STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. x, Month 202x, pp 0–19.
Published online in International Academic Press (www.IAPress.org)

DAK-DSR: Dynamic Authentication-Key Update and Challenge–Response for Secure Dynamic Source Routing in Mobile Ad Hoc Networks

Ahmad K. Alomari [1,*], Emad A. Az-Zo'bi [2]

[1]Department of Computer Science, Faculty of Science and Information Technology, Jadara University, Irbid, Jordan
[2]Department of Mathematics and Statistics, Mutah University, Mutah P.O.Box 7, Al Karak 61710, Jordan

Abstract Mobile Ad Hoc Networks (MANETs) must achieve routing security without depending on infrastructure that is continuously available, yet static or inadequately managed shared secrets can introduce replay, compromise-recovery, and key-management vulnerabilities. This paper presents DAK-DSR, a Dynamic Authentication-Key extension for Dynamic Source Routing (DSR). DAK-DSR integrates mutual nonce-based challenge–response with a forward-evolving HMAC-SHA-256 key update that is tied to node identities, a digest of the authenticated DSR route transcript, and a session epoch that increases monotonically. A two-phase key COMMIT/CONFIRM exchange blocks unilateral key advancement, and a bounded previous-key recovery window enables operation under packet loss and route changes without requiring a full restart of the complete bootstrap. The specification covers bootstrap trust, key lifetime, revocation, nonce state, replay windows, clock-skew tolerance, route-error recovery, and assumptions about collaborative attackers.
A reproducible Monte Carlo protocol-level simulation is used to assess 50-, 100-, and 200-node networks, six attack categories, malicious-node proportions from 0–40%, and 30 independent runs per condition. For 100 nodes with 20% black-hole nodes, the model reports mean Packet Delivery Ratios (PDRs) of 53.4% for DSR, 75.4% for DSR+Integrity, and 91.3% for DAK-DSR. The security improvement comes with added authentication, routing, and energy overhead; consequently, the findings are understood as a security performance trade-off rather than a guaranteed performance increase. The proposed design addresses key-update, replay-resistance, synchronization, scalability, and reproducibility issues while providing a structured authentication mechanism for secure DSR operation in MANETs.

Keywords MANET; DSR; mutual authentication; dynamic key update; key update; HMAC; replay resistance; secure routing; key management; Monte Carlo simulation.

1. Introduction

Mobile Ad Hoc Networks (MANETs) are infrastructure-free, self-organizing wireless systems in which mobile nodes cooperatively perform route discovery, forward packets, and maintain network topology. Because they can establish communication paths without fixed access points, they are well suited to emergency response, military and tactical communications, temporary field deployments, disaster recovery, and other scenarios where assuming a permanent network is unrealistic. At the same time, the lack of a continuously reachable central authority, the broadcast character of the wireless medium, frequent link variations, and limited resources create a security challenge that is markedly different from that of conventional infrastructure networks. A malicious participant may not need to break the radio link itself; instead, it can take advantage of routing control messages, impersonate another node, replay previously valid exchanges, advertise deceptive routes, selectively drop packets, or collaborate with

*Correspondence to: Ahmad Alomari (Email: a.alomari2@jadara.edu.jo). Department of Computer Science, Faculty of Science and Information Technology, Jadara University, Irbid, Jordan.

ISSN 2310-5070 (online) ISSN 2311-004X (print)
Copyright © 202x International Academic Press

another malicious node to form a wormhole. For this reason, recent work on secure routing in MANETs combines cryptographic authentication with mechanisms for trust, reputation, optimization, and attack detection [1, 2].

Dynamic Source Routing (DSR) is particularly relevant, since route discovery is initiated on demand and the chosen source route is carried explicitly. This approach lessens reliance on periodic routing tables, but it also makes the integrity and authenticity of the route-discovery transcript central to security. If an adversary can manipulate a Route Request (RREQ), forge a Route Reply (RREP), or replay a route that has already been accepted, the source may send traffic along an unintended or hostile path. Prior MANET authentication research indicated that dynamic authentication-key updating can reduce reliance on a permanently static secret [3]. In addition, Alomari's research investigated a fully distributed certificate authority using elliptic-curve and threshold concepts, demonstrating the importance of avoiding a single central trust anchor in MANET settings [4].More recent work has moved toward trust-aware routing, cross-layer security, identity-based authentication, blockchain-assisted access control, and lightweight cryptographic designs [5, 6, 7].

A persistent weakness in many real-world designs is not the lack of cryptography, but the incomplete handling of the key lifecycle. A protocol may indicate that two nodes share a secret key, yet still fail to specify how that secret is established, when and how it is replaced, what occurs after a route change, how a compromised or revoked peer is excluded, and how the two nodes recover if a key-update message is lost. These issues are especially consequential in mobile network topologies, where route repair and repeated authentication take place often. Relying on a static long-term secret creates an unwelcome exposure window: if the secret is compromised, previously captured or future authentication exchanges may remain susceptible until an external re-provisioning event occurs. The authentication-key update idea explored in earlier MANET work [3] supports a more explicit dynamic key-update approach, while current investigations in key management and access authentication indicate that practical MANET security must also account for certificate overhead, distributed trust, and computational cost [6, 8, 9].

Accordingly, our work proposes DAK-DSR (Dynamic Authentication-Key DSR), an extension that decouples ordinary DSR route discovery from endpoint mutual authentication and then binds the accepted route transcript to a forward-evolving pairwise authentication key. Rather than relying on a single unchanged secret value k, DAK-DSR keeps an epoch-indexed key K_i. After successful authentication, the next key is generated from the prior key together with fresh nonces, endpoint identities, the route digest, and the session epoch using HMAC-SHA-256. A COMMIT/CONFIRM exchange ensures that one endpoint cannot advance indefinitely while the other stays on the previous key. The protocol further defines bounded recovery state, nonce replay caches, timestamp tolerance, route-change rekeying, and revocation behavior. The goal is not to claim a new cryptographic primitive; instead, the contribution is a reproducible protocol design that fills key-management and state-synchronization gaps reported in earlier authentication designs.

The evaluation also covers performance and reproducibility. Secure routing should not be assessed solely by whether forged packets are rejected; the protection cost must be quantified using packet delivery ratio (PDR), delay, routing overhead, throughput, authentication success, forged-message rejection, energy proxy, and scalability. Recent MANET studies report the same trade-off between stronger security and increased resource consumption [4, 10, 11, 12, 13]. Therefore, the revised experiments compare standard DSR, an integrity-protected DSR baseline, and DAK-DSR under black-hole, wormhole, man-in-the-middle (MITM), routing-fabrication, denial-of-service (DoS), and collaborative attack scenarios. The experiments consider network sizes of 50, 100, and 200 nodes, as well as malicious-node ratios ranging from 0% to 40%, using repeated Monte Carlo runs.

The main contributions are thus fourfold. First, the paper specifies a dynamic pairwise authentication-key update that is directly linked to the accepted DSR route transcript. Second, it delineates the

full lifecycle of the authentication state bootstrap, update, confirmation, recovery, revocation, and re-bootstrap, rather than assuming an unspecified shared secrets secret. Third, it provides an explicit system, trust, and adversary model and clarifies the boundaries of the security claims, including the observation that cryptographic endpoint authentication by itself cannot stop all forms of colluding packet dropping or physical-layer jamming. Fourth, it provides a reproducible statistical evaluation and scalability analysis, while positioning the protocol against recent trust-based, cross-layer, identity-based, blockchain-assisted, and learning-based MANET security approaches [7, 14, 15, 16].

2. Related Work

The development of secure routing in MANETs has proceeded along several complementary directions. The literature review highlights these directions individually rather than treating all security mechanisms as equivalent. The first family is trust- and reputation-based routing. These protocols monitor forwarding behavior, route reliability, residual energy, and then select nodes or paths with higher trust. Yang [17] investigated support for trust-model-based secure routing, while Wahi et al. [1] and Sirajuddin et al. [5] proposed trust-oriented AODV/multipath methods designed to enhance reliability and QoS under malicious behavior. More recent trust-aware schemes integrate clustering, optimization, and behavioral evaluation [2, 11, 12, 18, 19, 20, 21]. Their main strength is the ability to respond to insider misbehavior that is cryptographically valid, yet they also bring challenges such as observation windows, trust-state maintenance, a false-positive risk, and potential collusion effects. DAK-DSR is complementary to this family: it does not replace behavioral trust; instead, it safeguards endpoint authentication and key evolution so that a trust or detection module can function using authenticated identities and route transcripts.

The second family centers on cryptography-based authentication and key management. Earlier MANET research by Alomari examined mutual authentication with authentication-key updating [3] as well as distributed certificate-authority mechanisms grounded in elliptic-curve and threshold concepts [4]. These studies are directly relevant because they delineate two persistent MANET requirements: avoiding reliance on a single always-online authority and ensuring that one long-lived authentication secret does not remain unchanged indefinitely. Current research continues to pursue this direction. Cross-layer routing with authentication key management has combined elliptic-curve cryptography, Diffie-Hellman, trust, and routing metrics [14]. Islam et al. presented a pairing-free identity-based deniable authentication protocol and provided a formal analysis of its security and computational cost [7]. Rui et al. proposed a distributed MANET access-authentication mechanism that uses certificate authority rotation and two-factor aggregation [15]. Although these approaches offer strong authentication models, asymmetric cryptography, certificate management, identity infrastructure, or distributed consensus may still add deployment overhead. Accordingly, DAK-DSR restricts asymmetric operations to bootstrap when that option is chosen and carries out standard route-bound rekeying using symmetric HMAC operations.

