Attribute-based Oblivious Message Retrieval

Jeriah Yu
Advisor: David Wu

Abstract

Recent developments in cryptography enable a system of Oblivious Message Retrieval using Homomorphic Encryption schemes. Oblivious Message Retrieval allows for untrusted servers to detect and retrieve encrypted messages efficiently on behalf of users without learning anything about message content or recipients, but retrieves messages solely based on user identities. We extend the functionality of Oblivious Message Retrieval with additional homomorphic encryption operations to support filtering retrieved messages with user-supplied conjunction policies while maintaining the privacy of user queries and filter parameters. Compared to the baseline implementation, our extension adds a computation and communication cost factor that scales approximately quasilinearly to the size of the conjunction.

Contents

1 Introduction

Assume that Alice wants to send a message to Bob over an email server. Without any cryptographic protection, the email server can read the message in plaintext. To prevent this, modern applications can protect message content between the sender and receiver with secure key agreement protocols and end-to-end encryption. However, the server can still learn metadata about Alice sending a message to Bob at a point in time. For full anonymity, the server should learn no information about the sender and receiver. One approach for receiver privacy is to encrypt the identity of Bob. However, the server now loses the ability to direct Bob's intended messages to his client. The onus is now on Bob to identify the messages in the server database that he is the receiver of without revealing his identity, and to avoid information leakage, the messages designated to Bob may be located anywhere in the encrypted data storage. While Bob can theoretically download an entire database or ledger and run a full scan through it, this is infeasible in practice with a large database or with small client-side computation, storage, or communication capacities.

Liu and Tromer [LT21] propose a solution of introducing a powerful detector server sitting between the database and client (Bob) that can operate on the entire dataset on behalf of the client and return the intended messages. This server must allow the client to retrieve all pertinent messages without learning any information about the client's identity or messages itself (the Oblivious Message Retrieval problem).

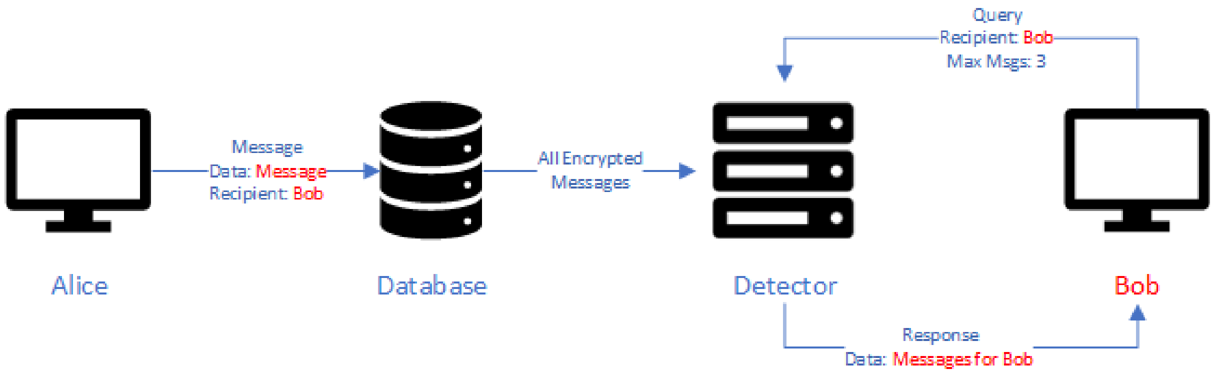

Figure 1: Simplified illustration of parties and exchanges involved in OMR setup and execution. Red text indicates information hidden by encryption. A detailed diagram of OMR is provided in [LT21]. Note that Bob's identity is hidden from the database and the detector and only known to Alice and Bob. This therefore requires a new solution for Bob to receive messages intended for him without relying on a third party to forward such messages.

The practical compact construction to achieve this goal introduced in their prior work combines two encryption schemes, one homomorphically, to provide privacy against the detector, but requires a prearranged upper bound in the number of pertinent messages and attempts to retrieve all messages pertinent to the user in one transaction. Increasing the upper bound is expensive in regards to both communication and computation. For a large dataset such as a blockchain ledger or big data storage, the total messages linked to a client's identity can vastly exceed this bound and potentially the client's resources, leading back to the original full-scan problem. One way to reduce the computational burden for the client is to further restrict the set of retrieved messages. In practice, clients will often perform fine-grained filtering on messages they wish to read (e.g. only retrieving unread or starred email). This limits the amount of messages a client may retrieve, which can improve concrete efficiency. We focus on this use case in this work.

We pose an extended problem of achieving feasible oblivious message retrieval with attribute-based filtering, where messages retrieved by the detector are filtered by user-provided queries at the detector server privately, without the detector learning any information about the filters. The filtered information should then be returned to the client in a smaller digest than the unfiltered messages. Having such filters makes progress towards allowing the OMR system to behave similar to SQL, used ubiquitously in data processing, while preserving privacy on both the query and the results. This enables practical private applications for consumers to query information like web searches without compromising their identity.

1.1 Our Contributions

We propose Attribute-based Oblivious Message Retrieval (ABOMR), an extension to Liu and Tromer's work that fulfill the additional requirements defined above. Using the formally defined notions of Oblivious Message Retrieval and its correctness and privacy notions, we show that the addition of user-supplied conjunction policies maintains similar correctness and security attributes already present in the system. We also show how the new user queries are protected from the detector. We first create an extension that takes the same parameters as the underlying OMR implementation to achieve security and correctness baselines, and then develop a scalable version that enables conjunction with multiple independent filters with minimal overhead. We implement these extensions using the Brakerski-Fan-Vercauteren (BFV) leveled homomorphic encryption scheme [FV12] and evaluate their performance relative to the OMR baseline. Our experimental results support an overall quasilinear relation between both computation and communication costs relative to the number of filter bits.

2 Preliminaries

2.1 (Ring) Learning With Errors

The Learning With Errors (LWE) and Ring Learning With Errors (RLWE) problems form the basis of the lattice-based cryptosystems used in this paper. RLWE can be reduced to the shortest vector problem (SVP) over ideal lattices [LPR12]. LWE can be reduced to worst-case GapSVP [Reg09], which is NP-hard.

