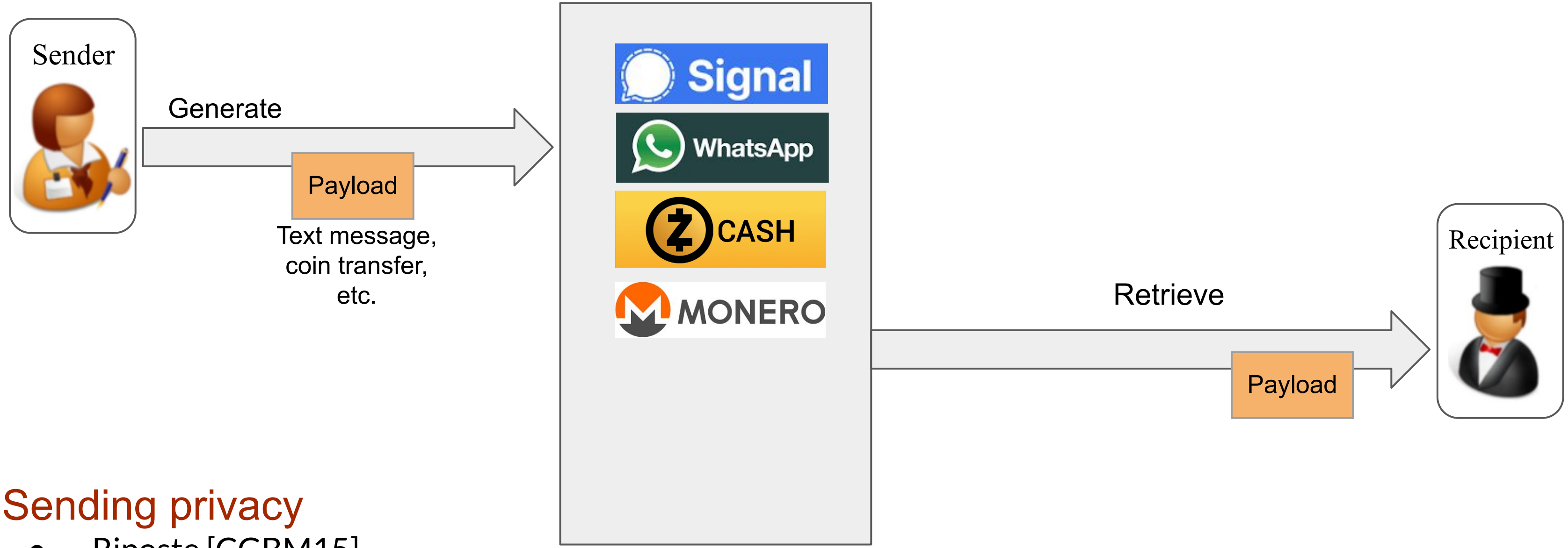
Oblivious Message Retrieval

Zeyu Liu

Eran Tromer

Columbia University

Motivation - anonymous message delivery systems



Sending privacy

- Riposte [CGBM15]
- Dandelion [VFV17]
- Dandelion++ [FVBDBMV18]
- Signal's Sealed Sender [Lun18]
- + improvement [MKARW21]
- Alpenhorn [LZ16] (uses Mixnet + IBE to establish "mailbox" bulletins)

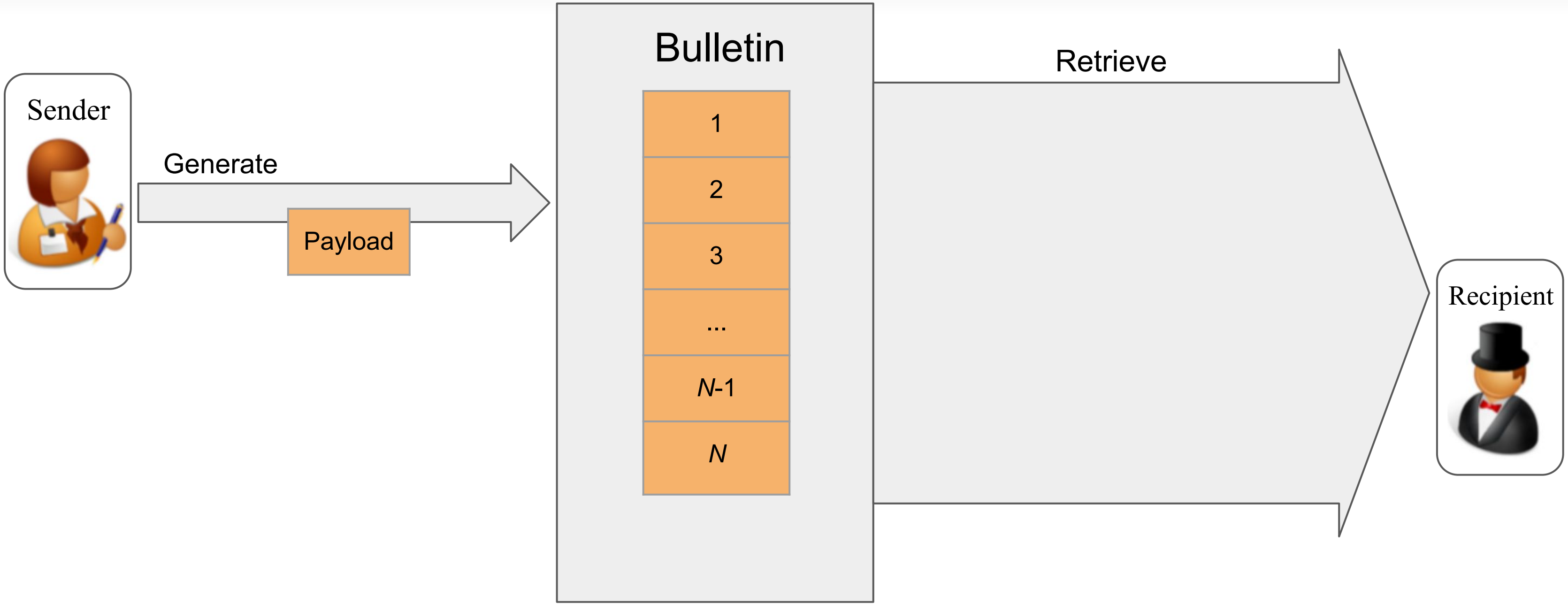
Bulletin privacy

- Zerocash [BSCGMTV14,...]
- Monero [Noe15,...]
- ...