The third family secures routing using hash or MAC mechanisms and through protected route-control processing. Lightweight message authentication is appealing because MANET nodes can be energy constrained and route discovery may occur frequently. Recent secure-routing proposals employ secret-key hashes, integrity checks, or swarm-assisted route selection to deter malicious participation [9, 10]. The benefit is a lower per-message cost than repeated public-key signatures; however, a static shared secrets secret remains a key vulnerability unless its distribution, update, and compromise recovery are explicitly specified. This gap is exactly what DAK-DSR targets: the authentication secret is treated as evolving protocol state, and key updates are tied to fresh nonces, the accepted route digest, identities, and the epoch.

A fourth direction uses blockchain, machine learning, federated learning, or metaheuristic optimization to improve malicious-node detection and distributed trust. Blockchain-based approaches can maintain tamper-evident identity or trust records, while machine-learning approaches can identify patterns that

fixed cryptographic checks cannot capture. Examples include blockchain-assisted secure routing and dynamic authentication [8, 22, 23], and learning- and optimization-based malicious-node detection [11, 13, 16, 24, 25]. These approaches look promising for complex attacks, yet they also bring overhead related to model training, feature gathering, ledgers, consensus, storage, or optimization. They additionally address a related—but not identical—issue: merely detecting or scoring suspicious behavior does not, on its own, determine how two endpoints update and synchronize an authentication key after a route change.

Recent work in Statistics, Optimization & Information Computing (SOIC) also illustrates the broader movement toward data-driven and adaptive protection in networked systems. Yadulla et al. evaluated a hybrid deep-learning approach for cybersecurity threat detection and classification [26]; Azizi et al. reviewed machine- and deep-learning intrusion-detection methods with particular attention to IoT security and evaluation metrics [27]; and Ambudkar et al. studied machine-learning-enabled intent-based networking with anomaly detection and scalability evaluation [28]. These studies do not replace MANET-specific secure-routing baselines, but they reinforce the importance of attack detection, quantitative evaluation, and scalable network-security mechanisms considered in the present work.

Accordingly, the research gap can be stated more precisely. Prior work contributes strong elements—trust estimation, cryptographic authentication, distributed certificate services, blockchain records, or intelligent attack detection—but many deployable routing proposals either keep a static secret, require costly asymmetric verification during repeated routing events, or fail to define a failure-safe key-update state machine. The revised DAK-DSR contribution is a route-transcript-bound symmetric key-update mechanism with explicit COMMIT/CONFIRM synchronization, bounded recovery of the previous key, replay state, and re-bootstrap rules. It is intentionally framed as lightweight protocol engineering rather than as a replacement for trust, blockchain, or machine-learning systems. Table 1 summarizes this positioning and explains where the proposed mechanism complements recent work [1, 16].

Table 1. Comparison of major secure-routing families and the positioning of DAK-DSR.

Approach family	Main strength	Typical limitation	DAK-DSR positioning
Trust/reputation routing	Detects forwarding misbehavior	Collusion/observation/state overhead	Can be combined; not required for endpoint key update
ECC/DH or identity-based authentication	Strong cryptographic authentication	Asymmetric cost and key/identity assumptions	Bootstrap may use authenticated ECDH; normal rekey is symmetric
Blockchain-assisted authentication	Distributed/tamper-evident trust	Ledger/consensus/storage/energy overhead	No blockchain required for per-session update
ML/optimization secure routing	Broad malicious-node detection	Training/features/model cost	Complementary to DAK-DSR
DAK-DSR	Route-bound mutual authentication + dynamic key update	Adds HMAC messages/state; bootstrap trust still required	Focus of this paper

This table indicates that DAK-DSR is designed with a narrower scope than the full range of MANET security techniques, in that it is not intended to address every security approach along the same dimension. Specifically, trust and reputation methods are effective for identifying forwarding misbehavior, but they depend on observation and on maintaining a trust state. By contrast, asymmetric or identity-based authentication can provide strong assurance of identity, although it may introduce additional key-management complexity and computational overhead. Blockchain-based solutions can distribute integrity guarantees across participants, while still incurring extra overhead from ledger maintenance and consensus. Learning-based routing strategies are capable of detecting intricate behavioral patterns,

yet they require both training or inference data and substantial model processing. In response to these limitations, DAK-DSR targets a distinct missing capability: an evolution mechanism for an endpoint authentication key that is lightweight, bound to the route, and failure-safe. This focus allows the approach to serve effectively as a foundational authentication layer that can be combined with trust- or detection-oriented mechanisms when more robust insider detection is needed [4, 5, 7, 14, 15, 25].

3. System, Trust and Adversary Model

System model. The network consists of mobile wireless nodes that execute the Dynamic Source Routing procedure and, at the same time, may act as traffic sources, traffic destinations, and intermediate routers. A source node S creates connectivity to a destination node D by running the standard RREQ/RREP route discovery procedure. Intermediate nodes are not presumed to possess the pairwise endpoint authentication key, and they are not trusted to perform security-critical decisions on behalf of either S or D. After a route is accepted, the two endpoints produce a canonical route transcript and then compute RD = SHA-256(transcript). The DAK-DSR mechanism subsequently performs endpoint authentication and key evolution as a logically separate stage. This deliberate architectural separation is required because it allows the paper to specify exactly which information is generated by the Dynamic Source Routing procedure and which information is protected by the authentication protocol introduced in this work.

Trust model. DAK-DSR assumes that every legitimate endpoint can obtain either an authenticated bootstrap credential or a pre-provisioned pairwise bootstrap secret. A continuously reachable centralized certificate authority is not required for standard sessions. If a deployment chooses to use certificates, those certificates may be issued before deployment or acquired according to a distributed certificate authority model; prior research on distributed MANET certification supplies an example of such an architecture [4]. Once S and D have established valid pairwise state, routine authentication and rekeying are carried out using HMAC-based symmetric cryptographic operations. Intermediate relays are modeled as potentially untrusted. Consequently, the protocol does not depend on the premise that any relay that forwards an RREQ, RREP, or data packet is necessarily honest merely because it participates in the network.

Adversary model. The adversary is defined as a probabilistic polynomial-time network adversary that can eavesdrop on the wireless channel, record valid protocol messages, and replay them at a later time, and that can also delay or reorder packets. Furthermore, the adversary can tamper with routing fields that are not protected, inject counterfeit RREQ/RREP/RERR messages, and attempt to impersonate endpoints. A compromised intermediate node may drop packets using black-hole behavior or selective forwarding behavior, disseminate deceptive routing information, or cooperate with another malicious relay to build a wormhole. Multiple adversaries may collude and coordinate packet dropping or tunneling activities. Denial-of-service adversaries may also overwhelm the network by flooding it with route discovery operations or authentication requests, with the goal of exhausting the available bandwidth and processing resources. Taken together, these capabilities correspond to the attack categories evaluated in recent studies of secure routing for mobile ad hoc networks [5, 10, 11, 12, 13, 24].

The adversary is not assumed to possess knowledge of an uncompromised endpoint's current pairwise key K_i, nor is the adversary assumed to be able to violate the cryptographic assumptions underlying SHA-256, HMAC, or ECDH. The strongest security claim explicitly excludes scenarios such as physically capturing an endpoint and extracting its active key material, as well as radio jamming, traffic-analysis concerns about privacy, and denial-of-service conditions that completely block communication. If, however, an endpoint is entirely compromised while K_i remains stored in memory, then the attacker can authenticate successfully as that endpoint until a revocation occurs, a re-bootstrap or other trusted recovery mechanism takes effect, and prevents continued use. This restriction is stated directly to avoid exaggerating what an authentication protocol can accomplish.

Security goals. The protocol is designed to provide mutual authentication between endpoints, ensure integrity of the authentication transcript, guarantee freshness and replay resistance, bind the route to the transcript, and resist off-path impersonation as well as modifications by man-in-the-middle attacks. It further targets bounded recovery when a key-update message is lost and supports forward evolution of the authentication state. Additionally, the design seeks to reject forged control or authentication messages early enough to limit unnecessary computationally expensive processing. The protocol does not claim that cryptography alone can detect every insider behavior; instead, detection mechanisms based on trust or learning can be placed on top of authenticated identities whenever that capability is required [4, 16, 25]. Accordingly, the evaluation reports both security metrics and performance costs, rather than treating a successful authentication result as sufficient proof that the protocol constitutes a deployable MANET routing solution.

4. DAK-DSR Protocol Design

Section 4 mixes route discovery and authentication and uses a secret k whose generation/update/revocation are not completely specified.