Definition (Learning With Errors Hardness Assumption)**.** The problem is defined with respect to parameters n, m, q, x, where x is an error distribution over $\mathbb{Z}_q$. Let q be a prime. Randomly sample of $A \leftarrow \mathbb{Z}_q^{n\times m}$, $s \leftarrow \mathbb{Z}_q^n$, $e \leftarrow x^m$, $r \leftarrow \mathbb{Z}_q^m$. We assume that a polynomial-computation bounded adversary has negligible advantage distinguishing between $(A, s^T A + e^T)$ and (A, r).

Definition (Ring Learning With Errors Hardness Assumption)**.** The problem is defined with respect to parameters n, d, q, x, where x is an error distribution over R_q. Let q be a prime and $R_q = \mathbb{Z}_q[x]/(x^{2^d}+1)$ be a cyclotomic ring. Randomly sample a set a of n polynomials and small secret polynomial s uniformly from R_q, and a set of n small noise polynomials e from x. Now let u be a set of n randomly sampled independent polynomials of R_q. We assume that a polynomial-computation bounded adversary has negligible advantage distinguishing between the set $(a_i, sa_i + e_i)$ with i indexing n, and the set (a_i, u_i).

For performance, it is common to convert the polynomials into evaluation form, where a polynomial of degree n is uniquely represented by its evaluation on $n + 1$ points. This can be performed with the number theoretic transform (NTT), a variant of the Fourier transform on number fields, using quasi-linear time. Multiplying two cyclotomic polynomials together then becomes a procedure of utilizing negacyclic convolution with inner products, which is faster than multiplication in coefficient form[CY23], as well as allowing for more straightforward SIMD parallelism. RLWE is used for many implementations of practical homomorphic encryption due to the efficiency it provides over standard LWE. The hardness of LWE follows from the hardness of RLWE.

2.2 Homomorphic Encryption

Homomorphic encryption is a method of encryption where the ciphertext can be used as input into a function without requiring decryption. The system performing the computation does not learn any information about the input data. This enables outsourcing of computation to untrusted parties, such as from the client to the detector in the OMR system. When the output of the circuit is decrypted, the result is equivalent to the result from performing the function on the plaintext data.

Rivest et al.[RAD78] first introduced the concept of homomorphic encryption and identified existing schemes with partial homomorphic encryption properties. However, the buildup of error when combining multiple ciphertexts makes these schemes limited in the amount of operations they can achieve. Gentry [Gen09] provided the first construction with lattice-based cryptography and developed a method for Fully Homomorphic Encryption (FHE) with an arbitrary amount of operations using the boostrapping technique.

Such a FHE scheme has a Recrypt function that homomorphically decrypts the ciphertext using an encrypted secret key, yielding a fresh ciphertext that re-encrypts the plaintext with reduced error, enabling further computation without loss of correctness.

Bootstrapping requires considerable computation as it requires implementing the decryption algorithm for the scheme homomorphically into the Recrypt function. For practical Oblivious Message Retrieval and the Attribute-based extension, we will consider leveled homomorphic encryption, which will avoid recryption functionality but has a limit on the operations that can be performed based on the modulus size. BFV is used in this leveled form. A leveled homomorphic encryption scheme contains the following algorithms:

- $\text{GenParams}(1^\lambda, L)$: takes security parameter λ and maximum multiplicative depth L for any single ciphertext and outputs model parameters pp.
- $\text{KeyGen}(pp)$: takes model parameters and outputs secret key, public key, and evaluation key.
- $\text{Encrypt}(pk, m)$: takes public key and message, outputs ciphertext.
- $\text{Decrypt}(sk, ct)$: takes secret key and ciphertext, outputs message.
- $\text{Eval}(pk, C, \{ct_i\})$: takes public key, a circuit, and input ciphertexts, outputs resulting ciphertext.

The scheme should satisfy standard definitions of completeness, soundness, semantic security, and an additional requirement of compactness: the output of Eval is bounded by a function of the security parameter $poly(\lambda)$ and L, but independent of the size of the circuit. In other words, an input of multiple ciphertexts into an Eval circuit should result in an output shorter than the input to ensure computation is being performed. This mitigates a trivial construction of delaying computation until Decrypt, defeating the purpose of homomorphic encryption.

In OMR, the scheme cannot report decryption success or failure as that would reveal pertinency information to the detector. Decryption must result in a plausible plaintext, so the Wrong-Key Decryption property is required in order to satisfy soundness properties. Wrong-Key Decryption ensures that decrypting a scheme's ciphertext with the wrong key yields the true value with at most random chance.

Definition (Wrong-Key Decryption)**.** For a FHE scheme with plaintext space $\mathbb{Z}_t, t \geq 2$.

$$(sk, pk) \leftarrow \text{KeyGen}(), (sk', pk') \leftarrow \text{KeyGen}()$$
$$ct \leftarrow \text{Enc}(pk, 1), m' \leftarrow \text{Dec}(sk', \text{Recrypt}(pk', ct))$$

Then $Pr[m' = 1] \leq \frac{1}{p} + \text{negl}(\lambda)$ holds.

2.3 Peikert-Vaikuntanathan-Waters (PVW)

PVW, a variant of Regev encryption, is used as the outer non-homomorphic scheme in the OMR system. Integer PVW is defined as follows:

- $\text{PVW.GenParams}(1^\lambda, l, q, \sigma)$: takes security parameter, ciphertext modulus q, plaintext modulus size (bits) l, and standard deviation for the noise Gaussian distribution σ, outputs model parameters n and w, the secret key dimensions. All following functions implicitly take the model parameters as input.
- $\text{PVW.KeyGen}()$: Sample secret key $sk \leftarrow \mathbb{Z}_q^{n \times l}$ and matrix $A \leftarrow \mathbb{Z}_q^{n \times w}$, and error matrix $E \leftarrow Z_q^{l \times w}$ from a Gaussian distribution. $pk = (A, P = sk^T A + E)$. Output sk and pk.
- $\text{PVW.Encrypt}(pk, \vec{m})$: For a message $\vec{m} \in \mathbb{Z}_2^l$, let $t = \frac{q}{2} \cdot \vec{m}$. Sample error vector $e \leftarrow \mathbb{Z}_2^w$. Output ciphertext $(Ae, Pe + t)$.
- $\text{PVW.Decrypt}(sk, ct = (\vec{a}, \vec{b}))$: Let $\vec{d} = \vec{b} - sk^T \vec{a}$. Round each element of d to the nearest value: $\lfloor \frac{\vec{d} + q/4}{q/2} \rfloor$.

