dNextG: A Zero-Trust Decentralized Mobile Network User Plane

Ryan W. West
ryan@ryanwwest.com
University of Utah
United States

Jacobus (Kobus) Van der Merwe
kobus@cs.utah.edu
University of Utah
United States

ABSTRACT

Recent technological and regulatory changes are paving the way to enable decentralized, zero-trust mobile network models. Properly secured decentralization allows or improves "inherently distributed" use cases such as military coalition mobile networks distributed between ally infrastructures or community-infrastructure networks. While zero-trust security has been flagged by the U.S. NIST as critical to modern networks, the security threats associated with decentralized mobile environments have not been thoroughly studied and have mostly focused on distributing only the Radio Access Network (RAN) portion of the mobile network, potentially leading to unreliable Quality of Service (QoS) and low security in the network core. We therefore introduce dNextG, a mobile core network user plane that provides a zero-trust security monitoring framework to enable reliable decentralization even in the presence of malicious internal network nodes. With dNextG, both centralized and decentralized node operators can run User Plane Functions (UPFs) and Base Stations without giving up any node control; instead, the nodes maintain a permissioned blockchain that tracks the average reputation of each node according to tamper-resistant connectivity tests that they must periodically perform on each other. We identify various types of malicious node threats including dropping or modifying traffic and lying about reputation, and design dNextG to overcome these threats and provide a long-term, reliable QoS. We implement and evaluate dNextG to illustrate that its design overcomes these threats and satisfies NIST's Zero-Trust Tenets 5 and 7. We provide an open-source, instantly replicatable version of dNextG on the POWDER (Platform for Open Wireless Data-driven Experimental Research) platform.

1 INTRODUCTION

Mobile Wireless Networks have traditionally been controlled by large, centralized corporations and governments. Only these entities could afford to purchase spectrum or own and operate the vast quantities of specialized hardware required to run sufficiently large networks, which allowed them to mainly enforce network security at only the network edge. However, with the advent of 5G and looming threat of cybersecurity attacks, new technological and regulatory changes are allowing this traditional architecture to evolve. Technology changes include the 5G Core using Network Function Virtualization (NFV) to run all network services on inexpensive Commodity Off-the-shelf (COTS) hardware; the 5G Radio Access Network (RAN) supporting enhanced wireless technologies for dynamic, affordable Small Cell base stations and Network Slicing to allow adaptive use of hardware to suit different workflows [3]. Regulatory changes include the National Institute of Science and Technology (NIST) highlighting the need for Zero-Trust network security and the Federal Communications Commission (FCC)'s establishment of the Citizens Broadband Radio Service (CBRS) spectrum band that enables tiered spectrum sharing, including an "open" general-access tier [7]. These changes drastically lower the barrier of entry into the mobile networking ecosystem, encourage heightened intra-network security, and pave the way to enable decentralized mobile network models. A decentralized network distributes the decision-making authority throughout its nodes, and because it requires zero-trust security to function, offers many benefits such as increased robustness against attacks or errors, community resource pooling without giving up control, transparency of data and decisions, resilient network longevity, and more. Decentralized networks could thus provide significant benefits or enable use cases such as affordable, crowdsourced community-infrastructure networks and military coalition networks that combine network resources distributed across ally infrastructures.

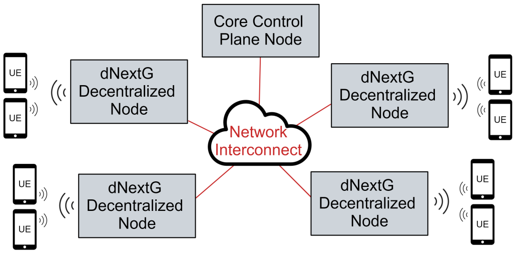

Core Control Plane Node
dNextG Decentralized Node
dNextG Decentralized Node
dNextG Decentralized Node
dNextG Decentralized Node
Network Interconnect
UE
UE
UE
UE
UE
UE
UE
UE

Figure 1: dNextG distributed over a (generic) network

However, managing the security of mobile decentralized networks, and even zero-trust centralized networks, remains an open problem. While standard, centralized LTE and 5G networks are equipped with encryption and authentication of both network users and servers, they are *not* designed to withstand malicious or non-functioning servers *that they do not control.* By definition, a decentralized network moves the control and trust from one centralized entity to many autonomous entities - thus, its architecture of servers or *nodes* is inherently controlled by various, independent people or groups. Even with the recent technological and regulatory advancements, attempting to decentralize and maintain a traditional network is infeasible, as without restraints, some decentralized nodes may cease to function correctly or act maliciously and thus wreak havoc on a network's Quality of Service (QoS) and correctness. Interest in decentralizing mobile networks is high, with successfully-deployed solutions such as Helium [12] and Pollen [1]

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Q2SWinet '23, October 30-November 3 2023, Montreal, QC, Canada

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0368-3/23/10…$15.00
https://doi.org/10.1145/3616391.3623427

already decentralizing the Radio Access Network (RAN) (which handles the actual wireless connectivity between base stations and end User Equipment (UE)) for hundreds of thousands of users. But even in these systems, the entire mobile Core Network—which handles user internet connectivity, authentication, mobility management, and more—still remains completely centralized and vulnerable to intra-node attacks. To combat these attacks, the U.S. NIST began focusing on "Zero-trust" frameworks that assume that there is no traditional network edge - rather than only monitoring for external threats, *every* node is treated like an adversary and is constantly monitored for threats [21], [13], [10]. In 2020, NIST released a publication emphasizing the need for Zero-Trust Architecture (ZTA) that defines seven tenets of Zero Trust, focusing on perimeterless security whereby all resources and communication regardless of origin must be dynamically authenticated and consistent monitoring of network security and QoS [22]. A 2021 Ericsson report applies these tenets to 5G technologies and states that "all owned assets in a telecom network should be monitored and their security posture should be evaluated continuously ... [including] RAN NFs [and] core NFs" [19]. While ZTA should be employed by all modern networks, it is quintessential for decentralized networks which are often connected *only* over public internet. These reports highlight the need for better zero-trust security in future mobile networks.

In response, this paper presents our work on dNextG, a decentralized mobile network *core user plane* architecture that provides a zero-trust security monitoring framework between decentralized network nodes. To our knowledge, dNextG is the first work to specifically focus on the user-plane security of a decentralized mobile network, but also works with a centralized mobile network to improve its zero-trust security. The user plane comprises all user-requested traffic including internet and video streaming, typically at well over 90% of all network traffic [11]. In dNextG, many independent node operators form a large, cooperative network that runs the user plane, but each operator retains complete control of their servers. To allow this, a *network-wide reputation system* is used to ensure high availability, reliable QoS, and correct, honest behavior of each operator node.

While any dNextG mobile network uses 3GPP technologies such as Service-Based-Architecture (SBA) and the Policy Control Function (PCF) to enforce all of NIST's ZTA tenets, dNextG specifically adds security monitoring capabilities to support tenets 5 ("The enterprise monitors and measures the integrity and security posture of all owned and associated assets") and 7 ("The enterprise collects as much information as possible about the current state of assets, network infrastructure and communications and uses it to improve its security posture") [22]. dNextG fills a critical gap in the monitoring of both centralized and decentralized zero-trust networks, allowing network operators to quickly identify and correct both threats and benign malfunctions in the Core user plane and even within the security monitoring framework itself. Decentralizing the network also further lowers the financial barrier to entry, as communities can now crowdsource to form a large mobile network. dNextG allows anyone with as little as one server to join an existing network and profit as an "operator". Users could pay far less to use a dNextG network compared to commercial mobile networks, when they are even available, and even provide other benefits such as low latencies with Local Breakout (LBO). The dNextG core user plane complements existing decentralized RAN solutions (e.g., Helium or Pollen [12], [1]), thus providing another building block towards a complete "zero-trust" distributed mobile architecture.