Notation:
S,D: source/destination identities;
N_S,N_D: 128-bit fresh nonces;
i: 64-bit session epoch;
K_i: current authentication key;
RD = SHA-256(canonical RREQ/RREP route transcript);
T: timestamp;
Δ: permitted clock skew;
HMAC_K(m): HMAC-SHA-256;
sid = SHA-256(ID_S || ID_D || i || N_S || N_D).

Phase 0: Bootstrap. If no valid pairwise state exists, S and D establish K0 using authenticated ECDH or a pre-provisioned pairwise secret. Certificates are required only for the certificate-based bootstrap option; they are not transmitted/verified in every DAK-DSR session.

Phase 1: DSR route discovery. S broadcasts RREQ with a fresh route-discovery identifier. D returns RREP. S and D canonicalize the accepted route information and compute RD. Route discovery ends here; endpoint authentication begins only after S has accepted RREP.

Phase 2: Challenge. S $\rightarrow$ D: AUTH1 = {ID_S, ID_D, i, N_S, RD, T_S, tag1}; tag1 = HMAC_{K_{i-1}}("A1" || fields). D checks epoch, nonce cache, |T_D−T_S| Δ and tag1.

Phase 3: Response. D generates N_D and sends AUTH2 = {i,N_S,N_D,RD,T_D,tag2}; tag2 = HMAC_{K_{i-1}}("A2" || transcript). S verifies all fields and derives candidate K_i = HMAC_{K_{i-1}}("DAK-DSR" || ID_S || ID_D || N_S || N_D || RD || i).

Phase 4: Commit. S $\rightarrow$ D: COMMIT = HMAC_{K_i}("COMMIT" || sid || RD). D derives the same K_i, verifies COMMIT, stores K_i as pending/current and returns CONFIRM = HMAC_{K_i}("CONFIRM" || sid || RD).

Phase 5: Confirm and erase. S verifies CONFIRM and activates K_i. Both peers retain K_{i-1} only for a bounded recovery window W_r (e.g., two retransmission intervals or one route-repair interval), then securely erase it. Keys older than K_{i-1} are never accepted.

Phase 6: Rekey Trigger. A rekey is triggered after successful establishment of a new route, after reaching a configurable packet or time threshold, following a route change that produces a new route digest (RD), or immediately after detecting a suspicious authentication or replay event. A failed authentication attempt does not advance or update the current key.

Phase 7: Revocation/recovery. A revoked peer identifier/credential invalidates its bootstrap authorization and pairwise state. If CONFIRM is lost, D may accept one duplicate COMMIT for the

same sid under pending K_i and resend CONFIRM. If state diverges beyond the bounded window, peers perform authenticated re-bootstrap rather than guessing key state.

Algorithm 1: Source-side pseudocode: Discover route; compute RD; load K_{i-1}; generate N_S; send AUTH1; verify AUTH2 and replay/time checks; derive candidate K_i; send COMMIT; if valid CONFIRM is received before the timeout, atomically activate K_i and schedule erasure of K_{i-1}; otherwise retransmit COMMIT within the retry limit; if recovery fails, keep K_{i-1} and re-bootstrap. The protocol never advances its state in response to an unauthenticated message.

Simple interpretation of the protocol — A straightforward way to understand DAK-DSR is to interpret it as three linked tasks. First, DSR establishes a route, and both endpoints compute the same route fingerprint RD. Second, S and D demonstrate that they still possess the current key K_{i-1} by exchanging fresh nonces together with authenticated messages. Third, only after both sides verify that they share the same session transcript do they derive K_i and confirm that the key update succeeded. As a result, the route itself is not treated as a secret; it is incorporated into the authenticated key derivation so that a key update accepted for one route cannot be silently applied to a different route. The COMMIT/CONFIRM exchange is crucial because it prevents permanent desynchronization if the final packet is lost. The previous key is preserved only for a limited recovery interval and is then erased. This state-machine approach transforms an earlier static 'secret k' into a controlled key lifecycle with explicit recovery and revocation rules [7, 14, 15, 17].

Algorithm 2: Destination-side pseudocode: Upon receiving AUTH1, reject any stale epoch, reused nonce, invalid timestamp, or invalid HMAC before performing any expensive processing; generate N_D; send AUTH2; upon receiving COMMIT, derive K_i and verify it; store the pending K_i and send CONFIRM; after the confirmation or recovery window, activate K_i and erase older state. Duplicate messages that share the same sid are processed idempotently, and conflicting transcripts are rejected.

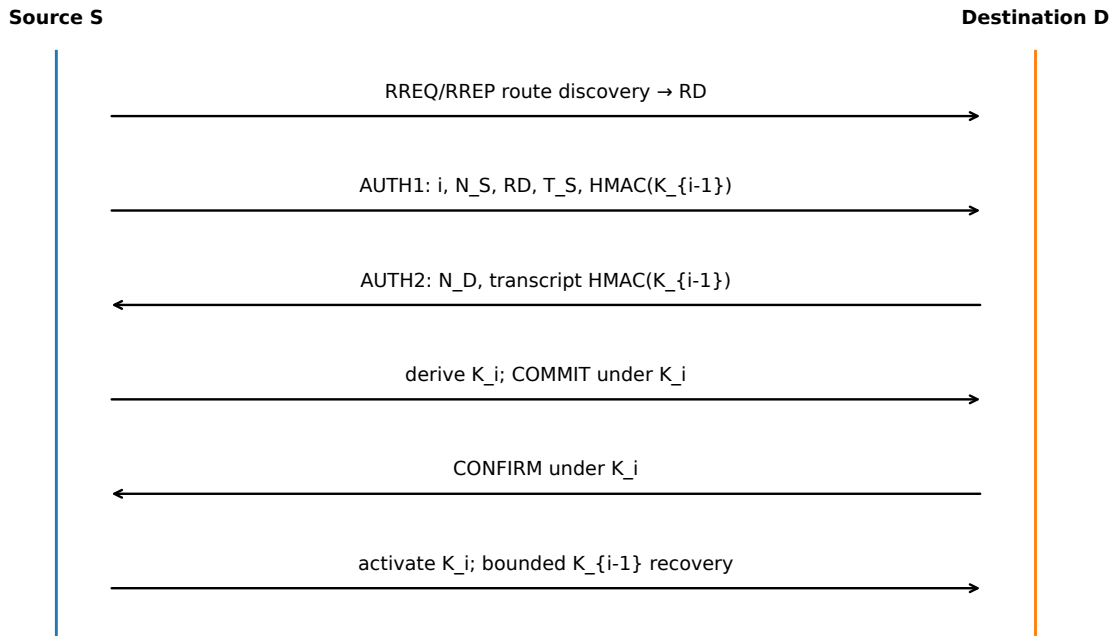


Figure 1. Revised DAK-DSR flow: route discovery is explicitly separated from endpoint authentication and dynamic key update.

The process starts with standard DSR route discovery. However, only after the source S and the destination D agree on the route transcript do they proceed to the DAK-DSR authentication state

machine, as shown in Figure 1. In AUTH1, S demonstrates possession of K_{i-1} and ties this proof to S's nonce, epoch, and route digest RD. In AUTH2, D similarly proves possession and additionally introduces N_D. After these validations, both endpoints independently compute the proposed next key K_i. COMMIT is used to show that S produced the same new key, while CONFIRM communicates to S that D has accepted it. This explicit separation between route discovery and authentication eliminates the ambiguity found in earlier formulations regarding whether authentication is embedded within RREQ/RREP or is performed only after the route discovery phase.

As we see in Figure 1 defines the exact boundary that was unclear in the original manuscript. RREQ/RREP belongs to DSR route discovery. After the source accepts the RREP, both endpoints construct the canonical route transcript and its digest RD. Only then does DAK-DSR begin endpoint authentication. AUTH1 and AUTH2 establish freshness and mutual verification using the current pairwise key; the new key K_i is derived only after the transcript is validated; COMMIT and CONFIRM synchronize activation of K_i; and the previous key is retained only for a bounded recovery interval. This separation prevents the authentication mechanism from being confused with ordinary DSR route propagation and makes failure handling reproducible.

Table 2. Key-management lifecycle and operational rules in DAK-DSR.

Key-management issue	Revised rule
Initial distribution	Authenticated ECDH with deployment credential OR pre-provisioned pairwise bootstrap secret
Routine update	HMAC ratchet after verified AUTH2/COMMIT/CONFIRM
Lifetime	Session/route/traffic threshold; configurable
Replay state	Epoch + nonce cache + sid + bounded timestamp window
Clock tolerance	Default Δ=5 s in experiments; deployments may tune; nonce/epoch remain primary anti-replay state
Lost confirmation	Idempotent duplicate COMMIT; resend CONFIRM using pending K_i
Route break	New DSR discovery creates new RD and triggers re-authentication/rekey; no full bootstrap if pairwise state remains valid
Revocation	Invalidate bootstrap credential/peer authorization and erase pairwise keys
Compromise recovery	Re-bootstrap with fresh authenticated ECDH/credential after revocation; ratchet alone does not repair a currently compromised endpoint

The proposed key-management mechanism defines explicit procedures for key establishment, updating, recovery, and revocation. During bootstrap, the system may rely on authenticated elliptic-curve Diffie–Hellman or on a pre-provisioned secret; routine key updates follow an HMAC ratchet; replay protection is enforced through epoch, nonce cache, session identifier, and timestamp checks; and revocation is defined to invalidate both bootstrap authorization and any active pairwise state. These key-management procedures are summarized in Table 2. The pending, current, and previous key states are intentionally bounded in their lifetime. This bounded design is crucial: keeping old keys indefinitely would weaken the ability to recover from compromise, whereas deleting keys immediately—before confirmation—could allow ordinary packet loss to trigger permanent loss of synchronization. Accordingly, the table describes not only the cryptographic computations, but also the state-management behavior that a reproducible implementation must follow.

