ACGuard5GC: Privacy-Preserving Prevention of Access Control Attacks within 5G Core Networks

Harsh Sanjay Pacherkar
School of Computing
Binghamton University
Binghamton, New York, USA
hpacher1@binghamton.edu

Guanhua Yan
School of Computing
Binghamton University
Binghamton, New York, USA
ghyan@binghamton.edu

Abstract

Recent revelations of various access control vulnerabilities in 5G core networks have raised severe concerns about their security. Unfortunately, these vulnerabilities are difficult to patch, as they stem from ambiguity and under-specification in 3GPP specifications. This research introduces a new practical solution called ACGuard5GC (**A**ccess **C**ontrol **Guard** for **5G C**ore Networks) to prevent the potential attacks exploiting these access control vulnerabilities in 5G core networks. Its key idea is to deploy reference monitors alongside the Service Communication Proxies (SCPs), which facilitate communications across different network functions in 5G core networks as well as enhancing their scalability and observability. The reference monitors in ACGuard5GC enforce multiple safety properties to prevent the existing access control attacks discovered in the literature. To address the privacy concern in distributed and cooperative defenses by multiple SCPs, ACGuard5GC applies private set intersection on information exchanged among different SCPs. For performance evaluation, we provide a reference implementation of SCP and instrument it with reference monitor functionalities that enforce different safety properties. Our experimental results with two existing 5G core network emulators, Open5GS and VET5G, demonstrate that ACGuard5GC is capable of preventing access control attacks effectively while incurring low operational overhead.

CCS Concepts

• **Security and privacy → Mobile and wireless security**.

Keywords

5G networks; access control attacks; privacy-preserving prevention

ACM Reference Format:
Harsh Sanjay Pacherkar and Guanhua Yan. 2025. ACGuard5GC: Privacy-Preserving Prevention of Access Control Attacks within 5G Core Networks. In *Proceedings of the 30th ACM Symposium on Access Control Models and Technologies (SACMAT '25), July 8–10, 2025, Stony Brook, NY, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3734436.3734450

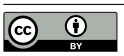
1 Introduction

5G networks hold the great promise of revolutionizing a wide range of industrial sectors, such as healthcare, automotive, and industrial automation. While 5G networks are still gradually rolled out worldwide, their expanded attack surfaces have motivated various investigations on the security threats posed to 5G networks [15, 17, 19, 21, 22, 25, 28, 32].

Recently, a number of access control vulnerabilities have been discovered in 5G core networks [12, 13, 27]. Successful exploitations of these vulnerabilities can lead to various attacks such as leakage of sensitive user data (e.g., user location), denial of service, and unauthorized service access. As these vulnerabilities stem from ambiguity and under-specification in 3GPP specifications, it is difficult to patch these vulnerabilities effectively in practice. On the other hand, it is difficult to customize existing intrusion detection/prevention systems (IDS/IPS) such as Snort [7] and Zeek [10] to defend against access control attacks because 5G core network packets are usually encrypted by the Transport Layer Protocol (TLS). Although PROV5GC [23], a defense system recently proposed for 5G core networks, is capable of *detecting* access control attacks based on provenance graphs constructed from logged signaling messages, it cannot *prevent* these attacks in real time.

Against this backdrop, this work explores how to leverage Service Communication Proxies (SCPs) for preventing access control attacks in 5G core networks. SCPs, which act as intermediaries capable of routing messages, managing interactions among different network functions (NFs), and balancing workload, play a pivotal role in enhancing the efficiency and reliability of service-based communications in a 5G core network. As SCPs, which were introduced in 3GPP Release 16, can improve communication efficiency among 5G core NFs by 30%, they have been increasingly adopted by major mobile network operators worldwide [26]. Based on the high traffic observability at SCPs, we develop a new system called ACGuard5GC (**A**ccess **C**ontrol **Guard** for **5G C**ore Networks), which deploys reference monitors at SCPs to enforce safety properties, a special type of security policies, for access control attack prevention. Assuming that SCPs themselves can be honest but curious, ACGuard5GC uses a privacy-preserving protocol based on private set intersection for multiple SCPs participating in cooperative defenses against access control attacks in 5G core networks.

In a nutshell, our main contributions are summarized as follows. ① Within the ACGuard5GC framework, we design seven safety properties whose enforcement by the reference monitors at the SCPs can prevent various existing access control attacks discovered in the literature. ② We identify a privacy issue when multiple semi-honest SCPs exchange information to enforce these safety properties cooperatively and therefore develop a distributed protocol based on private set intersection to achieve cooperative defenses while preventing unnecessary information leakage. ③ We provide a reference implementation of SCPs based on 3GPP specifications and instrument it with a reference monitor that enforces safety properties



SACMAT '25, Stony Brook, NY, USA

ACM ISBN 979-8-4007-1503-7/2025/07
https://doi.org/10.1145/3734436.3734450

to prevent access control attacks. ④ We evaluate our ACGuard5GC prototype with both the Open5GS 5G core network emulator [4] and the VET5G testbed [23] and our results show that ACGuard5GC is capable of preventing existing access control attacks effectively while incurring low operational overhead. Our code is publicly available at https://github.com/CyberSecurityScience/ACGuard5GC.

2 Primer on 5G Networks

2.1 5G network architecture

In a typical 5G network, whose architecture is illustrated in Figure 1, a User Equipment (UE) such as a mobile phone accesses 5G network services through a Radio Access Network (RAN) comprised of a number of base stations called gNodeBs (gNBs). In the data plane, the user traffic traverses one or multiple User Plane Functions (UPFs) between the gNBs and a data network (DN) such as the Internet and an IP Multimedia Subsystem (IMS). The control plane of a 5G core network follows a service-based architecture (SBA) where multiple network functions (NFs) communicate with each other through well-defined interfaces. Important NFs in the 5G SBA include Access and Mobility Management Function (AMF), Session Management Function (SMF), Authentication Server Function (AUSF), Policy Control Function (PCF), Network Function Repository Function (NRF), Unified Data Management (UDM), Network Slice Selection Function (NSSF), Network Exposure Function (NEF), Charging Function (CHF), Binding Support Function (BSF), and Application Function (AF).

The same physical 5G network can be used to support multiple logically isolated segments called *slices*, each of which can be tailored to meet specific service requirements. A critical concept enabling networking slicing is NSSAI (Network Slice Selection Assistance Information), which can be used by UEs to specify their desired network slices when connecting to the network, or by NFs to specify which slices are allowed to access their services. Each NF in the 5G SBA is characterized by an *NFProfile*, which is a list of NF attributes including *nfInstanceID* (unique ID of the NF instance), *nfType* (NF type, e.g., UPF), *fqdn* (Fully Qualified Domain Name (FQDN) of the NF), *sNssais* (network slice served by the NF), *allowedNfTypes* (NF types allowed to access the NF), *allowedNssais* (network slices allowed to access the NF), and *nfServices* (services provided by the NF). An NF can register its NFProfile with the NRF using the *registerRequest* message and later update its NFProfile using the *updateRequest* message.

The 5G SBA applies OAuth 2.0 [3], an industry-standard authorization protocol, to handle access control among various NFs in the core network. Within this framework, the NRF acts as the central authorization server to issue accessTokens for *consumer NFs* to access services provided by *producer NFs*. The process involves the following three steps: ① **Service discovery:** The consumer NF sends a *NFDiscoveryReq* message to the NRF, including a list of potential service NF instances as well as the query parameters. The query parameters can indicate the types and NSSAIs of the producer NFs to be discovered. The response message from the NRF contains a list of producer NFs discovered according to the consumer NF's request. ② **Access token request:** To access the service from a particular producer NF, the consumer NF contacts the NRF for an access token via an *AccessTokenReq* message, which includes its own NFInstanceID, the scopes of services, and the NFinstanceID or NFType of the producer NF. The NRF verifies whether the request is valid based on the producer NF's NFProfile. If valid, the NRF sends a digitally signed accessToken to the consumerNF. ③ **Service request:** The consumer NF uses the accessToken obtained from the NRF to request the service from the producer NF. The producer NF verifies the validity of the request based on the service scopes and expiration time included within the accessToken before providing service to the consumer NF.