PVW is complete, sound, and satisfies semantic security [PVW07] under the Learning With Errors hardness assumption. Because the detector operates on the outer scheme ciphertexts and needs to homomorphically decrypt them, the scheme should have a simple decryption algorithm. PVW's Decrypt method is suited for this purpose as it is simple, only requiring an inner product of vectors and a range check, which are implemented into BFV as described later.

2.4 Brakerski-Fan-Vercauteren (BFV)

BFV is used as the inner homomorphic scheme in the OMR system. The detector homomorphically decrypts PVW ciphertexts into BFV-encrypted plaintexts. BFV uses modulus switching to reduce error at each multiplicative level. This results in ciphertexts having smaller coefficient moduli deeper in the multiplicative circuit. This is the source of the leveling limit in BFV. BFV is constructed with the following methods:

- BFV.GenParams(1^λ): take security parameter λ, outputs model parameters (t, q, n), describing the plaintext space, ciphertext space, and polynomial degree, and Gaussian distribution X for error. Also output integer p used for relinearization. All following functions implicitly take the model parameters as input.

- BFV.KeyGen(pp): Samples secret key polynomial s from the cyclotomic ring $R_t = \mathbb{Z}_t[X]/(X^n + 1)$. Sample $a \leftarrow R_q = \mathbb{Z}_q[X]/(X^n + 1), e \leftarrow X$. Output public key $(-(a \cdot s + e), a)$. Sample $a' \leftarrow R_{p \cdot q}$, $e' \leftarrow X$. Output relinearization key $rlk = (-(a' \cdot s + e') + p \cdot s^2, a)$.

- BFV.Encrypt($pk = (a, b), m$): to encrypt message m, sample $r \leftarrow R_2, e_1, e_2 \leftarrow X$. Output $ct = (a \cdot r + e_1 + \frac{q}{t} \cdot m, b \cdot r + e_2)$.

- BFV.Decrypt($sk, ct = (c, d)$): output $\lfloor \frac{t \cdot (c + d \cdot sk)}{q} \rfloor$.

- BFV.Add(ct_1, ct_2): perform element-wise add.

- BFV.Multiply(ct_1, ct_2, rlk): compute components

$$c_0 = \frac{t \cdot ct_1[0] \cdot ct_2[0]}{q}, c_1 = \frac{t \cdot ct_1[0] \cdot ct_2[1] + t \cdot ct_1[1] \cdot ct_2[0]}{q}, c_2 = \frac{t \cdot ct_1[1] \cdot ct_2[1]}{q}$$

  . Return the relinearized output:

$$\left(c_0 + \left\lfloor \frac{c_2 \cdot rlk[0]}{p} \right\rfloor, c_1 + \left\lfloor \frac{c_2 \cdot rlk[1]}{p} \right\rfloor\right)$$

The BFV scheme encrypts message polynomials of R_t into ciphertexts consisting of two polynomials in R_q, a larger cyclotomic ring. BFV is complete, sound, and satisfies semantic security [FV12] under the Ring Learning With Errors hardness assumption. BFV can be extended to support Fully Homomorphic Encryption by including a BFV.Recrypt function, but this is not used in Oblivious Message Retrieval.

3 Models

3.1 System Model

The system, threat, and problem models of Oblivious Message Retrieval come from Liu and Tromer's work. There exists a database or bulletin board b that contains messages. Each message has a sender and designated receiver, whose identity should remain private. A message consists of a payload-clue pair. The payload contains the data that the receiver should obtain, while the clue provides a method for the intended recipient to detect the message as addressed to them. We extend the clue functionality from prior work to also include encrypted metadata in the form of boolean attributes, which are used to enable the recipient to filter for specific attributes in the retrieval process.

A separate server known as the detector D acts on behalf of a recipient P to detect and retrieve the messages in the database that are pertinent to P and match P's attribute filter query. The goal is for this detection, filtering, and retrieval to be oblivious: a malicious detector should learn nothing about the pertinency of messages according to identity, the recipient's filter query, or the messages that match or fail to match the filter. P provides D their detection key, their filter, and an upper bound k on pertinent messages, and D scans and accumulates all messages in the database into a digest m and return it to P. The size of m should be much smaller than the size of b and proportional to k.

P processes m to retrieve the pertinent payloads, assuming a semi-honest detector and no overflow above k. The metrics of importance are the false positive rate (an impertinent message, either not addressed to P or not matching P's filter, is in the output), and the false negative rate (a pertinent message that is addressed to P and matches its given filters is not processed out of the digest). There can be multiple recipients, senders, and detectors on one shared database. The detector does not inherently store any long term information itself, and a recipient should be able to swap to separate detectors between query transactions without losing coherence.

3.2 Threat Model

We assume a polynomial computation-bounded adversary that can read all public information, including all public keys, the database information, and any communication across the four parties (sender, database, detector, recipient). The adversary can generate new messages and add them to the database, generate clue keys, and have other parties generate messages designating those keys as the intended recipient. The detectors, senders, and recipients should behave honestly but may collude by sharing information for completeness and soundness. For privacy, all parties except for the sender and intended recipient of a message may may collude. We do not consider adversaries mounting denial of service attacks in our threat model. We also do not consider leakage of message length in the system and threat models, as the OMR system can serve as a key encaspulation mechanism for a separate retrieval system and thus only works with identically sized messages.

3.3 Oblivious Message Retrieval

The formal definition of an Oblivious Message Retrieval scheme is defined to have the following methods:

- OMR.GenParams($1^\lambda, \epsilon_p, \epsilon_n$): takes security parameter λ, false positive rate ϵ_p, false negative rate ϵ_n, and outputs model parameters for the scheme. All following functions implicitly take the model parameters as input.
- OMR.KeyGen(): outputs a secret key sk, and a public key pk that contains a clue key pk_{clue} and detection key pk_{detect}.
- OMR.GenClue(pk_{clue}, p): takes clue key and payload, outputs a clue c that is paired with the payload.
- OMR.Retrieve(D, pk_{detect}, τ): takes a database of messages D (composed of payload-key pairs generated by OMR.KeyGen with multiple user public keys and payloads under the same cryptographic scheme), a detection key for the intended recipient pk_{detect}, and a pertinent messages upper bound τ, and outputs a digest of messages m.
- OMR.Decode(m, sk): takes the digest m and corresponding secret key sk of the designated recipient, and outputs the decoded payloads or reports overflow of k occurred.

