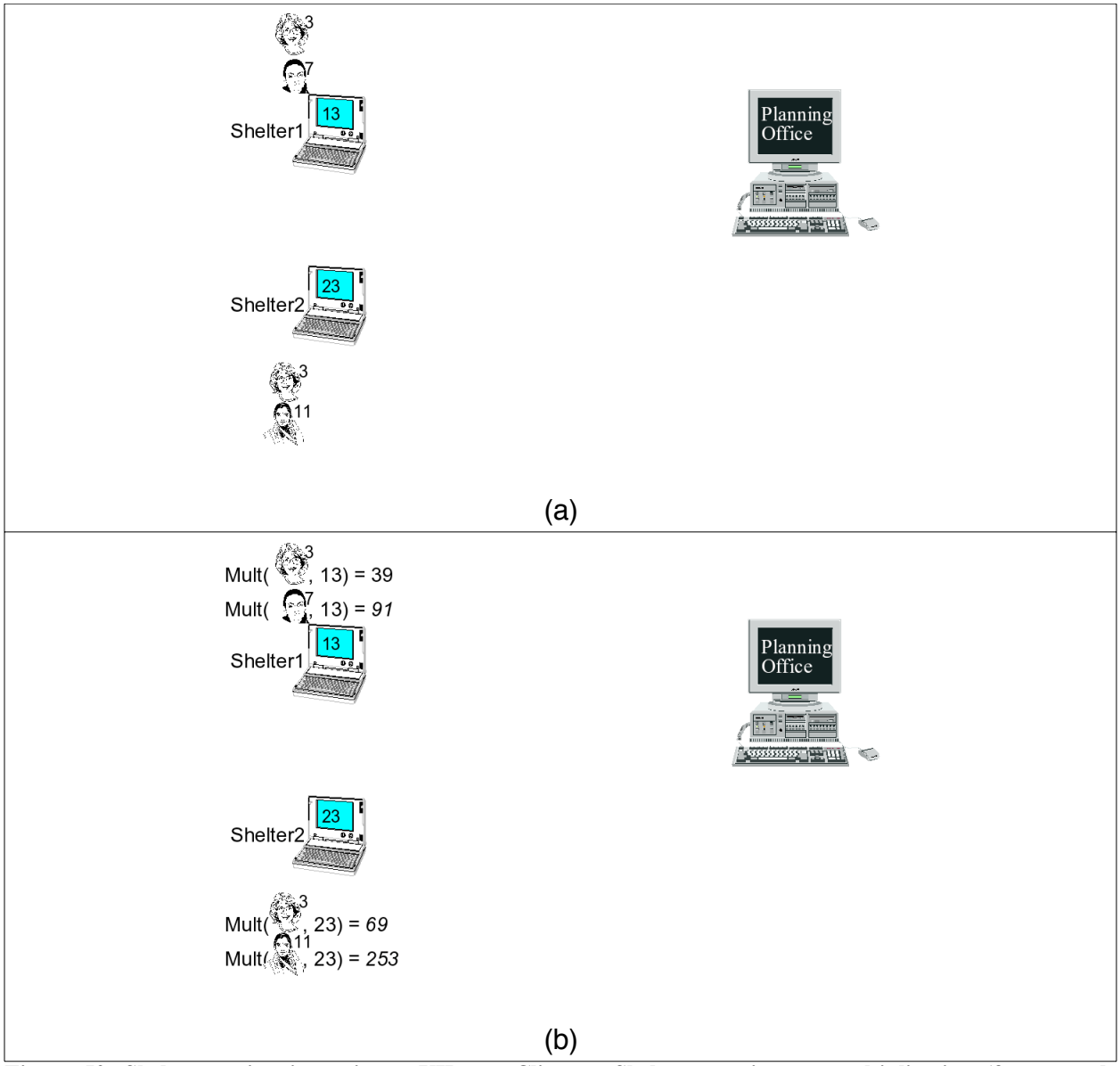


Figure 50. Shelters assign inconsistent UIDs to Clients. Shelters use integer multiplication (for example purposes only) to assign UIDs. In (a), Clients having source information 3 and 7 visit Shelter 1 and Clients having source information 3 and 11 visit Shelter 2. Shelter 1 has a private value of 13 and Shelter 2 has a private value of 23. In (b), Shelter 1 assigns UID 29 to the Client whose source information is 3 and 91 to the Client whose source information is 7. Shelter 2 assigns UID 69 to the Client whose source information is 3 and 253 to the Client whose source information is 11.

In the third step, Shelters de-duplicate the UIDs so the Planning Office can learn which visits relate to the same Clients. Figure 52 provides a step-by-step depiction. The Planning Office received UIDs 69 and 253 from Shelter 2. It forwards these to Shelter 1 for mixing; see Figure 52(a). Shelter 1 multiplies each UID by its private value 13. As shown in Figure 52(b), Shelter 1 returns the values 897, which is 13 multiplied by 69, and 3289, which is 13 multiplied by 253. The Planning Office stores the results received from Shelter 1. Because there are no other Shelters, these values are the complete mixes for the UIDs 69 and 253 received from Shelter 2.

The Planning Office received UIDs 39 and 91 from Shelter 1. It forwards these to Shelter 2 for mixing; see Figure 52(c). Shelter 2 multiplies each UID by its private value 23. As shown in Figure 52(d), Shelter 2 returns the values 897, which is 23 multiplied by 39, and 2093, which is 23 multiplied by 91. The Planning Office stores the results received from Shelter 2. Because there are no other Shelters, these values are the complete mixes for the UIDs 39 and 91 received from Shelter 1.

The Planning Office now learns which visits relate to the same Clients by examining which complete mixes are the same. As shown in Figure 53, two records have the same complete mix 897. One is the record with UID 39 from Shelter 1. The other is the record with UID 69 from Shelter 2. These two records relate to the same Client.

This example successfully de-duplicated the UIDs because of the commutative property of integer multiplication (used for exemplary purposes only). The order in which the multiplication occurs does not matter. When the Client whose source information is 3 visited Shelter 1 and the resulting UID 39 was mixed by Shelter 2, the result was: (3 * 13) * 23 = 897. When the same Client visited Shelter 2 and resulting UID 69 was mixed by Shelter 1, the result was: (3 * 23) * 13 = 897. Multiplying the source information by 13 and then 23 yields the same result as multiplying the source information by 23 and then 13.

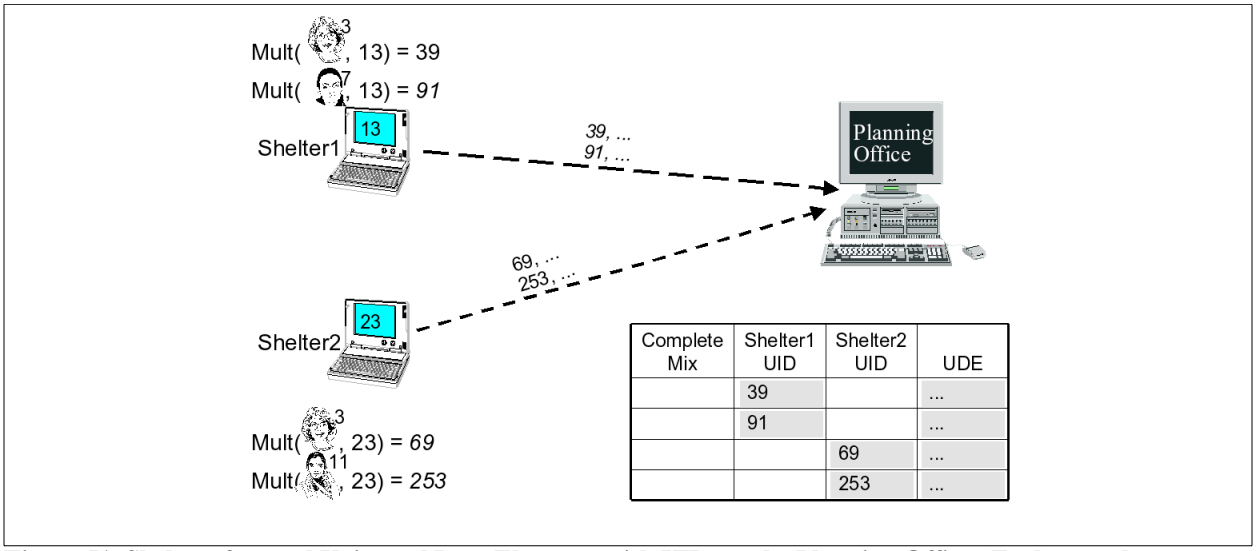


Figure 51. Shelters forward Universal Data Elements with UIDs to the Planning Office. Each record represents a Shelter visit. There are four visits, one for each of the UIDs 39, 91, 69, and 253.