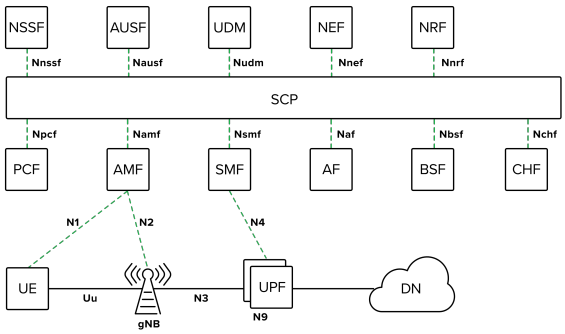


Figure 1: 5G network architecture with SCP

2.2 SCP

3GPP Release 16 [11] has introduced four communication models. While Model A and B support direct communications without and with NRF interactions, respectively, Model C and D both require a new HTTP/2-based NF called SCP serving as a communication proxy for handling signaling messages among various NFs in the 5G SBA. Like Diameter Routing Agents in 4G Long-Term Evolution networks, SCPs enable message routing, load balancing, signalling monitoring, overload handling, and service discovery delegation.

The key difference between Model C and Model D lies in whether service discovery is delegated to SCPs or not. In Model C, the SCPs are responsible for forwarding messages between consumer NFs and producer NFs, but a consumer NF needs to perform service registration and service discovery with the NRF by itself as discussed in Section 2.1. Once accessTokens are obtained from the NRF, the consumer NF sends its service request messages to an assigned SCP, which forwards them either directly to the producer NF or to other SCPs for further routing. In contrast, in Model D, during service registration, an NF sends a RegisterNF service request to an SCP, which forwards the message to the NRF. Similarly, during service discovery, the consumer NF does not send the request message directly to the NRF. Instead, the message is forwarded to an SCP, which performs service discovery on behalf of the consumer NF based on the discovery parameters contained with the service request message (e.g., *3gpp-Sbi-Discovery-target-nf-type*, *3gpp-Sbi-Discovery-snssais*, and *3gpp-Sbi-Discovery-plmns*). The SCP creates a new discovery request message using these parameters and sends it to the NRF to discover an appropriate producer NF. Given the producer NF discovered by the NRF, the SCP further requests an OAuth 2.0 accessToken from the NRF and adds it to the service request message and removes the discovery headers (i.e., *3gpp-Sbi-Discovery-**) before forwarding it to the producer NF or other SCPs

for further routing. In case of the service request being forwarded to other SCPs, the discovered target's FQDN is added to the service request in its *3gpp-Sbi-Target-ApiRoot* header. As SCPs in Model D perform service discovery and selection on behalf of consumer NFs, they offer more control over signalling monitoring, route selection and load balancing than their counterparts in Model C.

3 Motivation, Rationale, and Threat Model

Motivation: access control attacks in 5G core networks. Even with the incorporation of industry-standard OAuth2.0 authorization framework within the 5G core network, various attacks have been found possible against the access control mechanism described in the 3GPP specifications, which are knowingly ambiguous and under-specified [13, 16, 22, 27]. Unfortunately, as these access control attacks abuse specification-level vulnerabilities, they are difficult to prevent using traditional patching mechanisms.

Rationale: SCP-based reference monitor. Due to ambiguity of 3GPP specifications, NF vendors may interpret the 5G standards differently in their implementations, thus leading to possible access control attacks. Due to the practical difficulty of patching the NFs in the 5G SBA to resolve the mismatches in their expected behaviors, a mobile network operator can deploy reference monitors within the 5G core network to prevent access control attacks abusing these mismatches. As SCPs act as central hubs in routing and forwarding signalling messages within 5G core networks, they can be instrumented as reference monitors to enforce the security policies capable of preventing various access control attacks.

Threat model: malicious NFs and semi-honest SCPs. This work assumes that NFs in the 5G SBA may have been compromised by the adversary through attack vectors such as remote code injection and backdoor implantation to allow arbitrary malicious behaviors (**malicious NFs**). A malicious NF can perform various access control attacks as discussed in the literature [12, 13, 27]. We also assume that SCPs, which participate in collaborative defenses for 5G core networks, are not compromised, but they are honest but curious, which means that they do not deviate from the defined protocol but may attempt to gain unnecessary information about which slices other NFs or UEs belong to (**semi-honest SCPs**).

4 ACGuard5GC Design

We propose the ACGuard5GC framework, whose workflow is illustrated in Figure 2, to prevent access control attacks in a 5G core network. The network operator defines a set of security policies that all NFs in the 5G core network must follow. These policies are translated into specific rules enforceable by the reference monitors deployed alongside the SCPs. The SCPs monitor the communication messages transmitted among the NFs and apply these rules to block those messages that violate the pre-defined security policies. As there can be multiple SCPs deployed, each of which observes only a partial subset of all the communication messages in the 5G core network, the SCPs exchange reference monitor assistance (RMA) messages among each other to obtain necessary information from other SCPs to fulfill reference monitor functionalities.

The security policies used by ACGuard5GC are safety properties, which are provably enforceable by reference monitors and can prevent "bad things" such as access control attacks [24]. The

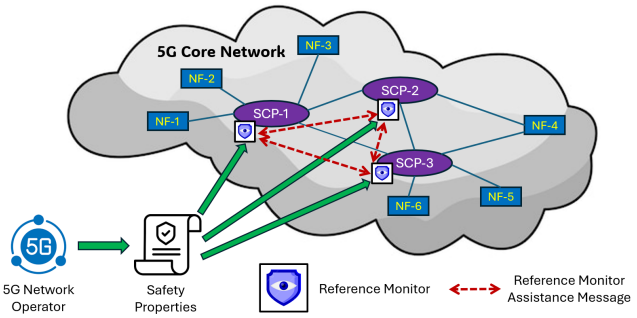


Figure 2: ACGuard5GC workflow

reference monitor deployed at each SCP examines the HTTP/2 messages transmitted among various NFs or other SCPs. We define an execution trace as the sequence of events recorded by the reference monitor. Let Ψ be the universe of all such execution traces. A security policy P can be defined as a logical predicate on sets of execution traces. If we model a system S as a set of its actual execution traces, i.e., $\Sigma_S \subseteq \Psi$, then system S satisfies security policy P if and only if $P(\Sigma_S)$ is always true.

The three requirements for a security policy P to be a safety property are: (1) **Enforceability:** The security policy P can be specified in the form $P(\Pi) : (\forall \sigma \in \Pi : \widehat{P}(\sigma))$, where $\widehat{P}$ is a predicate on individual executions. (2) **Prefix closure:** $(\forall \tau' \in \Psi^- : \neg\widehat{P}(\sigma)) \rightarrow (\forall \sigma \in \Psi : \neg\widehat{P}(\tau'\sigma))$, where Ψ^- denotes the set of all finite prefixes of elements in set Ψ. (3) **Finite refutability:** $(\forall \sigma \in \Psi : \neg\widehat{P}(\sigma)) \rightarrow (\exists i : \neg\widehat{P}(\sigma[..i]))$, where $\sigma[..i]$ denote the first i elements in σ.

As security policies accepted by ACGuard5GC must be safety properties, they can be recognized by a type of Büchi automata called security automata. Formally speaking, security automata $A = (Q, Q_0, I, \delta)$ can be defined with a countable set Q of automaton states, a countable set $Q_0 \subseteq Q$ of initial automation states, a countable set I of input symbols, and a transition function $\delta : (Q \times I) \rightarrow 2^Q$. These security automata are implemented by the SCPs to reject an observed message if it violates any of the pre-defined security policies.

This work assumes that SCPs are deployed under Model D (see Section 2.2) because they are the preferred choices when mobile network operators upgrade their networks from Release 15 to Release 16 or start their 5G network deployments directly from Release 16 [1, 26]. When there are multiple SCPs in the 5G core network, the reference monitor at an SCP has only partial observations of all the HTTP/2 messages transmitted among the NFs in the 5G SBA. Although they can use RMA messages to request Single Network Slice Selection Assistance Information (S-NSSAI) of particular NFs or UEs from other SCPs, ACGuard5GC requires that the SCPs, which are assumed to be honest but curious under the semi-honest SCP threat model, must follow the *need-to-know security principle*: an SCP should not know which slice an NF or UE belongs to except the knowledge that is necessary for it to enforce the safety properties for access control attack prevention. ACGuard5GC applies private set intersection when SCPs use RMA messages to facilitate cooperative defenses against access control attacks in the 5G core.

