Security analysis of NWDAF analytics request response

This report in another language: 中文.

19.8.0 · ProVerif+ Agent report

Documents	5G System; Network Data Analytics Services; Stage 3 (protocol definition)
Properties examined	11: 10 hold, 1 fail (1 attack)
Findings	1; known results reproduced in Appendix B: 1

Contents

1 Protocol and scope
2 Findings
3 Properties examined
4 Limits of this analysis
Appendix A: Reading of each property
Appendix B: Known results reproduced
Appendix C: Claims and prover answers
Appendix D: How the analysis was run

1 Protocol and scope

The Nnwdaf_AnalyticsInfo API lets one NF service consumer (for example the PCF) ask one NWDAF for aggregated analytics about the network, here the network performance of a network area or slice, by one HTTP GET on the analytics resource. The consumer names the analytics with the event-id and may add reporting requirements such as the time window in ana-req; the NWDAF answers with AnalyticsData in a 200 OK, or with 204 No Content when it holds no such data. Access is authorised with OAuth2: the NRF issues the consumer an access token for the nnwdaf-analyticsinfo scope and the NWDAF checks it, over a mutually authenticated TLS connection between the two NFs.

When the exchange ends the consumer is sure that the analytics it holds answer its own request and were produced by the NWDAF it addressed, not by another NF and not by the network. The NWDAF is sure that the request came from the NF instance named in the access token, that the token was issued by its NRF, names the NWDAF as audience, carries the scope of the requested operation and has not expired, and that no other NF has read or altered the exchange.

The parties:

- **NF service consumer**: e.g. PCF, requests analytics. Holds: NF instance id, TLS client certificate, NRF public key.
- **NWDAF**: producer of the analytics. Holds: NF instance id, TLS server certificate, NRF public key, network measurement data.
- **NRF**: authorization server (OAuth2). Holds: NF profile of the consumer, token signing key, own identity.

Figure 1 shows the exchange as the documents specify it.

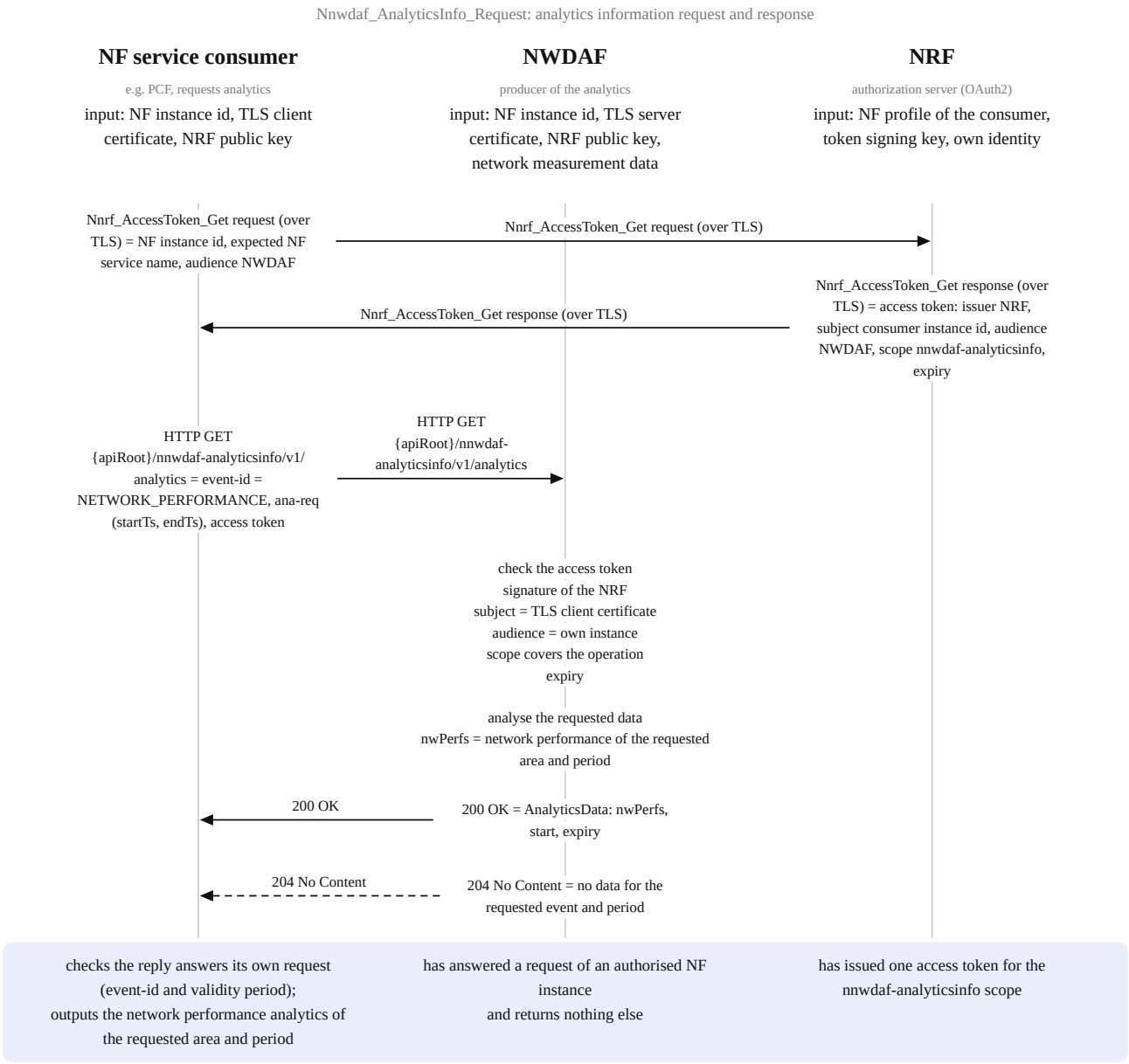


Figure 1. The protocol as specified.

The scope of the analysis is in Appendix D: what was analysed, what was left out, and the sections of the documents it is built from.

2 Findings

The analysis has 1 finding: what follows is not stated in the documents. For 1 of them it is unclear whether the documents state it. 1 result that the documents already state is not counted and is in Appendix B.

Within the modelled subflow the API gives what the documents require: the response binding, the token checks and the separation between NFs all hold against the network, against a malicious legitimate NF and against the holder of a stolen token. The one finding is a scope ambiguity: read literally, Table 5.2.9-1 describes nnwdaf-analyticsinfo as access to the API as a whole, and a producer that implements that reading serves the context data of analytics subscriptions to a token that carries only that scope. No other weakness was found; the known OAuth2 bearer-token gap is reproduced and shown to be closed by the TLS subject binding that TS 33.501 requires. All eleven targets are

answered, ten of them proved for every attacker set for them, with controls; the two readings of the scope question give one proof and one attack. The results rest on an idealised TLS transport, a simplified NRF authorization policy and a single analytics event type, so they are proofs about this model rather than about every deployment.

The attack figures use the columns of Figure 1. Red is what the attacker does. Yellow marks a message that a party sends although the specified exchange does not have it at that point, and the outcome that the attack breaks.