Receiver privacy

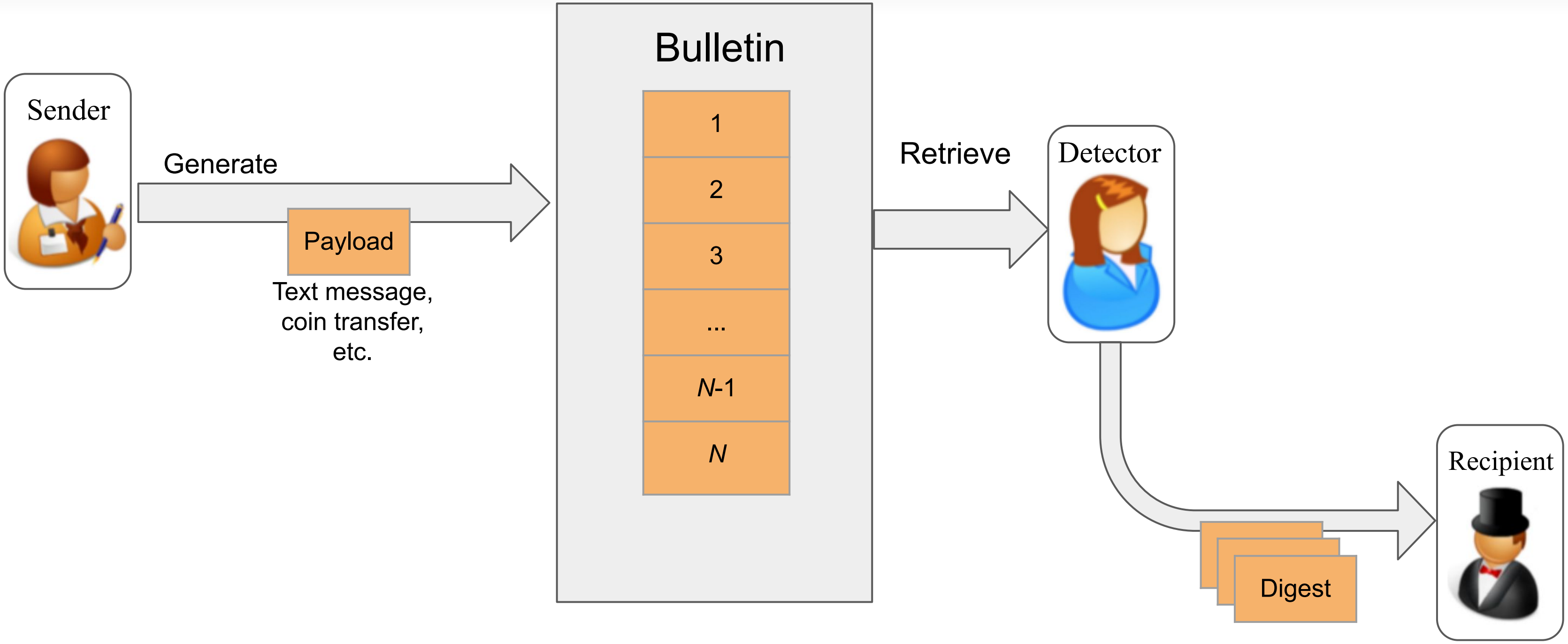
How to receive pertinent messages without leaking metadata?

Message retrieval via full scan



Too expensive in
bandwidth, computation

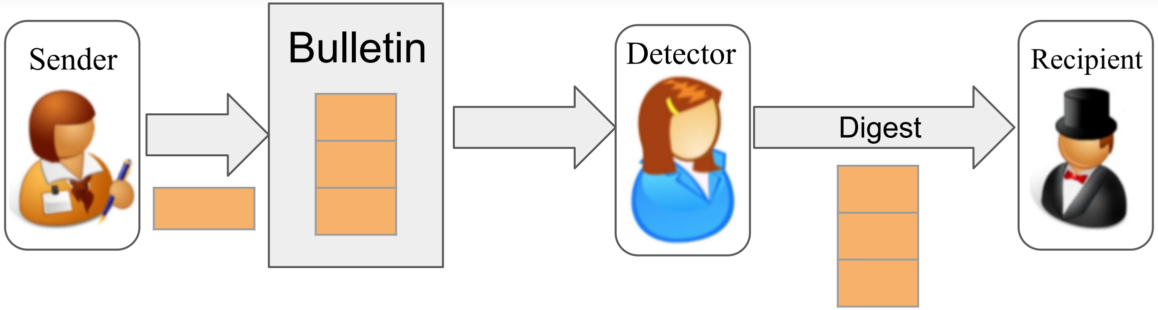
Message retrieval using a detector



Message retrieval using a detector - prior work

Distilled full scan [ZIP-307]

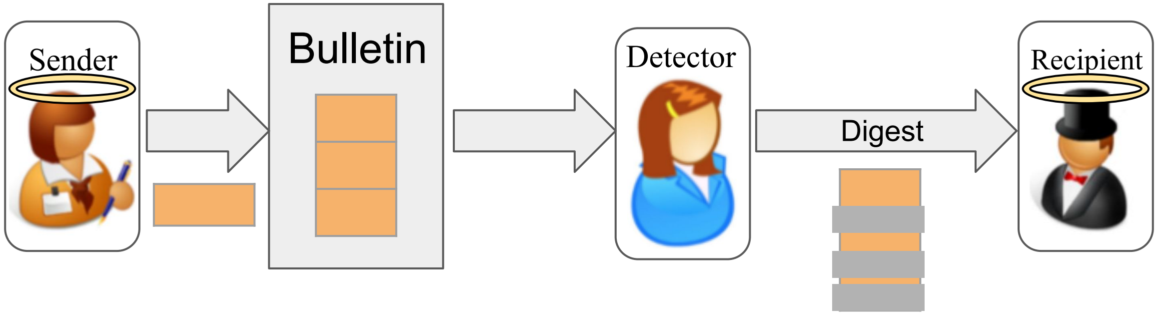
Digest size linear to N



Fuzzy Message Detection [BLMG21]

Decoy-based
→weak privacy [Lew21][SPB21]

Honest senders & recipients

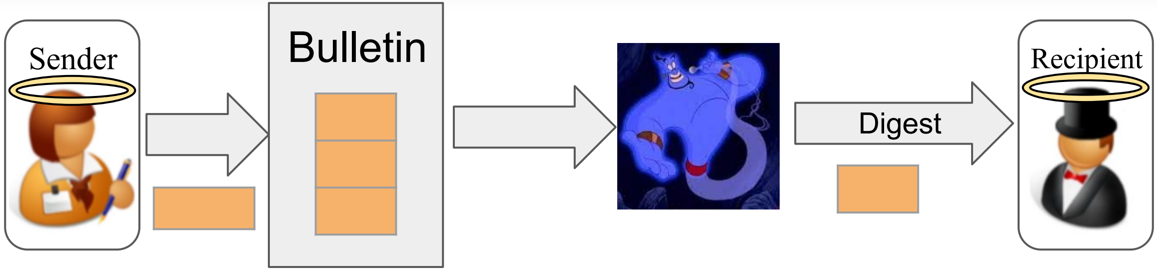


Message retrieval using a detector - prior work (cont.)