5 Access Control Attacks with Model-D SCPs

We use an example 5G core network illustrated in Figure 3 to explain how various access control attacks discovered in the literature [12, 13, 27] can be adapted to the SCPs under Model D. In this network there are three slices, each served by a primary SCP.

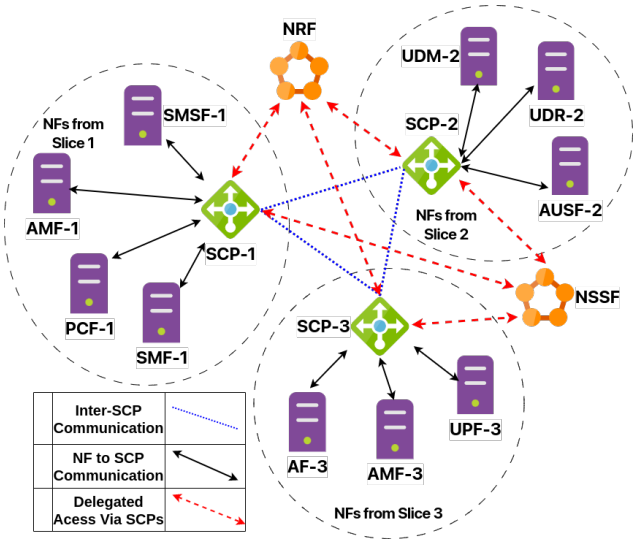


Figure 3: 5G network Deployed for Control Plane Attack

Confused producer attack [13]. It is assumed that AF-3 is malicious, the attack target is AMF-1, to which AF-3 has no access rights, and the attacker knows the FQDN of AMF-1. ① AF-3 sends a service request with *3gpp-sbi-discovery-target-nf-type*=AMF to SCP-3. The service request also includes an additional *3gpp-sbi-target-apiroot* header indicating the apiRoot of the target URI to be AMF-1. ② Upon receiving the request, SCP-3 checks the discovery parameters, and performs the service discovery and access token procedure with the NRF, from which it gets an access token whose audience is NF type AMF. ③ Given that the apiRoot is AMF-1, SCP-3 performs SCP discovery to determine that the primary SCP for AMF-1 is SCP-1. ④ SCP-3 adds the token previously acquired from the NRF to AF-3's original request and forwards it to SCP-1. ⑤ SCP-1 forwards the request to AMF-1. As the access token indicates its audience to be NF type AMF, AMF-1 accepts the service request.

Default over-privilege attack [13]. It is assumed that AF-3 is malicious and the attack target is AMF-1, to which AF-3 has no access rights. ① AF-3 sends an NFUpdate request to the NRF via SCP-3 with the aim of changing the sNssais in its NFProfile to `empty`. ② AF-3 sends a service request to SCP-3, which includes a *3gpp-sbi-target-apiroot* header with AMF-1 to be the apiRoot target and an empty *3gpp-sbi-discovery-snssai* field. ③ SCP-3 performs the service discovery and access token procedure with the NRF. As the NRF treats an empty sNssais as all possible sNssais' allowed by default, it returns an access token for AMF-1. ④ SCP-3 forwards AF-3's service request with the access token to SCP-1. ⑤ SCP-1 forwards AF-3's service request to AMF-1, which accepts the request due to the valid access token.

Parameter misuse attack [13]. It is assumed that AF-3 is the attacker, which does not have access rights to UE data in Slice 1, and UDM-2 is the attack target, whose allowedNssai's include Slices 1, 2, and 3. ① AF-3 sends a service request for UE data in Slice 1 to UDM-2 via SCP-3. ② SCP-3 discovers UDM-2 and gets an access token from the NRF. ③ SCP-3 forwards the request with the access token to SCP-2, which further forwards the request to UDM-2. ④ UDM-2 accepts the service request based on the access token and responds with the sensitive UE data from Slice 1.

Authorization bypass attack [13]. It is assumed that AF-3 is the attacker and AMF-1 is the attack target. ① AF-3 sends a service request towards AMF-1 via SCP-3. For delegated discovery, AF-3 uses a *3gpp-sbi-discovery* header whose *requester-snssai* and *target-snssai* fields are both set to 1. It is noted that the *requester-snssai* field is spoofed as AF-3 belongs to Slice 3. ② SCP-3 performs delegated service discovery and access token requests on behalf of AF-3. The NRF grants the access token because the spoofed *requester-snssai* field indicates that AF-3 belongs to Slice 1. ③ SCP-3 forwards the service request with the access token to AMF-1 via SCP-1. ④ AMF verifies the access token and provides the requested service to AF-3.

Token reuse attack [13]. Under Model D, SCPs perform service discovery and access token requests on behalf of the NFs. Unless SCPs are compromised, which is not assumed by this work, token reuse attacks cannot be conducted successfully by the attacker.

AMF re-allocation attack [27]. It is assumed that SMF-1 is malicious and the attack target is AMF-1. ① SMF-1 sends a service request towards AMF-1 via SCP-1. The service discovery parameters include the requested service as `namf-comm`, *requester-nf-type* as SMF and *target-nf-type* as AMF, sNssai as 1, and the API endpoint as `ue-context-release`. ② SCP-1 performs the delegated service discovery and access token requests on behalf of SMF-1. The NRF grants the access token because the request asks to access service `namf-comm`, which is allowed. ③ SCP-1 forwards the service request with the access token to AMF-1. ④ AMF-1 ensures the validity of the access token and then provides the requested service to SMF-1. However, although an SMF is allowed to use the `namf-comm` service provided by an AMF, it is not supposed to invoke the `ue-context-release` API endpoint.

Subscription Data management [27]. When SCPs are used in the 5G core network, the attack steps remain the same as in the AMF re-allocation attack, except that the attack target is the UDM and the API endpoint, which is under *nudm – sdm*, is normally not invoked from the NF type used in the attack.

Data-Repository service exposure [27]. The attack steps remain the same as in the AMF re-allocation attack, except that the attack target is the UDR, which stores UEs' subscription data, and the API endpoint, which is under *nudr–sdr*, is normally not invoked from the NF type used in the attack.

Negated OAuth policy attack [27]. The OpenAPI specification used to generate APIs for 5G core NFs has a flaw: the OAuth tokens used for access control are marked as optional. If the APIs generated do not enforce the Oauth tokens as mandatory, NFs can directly access the services without OAuth tokens. As access token requests are performed by the SCPs under Model D, a successful negated OAuth policy attack requires cooperation from an SCP to use an empty access token in the service request forwarded to the victim target NF. Therefore, under the semi-honest SCP threat model, negated OAuth policy attacks are not possible.

Client	Server	API Endpoints
AMF	UDM	POST, DELETE /{ueId}/sdm-subscriptions/* GET /{supi}/nssai PUT, GET /{supi}/am-data/* GET {supi}/smf-select-data GET {supi}/ue-context-in-smf-select-data
SMF	UDM	GET /shared-data POST /shared-data-subscriptions
SMF	UDM	GET /{supi}/sm-data
AUSF	NRF	PUT /nf-instances/nfInstanceID (NFProfile) PATCH /nf-instances/nfInstanceID (PatchData)

Table 1: API endpoint examples

Slicing attack: unauthorized access [12]. The attack steps are the same as in the authorization bypass attack.

Slicing attack: denial of service [12]. This attack utilizes a *3gpp-sbi-oci* header indicating overload or congestion in the 5G core network. The notified NF can take actions such as rejecting requests from the NFs from the affected slices. It is assumed that AMF-3 is the attacker and the targets include all NFs in Slice 1, which try to communicate with UDM-2. ① AMF-3 sends a service request to SCP-3 directed at UDM-2. The request message includes a *3gpp-sbi-oci* header indicating Slice 1 to be congested. ② SCP-3 performs the service discovery and access token requests on behalf of AMF-3 with the NRF. SCP-3 discovers UDM-2 and obtains a valid access token from the NRF. ③ SCP-3 forwards AMF-3's original service request along with the access token to UDM-2 via SCP-2. ④ Based on the *3gpp-sbi-oci* header, UDM-2 wrongly believes that Slice 1 is congested and thus blocks incoming requests from this slice for some time, leading to a denial of service attack.