The scheme should satisfy completeness, soundness, and privacy. The compact OMR scheme also satisfies the compactness property.

Property (Completeness and Soundness)**.** Let a database D, set of pertinent messages s, and a key pair $(sk, (pk_{clue}, pk_{detect}))$ be generated. Let
$m \leftarrow$ OMR.Retrieve(D, pk_{detect}, k'), $pl \leftarrow$ Decode(m, sk). For an arbitrary upper bound on pertinent messages k', if $|s| > k'$, pl should return overflow, otherwise for completeness $Pr[x_j \in pl | j \in s] \geq 1 - \epsilon_n - \text{negl}(\lambda)$, and for soundness $Pr[x_j \in pl | j \notin s] \leq \epsilon_p + \text{negl}(\lambda)$.

Property (Privacy)**.** Let A be a polynomial computation-bounded adversary. Run OMR.KeyGen twice to generate $(sk, (pk_{clue}, pk_{detect}))$ and $(sk', (pk'_{clue}, pk'_{detect}))$. A may see all public key elements and chooses a payload x. $c \leftarrow$ OMR.GenClue(pk_{clue}, x), $c' \leftarrow$ OMR.GenClue(pk'_{clue}, x). For computational privacy, A should have negligible advantage distinguishing between c and c': $|Pr[A(c) = 1] - Pr[A(c') = 1]| \leq \text{negl}(\lambda)$.

Property (Compactness)**.** For any well-formed database D and keypair $(sk, (pk_{clue}, pk_{detect}))$ generated by the scheme, let $m \leftarrow$ OMR.Retrieve(D, pk_{detect}, k). $|m|$ should scale approximately polylogarithmically with the total number of messages $|D|$. This is to require that the digest communicated from the detector to the recipient scales linearly in the number of pertinent messages, not by the entire database size.

4 Constructions

4.1 Compact Oblivious Message Retrieval

Our work extends on OMRp2, a practical, compact Oblivious Message Retrieval protocol developed by Liu and Tromer. This was achieved using multiple computation and communication optimizations such as pertinency vector packing, BFV parallelization via SIMD, sparse random linear codes, and communicating seeds for random generation. While these improve the resulting performance, we defer the details of specific optimizations to Liu and Tromer's prior work and focus on the overall procedures in OMRp2 that are pertinent to our work, which are described in Algorithm 1.

At a high level, OMRp2 encrypts the PVW secret key under BFV, clues are generated by encrypting a string of zeros under PVW public key of the intended recipient, and the detector homomorphically performs the PVW decryption operation on all PVW clues in the database using the recipient's PVW sk ciphertext. It then homomorphically checks if the result is a string of zeros for each clue, and deems such a message pertinent. Pertinent clues are homomorphically multiplied with their payloads, and a compressed digest is computed of these pertinent payloads and sent back to the recipient, which decodes using the BFV secret key. Liu and Tromer's work shows that correctness, soundness, privacy, and compactness of the OMR problem are satisfied. Due to its simplicity, the homomorphic decryption of PVW ciphertexts in OMRp2.Retrieve only requires computing an inner product and range checking.

- Inner Product - InnerProd($pp_{BFV}, pk_{BFV}, ct_{PVWsk}, x$): takes the BFV model parameters and public key, PVW secret key under BFV encryption, and the clue PVW ciphertext as input. Performs an inner product of the clue's PVW ciphertext with the secret key under BFV evaluation with the public key.
- Range Checking - RangeCheck($pp_{BFV}, pk_{BFV}, ct, r$): takes as input the BFV model and public key, the ciphertext of an encrypted plaintext element $u \in \mathbb{Z}_t$, and a range specifier r. Outputs 0 if $u \in [-r, r]$ is evaluated to true homomorphically, 1 otherwise.

The algorithm details of InnerProd and RangeCheck are described further in [LT21] section 7.2. Of particular importance, range checking differs from the rounding range defined in the PVW scheme in 3.3. The standard definition is to use $r = t/4$ which would lead to a false positive rate $\epsilon_p = 1/2$. By reducing $r < t/4$, the false positive rate can be reduced to $p = (2r + 1)/q$ while maintaining the correctness and soundness of PVW as long as the error does not exceed range. Additionally, the l identical ciphertexts of zeros for pertinent messages are used for soundness amplification to further reduce false positives: the final false positive rate $\epsilon_p = p^l$, since a false positive requires all l independent range checks to return 0. We now define the relevant subset of OMRp2:

- OMRp2.GenParams($1^\lambda, \epsilon_p, \epsilon_n$):
  Generate pp_{BFV}, pp_{PVW} from BFV.GenParams and PVW.GenParams.
  Select range r such that $\epsilon_p \leq \frac{2r+1}{pp_{BFV}.t}$.
  Output $pp = (1^\lambda, \epsilon_n, \epsilon_p, pp_{BFV}, pp_{PVW}, r)$.
  All following functions will implicitly take the pp model parameters as input.
- OMRp2.KeyGen():
  Get $(sk_{PVW}, pk_{PVW}) \leftarrow$ PVW.KeyGen(), $(sk_{BFV}, pk_{BFV}) \leftarrow$ BFV.KeyGen().
  Encrypt the PVW secret key under BFV: $ct_{PVWsk} \leftarrow$ BFV.Enc(pk_{BFV}, sk_{PVW}).
  Output $(sk = (sk_{BFV}), pk = (pk_{clue} = pk_{PVW}, pk_{detect} = (pk_{BFV}, ct_{PVWsk})))$.
- OMRp2.GenClue(pk_{clue}, x) (where x is the payload):
  Let $\vec{m} \leftarrow (0, ..., 0) \in \mathbb{Z}_t^l$.
  Compute $c \leftarrow$ PVW.Enc($pk_{clue}, \vec{m}$) and output (x, c).
- OMRp2.Retrieve($B, pk_{detect} = (pk_{BFV}, ct_{PVWsk}), k$):
  Parse the database B of payload-key pairs generated with OMRp2.GenClue to get $\{(x_1, c_1), ..., (x_N, c_N)\}$.
  For each index i in this database, run:
  - $a_1 \leftarrow$ InnerProd($pp_{BFV}, pk_{BFV}, ct_{PVWsk}, x_i$)

  - $a_2 \leftarrow \text{RangeCheck}(pp_{BFV}, pk_{BFV}, a_1, r)$
  - $a_3 = \prod_{i=0}^{l-1}(1 - a_2[i])$

  Then calculate the pertinency vector $PV \leftarrow \text{PVUnpack}(pp_{BFV}, pk_{BFV}, a_3)$. Combine PV with payload and convert into a compact digest M, and output M.