To this end, each dNextG node runs a User Plane Function (UPF) and possibly a base station to decentralize the entire network user plane. While the core network's *control plane* remains centralized, almost all traffic is handled exclusively by the decentralized nodes. To maintain a stable network QoS, nodes must consistently test a verifiably random subset of each other's behavior by initiating anonymous psuedo-mobile-user connections through other nodes and sharing the connection success/failure results. A shared, immutable blockchain ledger stores and safely distributes each node's observations of its peer nodes, acting as a public reputation monitor for all nodes and users. Prior works prove that by using a Byzantine Fault Tolerance (BFT) consensus protocol, the blockchain state remains correct and unhindered as long as a certain majority (commonly two-thirds [6]) of the network operates honestly [1], allowing all honest nodes to provide services even in the presence of uncontrollable, malicious nodes. Users can monitor the state to select which nodes carry their network connection, and can choose low-latency local breakout nodes and avoid unreliable low-reputation nodes. This incentivizes high reputation and correct behavior, as node operators' profit is directly tied to the amount of traffic they handle. The high-level architecture of dNextG is shown in Figure 1.

We make the following contributions:

1. We introduce the first design of a zero-trust, decentralized mobile network core user plane monitoring system that protects against various threats by having distributed nodes contribute to a blockchain to realize a network-wide reputation system. Users can avoid low-reputation nodes and instead choose local, low-latency and/or high QoS nodes.
2. We develop a prototype implementation of dNextG that realizes a fully functional 5G network.
3. We evaluate our design to illustrate that dNextG addresses the threat model associated with the user-plane of a distributed 5G network core at scale.
4. We provide an open-source, instantly reproducible dNextG instance in the POWDER platform that executes on real hardware. (We will also release the dNextG prototype software as open source as part of this environment.)

2 BACKGROUND

2.1 Mobile Networks

Mobile Network processes and traffic are logically categorized into the control plane, which includes all network management processes, and the user plane (or data plane), which handles all user-specific data such as internet-related traffic. Mobile networks are physically split between the Radio Access Network (RAN) and the Core. The RAN consists of base stations (e.g. radio towers or city-block Small Cell devices), equipped with antenna and hardware capable of sending and receiving wireless signals) and User Equipments (UEs) (e.g. smartphones, cars, Internet of Things (IoT) devices) that wirelessly connect to the mobile network through

[1] Many BFT algorithms are designed and proven such that up to one-third of the nodes running them can be malicious yet cannot inhibit correct blockchain operation; however, this ratio is parameterizable and depends on the algorithm.

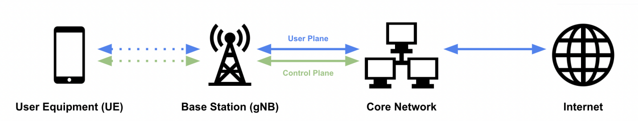


Figure 2: Mobile Network Architecture

base stations. In contrast, the mobile network core refers to the backend network servers which carry out critical network processes (which traditionally required specialized hardware but are now typically realized through Virtual Network Functions (VNFs) on COTS hardware).

In a 5G Core, the Unified Data Management (UDM) stores network state, such as UE subscriber data, and the Authentication Server Function (AUSF) uses this data to authenticate newly connected UEs. The Access and Mobility Management Function (AMF) serves as the entrypoint for base stations (known as gNBs) to connect to the core and handles key functions such as initiating new UE connections. The Session Management Function (SMF) creates and manages Packet Data Unit (PDU) sessions on behalf of UEs to provide them internet connectivity. The Network Repository Function (NRF) tracks all VNFs' IP addresses similar to a DNS server. Other VNFs handle metrics, discoverability, billing, routing texts and calls, interfacing with external mobile networks, and more.

The only VNF that handles user plane traffic is aptly named the User Plane Function (UPF), and interfaces with the SMF and gNB to route traffic between UEs and the internet via PDU sessions. PDU sessions are sent via GPRS Tunneling Protocol (GTP) tunnels which encapsulate user plane packets between the UE and UPF.

The UPF also acts as a Network Address Translation (NAT) layer by managing UE IP addresses and working with the core control plane to migrate PDU sessions to other UPFs when necessary. While the control plane is quite complex, its operations are typically completed in short bursts, and most network traffic thus belongs to the user plane. This often requires more UPFs to be deployed than control plane VNFs to handle the larger traffic load, and often in geographically diverse areas.

2.2 Blockchain and Trust

A blockchain is an immutable, distributed database of transactions that seeks to achieve decentralized consensus of all data published to it among untrusted groups, without third party management. Blockchain offers data provenance: any attempts to modify previously added data can be immediately detected by all clients and blockchain nodes (known as validators). Validators maintain copies of the ledger and may add data in transactions grouped into blocks. New blocks are broadcast between network validators and then collectively, atomically added by a majority of the validators at approximately the same time. To detect if blocks have been modified, each block contains includes the hash of the previous block added to the ledger. If any block were changed, that block's hash, present in the next block in the 'chain', would no longer match, breaking the chain of matching block hashes. This guarantees strong immutability but requires that the ledger be fully readable for users to validate that it has not been tampered with.

A consensus protocol allows a group of validators to share existing blocks and agree to add new blocks. It consists of a series of processes in which validators share new blocks, agree if and when to add them to the ledger, manage incoming and departing validators, and maintain network order. These protocols vary significantly by implementation but are critical to providing a reliable root of trust—rather than trusting one entity, users rely on the protocol itself being enforced across all validators, which is typically provably true as long as a majority of validators act correctly.

Blockchains can be categorized by use into three different types: public, consortium, and private: (i) On a **public** blockchain (e.g. Bitcoin), anyone can anonymously write data to the ledger, read the entire ledger, and operate a node that participates in consensus. (ii) A **private** blockchain is the opposite—it is accessible only internally to the company that runs it, which controls costs and technical details. (iii) A **consortium** blockchain contains both public and private components—it typically allows anyone to read and possibly write transactions to the ledger, but restricts validator participation to a set of known operators. Authority is distributed between the node operators and leadership is often conducted through a committee of validator operators. With a limited, known number of nodes participating in consensus, this allows the blockchain to use more specialized, secure consensus protocol families such as BFT and achieve higher speeds.

3 USE CASES

An entity or group of entities with access to different resources have the common goal of creating an operational 5G network, but may not fully trust every participating node and strive for zero-trust security. They may also need to spread costs between multiple entities. For both of these problems, dNextG can help. By requiring independent nodes to verify each other's behavior constantly, misbehaving nodes can be automatically detected and routed around by the network via reputation reports, preserving long-term network correctness and reliability. We present two example use cases:

(1) **Coalition Military Network.** A nation may need to deploy and share a mobile network with an ally nation to jointly conduct tactical missions. The host nation may not have control over all infrastructure, especially if a mission occurs in the ally nation, requiring trust in ally-owned-and-operated infrastructure. dNextG allows the host nation to prevent the ally or other forces from taking over the network by running the Core control plane and ensuring no other single entity can control more than a minority of the participating nodes. Anomalous internal behavior by any nodes (allies or self) can be detected and routed around or corrected. This can be extended to any single, centralized entity (e.g., company, government, etc.) concerned about internal security threats. dNextG's zero-trust architecture requires all nodes to be constantly evaluated regardless of owner; thus, internal node problems can be detected and routed around until corrected.
(2) **Decentralized community.** A group of individuals may each have a computer and possibly a base station and wish to retain hardware control, but also profit off of their equipment by offering a collective mobile network service. Currently-deployed, crowdsourced mobile networks such as Helium and Pollen already run hundreds of thousands of decentralized base stations, showing the demand for this use case. dNextG allows these individuals to also

collectively form an entire zero-trust user plane for a mobile network. Combined with a decentralized RAN such as Helium/Pollen, dNextG could be used to also compensate highly reputable nodes based on the amount of user traffic they successfully deliver. A trusted entity would run the network core control plane VNFs, but all user plane traffic would flow through the decentralized community nodes. This system could be especially beneficial in remote areas or third-world countries where existing mobile connectivity options may be expensive and/or limited.

