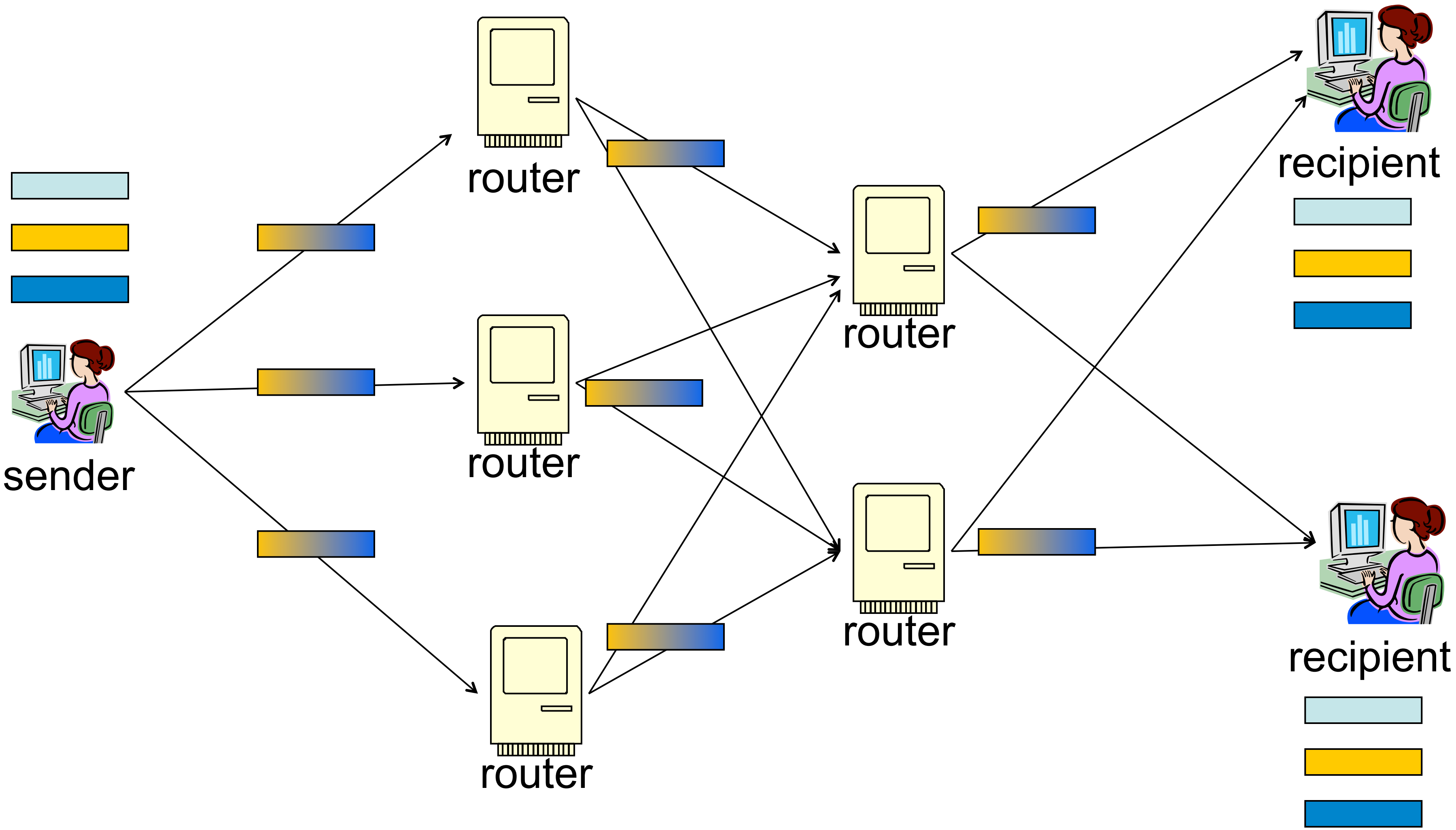
Signing a Linear Subspace:
Signature Schemes for Network Coding

David Mandell Freeman
CWI & Universiteit Leiden

IPAM Retreat: Securing Cyberspace
9 June 2009

Network coding [ACLY'00]



Applies to online and offline (e.g. BitTorrent) applications

Linear network coding [LYC'03]

To transmit a file F do:

- Write **F** as a sequence of vectors

$$\mathbf{v}'_1, \ldots, \mathbf{v}'_m \in (F_p)^n$$

- *Augment* each vector:

used for decoding

$$v_1 = (\text{---}\ v_1'\ \text{---}, 1,0, \ldots,0,0,0,\ldots,0) \in (F_p)^{n+m}$$
$$v_2 = (\text{---}\ v_2'\ \text{---}, 0,1, \ldots,0,0,0,\ldots,0)$$
$$\vdots$$
$$v_i = (\text{---}\ v_i'\ \text{---}, 0,0, \ldots,0,1,0,\ldots,0)$$
$$\vdots$$
$$v_m = (\text{---}\ v_m'\ \text{---}, 0,0, \ldots,0,0,0,\ldots,1)$$

- Transmit $\mathbf{v}_1, \ldots, \mathbf{v}_m$ into the network.

