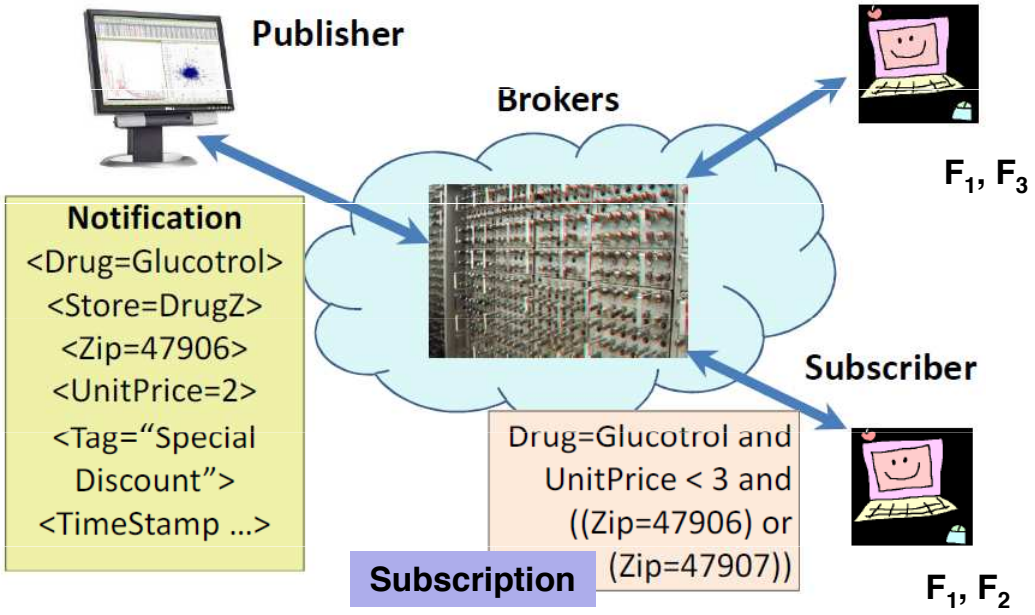
ν-CAPS: A Confidentiality and Anonymity Preserving Routing Protocol for Content-Based Publish-Subscribe Networks

Amiya Kumar Maji and Saurabh Bagchi

Dependable Computing Systems Lab (DCSL) &
The Center for Education and Research in
Information Assurance and Security (CERIAS)
School of Electrical and Computer Engineering
Purdue University

PURDUE UNIVERSITY

Content-Based Publish-Subscribe (CBPS)



Filters = Unique subscriptions across users

PURDUE UNIVERSITY

Advantages of CBPS Networks

- Dynamic many-to-many communication
- Asynchronous
- Publisher-subscriber decoupling
- Fine-grained expression of interest
- Low latency

- Example pub-sub systems
  - Siena [TOCS01]
  - Gryphon [DSN02]
  - RTI Data Distribution Service [www.rti.com/products/dds/]
  - PubSubHubbub [code.google.com/p/pubsubhubbub/]

Security Goals

- Baseline CBPS **trusts** Brokers
  - What if Brokers are compromised (malicious)?
  - What if Publishers, Subscribers do not trust Brokers?
- Can we build an **efficient** CBPS system where
  - Brokers do not know notification content
    - Notification Confidentiality
  - Brokers do not know subscription content
    - Subscription Confidentiality
  - A Subscriber does not know other recipients of a notification
    - Subscriber Anonymity
  - Brokers can learn which filters match a notification only if the filter is present locally
    - Filter Anonymity

Contributions

- Present *v*-CAPS, a secure CBPS routing scheme, consisting of two protocols
  - Routing Vector (RV) Protocol supports
    - Notification Confidentiality
    - Subscription Confidentiality
  - Secure Routing Vector (SRV) Protocol additionally supports
    - Subscriber Anonymity
    - Filter Anonymity
- Deploy SRV, RV, and Baseline (Siena) on PlanetLab and measure their performances

Contents

Threat Model and Assumptions

- Publishers
  - Trusted
- Brokers
  - Honest but curious
- Subscribers
  - Curious
- Assumptions
  - Solution to group key distribution [Prakash et al., Usenix Security 2001]
  - Distributed spanning tree building