4 THREAT MODEL

dNextG's key use, a decentralized core network user plane, brings with it the threat of undesirable node behavior. Without a centralized actor to monitor or correct malicious or accidental bad behavior, this exacerbates the already difficult challenge of maintaining a performant network with reliable QoS. Even with a security monitoring system in place, that system could itself be compromised, and the same is true for nodes in a centralized network. In response, the dNextG security monitoring system has been designed, implemented, and evaluated to support NIST's zero-trust Tenets 5 and 7 [22] by protecting against the following categories of threats:

(1) **Dropping or modifying network traffic**. A node may drop or change its UPF's user or control plane packets. As long as user plane traffic is end-to-end encrypted with, for example, HTTPS/TLS, any modifications to this traffic can be detected. dNextG also detects dropping traffic, as reputation tests are designed to be indistinguishable from real user traffic, and any dropped traffic can cause a test to fail and decrease reputation. While a node can reconfigure its VNFs, the Core decides whether to accept these changes and route user plane traffic to it. Dropping or changing control plane packets resulting in incomplete UE attachment procedures or internet connectivity causes reputation test failures. The remaining network can also blacklist the node if necessary.
(2) **False reputation reports**. Nodes testing other nodes can publish false reputation reports. However, they can only publish on behalf of themselves, and because the ledger is transparent and immutable, reports that consistently fail nodes regardless of their behavior can be detected as anomalies compared to honest reports. Nodes must also respond to user data queries and may lie about the blockchain state; however, clients can simply cross-compare the reported blockchain results from several nodes to mitigate this.
(3) **Node Collusion**. A set of dishonest nodes could always pass each other's tests to quickly gain reputation. dNextG mitigates this by rejecting reputation reports of tested node subsets that were not generated in a verifiably random way, making it very difficult for nodes to select which other nodes they get to test and thus making it difficult for collusion to have an impact. dNextG also rate-limits report submissions, preventing a node from submitting a burst of reports to boost (or drop) another's reputation.
(4) **Failing to report reputation**. nodes may stop publishing new reputation reports, which is bad as the averaged reputation of each node is only accurate if it is current. dNextG counters this by requiring reports to be regularly reported—a node with an insufficient number of recent reports automatically has its reputation lowered.
(5) **Abusing reputation-testing credentials**. A node may abuse its temporary credentials for free data connectivity, but dNextG counters this by making said credentials extremely short-lived and rate-limits their issuance.
(6) **Identifying and allowing reputation test traffic, but not regular traffic**. A malicious node may try to detect which traffic belongs to a reputation test to only allow that traffic through. However, reputation test data is identical to all other user plane traffic in terms of content and structure, as the simulated-UE portion of the stack ends well before the malicious node's UPF ever sees the packets. The malicious node could still try to identify patterns that distinguish reputation test packets from other packets such as timing frequencies or destination URLs, which dNextG counters with rules about random test subject selection and timing.

5 dNextG DESIGN

A dNextG mobile network is composed of a group of computers or *nodes*, each responsible for running various VNFs and/or base stations. Most nodes are decentralized and form the core user plane by running a UPF and the RAN by potentially running a base station. Separately, a centralized node or nodes n_c operates the control plane VNFs of the network core. dNextG does not emphasize better performance than a traditional network, but instead better reliability and new use cases involving many untrusted actors. Despite this, decentralized nodes need not surrender any control, only needing to expose the network ports required for their UPF and base station to communicate. That being said, what exactly incentivizes nodes to operate reliably? The answer is **reputation**. Reputation allows the network to evaluate the performance and honesty of its members and is based on a node's ability to fulfill user plane network services. As the decentralized nodes have no leader, they must constantly test each other's abilities and share their observations. Nodes are specifically tested on the ability of their UPFs to create and maintain stable PDU internet connectivity sessions.

5.1 Reputation Report Generation

A decentralized node n_t wishing to perform a reputation test first selects another node to challenge n_s (the method of selection is covered later). n_t creates a pseudo-UE (a software-simulated UE) by requesting the control plane node n_c to create and send over ephemeral subscriber credentials, Data Network Names (DNNs), and other data that real UEs typically store in Subscriber Identity Module (SIM) cards and non-volatile memory. This data includes information such as an International Mobile Subscriber Identity (IMSI) number, cryptographic keys for authentication, preferred Data Network Names (DNNs), and Network Slice Selection Assistance Information (NSSAI). As this information is valid for only a varying amount of seconds and regenerated per test, n_s cannot identify or distinguish psuedo-UEs from real UEs over time, and n_t cannot abuse these credentials for meaningful free data transfer as n_c limits the frequency at which testing credentials can be requested.

Once n_t has valid temporary UE credentials, its psuedo-UE connects to the network through a base station application's open network ports. If n_t uses its own base station, no other node will know that the psuedo-UE is virtual. n_t may also connect its psuedo-UE to another node's base station that is not n_s; however, this will reveal to that node that the pseudo-UE is virtual as the protocol stack reveals that some Physical Layer activity is simulated. Subsequent control plane communication (indistinguishable between real and psuedo-UEs) flows between the base station and n_c to authenticate the UE to the network; set up a PDU session based on, for example, the chosen DNN or network slice; and select the correct UPF for that session. n_t specifies that the PDU session should go through the UPF of n_s by sending over its preferred DNN or NSSAI during the initial handshake between the psuedo-UE and n_c. n_c communicates this to the UPF at n_t as a part of PDU session setup.

After a PDU session is successfully configured, n_c informs the base station which UPF the psuedo-UE's user plane traffic will flow through. Subsequently, most traffic flows between n_t's psuedo-UE, the base station, the UPF, and the internet. At this point, n_t must simply verify that its psuedo-UE can securely connect to a trusted internet website using Transport Layer Security (TLS) (and n_t should psuedo-randomly vary which website it verifies per test to lower detectability by n_s). TLS and verifying Certificate Transparency Logs [14] are necessary to ensure that n_s is not masquerading as the external website, as TLS requires the psuedo-UE to verify that website's public certificate, which is infeasible to spoof with Certificate Transparency Logs. If the psuedo-UE of n_t can successfully connect to an external website using TLS through the UPF of n_s, then n_s passes n_t's reputation test. Otherwise, n_s fails the test. Example reasons could include server downtime for n_s, attempts by n_s to interfere with the connection, n_s selectively dropping traffic to save bandwidth, and network congestion/partitions. While some reasons (e.g., node dishonesty) are more severe than others (e.g., temporary server unavailability), if n_s cannot satisfy n_t's test, it fails[2]. Implementations may also report the failure reason to categorize bad behavior. n_t periodically repeats this operation for a verifiably random subset of other nodes, which is explained further below. It then combines a current timestamp and the results of every tested node into a transaction and submits it to the dNextG blockchain.

