SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2026, Vol. 69, Iss. 9, 190301:1–190301:16
https://doi.org/10.1007/s11432-026-4905-0

Special Topic: User-Centric Intelligent Networking

NameShield: secure and dynamic group communication for first responders in emergency management

Hongmiao YU[1], Jiachen CHEN[2], Silas RICHELSON[1] & K. K. RAMAKRISHNAN[1*]

[1]*Department of Computer Science and Engineering, University of California Riverside, Riverside 92521, USA*
[2]*Wireless Information Network Laboratory (WINLAB), Rutgers University, North Brunswick 08902, USA*

Received 10 January 2026/Revised 24 March 2026/Accepted 9 April 2026/Published online 26 August 2026

Abstract Group communication among first responders is critical to the success of public safety missions. The operational environment of such missions introduces unique requirements, including highly dynamic group membership, tolerance to communication disruptions, the need to maintain confidentiality, and enable auditability. However, commercial group communication applications are typically designed for networks that provide timely and reliable delivery, which do not necessarily satisfy these requirements. Thus, a group communication solution specifically designed for public safety missions—one that accounts for all of these requirements—is needed. We present *NameShield*, a secure and efficient framework for dynamic group communication in public safety and emergency response scenarios. *NameShield* adopts a holistic, layered architecture that integrates security, information dissemination, and networking to support reliable and confidential communication among highly dynamic groups of first responders. At the information layer, *NameShield* builds on a role-based graph namespace to realize an efficient publish/subscribe communication model that naturally captures organizational structure and group dynamics. The network layer of *NameShield* is intentionally decoupled from specific communication technologies and can operate over any substrate that provides multicast, making it suitable for network conditions during public safety missions. To maintain message confidentiality, *NameShield* incorporates a message-oriented key-policy attribute-based encryption (KP-ABE) mechanism that enables fine-grained, per-message access control and supports frequent membership changes without requiring expensive group rekeying. Together, these design choices allow *NameShield* to provide strong security guarantees while remaining flexible, scalable, and robust under the operational constraints of emergency response environments.

Keywords computer communication networks, cryptography and security, publish/subscribe systems, group communication, attribute-based encryption (ABE), public safety communication systems

Citation Yu H M, Chen J C, Richelson S, et al. *NameShield*: secure and dynamic group communication for first responders in emergency management. Sci China Inf Sci, 2026, 69(9): 190301, https://doi.org/10.1007/s11432-026-4905-0

1 Introduction

Efficient and secure group communication plays a critical role in the success of public safety missions, where first responders must exchange information in a timely manner across different groups. Communication in such settings exhibits several distinctive characteristics. (1) Highly dynamic group membership. First responders frequently form temporary, task-oriented groups during public safety missions. For example, in a rescue operation, responders from different departments and roles collaborate to form a team. These groups are highly dynamic in two respects. First, individual responders may rotate in and out of roles depending on availability, causing group membership to change frequently. Second, the set of active roles may itself change over time. For instance, in a distribution center, responders with different functionalities may arrive and depart within short time intervals, continuously reshaping the group. Consequently, message dissemination may often target a different recipient set on a per-message basis. (2) Damaged and unreliable infrastructure. Public safety missions typically take place during or after disasters or crises, when network infrastructure may be partially damaged or congested. Communication links may be unreliable, messages may be delayed, or experience disruption because of heavy traffic from both responders and civilians. As a result, the communication environment often resembles a delay-tolerant network (DTN) [1] rather than a conventional, stable network. (3) Confidentiality and anonymization. Communications among first responders can be highly sensitive and need to be protected from unauthorized disclosure. Leakage of operational information to the public could cause unnecessary panic, and leakage to adversaries could directly compromise mission success. In addition to confidentiality, recipient anonymization is also important, since revealing which teams or roles are communicating can itself allow an adversary to infer ongoing or upcoming operations. For example, observing a

* Corresponding author (email: kk@cs.ucr.edu)

surge in traffic involving a special weapons and tactics (SWAT) team could indicate an imminent tactical action. (4) Auditability and accountability. Finally, communications must be auditable by authorities with the appropriate permissions. Auditability supports post-incident analysis, accountability, and the detection of misbehavior, while still preserving operational security during the mission. These characteristics together motivate the need for a group communication system that supports dynamic membership, resilience to disruption, confidentiality with anonymity, and auditability.

While commercial group messaging applications, such as WhatsApp and Signal, provide strong confidentiality and efficiency guarantees for normal environments, they are not well-suited for the form of group communication in public safety missions that we are concerned about, which exhibit fundamentally different requirements. Commercial messaging platforms do not encode role-based or professional information for group members, often resulting in unnecessary and potentially wasteful message dissemination. In practice, administrators may have to create a large, static group that includes all responders involved in a mission to cover all possible use cases. Messages intended only for responders in a specific role are nevertheless delivered to all members of this large group, including those for whom the information is irrelevant. This behavior not only raises security concerns by exposing sensitive information to unintended recipients but also introduces efficiency and usability problems. Responders who are repeatedly exposed to irrelevant messages may become overwhelmed and fail to notice critical information related to their assigned roles. Commercial group messaging applications are also ill-suited to the highly dynamic nature of public safety operations. Group membership can change frequently as missions evolve. However, commercial systems typically require explicit and often costly group formation and reconfiguration procedures. This introduces significant overhead when members join or leave and prevents flexible, per-message definition of recipient groups. In addition, many commercial messaging systems rely on centralized servers for message delivery, which may be inaccessible during disasters due to infrastructure failures. Furthermore, their security mechanisms assume reliable, ordered, and lossless message delivery. For example, message layer security (MLS) [2] requires a reliable, in-order delivery guarantee, making it poorly suited for the disrupted network conditions common in emergency scenarios. Finally, commercial messaging applications do not support authorized auditing. They typically rely on strict end-to-end encryption, which, while appropriate for preserving confidentiality and user privacy in normal settings, makes legitimate auditing difficult. These limitations highlight the need for a group communication system specifically designed to address the security, dynamism, resilience, and accountability requirements of public safety operations.

Namespace-based group communication [3, 4] is a promising solution for efficient group communication for public safety missions. Unlike commercial group messaging applications, which require explicit group creation and maintenance, namespace-based communication maintains a system-wide namespace and uses it as the fundamental abstraction for group formation. As described in [5], names correspond to the roles of first responders. The namespace is typically derived from real-world administrative and organizational structures. In such systems, the namespace is organized hierarchically, and each name represents a specific role of a first responder. First responders subscribe to the names corresponding to their assigned roles. When a sender publishes a message, it selects one or more names from the namespace, and the system implicitly expands the selected names to include their descendants in the hierarchy. As a result, all subscribers of the selected names and their descendant names receive the message. This design ensures that messages are delivered only to responders whose roles are relevant, reducing both information leakage and unnecessary message dissemination that has the potential of overloading the attention of responders with information not relevant to them. Namespaces also naturally support dynamic group management. As introduced in [4], names in the namespace are managed flexibly within a dynamic graph structure. Names and name-relationships can be added, removed, or modified during a public safety mission. When a task-oriented group needs to be formed, an administrator can simply introduce a new name into the namespace and link existing names to it, thereby creating the desired group structure without explicit group reconfiguration. Another benefit of the namespace is that it reduces the cognitive and operational burden on senders. During public safety missions, the individual responder occupying a particular role may change frequently due to shift rotations or task reassignment. Identifying which individual currently fulfills a given role can therefore impose additional overhead on the sender. With a role-based namespace, senders can target roles directly rather than specific individuals, as responders subscribe to names corresponding to the roles they currently perform. This flexibility makes namespace-based group communication particularly well-suited to the highly dynamic and role-driven nature of public safety operations.