Private Signaling 1 [MSSSV21]

Trusted hardware
(e.g., Intel SGX)

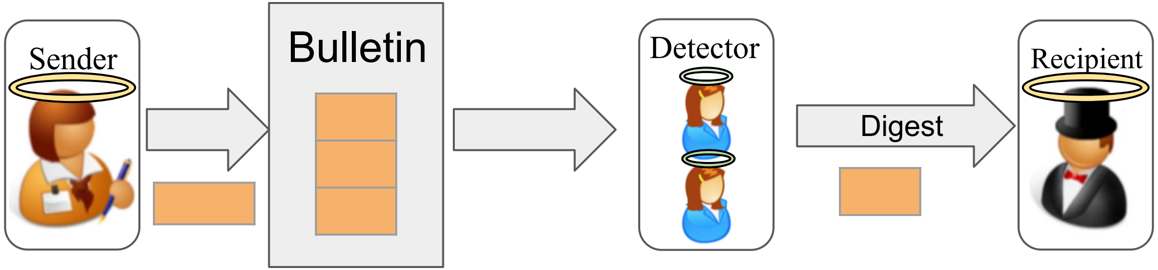
Honest senders & recipients



Private Signaling 2 [MSSSV21]

Two communicating but
non-colluding servers

Honest senders & recipients

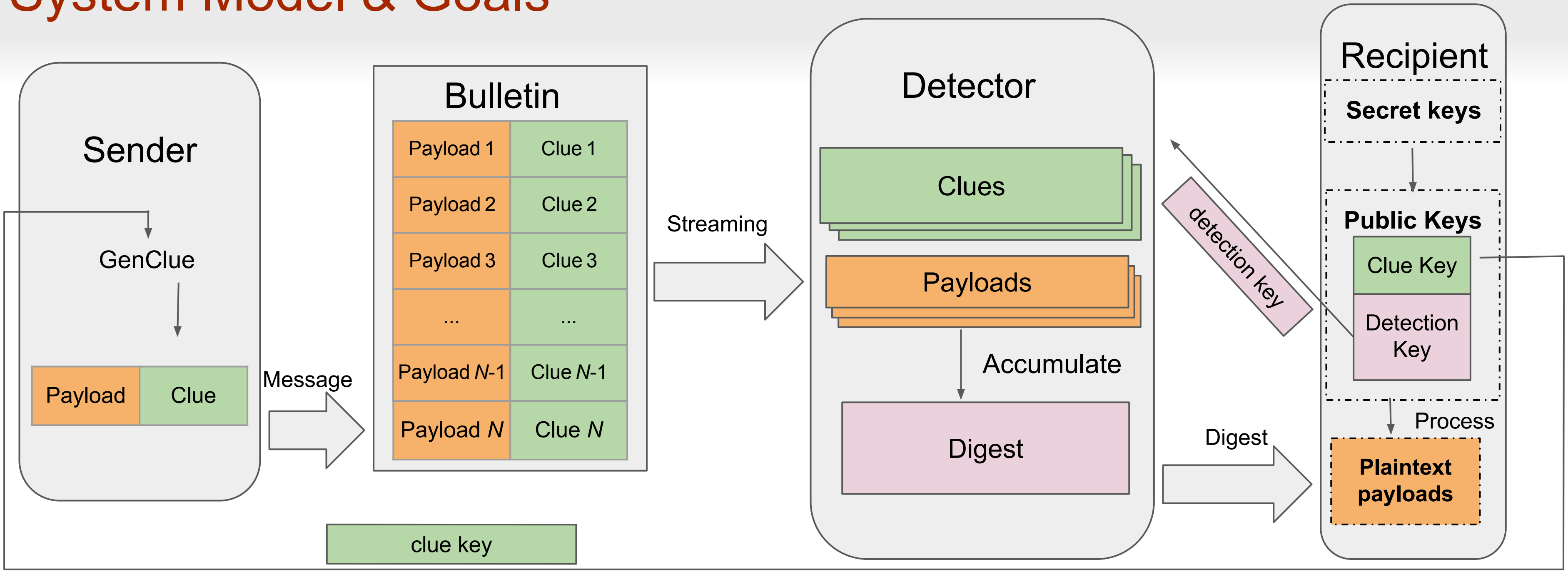


Our results

Oblivious Message Retrieval (and Detection) that is

- fully private
- under strong, hitherto-unachieved security notions
- based on Fully Homomorphic Encryption
- + bespoke application-driven optimizations
- practical for Bitcoin-scale private messaging

System Model & Goals



Functionality:

Oblivious Message Detection (**OMD**)

Oblivious Message Retrieval (**OMR**)

Goals:

- Detector learns nothing about a recipient
  - Which messages are pertinent and which are not
  - Who is doing the retrieval
- Digest size is much smaller than the bulletin size
  (ideally: proportional only to the number of pertinent messages)

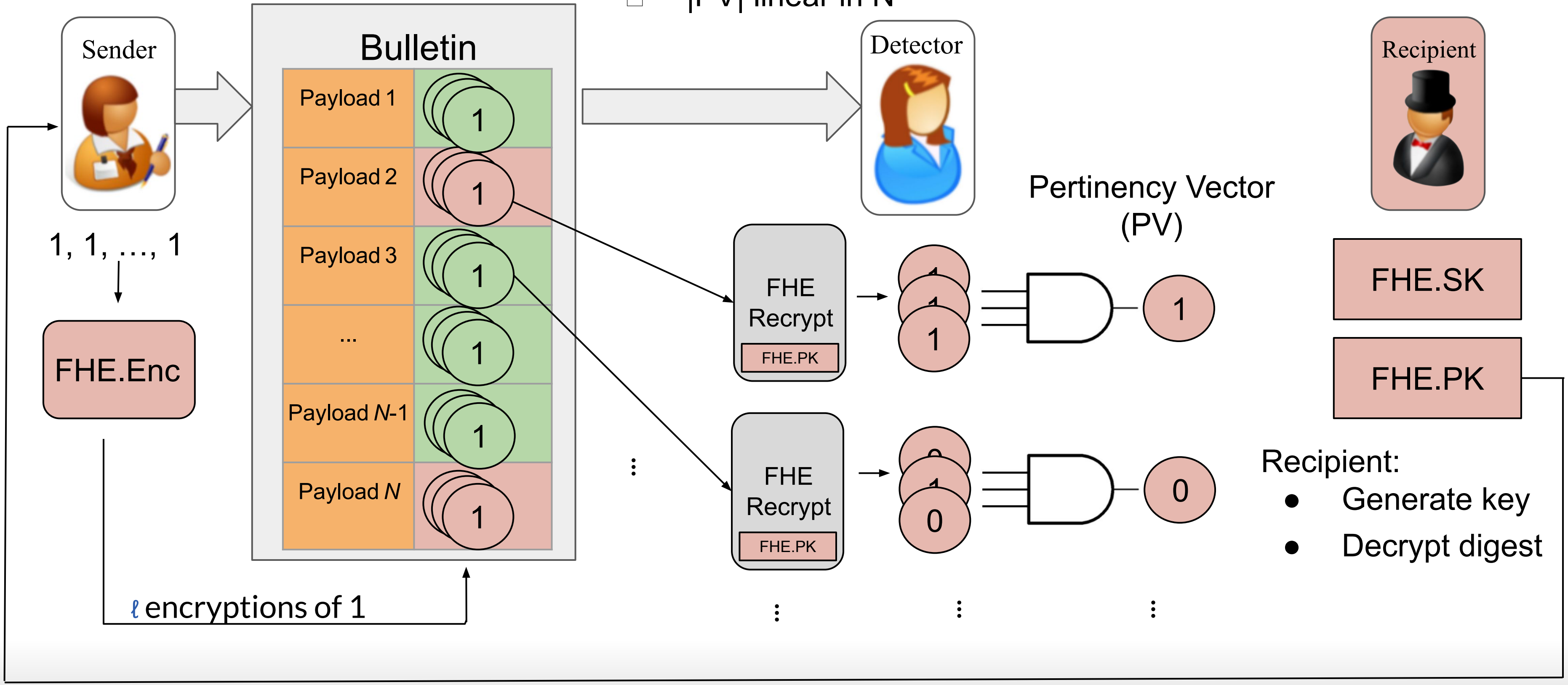
Generic Approach using Fully Homomorphic Encryption