Blockchain data is *immutable*, preventing any node from changing its reported tests over time and thus discouraging nodes from falsifying reports which could be recognized as anomalous. dNextG is designed to use a consortium BFT consensus algorithm which operates the overall blockchain correctly amidst malicious nodes[3]. The number of malicious or unhealthy nodes that a network can tolerate varies by algorithm, but the popular PBFT (Practical BFT) family of algorithms has formally proven that up to one-third of participating nodes may act improperly and the remaining network can still correctly read to and write from the blockchain ledger as usual [15] [6]. BFT algorithms also generally allow orders-of-magnitude higher transaction throughput than Proof of Work (e.g., Bitcoin) as block confirmation time is very shot. By using an algorithm such as PBFT, dNextG inherits these properties.

Each decentralized node maintains a copy of the current blockchain ledger using a *validator* program. The validator stores blocks, validates and packages new reports into transactions that are batched into new blocks, gossips newly created blocks to other nodes' validators, verifies the authenticity of newly received blocks, and participates in consensus in order to collectively agree whether to add a new block. n_t may only publish reputation report transactions authored by itself; if n_t's validator tries to publish a report that purports to originate from n_s, honest node validators will detect and reject it, thus stifling a majority vote and leaving the blockchain state unharmed. dNextG makes this possible by requiring each validator to sign its reputation report using its private key and include its public key in that transaction's header. All other node validators can use the public key to verify that the transaction signature was signed with the owner's private key and is thus authorized and non-repudiable. If n_t wanted to maliciously publish false reputation reports acting as n_s, it would need to obtain n_s's private key, which is infeasible with proper security procedures in place.

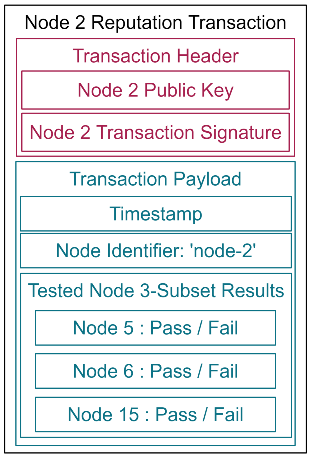


Figure 3: Reputation Transaction

The blockchain also allows dNextG to both scale and enforce regular, evenly-distributed reputation reports across all nodes. As mentioned previously, n_t must select a subset of peer nodes to test based on the hash of the most recently-committed block concatenated with a known, unique identifier, which are both included in the transaction that n_t later submits. As used in [12], a block hash can be used as a random seed to provide a verifiable, deterministic source of randomness. Adding the unique identifier maintains determinism while allowing different nodes that use the same seed block to select a different subset of nodes. n_t uses the random seed to pick a subset of 3 of its peer nodes[4] at the beginning of every testing period, which produces an even distribution of nodes for n_t to test. n_t must submit a transaction within 8 minutes of the seed block's hash and at least every $2N$ blocks, where N is the number of decentralized nodes, or else its reputation will be halved.

This has four benefits. First, a 3-node subset allows the reputation system to scale linearly, as each node's testing burden does not increase with more nodes. Second, this inhibits collusion or targeting other nodes with false tests, as validators can reject transactions if the subset of nodes was not actually provably random; malicious nodes have little opportunity to pick which nodes they report on. All nodes will, on average, receive equal amounts of tests that increase (or decrease) their reputation. Third, consistently-timed

[2] Nodes could also employ other means whereby to judge their peer nodes, such as reporting poor reputation as a result of malicious behavior.

[3] dNextG does not necessarily require a BFT algorithm; other consensus algorithms may be used. However, the consensus algorithm in question should be chosen with care in order to provide sufficient security guarantees for who can write to the blockchain and the type of supported fault tolerance.

[4] This could also be dynamic according to N, but the minimum size of N is 4 so 3 is the largest number guaranteed to work for all network sizes.

reputation reporting is enforced, as validators can easily detect nodes that do not follow reporting rules and halve their reputations. And fourth, nodes being tested are even less likely to succeed at detecting a test as the frequency at which they are tested becomes random and unpredictable (especially if the testing nodes also vary the interval at which they test). A malicious node could still attempt to select which 'latest' block hash it will use, testing likely up to N of them in order to pick which subset of random nodes it prefers to test the most. A 3-node subset is thus chosen because 3 nodes reduces the effectiveness of this strategy, because less than 3 nodes per transaction would result in slower reputation reaction to behavior (because there are fewer tests overall), and because 3 is the maximum subset size in the minimum-sized network (4 nodes).

The validator also prevents reputation spamming by only allowing one transaction of 3 tests per node per N blocks. With minimum and maximum timing requirements enforced for reputation report submission, the network reputation state stays current without exploding the ledger size.

5.2 Optimizing UE Routing via Reputation

Reputation reports from n_t and other nodes are averaged together to provide an overall reputation for each node ranging from 0 to 10. High averaged reputation nodes generally fulfill their UPF's routing services, whereas lower reputation implies lower reliability. The last 100 reports for every node are averaged, and then these averages are themselves averaged across all reporting nodes to produce a single network-wide reputation state for each node (which is halved for nodes that do not follow the reporting frequency rules).

Clients (such as UEs) may download transactions to analyze from the decentralized nodes via their blockchain validators, or simply rely on the validators to compute reputations to save time and bandwidth. Transactions are typically only a few hundred bytes in size and downloading new ones can be infrequent, so the overhead of downloading blockchain data is minimal. The client can optionally request the same data from several nodes to verify the consistency of the results. Under a correctly-operating BFT blockchain, if the UE contacts more than one-third of the network nodes that all deliver the same response, then that response will be correct [6] (but a high confidence of correctness can still be obtained by contacting even two nodes).

UEs can use reputation and other network information collected by dNextG (e.g. inter-node latencies, self-reported/ISP capacities, specific Quality of Service (QoS) level) to select the best UPF node according to their own parameters. Running UPFs locally offers low-latency Local Breakout (LBO), which recent measurement studies have shown public Mobile Network Operators rarely offer [2]. One UE may choose a UPF node with low latency and high QoS with only medium network reputation, while another might value high reputation over latency. The UE reports its choice to n_c which routes the PDU session through the UE's chosen UPF. The UE periodically refetches the current reputation state and may select a new UPF if the current one loses too much reputation, the connection drops repeatedly, or other network factors change. The UE may back up past reputation values in case the current UPF availability decreases and the network becomes unavailable for fresh lookups.

The SMF, run by n_c, chooses which UPF will handle a particular UE's current PDU session. It is assumed above that n_c simply honors all UE's valid UPF routing requests (as long as they are valid—e.g., requesting a slice with a higher QoS than it is not paying for should be denied), but it may alternatively choose the best routes through the network for UEs by also periodically fetching and analyzing the current network state. Because n_c decides whether to grant the UE's UPF preference, this option may be more realistic. While n_c could conceivably route user plane traffic from one node's base station to the same node's UPF for the lowest latency, n_c operators should disallow this because it gives significant, unmonitorable power to that node, reversing many advantages of decentralization that military coalitions or security-sensitive entities are hoping to benefit from.

The auditabilty of all past report history also allows reputation to be evaluated by other metrics such as frequency of reporting. As previously described, nodes must consistently evaluate each other and distribute their reports to maintain a high accuracy of the current reputation. Evaluators must simply check the frequency of reports to see if a particular node is reporting as expected, and halve that node's reputation if not. They can also look for anomalous or inconsistent report history in one node compared to the others. Transparent ledgers allow the evaluation method to vary from one user to another, as everyone can access the entire history of reports and decide which node has the best reputation for *their* needs.