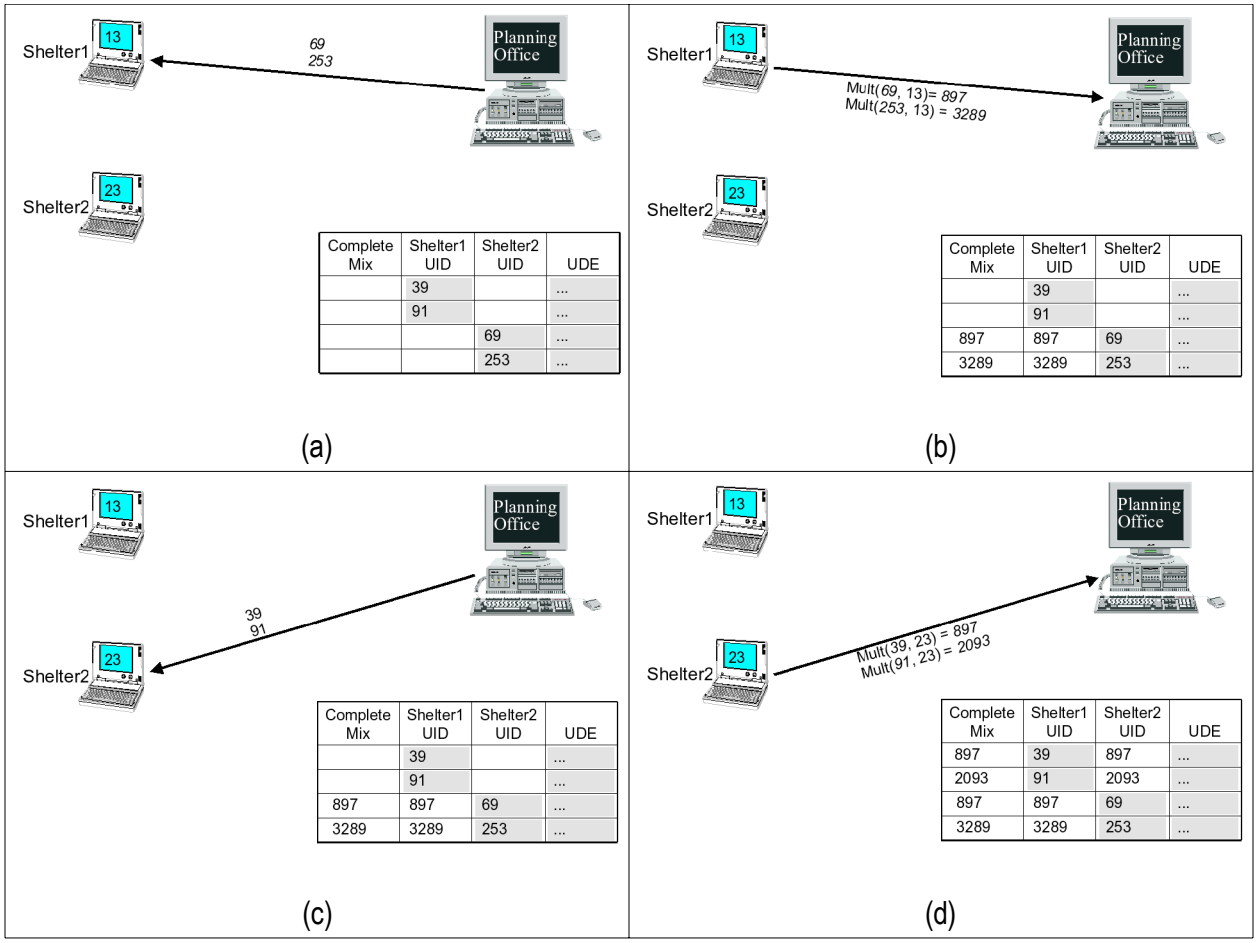


Figure 52. Mixing to de-duplicate UIDs. In (a), the Planning Office forwards the UIDs received from Shelter 2 to Shelter 1. In (b), Shelter 1 returns the mixed results to the Planning Office. In (c), the Planning Office forwards the UIDs received from Shelter 1 to Shelter 2. In (d), Shelter 2 returns the mixed results to the Planning Office.

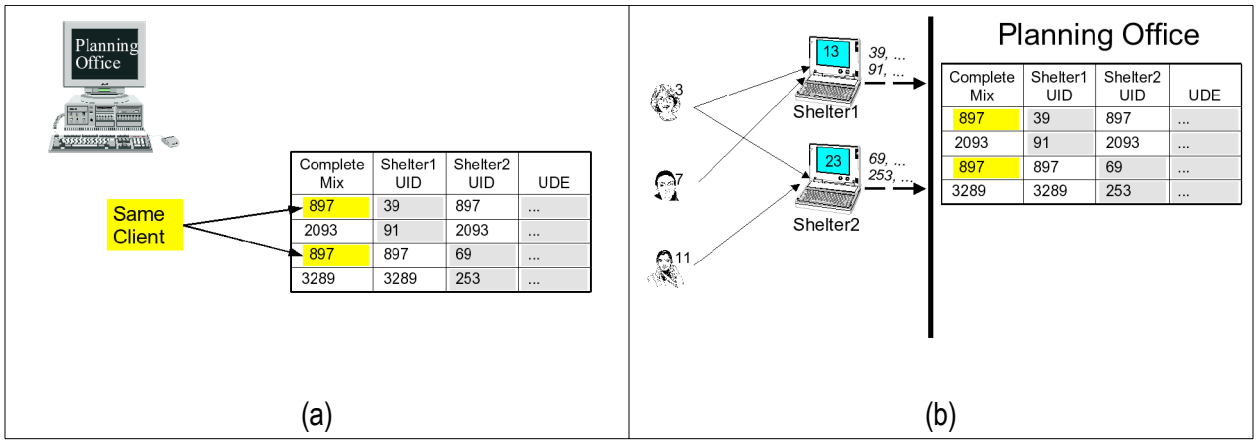


Figure 53. Planning Office learns which records relate to the same Client. Two records have the same complete mix, 897, revealing that two of the records relate to the same Client.

Example.
Here is another example with three Clients visiting two Shelters. The example uses symbolic notation to denote the use of the PrivaMix function. Figure 54 shows Client 1 visiting both Shelters, where Client 2 only visits Shelter 1 and Client 3 only visits Shelter 2. From Client Social Security numbers (SSN), Shelter 1 produces ax4 as Client 1's UID and 1804 as Client 2's UID. Shelter 2 produces b3s7 as Client 1's UID and ghre as Client 3's UID. The generation of the UIDs depicted in Figure 54 concludes step 1, the assignment of inconsistent UIDs.

Each Shelter provides the Planning Office with the Client UID and associated Universal Data Elements for each Client that visited the Shelter. Figure 55 shows the compiled results at the Planning Office. Shelter 1 had two Clients whose UIDs are ax4 and 1804. Shelter 2 had two Clients whose UIDs are b3s7 and ghre. At this time, the Planning Office knows information about four Client visits, but does not know the total number of distinct Clients or which Clients at Shelter 1 also visited Shelter 2. This concludes Step 2, delivery of visit information to the Planning Office.

Step 3 involves de-duplicating the UIDs; see Figure 56. The Planning Office sends the UIDs received from Shelter 2, b3s7 and ghre, to Shelter 1 for mixing. Similarly, it sends the UIDs from Shelter 1, ax4 and 1804, to Shelter 2 for mixing. These appear in Figure 56(a). The result of Shelter 1's mixing of b3s7 is H2732 and of ghre is 0yfh02, as shown in Figure 56(b). Similarly, the result of Shelter 2's mixing ax4 is H2732 and of 1804 is nw450, as shown in Figure 56(b). Therefore, the results from mixing are H2732 and 0yfh02 from Shelter 1 and H2732 and nw450 from Shelter 2. Finally, the Planning Office associates the mixed values to the original UIDs to learn that ax4 and b3s7 relate to the same Client, but UIDs 1804 and ghre relate to different and distinct Clients; see Figure 56(c). The Client whose SSN was the same has the same complete mix, notwithstanding the order of mixing.

Both examples provided so far have involved only two Shelters. A larger example appears at the end of Section 8.2. It better demonstrates the scalability of mixing.