Each intermediate node: receives $\mathbf{w}_1, \ldots, \mathbf{w}_t \in (F_p)^{n+m}$

- chooses random constants $a_1, \ldots, a_t \in F_p$
- forwards $a_1\mathbf{w}_1 + \ldots + a_t\mathbf{w}_t$ to all its neighbors.

Decoding

Recipient receives vector:

$$\mathbf{w} = (\text{---}\ \mathbf{w}'\ \text{---}\ ,\ \underbrace{c_1, \ldots, c_m}_{\text{augmented coordinates}}) \in (F_p)^{n+m}$$

Then $\mathbf{w}' = c_1\mathbf{v}'_1 + \ldots + c_m\mathbf{v}'_m \in (F_p)^n$

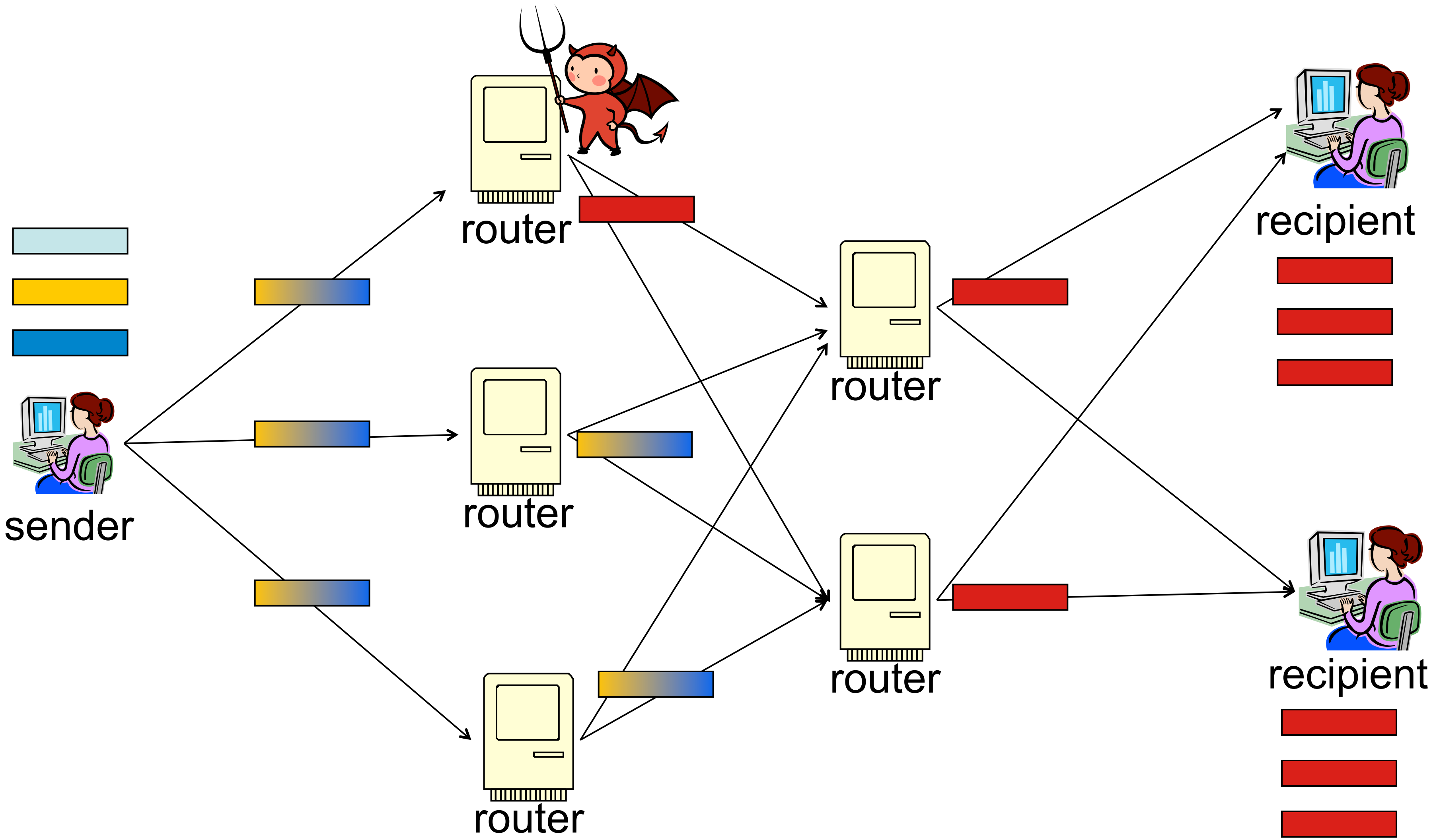
$\Rightarrow$ Recipient can recover $\mathbf{v}'_1, \ldots, \mathbf{v}'_m$ from <u>any</u> *m* vectors that form a full rank system