Beyond efficient communication among first responders, which is enabled by the namespace, security is another critical concern. Adversaries may eavesdrop on communication channels to obtain sensitive information, and unauthorized users may gain access to messages due to the open and heterogeneous network environments often used in public safety missions (e.g., in device-to-device networks, where user devices participate directly in message forwarding). To protect sensitive information during transmission, ensuring message confidentiality is essential. Among different encryption schemes, attribute-based encryption (ABE) offers distinct advantages for public safety

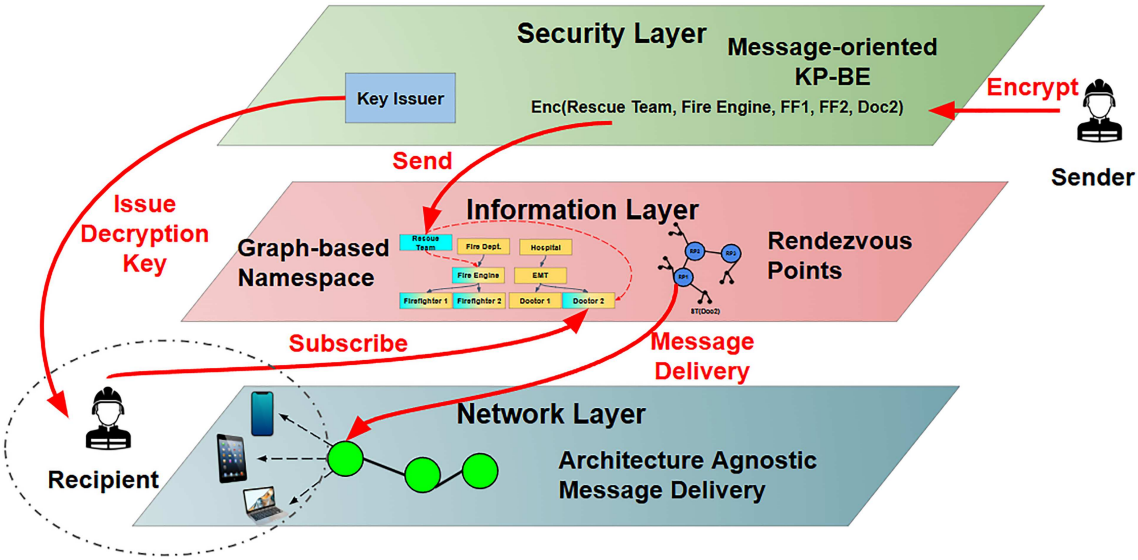

Figure 1 (Color online) Overall architecture of *NameShield* supporting confidential communication among dynamic groups.

missions. The ABE solution provides several advantages over MLS and similar commercial group messaging mechanisms, making it particularly suitable for public safety missions. ABE has several advantages. (1) No group key establishment: MLS and other commercial group encryption schemes maintain a shared group key among all group members, which is used by senders to encrypt messages. Establishing and maintaining this key requires a complex protocol that relies on ordered and lossless control message delivery and must be updated whenever membership changes. In contrast, ABE supports one-to-many encryption: a sender can encrypt a message using a public key and specify the intended recipients by including their role names as attributes in the encryption policy. These attributes can be changed on a per-message basis, eliminating the need for a shared group key and enabling efficient support for highly dynamic groups. (2) Fine-grained access control: ABE allows flexible expression of access policies over these attributes, enabling fine-grained control over who can decrypt a message. For example, a timestamp attribute can be used to restrict access to messages within a specific time window, ensuring that only recipients with valid, unexpired keys can decrypt them. Such temporal and contextual controls are difficult to achieve with traditional end-to-end encryption and are especially valuable in public safety scenarios. Key revocation otherwise can be a significant challenge for group communication. (3) Accountability through centralized key issuance: ABE relies on a centralized authority to issue decryption keys to recipients. While such centralization may raise privacy concerns in commercial applications, it is particularly well suited for these public safety applications. A trusted authority operated by a government agency or public safety organization can issue and manage keys, enabling authorized auditing, oversight, and post-mission analysis when required. These properties make ABE a natural fit for secure, flexible, and accountable group communication in public safety missions. However, standard ABE is not designed with namespace-based communication in mind. Mapping names to attributes and accommodating the dynamic evolution of the namespace therefore require additional care. Consequently, ABE must be adapted to fit this setting. In *NameShield*, we leverage the idea in [6] to integrate ABE with a graph-based namespace and provide confidentiality for communication among first responders.

In this paper, we propose *NameShield*, a system that provides secure and efficient group communication for public safety missions. *NameShield* integrates the role-based, dynamic graph namespace model from [4,5] with the ABE-based encryption approach from [6] to form a holistic framework for confidential, dynamic, and flexible group communication among first responders. *NameShield* uses the namespace to construct a role-based publish/subscribe framework that enables timely dissemination of relevant information, and it implements a per-message encryption mechanism based on ABE. Figure 1 illustrates the architecture of *NameShield*. The system is organized into three layers: the security layer, the information layer, and the network layer. The core of *NameShield* is the information layer. At this layer, *NameShield* maintains a role-based namespace and a set of rendezvous points (RPs). First responders subscribe to names in the namespace to receive corresponding messages, and senders publish messages by specifying names. The RPs manage the subscription trees for names and forward messages to the appropriate subscribers. Beneath the information layer lies the network layer, which is responsible for delivering messages to subscribers. *NameShield* is agnostic to the underlying network architecture and can operate over a variety of network technologies. Above the information layer is the security layer, which implements a message-oriented

KP-ABE scheme [6] to provide confidentiality for messages.

2 Background and related work

2.1 Namespace-based publish/subscribe system

Namespaces have been widely adopted in network applications and protocols, with domain name system (DNS) [7] serving the needs of society for many decades now. Information-centric networking (ICN) [8] also uses naming as the fundamental framework to build on top of. A namespace typically serves as an abstraction layer between resources and users, providing a flexible and human-friendly way to organize and access information. For example, in ICN, network content can be aggregated and categorized under names corresponding to specific topics. Users can request content by name without needing to know the location or identity of individual data objects. Several architectures [9–11] further exploit namespaces at the network or higher layer to enable efficient content distribution to users, improving scalability, mobility support, and robustness.

A name-based publish/subscribe model [12, 13] was originally developed in the context of ICN and has since been adopted in domains such as the Internet of Things (IoT) [14] and smart cities [15]. Unlike traditional publish/subscribe systems that rely on long-lived connections [16] or brokers [17], ICN-based namespace publish/subscribe systems address scalability and complexity challenges by leveraging in-network name-based routing and caching. CNS [5] further enhances this model by introducing a role-based namespace, which facilitates targeted information dissemination. POISE [4] extends CNS by relaxing its strict hierarchical structure and replacing it with a dynamic graph-based namespace, enabling more flexible and adaptive group formation. Figure 2 illustrates an example of a role-based dynamic graph namespace. Recipients subscribe to names in the namespace, and senders select the role names when publishing messages without needing to consider individual recipients.

2.2 ABE and its application

ABE [18] extends traditional public key encryption [19] to support fine-grained access control over encrypted data. ABE has been widely adopted in practical applications, including cloud computing [20] and healthcare systems [21]. Cloudflare has also developed and deployed an ABE-based system for managing the transport layer security (TLS) keys of customers in production environments [22].

ABE uses"attributes" and a Boolean policy—referred to as an "access structure"—to define which recipients are authorized to decrypt a ciphertext. An access structure is typically expressed as a Boolean formula over attributes using logical operators such as AND ($\wedge$) and OR ($\vee$), for example, $(attr_1 \vee attr_2) \wedge attr_3$ may be an access structure $\mathbb{A}$. A recipient can decrypt a ciphertext if and only if their set of attributes satisfies the access structure. For instance, the policy $(attr_1 \vee attr_2) \wedge attr_3$ permits a recipient with attributes $\{attr_1, attr_3\}$ to decrypt, but not a recipient with only $\{attr_1, attr_2\}$. Depending on where the access structure is placed, ABE can be classified into two categories: key-policy ABE (KP-ABE), in which the access structure is associated with the decryption key, and ciphertext-policy ABE (CP-ABE), in which the access structure is included in the ciphertext. ABE schemes are typically defined by four algorithms. Below, we present these four algorithms for KP-ABE; CP-ABE follows a similar structure, except that the access structure is placed in the ciphertext instead of the decryption key.

• $\mathsf{Setup}() \to (\mathsf{mpk}, \mathsf{msk})$ **:** This algorithm is executed by the key issuer during the initialization stage. It outputs a master public key mpk, which is published for encryption, and a master secret key msk, which is kept private and used to generate decryption keys.

• $\mathsf{KeyGen}(\mathsf{msk}, \mathbb{A}) \to \mathsf{sk}$ **:** This algorithm is executed by the key issuer to generate decryption keys. It takes as input the master secret key msk and an access structure $\mathbb{A}$, and outputs the corresponding decryption key sk.

• $\mathsf{Enc}(\mathsf{mpk}, S, \mathsf{msg}) \to \mathsf{ct}$ **:** This algorithm is executed by the sender to encrypt a message. It takes as input the master public key mpk, an attribute set S, and the message msg, and outputs the ciphertext ct.

• $\mathsf{Dec}(\mathsf{sk}_{\mathbb{A}}, \mathsf{ct}_S) \to \mathsf{msg}'$**:** This algorithm is executed by the recipient to decrypt a message. It takes as input the decryption key $\mathsf{sk}_{\mathbb{A}}$ and the ciphertext ct_S. If the attribute set S satisfies the access structure $\mathbb{A}$, decryption succeeds, and the plaintext msg' is recovered; otherwise, decryption fails.