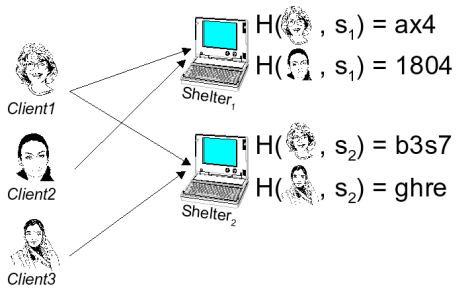
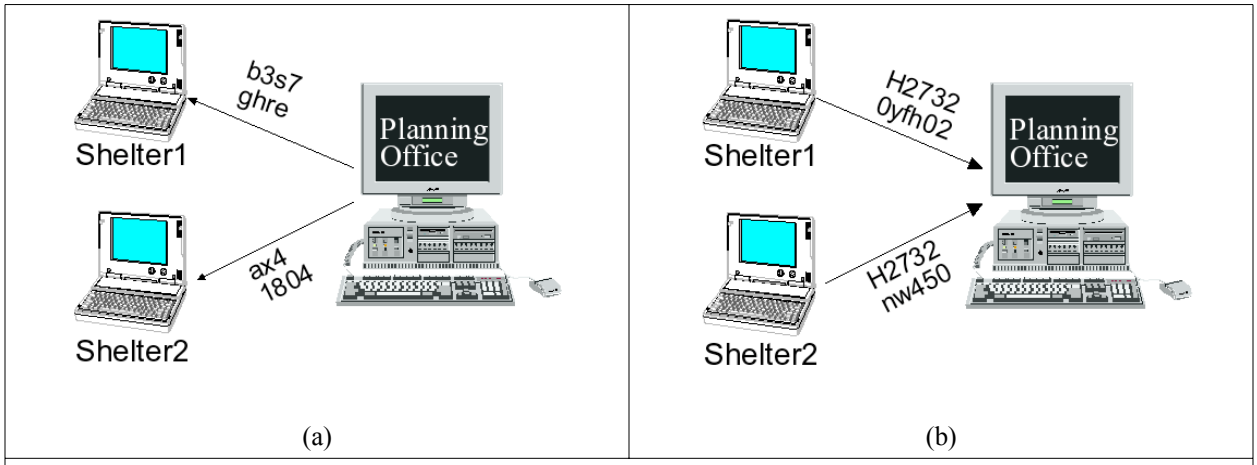


Figure 54. Each Shelter generates a UID for each Client that visits the Shelter using the Client's SSN and the a private value v_i held at the Shelter. The UID is generated using a strong, commutative PrivaMix function F. Each shelter customizes the use of the PrivaMix function because each Shelter has a different private value v_i.

Shelter	UID	UDE
Shelter 1	ax4	…
Shelter 1	1804	…
Shelter 2	b3s7	…
Shelter 2	ghre	…

Figure 55. The Planning Office compiles a table of information provided by the Shelters of Client visits. UDE refers to the Universal Data Elements that comprise the Dataset. UIDs are the Client UIDs issued at each Shelter.



Complete Mix	Shelter1 UID	Shelter2 UID
H2732	ax4	b3s7
nw450	1804	
0yfh02		ghre

(c)

Figure 56. PrivaMix Protocol for de-duplication of UIDs: (a) the Planning Office forwards UIDs to be mixed; (b) Shelters send back the mixed results; and (c) Planning Office compares complete mixes to original UIDs to learn that one Client visited both Shelters and two Clients visited one Shelter each.

8.2 Technical presentation

This subsection provides a formal description of the PrivaMix approach. Non-technical readers may advance to the claims subsection which follows this subsection without loss of understanding.

Strong Function.
A one-way (or strong) function has the property that if **F** is a strong function, $\mathbf{F}(x)=y$ computes in polynomial time, and it is computationally infeasible to compute its inverse $\mathbf{F}^{-1}(y)=x$.

The "Commutative Property".
Each Shelter *j* hashes a Client's source information (SSN_i) using the Shelter's private value s_j and a strong function **F** such that: $\mathbf{F}(SSN_i, s_j) = UID_{ij}$. Benaloh and de Mare [28] showed that the Shelters' private values can be chosen appropriately so that

$\mathbf{F}(\mathbf{F}(SSN_a, s_j), s_k) = \mathbf{F}(\mathbf{F}(SSN_b, s_k), s_j)$, only if $SSN_a = SSN_b$.

8.2.1. Formal Description

Definitions
Let $S = \{S_1, S_2, ..., S_n\}$ be *n* Shelters having private values $s_1, s_2, ..., s_n$, respectively.
Let $C = \{C_1, C_2, ..., C_m\}$ be *m* Clients having source information $SSN_1, SSN_2, ..., SSN_m$, respectively.
Let *P* be the Planning Office .
Let **F** be a strong function with the commutative property that generates the UID.
We write: $\mathbf{F}(SSN_i, s_j) = UID_{ij}$

Problem Statement
For each SSN_i, *P* learns (S_j, UID_{ij}) without re-identifying Client *i* or learning SSN_i.

3-Step PrivaMix Protocol (also known as "PrivaMix")
Step 1. Shelters compute UID_s.
For each C_i visiting Shelter S_j, Shelter S_j computes $\mathbf{F}(SSN_i, s_j) = UID_{ij}$

Step 2. Data to Planning Office.
P receives a table having attributes $\{S_j, \mathbf{F}(SSN_i, s_j), UDE_{ij}\}$ where UDE are values of the Universal Data Elements for Client C_i at Shelter S_j.
P computes multi-set[14] $\boldsymbol{H} = \{ \mathbf{F}(SSN_i, s_j) : \mathbf{F}(SSN_i, s_j) = UID_{ij}\}$

Step 3. De-duplicate UIDs.
For $k = 1$ **to** *n* **do**:
P computes multi-sets:

$$\boldsymbol{H}_1\{x : x=\mathbf{F}^{a}(\mathbf{F}(SSN_i, s_j)...) \text{ where } x\in\boldsymbol{H}, a\geq 0, j\neq k\}$$

$$\boldsymbol{H}_2\{x : x=\mathbf{F}^{a}(\mathbf{F}(SSN_i, s_j)...) \text{ where } x\in\boldsymbol{H}, a\geq 0, j=k\}$$

P sends $\boldsymbol{H}_1$ to S_k
S_k sends *P*:

$$\boldsymbol{H}_{Sk}\{\mathbf{F}(x, s_k) : x=\mathbf{F}^{a}(\mathbf{F}(SSN_i, s_j)...) \text{ where } x\in\boldsymbol{H}_1\}$$

P computes $\boldsymbol{H} = \boldsymbol{H}_{Sk} \cup \boldsymbol{H}_2$

14 A multi-set is a set in which an element may appear multiple times. $\boldsymbol{H}$, $\boldsymbol{H}_1$, $\boldsymbol{H}_2$, and $\boldsymbol{H}_{Sk}$ are multi-sets.

Example.
Here is an example involving three Clients and three Shelters. Figure 57 shows Client 1 visiting Shelters 1 and 2, where Client 2 only visits Shelter 2 and Client 3 only visits Shelter 3. Each Shelter has a strong function (**F**), having the commutative property. **F** is customized to each Shelter *j* by the Shelter' private value s_j. Each Shelter uses its customized function to compute a UID for each Client *i* using Client *i*'s source information (SSN_i). The value $\mathbf{F}(SSN_i, s_j)$ is UID_{ij} for Client *i* at Shelter *j*.

Each Shelter provides the Planning Office with the UID and associated Universal Data Elements of each Client that visited the Shelter. Figure 58 shows the compiled results in a table produced by the Planning Office.

To de-duplicate UIDs, the Planning Office sends a set of values to each Shelter to mix (apply its private value using function **F**). In Figure 59, the Planning Office sends the original UIDs from Shelter 2 and Shelter 3 to Shelter 1 for mixing. Shelter 1 sends the mixed results back to the Planning Office ending the first round of de-duplication.

In the next round of de-duplication, the Planning Office sends values to Shelter 2 for mixing; see Figure 60. The Planning Office sends the original UID from Shelter 1 for Client 1. It also sends the mixed value from Shelter 1 of the UID originally provided by Shelter 3. Together, these are values from other Shelters not yet mixed by Shelter 1.

In the final round of de-duplication, the Planning Office sends values to Shelter 3 for mixing; see Figure 61. These values are those UIDs that originated from Shelters 1 and 2 but in their current mixed form.

The rounds of de-duplication continue for as many Shelters involved. There is one round per Shelter. Each Shelter receives the UIDs originally contributed from the other Shelters, but the value used is either the original or the mixed value. When the rounds are completed, each Shelter has applied the function with its private value once on each UID, though the order in which the mixing occurred varied. The commutative property of the function guarantees that the final mixed values will be the same only if the original source information was the same. Figure 62 summarizes the findings from this example.

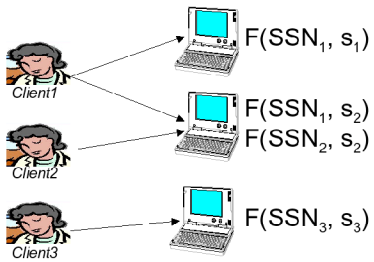


Figure 57. Each Shelter computes a UID for each Client using a strong function F, which has the commutative property. F is customized to each Shelter *j* using v_j. Client *i*'s source information is denoted as SSN_i.