- i.e. any basis of the subspace spanned by $\mathbf{v}_1, \ldots, \mathbf{v}_m$

Benefits: achieves channel capacity and is resilient to packet loss

The pollution problem

- Just one corrupt router can pollute the entire network!



Some non-solutions:

Sign each basis vector v_i:

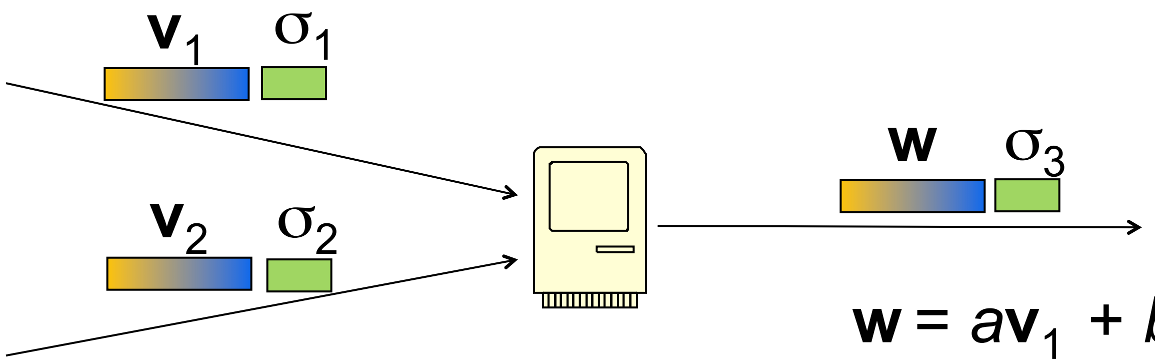
- Received vectors are different from basis vectors
  ⇒ signatures useless.

Sign original file F; then verify signature after decoding:

- Problem: suppose $t > m$ packets are received.
  Recipient must try $\binom{t}{m}$ subsets until a subset
  containing only valid vectors is found.

Signatures for network coding

Linearly homomorphic signatures:



$\mathbf{w} = a\mathbf{v}_1 + b\mathbf{v}_2$

$\sigma_3 = \mathbf{combine}(a, \sigma_1, \; b, \sigma_2)$

- Can obtain signatures on all vectors in span($\mathbf{v}_1, \ldots, \mathbf{v}_m$).
- Hop-by-hop containment:
  every node can verify signature before forwarding vector.
- Recipient drops all vectors with an invalid signature.

Related work

Early proposals:

Krohn, Free**d**man, and Mazières (2004)

Zhao, Kalker, Médard, and Han (2007)

Charles, Jain, and Lauter (2006)

- All are one time signatures:
  PK must be refreshed after every transmission.

- First two schemes generate large signatures:
  m group elements per vector.

Our contributions
(PKC 2009, joint with D. Boneh, J. Katz, B. Waters)

- Well-defined security model for network coding.

  Supports many-time use of a single PK.

- Two efficient schemes secure in our model:

  First is more useful in practice;

  Second has a weaker computational assumption.

- Lower bound on length of secure signatures.

  Our schemes achieve the bound (asymptotically).

Homomorphic network coding signatures

Setup$(1^k, N) \to p, PK, SK$

- Vectors to be signed live in $(F_p)^N$.

Sign$(SK, id, \mathbf{v} \in (F_p)^N) \to \sigma$

- *id*: identifier that binds together all vectors in a file.
- To sign a vector space $V = \text{span}(\mathbf{v}_1, \ldots, \mathbf{v}_n)$, choose *id* and run: $\text{Sign}(SK, id, \mathbf{v}_1), \ldots, \text{Sign}(SK, id, \mathbf{v}_n)$.