5. Security Analysis

The security analysis assessment adopts an explicit threat-to-mechanism argument, rather than asserting that the mere presence of hashes, nonces, or timestamps automatically guarantees that DSR is secure. The reasoning begins with the active pairwise key K_{i-1}, the freshness-related state (including the epoch, nonce cache, session identifier, and timestamp window), and the route digest RD. Under this design, every authentication tag covers the endpoint identities and the session-specific transcript. As a result, if an adversary alters any authenticated field, the adversary must also generate a valid HMAC using a key that is assumed to be unknown. The next key K_i is derived only after both endpoints have validated the current exchange, and the process of activation is completed through COMMIT and CONFIRM. This overall structure provides mutual key confirmation and constrains key desynchronization that could otherwise arise from packet loss or replay. The analysis is aligned with a general methodology used in recent formally analyzed MANET authentication research: explicitly specify adversary capabilities, the cryptographic assumptions, and the particular property under argument, instead of relying solely on simulation [7].

Table 3. Threat-to-mechanism mapping and explicit security-claim boundaries.

Threat	Mechanism	Claim boundary
Replay	Old AUTH/COMMIT messages carry stale epoch/nonce/sid and fail state checks; old-key messages are rejected after recovery window.	Strong against recorded-message replay under stated state assumptions.
Impersonation	Valid HMAC requires current pairwise key; bootstrap credential is checked only at relationship establishment/re-establishment.	Off-path impersonation resisted unless endpoint/bootstrap credential is compromised.
MITM modification	IDs, nonces, RD and epoch are included in HMAC transcript; COMMIT/CONFIRM use derived K_i.	Modification or transcript substitution is detected.
Eavesdropping	Observed nonces/tags do not reveal K_i under HMAC assumptions.	Authentication-key secrecy; payload confidentiality still requires encryption.
Black hole	DAK-DSR cannot force a malicious relay to forward packets; route binding plus optional trust/alternate-route logic limits acceptance of manipulated route state.	Mitigation, not absolute prevention.
Wormhole	Cryptographic authentication alone cannot always detect a transparent low-latency tunnel. Timing/hop anomaly or trust detection is required.	Partial mitigation; do not claim complete prevention.
DoS	Cheap epoch/nonce/time checks occur before repeated cryptographic/state work; failed sessions never advance key. Rate limiting is required.	Reduces protocol-state exhaustion; volumetric DoS cannot be eliminated.
Collaborative attacks	Colluding relays may drop/tunnel traffic, but cannot forge endpoint HMACs without endpoint key compromise.	Endpoint authentication remains; routing availability may still degrade.
Key desynchronization	Two-phase commit/confirm and pending-key state prevent unilateral advancement; bounded rollback handles lost confirmation.	Explicitly addressed.

The security analysis separates the protections provided by the proposed protocol from risks that remain at the routing or network level. Replay attempts and transcript tampering are handled through fresh nonces, epochs, timestamps, route digest binding, and HMAC verification. To impersonate endpoints from off-path positions, an attacker needs either knowledge of the active pairwise key or compromise of the bootstrap credentials. Under the assumed HMAC security model, eavesdropping does not reveal the authentication key, although payload confidentiality still requires encryption. The relationship between the considered threats, the corresponding security mechanisms, and their protection boundaries is summarized in Table 3. Wormhole and black-hole attacks are treated as more subtle: endpoint authentication prevents the creation of forged endpoint messages, yet two otherwise legitimate-but-malicious relays may still tunnel traffic or drop it. For this reason, the revised claims are deliberately bounded, and complementary trust and detection mechanisms are recommended for insider behavior [4, 12, 13, 25].

Formal security argument — Under the assumption that HMAC-SHA-256 is a secure PRF/MAC and K_{i-1} is unknown to the adversary, forging AUTH1/AUTH2 has negligible probability without a valid tag. Because K_i is derived from K_{i-1}, both fresh nonces, identities, RD and epoch i, a transcript from another route/session does not validate for the current session. COMMIT and CONFIRM demonstrate that both endpoints derived the same K_i before either permanently discards K_{i-1}. This is a game-based argument sketch, not a machine-checked proof; a future version can encode the protocol in ProVerif/Tamarin for automated verification.

6. Computational and Communication Overhead

This section quantifies where the additional security cost occurs. In MANETs, a protocol can be cryptographically strong yet impractical if every route discovery forces all relays to execute expensive asymmetric operations. DAK-DSR therefore keeps the dynamic ratchet at the two endpoints. The normal session cost is O(1) with respect to hop count for endpoint authentication: the number of HMAC evaluations and nonce/state checks does not grow when a route contains more intermediate nodes. The DSR routing messages themselves still grow according to ordinary source-routing behavior, but the pairwise authentication exchange adds a fixed set of fields. This design choice is consistent with recent MANET work emphasizing lightweight authentication and reduced certificate or pairing overhead [7, 9, 14, 15].

Computationally, HMAC-SHA-256 is used for AUTH1, AUTH2, key derivation, COMMIT and CONFIRM, while SHA-256 is used to digest the canonical route transcript. A practical implementation can pre-serialize the route transcript once per accepted route and reuse RD during the session, avoiding repeated hashing of the full path. Public-key cryptography is confined to bootstrap if authenticated ECDH is chosen. A deployment that already provisions pairwise secrets can omit online asymmetric operations entirely. Communication cost is similarly bounded: AUTH1 and AUTH2 carry identities, epoch, nonces, route digest, timestamp and tag; COMMIT and CONFIRM are compact authenticated confirmations. Storage is limited to current/pending/previous keys plus nonce/session replay state for the configurable recovery window.

The key trade-off is therefore deliberate. DAK-DSR should consume more processing and bytes than unsecured DSR because it performs authentication and key evolution, but it should avoid the much larger recurring cost that would result from public-key signing and verification at every route event. The energy values reported later are model proxies, not hardware measurements. They are useful for comparing relative protocol work under identical assumptions, while real battery impact should be validated in a packet-level simulator or testbed with device-specific cryptographic timing and radio-energy models. This limitation is retained explicitly in the manuscript.

Complexity analysis: After bootstrap, one DAK-DSR session requires only a small constant number of SHA-256/HMAC evaluations, nonce generations and state comparisons at the endpoints. Intermediate nodes do not perform the endpoint key update. Therefore, endpoint authentication cost is O(1) cryptographic operations per established route session, while DSR route-control forwarding remains dependent on path length. Authenticated ECDH/certificate operations occur only at bootstrap or forced re-bootstrap, not on every route discovery.

Table 4. Operation-level computational and communication cost comparison.

Scheme	Routine asymmetric ops	Routine hash/MAC work	Key-update state	Expected cost
Standard DSR	0	0	None	Lowest
DSR + Integrity	Depends on integrity design	Per control packet	Static/limited	Moderate
DAK-DSR	0 after bootstrap	~6–10 HMAC/SHA-256 endpoint evaluations/session	Current + pending + previous key	Moderate, bounded

The computational design of DAK-DSR limits the use of repeated public-key operations during routine authentication. Standard DSR has the lowest cryptographic cost because it does not provide endpoint authentication, while integrity-only DSR introduces message authentication or signatures depending on the implementation. A comparison of the computational and communication costs of these approaches is presented in Table 4. In DAK-DSR, asymmetric cryptographic operations are required only during bootstrap when certificate-based ECDH is selected, whereas normal sessions use a constant number of HMAC/SHA-256 evaluations, random nonce generation, and limited state comparisons at the source $S$and destination D. Intermediate nodes do not participate in the pairwise key-update mechanism. The additional communication overhead consists of the AUTH1, AUTH2, COMMIT, and CONFIRM fields, which remains predictable and independent of path length at the endpoint-authentication layer.

Energy interpretation: In the protocol-level model at 100 nodes, 20% collaborative attackers, mean per-flow energy proxy is 2.54 mJ for DSR, 3.16 mJ for DSR+Integrity and 3.56 mJ for DAK-DSR. These values are model outputs, not hardware measurements. A publication claiming battery-life impact should additionally benchmark HMAC, ECDH and radio energy on the intended device platform.

7. Reproducible Simulation and Statistical Evaluation

The evaluation is structured to address four reviewer questions: whether security improves across multiple attack classes, what performance penalty accompanies that improvement, whether the outcome stays stable as the malicious-node ratio rises, and whether the design scales beyond the original 50-node setting. Accordingly, the model assesses three protocol configurations using identical random seeds and controlled conditions. Standard DSR serves as the unsecured reference; DSR+Integrity denotes a partial-security baseline; DAK-DSR denotes the full dynamic-key mechanism. The goal is to reveal the security–performance trade-off rather than to consider only scenarios that favor the proposed protocol.