2.1 The analytics scope may cover context transfer

Kind	Weakness of the protocol as specified
New	Unclear: the documents state it in part, or could not be searched
Value	medium
Properties	Z05 (section 3)
Source text	s0460, s0458, s0324, s0347
Shown by	ProVerif attack query `z05b`; Figure 2

What happens. Table 5.2.9-1 describes the scope nnwdaf-analyticsinfo as "Access to the Nnwdaf_AnalyticsInfo API" and defines a second scope, nnwdaf-analyticsinfo:contexttransfer, for the ContextTransfer operations, without saying that the first scope alone excludes them. A producer that implements the first description literally serves a ContextTransfer request that presents a token carrying only nnwdaf-analyticsinfo, and returns the context information of analytics subscriptions. The model shows both readings: with a per-operation scope check the request is refused, without it the context data is served.

What the attacker needs. A legitimate NF instance with its own TLS client certificate and an access token for the nnwdaf-analyticsinfo scope, and a producer that reads the API-level scope as covering every operation of the API. No stolen credential is needed.

What the attacker gains. The NF obtains the context the NWDAF keeps about the analytics subscriptions of other consumers, which is the data the separate contexttransfer scope exists to protect, and can use it to learn which analytics other NFs subscribed to and with which filters.

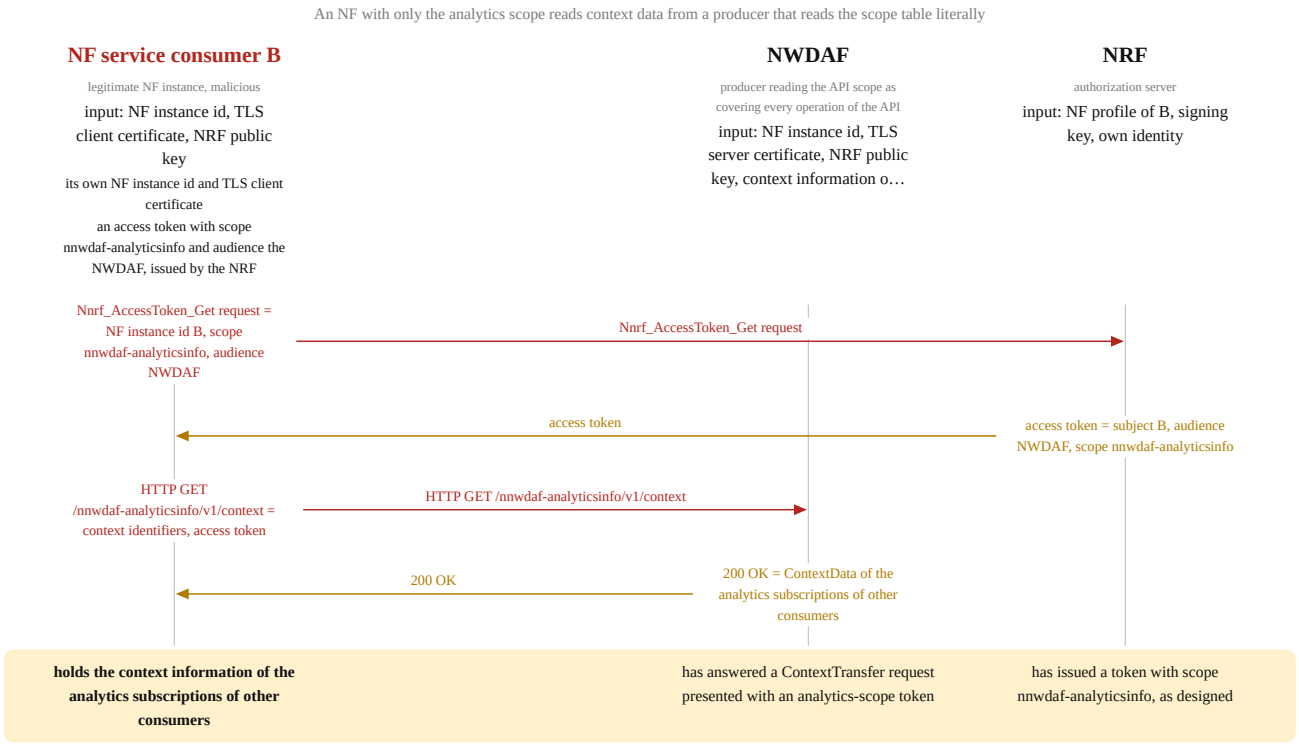


Figure 2. An NF with only the analytics scope reads context data from a producer that reads the scope table literally. Property Z05; attack query z05b.

Proposed change. Say in Table 5.2.9-1 that nnwdaf-analyticsinfo covers the analytics resource operations only and that the ContextTransfer operations require nnwdaf-analyticsinfo:contexttransfer, and require the producer to check the scope against the requested operation (TS 33.501 clause 13.4.1.1.2 step 2 already asks for this). Checked on a variant of the model: with that change the analytics-scope request for context data is refused, the normal run is still found and all other recorded answers still hold.

Check of the change with ProVerif. The change was put into a copy of the model: Table 5.2.9-1 is made unambiguous: the scope "nnwdaf-analyticsinfo" covers the analytics resource operation only, and the ContextTransfer operations require "nnwdaf-analyticsinfo:contexttransfer". The NWDAF consequently checks the token scope against the requested operation (context transfer needs the contexttransfer scope), and the NRF can issue that scope to an NF that is authorised for context transfer. On the changed model ProVerif proves `z05b`: the attack is no longer possible there. The normal run of the protocol is still found. The other 16 recorded answers are unchanged. This is a first check on the model: it shows that the change removes this attack there, not that it introduces no other problem. Changed model: `records/fixes/0041-fix-scope.pv`.

3 Properties examined

Each row is one property put to ProVerif. The result is ProVerif's answer on the analyst's model of the protocol; value and whether the documents already state the result are the analyst's judgement, with the reasons in Appendix A.

ID	Property	Compromise assumed	Result	Value	In the documents
A03	**Request answered as requested.** Whenever the NWDAF accepts an analytics request and returns 200 OK, there is a matching real run of the requester, and both sides agree on the analyticsId, the request filter conditions and the requester identity.	no compromise	Holds	medium	yes
A08	**Response bound to request and requester.** The analyticsId, filter conditions and requester identity corresponding to an analytics response returned by the NWDAF match some real request, and the response is delivered only to that requester.	no compromise	Holds	medium	yes
D01	**Accepted analytics is a genuine 200 OK.** Every time a consumer accepts analytics data, it corresponds to a genuine send by the NWDAF, with matching payload and status code; an error response is never accepted as a successful result.	no compromise	Holds	low	yes
K01	**Access token stays with its NF.** An attacker cannot obtain an OAuth2 access token belonging to another NF.	no compromise	Holds	medium	yes
Z01	**200 OK needs a valid token.** Before the NWDAF returns an analytics response, the requester must hold a valid token issued by the NRF that covers the nnwdaf-analyticsinfo scope.	no compromise	Holds	medium	yes
Z03	**Token audience and subject binding.** The audience and scope of a token accepted by the NWDAF match the service and operation of this request, and the token cannot be reused by another NF or service.	no compromise	Holds	medium	yes
Z05	**Analytics scope does not reach context data.** A legitimate NF holding only the nnwdaf-analyticsinfo scope cannot obtain context transfer related data or other domain assets.	no compromise	Fails (1 attack)	medium	unclear
A08.8	**Response tied to the real request.** The analyticsId, filter conditions and requester identity corresponding to a response returned by the NWDAF match some real request.	no compromise	Holds	low	yes
U01	**No other NF's TLS response.** A legitimate NF will not accept, in its TLS session, a response belonging to another NF's TLS session.	no compromise	Holds	medium	yes
U02	**Error response not accepted as analytics.** A successful analytics result accepted by a consumer corresponds only to a 200 OK response of the NWDAF; an error response is never accepted as a successful result.	no compromise	Holds	low	yes
X01	**NRF issues token to its owner.** The NRF issues an access token whose subject claim is a given NF instance id only to a token request that the NRF has authenticated as that same NF instance; no other NF and no attacker on the network can obtain a token naming that NF.	a legitimate NF instance id, its own TLS client certificate and private key, and its own access tokens obtained from the NRF; it may address any service operation of the API with any parameters	Holds	medium	yes

Goals the documents state that the drafted list of properties did not have, added by the analyst (Appendix D): X01.