Sender:

- Use recipient's clue key
- Generate ℓ FHE.Enc(1)

Detector:

- Recrypt all ℓ ciphertexts for each clue (totally N clues)
- Use AND gate to compress ℓ ciphertexts into one ciphertext
- |PV| linear in N

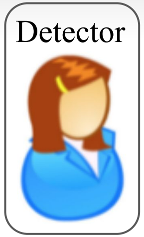


Recipient:

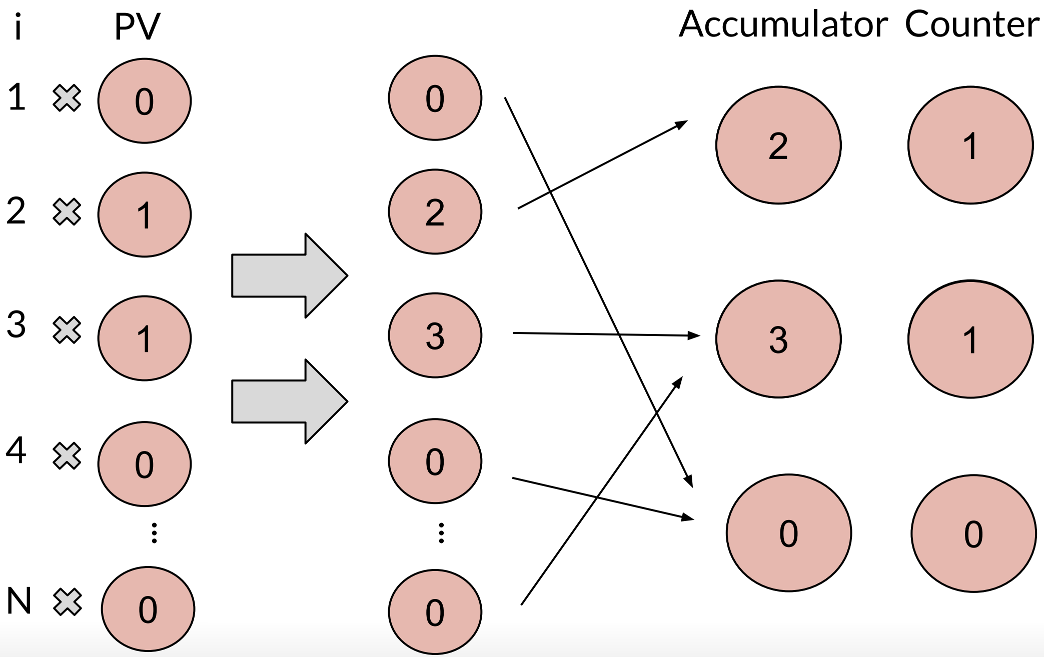
- Generate key
- Decrypt digest

Compressing the Pertivency Vector to o(N)

a la Private Stream Search [OS05]

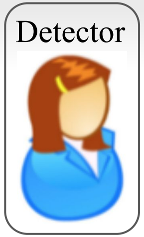


- Initialize m accumulator+counter ciphertexts to 0 (m > k)
- For each message i ∈ [N] and its PV_i (encrypting 0 or 1)
  - Add $PV_i \times i$ to a pseudorandomly-chosen accumulator
  - Increment the corresponding counter by PV_i
- Send the accumulators and counters to the recipient

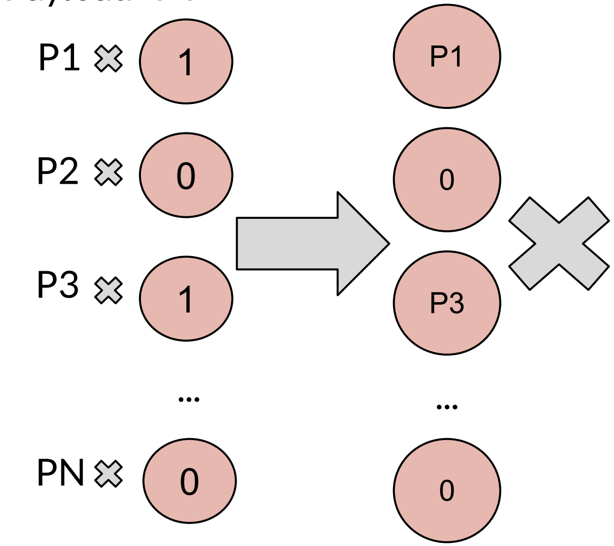


Collisions (counter>1) possible, handled by repetitions and deduction

From detection to retrieval

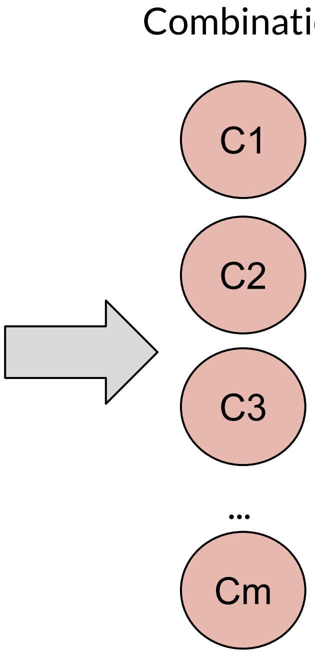


- Compute (PV_i ✖ $Payload_i$)
- Multiply it by a pseudorandom weights matrix of width m
- Yields m encrypted linear combinations of the pertinent payloads.
  If at least k are linearly independent, recipient can solve and obtain all pertinent payloads



Weights

w(1,1)	w(1,2)	w(1,3)	...	w(1,m)
w(2,1)	w(2,2)	w(2,3)	...	w(2,m)
w(3,1)	w(3,2)	w(3,3)	...	w(3,m)
...	...	...	...	w(4,m)
w(N,1)	w(N,2)	w(N,2)	...	w(N,m)