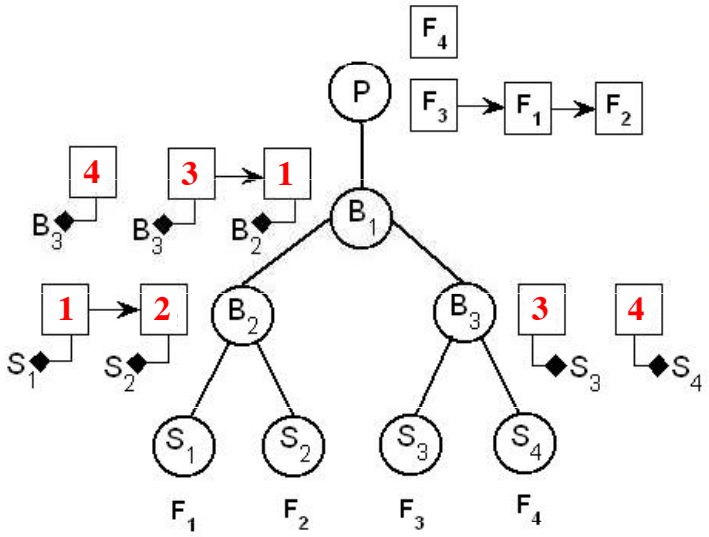
PURDUE UNIVERSITY

Solution Idea

- Observations
  - Filter matching on encrypted notifications is several orders costlier than matching plaintext notifications
  - Brokers in baseline perform two tasks
    - Match notification against filters
    - Compute recipient list of matched filters
- Separation of duty
  - Publisher computes filter *Match*() in plaintext
  - Send result of filter *Match*() to brokers
  - Brokers compute recipient list

PURDUE UNIVERSITY

CBPS Data Structures: Filter Posets



v-CAPS

P: Publisher
B: Broker
S: Subscriber
F: Filter

Covering Relation:

$F_2 \prec F_1 \prec F_3$

$F_2 < F_1$* means *F_2* is more specific than *F_1

***F_1* = (*price* > 5)**
***F_2* = (*price* > 10)**

PURDUE UNIVERSITY

RV Overview

- Replace filters with filterIDs at Brokers
- Publishers maintain filter posets (no network info)
- Primitives
  - Subscribe
    - Phase I:

      Subscriber contacts Publisher

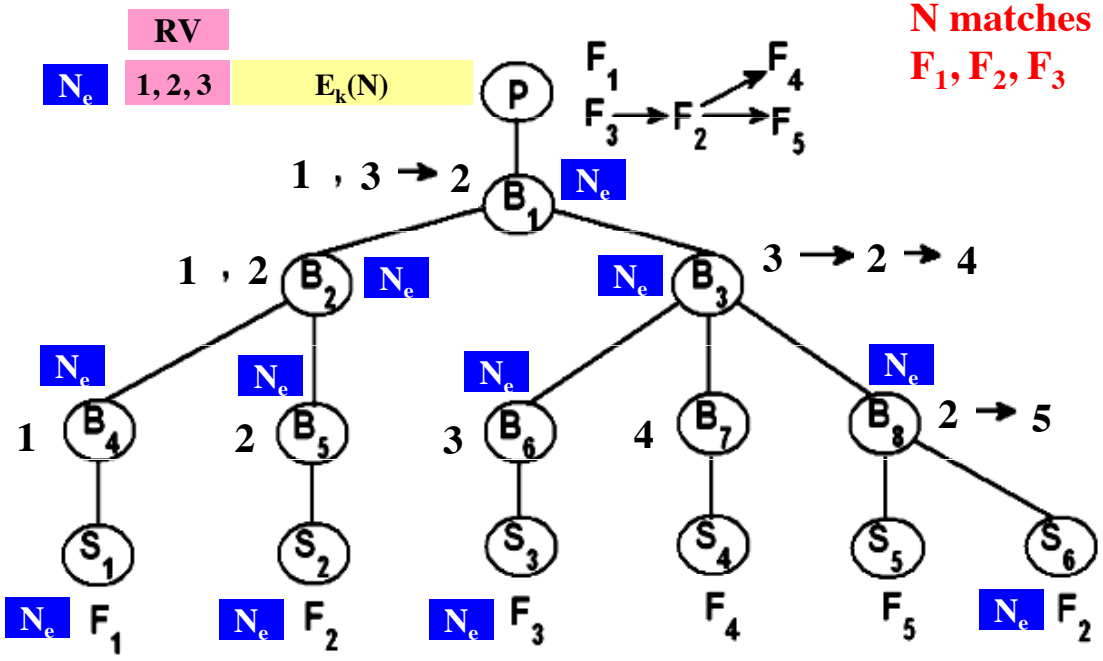
      Gets filterID, location in filter poset
    - Phase 2:

      Propagate subscription message among Brokers based on filterID
  - Publish

    $N_e =$<RV>F_{match}</RV><Payload>$E_{K_N}(N)$</Payload>
  - Match
    - Read F_{match} and compute recipients

PURDUE UNIVERSITY

RV Routing Example



Need for SRV

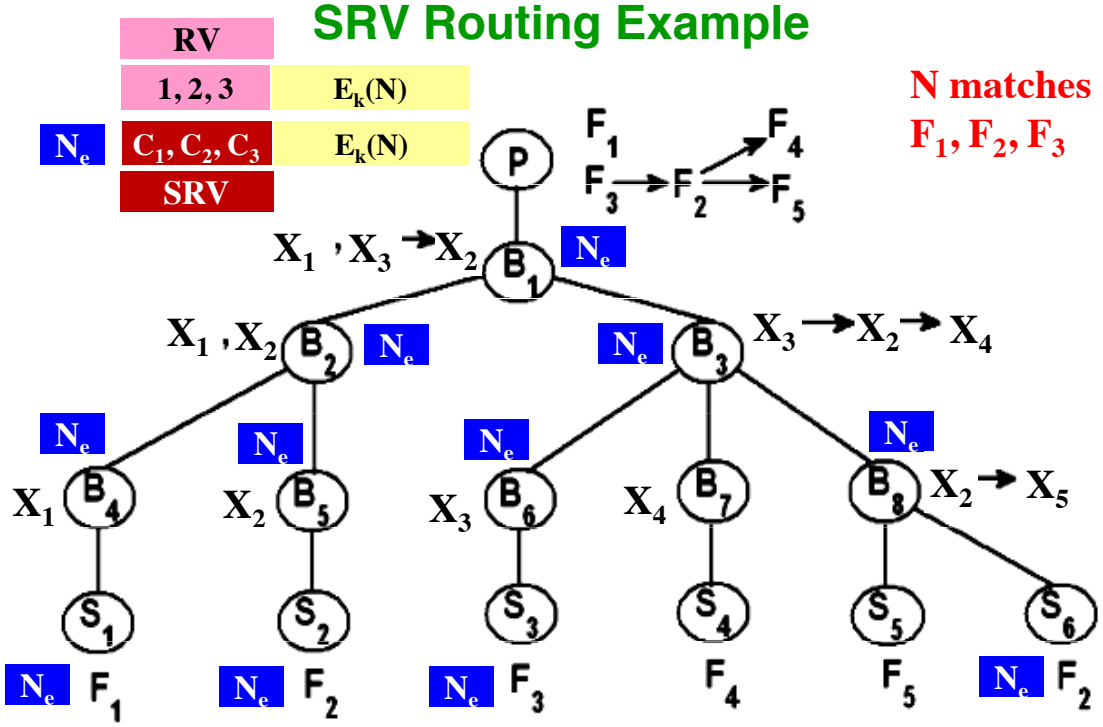
- Brokers can inspect **all** filterIDs in RV
  - Can infer recipient information with external knowledge
- Subscriber *x* **knows** *y* received message with filterID 1
  - Future message with filterID 1 in header (RV) will go to *y*
- Stricter requirements:
  - Brokers should know presence of filters in RV only if they have that filter (Filter Anonymity)
  - Subscribers should not learn commonality across notifications (Subscriber Anonymity)

SRV Overview

- Encrypt RV with encryption technique by Song et al. [S&P 2000]
  - Restricted form of computation on encrypted data
- Does not hamper the generality of matching a filter with a notification
- To detect presence of filterID 1 Brokers need *match key* for 1
- *match key* sent to Brokers during subscription
- Pseudorandom sequence ensures successive invocations of $E_k(1)$ produce different cipher text
- *Match*() is much more expensive than in RV