Shelter	UID	UDE
Shelter 1	F(SSN1, s1)	…
Shelter 2	F(SSN1, s2)	…
Shelter 2	F(SSN2, s2)	…
Shelter 2	F(SSN3, s3)	…

Figure 58. Planning Office knowledge after receiving UIDs and Universal Data Elements from Shelters.

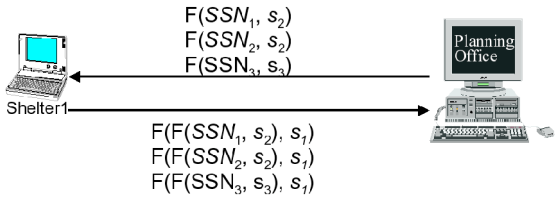


Figure 59. Round 1 of de-duplication. Original UIDs from Shelters 2 and 3 are sent to Shelter 1 for mixing.

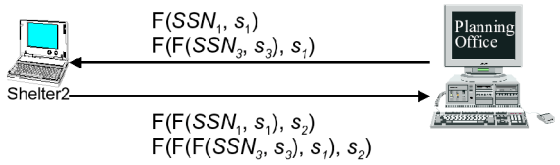


Figure 60. Round 2 of de-duplication. Shelter 2 receives the original UID from Shelter 1 and the mixed UID originating from Shelter 3.

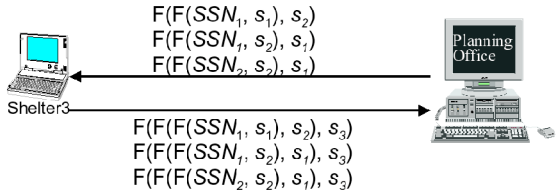


Figure 61. Round 3 of de-duplication. Shelter 3 receives the current mixed values of the UIDs that were originally contributed by Shelter 1 and Shelter 2.

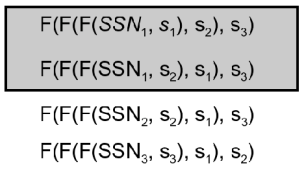


Figure 62. The final results from de-duplication. Values that are the same have the same source information and therefore are considered to relate to the same Client.

Below are variations to the generic PrivaMix approach. Each is described in detail following the list.

8.2.2. PrivaMix Variation 1: Shelters mix among themselves, without the Planning Office.
8.2.3. PrivaMix Variation 2: Shelters check that UIDs are legitimate
8.2.4. PrivaMix Variation 3: Matching UIDs to Universal Data Elements
8.2.5. PrivaMix Variation 4: Providing aggregate count distributions, not Client-level data
8.2.6. PrivaMix Variation 5: Anonymizing client-level data
8.2.7. PrivaMix Variation 6: Using web browsers for mixing

8.2.2. PrivaMix Variation 1: Shelters mix among themselves, without the Planning Office.

As stated above, Step 3 describes communication controlled by the Planning Office, but other models are just as valid. In the version above and in previous examples (Section 8.1), the Planning Office communicates with each Shelter in turn. Alternatively, in Step 3 the de-duplication could be done among the Shelters themselves with complete mixes and original UIDs sent to the Planning Office. Details of this variation appear below. In this writing, de-duplication assumes communication the Planning Office controls communication unless stated otherwise.

Variation of Step 3. De-duplicate UIDs.

1. Shelters randomly select a permutation of themselves. See protocol described in [29].
2. Shelters pass all ($Shelter_i$, UID_i, mix_i) triples to the first Shelter in the permutation for mixing. At this point, each mix_i is the same as UID_i.
3. When the Shelter has mixed each mix_i, it replaces the mix_i value in each triple with its further mixed result. Updated triples are then passed to the next Shelter in the permutation for mixing. Processing continues until the triples have complete mixes.
4. The final Shelter in the permutation forwards the final triples to the Planning Office.

The advantage of this variation to PrivaMix is that Shelters compute complete mixes without Planning Office involvement. Side effects are: (a) all Shelters learn the number of Client visit records at each Shelter; and, (b) the last Shelter(s) know the de-duplicated results.

8.2.3. PrivaMix Variation 2: Shelters check that UIDs are legitimate

Shelters can effect a check on the number of UIDs to mix by computing the total number of UIDs to mix among themselves before Step 3 of the PrivaMix Protocol begins. Then, during the PrivaMix Protocol, each Shelter can validate whether the number of values asked to mix by the Planning Office is correct. There are many ways to do this. An adaptation of the PrivaSum Protocol [27] allows the Shelters to jointly compute the total number of UIDs without Shelters knowing the number of UIDs from any particular Shelter.

8.2.4. PrivaMix Variation 3: Matching UIDs to Universal Data Elements

The description of the PrivaMix Protocol does not explain how complete mixes are matched to the original UID and UDE information provided in Step 2. There are many equally valid ways to accomplish this. A simple way is to maintain the order in which values are passed. If the Planning Office provides Shelter x with the stream v_1, v_2, v_3, ..., as values to mix, then the stream from the Shelter back to the Planning Office should be the mixes in the following order: $\mathbf{F}(v_1, s_x)$, $\mathbf{F}(v_2, s_x)$, $\mathbf{F}(v_3, s_x)$,... This approach has the advantage that Shelter and original UID information is not part of the communications.

8.2.5. PrivaMix Variation 4: Providing aggregate count distributions, not Client-level data

Rather than PrivaMix providing Client-level data to the Planning Office, PrivaMix can alternatively provide aggregate de-duplicated count distributions. As described so far, PrivaMix provides the Planning Office with a detailed visit record for each Client; this is termed Client-level data. The Universal Data Elements (Section 3.5) describe Client-level data. In this variation of PrivaMix, the Planning Office would instead get distributions of how many Clients matched particular characteristics. An example of an aggregate count distribution is a breakdown of the number of Clients in age ranges. Providing the Planning Office with aggregate count distributions gives the Planning Office exactly the information needed for reports, such as the AHAR (Section 3.6) without revealing Client-level details that can compromise privacy by enabling data linking. There are several ways to modify PrivaMix to provide aggregate count distributions. Here is one way.

In Step 2 of the PrivaMix Protocol, Shelters send Client-level data, specifically the Universal Data Elements (Section 3.5), to the Planning Office. Step 3 then involves de-duplication of UIDs only. Alternatively, Shelters can send the same Client-level information to the Planning Office, but the data is not explicitly made accessible to Planning Office personnel. The data may be held by the PrivaMix software operating on the Planning Office machine without allowing Planning Office personnel the ability to view or access the information. These data may be encrypted and/or held in temporary memory.

When UID de-duplication completes in Step 3, the PrivaMix software operating on the Planning Office machine provides aggregate de-duplicated count distributions following a configurable script that describes which combination of values to aggregate and count. For example, Figure 6

shows the kinds of questions the AHAR answers. PrivaMix could answer these questions directly to the Planning Office without sharing Client-level data.

While this variation improves privacy by being a significant guard against unwanted data linking (Section 4.2), it also limits the use of de-duplicated results. The only result is aggregate count information. Other count information would not be available.

Another concern with this variant is error-checking. Typing mistakes and inconsistent values appearing in different records relating to the same Client becomes more difficult to spot and address when personnel responsible for using the count distributions in reports cannot easily examine the data that formed the basis of the counts. Workarounds may be possible by including cross-counts and Shelter-based distributions.

8.2.6. PrivaMix Variation 5: Anonymizing client-level data

A way to help thwart data linkage threats within PrivaMix while still providing Client-level data is to anonymize the data after de-duplication. Formal protection models identify which values can be sensitive to linking and either generalize or suppress those values from the resulting dataset so that each record ambiguously relates to a minimum number of people [30][31]. For example, if a 80 year old woman is an outlier in the data because of her age, either her age would be removed from the data or generalized to a category having more people, such as "50 plus" as appropriate given the other ages appearing in the data. A downside of this approach is that Client-level data will contain generalized or suppressed values, which can make it harder to work with statistically or detect typographical errors.

The last two PrivaMix variations (Section 8.2.5 and Section 8.2.6) address ways within PrivaMix to improve privacy and help thwart data linkage threats. An alternative lies outside PrivaMix, in choosing non-identifiable Client-level data elements. Section 11 examines these trade-offs in detail.

8.2.7. PrivaMix Variation 6: Using web browsers for mixing

The generic description of PrivaMix described above establishes the need for each participant (Shelter, HMIS, and Planning Office) to have a computer on a shared network. This can be an expensive proposition if each participant needs a machine on a dedicated network, and an inexpensive proposition if each participant can alternatively use existing computers that already have Internet access. While it is reasonable to assume that virtually no participant has an extra machine available to devote to mixing alone, it is reasonable to assume that each participant already has a machine (computer or even mobile phone) for email communication and web browsing, as these have become fundamental means of sharing information. There is nothing inherent in the PrivaMix Protocol that prohibits its execution on these devices using commonly used web browsing[15] software originally shipped with these machines.