Since the introduction of ABE [18], the research community has worked extensively to improve its efficiency and practicality. GPSW [23] improved the efficiency of KP-ABE but restricted attributes to be from a predefined universe. Subsequent work, such as ABGW [24], removed this restriction and supported arbitrary strings as attributes, but at the cost of higher decryption overhead. More recent constructions, including FABEO [25] and FAME [26], have significantly improved decryption performance, making ABE substantially more practical for real-world deployments.

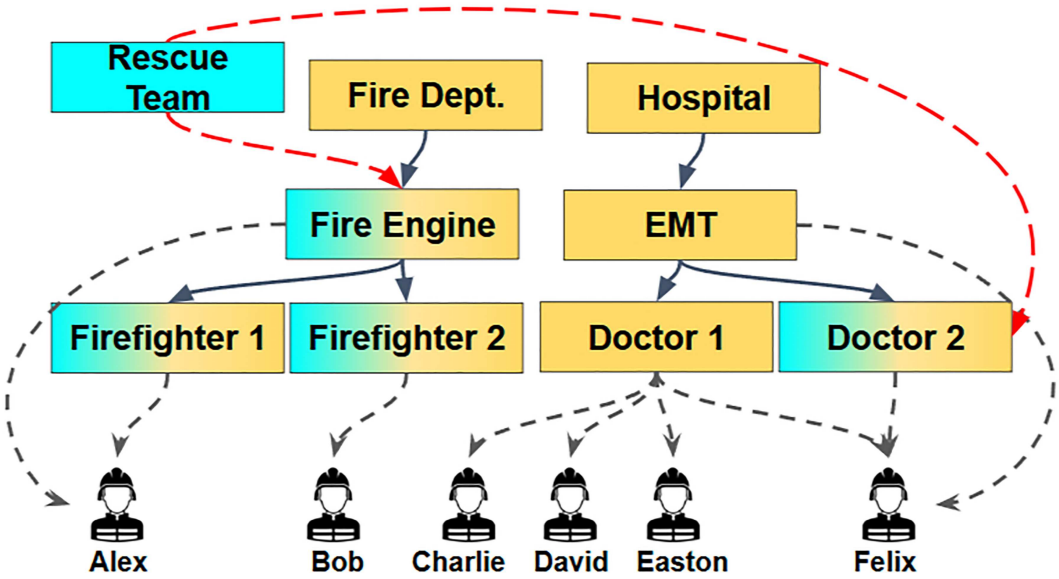

Figure 2 (Color online) Namespace for emergency group communication.

2.3 Secure group communication

Secure group messaging (SGM) [27] and continuous group key agreement (CGKA) [28], as realized in systems such as MLS [2, 27], have been proposed for commercial group messaging applications. These approaches rely on a tree-based key management protocol called TreeKEM [29–31]. TreeKEM requires all group members to maintain a shared binary tree that represents the group key state. Whenever the group membership changes (due to joins, leaves, or key updates), all members must synchronize their local view of the tree. This synchronization process depends on a reliable delivery service—typically provided by a centralized server—to ensure that control messages are delivered in order and without loss. Consequently, these protocols are not suitable for disrupted networks commonly encountered during public safety missions. ABE-based group communication enables per-message group formation and avoids the complex key management protocols required by SGM and CGKA. As a result, it does not depend on the reliable, in-order delivery of control messages and is therefore well suited for the disruption-prone and delay-prone networks encountered in public safety missions.

3 Name-based group messaging for emergency communications

NameShield leverages the role-based dynamic graph namespace from [4,5] to construct an efficient publish/subscribe framework. The namespace enables flexible and dynamic group creation: senders can define groups on a per-message basis, avoiding the complex and costly group formation procedures required by traditional group communication systems.

3.1 Publish/subscribe based on graph-based namespace

Figure 2 illustrates an example of a role-based dynamic graph namespace. The names in the namespace are divided into two categories: the administrative hierarchy (yellow) and the incident hierarchy (cyan). The administrative hierarchy reflects the organizational structure in the real world. First responders subscribe to one or more names corresponding to their roles prior to a public safety mission. When a sender publishes a message, it selects one or more names from the namespace, and the system automatically expands these names to include all their descendants in the hierarchy, thereby determining the set of recipients. For example, if the sender selects the name “Fire Engine”, the system expands it to include “Fire Engine”, “Firefighter 1”, and “Firefighter 2”, and the subscribers “Alex” and “Bob” receive the message. The incident hierarchy captures the dynamic nature of group formation during public safety operations. During a mission, administrators can create new names corresponding to emerging incidents and assign responders by linking existing role names as descendants of the new incident name. As shown in Figure 2, when a rescue operation arises, the administrator can create the name “Rescue Team” and assign “Fire Engine”, “Firefighter 1”, “Firefighter 2”, and “Doctor 2” as its descendants. When a sender selects “Rescue Team” as the

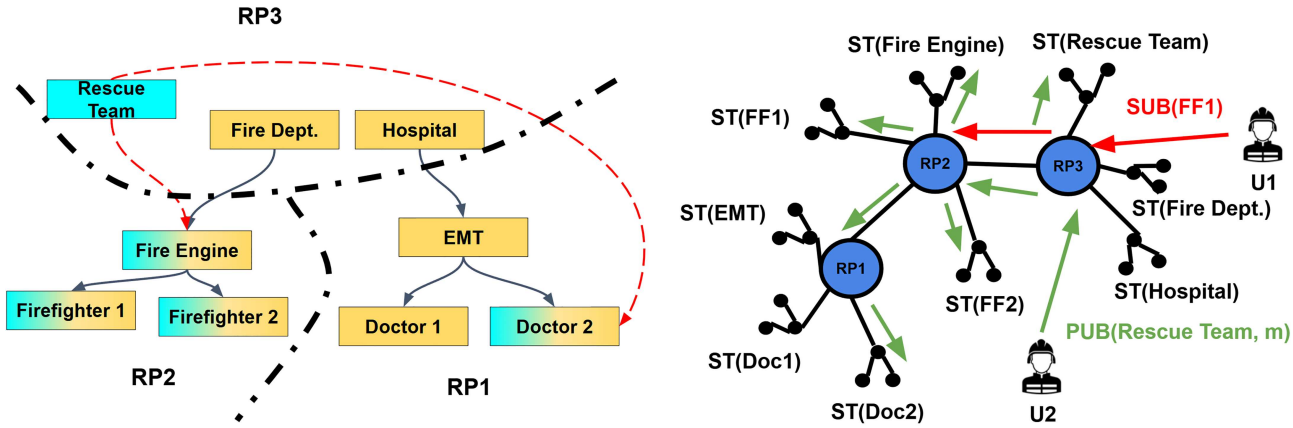

Figure 3 (Color online) An example for rendezvous points.

recipient, the system automatically expands this name and delivers the message to all subscribers of the descendant names, namely "Alex", "Bob", and "Felix".

Senders are not required to strictly follow the hierarchical auto-expansion rules of the namespace. Instead, they may flexibly include or exclude names as needed. To support this flexibility, a richer semantic language [6] can be used to describe name expressions. This language provides the operators `intersect` ($\cap$), `union` ($\cup$), `complement` / `not` ($\neg$), `brackets` ((), and `expand` ($\oplus$). The first four operators behave analogously to standard set operations. The `expand` operator ($\oplus$) expands a name to include all of its descendants in the namespace hierarchy.

We illustrate the expressiveness of this language using examples based on Figure 2. To send a message to the name "Fire Engine" and all of its descendants, the sender can use the expression $\oplus$(Fire Engine). This corresponds to the basic case where the sender selects a single role name. The language also enables selection across multiple hierarchies. To send a message to both "Fire Engine" and "EMT", as well as all their respective descendants, the sender can use the union expression $\oplus$(Fire Engine)$\cup\oplus$(EMT). To target only those recipients that belong to both "Rescue Team" and "Fire Engine", the sender can use the intersection expression $\oplus$(Rescue Team)$\cap\oplus$(Fire Engine). This is useful when selecting specific roles within a particular incident. Conversely, to select first responders under "EMT" who are not part of "Rescue Team", the sender can use the expression $\oplus$(EMT)$\cap\neg\oplus$(Rescue Team), which is helpful for identifying available resources not participating in a given incident.

3.2 Namespace management with rendezvous points

NameShield leverages the design in [4] to use rendezvous points (RPs) to manage name subscriptions and deliver messages to recipients. RPs provide a distributed mechanism for maintaining the state related to the namespace within the network and efficient forwarding of messages. As a result, not every node in the network has to maintain application layer subscription state. We can thus achieve a multiple-sender-multiple-receiver publish/subscribe system at the application layer with the assistance of RPs in the network. In disaster scenarios, such a design is particularly helpful as network infrastructure may be disrupted. Relying on a single centralized application server for managing the pub/sub system would be undesirable. A single server can become both a single point of failure and a bottleneck. Thus, an RP-based dissemination system with help at the network layer can improve resilience, scalability, and robustness under network disruptions.