Slicing attack: information leakage [12]. The attack steps are the same as in the parameter misuse attack.

6 Security Policies

This section presents what security policies are needed in order to prevent access control attacks in the 5G core network. These security policies are designed based on the logical gaps in the 3GPP specifications that enable potential access control attacks. A mobile network operator can customize these security policies according to its network implementation. For ease of exposition, we consider a single SCP which observes all signaling messages in the 5G SBA.

6.1 Safety Properties

We define a trace σ as a sequence of HTTP/2 messages observed by the SCP, i.e., $\sigma = \{m_1, m_2, m_3, ...\}$. For each message, we define its msgType attribute to indicate its message type (**HTTP2Request** or **HTTP2Response**).

If $m \in \sigma$ is an HTTP/2 request (i.e., m.msgType = **HTTP2Request**), we define the following attributes which can be extracted from message m for access control purposes:

- sndSlices: a set of the sender NF's slices in the *3gpp_discovery_requester_snssais* header.
- sndIP: the sender NF's IP address.
- rcvFQDN: the receiver NF's FQDN, which is established from delegated service discovery.
- sndType: the NF type of the sender NF.
- rcvType: the NF type of the receiver NF.
- ueIDs: the set of UE IDs contained in the messages represented as SUPIs.
- apiType: the API type of the request in HTTP/2 parlance (e.g., GET, DELETE, and UPDATE)
- apiEndpoint: the API endpoint of the request.
- ociSlices: the set of S-NSSAIs appearing in the *3gpp-Sbi-Oci* header. It is set to be $\emptyset$ if there is no OCI header or no S-NSSAIs are given in the OCI header.
- targetRoot: the *3gpp-Sbi-Target-apiRoot* header, which is set to be NULL if absent, identifies the primary location to access a specific NF within the 5G core network.

Map *NFData* is used to store two attributes for each NF:

- slices: the set of slices that the NF belongs to.
- allowedSlices: the set of slices that the NF can accept the requests from.

Map *NFData* can be queried based on either the NF's FQDN or its IP address. The information stored in *NFData* is collected and updated based on the RegisterNF and UpdateNF service request messages going through the SCP. The RegisterNF service request contains an initial NFProfile of a particular NF that is sent to NRF. This NFProfile contains all the identification and configuration information of the NF, including its FQDN, IP address, NF instance identifier, S-NSSAI and allowed SNSSAI. These information is used by the SCP to populate its *NFData* map for this particular NF. Similarly, the UpdateNF service request messages are used to update the NFProfile information stored at the NRF. Hence, the new information stored in these messages is used to update the *NFData* map when they are forwarded by the SCP.

Similarly, map *UEData* stores the mapping from an UE's SUPI to its allowedSlices attribute, which indicates the set of slices that can serve the UE after successful registration. The SCP monitors GetAMData or GetNssai service requests, which are part of the Subscription Data Management service provided by the UDM. Here, the AMF includes the UE SUPI as a query parameter to retrieve from the UDM, whose response contains the subscribed slices of the UE. As the SCP already has the information about the AMF's assigned slices, an intersection of the AMF's slices and the UE's subscribed slices is derived to fill the *UEData* map for this UE.

Let *DefinedEndpoints* denote the set of API endpoints defined between two NF types in the 5G specifications. Table 1 gives a few API endpoint examples. Given an NF type tuple (s, r), *DefinedEndpoints*$[s, r]$ returns the set of API endpoints allowed between NF types s and r according to 5G specifications. A 5G network operator can update *DefinedEndpoints* based on the API endpoints deployed by the NFs in its network. We use *UEDataEndpoints* to denote the full set of API endpoints in *DefinedEndpoints* containing sensitive UE data. For example, */namf-loc/v1/<UEID>/provide-loc-info* is an API endpoint at the AMF storing the sensitive location data of the UE identified by UEID.

Next, we define a predicate set $\widehat{P} ::= \{P_1, P_2, P_3...\}$, each of which is a policy predicate enforceable by the SCP.

- $P_1 =: (m.\text{msgType} = \textbf{HTTP2Request} \wedge m.\text{rcvType} \neq \textbf{NRF}$ or **NSSF**$) \rightarrow (m.\text{sndSlices} \neq \emptyset)$. This policy indicates that a service request should contain a nonempty set of the sender NF's slices unless the NRF or NSSF is the receiver NF.

- P_2 =: (m.msgType = **HTTP2Request**) → (r.sndSlices ⊆ $NFData[r.\text{sndIP}]$.slices). This policy means that for each service request message, its sender NF's slices should match those stored in map *NFData*.
- P_3 =: (m.msgType = **HTTP2Request** ∧ m.ociSlices ≠ ∅) → (m.ociSlices ⊆ m.sndSlices). This policy means that if the request message contains a *3gpp-Sbi-Oci* header with S-SNSSAIs, then the slices indicated by these IDs should belong to those served by the sender NF.
- P_4 =: (m.msgType = **HTTP2Request** ∧ m.apiEndpoint ∈ *UEDataEndpoints*) → (∀i ∈ m.ueIDs : $UEData[i]$.allowedSlices ∩ m.sndSlices ≠ ∅). This policy means that if the current request's API endpoint allows to obtain UE Data, then the UEs identified by the message should allow the services from the slices to which the sender NF belongs to.
- P_5 =: (m.msgType = **HTTP2Request** ∧ m.targetRoot ≠ NULL) → ($NFData[m.\text{targetRoot}]$.allowedSlices ∩ m.sndSlices ≠ ∅). This policy indicates that if the target API root is present in the service request, the allowed slices of the indicated NF should overlap with sender NF's slices.
- P_6 =: (m.msgType = **HTTP2Request**) → (m.apiEndpoint ∈ *DefinedEndpoints*[m.sndType, m.rcvType]). The policy indicates that the API endpoints of the current service request must be allowed for the sender and the receiver's NF types.

If an NF can send arbitrary NFUpdate requests to the NRF to update its or other NFs' NFProfile's, the aforementioned policies can be easily bypassed. According to 5G specifications, NFUpdate messages are sent to the NRF using either HTTP PUT or PATCH requests, as shown in Table 1. ACGuard5GC requires that all NFUpdate messages must be authorized by the 5G Operations, Administration, and Maintenance (OAM). We assume that authorized NFUpdate messages are signed by the OAM and its digital signature can be verified by the SCP, which is preconfigured with the OAM's public key. We introduce the following policy to ensure the validity of NFUpdate messages:

- P_7 =: ((m.msgType = **HTTP2Request**) ∧ (m.rcvType = **NRF**) ∧ (m.apiType = PUT or PATCH) ∧ (m.apiEndpoint contains "/nf-instances/")) → m is signed by OAM).

Type of Access Control Attack	Security Policy
Confused producer attack	P_5, P_7
Default over-privilege attack	P_1, P_7
Parameter misuse attack	P_4, P_7
Authorization bypass attack	P_2, P_7
Token reuse attack	~
AMF re-allocation attack	P_6
Subscription data management attack	P_6
Data-repository service exposure attack	P_6
Negated OAuth policy attack	~
Slicing attack: unauthorized access	P_2, P_7
Slicing attack: denial of service	P_3, P_7
Slicing attack: information leakage	P_4, P_7

Table 2: Security policies responsible for attack prevention

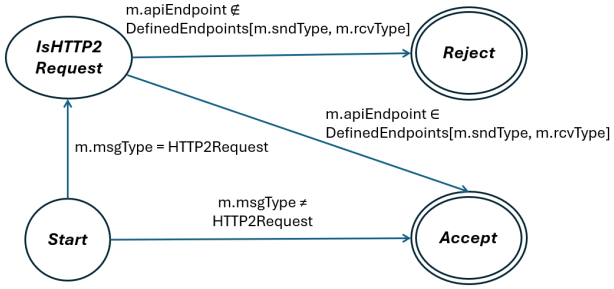


Figure 4: Security automaton for policy P_6

Table 2 presents the security policies that can be used to prevent each specific access control attack effectively. For example, the confused producer attack requires to specify the victim NF as the target API root to which the attacker's NF does not have access rights and can thus be prevented by enforcing policy P_5. Moreover, policy P_7 prevents the attacker NF from sending arbitrary NFUpdate messages to the NRF to change the victim NF's NFProfile. It is easy to prove that all policies P_{1-7} satisfy the three requirements of safety properties: enforceability, prefix closure, and finite refutability.