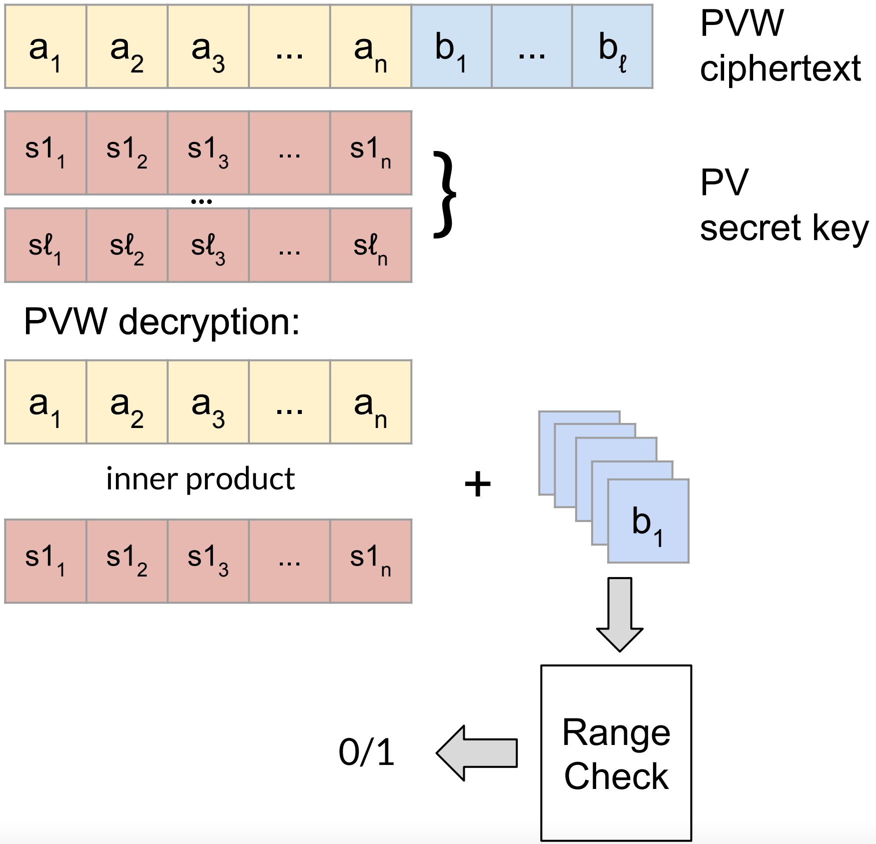
Using suitable **Sparse Linear Random Coding**
Cost per pertinent message is $\tilde{O}(1)$

Thus far

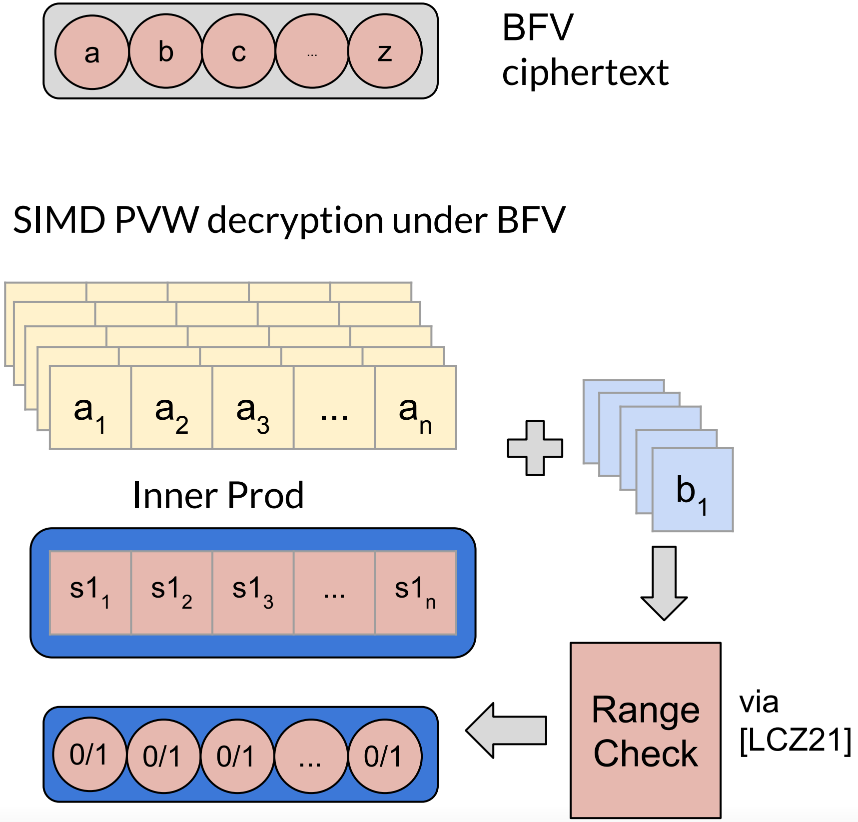
- Generic FHE-based
  - Oblivious Message Detection
  - Oblivious Message Retrieval
    - Asymptotically efficient and succinct
      - o(N) communication cost
      - Õ(N) computational cost
    - Impractical
      - FHE has high computational cost and communication cost
      - Take milliseconds to do an AND gate (TFHE [CGGI20])
      - One ciphertext can be kilobytes (TFHE [CGGI20]) or more