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: Response tied to analyticsId/request conditions/requester identity	Yes, but the association is carried by the transport, not by the response body: the accepted analytics answers the event-id and token of that NF's own request, and the requesting identity is the one in its TLS client certificate. AnalyticsData contains no field naming the request, only the validity period start/expiry, so an implementation must keep the HTTP request/response pairing (the HTTP/2 stream) with the response.	A03, A08, A08.8
Q2: A legitimate NF cannot receive another TLS session's response	Yes: a response is encrypted and signed under a session key that the requesting NF generated for that request, so another NF cannot read it or have it accepted in its own session. The result rests on the modelled TLS session and says nothing about a deployment where TLS is terminated at an intermediary.	U01, A08
Q3: An error response is not accepted as a successful analytics result	Yes: a 204 No Content answer is recorded as an error with its status code and never accepted as an analytics result, because success and error replies are distinguished before the body is used. The model does not cover the 400 and 500 bodies, whose problemDetails fields are not modelled.	U02, D01

4 Limits of this analysis

A result here is a result on a model of the protocol: it says what the specified messages and checks give or fail to give, with cryptography taken as ideal, and nothing about an implementation.

What this analysis does not cover, as the analyst states it:

- The SBI transport is idealised: a fresh session key is shipped under the producer's public key and every later message is encrypted and signed under it. The TLS handshake, certificate chain validation, record replay protection and forward secrecy of session keys are not modelled.
- The NRF's authorization policy is simplified: it grants the analytics scope to any authenticated NF and never issues the contexttransfer scope, so per-NF and per-slice authorization is not analysed.
- Token expiry is a fixed validity window compared by equality; revocation, clock skew and the reuse of a stored token during its validity time are only touched by the phase-1 leak.
- The request filter is represented by the event-id alone; the time window, network area and slice attributes are not modelled, and of the error responses only 204 No Content is modelled (400, 403, 500 and the problemDetails causes are not).
- Only direct communication between consumer and producer is modelled; an SCP or SEPP in the path, roaming and indirect communication are out of scope, and with them the trust those intermediaries require.
- There is one NWDAF and one analytics event type; multi-NWDAF aggregation, context transfer as a full flow and UE-specific analytics are not modelled beyond the scope check of Z05.
- The strength of the primitives, the entropy of tokens and offline guessing are not modelled; signatures and public-key encryption are perfect in the model.

What stays open for each property, and what would settle it, is in Appendix A.

Appendix A: Reading of each property

The analyst's own reading of the prover's answers, not an answer of the prover: what the answers mean for the protocol, what the reading rests on, whether it can be used as it stands, and what it is worth.

A03: Request answered as requested

Whenever the NWDAF accepts an analytics request and returns 200 OK, there is a matching real run of the requester, and both sides agree on the analyticsId, the request filter conditions and the requester identity.

The NWDAF binds every accepted request to the NF instance that opened the TLS session, taken from its client certificate, and its 200 OK answers that request's event-id and token. Neither the network, nor another NF, nor the holder of A's token can make a request be attributed to a different NF.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It confirms the identity and request binding the documents require and shows which check carries it; no simplification of a primitive touches the result, but the TLS identity binding is an assumption of the model.
- Stated in the documents: yes. Where the documents say it: ts-33-501:8028, s0008, s0248
- Rests on: the consumer identity comes from a TLS client certificate issued by a CA to the NF instance id; the request carries no requester identity of its own; the filter is represented by the event-id only; the responder checks the token subject against that certificate, as TS 33.501 requires for direct communication
- Next step: Model the time window and area filters of ana-req to ask the same agreement for a filter the NWDAF must echo, which the response body has no field for.

A08: Response bound to request and requester

The analyticsId, filter conditions and requester identity corresponding to an analytics response returned by the NWDAF match some real request, and the response is delivered only to that requester.

Every analytics datum a consumer accepts is the 200 OK the NWDAF produced for that consumer's own request, encrypted and signed under a key only that consumer and the NWDAF hold. Neither the network nor another NF can have a response accepted.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It is the central delivery guarantee of the exchange and holds without a simplification of a primitive, but it rests on the idealised TLS session.
- Stated in the documents: yes. Where the documents say it: s0248, s0315
- Rests on: the session key is fresh per request and known only to the two endpoints; correlation of response to request is modelled by the session key and a fresh nonce, not by a field of AnalyticsData
- Open: the response body has no field naming the request, so the binding is the transport's; the model does not model HTTP framing
- Next step: Model the HTTP/2 stream identifier explicitly to ask the same property over a reused TLS connection with several concurrent requests.

D01: Accepted analytics is a genuine 200 OK

Every time a consumer accepts analytics data, it corresponds to a genuine send by the NWDAF, with matching payload and status code; an error response is never accepted as a successful result.

An analytics payload a consumer accepts was carried by a genuine 200 OK of the NWDAF with the same event-id, and a 204 answer never becomes an analytics result. The status code and the payload cannot be separated.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: It confirms a requirement the documents state; the interesting part (the response came from the NWDAF at all) is the same result as A08.
- Stated in the documents: yes. Where the documents say it: s0248, s0253
- Rests on: the model distinguishes success and error bodies by a tag inside the encrypted, signed reply; 400 and 500 problemDetails bodies are not modelled
- Next step: Add the 400 and 500 branches with their problemDetails cause to see whether any of them can be confused with a success body.

U02: Error response not accepted as analytics

The analytics success result accepted by the consumer corresponds only to a 200 OK response from the NWDAF; error responses are not accepted as success results.

A 204 No Content answer is recorded as an error with its status code and never as analytics, and an error the consumer records corresponds to a real error response of the NWDAF. The model has no path that turns an error into a success.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: It answers a validation requirement of TS 33.501 5.9.2.1 for the one error branch that is modelled.
- Stated in the documents: yes. Where the documents say it: ts-33-501:2875, s0253, s0315
- Rests on: only 204 is modelled among the error responses; success and error are told apart inside the authenticated, encrypted reply
- Next step: Model the 400 and 500 responses to ask whether a requester can be made to treat a ProblemDetails body as analytics.

K01: Access token stays with its NF

An attacker cannot obtain an OAuth2 access token that belongs to another NF.