- OMRp2.Decode(M, sk): (Details omitted)
  Take the compact digest M and decrypt using the BFV secret key.
  Decompacts the digest to solve for message indices and payloads for the client, or returns overflow.

4.2 Single Attribute Filter

We first develop an initial extension to support one client flag. This construction supports a single boolean attribute using one additional ciphertexts. ℓ is increased to five, and we modify OMRp2.GenClue to also take in a boolean flag value $f \in \{0, 1\}$. The first four of $\vec{b}$ remains 0s as before and continues to be used for identifying message pertinency through the inner product and range checking procedures. The final value is set to $1 - f$. Prior to calling OMRp2.Retrieve, the recipient decides whether they want to filter on boolean flag g, and creates a vector $i \in \mathbb{Z}_2$ representing the filter: $i[0...\ell - 2] = 1$, $i[\ell - 1] = g$. The first four entires are still used to ensure identity matching for pertinency. i is then encrypted under BFV public key to get j.

OMRp2.Retrieve is modified to take in j as input, and $a_3 = \prod_{i=0}^{\ell-1}(1 - a_2[i])$ is changed to $a_3 = \prod_{i=0}^{\ell-1}(1 - (a_2[i] \cdot j[i]))$ to take into account the . $a_2[i]$ is inverted as zero-strings are pertinent, thus requiring an inversion to have pertinency represented as an encrypted 1. This is necessary to allow retrieval to function as an inner product. Therefore, this circuit matches common binary filtering, where if $j[i] = 1$, the attribute must be true in the clue to remain pertinent, and if $j[i] = 0$, the attribute is ignored. The rest of the algorithm continues as before. The modifications are formalized below as ABOMRp1:

- ABOMRp1.GenParams($1^\lambda, \epsilon_p, \epsilon_n$): Unmodified from OMRp2.GenParams.
  All following functions will implicitly take the pp model parameters as input.

- ABOMRp1.KeyGen(): Unmodified from OMRp2.KeyGen.

- ABOMRp1.GenClue(pk_{clue}, x, f): (f is the filtering attribute)
  Define $\vec{m} \leftarrow 0^\ell || (1 - f) \in \mathbb{Z}_t^{pp.\ell+1}$.
  Calculate $c \leftarrow \text{PVW.Enc}(pk_{clue}, \vec{m})$, and output (x, c).

- ABOMRp1.GenFilter($pk_{detect} = (pk_{BFV}, ct_{PVWsk}), f$):
  Define $\vec{i} \leftarrow 1^\ell || f \in \mathbb{Z}_t^{pp.\ell+1}$.
  Output $j \leftarrow \text{BFV.Enc}(pk_{BFV}, \vec{i})$.

- ABOMRp1.Retrieve($B, pk_{detect} = (pk_{BFV}, ct_{PVWsk}), k, j$):
  Parse the database B of payload-key pairs generated with OMRp2.GenClue to get $\{(x_1, c_1), ..., (x_N, c_N)\}$.
  For each index i in this database, run:

  - $a_1 \leftarrow \text{InnerProd}(pp_{BFV}, pk_{BFV}, ct_{PVWsk}, x_i)$.
  - $a_2 \leftarrow \text{RangeCheck}(pp_{BFV}, pk_{BFV}, a_1, r)$.
  - $a_3 = \prod_{i=0}^{l-1}(1 - (a_2[i] \cdot j[i]))$ (modified).

  Then calculate the pertinency vector $PV \leftarrow \text{PVUnpack}(pp_{BFV}, pk_{BFV}, a_3)$. Combine PV with payload and convert into a compact digest M, and output M.

- ABOMRp1.Decode(M, sk): Unmodified from OMRp2.Decode.

The first four ciphertexts simulate the identical output of a_3 as OMRp2 with negligible difference, thus correctness, soundness, privacy, and compactness for this follows from prior work. If the user chose to not filter on the attribute, the distribution for the fifth ciphertext also matches that of OMRp2, so it inherits the same properties. If the user does filter on the attribute and the clue does not contain it, the PV section allocated to the flag will homomorphically evaluate to 0 with error from the underlying encryption scheme. This is a tighter guarantee of filter soundness than the random result from wrong key decryption. By correctness, soundness, and security of BFV, the overall larger scheme maintains the same required properties of the OMR problem. Due to BFV encryption, the detector remains oblivious to whether $f = 0$ or $f = 1$ on either the clue or user filter sides.

Let ϵ_p' be the false positive rate of OMRp2 with parameter ℓ. In the case of no filtering, the false positive rate remains ϵ_p'. In the case of filtering, the false positive rate of this construction becomes $\epsilon_p = {\epsilon_p'}^{\frac{\ell+1}{\ell}}$, as an impertinent message now has a $p^{\ell+1}$ chance of being included in the digest compared to p^{ℓ} from earlier analysis. Let ϵ_n' be the false negative rate of OMRp2 with parameter ℓ. $\epsilon_n' = 1 - (1-n)^{\ell}$, where n is the false negative rate of an individual ciphertext that arises from the lattice protocols. Then the worst case false negative rate of this construction $\epsilon_n = 1 - (1 - \epsilon_c)^{\ell+1}$.

4.3 Conjuncted Attribute Filter

We proceed to modify the preceding scheme and support multiple attributes. The first l values in the GenClue and GenFilter vectors remain 0 and 1 respectively in order to preserve the correctness and soundness of OMR and maintain low error rates. Each slot after can now independently contain their own attribute flags, so GenClue and GenFilter now take attribute vectors. By keeping the same circuit in the Retrieval step, these individual attribute filters are conjuncted by intersection. This modification forms the ABOMRp2 construction:

- ABOMRp2.GenParams($1^{\lambda}, \epsilon_p, \epsilon_n$): Unmodified from OMRp2.GenParams.
  All following functions will implicitly take the pp model parameters as input.