In *NameShield*, each name may have multiple subscribers. The set of subscribers associated with a name can therefore be viewed as a multicast group and managed using a multicast tree. Each RP acts as the core for multicast trees corresponding to the names it hosts. In addition to maintaining subscription trees, RPs also participate in managing the namespace. Each name is mapped to a responsible RP via a name-to-RP resolution service, which enables RPs to forward messages to one another as needed.

Figure 3 illustrates an example deployment with three RPs: RP1, RP2, and RP3. Each RP manages a subgraph of the namespace and maintains the subscription trees (STs) associated with the names it hosts. For example, RP1 acts as the core for the subscription trees of "EMT", "Doctor 1", and "Doctor 2". The name-to-RP mapping service associates each name with its hosting RP; for instance, the name "Firefighter 1" is mapped to RP2. The path followed by a subscription is shown in red. Suppose user U1 wishes to subscribe to "Firefighter 1". U1 issues a subscription request SUB (Firefighter 1) to its first-hop RP, RP3, without knowing which RP hosts that name. RP3 resolves the name-to-RP mapping and forwards the request as a unicast message to RP2, thereby joining

the subscription tree ST (Firefighter 1). The publication path is shown in green. Suppose user U2 publishes a message m to all subscribers of "Rescue Team", which implicitly includes subscribers of all the descendant names. U2's first-hop RP, RP1, performs a name expansion on the name "Rescue Team" and multicasts m downstream along ST (Rescue Team). RP1 also detects that some descendants of "Rescue Team" are hosted by other RPs, and therefore unicasts m to those RPs. Each of those RPs then multicasts m along their local subscription trees. As a result, both subscriptions and publications require users to send only a single message addressed to a single name. The network automatically resolves, expands, and disseminates messages to the appropriate recipients.

3.3 Network layer architecture agnostic message delivery

The network layer has to deliver messages to the actual recipients on the basis of the subscription state and the corresponding multicast trees maintained by the RPs. Any underlying network layer (whether unicast, multicast, or even broadcast) or application-layer multicast can be used to implement the mechanisms developed in this section. We outline several possible implementations below.

• **Internet protocol (IP) networks.** A natural implementation of the network layer is based on IP networking. IP supports both unicast and multicast, and RPs can be deployed with routers that use protocol independent multicast-sparse mode (PIM-SM) (IP multicast) [32] to disseminate messages along subscription trees.

• **Information-centric networking (ICN).** Since *NameShield* is built around name-based publish/subscribe semantics, ICN and its variants (e.g., NDN [8] and MobilityFirst [11]) are a natural fit. In this setting, RPs can be co-located with ICN routers and messages can be forwarded using name-based routing. Example implementations are described in [4, 12].

• **Device-to-device (D2D) networking.** *NameShield* can also be implemented over a D2D network, which is better suited for a disruption-prone network. In this case, selected devices act as RPs and disseminate messages using multicast. An example of such an approach is presented in [3].

• **Server-based unicast.** *NameShield* can also be realized using pure unicast communication. A central server hosts all RPs and delivers messages individually to each subscriber. While this approach introduces a single point of failure and is therefore not recommended for public safety missions, it may be acceptable in other deployment scenarios, as discussed in [4].

3.4 Notification latency for message delivery

We believe an RP-based design, which does not require every node in the network to maintain application-layer subscription state, may achieve better performance than an approach (e.g., ICN-based systems) that maintains subscription state at every router. To quantify this advantage, we used the POISE simulator to compare POISE [4] with CNS [5], a name-based publish/subscribe system built on top of NDN [9]. The experiments use a workload consisting of more than 1000 names under two publication regimes: (1) moderate load, with an average arrival rate between 1500 and 2000 pkt/s, and (2) intense load, with an arrival rate between 1500 and 3500 pkt/s. Performance is evaluated based on notification latency, which is defined as the time required to deliver a publication to all intended recipients. Figure 4 presents the cumulative distribution function (CDF) of the notification latency. For 90% of message deliveries, POISE achieves a latency below 0.03 s, whereas CNS requires more than 300 s for 90% of deliveries. This is because POISE dynamically partitions the graph and balances the load arriving at names across different RPs, thus reacting dynamically to the traffic being sent to the different names in the namespace. Thus, it significantly improves overall performance (especially latency) in the delivery of publications. Not only does it achieve lower notification latency, but because it avoids congestion at the RPs, queueing delays at the RPs are minimized, resulting in lower variability in the notification latency. In contrast, CNS uses a hierarchical namespace that is more or less static. The message transmission load is not dynamically balanced between RPs, which can create congestion and result in much higher notification latency.

3.5 Resilience to network disruptions

Infrastructure damage during public safety missions can severely impact network connectivity, resulting in the network being partitioned, resulting in regions without global connectivity. The design of *NameShield* helps mitigate the effects of such scenarios. We consider two types of partitioning. (1) Subscribers not being able to reach their RPs. In this case, subscribers are temporarily disconnected from the RPs that manage their subscriptions. (2) Some RPs become isolated from other RPs. Here, subsets of the namespace and their associated subscription trees are partitioned from the rest of the system.

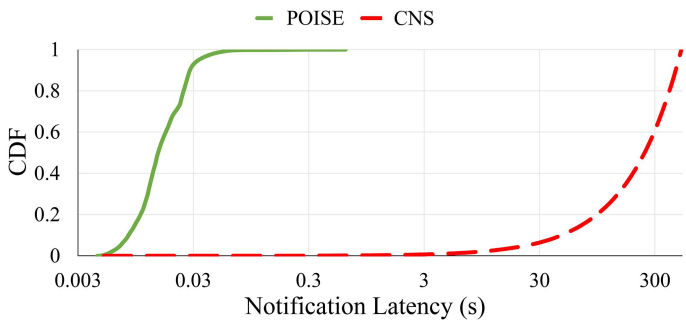

Figure 4 (Color online) POISE vs. CNS (notification latency).

Because each RP manages an independent subset of the namespace and its corresponding subscription trees, failures or partitions in one part of the system do not result in a global failure. Other parts of the system can continue to operate. To further support communication under such conditions, techniques similar to those proposed in [33] can be employed. Subscribers can cache messages for the names to which they are subscribed, and opportunistically exchange them with other subscribers when they encounter each other. In addition, data mules—for example, vehicles or mobile nodes that physically carry stored data between disconnected regions—can be used. A data mule may subscribe to all names in the namespace and disseminate stored messages to the appropriate RPs and subscribers when it enters the partitioned area. As reported in [33], simulations of two long-term partitioned shelters-each containing five first responders and generating three updates-demonstrate the effectiveness of this approach. Without a data mule, responders within each shelter can only reach *local consensus*, taking approximately three time units. After a data mule is introduced to exchange updates between shelters, *global consensus* can be reached in about two time units. The results in [33] show that the approach can efficiently handle network partitions during disaster scenarios.

4 Secure message transmission with ABE

We leverage the approach proposed in [6] to construct an encryption mechanism within a graph-based namespace framework. In [6], a message-oriented abstraction in the encryption layer is built on top of the graph-based namespace to provide message confidentiality. This abstraction layer can be instantiated using either KP-ABE or CP-ABE. We describe the details of our chosen construction in this section.

4.1 Threat model

In this work, we do not consider adversaries that take explicit actions to disrupt the system. Specifically, we assume that attackers cannot block message transmission, modify the messages in transit, or overwhelm the system (e.g., by launching a DDoS attack).

Our target adversaries are curious, those that are capable of intercepting network traffic within our system (e.g., by sniffing traffic or gaining access to network links, routers, or switches) and attempting to extract message content or infer the identity of the recipient. Our goal is to preserve the confidentiality of the message and, potentially, the anonymity of the recipients.

Another class of adversaries consists of entities that possess valid secret keys but attempt to decrypt messages not intended for them. For example, such attackers may originate within an emergency department (e.g., internal adversary). Our objective is to prevent these adversaries from decrypting unauthorized messages, even if they collude by combining their secret keys intended to be used to decrypt messages for which they are the intended recipients.

We have not addressed the support for the authentication of users in this work. We also do not explicitly address the anonymity of the sender here. We acknowledge that a complete group communication solution should address these properties. Existing authentication mechanisms (e.g., ECDSA [34]) can be readily integrated into our framework; however, we have deferred this integration to future work. We also believe that existing approaches for sender anonymity (including onion routing [35]) can be adopted to preserve sender anonymity.