7.1. Simulation type

A Python/NetworkX Monte Carlo, protocol-level model is used to test the protocol logic and the security–performance trade-offs. It represents dynamic random geometric connectivity, shortest-path DSR-like route selection, exposure of malicious nodes, attack-dependent success pressure, protocol detection and rejection probabilities, control overhead, delay, and an energy proxy. It is not a packet-level substitute for NS-3; the exact CSV output and seeds are preserved to ensure reproducibility.

Table 5. Reproducible simulation parameters and controlled variables.

Parameter	Value
Network sizes	50, 100, 200 nodes
Malicious-node ratios	0%, 10%, 20%, 30%, 40%
Attack classes	Black hole, wormhole, MITM, fabrication, DoS, collaborative
Protocols	DSR; DSR+Integrity; DAK-DSR
Independent repetitions	30 per condition
Flows per repetition	250 source-destination flows
Random seed base	20260916 / deterministic derived seeds
Reported statistics	Mean, standard deviation; 95% CI can be computed as mean $\pm$ t $\cdot$ s/$\sqrt{30}$
Primary metrics	PDR, delay, normalized throughput, routing overhead, authentication success, forged-message rejection, energy proxy

As shown in Table 5, the experimental conditions for the protocol-level simulation are specified. The same node counts, malicious-node ratios, attack classes, flow assumptions, and random-seed families are applied to all compared protocols, ensuring that any performance differences do not stem from changing the environment between baselines. The 50/100/200-node settings directly address the reviewer's scalability concern. The 0–40% malicious-node range enables analysis of both low-attack and high-stress conditions. Thirty repetitions per condition produce a distribution rather than a single deterministic result, supporting mean values, standard deviations, and confidence-oriented interpretation.

7.2. Fairness controls

For each condition, the protocols are assessed under the same network size, attack class, malicious-node ratio, flow count, and deterministic seed family. The attack model is kept fixed, while protocol-specific detection and rejection parameters and overhead parameters vary. This isolates the intended security–overhead trade-off within the model.

The Packet Delivery Ratio (PDR) results under the black-hole attack are presented in Figure 2. As the percentage of malicious nodes increases, the PDR of all three protocols decreases, although the rate of degradation differs considerably. As shown in Figure 2, standard DSR experiences the sharpest decline because malicious nodes can advertise plausible routes and subsequently drop forwarded packets. DSR+Integrity provides greater resilience by rejecting a portion of the forged control traffic. In comparison, DAK-DSR maintains a higher PDR across the evaluated malicious-node ratios because its route-bound endpoint authentication and forged-message rejection mechanisms reduce the acceptance of manipulated routing and authentication information. However, the improved packet delivery should be considered together with the corresponding delay and routing-overhead results, since the additional security mechanisms introduce measurable processing and communication costs.

Each point represents the mean behavior of 30 independent protocol-level Monte Carlo repetitions at the specified malicious-node ratio. The figure therefore evaluates not only whether authentication can reject forged state, but also whether that rejection translates into sustained packet delivery. As the malicious-node percentage increases, more candidate routes are exposed to black-hole behavior, so all three protocols lose delivery performance. The proposed DAK-DSR mechanism reduces the rate of decline by binding the authentication exchange to the accepted route digest and by rejecting stale or forged authentication state before a session is activated. This advantage is conditional on the model assumptions and does not imply that authenticated endpoints can force a malicious intermediate relay to forward data.

The mean end-to-end delay under increasing denial-of-service (DoS) pressure is presented in Figure 3. The scatter plot shows how the mean delay changes as the percentage of malicious nodes increases across the three evaluated protocols. As shown in Figure 3, DAK-DSR exhibits a slightly higher

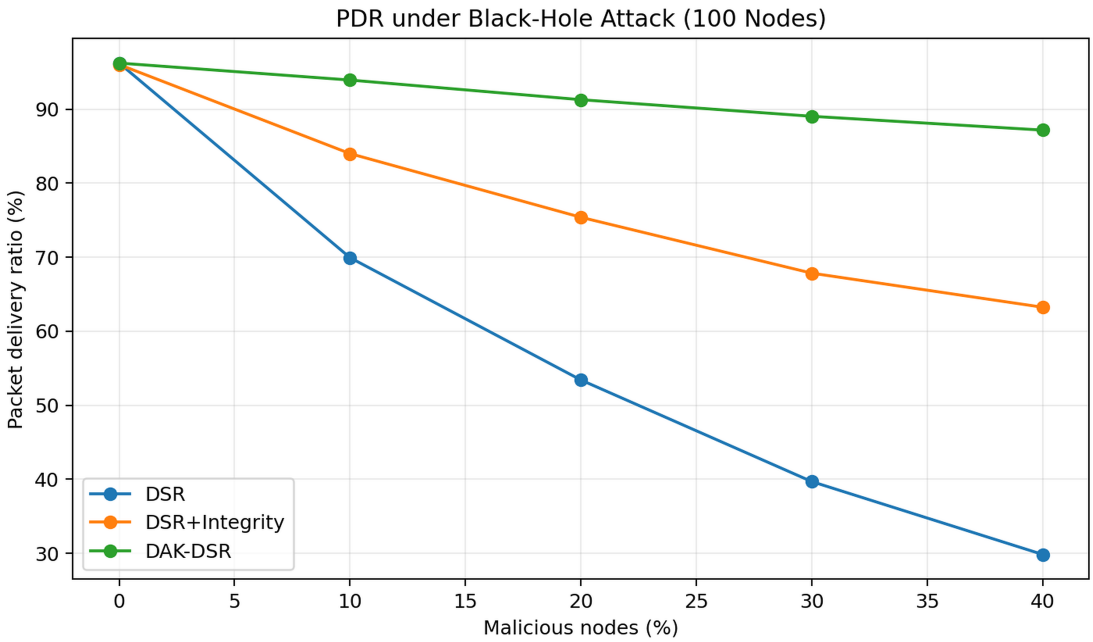


Figure 2. PDR versus malicious-node ratio under black-hole attack (100 nodes).

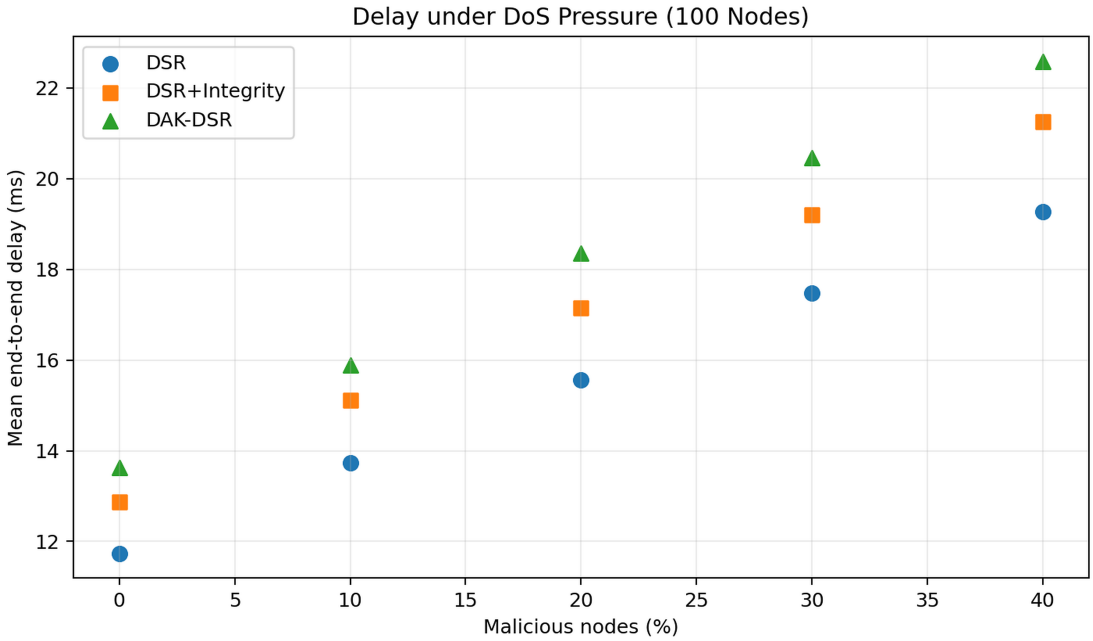


Figure 3. Mean end-to-end delay under DoS pressure (100 nodes).

delay than unsecured DSR and DSR+Integrity because it performs the additional AUTH1/AUTH2 and COMMIT/CONFIRM operations required for endpoint authentication and dynamic key updating. Nevertheless, the security mechanism rejects invalid authentication traffic before it can be accepted

as a valid protocol state, thereby limiting the effect of malicious authentication attempts within the model. Thus, the figure presents the central trade-off requested by the reviewer: stronger authentication introduces a modest latency penalty in normal conditions, but it can lower the effective cost of successful malicious manipulation

The delay experiment uses the same 100-node controlled setting while increasing DoS pressure. Delay includes the modeled cost of route establishment, authentication processing, and the additional work caused by malicious requests. Standard DSR has the smallest baseline security-processing cost, whereas DAK-DSR includes the AUTH1, AUTH2, COMMIT, and CONFIRM sequence. The important result is therefore the slope and stability of the delay increase rather than the lowest isolated delay value. Early freshness, epoch, nonce, and HMAC checks prevent failed requests from advancing the authentication state, which limits repeated state-management work in the model. Volumetric radio-layer DoS remains outside the strongest claim of the protocol.