5.3 Governance and Incentive Model

New operators may wish to join an existing dNextG network. Consortium blockchains typically track node membership on the ledger and require a majority of existing nodes to agree to add the new node as a validator or make other changes via consensus and governance committee. The difficulty level of the node approval process can be made stringent (e.g., require proof of location/identity) to minimize the possibility of malicious operating nodes. In such a system, Sybil attacks require an adversary to first convince a majority of the existing operators that their pool of potential nodes are actually each independently controlled and operated by real, trustworthy people. This system could also monitor or operate n_c (e.g., to ensure UE-UPF routing requests are honored or appropriately chosen for UEs when applicable) and vote to take action or switch to a new n_c if problems arise. dNextG does not provide a secure decentralization design for the RAN, which other related works have already explored [12], [1]. If a more secure RAN is desired, the network operators should combine dNextG with existing methods such as Proof of Coverage and trusted code execution.

In decentralized communities, operators may wish to profit directly off of their nodes' services, which incentivizes them to maintain a high reputation to handle more traffic. RAN-based decentralized networks like Helium and Pollen handle this by allowing users to purchase data credits, tracking which nodes handle said users' data routes, and transferring said data credits to said node operators which can then be exchanged for fiat currency [12], [1]. The dNextG blockchain could also track these data credits and reward users on both traffic load and reputation testing; however, we defer the actual design of this to future work because decentralized RAN and core user plane incentives should be directly integrated. dNextG alone remains valuable when zero-trust security is prioritized over direct node profits, such as in coalition military networks or security-sensitive entities.

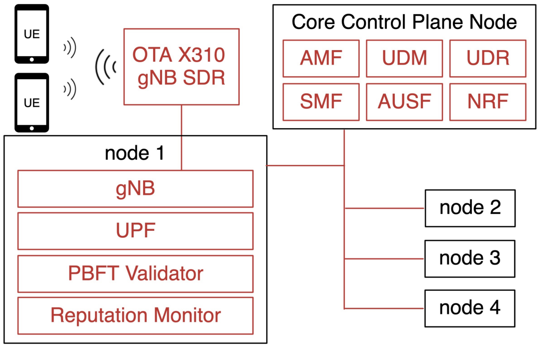


Figure 4: Example implemented dNextG architecture with one real gNB attached and three simulated gNBs

6 dNextG IMPLEMENTATION

We have implemented an instantly replicatable version of dNextG on the POWDER platform [5] which includes a complete 5G Core and RAN with OpenAirInterface (OAI). It uses the Hyperledger Sawtooth blockchain with the Plenum Byzantine Fault Tolerance (PBFT) consensus protocol to ensure that a two-thirds majority of honest, functioning nodes operates correctly [6]. Sawtooth transactions are categorized by family to be handled differently depending on their contents. A 'dNextG' family transaction processor validates and aggregates each node's reputation reports at a unique blockchain address writable only by that node. This implementation maps a unique DNN to each UPF to allow UE route selection, which UEs provide when connecting to the AMF. This gets forwarded to the SMF which ultimately handles UPF route selection.

PBFT requires at least 4 nodes[5], which each run a UPF, gNB (5G base station), PBFT validator, and dNextG reputation monitor program as shown in Figure 4. A set of N decentralized nodes are each identified from 1 to N. One additional node runs the 5G Core Control plane, including the AMF, SMF, AUSF, NRF, UDR, and UDM. Both this node and UEs monitor and enforce UPF routing based on reputation, but only the UE realizes this in our implementation. This node can also optionally be used to simulate a real UE's connection, monitoring of and adapting to average reputation reports, and data usage through any of the other nodes' gNBs and UPFs.

Nodes run Ubuntu 18.04 typically on Xeon 8-core processors with 4GB RAM. The 5-or-more-node topology, installation scripts, dNextG programs, and all other configurations are stored in a Git repository registered as a POWDER profile.[6] Simulated UEs are used by default, but the profile is tested with real gNBs (National Instruments X310 SDRs each with a UBX-160 daughter card wideband transceiver) and UEs (Intel NUCs with Quectel RM500Q-GL 5G Modules/SIM cards). Running simulated UEs for purposes of short-lived reputation tests has negligible impact on the decentralized nodes and their gNB processes, with typically under 3% CPU utilization, which lines up with OAI performance measurement studies such as by Nakkina et. al [17]. Over-the-air connectivity has been tested in the 3550-3600 MHz CBRS spectrum range.

A node's reputation monitor program by default waits 3 minutes between testing UPFs. Because it takes time to generate each UE, connect it to the network, test a UPF, and send the reputation results to the PBFT validator, each node publishes a new transaction every 4-5 minutes. Each individual transaction includes, for its subset of 3 tested nodes, boolean value True if a node's UPF passed the test and False if it failed, the node id *nid* of node issuing the tests, and the block number of the block hash used as a random seed to verifiably generate the set of tested nodes. This is shown in an example 17-node setup in Figure 5. Sawtooth can be queried by transaction, block, or address. Each node's address state is entirely derived from block transactions specifying that address and reflects that node's current observations of the remaining nodes' average reputations. Reputation is rated from 0-10 where 10 represents high reputation, as shown in a 17-node setup in Figure 5 ('true' or 'false' indicating a passed or failed test). The current reputation for each node is halved if that node is not regularly submitting reports. UEs may use this validator-calculated reputation or fetch the transactions themselves to determine their ideal network route. UEs also select the next highest reputation node's UPF in the event that the overall highest reputation node UPF route repeatedly fails.

When a transaction is uploaded, the validator forwards it to the 'dNextG' family transaction processor for validation. The public key and signature of the reputation report payload are included in the transaction for identity verification to ensure nodes cannot write to other node's Sawtooth addresses. Once this and the payload are validated, the transaction is included in a new block. Each node exposes their validator through an API that the core control plane node and UEs can use to access all reputation states transactions.

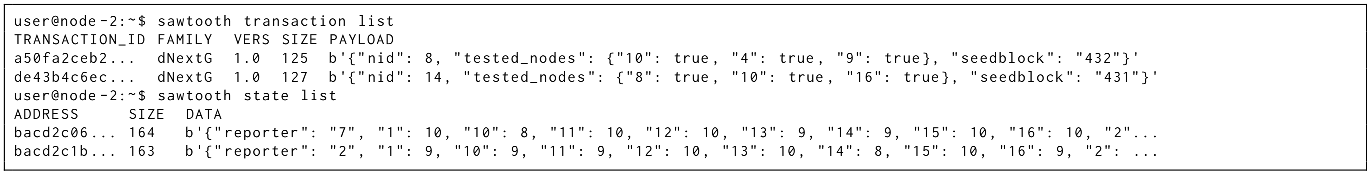
7 EVALUATION

We evaluate the performance and resistance to security threats of dNextG. The following evaluations were performed on dNextG POWDER experiments by manually controlling the processes of each node with various scripts, thus allowing us to simulate each type of malicious behavior from the threat model.

Under normal operation, a node's UPF passes other nodes' reputation tests ~98% of the time and holds average reputations between 9.75 and 10. UEs successfully connect to the internet through their chosen PDU sessions 98% of the time. We first demonstrate the nodes' reputation state in a 4-decentralized-node experiment by starting all nodes' PBFT validators, gNBs, and VNFs with a default reputation of 5 and connecting several UEs. This allows all nodes' UPFs to fulfill PDU sessions and pass reputation tests; quickly increasing the average reputations of all 4 nodes to 10, as illustrated in the beginning of Figure 6. The figure shows a three-hour period of the averaged reported reputations, where each node submits a new report every 4-5 minutes.