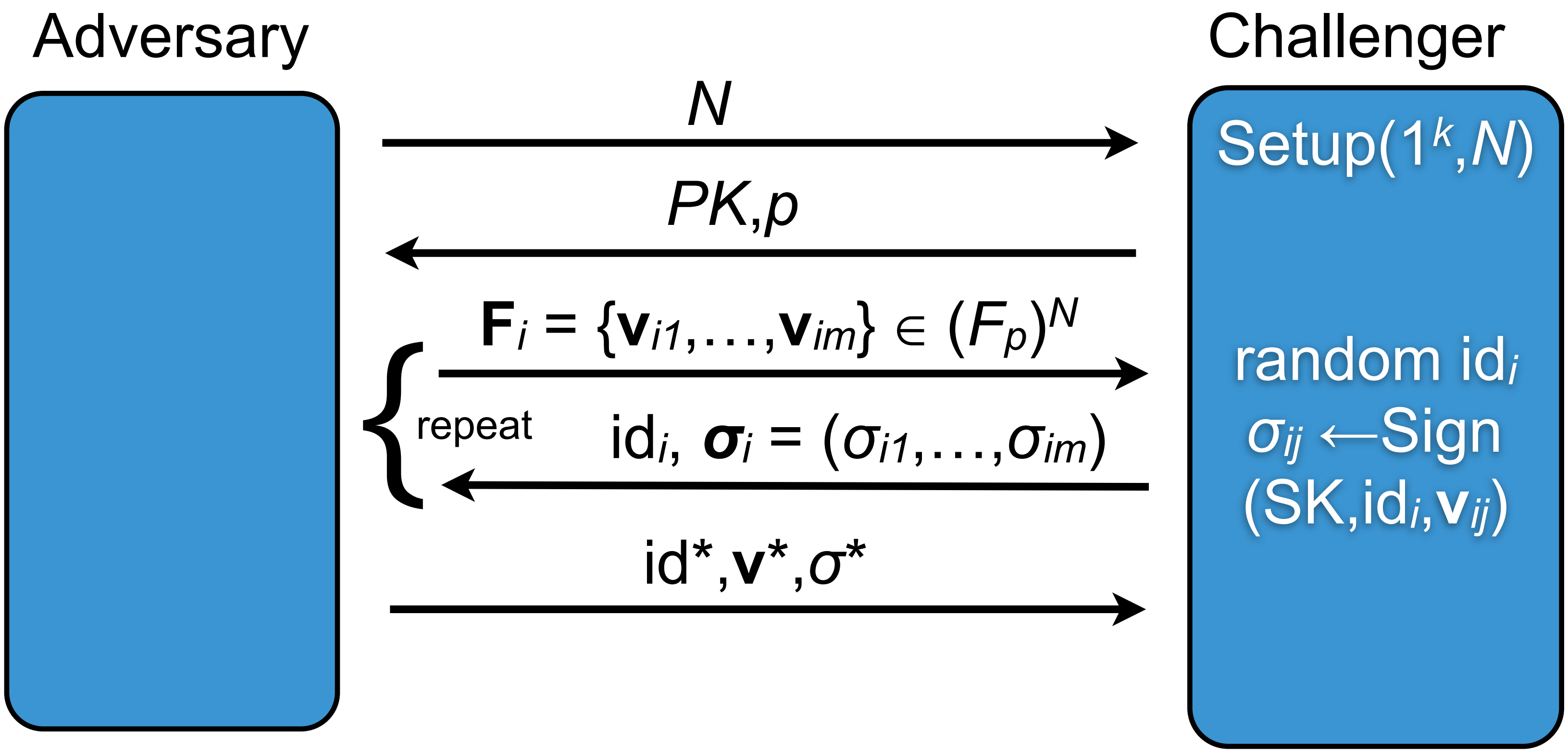
Verify$(PK, \text{id}, \mathbf{v}, \sigma) \to \{0,1\}$

- Checks if σ is a valid signature on $\mathbf{v}$ for identifier *id*.

Combine$(PK, \text{id}, (a, \sigma_1), (b, \sigma_2)) \to \sigma \quad (a, b \in F_p)$

- If σ_1, σ_2 are sigs. for $\mathbf{v}$, $\mathbf{w}$, resp., both with identifier *id* then σ should be a valid signature for $a\mathbf{v} + b\mathbf{w}$.