Key optimization: reduce clue size, lightweight recryption

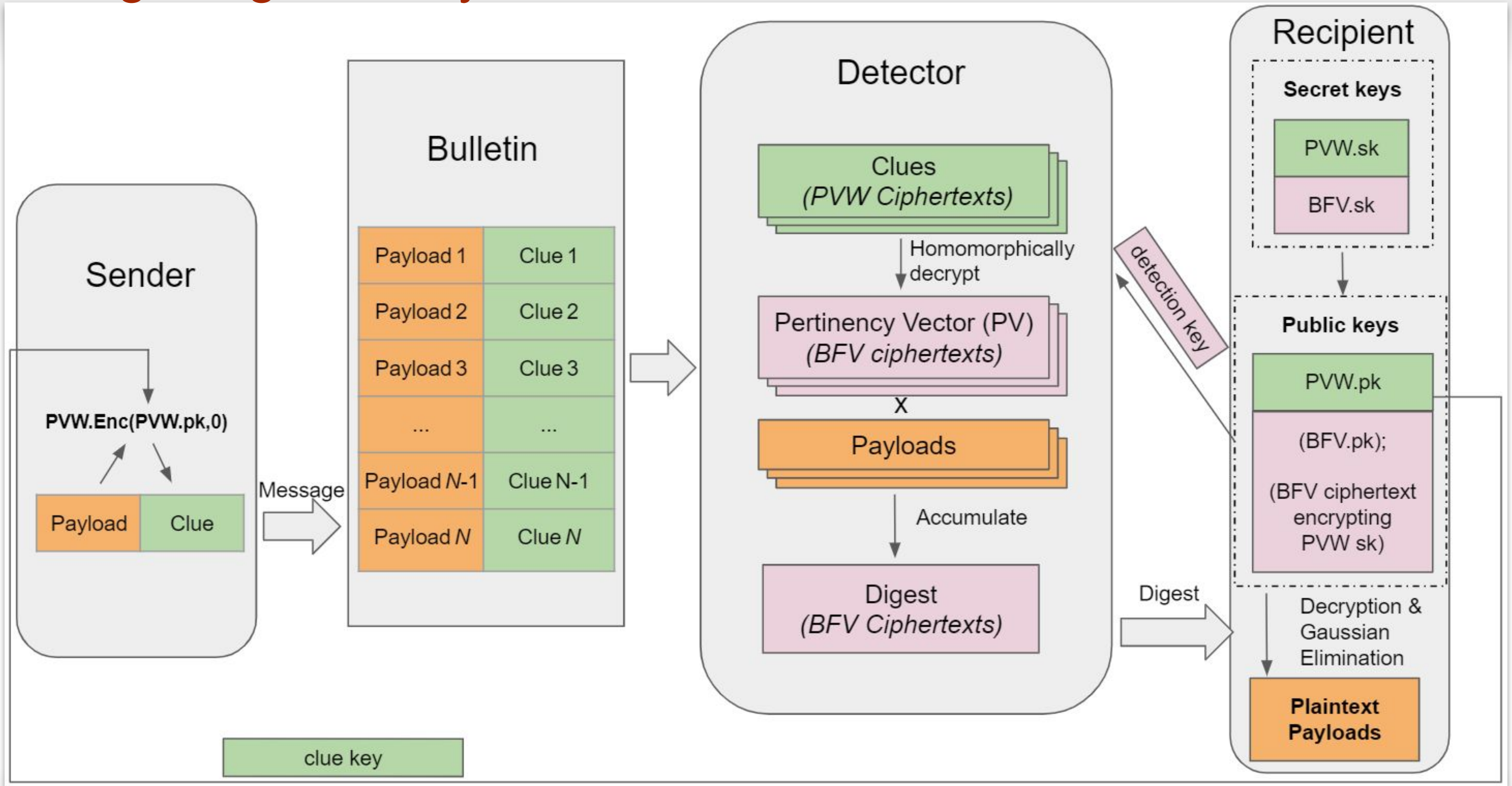
PVW Encryption for ℓ bits [PVW07]



BFV Homomorphic Encryption [Bra12][FV12] supports packed SIMD field operations



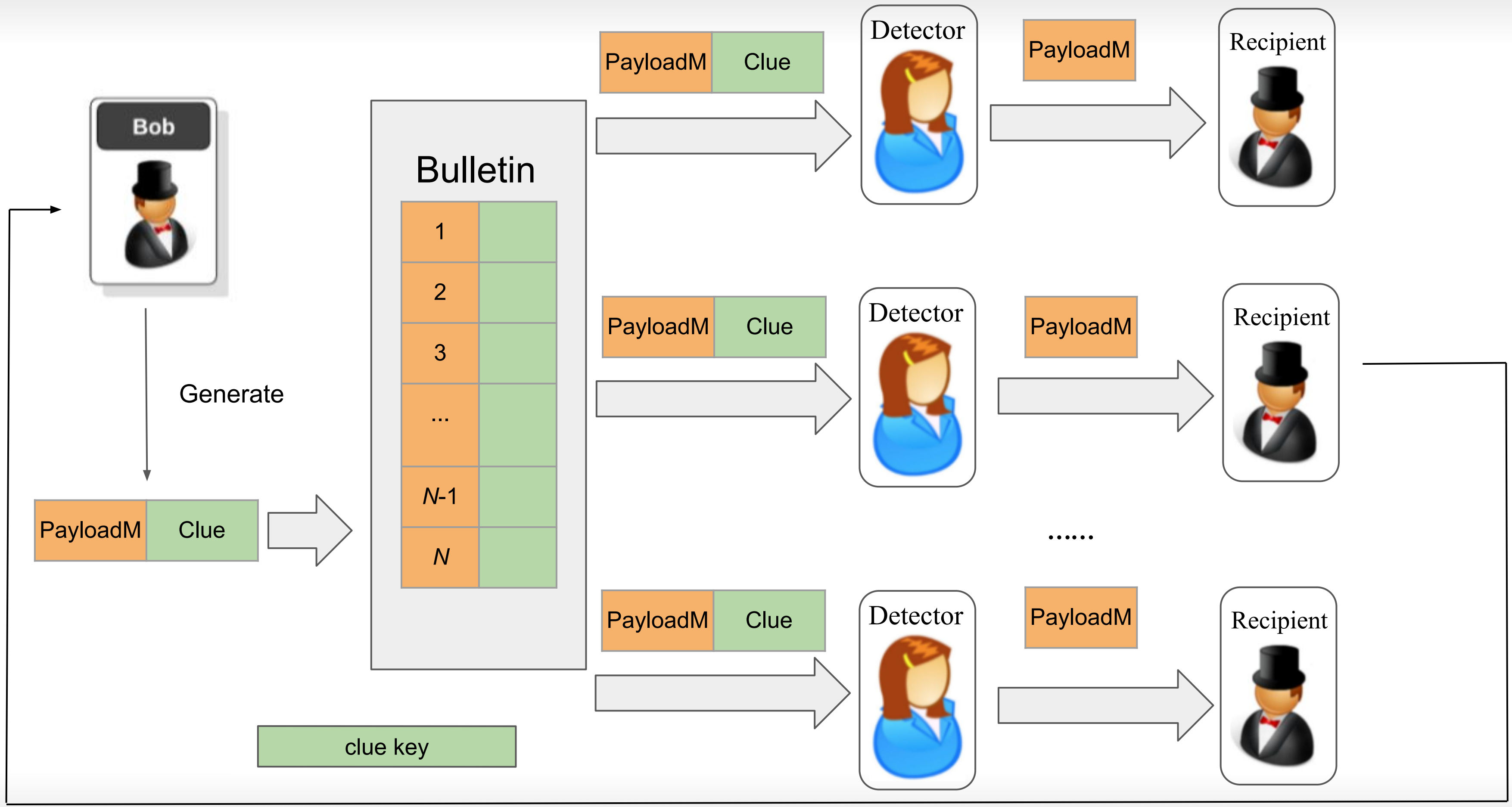
Putting it together: hybrid PVW+BFV OMR/OMD