- ABOMRp2.KeyGen(): Unmodified from OMRp2.KeyGen.

- ABOMRp2.GenClue($pk_{clue}, x, \vec{f}$): (modified, f is the vector of attributes)
  Define $\vec{m} \leftarrow 0^{\ell} || \vec{1} - \vec{f} \in \mathbb{Z}_t^{pp.\ell + |f|}$.
  Calculate $c \leftarrow$ PVW.Enc($pk_{clue}, \vec{m}$), and output (x, c).

- ABOMRp2.GenFilter($pk_{detect} = (pk_{BFV}, ct_{PVWsk}), \vec{f}$): (modified)
  Define $\vec{i} \leftarrow 1^{\ell} || \vec{f} \in \mathbb{Z}_t^{pp.\ell + |f|}$.
  Output $j \leftarrow$ BFV.Enc($pk_{BFV}, \vec{i}$).

- ABOMRp2.Retrieve($B, pk_{detect} = (pk_{BFV}, ct_{PVWsk}), k, j$):
  Parse the database B of payload-key pairs generated with OMRp2.GenClue to get $\{(x_1, c_1), ..., (x_N, c_N)\}$.
  For each index i in this database, run:

  - $a_1 \leftarrow$ InnerProd($pp_{BFV}, pk_{BFV}, ct_{PVWsk}, x_i$).
  - $a_2 \leftarrow$ RangeCheck($pp_{BFV}, pk_{BFV}, a_1, r$).
  - $a_3 = \prod_{i=0}^{l-1}(1 - (a_2[i] \cdot j[i]))$.

  Then calculate the pertinency vector $PV \leftarrow$ PVUnpack(pp_{BFV}, pk_{BFV}, a_3). Combine PV with payload and convert into a compact digest M, and output M.

- ABOMRp2.Decode(M, sk): Unmodified from OMRp2.Decode.

If a message is impertinent, the resulting PV value for that message after unpacking will remain 0 with high probability, which satisfies OMR soundness with the rest of the algorithm. If a message is pertinent to the user identity, the first ℓ values will be 0 while the remaining values will homomorphically decode to the clue flag value. With high probability, the final conjunction PV will be based on the user's filters, and a passing message will have a PV of 1. Then completeness follows from the OMR protocol. Privacy of the filters and user identity come from BFV security and the existing OMR privacy respectively, thereby satisfying the OMR problem requirements.

Let p be the false positive rate for each ciphertext. With l ciphertexts, the prior false positive rate was p^{ℓ}. If the number of flags $|f| = \ell$, the false positive rate for an impertinent message is $p^{2\ell}$ as every ciphertext of the 2ℓ has a p probability of range check resulting in pertinency. For $\ell + |f|$ ciphertexts, $\epsilon_p = p^{\ell+|f|}$. Let n be the false negative rate for each ciphertext. If a message is truly pertinent, every ciphertext including the attribute flags should decrypt successfully to their correct value with negligible error, so error is solely from the underlying encryption scheme's false negative rate. The overall false negative rate ϵ_n is the union bound of all ciphertext false negatives n. For BFV leveled homomorphic multiplication, doubling the number of ciphertexts (ℓ) causes the multiplicative depth to increase by 1, necessitating an increase in coefficient modulus and thus computation and communication. Therefore, the coefficient modulus scales logarithmically to the amount of attributes $|f|$, while the amount of ciphertexts involved in computation and communication scale by a factor of $|f| \times \ell$. This results in a quasilinear cost, with constant scaled by ℓ.

5 Performance

The protocols were implemented in C++ with the PALISADE library [22] for PVW and Microsoft SEAL [23] for BFV. The parameters used for testing are derived from prior work, with modifications due to computation constraints limiting the use of hardware SIMD acceleration: the size of the database is $|b| = 4096$, the bound on pertinent messages is $k = 5$, and the payload size is 612 bytes. For PVW encryption, we set $n = 450, q = 65537, \sigma = 1.3, w = 16000, \ell = 4$. Effective $\ell = 8$ as we set $|f| = \ell$.

The doubling of effective ℓ necessitates an additional layer of homomorphic multiplication to compute the pertinency, and the combination of the user filter with the RangeCheck output also requires an additional layer. To handle an increase in multiplicative depth of two, an increase in the BFV coefficient modulus by 60 bits was necessary for the provided parameters. This results in the original OMRp2 and ABOMRp2 implementations having a SEAL-reported remaining noise budget of 5 bits. Thus, the two layers of multiplication consume 60 bits of noise, which is in line with the difference in modulus size between Oblivious Message Detection and Oblivious Message Retrieval (which differ by two multiplicative layers) discussed in [LT21] section 10. With these changes, the parameters for BFV encryption are set so that $D = 4096, \log(Q) = 790 + 60 = 850, r = 850$. Other parameters remain unchanged from the OMR protocol.

		OMRp2	ABOMRp1	ABOMRp2
	DB Size ($\|b\|$)	4096	4096	4096
	Payload Size	612 bytes	612 bytes	612 bytes
	Pertinency Bound ($\|k\|$)	5	5	5
PVW Params	n	450	450	450
	q	65537	65537	65537
	σ	1.3	1.3	1.3
	w	16000	16000	16000
	l for pertinency	4	4	4
	l per flag	N/A	1	1
	No. of Attributes ($\|f\|$)	N/A	1	4
	Total ℓ	4	5	8
BFV Params	$\log(q)$	790	850	850
	t	65537	65537	65537
	n	4096	4096	4096

Table 1: Summary of Parameters

These parameters provide baseline OMRp2 error rates of $\epsilon_p = 2^{-21}$ and $\epsilon_n = 2^{-30}$. Each of the l ciphertexts therefore has an individual false positive rate of $\frac{2\times 850+1}{65537} \approx 2^{-5}$ and false negative rate 2^{-32}. For an effective $l = 8$, the worst case false positive rate for both attribute-based implementations is still bounded at 2^{-21}, and the best case $\epsilon_p = (2^{-21})^2 = 2^{-42}$. The worst case false negative rate $\epsilon_n = 1-(1-2^{-32})^{\ell}, \approx 2^{-29}$ for 8 attributes and $2^{-29.68}$ for 5 attributes.