PURDUE UNIVERSITY

SRV Routing Example



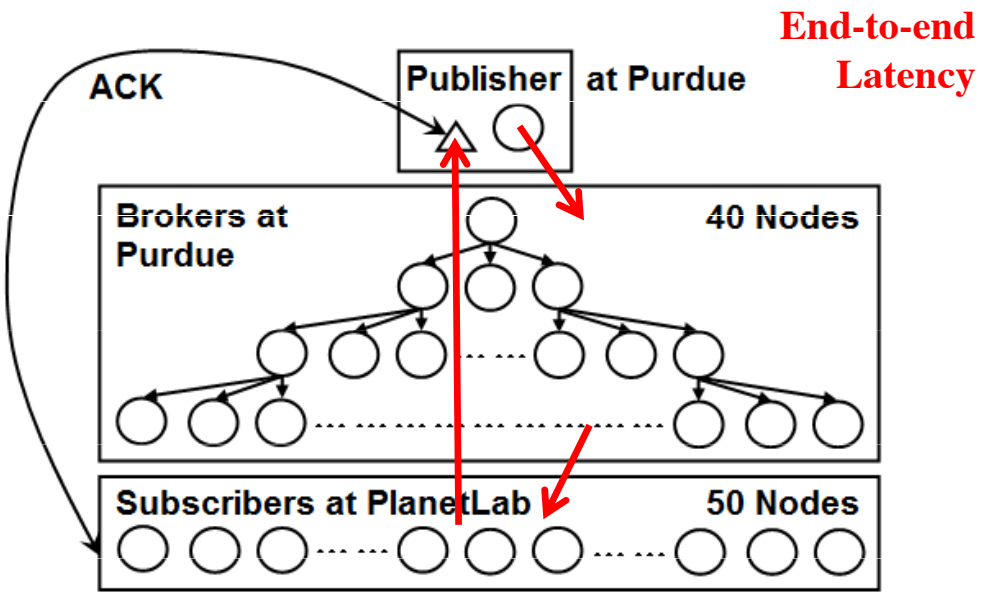
PURDUE UNIVERSITY

Experimental Results

- We measure
  - End-to-end latency for notification propagation
    - Notification Popularity
    - Classify **Popular**, **Moderate**, **Esoteric** based on popularity distribution
  - Computational overhead for notification propagation
  - Subscription cost
    - Compute time for adding a **new** subscription
  - Message overhead
    - Additional bytes per notification per subscriber

PURDUE UNIVERSITY

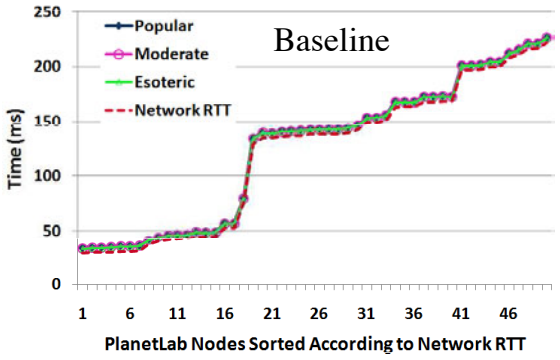
Experimental Setup

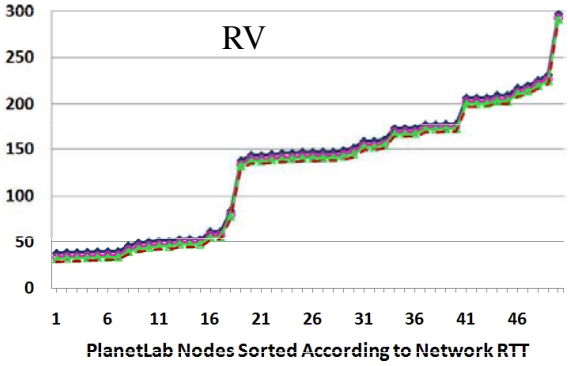


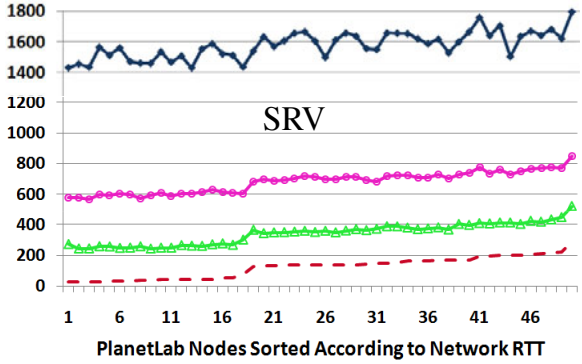
- ***ssbg*** workload generator
- 1000 processes, upto 100,000 subscriptions