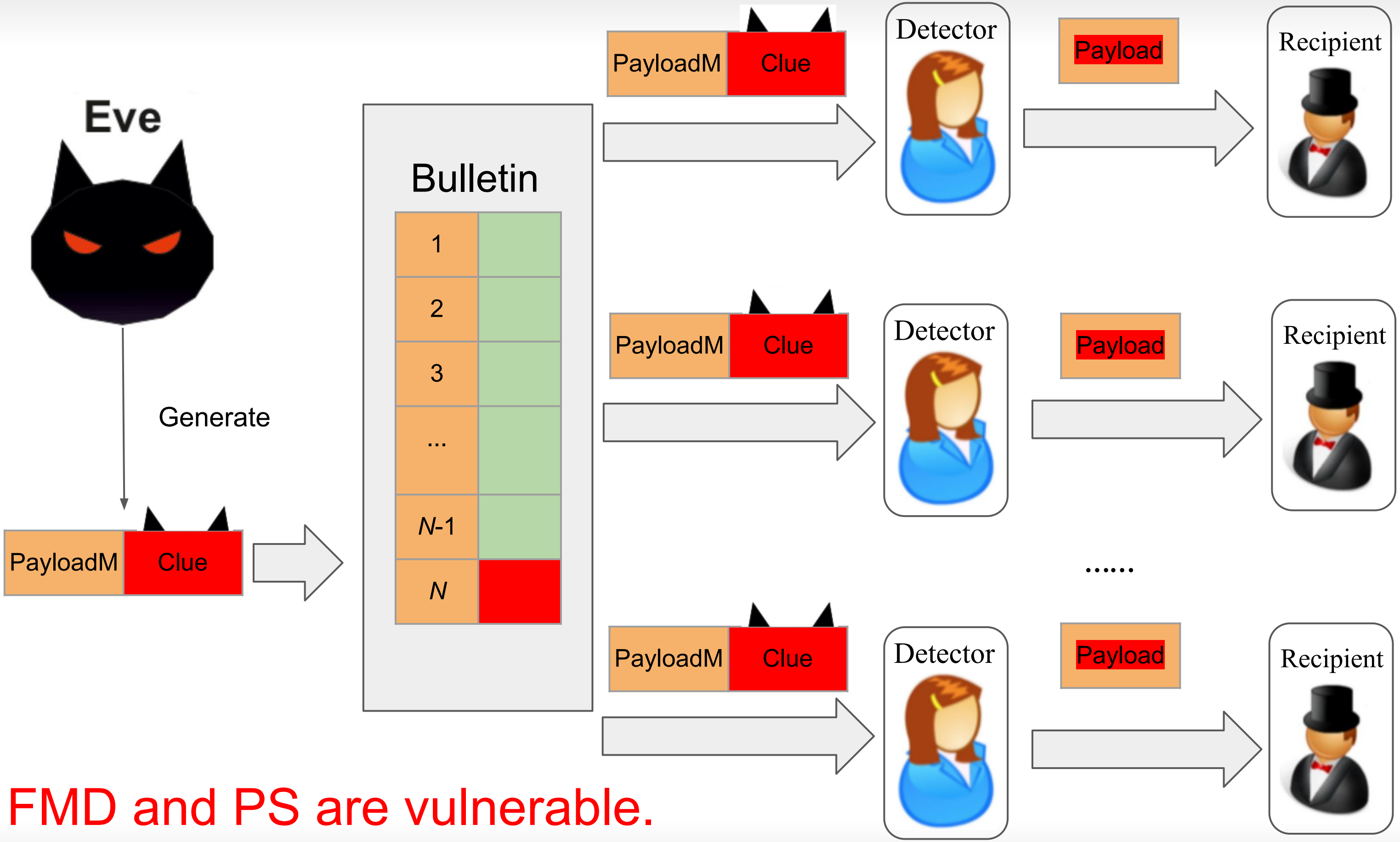
Additional techniques

- FHE tailoring
  - Optimized ladder of moduli
  - Homomorphic operation scheduling (e.g., multiplication vs. rotation)
  - Symmetric BFV encryption
  - Level-specific homomorphic rotation keys
- Scheme optimizations
  - Deterministic bitwise index retrieval
- Application tailoring
  - Memory footprint reduction
  - Streaming updates with low-latency finalization

Denial of Service Attacks

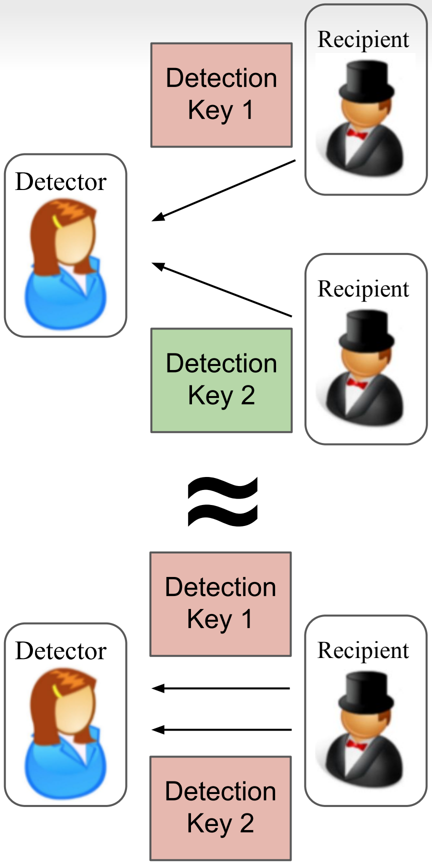


Denial of Service Attacks (mitigated)

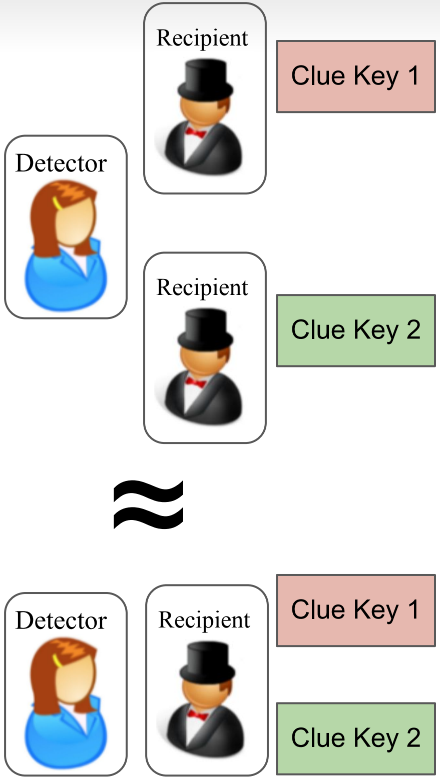


FMD and PS are vulnerable.