Tests were run on a laptop Intel i7-11370H processor running at high performance without hardware acceleration. The new attribute features appear to retain relatively practical runtime costs in real world testing compared to the baseline implementation, as detailed in Table 2.

	OMRp2	ABOMRp1	ABOMRp2
Processing Time (sec)	146	189	270
Time per Message	0.036	0.046	0.066
Worst-case ϵ_p	2^{-21}	2^{-21}	2^{-21}
Worst-case ϵ_n	2^{-30}	$2^{-29.68}$	2^{-29}

Table 2: Summary of Performance from Static Parameters of Table 1

Runtime scales linearly within one level as a function of the number of ciphertexts. However, at each new level requiring an additional layer of multiplication, the BFV moduli have to increase, leading to a performance hit when the number of attributes increases by a factor. This is exhibited in Figure 1 when the number of attributes increases from 12 to 13, leading to ℓ increasing from 16 to 17 and requiring a further 60 bit increase in modulus size to 910 bits.

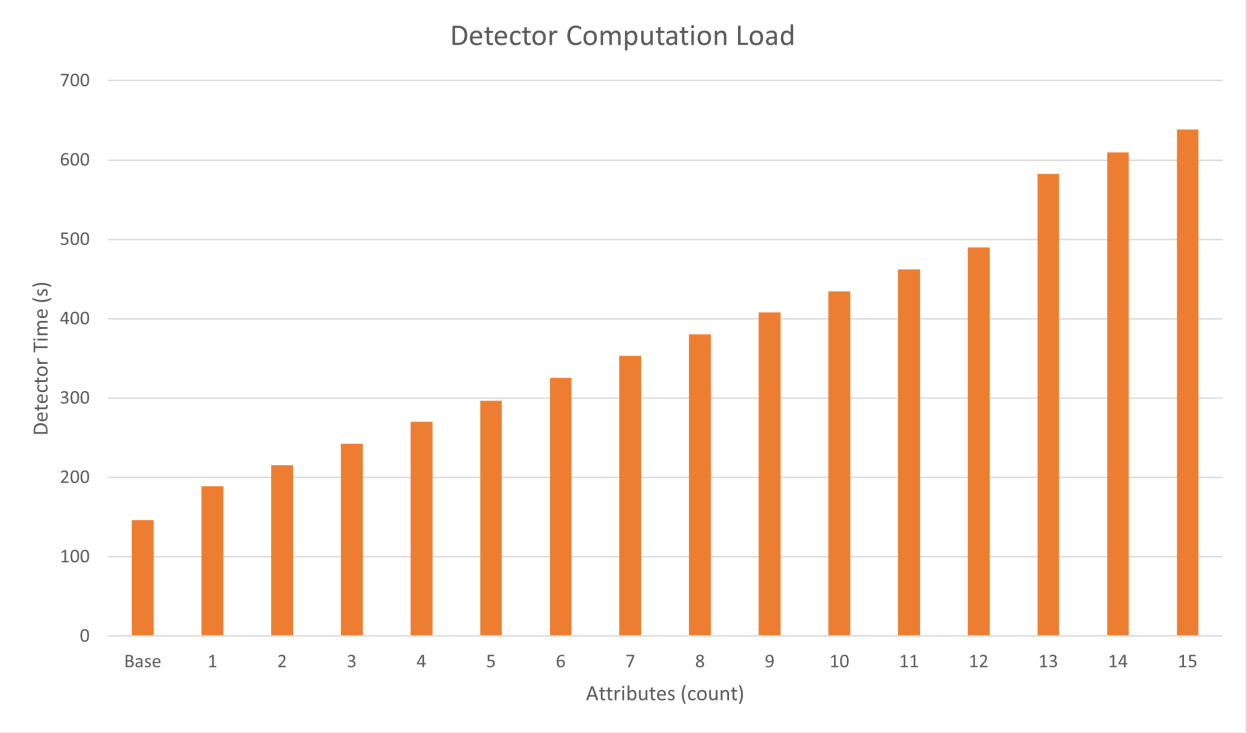

Figure 2: Detector runtime as a function of number of attributes. Base is OMRp2 without attributes. Adding additional attributes scales linearly within small increments as additional ciphertexts are added, with jumps in runtime at logarithmic intervals due to increased cryptosystem parameter values.

6 Related and Future Work

Related Work

Group Oblivious Message Retrieval Another extension of Oblivious Message Retrieval is to support multiple recipients per message. Liu et al. [LTW23] propose an extension to the original OMR protocol that support Ad-hoc and Fixed Group Oblivious Message Retrieval, as well as additional optimizations to algorithms that forms the baseline for this paper. Group OMR is especially useful for multiparty communication, such as encrypted group chats. The construction used for Ad-hoc Group OMR differs from standard OMR by using polynomial interpolation, however the addition of attribute filters should still be readily applicable as the flags stored in the clues can be encoded in the polynomial naturally and the retrieval step remains unaffected following the range checking procedure. Fixed Group OMR is constructed under the premise that the recipients do not change after being formed, and therefore uses a more efficient multi-recipient encryption (MRE) method with lower detection costs. This MRE scheme replaces the outer Peikert-Vaikuntanathan-Waters (PVW) scheme used to encrypt the clues, but also does not affect the inner Brakerski-Fan-Vercauteren (BFV) scheme where the pertinency vector can be homomorphically conjugated with user filters. Thus, attribute-based filtering can be readily applied to Fixed Group OMR on a per-user basis. The authors discuss a similar direction to send distinct payloads to each user in a single message to the group by interpolating a linear transformation function linking homomorphically encrypted user IDs to payloads.

Private Information Retrieval Private Information Retrieval (PIR) is a related problem where a client requests data at an index i in a database from a server without the server learning anything about i. For the purposes of Oblivious Message Retrieval, multi-query PIR is necessary as there may be multiple pertinent messages, but this still requires the client to know the indices of the data to retrieve. Groth et al. [GKL10] demonstrate a method for efficient multi-query computational PIR, known as batch PIR, with communication scaling by the number of pertinent messages rather than database size, which can therefore be used between the step from Oblivious Message Detection (finding the indices of pertinent messages) to Oblivious Message Retrieval. However, the construction for OMR already provides for retrieving the pertinent messages in an efficient manner via inner products and compaction, which is more efficient in communication to the client and homomorphic operations than separately communicating pertinent indices and performing homomorphic evaluation for PIR.