4.2 Message-oriented solution

The message-oriented abstraction in the encryption layer (hereafter referred to as the *message-oriented solution*) serves as an interface for implementing ABE-based encryption over the graph-based namespace. This approach

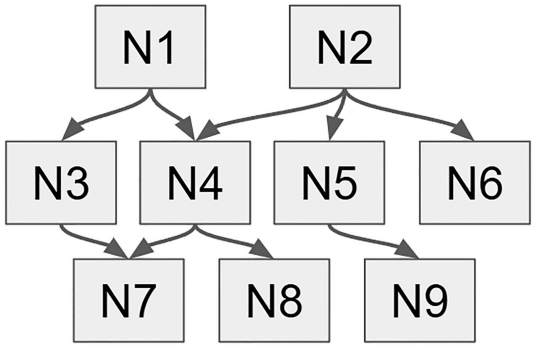

Figure 5 Example namespace.

Table 1 Message-oriented solution for example namespace.

Target name N	Name contains	
	Send to N (names in cipher)	Subscribe to N (names in key)
$N1$	$N1, N3, N4, N7, N8$	$N1$
$N2$	$N4, N5, N6, N7, N8, N9$	$N2$
$N3$	$N3, N7$	$N3$
$N4$	$N4, N7, N8$	$N4$
$N5$	$N5, N9$	$N5$
$N6$	$N6$	$N6$
$N7$	$N7$	$N7$
$N8$	$N8$	$N8$
$N9$	$N9$	$N9$

leverages the highly dynamic group semantics provided by the namespace and allows names to change without requiring recipients to obtain new decryption keys. The core idea is that the ciphertext contains the set of target names, while each recipient holds a decryption key associated with the name(s) to which it is subscribed. Decryption succeeds if and only if the recipient's name matches one of the names included in the ciphertext. Figure 5 and Table 1 illustrate this process. When a sender selects the name $N1$, the system automatically expands it to include all its descendants, resulting in the name set $\{N1, N3, N4, N7, N8\}$ being included in the ciphertext. A recipient subscribed to $N7$ holds a decryption key associated with $N7$ and is therefore able to decrypt the message sent to the larger group defined by $N1$.

The message-oriented solution is particularly well suited for dynamic namespace updates. Because the decryption key does not encode any hierarchical structure, recipients do not need to update their keys when the namespace changes (e.g., when a responder is reassigned to a different incident or the administrative hierarchy is modified). Continuing with the example in Figure 5, suppose that $N7$ is no longer a descendant of $N1$ and is reassigned under $N5$. The sender can simply exclude $N7$ when encrypting messages targeted to $N1$, and include $N7$ when encrypting messages targeted to $N5$. The subscriber of $N7$ does not need to update its decryption key in either case.

4.3 Implementing message-oriented solution with ABE

The message-oriented solution can be instantiated using either KP-ABE or CP-ABE. To realize the idea that a message can be decrypted if and only if the ciphertext and the decryption key share at least one common name, we use an access structure consisting solely of logical "`OR`" operators. In the following section, we describe how to implement the message-oriented solution with an all-`OR` access structure in both KP-ABE and CP-ABE.

4.3.1 *Message-oriented NameShield with KP-ABE*

In KP-ABE, the access structure is embedded in the decryption key. For each recipient, the decryption key contains an access structure formed by connecting all names to which the recipient subscribes using the operator `OR`. When encrypting a message, the sender constructs an attribute set that contains the selected target name and all of its descendants. As illustrated in Figure 6(a), a subscriber such as Alex, who subscribes to the names "Fire Engine" and "Firefighter 1", is issued a decryption key with the access structure (Fire Engine$\vee$Firefighter 1). Another subscriber, Bob, who subscribes only to "Firefighter 2", receives a decryption key with the access structure consisting of a single

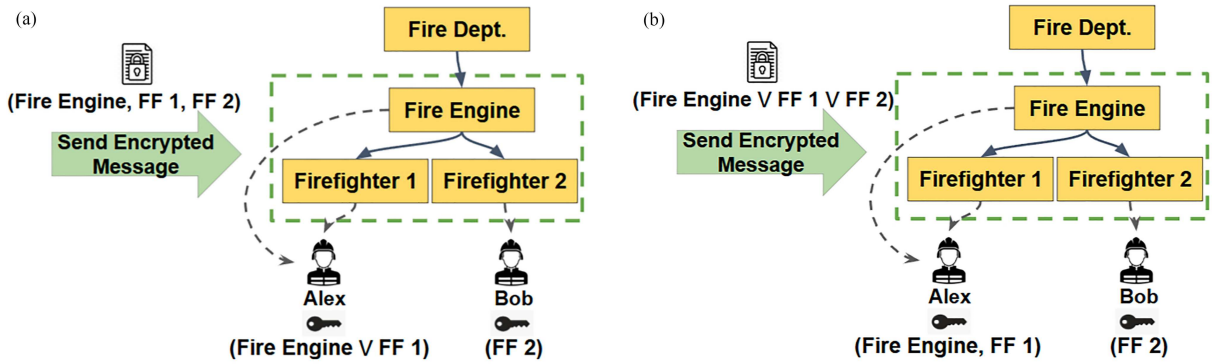

Figure 6 (Color online) Message-oriented solution w/ABE. (a) KP-ABE; (b) CP-ABE.

attribute, namely "Firefighter 2". In this case, the decryption key can decrypt any ciphertext whose attribute set contains "Firefighter 2", regardless of any additional attributes. When a sender targets the name "Fire Engine" and all of its descendants, it constructs the attribute set {Fire Engine, Firefighter 1, Firefighter 2}. As a result, the attributes in the message satisfy the access structure at Alex and Bob; thus, they are able to decrypt the message.

4.3.2 *Message-oriented NameShield with CP-ABE*

In a manner similar to the KP-ABE construction, the message-oriented solution can also be implemented using CP-ABE with an access structure composed solely of logical `OR` operators. In CP-ABE, however, the access structure is contained in the ciphertext rather than in the decryption key. During encryption, the sender constructs an access structure consisting of the target name and all of its descendant names, connected by `OR`. As illustrated in Figure 6(b), the decryption keys issued to recipients contain attribute sets corresponding to the names to which they subscribe. For example, Alex is issued a key with the attribute set {Fire Engine, Firefighter 1}, while Bob is issued a key containing only the attribute "Firefighter 2". Here, "Firefighter 2" represents an attribute in the recipient's attribute set, rather than an access structure. When the sender targets the name "Fire Engine" and all of its descendants, it constructs the access structure (Fire Engine $\vee$ Firefighter 1 $\vee$ Firefighter 2). As a result, both Alex and Bob satisfy the access structure and are able to decrypt the message.

4.3.3 *Comparing KP-ABE with CP-ABE*

Both the KP-ABE and CP-ABE implementations of the message-oriented solution provide confidentiality for messages during transmission; the primary difference between them lies in performance. In recent ABE constructions [25,26], KP-ABE and CP-ABE exhibit comparable decryption performance. The main performance difference arises in encryption and key generation, specifically in the cost of constructing access structures. Constructing an access structure—typically implemented as a tree [23] or matrix [25] in practice—incurs slightly higher overhead than constructing a simple attribute set. In *NameShield*, decryption keys are typically generated prior to the public safety mission, whereas encryption occurs during the mission when latency and computational overhead are more critical. Therefore, placing the cost of access-structure construction in the key-generation phase, as in KP-ABE, reduces the online overhead during mission operation. For this reason, we adopt KP-ABE as the underlying primitive for *NameShield* and use it in the remainder of the paper.

We quantitatively validate our choice of KP-ABE as the final design by comparing the performance of KP-ABE and CP-ABE when instantiated within the message-oriented solution. We use the FABEO construction [25], implemented on top of the Charm framework [36], as the underlying ABE scheme and evaluate the performance of its four component algorithms (Setup, KeyGen, Enc, Dec) in our setting. For the Setup algorithm, both KP-ABE and CP-ABE complete in approximately 16 ms. This is expected, since both schemes perform similar cryptographic initialization and no attributes or access structures are involved at this stage. Figure 7 shows the execution time of the KeyGen algorithm. As the number of subscribed names increases, the key generation time increases for both schemes. KP-ABE incurs a higher cost than CP-ABE because it constructs the access structure during key generation. However, since KeyGen is typically executed prior to mission deployment, this overhead does not affect operational performance during public safety missions. To fairly compare Enc encryption and Dec decryption performance, we consider a scenario in which each recipient subscribes to a single name, and the sender increases the number of recipients per message. The results are shown in Figure 8. As expected, CP-ABE requires more time for encryption than KP-ABE because CP-ABE constructs the access structure at encryption time, whereas

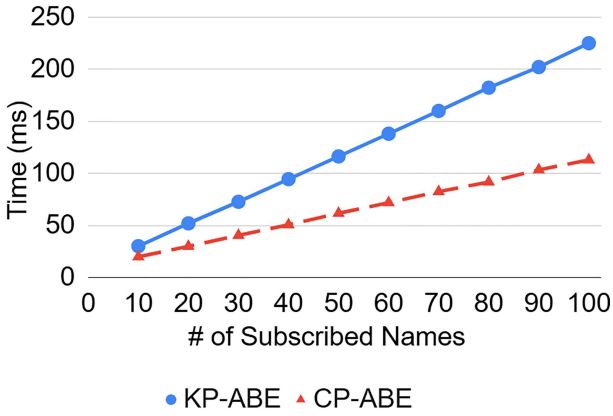

Figure 7 (Color online) Key generation time.