15 A "web browser" is a computer program that allows users to view and share information over the World Wide Web. Virtually all computers, and even some mobile phones, come with web browsers. Internet Explorer is the most common web browser on machines running the Windows operating system. Other popular web browsers are Firefox and Safari.

Implementing PrivaMix through web browsers enables a wide array of existing computers to participate in mixing without installing any special software on the machines and without any special concern to user training. The important condition is that the Shelter's private value remains private to the Shelter's web browser. This is done seamlessly as described below.

When a web browser visits a website devoted to running the PrivaMix Protocol, software for producing UIDS and mixes seamlessly downloads itself into the web browser's operational environment. PrivaMix software then remains active on the machine as long as the web browser views the PrivaMix website. Once the web browser visits another website or stops running altogether, the PrivaMix software is no longer available. All machines participating in mixing should therefore have web browsers viewing the PrivaMix website throughout the process.

Here is a walk-through the PrivaMix Protocol using web browsers. To begin, all participants visit a web address[16] of a web server running the PrivaMix software. The server may run on a machine at the Planning Office. Alternatively, a third party facilitator may provide a server, which can be used by one or more Planning Offices. No additional privacy concerns result from either a Planning Office or a third party hosting the server.

Each participant must provide a previously agreed upon password to authenticate its machine; otherwise access to the PrivaMix software is not allowed. This prohibits others from wrongfully participating in a mix.

All communications between participating machines and the server are encrypted using existing web browser software. Web browsers already include encryption software. An example of its use occurs when conducting credit card and financial transactions using a web browser. Encrypting communications thwarts eavesdropping attempts.

Software containing the PrivaMix function seamlessly downloads into the Shelter's web browser environment, allowing the Shelter machine to produce UIDs and mixes as needed. After downloading the PrivaMix function, the Shelter's machine randomly selects a private value. The Shelter's web browser holds this value privately. It is never transmitted to the server.

The user of the Shelter machine selects a file to upload. This file contains Client source information and Universal Data Elements, one row for each Client. Immediately prior to forwarding a Client's Universal Data Elements to the server, the Shelter's machine computes the Client's UID and then forwards the UID and the Universal Data Elements. The web browser performs these computations automatically.

When all Shelters complete the uploading of Client information, the server contains all Client visit information but the result does not reveal which Clients across visits are the same. The Shelters must mix UIDs to de-duplicate. The server orchestrates mixing, one Shelter at a time, as described earlier. A Shelter machine uses its privately held value and copy of the PrivaMix function to mix. The final de-duplicated results appear first at the server, which then forwards the de-duplicated result to the Planning Office.

16 A web address is often known as a URL (Uniform Resource Locator). The information stored at the web address is termed a web page. Web pages occurring from the same server idenitfy a website. Common web addresses end in .com or .edu. Examples are: www.google.com and privacy.cs.cmu.edu.

In summary, these variations, provide the following recommendations.

Recommendation #22: If desirable, use a variation of the PrivaMix Protocol to have a party other than the Planning Office orchestrate mixing. One variation (Section 8.2.2) describes how Shelters perform mixes among themselves and then forward de-duplicated results to the Planning Office. Another variation (Section 8.2.7) describes how a third-party might orchestrate de-duplication and then forward results to the Planning Office.

Recommendation #23: Thwarting data linkage threats requires further privacy consideration, realized as variations of PrivaMix and/or dictates on data elements. Rather than PrivaMix providing Client-level data to the Planning Office, PrivaMix can alternatively provide aggregate de-duplicated count distributions (Section 8.2.5). A way to help thwart data linkage threats within PrivaMix while still providing Client-level data is to anonymize the data after de-duplication (Section 8.2.6). An alternative that lies outside of PrivaMix is to chose non-identificable Client-level data elements (Section 11).

Recommendation #24: An economical implementation of the PrivaMix Protocol involves using traditional web browsers already provided with computers (Section 8.2.7). Doing so has the advantage that no dedicated machine is needed, that no additional software has to be installed, and that no intense user training is needed.

8.3 Requirements of a PrivaMix function

Given secret shelter and client information as integers, a desired PrivaMix function (**F**) must satisfy the following six requirements.

The first requirement for a PrivaMix function is as follows. Given secret source information for the client c_i and private information for any pair of shelters s_x and s_y, it should be highly unlikely that $\mathbf{F}(ci, sx) \neq \mathbf{F}(c_i, s_y)$.

The first requirement for a PrivaMix function states that the UIDs for the same client *i* appearing at different shelters *x* and *y* should not be the same. It should be highly likely that $u_{ix} \neq u_{iy}$ where $\mathbf{F}(c_i, s_x) = u_{ix}$ and $\mathbf{F}(c_i, s_y) = u_{iy}$. This is the "inconsistent assignment" requirement.

The second requirement for a PrivaMix function is that it has the property that $\mathbf{F}(c_i, s_j)=u_{ij}$ computes in polynomial time, and it is computationally infeasible to compute its inverse $\mathbf{F}^{-1}(u_{ij})=(c_i, s_j)$.

The second requirement states that **F** must be a one-way function[17]. Applying **F** to secret client and shelter information should compute in real-time. However, given a result of **F**, the secret client and shelter information should not compute in any reasonable time. This is the "one-way" requirement.

17 The cryptography community commonly uses the term "one-way function" to refer to a hash or encryption function for which it is reasonably fast to compute the hash or encryption value, but computationally infeasible to reverse the process.

Given a client c_i, let the term "complete mix" over n shelters refer to z_i such that:
$\mathbf{F}_n(\ldots\mathbf{F}_2(\mathbf{F}_1(c_i, s_1), s_2), \ldots, s_n) = z_i$

Let the term "sub-mix" refer to $\mathbf{F}_y(\ldots\mathbf{F}_x(c_i, s_x)\ldots, s_y) = z_{it}$ where t is the sequence of shelters involved in the mix, $|t| < n$, and no shelter in the sequence appears more than once. For a complete mix, t is a sequence containing all shelters.

For n shelters, there are $n!$ ways in which to arrange them ("permutations"). Given a permutation p of n shelters and client c_i, let z_i^{p} be the complete mix for c_i over the n shelters arranged by permutation p.

The third requirement for a PrivaMix function is that for all $n!$ permutations of n shelters, $z_i^{\mathsf{p}} = z_i^{\mathsf{q}}$ where p and q are any two of the $n!$ permutations of shelters.

The third requirement states that a PrivaMix function must be a commutative cipher[18]. For a given c_i, all complete mixes over the same n shelters yield the same value z_i regardless of the order in which the shelters mix. This is the "commutative" requirement.

The fourth requirement for a PrivaMix function is that client c_i cannot be learned even if u_{ij}, z_i and any sub-mixes z_i^t are shared.

According to the fourth requirement, a PrivaMix function must not reveal the secret client information even if sub-mixes are revealed. This is the "privacy" requirement.

The fifth requirement of a PrivaMix function is that given two UIDs, u_{ix} and u_{jy}, then $u_{ix}=u_{jy}$ if and only if i and j refer to the same client.

The fifth requirement states that the PrivaMix function must be collision free. It must be highly likely that if any two UIDs or complete mixes are the same, the originating clients are the same. This is the "collision free" requirement[19].

The sixth requirement for a PrivaMix function is that the complete mixes for all shelters visited by the same client must be the same. Given two complete mixes, z_i and z_j, then $z_i=z_j$ if and only if i and j refer to the same client.

The sixth requirement refers to the correctness of mixing results. For each shelter visited by a client, the complete mix for that client must be the same as those complete mixes of all other shelters visited by the same client. This is the "correctness" requirement[20].

18 The cryptography community uses the term "commutative cipher" across multiple players to refer to a hashing or encrypting computation that provides the same value regardless of the order in which the hash or encrypted is performed.

19 In order to satisfy the collision free requirement, the source information for a client must be unique and well chosen. This section, which describes the requirements for a PrivaMix function, assumes the source information, which is termed the client's secret value in this section, is unique and well chosen. More discussion about selecting appropriate client source information appears in Section 10.

20 In order to satisfy the correctness requirement, the source information for a client must be reliably provided at each shelter visited. This section, which describes the requirements for a PrivaMix function, assumes the source information, is reliably provided at each shelter visited. More discussion about the reliability of client source information appears in Section 10.

The recommendation below summarizes the six requirements of a PrivaMix function.