Attribute-based Encryption Our work on generalizing OMR to support fine-grained message retrieval is conceptually similar to Attribute-Based Encryption (ABE), where a message can contain attributes such that only a user who's identity has authority through those attributes can decrypt the message. Goyal et al. [Goy+06] introduce Key-Policy ABE, where the user key contains the access control dictating what it can decrypt while the ciphertexts are labeled with attribute flags, building on work by Sahai and Waters [SW04] introducing ABE through the use of Fuzzy Identity-Based Encryption. These works both generalize public-key encryption to support fine-grained access control.

Sender Privacy While OMR is concerned with recipient privacy, the complement problem of sender privacy is addressed in Riposte by Corrigan-Gibbs et al. [CBM15] Riposte is a system that protects broadcast messaging and allows multiple senders to anonymously add messages to a bulletin board or database, as long as it is distributed across at least three non-colluding servers. Combining Oblivious Message Retrieval and Sender Privacy allows for the privacy of both senders and receivers from other parties such as the database, detector servers, or observers.

Future Work

Conjunction Circuits The conjunction filter currently only supports logical AND of all filters together via homomorphic multiplication. In order to enable more flexible computations, support for homomorphic addition acting as logical XOR allows for construction of negations, leading to NAND operations which can be used to build universal circuits. However, while the inputs to the circuit, the user-defined filters, are private, the circuit itself in the current implementation is not. It remains to be studied whether the circuit can also be hidden from the detector or if the adjustments to security guarantees are necessary.

Threshold Filtering A further extension to supporting flexible conjunction circuits is the ability to filter based on integral values, potentially in combination with the boolean attributes presented here. If arbitrary circuits are supported, it would be possible to create binary comparison circuits with clues and user filters containing binary representations of values. It's also to be explored whether a different homomorphic encryption scheme than BFV or different parameters for plaintext and ciphertext spaces may support threshold filtering more naturally.

Compact Filters The current constructions require communication of filter flags to scale with the number of attributes available. Using similar strategies for compacting the digest returned from the detector to the recipient to scale with pertinent messages, it may be possible to compact the communcation of filters from the recipient to the detector to scale with just set or unset flags.

7 Conclusion

In this paper, we present a method for extending the functionality of Oblivious Message Retrieval to support more flexible user requests. We develop a prototype construction using homomorphic encryption operations to support filtering pertinent messages according to recipient-defined metrics while hiding those metrics from other parties including the detector serving acting on the recipient's behalf. Our experimental results showed that this construction maintains an acceptable level of correctness and practicality against the baseline OMR implementation while adding new features. While the presented implementation does not enable arbitrary computation on attribute inputs, this implementation makes progress towards further research into combining attribute-based methods with cryptosystems utilizing homomorphic schemes.

I am grateful to David Wu for his unwavering support and guidance throughout the research process; to Zeyu Liu for answering questions on OMR and providing parameters that run much faster for testing; and to Hovav Shacham, Siddhartha Chatterjee, Lorenzo Sadun, and Theresa Martines for their work as part of the honors thesis committee and coordinating the defense process.

References

[RAD78] Ronald L. Rivest, Len Adleman, and Michael L. Dertouzos. *On Data Banks and Privacy Homomorphisms*. https://luca-giuzzi.unibs.it/corsi/Support/papers-cryptography/RAD78.pdf. 1978.

[SW04] Amit Sahai and Brent Waters. *Fuzzy Identity Based Encryption*. Cryptology ePrint Archive, Paper 2004/086. https://eprint.iacr.org/2004/086. 2004.

[Goy+06] Vipul Goyal et al. *Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data*. Cryptology ePrint Archive, Paper 2006/309. https://eprint.iacr.org/2006/309. 2006.

[PVW07] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. *A Framework for Efficient and Composable Oblivious Transfer*. Cryptology ePrint Archive, Paper 2007/348. https://eprint.iacr.org/2007/348. 2007.

[Gen09] Craig Gentry. "Fully Homomorphic Encryption Using Ideal Lattices". In: *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*. STOC '09. Bethesda, MD, USA: Association for Computing Machinery, 2009, pp. 169–178. ISBN: 9781605585062. DOI: 10.1145/1536414.1536440.

[Reg09] Oded Regev. "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography". In: *J. ACM* 56.6 (Sept. 2009). ISSN: 0004-5411. DOI: 10.1145/1568318.1568324.

[GKL10] Jens Groth, Aggelos Kiayias, and Helger Lipmaa. "Multi-Query Computationally-Private Information Retrieval with Constant Communication Rate". In: *Proceedings of the 13th International Conference on Practice and Theory in Public Key Cryptography*. PKC'10. Paris, France: Springer-Verlag, 2010, pp. 107–123. ISBN: 3642130127. DOI: 10.1007/978-3-642-13013-7_7.

[FV12] Junfeng Fan and Frederik Vercauteren. *Somewhat Practical Fully Homomorphic Encryption*. Cryptology ePrint Archive, Paper 2012/144. https://eprint.iacr.org/2012/144. 2012.

[LPR12] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. *On Ideal Lattices and Learning with Errors Over Rings*. Cryptology ePrint Archive, Paper 2012/230. https://eprint.iacr.org/2012/230. 2012.

[CBM15] Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. "Riposte: An Anonymous Messaging System Handling Millions of Users". In: *CoRR* abs/1503.06115 (2015). arXiv: 1503.06115.

[LT21] Zeyu Liu and Eran Tromer. *Oblivious Message Retrieval*. Cryptology ePrint Archive, Paper 2021/1256. https://eprint.iacr.org/2021/1256. 2021.

[22] *PALISADE lattice cryptography library (release 1.11.9)*. https://palisade-crypto.org/. 2022.

[CY23] Steph Cheng and Jeriah Yu. *Parallelization of the Number Theoretic Transform*. https://www.cs.utexas.edu/~jeriah/files/parallel_ntt.pdf. 2023.

[LTW23] Zeyu Liu, Eran Tromer, and Yunhao Wang. *Group Oblivious Message Retrieval*. Cryptology ePrint Archive, Paper 2023/534. https://eprint.iacr.org/2023/534. 2023.

[23] *Microsoft SEAL (release 4.1.1)*. https://github.com/Microsoft/SEAL. 2023.