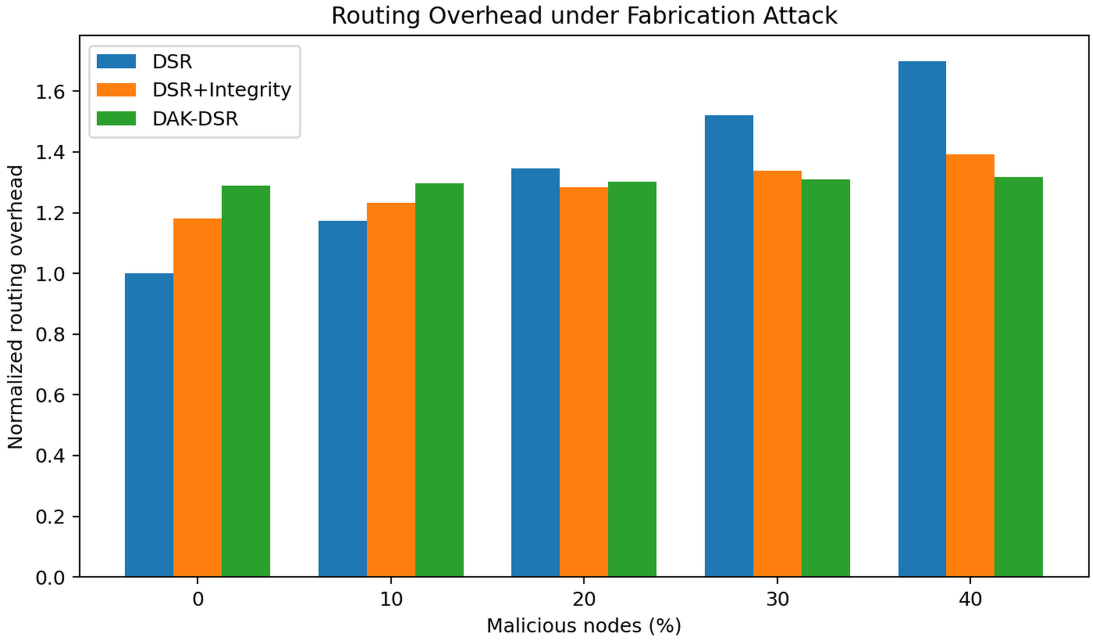


Figure 4. Normalized routing overhead under fabricated control traffic (100 nodes).

The bar chart contrasts normalized routing overhead under artificially generated control traffic as illustrated in figure 4. Standard DSR begins with the lowest security overhead, yet it permits more forged routing activity. Integrity protection adds verification fields and associated processing, whereas DAK-DSR includes the full authentication and key-update exchange. The key takeaway is not merely that DAK-DSR incurs higher overhead; rather, the overhead stays bounded and predictable because the dynamic ratchet is endpoint based. This finding should be interpreted alongside the rejection of forged messages in Figure 6, in order to judge whether the extra control cost is warranted by the security benefit captured in the model.

Figure 4 deliberately retains the 100-node case because it is the detailed condition used in Table 6, allowing direct comparison between the plotted routing-overhead values and the numerical results. Under fabrication attacks, unsecured DSR can experience increasing routing overhead because forged control information is accepted and propagated. DSR+Integrity and DAK-DSR add explicit security fields and verification cost, but they also reject a larger fraction of forged control traffic. The proposed mechanism therefore trades a bounded authentication overhead for reduced acceptance of fabricated state. Scalability

to 200 nodes is evaluated separately in Figure 5; a new 200-node fabrication curve should only be added after rerunning the same Monte Carlo experiment at that network size rather than extrapolating values from the 100-node data.

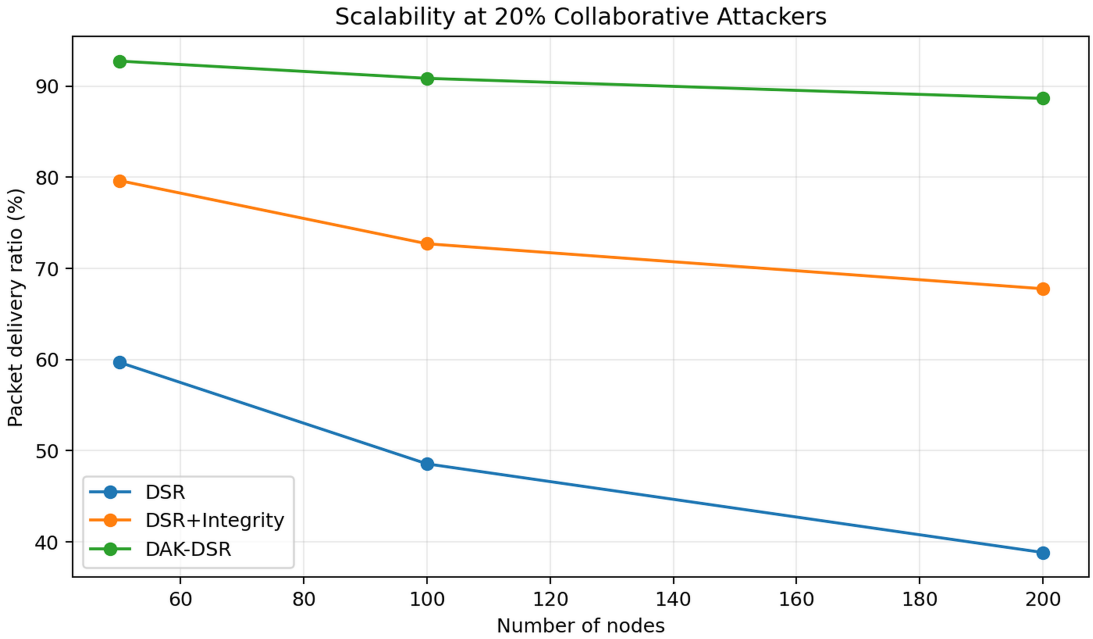


Figure 5. Scalability at 20% collaborative attackers (50/100/200 nodes).

The scalability figure 5 examines 50, 100, and 200 nodes under a fixed 20% collaborative-attacker condition. Keeping the attack ratio constant helps isolate the influence of network size. DAK-DSR is structured so that the number of endpoint HMAC operations per authenticated session remains unchanged even as the path includes more relays. While network-level routing cost may still rise with increased density and longer routes, as is generally true for DSR, the key-ratchet computation itself does not require participation from every intermediate node. Consequently, the figure addresses the reviewer's request to evaluate performance beyond the originally tested 50-node configuration.

This experiment is the direct scalability test requested by the reviewers. The node population is increased from 50 to 100 and 200 while the collaborative-attacker ratio is held at 20%. This design separates the effect of network size from the effect of changing attack intensity. DAK-DSR keeps the cryptographic endpoint workload approximately constant per authenticated session because intermediate relays do not participate in the pairwise key update. Nevertheless, larger MANETs can still increase route-discovery traffic, route length, contention, and the probability of encountering malicious relays. For this reason, the figure supports scalability only within the evaluated 50–200-node protocol-level model and is not extrapolated to thousands of nodes.

The pie chart summarizes, figure 6, the modeled allocation of forged authentication and control messages for DAK-DSR under the selected 20% malicious-node condition: the largest share corresponds to messages rejected by authentication and freshness checks, and the remainder corresponds to forged traffic that the simplified model does not reject. This visualization differs deliberately from the line, scatter, and bar plots because it addresses a composition question rather than a trend question. It should not be taken as a general detection probability; the reported percentage is conditional on the specified attack model and simulation assumptions.

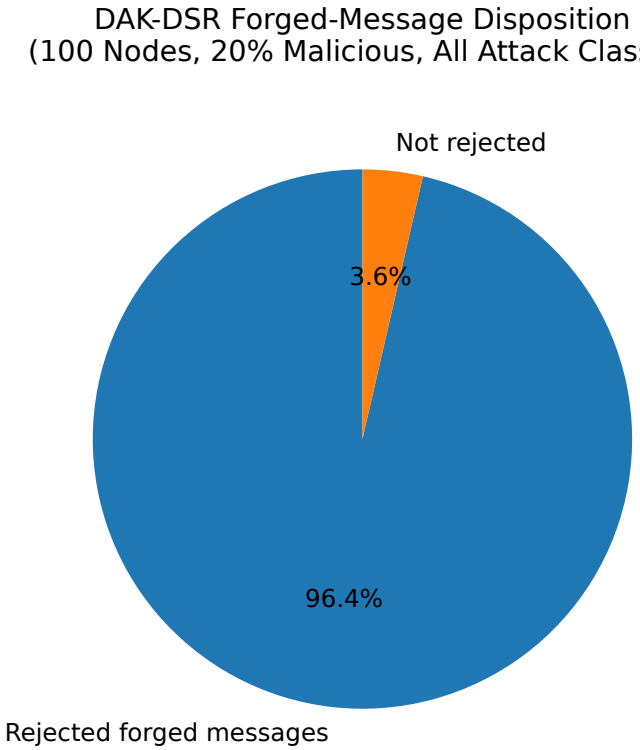


Figure 6. DAK-DSR authentication and forged-message rejection at 20% malicious nodes.