6.2 Security Automata

As each policy P_i is a safety property, it can be enforced by a corresponding security automaton A_i [24]. Using policy P_6 as an example, the security automaton $A_6 = (Q, Q_0, I, \delta)$ can be defined as follows. Set Q contains four states: Q = {*Start*, *IsHTTP2Request*, *Accept*, *Reject*}. The initial state Q_0 is *Start*. The input symbols are the various attributes extracted from the messages. The transition function δ is illustrated in Figure 4.

On the arrival of each message m, the reference monitor at the SCP sequentially applies the seven security automata A_{1-7} and as long as any of them enters a *Reject* state, message m is deemed as dangerous and thus blocked. As these dangerous messages are not delivered to their intended target NFs, the access control attacks can be effectively prevented.

7 Privacy-Preserving Cooperative Defense

In 5G core networks, SCP deployments can be deployed per slice, per shared slice, or per Public Land Mobile Network (PLMN). There may be multiple SCPs on the paths between consumer and producer NFs. Under a multi-SCP deployment, as there is no single SCP with the full view of communication messages in the 5G SBA to enforce the security policies defined in Section 6.1, cooperation is needed among the SCPs for effective prevention of access control attacks.

7.1 Reference monitor assistance API

Examining all policies P_{1-7}, we note that enforcement of P_2 and P_5 requires $NFData$ while P_4 checks $UEData$ for the slice IDs of particular UEs. In a multi-SCP scenario, both $NFData$ and $UEData$ can be distributed at different SCPs. However, if these data are stored at arbitrary places, it can be challenging to design a distributed protocol that can discover and share these data efficiently.

When SCPs are used under Model D, an NF is configured with an SCP to perform delegated service discovery and access token request procedures on its behalf. We call this SCP the *primary*

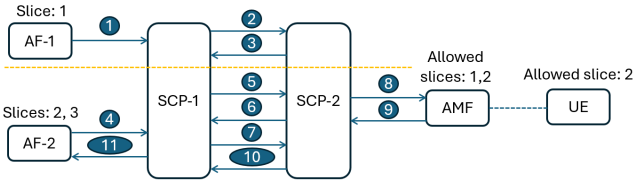


Figure 5: Signaling messages in a multi-SCP scenario. Messages 2 and 5 are RMA request messages from SCP-1. Messages 3 and 6 are their corresponding responses from SCP-2.

SCP of this NF. The primary SCP is responsible for managing the NF's slices and allowedSlices attributes in its local *NFData* map. As the UEs' slices are assigned by the AMFs in 5G core networks, the primary SCP of an AMF is responsible for managing its UEs' allowedSlices attributes stored in its local *UEData* map.

To facilitate cooperative defenses, each SCP adds a new *Reference Monitor Assistance (RMA)* API endpoint "/nscp-rma" to assist other SCPs in checking an NF's slices and allowedSlices attribute stored in its local *NFData* map or a UE's allowedSlices attribute stored in its local *UEData* map. This API endpoint is invoked by RMA messages received from other SCPs.

We use a simple example to illustrate how cooperative defenses can be achieved with RMA messages. Figure 5 presents a multi-SCP scenario where SCP-1 and SCP-2 need information exchange to enforce policy P_4. ① AF-1 sends a *ProvideLocationInformation* request with the UE's ID to the AMF where the sender NF's slice ID is 1. ② SCP-1 processes the request, performs a service discovery procedure and get the AMF as the destination NF. While checking policy P_4, SCP-1 notices that *ProvideLocationInformation* is an API concerning sensitive UE data (i.e., belonging to *UEDataPoints*). As the UE's ID does not exist in its local *UEData* map, SCP-1 uses the RMA API endpoint of SCP-2, which is the primary SCP of the AMF, to request for the UE's allowSlices attribute. The RMA request message also includes the AF's slice ID, which is Slice 1. ③ On the arrival of the RMA request message from SCP-1, SCP-2 checks whether AF-1's slice ID overlaps with the UE's allowedSlices stored in its local *UEData* map. If it does, a positive response RMA message is sent from SCP-2 to SCP-1; otherwise, a negative one is sent. In this example, as the UE's allowedSlices only includes Slice 2, a negative RMA response message is sent to SCP-1. Due to the negative RMA response message received from SCP-2, SCP-1 determines that policy P_4 is violated and thus rejects the service request message from the AF to the AMF.

In a different case, AF-2, which belongs to both Slices 2 and 3, does a similar request. Hence, messages 4 and 5 are the same as messages 1 and 2, except that the sender NF is AF-2 instead. However, as AF-2's slices include 2, which overlaps with the UE's allowed slices, message 6 returns a positive RMA response message. As no policy is violated, SCP-1 forwards AF-2's service request to SCP-2 (message 7), which further forwards it to the destination AMF (message 8). The AMF returns the UE's current location to AF-2 through messages 9, 10, and 11.

7.2 Private set intersection

In the protocol described in Figure 5, SCP-2 may know the exact slices that the two AFs belong to, which is unnecessary. If the two RMA request messages (messages 2 and 5) contain both the sender NF's identity and the slices it belongs to, SCP-2 can trivially obtain such knowledge. A slight modification to the protocol can be hiding the sender NF's identity in message 2 or 5 with only the sender NF's slices included. Even without knowing the sender NF's identity, SCP-2 can still check whether the sender NF's slices overlap with the UE's allowed slices and thus returns the correct RMA response message to SCP-1. However, if the response is positive and thus no policy is violated, SCP-1 continues to forward AF-2's request message to SCP-2 (message 7), by which SCP-2 infers that AF-2 belongs to Slices 2 and 3 (from message 5). It is noted that AF-2's service request message forwarded from SCP-1 to SCP-2 (message 7) does not include the sender NF's slices, which are used only in the service discovery and access token requests by SCP-1 with the NRF and thus stripped from the message forwarded to SCP-2 and then to the AMF (see Section 2.2).

To address the data leakage issue, a potential solution is to let SCP-1 and SCP-2 calculate privately the intersection between the AF's slice IDs and the allowed slice IDs of the UE. The problem can be stated as follows: assuming that there are two SCPs, SCP_a and SCP_b, each possessing a set of IDs, which are denoted as D_a and D_b, respectively, how can SCP_a know whether $|D_a \cap D_b| = 0$ while ensuring that neither party knows any IDs possessed by the other one, including those in the intersection?

ACGuard5GC uses a protocol inspired by Facebook's private ID matching scheme [18]. It consists of the following steps. ① SCP_a creates a private key k_a. For each ID item $x \in D_a$, it calculates its hash value $H(x)$ and then encrypts it with its private key to obtain Diffie-Hellman value $(H(x))^{k_a}$. SCP_a shuffles the resultant encrypted hash ID set to get U_a, and sends U_a, along with a hint q, to SCP_b using an RMA request message. The hint q indicates what data at SCP_b should be used for private set intersection. ② On the arrival of the RMA request message from SCP_a, SCP_b first uses hint q to obtain the ID set of interest, D_b, from its local $NFData$ or $UEData$ map. SCP_b also creates its own private key k_b. For each ID item $y \in D_b$, it calculates its hash value $H(y)$ and then encrypts it with its private key to obtain Diffie-Hellman value $(H(y))^{k_b}$. Similarly, SCP_b shuffles this encrypted hash ID set to get $U(b)$. Next, for each item $z \in U_a$ received from SCP_a in its RMA request message, SCP_b encrypts it with its private key to obtain z^{k_b}. Let $U_{a,b}$ denote the dataset after encrypting each item in U_a using SCP_b's private key k_b. Finally SCP_b sends both U_b and $U_{a,b}$ to SCP_a in an RMA response message. ③ For each item in U_b in the RMA response message, SCP_a uses its private key k_a to encrypt it. Let $U_{b,a}$ be the resultant dataset. SCP_a further checks whether $U_{b,a}$ intersects with $U_{a,b}$, which is obtained from the RMA response message as well. Assuming that there is no hash collision, we have $|U_{b,a} \cap U_{a,b}| = |D_a \cap D_b|$. Due to encryption with the private keys k_a and k_b and the one-way hash function H, neither SCP_a nor SCP_b is able to recover the original IDs possessed by the other party.