A's access token never reaches the network or another NF while it is in use: it travels only inside the TLS session with the NRF and the one with the NWDAF. When a stored token is read after the run the attacker has it, which is the expected cost of storing a bearer token.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It shows the token is protected where the documents require it and shows exactly what the storage exposure gives away, but token storage is only modelled as a post-run leak.
- Stated in the documents: yes. Where the documents say it: rfc6749:2859, s0455, s0456
- Rests on: the token request and the service request are carried in the modelled TLS session; the leak of a stored token is modelled after the run, not during it
- Open: whether a token read from storage can be used while A is still running is not modelled

- Next step: Model concurrent use of a stored token to ask whether a stolen token can be used while its owner is active.

Z01: 200 OK needs a valid token

Before the NWDAF returns an analytics response, the requester must hold a valid token issued by the NRF that covers the nnwdaf-analyticsinfo scope.

Every 200 OK rests on an NRF-signed token whose scope covers the requested operation; without such a token the NWDAF answers nothing. A forged, expired or wrong-scope token is refused whatever the requester claims.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It is the authorization guarantee the API rests on, proved with a control in which a producer without the token check is fooled; the NRF's own policy is simplified.
- Stated in the documents: yes. Where the documents say it: ts-33-501:8038, s0455, s0456, s0460
- Rests on: the NRF grants the analytics scope to any authenticated NF; expiry is a fixed validity window, revocation is not modelled
- Next step: Model the NRF's per-slice and per-NF-type authorization to ask what a token issued for one slice lets its holder do.

Z03: Token audience and subject binding

For a token the NWDAF accepts, its audience and scope match the service and operation of this request, and it cannot be reused by another NF or service.

The NWDAF accepts a token only if its audience names this NWDAF instance and its subject is the NF instance in the TLS client certificate of the session. An NF that presents a token naming another NF is refused, so a token read from another NF's storage cannot be reused.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It shows the exact check that closes the bearer-token gap RFC 6749 names, with a control in which a producer without the subject check serves the stolen token; the instance-level audience is the strongest reading.
- Stated in the documents: yes. Where the documents say it: rfc6749:2879, ts-33-501:8028, s0455
- Rests on: the audience is modelled at NF instance level; the NF-type audience that TS 33.501 also permits would let any instance of the type accept the token; the client certificate binds the NF instance id to its key; the token is a signed tuple with one scope
- Next step: Model the NF-type audience and several NWDAF instances to ask whether a token meant for the type can be replayed at another instance of the same type.

Z05: Analytics scope does not reach context data

A legitimate NF holding only the nnwdaf-analyticsinfo scope cannot obtain context transfer related data or other domain assets.

Under the reading that the producer checks the token scope against the requested operation, an NF holding only nnwdaf-analyticsinfo obtains no context data. Under the other reading of Table 5.2.9-1, which describes that scope as access to the API as a whole, the producer answers a ContextTransfer request with the context information of analytics subscriptions.

- Result: Fails (1 attack); answers what is asked: partly; usable with caveats
- Value medium: It turns a scope table that can be read two ways into a concrete difference of behaviour, which is what an editor needs; the attack is a reading, not a demonstrated violation of the text.
- Stated in the documents: unclear. Source text it rests on: s0460, s0458, s0324
- Rests on: the context data is modelled as one value of the NWDAF, not as per-subscription records with their own authorization; the NRF never issues the contexttransfer scope, so the only token in play is the analytics one; the context store is modelled as one value, so the attack shows access, not which records were selected
- Open: whether the intended reading of Table 5.2.9-1 makes the API scope exclude ContextTransfer is an editorial question the model cannot settle; the context flow is modelled only as one request answered with the context store
- Next step: A clarification of Table 5.2.9-1; the recommended wording was checked on a variant of the model and removes the attack.

A08.8: Response tied to the real request

The analyticsId, filter conditions and requester identity that the response returned by the NWDAF corresponds to are consistent with some real request.

The response a consumer accepts answers a request with that event-id and filter, and only the consumer that made the request receives it. The analyticsId and the filter are the ones the NWDAF accepted, not ones supplied with the answer.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: It is the same result as A08 asked in the words of the user question.
- Stated in the documents: yes. Where the documents say it: s0008, s0248
- Rests on: the filter is represented by the event-id; the binding is the session key and a fresh nonce, since the body has no request field
- Next step: Model the ana-req time window to ask the same question for a filter the response body does not carry.

U01: No other NF's TLS response

A legitimate NF does not accept, in its own TLS session, a response that belongs to another NF's TLS session.

A consumer accepts a response only inside the session that requested it: the response is encrypted under a fresh key of that request and the consumer checks its own nonce and event-id. Another NF's session response is neither readable nor acceptable to it.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It is the mutual-authentication requirement of TS 33.501 5.9.2.1 for this exchange, and it holds without a simplification of a primitive, but it rests on the idealised transport.
- Stated in the documents: yes. Where the documents say it: ts-33-501:2871, s0455
- Rests on: the session key is generated by the requesting NF and shipped under the producer's public key; TLS is assumed to be end to end between the two NFs
- Next step: Model an SCP in the path to ask what the requirement gives when TLS is not end to end.

X01: NRF issues token to its owner

The NRF issues an access token whose subject claim is a given NF instance id only to a token request that the NRF has authenticated as that same NF instance; no other NF and no attacker on the network can obtain a token naming that NF.

The NRF issues a token naming a given NF instance only on a token request from a session authenticated as that same instance. No other NF and no attacker on the network obtains a token naming A, so the subject of a token can be trusted by the producer.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It is the premise the Z03 subject check depends on, and it is proved rather than assumed; the NRF's policy is simplified.
- Stated in the documents: yes. Where the documents say it: ts-33-501:7964, s0456
- Rests on: the NRF authenticates the requester by its TLS client certificate and compares the requested subject with it; the NRF's authorization policy beyond that comparison is simplified
- Next step: Model the NRF's profile checks (NF type, slices) to ask the same question for those parameters.

Appendix B: Known results reproduced

A property that fails as the documents already state, or that they set aside, is not a finding. What follows is of that kind: the analysis reproduces it on the model, which gives the exact conditions.

B.1 A stolen access token cannot be used by another NF

Kind	Guarantee confirmed
Stated in the documents	rfc6749:2879, ts-33-501:8028
Value	medium
Properties	Z03 (section 3)

What happens. RFC 6749 notes that OAuth2 gives the resource server no way to tell that the token presented to it was issued to the presenting client; in this API a token that A stored and that an attacker read is presented by another NF with its own certificate. The NWDAF refuses it because the subject claim of the token is compared with the NF instance in the TLS client certificate, as TS 33.501 requires for direct communication. The model shows the presentation and the refusal in the same run.

What the attacker needs. The attacker has an access token read from another NF's storage and its own TLS credentials; it does not have the token owner's private key.

What the attacker gains. Nothing: the token stays bound to the NF it was issued to, so the bearer-token weakness that OAuth2 leaves open does not carry over to this API as long as the subject check is implemented.