If a node's UPF ceases to function (due to malicious motive or malfunction), existing PDU sessions drop and affected UEs must reconnect through another UPF. However, new UE PDU sessions may continue to be routed through the faulty UPF and fail until reputation tests decrease the node's reputation, which occurs within 4 minutes (though a client UE will switch to a different UPF upon failures much sooner than this). We illustrate this by turning off node 4's UPF after 80 minutes, and its average reputation soon

[5] A 4-node network tolerates up to one malicious node before honest validators can no longer reach consensus to add to the blockchain ledger.
[6] This profile is available at https://github.com/ryanwwest/dNextG and can be instantiated, automatically deployed on a configurable number of physical nodes, and interacted within minutes to experiment with and replicate our evaluation.



```
user@node-2:~$ sawtooth transaction list
TRANSACTION_ID FAMILY  VERS SIZE PAYLOAD
a50fa2ceb2...  dNextG  1.0  125  b'{"nid": 8, "tested_nodes": {"10": true, "4": true, "9": true}, "seedblock": "432"}'
de43b4c6ec...  dNextG  1.0  127  b'{"nid": 14, "tested_nodes": {"8": true, "10": true, "16": true}, "seedblock": "431"}'
user@node-2:~$ sawtooth state list
ADDRESS     SIZE  DATA
bacd2c06... 164   b'{"reporter": "7", "1": 10, "10": 8, "11": 10, "12": 10, "13": 9, "14": 9, "15": 10, "16": 10, "2"...
bacd2c1b... 163   b'{"reporter": "2", "1": 9, "10": 9, "11": 9, "12": 10, "13": 10, "14": 8, "15": 10, "16": 9, "2": ...
```

Figure 5: Sawtooth Example Transactions and State (output simplified)

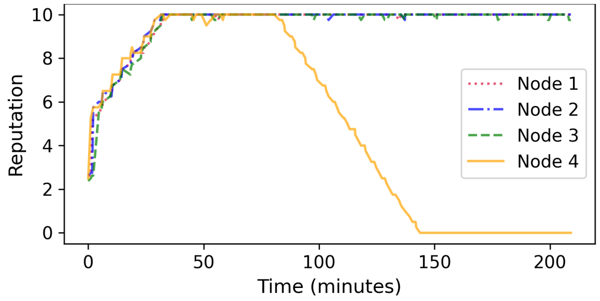


Figure 6: Average reputations when node 4's UPF fails

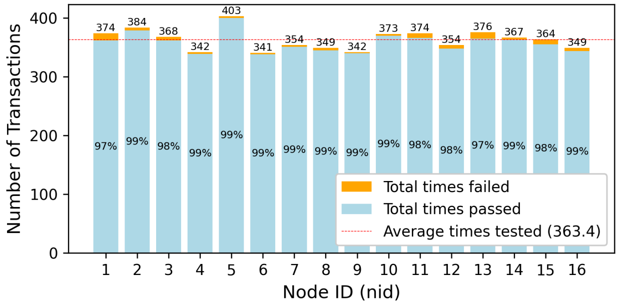
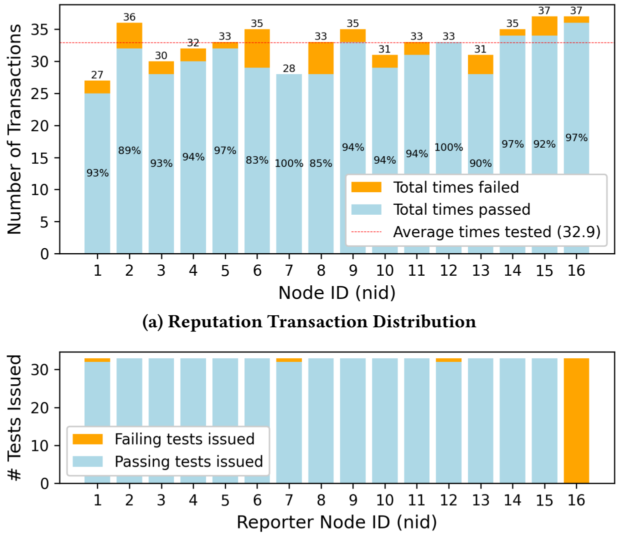


Figure 7: Reputation Transaction Distribution By Node



(a) Reputation Transaction Distribution

(b) Reputation Tests Issued By Node

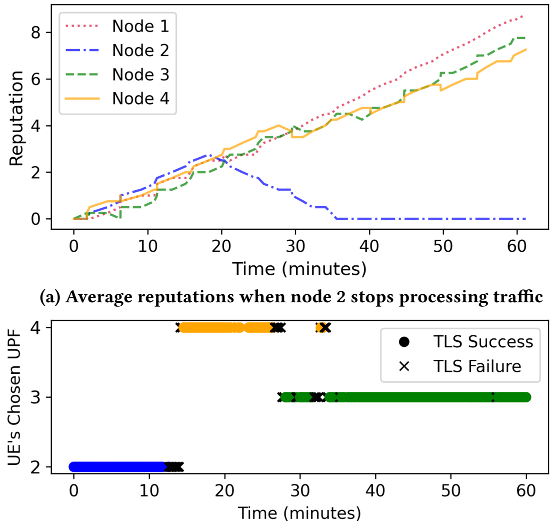
Figure 8: Reputation Tests When Node 16 Lies

drops. This may also occur if it tries to drop or modify user or control plane traffic in other ways since that traffic may belong to a reputation test.

To test fairness and randomness in node test selection, we next instantiate a 16-decentralized-node experiment and measure the distribution of how many times each node was tested and reported in a reputation transaction over a 12-hour period, as shown in Figure 7. The results show, in addition to that the network provides a reliable QoS to its users, that the 3-node random subset that each tester node must calculate produces a relatively even distribution—all nodes have a fair chance to be validated, and attempts by malicious nodes to select their own node subset to test are rejected by the Sawtooth validators. The same results have been tested with 100 nodes and we also observe that the effort and time expended per node to test and report reputation is scalable, as it remains constant with network size (on average, 250MB RAM and 50% of one CPU core to run a gNB, 125MB RAM and 80% of one CPU core for the pseudo-UE, and negligible resources for remaining dNextG processes). Importantly, the data also reveals that the interval in which a particular node UPF receives tests is highly variable and, as long as the destination URL tested is also random, that node cannot feasibly distinguish between test traffic and real traffic. However, this implementation's BFT algorithm, PBFT, scales poorly above 100 nodes, and we advise using a scalable BFT algorithm such as tPBFT for larger deployments [24].

Subsequently, we make node-16 malicious by having it always fail all other nodes in its reputation report transactions for a 1-hour period. Figure 8a shows a small impact on the reputations of some other nodes that node-16 happened to select and report failures for (node-16 cannot target a particular node). However, Figure 8b shows how easily node-16's anomalous reports can be distinguished from the rest (via its Sawtooth address state), as it reports only failures in all 11 transactions it can send, whereas there are only 3 other failed tests by all other nodes combined in this time, each for a different node. The dNextG governance committee or additional algorithms could easily detect and take action on node-16 or for similarly colluding nodes, so the small impact would also realistically be short-lived.