Following the same example shown in Figure 5, SCP-1 and SCP-2 act as SCP_a and SCP_b, respectively, in the protocol. The RMA request and response messages are generated according to the private set intersection protocol. Both RMA request messages 2 and 5 include the same hint q, which is the UE's ID whose allowed slices should be compared against the sender NF's slices. When

the sender NF is AF-1 and AF-2, D_a is $\{1\}$ and $\{2, 3\}$, respectively, while $D_b = \{2\}$ in both cases.

8 Implementation Details

SCP implementation. We have implemented the basic features of an SCP based on Release 16 of 3GPP specifications using the Rust programming language, including its hyper and hyper-tls libraries for HTTP2 layer with TLS, tokio for asynchronous code execution, and swagger for HTTP request contexts. This SCP supports delegated service discovery and access token request procedures as described in Section 2.2. It also supports subscribe/notify messages and the Model D communication model with TLS enabled. It implements target NF indications using *3gpp-Sbi-Target-ApiRoot* and *3gpp-Sbi-Routing-Binding* headers. The information contained in these headers allows an SCP to select a suitable target NF for handling this particular request. An SCP can be assigned as the next-hop by others, thus allowing it to serve as intermediaries in multi-SCP scenarios. Each SCP keeps state information regarding the consumer and producer NFs to facilitate service discovery and message routing.

We have incorporated our SCP implementation into the VET5G testbed [31], which provides end-to-end 5G network emulation capabilities with multiple UE/RAN options supported: UERANSIM [9], Android Virtual Devices (AVDs) communicating with gNBs emulated by OAI [5], and Commercial-Off-The-Shelf (COTS) 5G phones communicating with gNBs emulated by srsRAN [8]. We have also modified the NFs in VET5G to support communications through SCPs operated under Model D.

ACGuard5GC implementation. We have instrumented the vanilla SCP with a reference monitor that uses security automata to enforce the seven safety properties described in Section 6. The instrumented SCP uses the *NFData* and *UEData* maps to store contextual information extracted from RegisterNF/UpdateNF and GetAMData/GetNssai service request messages, respectively. Both *NFData* and *UEData* are implemented with Rust's *hashmap* data structure.

To support cooperative defenses, each SCP implements the RMA API to support the */nscp-rma* endpoint based on its existing HTTP/2 interface. We created our private set intersection protocol based on Facebook Research's Private-ID repository [6], which includes a collection of algorithms for matching records between multiple parties while preserving data privacy. In our implementation, the keys are generated as a 512-bit integer based on the Curve25519 scalar. Data records are hashed by SHA-512 and encrypted using the ECRistretto cipher. For data permutation, we simply use the rand crate of the Rust language.

In comparison with the vanilla SCP, which is implemented with ∼6.7K lines of Rust code, the instrumented SCP with reference monitor capabilities uses ∼10K lines of Rust code.

Interoperability with Open5GS [4]. To ensure that ACGuard5GC can be deployed in other 5G core network implementations, we choose Open5GS [4] because it also supports SCPs. As our SCP implementation is based on Release 16 of 3GPP Specifications, we used the Open5GS version tagged as 2.5.9, which is the last one supporting Release 16 before transitioning to Release 17. When incorporating ACGuard5GC into Open5GS, we faced problems with HTTP2 API discrepancies such as their use of *user-agent* headers instead of *3gpp-sbi-discovery-requester-nf-type* ones. We make minor modifications to ACGuard5GC to ensure that a UE can be registered into the Open5GS core network through our SCP implementation.

9 Performance Evaluation

9.1 Experimental setup

We configure the 5G core network as in Figure 3. By default, we use VET5G for 5G core network emulation. We consider two scenarios: **single-SCP**, where SCP-1, SCP-2, and SCP-3 are the same SCP, and **multi-SCP**, where SCP-1, SCP-2, and SCP-3 are distinct SCPs. We simulate 50,000 UEs, whose network services are split over the two AMFs, AMF-1 and AMF-3. Each experiment lasts three hours.

UE activities: We modify and configure the UERANSIM to simulate background UE activities with a two-tiered ON/OFF process. At the high level, each UE initiates registration and deregistration procedures periodically. The durations of both of its registered (i.e., ON) and deregistered (i.e., OFF) states follow an exponential distribution with mean time of 5,000 seconds. At the low level, when an UE is registered into the 5G network, it periodically switches between the *CM_CONNECTED* and *CM_IDLE* states. The durations of these two states follow a Pareto distribution with mean time of 500 seconds. Once an UE switches to a *CM_CONNECTED* state, it sends an SMS message to another randomly selected UE and thereafter initiates a PDU session establish request to create a data connection with the Internet.

Attack simulation: AF-3 is assumed to be under the attacker's control to perform all the access control attacks.

Machines used: Single-SCP experiments are performed on a Linux machine running Debian 12. It has an Intel Xeon Gold 6338 CPU with 64 hyper-threaded cores and 512GB RAM (Machine A). For multi-SCP scenarios, experiments are performed on multiple servers: As depicted in Figure 3, NFs in Slice 1, including SCP-1, are deployed on Machine A, NFs in Slice 2, including SCP 2, are deployed on a Debian 11 machine that has an Intel Xeon Gold 6326 CPU with 32 hyper-threaded cores and 128GB RAM, NFs in Slice 3, including SCP-3, are deployed on a Ubuntu 22.04 machine that has an Intel Xeon Gold 6140 CPU with 36 hyper-threaded cores and 192GB RAM, NRF runs on a Linux Mint 21.3 machine with Intel i7-1360P and 64GB RAM, and NSSF runs on a Ubuntu 24.04.1 machine with Intel i7-7700K and 32GB RAM. All these machines belong to the same Local Area Network.

9.2 Attack prevention effectiveness

To study ACGuard5GC's effectiveness in preventing access control attacks within 5G core networks, we consider these networks emulated by both VET5G and Open5GS. As VET5G is a security-oriented testbed for 5G networks, it allows us to simulate the access control attacks listed in Table 2. Recall that neither the token reuse attack nor the negated OAuth policy attacks can be performed successfully with SCPs deployed under Model D (see Section 5).

When Open5GS is used as the 5G core network emulator, we realized the absence of some security features, such as SNSSAI not used for NF registration and discovery and lack of support for OAuth tokens in service requests, suggesting that access control attacks based on slices and OAuth tokens cannot be simulated in a realistic manner. Moreover, Open5GS does not support the *3gpp-sbi-oci* headers and is thus unable to simulate the slicing attack that

Access Control Attack	Open5GS			VET5G		
	AO	**AP**	**FP**	**AO**	**AP**	**FP**
Confused producer	✘	✘	✘	29	29	0
Default over-privilege	✘	✘	✘	31	31	0
Parameter misuse	✘	✘	✘	33	33	0
Authorization bypass	✘	✘	✘	35	35	0
Token reuse	✘	✘	✘	✘	✘	✘
AMF re-allocation	5	5	0	10	10	0
Subs data management	5	5	0	7	7	0
Data-repo service exposure	5	5	0	10	10	0
Negated OAuth policy	✘	✘	✘	✘	✘	✘
Slicing: unauthorized access	✘	✘	✘	35	35	0
Slicing: denial of service	✘	✘	✘	25	25	0
Slicing: information leakage	✘	✘	✘	33	33	0

Table 3: Attack prevention results under different 5G core network emulators. AO: attack occurrences; AP: attacks prevented by ACGuard5GC; FP: false positives. ✘ corresponds to a case where the attack cannot be successfully simulated. Subs: subscription; repo: repository.

leads to denial of service. Eventually, we managed to simulate only three access control attacks within the Open5GS-emulated core network, which are AMF re-allocation attack, subscription data management attack, and data-repository service exposure attack.