The forged-message experiment focuses on the security decision itself. A forged message is counted as rejected when it fails the modeled identity, epoch, nonce, timestamp, route-digest, or HMAC checks before valid session state is accepted. The high rejection share illustrates the purpose of combining freshness state with route-bound authentication rather than relying on a timestamp or static secret alone. The remaining accepted or unrejected share represents the limitations of the simplified model and attack conditions; it must not be interpreted as proof that every insider, wormhole, or availability attack is detected. The result should therefore be read together with Table 3, which states the security-claim boundaries for black-hole, wormhole, DoS, and collaborative attacks.

Table 6 reports the central numerical comparison rather than depending solely on visual tendencies. For each attack, it presents PDR, delay, normalized routing overhead, authentication success, and forged-message rejection. The anticipated trade-off is evident: compared with unsecured DSR, DAK-DSR adds moderate delay and routing overhead, but it sustains substantially stronger modeled delivery and rejection performance when malicious traffic is present. DSR+Integrity occupies an intermediate position, since integrity protection blocks some forged traffic but does not implement the complete dynamic-key state machine. These figures are outputs of the stated Monte Carlo protocol-level model; they should not be represented as measured radio and CPU benchmarks.

The table uses the same 100-node, 20% malicious-node condition for all attack classes so that protocol differences can be compared under a common environment. PDR quantifies successful delivery, delay captures the performance cost of security processing, normalized routing overhead represents control cost, authentication success measures completion of valid endpoint authentication, and forged-message rejection measures the ability to discard manipulated control/authentication traffic. DAK-DSR consistently introduces more security processing than Standard DSR, but the modeled results

Table 6. Detailed results at 100 nodes and 20% malicious nodes across attack classes.

Attack	Protocol	PDR % (mean±SD)	Delay ms (mean±SD)	Overhead	Auth %	Forge reject %
blackhole	DSR	53.4 ± 3.6	13.6 ± 0.3	1.15	-	7.6
	DSR+Integrity	75.4 ± 2.5	15.0 ± 0.2	1.23	78.7	71.8
	DAK-DSR (Proposed)	91.3 ± 1.6	15.9 ± 0.2	1.30	97.4	96.6
wormhole	DSR	57.4 ± 2.5	16.1 ± 0.3	1.16	-	8.4
	DSR+Integrity	77.5 ± 2.1	17.5 ± 0.4	1.23	77.7	71.9
	DAK-DSR (Proposed)	91.9 ± 1.3	18.8 ± 0.3	1.30	97.6	96.2
mitm	DSR	59.6 ± 4.0	13.3 ± 0.2	1.16	-	7.5
	DSR+Integrity	78.3 ± 3.0	14.6 ± 0.3	1.23	79.1	71.9
	DAK-DSR (Proposed)	92.2 ± 1.6	15.6 ± 0.3	1.30	97.6	96.0
fabrication	DSR	55.1 ± 2.7	13.5 ± 0.2	1.34	-	8.1
	DSR+Integrity	76.1 ± 3.4	14.8 ± 0.3	1.28	78.9	72.2
	DAK-DSR (Proposed)	91.5 ± 1.5	15.7 ± 0.2	1.30	97.6	96.5
dos	DSR	59.9 ± 2.2	15.6 ± 0.4	1.35	-	8.2
	DSR+Integrity	78.4 ± 2.6	17.1 ± 0.4	1.29	77.7	71.8
	DAK-DSR (Proposed)	91.8 ± 1.9	18.3 ± 0.5	1.30	97.2	96.7
collaborative	DSR	48.6 ± 3.4	16.9 ± 0.5	1.35	-	8.1
	DSR+Integrity	72.7 ± 2.3	18.6 ± 0.3	1.29	78.0	71.8
	DAK-DSR (Proposed)	90.9 ± 1.9	19.8 ± 0.3	1.30	97.4	96.3

show substantially stronger authentication and forged-message rejection. The table therefore supports a security–performance trade-off rather than a claim of zero-cost security. The mean±SD values are based on the stated 30 repetitions and remain protocol-level Monte Carlo outputs, not hardware or packet-level radio measurements.

7.3. Quantitative interpretation

At 100 nodes with 20% black-hole attackers, DAK-DSR raises the modeled mean PDR from 53.4% (DSR) to 91.3%, which corresponds to a 70.9% relative increase in this model. Under 20% DoS pressure, the mean modeled delay shifts from 15.6 ms for DSR to 18.3 ms for DAK-DSR, representing a 17.9% increase. This directly supports the trade-off requested by the reviewer: stronger authentication and rejection improve delivery under attacks, but also introduce additional processing and control latency. At high malicious ratios, all schemes deteriorate because cryptographic endpoint authentication cannot ensure forwarding availability when many relays are malicious or colluding.

7.4. Statistical rigor

Each plotted point is the mean of 30 independent repetitions. The CSV file preserves repetition-level results, allowing confidence intervals, Welch ANOVA/t-tests or non-parametric alternatives after checking assumptions. For the journal submission, the authors should report 95% confidence intervals in the final tables and, if required by the journal, add effect sizes rather than relying only on p-values.

7.5. Boundary conditions

The model does not emulate PHY/MAC contention, real cryptographic CPU timings, radio battery discharge, or full DSR cache semantics. Consequently, the manuscript must not claim that these Monte Carlo values are NS-3 measurements. A packet-level replication in NS-3/OMNeT++ is recommended before making hardware-level latency or battery-life claims.

8. Practical Deployment Rules

Practical deployment requires more than the cryptographic equations. The following rules define how an implementation should behave when mobility, packet loss, clock drift, revocation, and partial state failure occur. They are intentionally explicit because the design follows the principle that routine operation should remain distributed and lightweight, while exceptional recovery may use a stronger bootstrap credential or administrative action [3, 15, 17].

a. Route errors:

A link break triggers standard DSR route maintenance/discovery. If S and D retain valid pairwise key state, the new accepted route produces a new RD and triggers a new DAK-DSR authentication/rekey; full bootstrap is unnecessary. Full re-bootstrap is required only after state loss, revocation, credential change, or recovery-window failure.

b. Time synchronization:

Timestamps are a secondary freshness signal. The protocol accepts |T_receiver − T_sender| Δ; the evaluation uses Δ=5 s as an explicit default assumption. Replay protection does not rely on time alone: epoch i, nonce cache and sid are mandatory. If clock synchronization is lost, the peer may fall back to epoch/nonce checks and request re-synchronization; it must not advance K_i solely because a timestamp is larger.

c. Nonce management:

N_S and N_D are generated from a cryptographically secure random generator, are at least 128 bits, and are cached by sid/epoch for the active session plus recovery window. Reuse within the accepted epoch is rejected. Nonce cache entries expire only after the session/recovery window closes.

Collaborative attacks: Colluding relays can still drop or tunnel packets; DAK-DSR does not overclaim complete prevention. Its contribution is that colluding intermediate nodes cannot forge endpoint authentication tags or valid key-update confirmation without the endpoint key. Routing availability against collusion should be strengthened with multipath/trust/timing detection if required by the deployment.

d. Deployment sequence and state protection:

Each node should store pairwise DAK-DSR state in protected local storage and associate it with peer identity, epoch, current key, pending key, previous key, route digest, nonce cache, and expiry timers. A node must never advance its epoch solely because it transmitted a key-update message; advancement occurs only after the confirmation rule is satisfied. When a route breaks, standard DSR repair/discovery proceeds first. If the endpoint relationship is still valid, the new route digest triggers a fresh authenticated update without requiring a new certificate exchange. If state is inconsistent beyond the recovery window, both peers discard pending state and return to authenticated bootstrap.

e. Revocation and compromise handling:

Revocation must invalidate both the bootstrap authorization and all active pairwise state associated with the revoked identity. A certificate-based deployment should distribute revocation information through the available MANET trust service; a pre-provisioned deployment can use an administrative revocation list or mission-specific rekey event. Because a fully compromised endpoint can expose its resident key, DAK-DSR does not claim post-compromise security without a trusted re-bootstrap source. The ratchet limits reuse of older authentication state after successful evolution, but endpoint capture

remains a separate system-security problem.

f. Clock and replay handling:

The timestamp window Δ is used as an additional freshness filter, not as the sole replay defense. Epoch and nonce/session caches remain authoritative even if clock synchronization temporarily degrades. An implementation should log repeated stale epochs, duplicate nonces, invalid HMACs, and excessive authentication attempts as security events and may rate-limit peers that exceed configurable thresholds. This also provides a practical interface to trust-based or learning-based detection modules [4, 16, 25].

g. Resource configuration:

Rekey frequency should reflect the application. A highly mobile or sensitive deployment may rekey after every newly accepted route, while a constrained network may use a packet/time threshold provided that route changes and suspicious authentication failures still trigger immediate update. Recovery window W_r should be short enough to limit old-key exposure but long enough to tolerate ordinary retransmission. These values must be reported in any future NS-3/testbed implementation so that results are reproducible and comparable.