Appendix C: Claims and prover answers

The prover's record for each target: the level reached, the query that reached it, whether a control was checked, and the attacks with the analyst's verdict on each.

target	level	bound	lemma	control	attacks	claims
A03	proved		a03_id	checked		6 proved
A08	proved		a08	checked		3 proved
D01	proved		a08	checked		2 proved
K01	proved		k01	checked		2 proved
Z01	proved		z01a	checked		6 proved
Z03	proved		z03a	checked		6 proved
Z05	proved		z05a	checked	z05b (depends_on_ambiguity)	2 proved, 1 attack
A08.8	proved		a08	checked		2 proved
U01	proved		a08	checked		3 proved
U02	proved		a08	checked		4 proved
X01	proved		x01	checked		4 proved

Claim by claim, in the analyst's wording of each claim; the answer to each is the prover's:

target	claim	attacker	answer	query	control
A03	Requester identity: whenever the NWDAF accepts a request, the NF instance it attributes the request to is the instance that opened that request (the identity in its TLS client certificate), not another NF.	`network`	proved	a03_id	checked
A03	Requester identity, against a legitimate but malicious NF with its own certificate and tokens: the identity the NWDAF binds is still the instance that opened the request, not the one the attacker would name.	`malicious_nf`	proved	a03_id	checked
A03	Requester identity, against an attacker holding A's stored access token but not A's TLS key: a request it sends is never attributed to A.	`stolen_token`	proved	a03_id	checked
A03	AnalyticsId and filter agreement: a 200 OK answers a request the NWDAF accepted with the same event-id and the same access token.	`network`	proved	a03_ev	checked
A03	AnalyticsId and filter agreement against a malicious legitimate NF: a 200 OK still answers an accepted request with the same event-id and token.	`malicious_nf`	proved	a03_ev	checked
A03	AnalyticsId and filter agreement against the holder of A's stored token: a 200 OK still answers an accepted request with the same event-id and token.	`stolen_token`	proved	a03_ev	checked
A08	The response body a consumer accepts is exactly the 200 OK the NWDAF sent to that consumer for that request; the network never gets a response accepted on another NF.	`network`	proved	a08	checked

target	claim	attacker	answer	query	control
A08	A malicious legitimate NF cannot make another NF's consumer accept a response: an accepted response was sent to that very consumer.	`malicious_nf`	proved	a08	checked
A08	An attacker that holds A's access token but not A's TLS key cannot make a consumer accept a response the NWDAF produced for another request.	`stolen_token`	proved	a08	checked
D01	An analytics payload the consumer accepts was carried by a genuine 200 OK of the NWDAF with the same payload and the same event-id.	`network`	proved	a08	checked
D01	A malicious legitimate NF cannot make a consumer accept an analytics payload the NWDAF did not send in a 200 OK to it.	`malicious_nf`	proved	a08	checked
K01	The access token of NF service consumer A is never known to the attacker during the run.	`network`	proved	k01	checked
K01	Another NF service consumer, with its own credentials and TLS session, never learns A's access token.	`malicious_nf`	proved	k01	checked
Z01	Every 200 OK rests on an access token that the NRF issued.	`network`	proved	z01a	checked
Z01	A malicious legitimate NF obtains no 200 OK without an NRF-issued access token.	`malicious_nf`	proved	z01a	checked
Z01	An attacker holding a stolen token still obtains no 200 OK except on a token the NRF really issued.	`stolen_token`	proved	z01a	checked
Z01	The token a 200 OK for analytics rests on carries the nnwdaf-analyticsinfo scope, i.e. the scope of the requested operation.	`network`	proved	z01b	checked
Z01	A malicious legitimate NF obtains no analytics with a token whose scope does not cover the operation.	`malicious_nf`	proved	z01b	checked
Z01	A stolen token is only accepted where its scope covers the requested operation.	`stolen_token`	proved	z01b	checked
Z03	The token a 200 OK rests on names this NWDAF instance as its audience.	`network`	proved	z03a	checked
Z03	A malicious legitimate NF obtains no 200 OK on a token whose audience is another service or instance.	`malicious_nf`	proved	z03a	checked
Z03	A stolen token is only accepted by the NWDAF instance its audience names.	`stolen_token`	proved	z03a	checked
Z03	The token a 200 OK rests on names as subject exactly the NF instance in the TLS client certificate of that session.	`network`	proved	z03b	checked
Z03	No NF can present a token naming another NF as subject and be served.	`malicious_nf`	proved	z03b	checked
Z03	Holding A's access token does not let another NF use it: the subject is checked against the TLS client certificate, so the token cannot be reused by another NF.	`stolen_token`	proved	z03b	checked
Z05	A conformant NWDAF that checks the scope per operation serves no context data to a token that carries only nnwdaf-analyticsinfo.	`network`	proved	z05a	checked

target	claim	attacker	answer	query	control
Z05	A legitimate NF holding only the nnwdaf-analyticsinfo scope obtains no context data from a producer that checks the scope per operation.	`malicious_nf`	proved	z05a	checked
Z05	Second reading of Table 5.2.9-1: a legitimate NF holding only the nnwdaf-analyticsinfo scope obtains the context-transfer data of the NWDAF, i.e. other consumers' analytics subscription context.	`malicious_nf`	attack (depends_on_ambiguity)	z05b (the control of z05a)	
A08.8	The delivered response answers a request with that analyticsId and filter, and only that requester receives it.	`network`	proved	a08	checked
A08.8	A malicious legitimate NF cannot obtain a response that answers another NF's request, nor pass its own request off as another's.	`malicious_nf`	proved	a08	checked
U01	A legitimate NF never accepts, inside its own TLS session, a response the NWDAF produced for another NF's session.	`network`	proved	a08	checked
U01	A malicious legitimate NF cannot deliver another NF's session response into its own TLS session.	`malicious_nf`	proved	a08	checked
U01	Holding A's stored access token does not let the attacker receive A's session response.	`stolen_token`	proved	a08	checked
U02	An error response (204 No Content) is never accepted as an analytics result: every accepted analytics is a 200 OK.	`network`	proved	a08	checked
U02	A malicious legitimate NF cannot get an error response accepted as an analytics result, not even by choosing the event-id.	`malicious_nf`	proved	a08	checked
U02	An error the consumer records comes from a real NWDAF error response with the same status code.	`network`	proved	u02	checked
U02	An error the consumer records comes from a real NWDAF error response with the same status code, also when a malicious NF drives the request.	`malicious_nf`	proved	u02	checked
X01	The NRF issues a token naming a given NF instance only on a token request from that same instance.	`network`	proved	x01	checked
X01	A legitimate but malicious NF cannot obtain a token that names another NF instance as subject.	`malicious_nf`	proved	x01	checked
X01	No attacker on the network obtains a token naming NF service consumer A.	`network`	proved	k01	checked
X01	No other NF obtains a token naming NF service consumer A.	`malicious_nf`	proved	k01	checked

Appendix D: How the analysis was run

An LLM analyst builds the model of the protocol from the documents and puts each property to ProVerif; every result in section 3 is an answer of the prover.

ProVerif+ Agent session session-20261114T215055, 2026-11-14. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 47, prover 9 s, wall 16 min, cost 0.142 USD.

ProVerif: 97 proofs run, 97 decided, 0 without an answer, 7 s, 42 answers recorded.