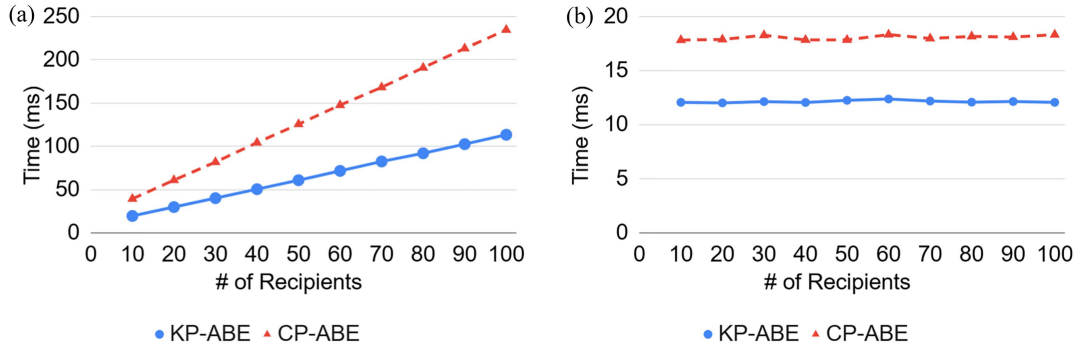

Figure 8 (Color online) Performance for message-oriented solution w/ABE. (a) Encryption performance; (b) decryption performance.

KP-ABE does not. Both schemes exhibit similar decryption performance, although CP-ABE incurs slightly higher overhead due to the need to process a larger access structure included in the ciphertext. These results confirm our design choice of using KP-ABE for implementing the message-oriented solution in *NameShield*.

4.4 Fine-grained access control

One advantage of ABE is its support for fine-grained access control through attributes. Following the approach in [6], we use a timestamp attribute to limit the validity period of decryption keys. This mechanism is particularly valuable in public safety missions, as it provides both perfect forward secrecy [37,38] and backward secrecy [39]. If a recipient's decryption key is compromised, an adversary can decrypt messages only within the key's valid time window, but not messages sent before or after that period.

The core idea of this approach is to divide time into fixed-length intervals and use the start time of each interval as a timestamp attribute. In KP-ABE, when issuing a decryption key to a subscriber, the access structure includes not only the subscribed names but also the corresponding timestamp attribute, connected by an operator `AND`. During encryption, the sender includes the timestamp attribute of the current time interval in the attribute set. As a result, only keys that contain the correct timestamp and are valid for the corresponding interval can decrypt the message. As illustrated in Figure 9 for this example, time is divided into one-hour intervals. Alex, who subscribes to "Fire Engine" and "Firefighter 1", is issued a decryption key valid for the interval from "2025-12-31 23:00" until "2026-01-01 00:00". The access structure in Alex's key is (Fire Engine $\vee$ Firefighter 1) $\wedge$ 2025-12-31 23:00. For *Message 1*, which is sent during this interval and targets "Firefighter 1", the sender encrypts using the attribute set {Firefighter 1, 2025-12-31 23:00}. For *Message 2*, which targets the same role but is sent in the next interval, the attribute set changes to {Firefighter 1, 2026-01-01 00:00}. Consequently, Alex's key can decrypt *Message 1* but not *Message 2*.

4.5 Key issuer and key distribution

ABE relies on a centralized key issuer to generate decryption keys for recipients. In *NameShield*, the key issuer is operated by a trusted higher authority and deployed in a secure environment. When distributing decryption keys,

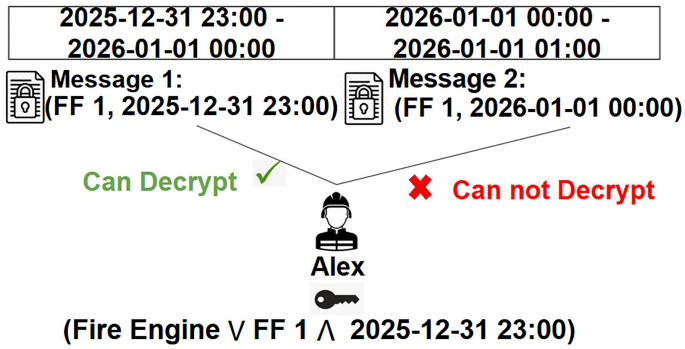

Figure 9 (Color online) Fine-grained access w/timestamp attribute.

the key issuer needs to establish a secure channel (e.g., TLS [40]) with the recipient.

In *NameShield*, decryption keys are distributed to recipients prior to public safety missions whenever possible, based on their assigned roles, in order to avoid reliance on network connectivity during mission execution. However, several special cases must be considered.

Key update upon expiration. During a mission, a decryption key may expire. Obtaining an updated key from the key issuer can be difficult if network connectivity is disrupted. To mitigate this issue, recipients can pre-request decryption keys for future time intervals in advance. This is feasible because *NameShield* uses a timestamp attribute to control key validity, and future timestamps can be determined ahead of time.

Key regeneration for auditing. For auditing purposes, authorized higher authorities may require access to historical messages. Instead of requesting old decryption keys from recipients—who may have deleted them—the key issuer can regenerate the corresponding keys directly. This is possible because ABE attributes are represented as text strings, and the key issuer retains the master secret needed to recreate any valid decryption key.

4.6 Dealing with network disruptions

Similar to the information layer, the security layer of *NameShield* is also designed to tolerate network disruptions and delays. In *NameShield*, a recipient's decryption key includes only the name(s) to which the recipient is subscribed, while the sender includes the full set of target names in the ciphertext during encryption. When the namespace changes, the sender updates the names included in the ciphertext, rather than requiring recipients to obtain new decryption keys. This design is particularly advantageous when recipients are located in regions that are temporarily disconnected from the key issuer. Even under such conditions, recipients can continue to decrypt relevant messages using their existing keys when the namespace is updated, thereby maintaining communication despite the network disruption.

The attribute-based design for confidentiality also helps mitigate the impact of network disruptions. As discussed earlier, recipients can pre-request decryption keys for future time intervals from the key issuer. This capability is particularly useful when a recipient anticipates entering a partitioned or disconnected area (e.g., a shelter). In such a case, the recipient can obtain the necessary decryption keys in advance and continue to access authorized messages while being disconnected from the key issuer.

4.7 Comparing with commercial group communication schemes

Secure group messaging protocols are designed for commercial group communication, where group membership is typically far less dynamic than in public safety missions. These protocols rely on relatively complex group formation and maintenance procedures, but use lightweight cryptographic operations such as public-key encryption for providing message confidentiality. A natural question is whether the use of lightweight encryption can compensate for the overhead of group formation in highly dynamic environments. To evaluate this question quantitatively, we compare the performance of MLS, a representative secure group messaging protocol, with *NameShield*. We use the metrics of group formation overhead and encryption/decryption costs.