Next, we illustrate how a UE can automatically reroute between UPFs using reputation. Figure 9a shows the reputations of a 4-decentralized-node network initialized at 0. The nodes operate normally and pass reputation tests for about 13 minutes, at which point node-2 stops processing UPF packets; its average reputation thus drops back to 0. We also began by configuring a UE to connect through node-1's gNB using node-2's UPF by default (via node-2's unique DNN). The UE tries to connect to google.com and verify TLS several times a minute, and requests the current reputation information from each node's validator every 20 seconds. If the current UPF's reputation drops .5 or more below another UPF (that does not belong to currently used gNB's node), the UE reconnects



(a) Average reputations when node 2 stops processing traffic

(b) UE TLS Results Through Chosen UPF and node 1's gNB

Figure 9: UE Reactivity to Reputation

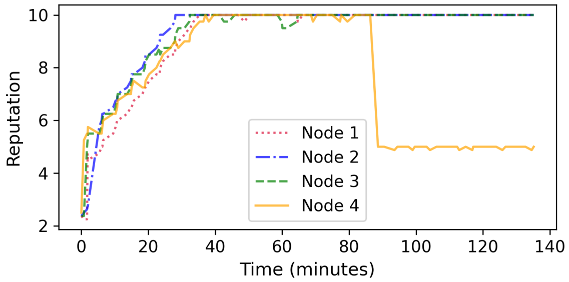


Figure 10: Average reputations when node 4 stops publishing

to the higher-reputation UPF. It also tries to reconnect using the next-highest reputation UPF if it experiences repeated TLS-based internet connection failures. Figure 9b shows how the UE initially connects through node-2's UPF and constantly verifies the TLS connection for the first 13 minutes. node-2's UPF then stops processing packets which causes TLS tests to fail. After one minute, the UE instead selects node-4's UPF and passes its first TLS verification 30 seconds later. This continues until minute 27 when connectivity issues arise through node-4, so the UE connects through node-3's UPF. Some brief fluctuations in average reputation, caused by a few failed reputation tests by node-3, later cause the UE to temporarily connect through node-4 until node-3's reputation is solidly in the lead, at which point it uses node-3's UPF with a 99% rate connection success. Such intermittent UPF hopping can be avoided by tuning the UE's reputation preferences to continue using the current UPF until disconnection, even if its reputation decreases.

Finally, we demonstrate how a UE evaluates reputation when node-4 stops publishing reputation reports. After letting all nodes function properly for 70 minutes, we turn off node-4's reputation reporting as shown in Figure 10. At minute 86, node-4's reputation is decreased from 10 to 5 despite it fulfilling its UPF functions[7], due to it not publishing transactions regularly (though clients may evaluate reputation transactions however they wish).

8 RELATED WORK

While no other known mobile decentralized zero-trust solutions like dNextG exist at the time of writing, many other systems and research still have overlap which is distinguished from dNextG below. Numerous surveys explore research spanning blockchain and mobile networks, mentioning the possibility of sharing mobile network infrastructure and compensation according to use [18], [27], decentralized apps [28], and decentralized authentication [26]. Others survey how blockchain can enable intelligent spectrum sharing [8] and IoT Software Defined Networks (SDNs) [23], or cover the use of blockchain in future mobile networks [25].

Qiao et al. use blockchain for sharing Mobile Edge Computing (MEC) among idle nodes, using a reputation-based approach that influences how traffic is distributed among nodes, but unlike dNextG do not focus on end-to-end stability amidst malicious nodes [20]. Bakogiannis et al. allow independent decentralized operators with idle compute power to band together and be compensated for proper usage of their resources [4]. However, the paper is not meant for mobile networks, but cloud computing. Faisal et al. explore how trust is becoming increasingly important for mobile network providers who must often cooperate to fulfill SLAs, and design a blockchain system to track accountability metrics [9]. Metrics come from a protocol initiated by users that receive out-of-SLA service; unlike dNextG, it does not focus on routing traffic through the network.

Helium is a decentralized wireless LoWaRAN (low-power, wide-area) network actively deployed with over 400,000 hotspots [12]. It supports Internet-of-Things (IoT) devices and uses HoneyBadgerBFT and a hybrid proof-of-location and proof-of-time algorithm (known as Proof of Coverage) to ensure operators are constantly providing service at specific locations. While its hotspots and routers relate to dNextG's base stations and UPFs, its reputation scoring system supports only the RAN and only allows throughput on the order of kilobits per second. Furthermore, Helium runs on a public blockchain, requiring the need to provably, geographically locate its devices and hotspots to not track node membership.

Research surrounding decentralizing 5G networks is sparse and unlike dNextG, only decentralizes the RAN. Helium is testing adding 5G base stations to their network, but none of the 5G Core is decentralized and will use public Mobile Networks Operators such as FreedomFi to provide Core services. Separately, Pollen Mobile [1] is a decentralized 5G network and cryptocurrency similar to Helium which also uses Proof of Coverage to allow anyone to join the network and prove their gNB is providing coverage. Pollen uses dedicated testing UEs to submit validation reports whenever they detect node coverage. The network relies on a fully-centralized Core. Another work uses a consortium blockchain to allow a group of local actors with radio hardware to share their bandwidth with network users in exchange for rewards [16]. It uses proof-of-bandwidth to track the reliability of each endpoint, but lacks an implementation and relies on separate network operators for the entire 5G Core.

9 LIMITATIONS AND FUTURE WORK

While dNextG decentralizes the core user plane, which makes up most traffic, the core control plane remains centralized. Decentralizing the entire control plane may be possible, and also combining it with dNextG and decentralized RANs would produce an end-to-end,

[7] This also explains why figures show reputations starting at 2.5 instead of 5, as they will not have published 5 transactions for the first few minutes.

fully decentralized zero-trust mobile network. Connecting the core to traditional PLMNs and allowing technologies such as Roaming could also be explored as future work. Additionally, connections are assumed to be relatively stable, as while PBFT can tolerate up to one-third of its nodes operating incorrectly, only the majority portion of the network will continue to operate in the midst of partitions. Thus, military situations that require truly mobile networks deployed on moving vehicles with poor continuous reception may not work well. And like all reputation systems, dNextG is vulnerable to wait-and-gain attacks, where adversarial nodes act honestly until they have high reputation and then use it to drop traffic until their reputation drops in response. dNextG disincentives this with a strict node-identification entry governance model which greatly increases cost of doing so for said node operator identities.

Future implementations could support levels of QoS flows based on node capabilities, where UEs could prefer more reputable routing pathways with certain throughput and reliability QoS guarantees for a premium. UEs could also use network statistics besides reputation (e.g. inter-node latency) in evaluating their best path, which could be beneficial in large networks for more efficient, reliable, and shorter distance PDU sessions between nodes' gNBs and UPFs (i.e. allow a low latency node with medium-high reputation to be preferred over a high-reputation node). Load-balancing between the set of highest-reputation nodes may also become important with networks under heavy load to avoid overwhelming these nodes.

10 CONCLUSION

The recent changes in 5G and NextG mobile networks are opening the door to new forms of decentralized mobile architectures that offer numerous benefits and enable use cases from decentralized communities to military coalition networks. However, security and QoS challenges have kept end-to-end network decentralization out-of-reach. dNextG is the first to enable decentralization in the network core's user plane with its blockchain-based reputation system that relies on nodes to consistently test the connectivity of each other's UPFs. We provide a threat model of various actions that malicious nodes can take and an implementation and evaluation to show how dNextG addresses these threats and provides a reliable decentralized mobile network. dNextG is meant to propel decentralized mobile research forward to allow for scalable, affordable networks to connect the world.

ACKNOLWEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant Numbers 1827940 and 2232464. The authors would like to thank their families and Dustin Maas for their support.