In each experiment, we let AF-3 perform an access control attack at a random time interval with mean time of 10 minutes over a duration of 3 hours, leading to about 18 attacks per test run. Overall we performed 10 experiments. We log the occurrence of each attack as well as the time when it is prevented by a particular policy.

Table 3 summarizes our results from all the experiments. Clearly, for all those access control attacks that can be simulated successfully, ACGuard5GC is able to prevent them effectively. Moreover, no false positives have been observed in our experiments, suggesting that ACGuard5GC offers a viable approach to preventing access control attacks within 5G core networks in practice.

9.3 Computational resource usage

A new set of experiments are done to examine the resource usage of ACGuard5GC. As SCPs operated under Model D handle all the SBI-based traffic within the 5G core network, it is important to ensure that their reference monitor capabilities do not incur significant computational overhead. To monitor the CPU and memory usage per SCP in both single-SCP and multi-SCP scenarios, we use the Docker python library to log the Docker container stats every second for the total duration of the experiment. For the multi-SCP scenario, we aggregate the CPU and memory usage statistics of all three SCPs. Each experiment is executed 10 times to obtain the average resource usage results.

Figure 6 shows the average CPU usage in percentage and average memory usage in gigabytes (GBs) per SCP. We observe that the average CPU usage is 3.93% and 4.12% for the single-SCP scenario with policies turned off and on, respectively. Therefore, security policy enforcement causes the CPU usage to increase slightly by 4.8%. Similarly, in the multi-SCP scenario, the average CPU usage is 0.99% and 1.52% with policies turned off and on, respectively. The single-SCP scenario has a much higher CPU usage than the multi-SCP one because the single SCP handles the aggregated workloads from all of SCP-1, SCP-2, and SCP-3 in Figure 3. In comparison with the single-SCP deployment, the increase of CPU usage due to security policy enforcement is more significant, which is not surprising given that SCPs need to exchange information with the others based on the private set intersection protocol in the multi-SCP deployment.

Figure 6 also tells us that under the single-SCP scenario, the average memory usage has increased from 0.832 GB with security policies turned off to 1.019 GB with security policies turned on, leading to an increase of 22.48%. Similarly, under the multi-SCP scenario, the average memory usage has increased by 30.15% from 0.859 GB with security policies turned off to 1.118 GB with security policies turned on. The increased memory usage is mainly caused by the *NFData* and *UEData* maps maintained by each SCP.

As our SCP implements only the basic functionalities, the relative resource usage incurred by ACGuard5GC should be much less than the above measurements for a feature-rich SCP in practice.

9.4 Policy enforcement time

In this set of experiments test, we log the execution latency of each security policy inside the SCP. For each request received by the SCP, we let it go through all the policies one by one and measure the runtime for each policy. Figure 7, which summarizes the experimental results, shows that a single policy causes an average latency of 1.89 microseconds while all seven policies need 13.29 microseconds under the single-SCP scenario. Similarly, under the multi-SCP scenario, there is average latency of 2.05 microseconds per policy while the aggregate latency over all seven policies is 14.48 microseconds. The higher per-policy latency under the multi-SCP scenario is understandable because for some policies (i.e., P_4 and P_5), the SCP needs to request information from others based on the private set intersection protocol. Interestingly, for both single-SCP and multi-SCP scenarios, security policy P2 causes the highest execution latency. The root cause is that P2 is not only one of the only two security policies that apply to *all* HTTP2Request messages (the other one is P6) but also one of those that require to check the local *NFData* or *UEData* map.

For policies P_4 and P_5, we measure the RMA turnaround time, which is the duration between the arrival of an RMA request and the time that an RMA response is sent back, to assess the delay due to the private set intersection protocol. When processing an RMA request, the SCP needs to perform a lookup operation in its own *NFData* hashtable for the given ID, then encrypt and shuffle the data before sending the response back. The average RMA turnaround time for both policies P_4 and P_5 is 90.7 microseconds.

9.5 Comparison with other defense systems

Other IDS/IPS systems may be considered for preventing access control attacks within 5G core networks but they all have drawbacks. Snort [7] and Zeek [10], both are widely used in practice, need to be configured with custom pre-processors or scripts in order to parse 5G packets. These systems also need to be deployed at places inside the 5G SBA where they have a full view of the communication messages that are necessary for enforcing the security policies. Moreover, as 5G SBA traffic are protected with TLS, these

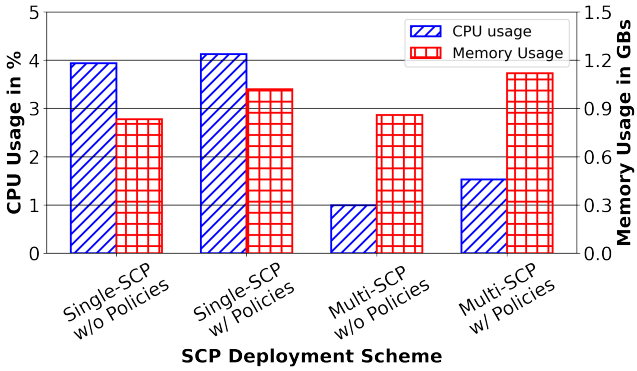


Figure 6: SCP Resource Utilization Comparison

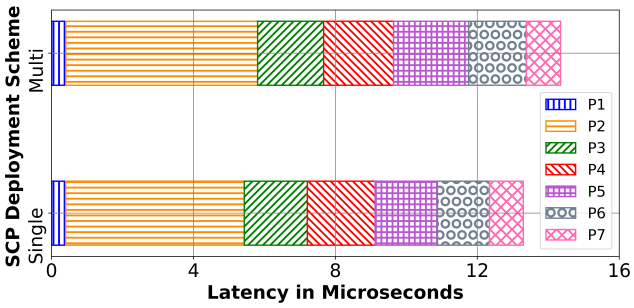


Figure 7: Policy Based Processing Latency

systems have to be configured with appropriate key materials in order for them to inspect encrypted packets. Practically speaking, this is difficult, if not impossible, to achieve as it requires extra trust relationships to be established between the IDS/IPS systems and the individual NFs in the SBA. By contrast, ACGuard5GC exploits the fact that SCPs operating under Model D are delegated by individual NFs to perform service discovery and access token requests on their behalf as well as the inherent trust relationships already established between the SCPs and these NFs.

PROV5GC [23] has recently been proposed for attack detection and attribution based on provenance graphs within 5G core networks. PROV5GC has been shown capable of detecting various attacks, including access control attacks such as the three slicing attacks discovered in [12]. However, PROV5GC is not designed to prevent access control attacks because it requires communication message logs collected by individual NFs to be sent to a centralized attack detection module, which is not capable of dropping malicious communication messages in-situ to prevent access control attacks. In contrast, as all service discovery and access token requests are delegated to SCPs under Model D, they can enforce security policies to prevent malicious ones from being delivered to their destination NFs as done by ACGuard5GC.

ACGuard5GC falls into the category of specification-based IDS/IPS [29]. ACGuard5GC thus shares the advantages of such defense systems, such as having low false positive rates and being able to detect novel attacks, as well as their drawbacks, such as the difficulty in creating precise and comprehensive policy rules and the constant need of keeping the policy rules up to date.

10 Related Work

Vulnerability discovery for 5G networks. The emergence of 5G networks has sparked a plethora of research activities on identifying their new vulnerabilities. Their Authentication and Key Agreement (AKA) protocols have been scrutinized by Basin *et al.* [15] and later by Cremers and Dehnel-Wild [20]. Hussain *et al.* has exposed various new attacks in 5G networks, such as Exposing the Device's TMSI and Paging Occasion attack and Installing Null Ciphering and Null Integrity attack [22], and privacy attacks to the paging protocols based on side channel information [21]. Vulnerabilities of messaging services [32], warning and emergence systems [17], and UE basebands [28] in 5G networks have also been investigated. This work is aimed at preventing access control attacks within 5G core networks, which have been revealed in previous works [12, 13, 27].