We compare the two approaches in terms of group formation time and encryption time as a function of the group size. To simulate a highly dynamic environment, we assume that a group formation process is required before message transmission. For MLS, we let the first group member initiate the group by sending control messages to all other members to establish the group state. For our experiment, we have all group members hosted on the same machine. After group formation, the sender encrypts a message addressed to the entire group. We record the total time for group formation and message encryption. Since MLS requires sender authentication, the

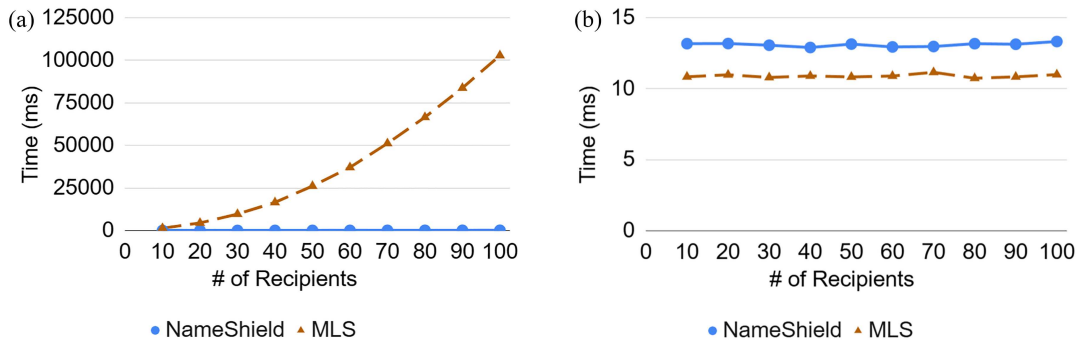

Figure 10 (Color online) Performance comparison between MLS and *NameShield*. (a) Group formation & encryption performance; (b) decryption performance.

measured time also includes the cost of generating a digital signature. For *NameShield*, no explicit group formation is required, so we measure only the message encryption time. To ensure a fair comparison, we also include the cost of generating an ECDSA [34] signature at the sender. For decryption, we measure the time required for a single group member to decrypt a message, including the cost of signature verification. We use the open-source implementation OpenMLS [41] for MLS. For *NameShield*, we use the same ABE-based implementation as in the previous experiment, namely FABEO [25] implemented on top of the Charm framework [36]. We note that this experimental setup is favorable to MLS, since hosting all group members on the same machine eliminates network transmission delays for control messages during group formation.

Figure 10(a) presents the performance of group formation and encryption. Even without network transmission delays, MLS incurs substantially higher overhead for group formation and encryption than *NameShield*, which does not require an explicit group formation process. For a group of 50 members, MLS is more than 400× slower than *NameShield*. We note that this behavior is not due to system overload, as our monitoring shows that the average system load remained well below the CPU's capacity throughout the experiment. Figure 10(b) shows the decryption performance. Both *NameShield* and MLS exhibit similar decryption costs, with *NameShield* being slightly higher (approximately 5 ms). These results indicate that the lightweight cryptographic primitives used by MLS cannot compensate for the overhead of group formation in highly dynamic environments.

4.8 Overall solution evaluation

In this section, we present an overall evaluation that combines the name-based group messaging framework with the proposed message-level encryption mechanism. We use the network simulator [42] provided in POISE to emulate a realistic network environment consisting of multiple forwarding routers and end hosts acting as publishers and subscribers. We adopt parameters similar to those used in POISE. We additionally incorporate both the encryption and decryption operations for each message published.

Specifically, we use the Rocketfuel 1221 Telstra [43] network map as the underlying topology and a graph-based "Disaster Management" category namespace derived from the Wikipedia [44] dataset as the naming structure. This namespace contains 1468 hierarchical names spanning six levels. We generate 95417 publications from the associated Wikipedia pages and files, and randomize the publication order to generate the workload on the network. We use our message-oriented solution to encrypt the publication with ABE.

We measure notification latency, which includes the time required to deliver a publication, as well as the encryption and decryption operations associated with each publication. As shown in Figure 11, adding encryption and decryption increases the notification latency for the publication. Compared to POISE without confidentiality, where 90% of publication messages have notification latency below 0.02 s, the message-oriented KP-ABE solution achieves notification latency below 0.05 s for 90% of the messages, while the CP-ABE solution takes 0.08 s for 90% of the messages. This observation is consistent with our conclusion in the previous section that the message-oriented CP-ABE solution has higher overhead compared to KP-ABE. Overall, the additional latency caused by the encryption and decryption operations, while significant compared to just the network delays, is still acceptable for delivering messages in this application of delivering messages to first responders. More importantly, compared to the simple delivery of unencrypted publications within the hierarchical (CNS) naming framework (which takes more than 3 s for almost all the messages (see Figure 4)), the addition of confidentiality within POISE maintains even the tail (90^{th} percentile) latency to be quite manageable, below 0.05 s.

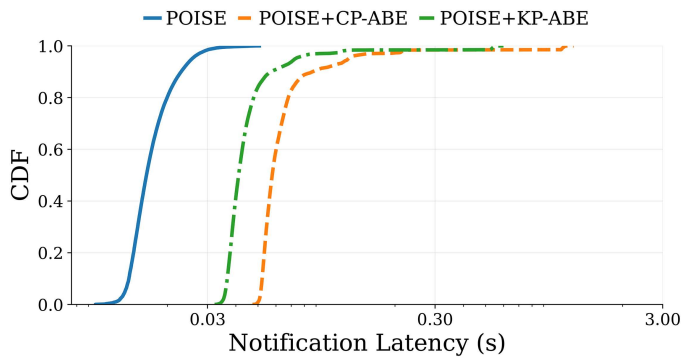

Figure 11 (Color online) Overall solution evaluation.

5 Recipient anonymization

Beyond message confidentiality, recipient anonymization is another important security concern. In the message-oriented KP-ABE solution, the attribute used for encryption corresponds to the recipient's name. As a result, the intended recipient set may be exposed during message transmission. This raises the need to protect the anonymity of the recipients, since an adversary monitoring the network could infer the recipients of one or more messages. In public safety missions, leakage of recipient information can be particularly dangerous. By observing which roles are receiving messages, an adversary may be able to infer operational intent or predict upcoming actions. For example, during a terrorist incident, a surge in messages addressed to a SWAT team could indicate that a tactical operation is imminent.

A naive approach to achieving recipient anonymity is to use an anonymous ABE scheme that hides attributes within the ciphertext. However, existing anonymous ABE constructions do not offer a practical solution in our setting. In general, anonymous ABE schemes can be categorized as fully-hidden [45–48]or partially-hidden [49, 50]. The fully-hidden ABE scheme attempts to conceal all attributes in the ciphertext. However, this approach typically incurs significant performance overhead. In standard ABE, attributes included in the ciphertext participate directly in the decryption process. When all attributes are hidden, a receiver may need to try and decrypt multiple times to determine whether a ciphertext matches its access policy. Some approaches [46, 48] mitigate this overhead by encoding attributes into specialized data structures, but these constructions lack formal security proofs, raising concerns about their robustness. Partially-hidden ABE seeks to reduce overhead by revealing limited structural information about attributes. In such schemes, each attribute is decomposed into a type and a value (for example, "Name: Alice"). During encryption, each type is restricted to a single value, ensuring that a receiver needs to only attempt decryption once. However, this model does not apply to our setting: all the role-based namespace identifiers share the same type, making partially-hidden ABE unsuitable for expressing role-based recipient sets.

An additional challenge arises when recipients are anonymous. Since recipient identities are not explicitly revealed, a receiver cannot determine whether it is an intended recipient without attempting decryption and observing whether it succeeds. This design is intentional, as messages may be delivered to non-targeted nodes (receivers at the network layer) to prevent identity leakage through traffic analysis. However, this means that non-targeted receivers must also attempt decryption. Because ABE decryption is computationally expensive, this behavior can introduce significant overhead, especially in scenarios with high message volumes. Therefore, a mechanism is needed that allows non-targeted receivers to efficiently determine that a message is not intended for them without performing full ABE decryption.

These requirements motivate the need for enhancements to our ABE-based design so that we simultaneously preserve confidentiality along with recipient anonymity, while enabling efficient rejection (avoiding decryption of received messages) by non-recipients. Here, we give a simple approach which is the topic of our ongoing work. We expect that it would be desirable to use a lightweight encryption primitive (e.g., symmetric-key encryption) to protect the attributes that appear in the clear, which are included in the ABE ciphertext. The corresponding symmetric keys are published via the namespace and distributed only to authorized first responders. In this design, non-targeted recipients can attempt a fast symmetric decryption to determine whether a message is intended for them, and reject irrelevant messages (that they are not the intended recipients) without incurring the cost of a full ABE decryption. Adversaries, who do not possess the appropriate keys, cannot recover the clear attributes and therefore gain no information about the intended recipients.

6 Conclusion

In this paper, we propose *NameShield*, a secure and efficient holistic framework for dynamic group communication among first responders during public safety missions. *NameShield* is composed of three layers: a security layer, an information layer, and a network layer. At the information layer, we leverage a role-based graph namespace proposed in [4] to construct an efficient publish/subscribe system for group communication. On top of this, we build a message-oriented KP-ABE mechanism based on the approach in [6] to provide confidentiality for messages during transmission. This design supports per-message confidentiality for highly dynamic groups and remains effective in disruption- and delay-prone network environments. The network layer of *NameShield* is agnostic to the underlying networking technology. As long as the network supports some form of multicast (at the application layer or network layer), *NameShield* can be deployed on top of it. Overall, *NameShield* provides a secure, flexible, and efficient communication framework tailored to the unique requirements of public safety missions.