REFERENCES

[1] 2022. Pollen Mobile White Paper. (2022). https://assets.website-files.com/61bcdf68fb5dc56d13dd117e/61f94f5e3659b7381a6a1750_Pollen%20Whitepaper%200.0.1.pdf

[2] Sergi Alcalá-Marín, Aravindh Raman, Weili Wu, Andra Lutu, Marcelo Bagnulo, Ozgu Alay, and Fabián Bustamante. 2022. Global mobile network aggregators: taxonomy, roaming performance and optimization. In *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services*. ACM, Portland Oregon, 183–195. https://doi.org/10.1145/3498361.3538942

[3] Jere Backman, Seppo Yrjölä, Kristiina Valtanen, and Olli Mämmelä. 2017. Blockchain network slice broker in 5G: Slice leasing in factory of the future use case. In *2017 Internet of Things Business Models, Users, and Networks*. 1–8. https://doi.org/10.1109/CTTE.2017.8260929

[4] Tasos Bakogiannis, Ioannis Mytilinis, Katerina Doka, and Georgios Goumas. 2020. Leveraging Blockchain Technology to Break the Cloud Computing Market Monopoly. *Computers* 9, 1 (2020). https://doi.org/10.3390/computers9010009

[5] Joe Breen, Andrew Buffmire, Jonathon Duerig, Kevin Dutt, Eric Eide, Anneswa Ghosh, Mike Hibler, David Johnson, Sneha Kumar Kasera, Earl Lewis, Dustin Maas, Caleb Martin, Alex Orange, Neal Patwari, Daniel Reading, Robert Ricci, David Schurig, Leigh Stoller, Allison Todd, Jacobus (Kobus) Van der Merwe, Naren Viswanathan, Kirk Webb, and Gary Wong. 2021. POWDER: Platform for Open Wireless Data-driven Experimental Research. (2021). https://doi.org/10.1016/j.comnet.2021.108281

[6] Miguel Castro, Barbara Liskov, et al. 1999. Practical byzantine fault tolerance. In *OsDI*, Vol. 99. 173–186.

[7] Federal Communications Commission. 2018. Promoting Investment in the 3550-3700 MHz Band - GN Docket No. 17-258. https://docs.fcc.gov/public/attachments/FCC-18-149A1.pdf

[8] Yueyue Dai, Du Xu, Sabita Maharjan, Zhuang Chen, Qian He, and Yan Zhang. 2019. Blockchain and Deep Reinforcement Learning Empowered Intelligent 5G Beyond. *IEEE Network* 33 (May 2019), 10–17. https://doi.org/10.1109/MNET.2019.1800376

[9] Tooba Faisal, Mischa Dohler, Simone Mangiante, and Diego R. Lopez. 2022. BEAT: Blockchain-Enabled Accountable and Transparent Infrastructure Sharing in 6G and Beyond. *IEEE Access* 10 (2022), 48660–48672. https://doi.org/10.1109/ACCESS.2022.3171984

[10] Evan Gilman and Doug Barth. 2017. *Zero Trust Networks*. O'Reilly Media, Sebastopol, CA.

[11] Linley Gwennap. 2008. Single-chip control/data-plane processors. https://www.linleygroup.com/uploads/WP_SCDP.pdf

[12] Amir Haleem, Andrew Allen, Andrew Thompson, Marc Nijdam, and Rahul Garg. 2018. Helium: A Decentralized Wireless Network. (2018). http://whitepaper.helium.com/

[13] John Kindervag. 2010. No More Chewy Centers: Introducing The Zero Trust Model Of Information Security. (2010).

[14] Ben Laurie. 2014. Certificate Transparency: Public, verifiable, append-only logs. *Queue* 12, 8 (Aug 2014), 10–19. https://doi.org/10.1145/2668152.2668154

[15] Jinyuan Li and David Mazieres. 2007. Beyond One-third Faulty Replicas in Byzantine Fault Tolerant Systems. (2007).

[16] Vincent Messié, Gaël Fromentoux, Xavier Marjou, and Nathalie Labidurie Omnes. 2019. BALAdIN for blockchain-based 5G networks. In *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*. 201–205.

[17] Sai Shruthi Nakkina, Sudhakar Balijepalli, and Chandra R. Murthy. 2021. Performance Benchmarking of the 5G NR PHY on the OAI Codebase and USRP Hardware. In *WSA 2021; 25th International ITG Workshop on Smart Antennas*.

[18] Dinh C. Nguyen, Pubudu N. Pathirana, Ming Ding, and Aruna Seneviratne. 2020. Blockchain for 5G and beyond Networks: A State of the Art Survey. 166 (2020), 102693. https://doi.org/10.1016/j.jnca.2020.102693

[19] Jonathan Olsson, Andrey Shorov, Loay Abdelrazek, and Jorden Whitefield. 2021. 5G Zero Trust – A Zero-Trust Architecture for Telecom. 5 (2021), 2–11. https://doi.org/10.23919/ETR.2021.9904691

[20] Guanhua Qiao, Supeng Leng, Haoye Chai, Arash Asadi, and Yan Zhang. 2019. Blockchain Empowered Resource Trading in Mobile Edge Computing and Networks. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. 1–6. https://doi.org/10.1109/ICC.2019.8761664

[21] Kapil Raina. 2022. What is Zero Trust Security? Principles of the Zero Trust Model. https://www.crowdstrike.com/cybersecurity-101/zero-trust-security/

[22] Scott Rose, Oliver Borchert, Stu Mitchell, and Sean Connelly. [n. d.]. Zero Trust Architecture. https://doi.org/10.6028/NIST.SP.800-207

[23] Pradip Sharma, Saurabh Singh, Young-Sik Jeong, and Jong Park. 2017. DistBlockNet: A Distributed Blockchains-Based Secure SDN Architecture for IoT Networks. *IEEE Communications Magazine* 55 (Jan 2017), 78–85.

[24] Song Tang, Zhiqiang Wang, Jian Jiang, Suli Ge, and GaiFang Tan. 2022. Improved PBFT algorithm for high-frequency trading scenarios of alliance blockchain. *Scientific Reports* 12, 11 (Mar 2022), 4426.

[25] Jiaheng Wang, Xintong Ling, Yuwei Le, Yongming Huang, and Xiaohu You. 2021. Blockchain-Enabled Wireless Communications: A New Paradigm towards 6G. 8, 9 (2021). https://doi.org/10.1093/nsr/nwab069

[26] Hui Yang, Haowei Zheng, Jie Zhang, Yizhen Wu, Young Lee, and Yuefeng Ji. 2017. Blockchain-based trusted authentication in cloud radio over fiber network for 5G. In *2017 16th International Conference on Optical Communications and Networks (ICOCN)*. 1–3. https://doi.org/10.1109/ICOCN.2017.8121598

[27] Ruizhe Yang, F. Richard Yu, Pengbo Si, Zhaoxin Yang, and Yanhua Zhang. 2019. Integrated Blockchain and Edge Computing Systems: A Survey, Some Research Issues and Challenges. *IEEE Communications Surveys & Tutorials* 21, 2 (2019), 1508–1532. https://doi.org/10.1109/COMST.2019.2894727

[28] Kaifeng Yue, Yuanyuan Zhang, Yanru Chen, Yang Li, Lian Zhao, Chunming Rong, and Liangyin Chen. 2021. A Survey of Decentralizing Applications via Blockchain: The 5G and Beyond Perspective. *IEEE Communications Surveys & Tutorials* 23, 4 (2021), 2191–2217. https://doi.org/10.1109/COMST.2021.3115797