What was analysed

Protocol	NWDAF analytics request response
Version	19.8.0
Analysed	One NF requests aggregated network performance analytics from one NWDAF; a single request/response with explicitly relevant filter conditions; excludes context transfer, external collection, multi-NWDAF aggregation, and UE-specific inputs.
Not analysed	The symbolic model does not prove computational security or post-quantum security of the algorithms; attack traces hold only under explicit assumptions and are not directly equivalent to specification vulnerabilities. Each property must have a reachability/positive-case control. Fixed to Rel-19. The external network attacker and the malicious legitimate NF holding its own credentials are distinguished separately; TLS authenticated encryption is modelled per endpoint, and a legitimate NF cannot read another NF's TLS session. OAuth2 subject/audience/scope/expiry and TLS identity binding follow the attached baseline; a service scope does not automatically equal resource ownership. Monotonic tables cannot model real deletion, and the finite-state lifecycle must be explicitly delimited. The goal is the specified API sub-flow, not complete 5G security.
Source text	460 numbered passages (154966 bytes) from the sections listed below. A reference such as s0001 in this report is one of these passages

The sections of the documents the analysis is built from:

Document	Section	Title
main	4.3.2.2	Nnwdaf_AnalyticsInfo_Request service operation
main	4.3.2.2.1	General
main	4.3.2.2.2	Request and get from NWDAF Analytics information
main	5.2	Nnwdaf_AnalyticsInfo Service API
main	5.2.1	Introduction
main	5.2.2	Usage of HTTP
main	5.2.2.1	General
main	5.2.2.2	HTTP standard headers
main	5.2.2.2.1	General
main	5.2.2.2.2	Content type
main	5.2.2.3	HTTP custom headers
main	5.2.3	Resources
main	5.2.3.1	Resource Structure

Document	Section	Title
main	5.2.3.2	Resource: NWDAF Analytics
main	5.2.3.2.1	Description
main	5.2.3.2.2	Resource definition
main	5.2.3.2.3	Resource Standard Methods
main	5.2.3.2.3.1	GET
main	5.2.3.2.4	Resource Custom Operations
main	5.2.3.3	Resource: NWDAF Context
main	5.2.3.3.1	Description
main	5.2.3.3.2	Resource definition
main	5.2.3.3.3	Resource Standard Methods
main	5.2.3.3.3.1	GET
main	5.2.3.3.4	Resource Custom Operations
main	5.2.4	Custom Operations without associated resources
main	5.2.5	Notifications
main	5.2.6	Data Model
main	5.2.6.1	General
main	5.2.6.2	Structured data types
main	5.2.6.2.1	Introduction
main	5.2.6.2.2	Type AnalyticsData
main	5.2.6.2.3	Type EventFilter
main	5.2.6.2.4	Void
main	5.2.6.2.5	Type AdditionInfoAnalyticsInfoRequest
main	5.2.6.2.6	Type ContextData
main	5.2.6.2.7	Type ContextElement
main	5.2.6.2.8	Type ContextIdList
main	5.2.6.2.9	Type HistoricalData
main	5.2.6.2.10	Type SpecificAnalyticsSubscription
main	5.2.6.2.11	Type RequestedContext
main	5.2.6.2.12	Type SmcceInfo
main	5.2.6.2.13	Type SmcceUeList
main	5.2.6.2.14	Type SpecificDataSubscription
main	5.2.6.2.15	Type UserDataCongestReq
main	5.2.6.2.16	Type NetworkPerfReq
main	5.2.6.2.17	Type ResourceUsageRequPerNwPerfType
main	5.2.6.2.18	Type AnalyticsAccuracyInfo
main	5.2.6.2.19	Type GroundTruthInfo

Document	Section	Title
main	5.2.6.2.20	Type MlModelAccuracyInfo
main	5.2.6.3	Simple data types and enumerations
main	5.2.6.3.1	Introduction
main	5.2.6.3.2	Simple data types
main	5.2.6.3.3	Enumeration: EventId
main	5.2.6.3.4	Enumeration: ContextType
main	5.2.6.3.5	Enumeration: AdrfDataType
main	5.2.6.4	Data types describing alternative data types or combinations of data types
main	5.2.6.4.1	Type ProblemDetailsAnalyticsInfoRequest
main	5.2.7	Error handling
main	5.2.7.1	General
main	5.2.7.2	Protocol Errors
main	5.2.7.3	Application Errors
main	5.2.8	Feature negotiation
main	5.2.9	Security

The purpose of the protocol as the source step located it in the documents: one NF requests aggregated network performance analytics from one NWDAF; a single request/response with explicitly relevant filter conditions; excludes context transfer, external collection, multi-NWDAF aggregation, and UE-specific inputs.

The properties against the documents

Before a query was written, the analyst set the drafted list of properties against what the documents themselves say the protocol is for. The name, what a property is about and what the documents say are the analyst's reading; the places named exist in the documents.

ID	Name	About	What the documents say	Where
A03	Request answered as requested	the NWDAF and the NF service consumer: the analyticsId (event-id), the request filter in ana-req and the requesting NF instance the NWDAF attributes the request to	The consumer sends the GET with the mandatory event-id and the ana-req filter (s0008), the NWDAF answers 200 OK with the analytics relevant for the requesting consumer (s0248), and access is authorized with OAuth2 (s0455).	s0008, s0248, s0455
A08	Response bound to request and requester	the NF service consumer and the AnalyticsData body of the 200 OK: which analyticsId and filter it answers and which consumer receives it	The 200 OK carries the analytics with the parameters relevant for the requesting NF service consumer (s0248, s0315); the response body has no field naming the request, so correlation is carried by the transport.	s0248, s0315
D01	Accepted analytics is a genuine 200 OK	the NF service consumer and the analytics payload with the HTTP status code of the response	The NWDAF answers 200 OK with AnalyticsData when the request is accepted and uses other status codes otherwise (s0248, s0253, s0315).	s0248, s0253, s0315

ID	Name	About	What the documents say	Where
K01	Access token stays with its NF	the OAuth2 access token of NF service consumer A, kept from the network and from the other NF service consumers	Access token credentials MUST be kept confidential in transit and storage and transmitted only over TLS (RFC 6749 10.3); the token claims the NF instance id of the consumer as subject (ts-33-501 13.4.1.1.2).	rfc6749:2859, ts-33-501:7968, s0455, s0456
Z01	200 OK needs a valid token	the NWDAF and the NRF-issued access token with the nnwdaf-analyticsinfo scope that the requesting consumer must hold	Access to the API may be authorized with OAuth2, the consumer obtains a token before consuming the service (s0455, s0456), the scope is nnwdaf-analyticsinfo (s0460), and the producer checks that the scope matches the requested operation (ts-33-501 13.4.1.1.2 step 2).	ts-33-501:8038, s0455, s0456, s0460
Z03	Token audience and subject binding	the NWDAF, the audience claim of the token (this NF instance) and the subject claim (the NF instance named in the consumer's TLS client certificate)	The token claims audience (NF type or instance) and subject (consumer instance id) (ts-33-501 7968, 7990); the producer checks audience against its own identity and, in direct communication, the subject against the TLS client certificate (ts-33-501 8028, 8032); RFC 6749 10.3 notes that OAuth2 alone gives no such binding.	ts-33-501:8028, ts-33-501:8032, rfc6749:2879, s0455
Z05	Analytics scope does not reach context data	the NWDAF Context resource (context information of analytics subscriptions) and an NF that holds only the nnwdaf-analyticsinfo scope	Table 5.2.9-1 defines nnwdaf-analyticsinfo as access to the Nnwdaf_AnalyticsInfo API and a second scope for the ContextTransfer operations (s0460); the context resource holds the analytics subscription context (s0324).	s0460, s0324, s0458
A08.8	Response tied to the real request	the analyticsId, the filter and the requester instance that the delivered response answers	The consumer requests analytics by event-id and ana-req filter (s0008) and the NWDAF answers with the analytics (s0248).	s0008, s0248
U01	No other NF's TLS response	two NF service consumers and the NWDAF response carried inside one consumer's TLS session	The NF service request and response procedure shall support mutual authentication for direct communication between consumer and producer (ts-33-501 5.9.2.1); the access is over the SBI transport with TLS per TS 33.501 and TS 29.500 (s0455).	ts-33-501:2871, s0455
U02	Error response not accepted as analytics	the NF service consumer and the difference between an error response (204 No Content, 400, 403, 500) and the 200 OK analytics result	204 No Content when the data does not exist and 400/403/500 with ProblemDetails otherwise (s0253, s0315); each NF shall validate all incoming messages (ts-33-501 5.9.2.1).	ts-33-501:2875, s0253, s0315
X01	NRF issues token to its owner	the NRF as authorization server and the subject claim of the access token it issues	The NRF authenticates the NF service consumer and shall verify that the NF instance id and type in the access token request match the public key certificate or the NF profile (ts-33-501 13.4.1.1.2 step 1, 8052); NRF and NFs requesting service shall be mutually authenticated (ts-33-501 5.9.2.2).	ts-33-501:2889, ts-33-501:7964, s0456