Security defenses for 5G networks. The revelations of new vulnerabilities in 5G networks have also motivated researchers to develop new defense techniques. Wen *et al.* proposed 5G-Spector, an O-RAN compliant service for detecting attacks that abuse layer-3 messages in 5G networks [30]. Atalay *et al.* proposed a new microservice-based framework to provide authentication, authorization, and discovery services for xApps in in the O-RAN architecture [14]. ACGuard5GC differs from these two previous works due to its focus on the security of 5G core networks rather than O-RAN. PROV5GC uses provenance graphs constructed from signalling messages for attack detection and attribution in 5G core networks [23]. PROV5GC is more intrusive than ACGuard5GC as it requires *all* NFs in the 5G SBA to add logging capabilities while ACGuard5GC only needs to instrument the SCPs with reference monitors for access control attack prevention.

11 Conclusions

This work introduces the ACGuard5GC system to prevent access control attacks within 5G core networks. Deployed alongside SCPs as reference monitors, ACGuard5GC enforces safety properties on signaling messages transmitted among the NFs in the 5G SBA through the SCPs. Under the honest but curious SCP threat model, ACGuard5GC applies private set intersection on the information exchanged among SCPs for cooperative defenses. Our results have shown that ACGuard5GC can prevent various access control attacks discovered in the literature with low operational overhead.

In our future work, we plan to extend ACGuard5GC as follows. First, we will add mechanisms to prevent access control attacks in other SCP deployment models. Second, we will incorporate standard security policy languages such as Google's CEL [2] and eXtensible Access Control Markup Language (XACML) into ACGuard5GC to allow for creation of comprehensive security policies capable of handling real-world scenarios. Last but not least, we will consider access control attack prevention under other threat models such as colluding NFs and attacks aimed at bypassing reference monitors.

Acknowledgments

We thank the reviewers for their feedback. This work is supported by U.S. National Science Foundation under award CNS-1943079.

References

[1] 2023. 5G signaling: C or D? https://gooram.medium.com/5g-signaling-c-or-d-265a15b55006.
[2] 2025. CEL: Fast, safe expression language. https://cel.dev/.
[3] 2025. OAuth 2.0. https://oauth.net/2/.
[4] 2025. Open5GS. https://open5gs.org/.
[5] 2025. OpenAirInterface. https://www.openairinterface.org/.
[6] 2025. Private-ID. https://github.com/facebookresearch/Private-ID.
[7] 2025. Snort. https://www.snort.org/.
[8] 2025. srsRAN - Your own mobile network. https://www.srsran.com/.
[9] 2025. UERANSIM: An Open Source State-of-the-Art 5G UE and RAN (gNodeB) Simulator . https://github.com/aligungr/UERANSIM.
[10] 2025. Zeek. https://zeek.org/.
[11] 3GPP. 2020. 5G; System architecture for the 5G System (5GS) (3GPP TS 23.501 version 16.6.0 Release 16) .
[12] AdaptiveMobile Security. 2021. A Slice in time: Slicing security in 5G core networks. https://info.adaptivemobile.com/5g-network-slicing-security.
[13] Mujtahid Akon, Tianchang Yang, Yilu Dong, and Syed Rafiul Hussain. 2023. Formal Analysis of Access Control Mechanism of 5G Core Network. In *Proceedings of ACM Conference on Computer and Communications Security*.
[14] Tolga O Atalay, Sudip Maitra, Dragoslav Stojadinovic, Angelos Stavrou, and Haining Wang. 2023. Securing 5G openran with a scalable authorization framework for xApps. In *IEEE Conference on Computer Communications*. IEEE.
[15] David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, and Vincent Stettler. 2018. A formal analysis of 5G authentication. In *Proceedings of ACM Conference on Computer and Communications Security*.
[16] Evangelos Bitsikas, Syed Khandker, Ahmad Salous, Aanjhan Ranganathan, Roger Piqueras Jover, and Christina Pöpper. 2023. UE security reloaded: Developing a 5G standalone user-side security testing framework. In *Proceedings of ACM Conference on Security and Privacy in Wireless and Mobile Networks*.
[17] Evangelos Bitsikas and Christina Pöpper. 2022. You have been warned: Abusing 5G's Warning and Emergency Systems. In *Proceedings of the 38th Annual Computer Security Applications Conference*. 561–575.
[18] Prasad Buddhavarapu, Andrew Knox, Payman Mohassel, Shubho Sengupta, Erik Taubeneck, and Vlad Vlaskin. 2020. Private Matching for Compute. Cryptology ePrint Archive, Paper 2020/599. https://eprint.iacr.org/2020/599 https://eprint.iacr.org/2020/599.
[19] Merlin Chlosta, David Rupprecht, Christina Pöpper, and Thorsten Holz. 2021. 5G SUCI-Catchers: Still catching them all?. In *Proceedings of ACM Conference on Security and Privacy in Wireless and Mobile Networks*.
[20] Cas Cremers and Martin Dehnel-Wild. 2019. Component-based formal analysis of 5G-AKA: Channel assumptions and session confusion. In *Proceedings of Network and Distributed System Security Symposium*.
[21] Syed Rafiul Hussain, Mitziu Echeverria, Omar Chowdhury, Ninghui Li, and Elisa Bertino. 2019. Privacy attacks to the 4G and 5G cellular paging protocols using side channel information. In *Proceedings of the Network and Distributed Systems Security Symposium*.
[22] Syed Rafiul Hussain, Mitziu Echeverria, Imtiaz Karim, Omar Chowdhury, and Elisa Bertino. 2019. 5GReasoner: A property-directed security and privacy analysis framework for 5G cellular network protocol. In *Proceedings of ACM Conference on Computer and Communications Security*.
[23] Harsh Sanjay Pacherkar and Guanhua Yan. 2024. PROV5GC: Hardening 5G Core Network Security with Attack Detection and Attribution Based on Provenance Graphs. In *Proceedings of ACM Conference on Security and Privacy in Wireless and Mobile Networks*.
[24] Fred B Schneider. 2000. Enforceable security policies. *ACM Transactions on Information and System Security (TISSEC)* 3, 1 (2000), 30–50.
[25] Jingwen Shi, Sihan Wang, Min-Yue Chen, Guan-Hua Tu, Tian Xie, Man-Hsin Chen, Yiwen Hu, Chi-Yu Li, and Chunyi Peng. 2024. IMS is Not That Secure on Your 5G/4G Phones. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. 513–527.
[26] Strategy Analytics. 2021. 5G Signaling and Control Plane Traffic Depends on Service Communications Proxy (SCP). https://carrier.huawei.com/~/media/cnbgv2/download/products/core/strategy-analytics-5g-signaling-en.pdf.
[27] Seaver Thorn, K Virgil English, Kevin RB Butler, and William Enck. 2024. 5GAC-Analyzer: Identifying Over-Privilege Between 5G Core Network Functions. In *Proceedings of ACM Conference on Security and Privacy in Wireless and Mobile Networks*.
[28] Kai Tu, Abdullah Al Ishtiaq, Syed Md Mukit Rashid, Yilu Dong, Weixuan Wang, Tianwei Wu, and Syed Rafiul Hussain. 2024. Logic Gone Astray: A Security Analysis Framework for the Control Plane Protocols of 5G Basebands. In *Proceedings of USENIX Security Symposium*. https://www.usenix.org/conference/usenixsecurity24/presentation/tu
[29] Prem Uppuluri and R Sekar. 2001. Experiences with specification-based intrusion detection. In *Proceedings of the International Workshop on Recent Advances in Intrusion Detection*. Springer, 172–189.
[30] Haohuang Wen, Phillip Porras, Vinod Yegneswaran, Ashish Gehani, and Zhiqiang Lin. 2024. 5G-SPECTOR: an O-RAN compliant Layer-3 cellular attack detection service. In *Proceedings of Network and Distributed System Security Symposium*.
[31] Zhixin Wen, Harsh Sanjay Pacherkar, and Guanhua Yan. 2022. VET5G: A Virtual End-to-End Testbed for 5G Network Security Experimentation. In *Proceedings of the 15th Workshop on Cyber Security Experimentation and Test*.
[32] Yaru Yang, Yiming Zhang, Tao Wan, Chuhan Wang, Haixin Duan, Jianjun Chen, and Yishen Li. 2024. Uncovering Security Vulnerabilities in Real-world Implementation and Deployment of 5G Messaging Services. In *Proceedings of ACM Conference on Security and Privacy in Wireless and Mobile Networks*.