*Recommendation #25: A PrivaMix function (**F**) must satisfy the following six requirements:*

(1) Inconsistent assignment: different shelters should generate different initial mix values for the same clients.
*(2) One-way function: **F** must be a one-way function.*
*(3) Commutative: **F** must be a commutative cipher.*
(4) Privacy: the secret client information cannot be learned given the sharing of complete and sub-mixes.
*(5) Collision-free: mixes from **F** must be collision-free.*
(6) Correctness: all complete mixes for the same client must be the same. Complete mixes for different clients should not be the same.

8.4 PrivaMix claims and limits

This section examines the appropriateness, correctness, and protection of PrivaMix. Discussion includes limitations, which do exist, though executing preliminary or additional secondary protocols, or adopting recognized best practices, as described, may improve restrictions. Before examining claims and limits, a summary of assumptions and threats appears.

Assumptions.
PrivaMix assumes Shelters are cooperative, non malicious participants that behave as instructed. The Planning Office is also a cooperative participant, but may attempt to learn private information during or after processing.

Review of Threats
Vulnerabilities appear when the Planning Office and/or a Shelter learns Client source information as a result of the existence or processing of UIDs. Only a Shelter that generates a UID for a Client should know the Client's source information.

PrivaMix de-duplicates UIDs while provably maintaining the privacy of Client source information. PrivaMix does not necessarily thwart data linkage attacks on the other Universal Data Elements, though a variation appeared earlier in Section 8.2 that provides an effective guard. As described in Section 4, a data linkage attack re-identifies Clients by matching combinations of values found in the Universal Data Elements to values found in other datasets. If matching is not based on UIDs, data linkage may lie outside the scope of the PrivaMix Protocol unless a variant is used to specifically address data linkage. The most effective way to thwart data linkage is to make sure the data elements of the client-level data cannot be linked to other readily available data. Section 11 revisits the identifiability of the Universal Data Elements in light of PrivaMix's protection for UIDs in order to make recommendations to thwart data linkage.

Here are the seven statements claimed about PrivaMix.

8.4.1. Usability claim. Communication time is linear in the number of Shelters.

8.4.2. Correctness claim. If the complete mixes are the same, the Clients representing the original UIDs presented the same source information..

8.4.3. Privacy claim. A dictionary attack by the Planning Office will not yield reliable re-identifications.

8.4.4. Privacy claim. Compromising a Shelter will not help the intimate stalker learn where a targeted Client is (or has been). Similarly, compromising the Planning Office will not help the intimate stalker learn where a targeted Client is (or has been).

8.4.5. Privacy claim. Even if the Planning Office pads the UIDs with known values, the Planning Office does not learn Client source information.

8.4.6. Limitation. If the Planning Office and at least one Shelter collude, the Planning Office can learn Client source information about the Shelter's Clients and the Shelter can learn other Shelters its Clients visited.

8.4.7. Limitation. If during the de-duplication protocol, the intimate stalker compromises both the Planning Office and a Shelter the targeted Client visited, the intimate stalker can learn the locations of all Shelters the Client visited. In addition, the Planning Office can learn the source information for that Client.

8.4.1. Usability claim. Communication time is linear in the number of Shelters.

Proof sketch:
In Step 3 of the PrivaMix Protocol (Section 8.2.1), the Planning Office sequentially requests each Shelter to mix all UIDs once.

The de-duplication step (Step 3) of the PrivaMix Protocol dictates the time it takes to execute PrivaMix. Adding a Shelter to the PrivaMix Network increases the time it takes to execute the PrivaMix Protocol because the additional Shelter will have to mix all UIDs once and mixing is done sequentially in the de-duplication step. The other steps can be done by Shelters in parallel.

Let v be the total number of Client visit records across m Shelters and t be the average time it takes a Shelter to mix the UIDs from all other Shelters, then the total time to execute the PrivaMix Protocol is on the order of ($v * t * m$). Therefore, time is linear in the number of Shelters (and visits).

Section 10 reports on a real-world experiment in which 4 Shelter computers work autonomously, without user intervention, to de-duplicate UIDs. It took about 45 minutes to de-duplicate UIDs for 2194 visits, which is less than 12 minutes per Shelter. In the real-world setting, there are 25 or fewer Shelter computers per Planning Office, so communication is practical.

8.4.2. Correctness claim. If the complete mixes are the same, the Clients representing the original UIDs presented the same source information..

Proof sketch:
This exploits the commutative property of the PrivaMix function. See Section 8.1 and Section 8.2 for examples and discussion.

8.4.3. Privacy claim. A dictionary attack by the Planning Office will not yield reliable re-identifications.

Proof sketch:
The size and nature of Shelter private values can be selected to ensure that exhaustively trying all possible combinations of private values over all possible values of source information is computationally infeasible.

Section 5.3 describes a dictionary attack in which the Planning Office tries all possible combination of values to see which values match those for whom the Client information is known. One example in Section 5.3 shows that encrypting Social Security numbers can be vulnerable to a dictionary attack when the Planning Office knows the encryption function. Within four seconds, the Planning Office can compute the UID for every possible Social Security number and then match UIDs from Shelters with those computed by the Planning Office to learn the Client's Social Security number.

PrivaMix protects against a dictionary attack by requiring the Planning Office to try all possible combinations of Shelter private values and Client Source information to learn Client source information. If Shelters chose sufficiently large private values, exhaustively attempting every combination is not feasible. See Figure 18.

Recommendations below help thwart dictionary attack possibilities.

Recommendation #26: Each Shelter must select a sufficiently private value so that efforts by the Planning Office to exhaustively compute all combinations of Shelter private values and Client source information (a dictionary attack) are not feasible. Most likely a Shelter's computer will be required to select a private value 512 bits or larger (as appropriate and most likely randomly selected at the start of each reporting period).

Recommendation #27: To help thwart the possibility of the Planning Office or other Shelters learning a Shelter's private value, a Shelter may not even explicitly know its own private value for a reporting period –i.e., the computer program may generate it internally and not explicitly reveal it.

Recommendation #28: To help thwart the possibility of the Planning Office or other Shelters learning a Shelter's private value, a Shelter may make its private value available to its copy of the PrivaMix function only while mixing over the PrivaMix Network. Other parties should not be able to invoke a Shelter's PrivaMix function with the Shelter's private value.

8.4.4. Privacy claim. Compromising a Shelter will not help the intimate stalker learn where a targeted Client is (or has been). Similarly, compromising the Planning Office will not help the intimate stalker learn where a targeted Client is (or has been).

Even though the intimate stalker may know the Client's source information, he cannot relate UIDs across Shelters because he does not know the private values of Shelters not colluding with him. Therefore, his compromising a Shelter or the Planning Office to find a targeted individual will not be fruitful for the same reasons described above for the dictionary attack.

8.4.5. Privacy claim. Even if the Planning Office pads the UIDs with known values, the Planning Office does not learn Client source information.

Consider the case in which the Planning Office makes a set of UIDs using its own copy of the PrivaMix function and a private value it selects. During de-duplication, the Planning Office then merges its made-up UIDs with the UIDs from the other Shelters for mixing. Because the Planning Office knows the source information that it used to construct its UIDs, any matches with UIDs from Shelters reveals the source information of actual Clients at particular Shelters.

In order to combat this attack, Shelters run simple protocols to validate the number of UIDs and/or to mix UIDs without Planning Office involvement (see Variation 1 and Variation 2 in Section 8.2). There are many other possible ways to accomplish these protections.

Recommendation #29: In order to prevent the Planning Office from padding UIDs with known values, the original PrivaMix approach should be modified to validate the number of UIDs and/or to mix UIDs without Planning Office involvement (see Variation 1 and Variation 2 in Section 8.2).

8.4.6. Limitation. If the Planning Office and at least one Shelter collude, the Planning Office can learn Client source information about the Shelter's Clients and the Shelter can learn other Shelters its Clients visited.

One of the most likely ways this collusion happens is when one of the shelters in the PrivaMix Network is the regional HMIS, because in many geographical regions, the staff of the HMIS is the same staff as the Planning Office (or CoC) and because there is a desire to de-duplicate visits across the domestic violence homeless shelters and the HMIS (not the domestic violence homeless shelters alone). Unfortunately, if the HMIS and the Planning Office collude, presumably because they are the same staff, the Planning Office can learn Client identities and the HMIS can learn which Clients by name, visited which Shelters. This is not allowed under VAWA (Section 7).

What makes this a significant threat to Client re-identifications is that unlike the Planning Office colluding with another domestic violence homeless shelter, a HMIS will usually contain most (if not all) Clients who visit any domestic violence homeless shelter. When a Client of a domestic violence homeless shelter seeks services beyond the domestic violence homeless shelter (e..g., meals or medical care), her source information appears in the HMIS related to those services. The HMIS knows her source information and the general services she received, but the HMIS does not

know she is a domestic violence homeless client or the domestic violence homeless shelter in which she resides. After de-duplication, the Planning Office learns the Client received services from a HMIS and from a domestic violence homeless shelter, and can use the identifying information in the HMIS to explicitly identify the Client.