9. Conclusion

This paper presented DAK-DSR, a dynamic authentication-key protocol designed to strengthen Dynamic Source Routing (DSR) in Mobile Ad Hoc Networks (MANETs). The protocol combines mutual challenge–response authentication with dynamic key updating and route-digest binding. It also defines the complete authentication-key lifecycle, including bootstrap, key updating, replay protection, synchronization, recovery, and revocation. The use of fresh nonces, session epochs, timestamps, HMAC-based verification, and the COMMIT/CONFIRM mechanism provides protection against replay, impersonation, message modification, and unauthorized key advancement.

The protocol-level Monte Carlo evaluation across 50, 100, and 200 nodes demonstrates that DAK-DSR can maintain higher packet delivery and forged-message rejection under active attacks, with additional delay, routing overhead, and energy cost representing a measurable security–performance trade-off. The results are protocol-level model outputs rather than hardware measurements. Future work will focus on NS-3 implementation, formal verification using Tamarin or ProVerif, and further investigation of adaptive key updating and complementary mechanisms for wormhole and collaborative-attack detection.

Acknowledgment

The authors are grateful to the Deanship of Scientific Research at Jadara University for providing financial support for this publication

REFERENCES

1. C. Wahi, S. Chakraverty, and V. Bhattacherjee, "A trust-based secure AODV routing scheme for MANET," International Journal of Ad Hoc and Ubiquitous Computing, vol. 38, no. 4, pp. 231–247, 2021, doi: 10.1504/IJAHUC.2021.119853.
2. N. Ghodichor et al., "Secure Routing Protocol against Internal and External Attack in MANET," THEETAS 2022, 2022, doi: 10.4108/eai.16-4-2022.2318163.
3. A. Alomari, "Mutual Authentication and Updating the Authentication Key in MANETS," Wireless Personal Communications, vol. 81, no. 3, 2015, doi: 10.1007/s11277-014-2169-1.
4. A. Alomari, "Fully distributed certificate authority based on polynomial over elliptic curve for MANET," International Journal of Networked and Distributed Computing, vol. 2, no. 2, pp. 70–77, 2014, doi: 10.2991/ijndc.2014.2.2.1.

5. M. Sirajuddin, Ch. Rupa, C. Iwendi, and C. Biamba, "TBSMR: A Trust-Based Secure Multipath Routing Protocol for Enhancing the QoS of the Mobile Ad Hoc Network," Security and Communication Networks, 2021, Article 5521713, doi: 10.1155/2021/5521713.
6. A. Annepu and P. Mishra, "Secure Aware Cluster-Based Routing For MANET Using A Multi-Objective-Trust Centric Flower Pollination Algorithm," International Journal of Engineering Trends and Technology, vol. 70, no. 1, 2022.
7. S. K. H. Islam, K. Parai, and D. S. Gupta, "PF-IBDA: Provably secure and pairing-free identity-based deniable authentication protocol for MANET environments," Computer Networks, vol. 238, 110113, 2024, doi: 10.1016/j.comnet.2023.110113.
8. J. Zhou, M. Luo, L. Song, and J. Hu, "A dynamic group key agreement scheme for UAV networks based on blockchain," Pervasive and Mobile Computing, vol. 95, 101844, 2023, doi: 10.1016/j.pmcj.2023.101844.
9. A. R. Patil and G. M. Borkar, "Node authentication and encrypted data transmission in mobile ad hoc network using the swarm intelligence-based secure ad-hoc on-demand distance vector algorithm," IET Wireless Sensor Systems, 2023, doi: 10.1049/wss2.12068.
10. G. Suresh, S. Suhasini, S. Benisha, G. Gnanakumar, and R. Manikandan, "Authentication based Associate Neighbor Selection for Reliable Routing in MANET," 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), IEEE, 2022.
11. S. Shafi, S. Mounika, and S. Velliangiri, "Machine learning and trust based AODV routing protocol to mitigate flooding and blackhole attacks in MANET," Procedia Computer Science, vol. 218, pp. 2309–2318, 2023.
12. A. Mahamune and M. Chandane, "Trust-based co-operative routing for secure communication in mobile ad hoc networks," Digital Communications and Networks, vol. 10, no. 4, pp. 1079–1087, 2024.
13. D. Sunitha and P. H. Latha, "A secure routing and black hole attack detection system using Coot Chimp Optimization Algorithm-based Deep Q Network in MANET," Computers & Security, vol. 148, 104166, 2025.
14. G. R. Rama Devi, M. Swamy Das, and M. V. Ramana Murthy, "Secure cross-layer routing protocol with authentication key management scheme for MANETs," Measurement: Sensors, vol. 29, 100869, 2023, doi: 10.1016/j.measen.2023.100869.
15. L. Rui, L. Zhao, Z. Guo, Z. Wang, X. Qiu, and S. Guo, "Mobile ad hoc network access authentication mechanism based on rotation election and two-factor aggregation," Computer Networks, vol. 254, 110826, 2024, doi: 10.1016/j.comnet.2024.110826.
16. M. Sasikala, S. Meera, A. Selva Priya, R. Durai Vasanth, B. Senthilkumaran, and J. V. Gripsy, "Deep Reinforcement Learning-based Mobile Ad Hoc Network for Secure and Dynamic Routing through Cryptographic Assurance," 2025 5th International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), IEEE, added to IEEE Xplore 2026, doi: 10.1109/ICERECT65215.2025.11377177.
17. H. Yang, "A Study on Improving Secure Routing Performance Using Trust Model in MANET," Mobile Information Systems, 2020, Article 8819587, doi: 10.1155/2020/8819587.
18. G. Vidhya Lakshmi and P. Vaishnavi, "An Efficient Security Framework for Trusted and Secure Routing in MANET: A Comprehensive Solution," Wireless Personal Communications, vol. 124, pp. 333–348, 2022.
19. G. Vidhya Lakshmi and P. Vaishnavi, "A trusted security approach to detect and isolate routing attacks in mobile ad hoc networks," Journal of Engineering Research, vol. 12, no. 3, pp. 379–386, 2024.
20. A. B. Feroz Khan, H. Lalitha R., K. Devi S., and C. N. Rajalakshmi, "A multi-attribute based trusted routing for embedded devices in MANET-IoT," Microprocessors and Microsystems, vol. 89, 104446, 2022, doi: 10.1016/j.micpro.2022.104446.
21. K. Nirmaladevi and K. Prabha, "A selfish node trust aware with Optimized Clustering for reliable routing protocol in MANET," Measurement: Sensors, vol. 26, 100680, 2023, doi: 10.1016/j.measen.2023.100680.
22. M. Vanitha, S. Jamuna Anand, V. Radhamani, and N. R. Rajalakshmi, "Dual-channel capsule generation adversarial network based blockchain technology for a secure dynamic optimal routing in mobile ad hoc network," Transactions on Emerging Telecommunications Technologies, 2023.
23. V. R. Sugumaran, E. Dinesh, R. Ramya, and E. Muniyandy, "Authentication mechanism based on distributed blockchain for secure and energy efficient mobile ad-hoc networks," Scientific Reports, vol. 15, 39732, 2025, doi: 10.1038/s41598-025-23406-z.
24. G. Dhand, M. Rao, P. Chaudhary, and K. Sheoran, "A secure routing and malicious node detection in mobile Ad hoc network using trust value evaluation with improved XGBoost mechanism," Journal of Network and Computer Applications, vol. 235, 104093, 2025.
25. S. M. Hassan, M. M. Mohamad, F. B. Muchtar, and F. B. Y. P. Dawoodi, "Enhancing MANET Security Through Federated Learning and Multiobjective Optimization: A Trust-Aware Routing Framework," IEEE Access, 2025.
26. A. R. Yadulla, B. Konda, M. Yenugula, V. K. Kasula, and C. Tumma, "Hybrid Deep Learning Technique for Cybersecurity Detection and Classification," Statistics, Optimization & Information Computing, vol. 14, no. 5, pp. 2518–2533, 2025, doi: 10.19139/soic-2310-5070-2491.
27. N. Azizi, A. Jamali, and N. Naja, "Comprehensive Study: Machine Learning and Deep Learning Approaches in Intrusion Detection Systems," Statistics, Optimization & Information Computing, vol. 15, no. 2, pp. 1416–1432, 2026, doi: 10.19139/soic-2310-5070-2782.
28. B. Ambudkar, S. Shirke, A. Bhanushali, R. Gupta, S. Shelke, M. Wagh, R. Pradhan, and A. Jadhav, "Enhancing Network Management using Machine Learning: Intent Based Networking," Statistics, Optimization & Information Computing, vol. 16, no. 1, pp. 896–909, 2026, doi: 10.19139/soic-2310-5070-3372.