References

1 Kevin Fall. Delay-tolerant networking architecture (RFC 4838). https://www.rfc-editor.org/info/rfc4838/, 2007
2 Alwen J, Coretti S, Dodis Y, et al. Modular design of secure group messaging protocols and the security of MLS. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, 2021. 1463–1483
3 Ramakrishnan K K, Yuksel M, Seferoglu H, et al. Resilient communication for dynamic first responder teams in disaster management. IEEE Commun Mag, 2022, 60: 93–99
4 Jahanian M, Chen J, Ramakrishnan K K. Graph-based namespaces and load sharing for efficient information dissemination in disasters. In: Proceedings of the 2019 IEEE 27th International Conference on Network Protocols (ICNP), 2019
5 Chen J C, Arumaithurai M, Fu X M, et al. CNS: content-oriented notification service for managing disasters. In: Proceedings of the 3rd ACM Conference on Information-Centric Networking, 2016
6 Yu H M, Chen J C, Ramakrishnan K K. SAFE: secure and flexible encryption for dynamic team communications in disaster management. In: Proceedings of the 2023 IEEE 31st International Conference on Network Protocols (ICNP), 2023
7 Mockapetris P. Domain names—concepts and facilities (RFC 1034). https://www.rfc-editor.org/info/rfc1034/, 1987
8 Jacobson V, Smetters D K, Thornton J D, et al. Networking named content. In: Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies (CoNEXT), 2009. 1–12
9 Zhang L, Afanasyev A, Burke J, et al. Named data networking. SIGCOMM Comput Commun Rev, 2014, 44: 66–73
10 Mukherjee S, Bronzino F, Srinivasan S, et al. Achieving scalable push multicast services using global name resolution. In: Proceedings of the 2016 IEEE Global Communications Conference (GLOBECOM), 2016. 1–6
11 Raychaudhuri D, Nagaraja K, Venkataramani A. MobilityFirst: a robust and trustworthy mobility-centric architecture for the future internet. SIGMOBILE Mob Comput Commun Rev, 2012, 16: 2–13
12 Chen J C, Arumaithurai M, Jiao L, et al. Copss: an efficient content oriented publish/subscribe system. In: Proceedings of the 2011 ACM/IEEE Seventh Symposium on Architectures for Networking and Communications Systems, 2011. 99–110
13 Fotiou N, Nikander P, Trossen D, et al. Developing information networking further: from psirp to pursuit. In: Proceedings of International Conference on Broadband Communications, Networks, and Systems, 2012. 1–13
14 Gündoğan C, Kietzmann P, Schmidt T C, et al. Hopp: robust and resilient publish-subscribe for an information-centric internet of things. In: Proceedings of the 2018 IEEE 43rd Conference on Local Computer Networks (LCN), 2018. 331–334
15 Shariat A, Tizghadam A, Leon-Garcia A. An ICN-based publish-subscribe platform to deliver UAV service in smart cities. In: Proceedings of 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2016. 698–703
16 Segall B, Arnold D, Boot J, et al. Content based routing with Elvin4. In: Proceedings of AUUG2K, 2000
17 Diao Y L, Rizvi S, Franklin M J. Towards an internet-scale XML dissemination service. In: Proceedings of the Thirtieth International Conference on Very Large Data Bases, 2004. 612–623
18 Sahai A, Waters B. Fuzzy identity-based encryption. In: Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2005. 457–473
19 Rivest R L, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM, 1978, 21: 120–126
20 Ge C, Susilo W, Baek J, et al. Revocable attribute-based encryption with data integrity in clouds. IEEE Trans Dependable Secure Comput, 2022, 19: 2864–2872
21 Li H, Yu K, Liu B, et al. An efficient ciphertext-policy weighted attribute-based encryption for the internet of health things. IEEE J Biomed Health Inform, 2022, 26: 1949–1960
22 Ladd W, Verma T, Venema M, et al. Portunus: re-imagining access control in distributed systems. In: Proceedings of the 2023 USENIX Annual Technical Conference, 2023. 35–52
23 Goyal V, Pandey O, Sahai A, et al. Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM conference on Computer and communications security, 2006. 89–98
24 Ambrona M, Barthe G, Gay R, et al. Attribute-based encryption in the generic group model: automated proofs and new constructions. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017. 647–664
25 Doreen R, Wee H. FABEO: fast attribute-based encryption with optimal security. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, 2022
26 Agrawal S, Chase M. FAME: fast attribute-based message encryption. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017. 665–682
27 Barnes R, Beurdouche B, Robert R, et al. The messaging layer security (MLS) protocol (RFC 9420). https://www.rfc-editor.org/rfc/rfc9420.html, 2023
28 Joël Alwen J, Coretti S, Jost D, et al. Continuous group key agreement with active security. In: Proceedings of Theory of Cryptography: 18th International Conference, 2020. 261–290
29 Barnes R. Subject: remove without double-join (in TreeKEM). https://mailarchive.ietf.org/arch/msg/mls/Zzw2tqZC1FCbVZA9LKERsMIQXik, 2018
30 Rescorla E. Subject: TreeKEM: an alternative to ART. MLS mailing list. https://mailarchive.ietf.org/arch/msg/mls/WRdXVr8iUwibaQu0tH6sDnqU1no, 2018
31 Bhargavan K, Barnes R, Rescorla E. TreeKEM: asynchronous decentralized key management for large dynamic groups a protocol proposal for messaging layer security (MLS). Dissertation for Doctoral Degree. Paris: Inria Paris, 2018
32 Fenner B, Handley M, Holbrook H, et al. Protocol independent multicast-sparse mode (PIM-SM): protocol specification (revised) (RFC 7761). https://www.rfc-editor.org/info/rfc7761/, 2016
33 Jahanian M, Ramakrishnan K K. CoNICE: consensus in intermittently-connected environments by exploiting naming with application to emergency response. IEEE ACM Trans Netwing, 2022, 30: 1926–1939

34 NIST. Digital signature standard (DSS) (FIPS 186-5). https://csrc.nist.gov/pubs/fips/186-5/final, 2023
35 Dingledine R, Mathewson N, Syverson P F, et al. Tor: the second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium, 2004. 303–320
36 JHUISI. Charm: a framework for rapidly prototyping cryptosystems. https://github.com/JHUISI/charm
37 Christoph G Günther. An identity-based key-exchange protocol. In: Proceedings of Workshop on the Theory and Application of Cryptographic Techniques, 1990. 29–37
38 Diffie W, Van Oorschot P C, Wiener M J. Authentication and authenticated key exchanges. Des Codes Crypt, 1992, 2: 107–125
39 Kim Y, Adrian P, Gene T. Simple and fault-tolerant key agreement for dynamic collaborative groups. In: Proceedings of the 7th ACM Conference on Computer and Communications Security, 2000
40 Eric Rescorla. The transport layer security (TLS) protocol version 1.3 (RFC 8446). https://www.rfc-editor.org/info/rfc8446/, 2018
41 OpenMLS Contributors. OpenMLS: a rust implementation of the messaging layer security (MLS) protocol. https://github.com/openmls/openmls, 2023
42 SAIDProtocol. Networksimulator. https://github.com/SAIDProtocol/NetworkSimulator, 2018
43 Mahajan R, Spring N, Wetherall D, et al. Inferring link weights using end-to-end measurements. In: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement, 2002. 231–236
44 Wikipedia contributors. Category: disaster management. https://en.wikipedia.org/wiki/Category:Disaster_management, 2018
45 Katz J, Sahai A, Waters B. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Proceedings of the 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2008. 146–162
46 Hao J, Huang C, Ni J, et al. Fine-grained data access control with attribute-hiding policy for cloud-based IoT. Comput Netws, 2019, 153: 1–10
47 Phuong T V X, Yang G, Susilo W. Hidden ciphertext policy attribute-based encryption under standard assumptions. IEEE Trans Inform Forensic Secur, 2016, 11: 35–45
48 Jin C C, Feng X Y, Shen Q N. Fully secure hidden ciphertext policy attribute-based encryption with short ciphertext size. In: Proceedings of the 6th International Conference on Communication and Network Security, 2016. 91–98
49 Long M, Chen L, Tian Y, et al. FEASE: fast and expressive asymmetric searchable encryption. In: Proceedings of the 33rd USENIX Security Symposium, 2024
50 Zhang Z, Zhang W, Qin Z. A partially hidden policy CP-ABE scheme against attribute values guessing attacks with online privacy-protective decryption testing in IoT assisted cloud computing. Future Generation Comput Syst, 2021, 123: 181–195