In order to combat collusion between the HMIS and the Planning Office, the final de-duplicated information made available to the Planning Office by PrivaMix must be rendered unlinkable to the HMIS data that produced it in part. Remedies include having PrivaMix provide only aggregate information or provably anonymizing released data elements. Section 12 discusses these remedies in detail.

Recommendation #30: Care must be taken to combat possible collusion between the HMIS and the Planning Office because in many geographical regions, the staff of the HMIS is the same staff as the Planning Office (or CoC) and because there is a desire to de-duplicate visits across the domestic violence homeless shelters and the HMIS (not the domestic violence homeless shelters alone). As a participant in PrivaMix, a HMIS poses a significant threat to Client re-identifications because a HMIS will usually contain most (if not all) Clients who visit any domestic violence homeless shelter. Remedies include having PrivaMix provide only aggregate information or provably anonymizing released data elements. Section 12 discusses these remedies in detail.

8.4.7. Limitation. If during the de-duplication protocol, the intimate stalker compromises both the Planning Office and a Shelter the targeted Client visited, the intimate stalker can learn the locations of all Shelters the Client visited. In addition, the Planning Office can learn the source information for that Client.

The following two recommendations establish best practices to help.

Recommendation #31: Client records Shelters provide to the Planning Office should only include Clients who are no longer residing at the Shelter. This is a helpful recommendation, but not wholly satisfactory because Clients may re-visit previously visited Shelters.

Recommendation #32: The Planning Office should destroy all copies of the original UIDs once the de-duplication is complete. Doing so, limits the opportunity for compromise.

The claims and limits mentioned above reflect the generic PrivaMix approach. A specific implementation of a system that uses the PrivaMix approach requires revisiting claims and limits specific to implementation details. Differences in implementations may include communication flow (e.g. Planning Office in the middle or Shelter-to-Shelter), information content (e.g., a stream of values, or a list of values with their originating Shelter), and selection of the privately held Shelter value (.e.g., random selection, or pre-selection). Section 9 describes a particular instantiation of a PrivaMix system used in a real-world experiment (Section 10).

Recommendation #33: A specific implementation of a system that uses the PrivaMix approach requires revisiting claims and limits specific to implementation details. Differences in implementations may include communication flow (e.g. Planning Office in the middle or Shelter-to-Shelter), information content (e.g., a stream of values, or a list of values with their originating Shelter), and selection of the privately held Shelter value (.e.g., random selection, or pre-selection).

8.5 Comparison to other UID technologies

This section compares the PrivaMix approach with the UID technologies examined in Section 6 using the criteria for assessing the utility ("warranty") and privacy ("compliance") of UID technologies introduced in Section 5. PrivaMix performs comparable to inconsistent hashing (Section 6.7) and distributed query (Section 6.8) making it generally better than encoding (Section 6.1), hashing (Section 6.2), encryption (Section 6.3), scan cards and RFIDs (Section 6,4), biometrics (Section 6.5), and consent (Section 6.6) at protecting privacy. Yet, its usefulness at de-duplicating is better than encoding, hashing, encryption, scan cards and RFID, but not better than biometrics or consent. The following sections examine these statements in detail.

There are two main variants of PrivaMix to consider: one in which the result is de-duplicated Client-level visit information (Section 8.5.1) and one in which the result is de-duplicated aggregate count distributions (Section 8.5.2). These pose different ways of addressing data linkage threats. As described in Section 8.2, providing the Planning Office with aggregate count distributions gives a privacy guard within PrivaMix to help thwart data linkage. Examining the identifiability of Client-level data the Planning Office receives (Section 11), provides a privacy guard outside of PrivaMix. This section ends with an overall comparison of PrivaMix and other UID technologies (Section 8.5.3).

8.5.1. Assessment with client-level results

When the result from executing PrivaMix is Client-level data at the Planning Office, as described in Figure 55, then a gross assessment of PrivaMix as a UID technology yields warranty and compliance statements comparable to those of inconsistent hashing.

See Figure 63 and Figure 64 for a gross assessment of using PrivaMix as a UID technology when the result is client-level data. Issues related to utility and the warranty statement appear in Figure 63. Issues related to privacy and the compliance statement appear in Figure 64. While shadings may identify some problems as being of severe or moderate concern, depending on implementation details, these problems may be sufficiently addressed with straightforward practices, policies, or technology decisions.

PRIVAMIX (with Client-level results) –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?* Serious de-duplication problems are likely if Clients provide non-verifiable source information inconsistently. On the other hand, source information that is not truthful, but consistently provided, is typically not a problem.
Verifiable source information		*Can problems occur if the UID is based on verifiable source information?* Using invariant Client information that can be consistently verified on each visit is likely to avoid problems. Even if the information is not correct, but consistently verified on each visit, no problems are likely. An example of invariant verifiable Client information is a reliably captured biometric.
Client confidence and trustworthiness		*How trustworthy is the UID likely to be perceived by Clients (as well as by those who regularly intake Clients)?* Like hashed UIDs, PrivaMix UIDs tend to appear cryptic, which can instill Client and intaker confidence and thereby avoid problems. Further, because UIDs are different across Shelters (and can even be different on multiple visits to the same Shelter), additional Client and intaker confidence can be attained. Problems may emerge based on the sensitivity of requested source information despite the cryptic appearance of the UID itself, bu in most PrivaMix implementations the UIDs are not ever visible.
Inflated accounting		*What are the circumstances under which de-duplication is likely to inflate the accounting?* Count inflation can occur in cases where a Client provides different source information on different visits. In these cases, different UIDs are generated and therefore will not match to each other even though they are assigned to the same Client. Count inflation can also occur in cases in which a Client provides incomplete or missing information on different visits, thereby producing different non-matchable UIDs across Shelters.
Deflated accounting		*What are the circumstances under which de-duplication is likely to deflate the accounting?* Count deflation is possible when different Clients provide identical complete and incomplete information. A glaring example occurs for Clients in which all relevant source information is missing. Attention should be paid to how these situations are addressed in UIDs across Shelters. Count inflation is more likely than deflation.

…continued on next page …

Handling bad or missing input		*What is the effect of bad, incomplete, or missing source information on performance?* Typing mistakes and incomplete or missing information can generate different UIDs for a Client than would have been generated with complete and properly entered information. This tends to inflate accounting by generating spurious UIDs for Clients having multiple visits. Incomplete and missing information may also inflate accounting. Inflation is more likely than deflation.

	Most severe/difficult problem
	Moderate problem
	A problem
	May be a problem
	No problem likely, or not applicable

Figure 63. Gross Warranty assessment of using PrivaMix (with Client-level results) as a UID technology.

PRIVAMIX (with Client-level results) –COMPLIANCE (PRIVACY) STATEMENT

Intimate Stalker		*What vulnerabilities exist for the intimate stalker?* Because each Shelter has a different UID for the same Client, access to Shelter information is limited to a Shelter-by-Shelter basis. Vulnerabilities that are able to be exploited by an intimate stalker are limited to the Planning Office, which controls the de-deduplicated result. Vulnerabilities at the Planning Office may be addressed by the selection of data elements that comprise the de-duplicated results, and by control and audit of de-duplicated results.
Re-identification: Linking		*What vulnerabilities exist for re-identification of UIDs (and Dataset) using data linkage on UIDs?* Because a different UID is generated at each Shelter a Client visits, and the UIDs are not used outside HMIS data, unauthorized linking on UIDs is not likely. Re-identification not using UIDs is possible. Remedies rely on anonymizing data values (Section 8.2.6) or data elements (Section 11).
Re-identification: Dictionary Attack		*What vulnerabilities exist for re-identification of UIDs (and Dataset) using a dictionary attack on UIDs?* Because there should be a large range of possible UID values, a different UID generated at each Shelter a Client visits, and the non-use of UIDs used outside de-deuplication, a dictionary attack is not likely to be fruitful because of the large number of possibilities. However, care must be taken to make sure that no additional UIDs are added to the mix by the Planning Office. See PrivaMix variation (Section 8.2.3) for a remedy.
Re-identification: Reversal		*What is involved in reverse engineering the UID construction method?* Because PrivaMix functions are strong, reversal is not usually an issue. But if a Shelters' PrivaMix function and private value are available to unlimited use by the Planning Office, re-identification can result. Care must be taken to control or limit the function's use to avoid unwanted dictionary attacks (discussed above) or reverse compilations. (A dictionary is more likely than an attempt to reverse compile the function.)
Exposure		*What legal or technical risks or liabilities may be introduced based on the existence of the resulting database or UID technology?* The existence of mixed UIDs used only in the HMIS-context is not likely to expose Clients to additional risks beyond those mentioned above.