Network coding security game



Adversary wins if:

Verify(PK,id*,$\mathbf{v}$*,σ*) = 1 and

(1) id*$\neq$ id$_i$ for all i, or

(2) id*= id$_i$ for some i, and $\mathbf{v}^* \notin$ span($\mathbf{F}_i$)

The scheme
(model: BGLS aggregate signatures)

Setup$(1^k, N) \rightarrow$ groups G_1, G_2, G_T of order $p > 2^k$; pairing e ;
hash function $H : \{0,1\}^* \times \{0,1\}^* \rightarrow G_1$

- SK = random $\alpha \in F_p$
- $PK = (h,u)$: h generates G_2, $u := h^\alpha$

Sign$(\alpha, id, \mathbf{v} = (v_1, \ldots, v_m)) \rightarrow \sigma := \left(\prod_{i=1}^{N} H(\text{id}, i)^{v_i} \right)^\alpha$

Verify$(h, u, \text{id}, \mathbf{v} = (v_1, \ldots, v_m), \sigma)$:

- compute $\gamma_1 = e(\sigma, h)$
- compute $\gamma_2 = e\left(\prod_{i=1}^{N} H(\text{id}, i)^{v_i}, u \right)$
- output 1 if $\gamma_1 = \gamma_2$, else output 0.

The homomorphic property

- Given **v** = (v_1,...,v_m) and **w** = (w_1,...,w_m), we have

$$\sigma_1 = \left(\prod_{i=1}^{N} H(\text{id}, i)^{v_i}\right)^{\alpha}, \qquad \sigma_2 = \left(\prod_{i=1}^{N} H(\text{id}, i)^{w_i}\right)^{\alpha}$$

- Signature on *a***v** + *b***w** is

$$\left(\prod_{i=1}^{N} H(\text{id}, i)^{av_i + bw_i}\right)^{\alpha} \quad = \quad \sigma_1^a \cdot \sigma_2^b$$

- So the **Combine** algorithm should be

Combine(*PK*,*id*,(*a*,σ_1),(*b*,σ_2)) = $\sigma_1^a \cdot \sigma_2^b$

Security of the signature scheme

Security is based on *co-computational Diffie-Hellman problem* (co-CDH):

- Given $g \in G_1$, $h \in G_2$, $h^x \in G_2$, compute $g^x \in G_1$.

Theorem: the above signature scheme is secure in our networking coding security model, assuming

- (1) co-CDH is infeasible in (G_1, G_2) and
- (2) the hash function H is modeled as a random oracle.

Proof idea (the interesting case):

- Adversary produces a forgery $(id^*, \mathbf{v}^*, \sigma^*)$ where $id^* = id_i$ from i^{th} query, but $\mathbf{v}^* \notin \text{span}(\mathbf{F}_i)$.
- Challenger uses linear independence to extract co-CDH solution.

A lower bound on signature length

Theorem:

- If bit length of signatures on *m*-dimensional subspaces of $(F_p)^N$ is $\leq \; m \log_2 p - 4m/p - 1$ then there is an adversary that makes one query and wins the security game with probability 1/2.
- i.e., per-vector signature length must be (roughly) $\geq \log_2 p$.

Our scheme achieves the lower bound (asymptotically)

- Assuming "optimal" pairing-friendly elliptic curves are used
  - 160-bit: Miyaji-Nakabyashi-Takano
  - 224-bit: Freeman
  - 256-bit: Barreto-Naehrig

More on the lower bound

Proof of the theorem (sketch)

- Number of m-dimensional subspaces of $(F_p)^N$ is $\approx p^{mN}$.
- If signatures are short, then many files have *trivial* signature (i.e., verifies for *all* vectors).
- Adversary chooses a random subspace V, obtains the signature σ, and produces a vector $\mathbf{v} \notin V$.
- With high probability σ is trivial and thus verifies on $\mathbf{v}$.

Further results
(joint with S. Agrawal, D. Boneh, X. Boyen)

What if multiple senders, each with their own PK/SK, want to send files via the network?

- Natural generalization of single-source security model can't be satisfied.

    Adversary that corrupts one sender can "frame" honest senders.

- Transmission *can* be secure if file ids are crypto-graphically generated.

    Add "IdTest" algorithm to allow recipient to verify ids.

- We construct a secure scheme based on the discrete log assumption.

    Not very efficient.

Open Problems

- Generalize (more efficient) pairing-based scheme to multi-source setting.
- Prove lower bound for multi-source scheme.
- Authenticate vectors with entries in rings other than F_p.

  e.g. $\mathbb{Z}_N$ for small *N*; $\mathbb{F}_{2^d}$ for some *d.*