Added from the documents by the analyst: X01.

The security statements of the documents, paragraph by paragraph

The host listed the 398 paragraphs of the sections in which the documents say what the protocol protects and against what, and the analyst said of each what it made of it: a property, an attacker, or why neither. Taken up: 24. Not taken: 374. Which paragraphs are listed follows from the layout of the standard; where each went is the analyst's judgement.

Paragraphs	Where	Taken up as	Why not
P001 to P002	rfc6749:2778-2786	No goal stated	Introduction to the security considerations; it points to the OAuth threat model and states no requirement.

Paragraphs	Where	Taken up as	Why not
P003 to P006	rfc6749:2790-2823	Z03, K01, `malicious_nf`	
P007 to P010	rfc6749:2827-2850	Z03, A03, `malicious_nf`	
P011 to P015	rfc6749:2859-2881	K01, Z01, Z03, `network`, `stolen_token`	
P016 to P021	rfc6749:2885-2917	Outside the scope	Refresh tokens; the Client Credentials grant used by this API issues none.
P022 to P025	rfc6749:2921-2943	Outside the scope	Authorization code grant; not used by the Client Credentials grant of this API.
P026 to P029	rfc6749:2947-2980	Outside the scope	Redirection URI manipulation; no redirection in the Client Credentials grant.
P030 to P033	rfc6749:2984-3003	Outside the scope	Resource owner password credentials; no resource owner in this machine-to-machine grant.
P034 to P035	rfc6749:3013-3019	K01, U01, `network`	
P036	rfc6749:3023-3028	D01, U01, `network`	
P037 to P039	rfc6749:3032-3041	The prover cannot be asked	Guessing of tokens and credentials is about the entropy of generated values; the symbolic model treats tokens as unguessable names and cannot state the 2^-128 bound.
P040 to P042	rfc6749:3045-3066	Outside the scope	Phishing attacks on end users; there is no user agent in this API subflow.
P043 to P048	rfc6749:3070-3110	Outside the scope	Cross-site request forgery; no browser session and no cookie in this machine-to-machine grant.
P049 to P050	rfc6749:3119-3138	Outside the scope	Clickjacking; no user interface in this API subflow.
P051 to P052	rfc6749:3142-3150	The prover cannot be asked	Code injection and input validation are implementation defects of parsers; the symbolic model cannot express them.
P053 to P054	rfc6749:3154-3167	Outside the scope	Open redirectors; no redirection endpoint is used.
P055 to P062	rfc6749:3170-3210	Outside the scope	Implicit grant and resource owner impersonation; the implicit grant is not used by this API.
P063	ts-33-501:2491-2491	The prover cannot be asked	Bidding down of security features is about the negotiation of algorithms and features; the model fixes the primitives and cannot state it.
P064 to P072	ts-33-501:2495-2511	Outside the scope	UE subscription authentication, serving network authentication and authorization; the access stratum and the UE are outside this API subflow.
P073 to P074	ts-33-501:2515-2517	Outside the scope	Key lengths for AS and NAS protection; the access network is outside the modelled subflow.
P075 to P078	ts-33-501:2523-2529	Outside the scope	Ciphering and integrity protection between UE and ng-eNB.
P079 to P089	ts-33-501:2533-2553	Outside the scope	User data and signalling confidentiality on the access network and UE side.
P090 to P103	ts-33-501:2557-2583	Outside the scope	User data and signalling integrity on the access network and UE side.

Paragraphs	Where	Taken up as	Why not
P104 to P110	ts-33-501:2587-2599	Outside the scope	Secure storage of subscription credentials in the UE and the UDM; no subscription credentials are modelled.
P111 to P125	ts-33-501:2603-2631	Outside the scope	Subscriber privacy (SUPI, paging, location); UE identifiers are outside the modelled analytics subflow.
P126 to P136	ts-33-501:2637-2659	Outside the scope	gNB user data and signalling confidentiality requirements.
P137 to P147	ts-33-501:2663-2683	Outside the scope	gNB user data and signalling integrity requirements.
P148 to P156	ts-33-501:2687-2703	Outside the scope	gNB setup and configuration requirements.
P157 to P158	ts-33-501:2707-2709	Outside the scope	Key management inside the gNB.
P159 to P160	ts-33-501:2713-2715	Outside the scope	Handling of user plane data in the gNB.
P161 to P162	ts-33-501:2719-2721	Outside the scope	Handling of control plane data in the gNB.
P163 to P168	ts-33-501:2725-2735	Outside the scope	Secure environment of the gNB.
P169 to P174	ts-33-501:2739-2749	Outside the scope	F1 interface requirements of the gNB.
P175 to P176	ts-33-501:2753-2755	Outside the scope	E1 interface requirements of the gNB.
P177 to P178	ts-33-501:2759-2761	Outside the scope	Requirements on the ng-eNB.
P179 to P185	ts-33-501:2767-2779	Outside the scope	Signalling data confidentiality on the ng-eNB interfaces.
P186 to P192	ts-33-501:2783-2795	Outside the scope	Signalling data integrity on the ng-eNB interfaces.
P193 to P196	ts-33-501:2799-2805	Outside the scope	Subscriber privacy on the ng-eNB interfaces.
P197 to P198	ts-33-501:2809-2811	Outside the scope	Requirements on the SEAF.
P199 to P201	ts-33-501:2819-2823	Outside the scope	Generic UDM and SIDF requirements.
P202 to P207	ts-33-501:2827-2837	Outside the scope	Subscriber privacy requirements on UDM and SIDF.
P208 to P210	ts-33-501:2841-2845	Outside the scope	Requirements on the AUSF.
P211 to P213	ts-33-501:2849-2853	Outside the scope	Requirements on the UDR.
P214	ts-33-501:2859-2859	`network`	
P215	ts-33-501:2865-2865	Outside the scope	Confidentiality, integrity and replay protection of NF discovery and registration; service discovery is outside the modelled request/response subflow.