PURDUE UNIVERSITY

End-to-end Latency

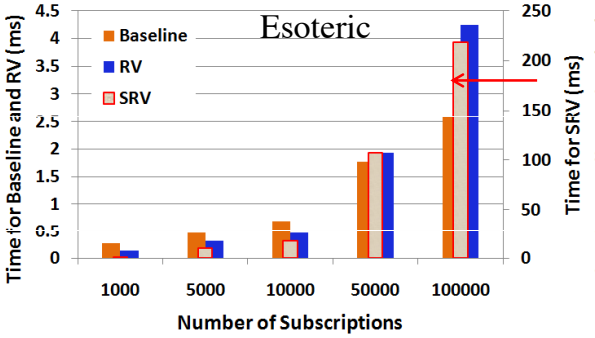


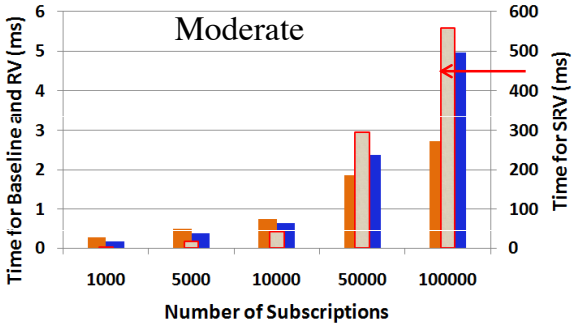


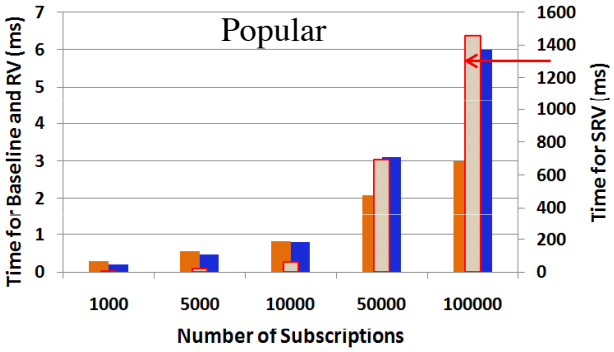


- Workload Size 100,000
- Baseline within ~5ms of RTT
- RV within ~10ms of RTT
- SRV varies with popularity types
- Anonymity has much higher cost than confidentiality alone

Computational Overhead vs Workload Size

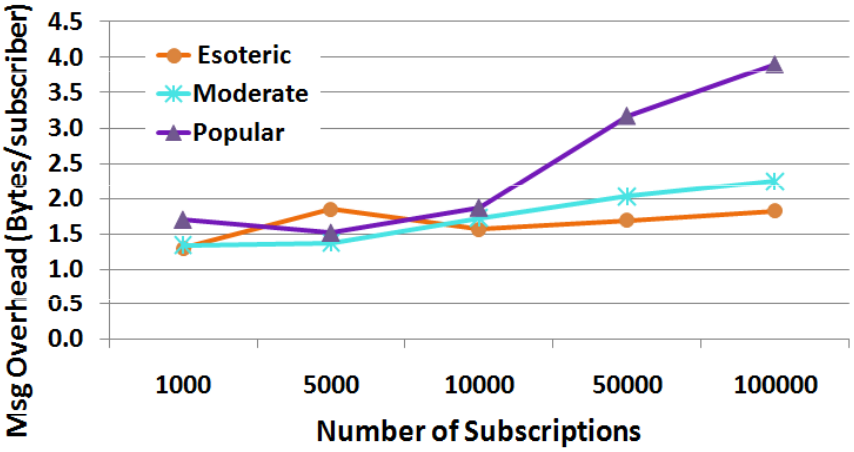






- For 100,000 subscriptions:
  - RV - Baseline = ~3ms
  - SRV takes 220 to1500ms
  - Cost at publisher for SRV ~4ms

Message Overhead



- Worst case cost per subscriber 16 bytes
  - "Virtual destination address"
- CBPS is built on the assumption that filters are subscribed by many subscribers

Comparison with Related Work

- Computation on encrypted data [SecureComm06, Purdue TR09]
  - Expensive in terms of time
  - Misrouting
  - Cannot support full generality of baseline filters
  - Message overhead
- Commutative Encryption [Sec09]
  - Need to send multiple copies of notifications
- v-CAPS can
  - Support full generality of baseline filters
  - Preserve confidentiality with very little overhead (RV)
  - No trusted third-party
- v-CAPS disadvantage
  - Loss of decoupling in Phase I of Subscribe()

Conclusion and Future Work

- Presented **C**onfidentiality and **A**nonymity preserving routing protocol for CB**PS** networks (*v*-CAPS)
- Largest wide-area deployment and experimentation of CBPS protocols
- End-to-end latency of RV is comparable to baseline

- SRV is costly, need to compute *Match*() in parallel for lower latency
- Fault tolerance of Publishers and Brokers
- Anonymizing layer between Subscribers and lowest level Brokers

Thank You