

Token Reuse Attack on 5G core Network

1 Introduction

The 5G system offers significant improvements in data speed, latency, and reliability compared to previous cellular networks. However, opening up the 5G core network to third parties presents new security and access control challenges. To address these challenges, the 5G system utilizes the OAuth 2.0 authorization framework as its access control mechanism for the first time. However, there has been no formal analysis of this access control mechanism to date.

The Service-Based Architecture (SBA) of 5G system enables third-party partners to set up additional/complementary core components, e.g., slices on top of network provider's. The introduction of third parties presents new security and access control challenges. To address these challenges, the 5G system utilizes the OAuth 2.0 authorization framework as its access control mechanism for the first time. However, there has been no formal security analysis of the design of this access control mechanism to date.

In our recent research endeavor, we attempted to formally verify the access control mechanism of 5G core. However, this poses several critical challenges, including the lack of commercially deployed 5G Core networks, incomplete open-source implementations, and scalability issues due to numerous configurations.

To address these challenges, we developed NGCoreVerifier, a model-based framework that leverages parameterized model checking to test various 5G core network configurations. We reduce the problem of verifying the access control mechanism of 5G core into a model checking problem. Our framework incorporates a modular design and several 5G system-specific abstraction mechanisms to ensure flexibility, customizability, and scalability of our analysis. Our testing upon 73 safety properties on 27 different core network configurations uncovered six new weaknesses in the 5G access control

mechanism which can be exploited by an compromised NF to gain unauthorized access to various sensitive resources. A research paper on our work is currently under review.

We responsibly disclose one of our findings in the following.

2 Token Reuse Attack

2.1 Main idea

Suppose an NF service consumer has obtained an access token for an NF service producer, and the producer now wants to revoke the consumer's access. To do this, the producer updates the authorization attributes in its NF/NFservice profiles and invokes the *NFUpdate* API call with the Network Repository Function (NRF). After the update, the consumer should no longer have access to the producer. However, our analysis has revealed that the consumer can still access the producer's services using the previously issued access token, as long as it has not yet expired.

Unfortunately, there is currently no way for the service producer to determine whether the access token was created before or after a consumer's access revocation. This presents a potential security vulnerability that should be addressed to ensure the integrity and confidentiality of the service provider's data.

2.2 Example demonstrating the vulnerability

Consider a partial core network setup (figure 1) where a compromised NF Service consumer C_1 wants some services from a NF Service producer P_2 registered in NRF.

Step ①: Consumer C_1 sends *accessTokenRequest* to NRF to access services of producer P_2 .

Step ②: NRF verifies the request and provides an access token t to consumer C_1 .

Step ③: Now producer P_2 updates the authorization attributes in its NF/NFService profiles and invokes the *NFUpdate* API call with NRF to commit the changes.

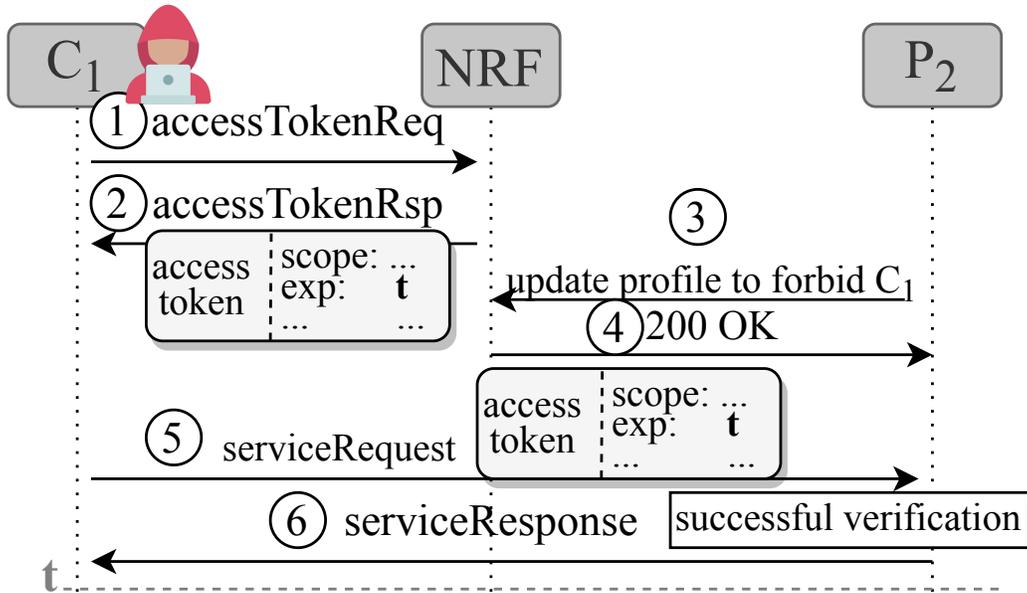


Figure 1: Demonstration of Token Reuse Attack

- Step ④: NRF acknowledges the change in NF/NFService profiles. From now on, producer P_2 should no longer approve the consequent service requests from consumer C_1 .
- Step ⑤: Anyway, consumer C_1 sends a service request with access token t to producer P_2 .
- Step ⑥: Producer P_2 verifies the access token t and provides access to the requested service to consumer C_1 provided the token is still unexpired. Producer P_2 cannot determine, by looking into access token t , if the token has become obsolete or not.

2.3 Adversary Assumption

No additional adversarial ability is required to successfully carry out this attack.

2.4 Root Cause

As the access token has been provided to the consumer NF before its access has been revoked, the producer NF or NRF can no longer force a malicious consumer NF not to use the old (but unexpired) access token. As the producer only checks the access token attributes and does not perform any other authorization checks, just by verifying the access token attributes, it cannot know if the token is expired or not.

2.5 Possible Fixes

From our speculation, there are two possible fixes to address the vulnerability.

1. **Introducing token verification interaction between producer NF and NRF.** To determine whether the consumer NF is using an obsolete access token or not, upon receiving *NFServiceRequest* from the consumer (step ⑤ in figure 1), the producer NF may ask NRF via a new API call *TokenVerificationRequest (token)* to verify whether the token is still valid or not. If the response from the NRF is positive, only then the producer NF shall provide resource access to the consumer NF.

This fix is similar to the one discussed in Solution #1 in 3GPP TR 33.855 [1]. However, we **do not recommend** this fix because, for each service access, the introduction of new NF interactions between the producer NF and NRF adds significant overhead in both the NRF and the producer NF which is too costly to realize.

2. **Introducing a new attribute in the access token.** This de-synchronization vulnerability can be solved using a new attribute in the access token which we call *timestamp*. This attribute will hold the time when the access token is created. Additionally, the producer NF shall also keep track of the most recent time of NF/NFService profile update in a variable (e.g. *lastUpdateTime*). Whenever, the producer receives *NFServiceRequest* from the consumer with an access token (step ⑤ in figure 1), it will verify *timestamp* of the access token against *lastUpdateTime* in addition to other checks. The producer will provide resource access to the consumer NF only after all the checks are successful. We recommend this fix over the one as it does not require the introduction of new NF interactions, thus, does not add significant overhead.

References

- [1] 3GPP. Study on security aspects of the 5G Service Based Architecture (SBA). TR 33.855. Version 16.1.0.