	Most severe/difficult problem
	Moderate problem
	A problem
	May be a problem
	No problem likely, or not applicable

System Trust
Which parties are heavily trusted?

Planning Offices are heavily trusted to control access and use of results.

Figure 64. Gross Compliance assessment of using PrivaMix (with Client-level results) as a UID technology.

8.5.2. Assessment with aggregate results

Rather than PrivaMix providing Client-level data, as described in Figure 55, the result can be the AHAR report itself or some other representation of aggregate count distributions. When the result is aggregate count distributions, a gross assessment of PrivaMix as a UID technology yields the same utility (or warranty), as shown in Figure 65, but improved privacy (or compliance), as shown Figure 66, than results with Client-level data.

Issues related to utility and the warranty statement appear in Figure 65. Issues related to privacy and the compliance statement appear in Figure 66. While shadings may identify some problems as being of severe or moderate concern, depending on implementation details, these problems may be sufficiently addressed with straightforward practices, policies, or technology decisions.

PRIVAMIX (with aggregate results) –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?* Serious de-duplication problems are likely if Clients provide non-verifiable source information inconsistently. On the other hand, source information that is not truthful, but consistently provided, is typically not a problem.
Verifiable source information		*Can problems occur if the UID is based on verifiable source information?* Using invariant Client information that can be consistently verified on each visit is likely to avoid problems. Even if the information is not correct, but consistently verified on each visit, no problems are likely. An example of invariant verifiable Client information is a reliably captured biometric.
Client confidence and trustworthiness		*How trustworthy is the UID likely to be perceived by Clients (as well as by those who regularly intake Clients)?* Releasing only aggregate count results can instill Client and intaker confidence and avoid problems, especially since aggregate results are based on PrivaMix UIDs that appear cryptic, and are different across Shelters (and can even be different on multiple visits to the same Shelter). Problems may still emerge based on the sensitivity of requested source information. Educating Clients and those who perform intake regularly and/or issuing privacy notices may help.
Inflated accounting	■	*What are the circumstances under which de-duplication is likely to inflate the accounting?* Count inflation can occur in cases where a Client provides different source information on different visits. In these cases, different UIDs are generated and therefore will not match to each other even though they are assigned to the same Client. Count inflation can also occur in cases in which a Client provides incomplete or missing information on different visits, thereby producing different non-matchable UIDs across Shelters.
Deflated accounting		*What are the circumstances under which de-duplication is likely to deflate the accounting?* Count deflation is possible when different Clients provide identical complete and incomplete information. A glaring example occurs for Clients in which all relevant source information is missing. Attention should be paid to how these situations are addressed in UIDs across Shelters. Count inflation is more likely than deflation.

…continued on next page …

Handling bad or missing input		*What is the effect of bad, incomplete, or missing source information on performance?* Typing mistakes and incomplete or missing information can generate different UIDs for a Client than would have been generated with complete and properly entered information. This tends to inflate accounting by generating spurious UIDs for Clients having multiple visits. Incomplete and missing information may also inflate accounting. Inflation is more likely than deflation.

	Most severe/difficult problem
	Moderate problem
	A problem
	May be a problem
	No problem likely, or not applicable

Figure 65. Gross Warranty assessment of using PrivaMix (with aggregate results) as a UID technology.

PRIVAMIX (with aggregate results) –COMPLIANCE (PRIVACY) STATEMENT

Intimate Stalker		*What vulnerabilities exist for the intimate stalker?* Because each Shelter has a different UID for the same Client, access to Shelter information is limited to a Shelter-by-Shelter basis. Vulnerabilities related to exploiting the Planning Office are very limited since only aggregate count information is available. Care should be taken for the Planning Office not to even save UIDs and mixed UIDs.
Re-identification: Linking		*What vulnerabilities exist for re-identification of UIDs (and Dataset) using data linkage on UIDs?* If only aggregate count information results, linkage on UIDs and the Dataset is limited. Vulnerabilities at the Planning Office may be further minimized by the system not releasing mixed or non-mixed UIDs.
Re-identification: Dictionary Attack		*What vulnerabilities exist for re-identification of UIDs (and Dataset) using a dictionary attack on UIDs?* Because there should be a large range of possible UID values, a different UID generated at each Shelter a Client visits, and the non-use of UIDs used outside de-deuplication, a dictionary attack is not likely to be fruitful because of the large number of possibilities. However, care must be taken to make sure that no additional UIDs are added to the mix by the Planning Office. Section 8.2.3 poses a remedy.
Re-identification: Reversal		*What is involved in reverse engineering the UID construction method?* Because PrivaMix functions are strong, reversal is not usually an issue. But if a Shelters' PrivaMix function and private value are available to unlimited use by the Planning Office, re-identification can result. Care must be taken to control or limit the function's use to avoid unwanted dictionary attacks (discussed above) or reverse compilations. (A dictionary is more likely than an attempt to reverse compile the function.)
Exposure		*What legal or technical risks or liabilities may be introduced based on the existence of the resulting database or UID technology?* The existence of mixed UIDs used only in the HMIS-context is not likely to expose Clients to additional risks beyond those mentioned above.

	Most severe/difficult problem
	Moderate problem
	A problem
	May be a problem
	No problem likely, or not applicable

System Trust
Which parties are heavily trusted?

Planning Offices are trusted to control access and use of results, but with aggregate results only, sharing concerns are minimal.

Figure 66. Gross Compliance assessment of using PrivaMix (with aggregate results) as a UID technology.

8.5.3. Overall comparison

Figure 67 compares PrivaMix , with Client-level and aggregate count distributions, with the UID technologies examined in Section 6. PrivaMix performs comparable to inconsistent hashing (Section 6.7) and distributed query (Section 6.8) making it generally better than encoding (Section 6.1), hashing (Section 6.2), encryption (Section 6.3), scan cards and RFIDs (Section 6,4), biometrics (Section 6.5), and consent (Section 6.6) at protecting privacy. Yet, the utility of its de-duplicated results is better than encoding, hashing, encryption, scan cards and RFID, but not better than biometrics or consent.

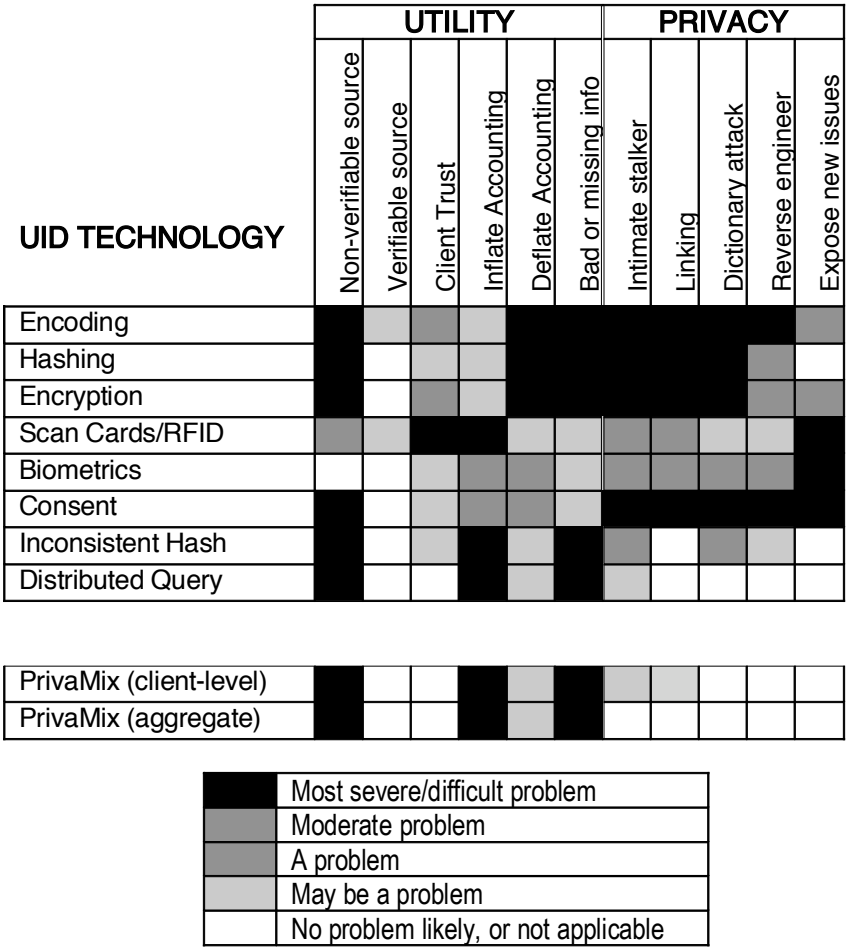


Figure 67. Summary of gross assessments of UID technologies,including PrivaMix variants.