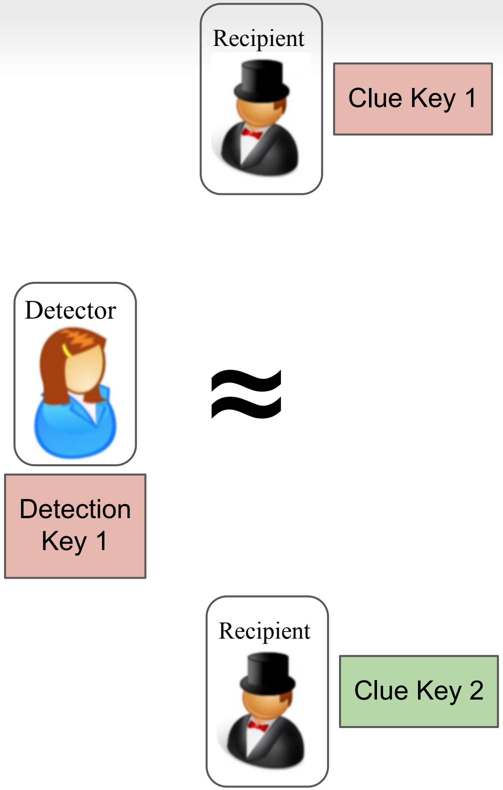
Key unlinkability (defined and attained) FMD and PS are vulnerable.



Detection-key to detection-key unlinkability



Clue-key to clue-key unlinkability



Clue-key to detection-key unlinkability

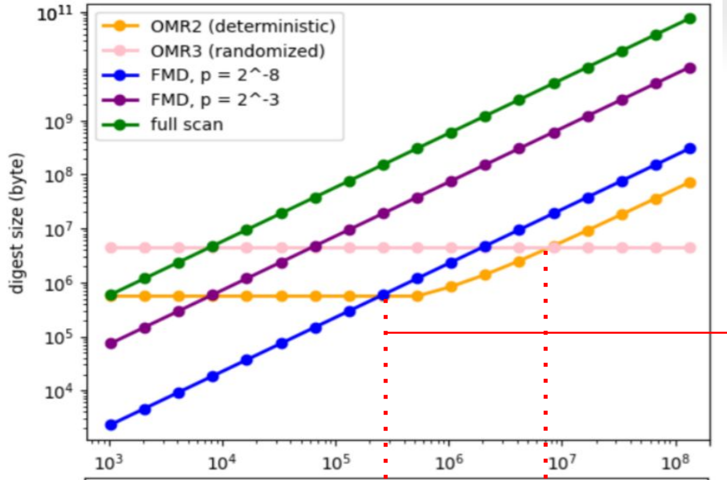
Detection benchmarks (N = 500,000, k = 50)

		Detection schemes			
		ZIP-307 [GH18, Ele]	PS1 [MSS+21]	PS2 [MSS+21]	OMDp1 §7.2
Communication (bytes/msg)		116	$\ll 1$	$\ll 1$ + 3M s↔s	0.56
Detector computation time (sec/msg)	1 thread	N/A	0.06	0.25	0.021
	2 threads				0.01
	4 threads				0.0099
Recipient computation total time (sec)	1 thread	70	$\ll 10^{-3}$	$\ll 10^{-3}$	0.005
Clue size (bytes)		N/A	32	32	956
Clue key size (bytes)		N/A	32	N/A	133 k
Detection key size (bytes)		N/A	64	920	99 M
Retrieval privacy		Full	Full	Partitioned across detectors	Full
Env. assumptions for privacy		None	TEE (SGX)	Non-colluding servers	None
Env. assumptions for Soundness+completeness		None	Honest S&R	Honest S&R	None

Retrieval benchmarks (N = 500,000, k = 50)

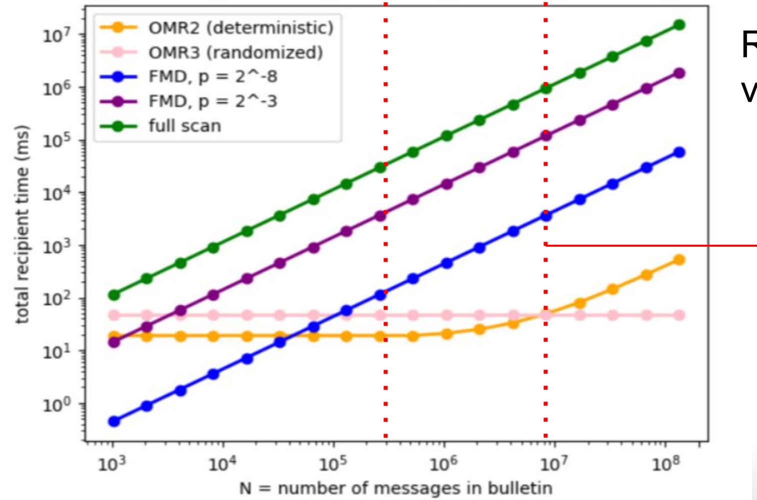
		Retrieval schemes (including detection)				
		Zcash full scan [Ele]	FMD1 [BLMG21] / [Lew21b]	FMD2 [BLMG21]	OMRp1 §7.3	OMRp2 §7.4
Communication (bytes/msg)		612	42	5.3	1.13	9.03
Detector computation time (sec/msg)	1 thread	N/A	0.011 / 0.00020	0.043	0.145	0.155
	2 threads				0.075	0.085
	4 threads				0.065	0.72
Recipient computation total time (sec)	1 thread	61	2.1	0.29	0.02	0.063
Clue size (bytes)		N/A	68 / 64.5	318,530	956	956
Clue key size (bytes)		N/A	1.5 k	1 k	133 k	133 k
Detection key size (bytes)		N/A	768	512	129 M	129 M
Retrieval privacy		Full	pN-msg-anonymity $p = 2^{-5}$	pN-msg-anonymity $p = 2^{-8}$	Full	Full
Env. assumptions for privacy		None	None	None	None	None
Env. assumptions for Soundness+completeness		None	Honest S&R	Honest S&R	None	None

Scaling of recipient costs



Digest size
vs. number of messages

For N>300,000 messages, our OMR1p has the lowest costs for the recipients



Recipient computation time
vs. number of messages

For N>10,000,000 messages, our OMR2p has the lowest costs for the recipients

… while attaining the strongest privacy guarantees and under minimal environmental/trust assumptions

Real-world prospects

- Concrete retrieval costs
  - $1.52 per million payments scanned (based on GCP cost)
  - $0.029/month for Zcash, $2.46/month for Monero
- Integration considerations
  - Payload Size: for Zcash, 612 bytes sizes
  - Clue Key Distribution
    - embedded in the recipient's public address
    - short URL from which the clue key can be fetched (senders using Tor or IPFS)
  - Clue Embedding
    - 956 bytes, close to a Zcash shielded transaction (which is 1.3kB)
    - extend the transaction format with a dedicated clue field
    - other ways like OP_RETURN field in Zcash transactions
  - Detection Latency
    - Streaming updates reduces latency to 0.0005 core-seconds/msg

Ongoing work

- Reducing detection cost
- Reducing size of clue, clue key, detection key
- DoS-Resistance from standard assumptions
- Integrity against fully-malicious detectors
- Group messaging
- Integrations

Paper

ia.cr/2021/1256

Code

github.com/ZeyuThomasLiu/ObliviousMessageRetrival