Paragraphs	Where	Taken up as	Why not
P216	ts-33-501:2867-2867	Outside the scope	Authorization of NF discovery and registration requests; discovery is outside the modelled subflow.
P217	ts-33-501:2869-2869	Outside the scope	Topology hiding between trust domains; the model covers a single operator domain and no discovery.
P218	ts-33-501:2871-2871	U01, A08, A03, D01, `network`, `malicious_nf`	
P219	ts-33-501:2873-2873	Outside the scope	The note removes mutual authentication for indirect communication through an SCP; the model covers the direct communication case only.
P220	ts-33-501:2875-2875	U02, A03, `network`, `malicious_nf`	
P221	ts-33-501:2879-2879	No goal stated	Description of what the NRF does; states no requirement.
P222	ts-33-501:2881-2881	X01, `malicious_nf`	
P223	ts-33-501:2883-2883	X01, `malicious_nf`, `network`	
P224	ts-33-501:2885-2885	Z01, Z03, K01	
P225	ts-33-501:2887-2887	No goal stated	Lead-in sentence to the list of NRF requirements.
P226	ts-33-501:2889-2889	K01, Z01, Z03, `network`, `malicious_nf`	
P227	ts-33-501:2891-2891	Z01	
P228 to P234	ts-33-501:2895-2907	Outside the scope	NEF and Application Function exposure requirements; the NEF and the AF are outside this NWDAF analytics subflow.
P235 to P241	ts-33-501:2911-2923	Outside the scope	Service Communication Proxy requirements; the model covers direct communication between consumer and producer.
P242 to P254	ts-33-501:2929-2953	Outside the scope	General requirements for end-to-end core network interconnection; roaming and SEPP are outside the modelled subflow.
P255 to P292	ts-33-501:2957-3031	Outside the scope	SEPP requirements for inter-PLMN interconnect; outside the single-operator subflow.
P293 to P303	ts-33-501:3035-3055	Outside the scope	Messages generated by roaming intermediaries; outside the modelled subflow.
P304 to P312	ts-33-501:3059-3075	Outside the scope	Protection of attributes on inter-PLMN interfaces; outside the modelled subflow.
P313 to P314	ts-33-501:3079-3081	Outside the scope	IPUPS functionality for inter-PLMN traffic; outside the modelled subflow.
P315 to P318	ts-33-501:3085-3091	Outside the scope	Requirements for NFs on interconnecting interfaces; outside the single-operator subflow.
P319 to P323	ts-33-501:3097-3105	Outside the scope	Configuration of signalling monitoring; out of scope of a single API request/response.
P324 to P327	ts-33-501:3109-3115	Outside the scope	Streaming of signalling monitoring data; out of scope.
P328 to P334	ts-33-501:3121-3133	Outside the scope	Security visibility and event logging; not a property of the API exchange.

Paragraphs	Where	Taken up as	Why not
P335 to P337	ts-33-501:3137-3141	Outside the scope	Security configurability and operator options; not a property of the API exchange.
P338 to P347	ts-33-501:3149-3167	The prover cannot be asked	Ciphering algorithm identifier values; the strength of the primitive is outside what a symbolic prover can state.
P348 to P357	ts-33-501:3171-3189	The prover cannot be asked	Integrity algorithm identifier values; the strength of the primitive is outside what a symbolic prover can state.
P358 to P369	ts-33-501:3193-3215	The prover cannot be asked	Algorithm selection and negotiation; the model fixes the primitives and cannot state a bidding-down property.
P370 to P371	ts-33-501:3219-3221	Outside the scope	Requirements on the 5G-RG; a wireline access entity outside the modelled subflow.
P372 to P375	ts-33-501:3225-3231	Outside the scope	Requirements on the NSSAAF; network slice authentication outside the modelled subflow.
P376 to P380	ts-33-501:6540-6548	Outside the scope	Security requirements on the N3 interface.
P381 to P382	ts-33-501:9656-9658	Outside the scope	EAP-TLS without subscription identifier privacy; UE access authentication outside the modelled subflow.
P383 to P385	ts-33-501:9662-9666	Outside the scope	EAP-TLS with subscription identifier privacy; UE access authentication outside the modelled subflow.
P386 to P388	ts-33-501:10893-10897	Outside the scope	Requirements on the IAB-node; access network entity.
P389	ts-33-501:10901-10901	Outside the scope	Requirements on the IAB donor; access network entity.
P390 to P392	ts-33-501:10905-10909	Outside the scope	Requirements on the 5GC supporting IAB; access network integration.
P393	ts-33-501:10913-10913	Outside the scope	Secure environment for IAB; access network entity.
P394	ts-33-501:10917-10917	Outside the scope	Requirements on the F1 interface for IAB.
P395	ts-33-501:11439-11439	Outside the scope	Requirements on the MBSF; multicast/broadcast outside the modelled subflow.
P396	ts-33-501:11443-11443	Outside the scope	Requirements on the MBSTF; multicast/broadcast outside the modelled subflow.
P397 to P398	ts-33-501:11899-11901	Outside the scope	NEF security requirements in the interworking annex; the NEF is outside the modelled subflow.

The parties and the protected values, one by one

The host listed the 20 parties and protected values that the drafted properties name, and the analyst said of each which attacker controls the party or holds the value, or why no such attacker was set. With an attacker: 6. Without: 14. The list follows from the properties; what was made of each entry is the analyst's judgement.

ID	Party or value	Kind	Attacker	Why none
S01	NF service consumer	party	`malicious_nf`	

ID	Party or value	Kind	Attacker	Why none
S02	NWDAF	party	Everything lost by definition	A dishonest or compromised NWDAF can answer any analytics it can compute and holds every token presented to it; with it the response binding and token secrecy it is asked to keep are lost by definition, and the documents ask none of them under a dishonest producer.
S03	NRF	party	Everything lost by definition	The NRF is the authorization server: a compromised NRF can forge any token, so every property that rests on a token is lost by definition and the documents ask none of them against a dishonest NRF.
S04	NF service consumer A	party	`stolen_token`	
S05	NF service consumer B	party	`malicious_nf`	
S06	analyticsId	value	Nothing gained	The analyticsId (event-id) is a public request parameter.
S07	event-filter	value	Nothing gained	The event-filter is a request parameter, not a secret; its confidentiality in transit is a TLS property of the exchange, asked by U01.
S08	requester identity	value	Nothing gained	NF instance ids are public identities.
S09	response body	value	Nothing gained	The response body is produced for the requesting consumer; a copy at that consumer is its own data, and no target asks for protection against the consumer that requested it.
S10	analytics payload	value	Nothing gained	Same as the response body: the analytics payload belongs to the request that produced it.
S11	HTTP status code	value	Nothing gained	HTTP status codes are public protocol values.
S12	OAuth2 access token	value	`stolen_token`	
S13	OAuth2 token	value	`stolen_token`	
S14	scope	value	Nothing gained	Scope strings are public and appear in every request.
S15	token audience	value	Nothing gained	The audience is public configuration of the deployment.
S16	context transfer data	value	`malicious_nf`	
S17	analytics data	value	Nothing gained	The analytics data is produced for the requesting consumer; no target asks that it be kept from that consumer.
S18	TLS session response	value	Outside the scope	The task fixes the SBI transport as TLS with endpoint authentication; a broken TLS session is outside the modelled attacker.
S19	analytics success result	value	Nothing gained	The success result is the analytics data of the requesting consumer.
S20	error response	value	Nothing gained	Error responses carry a status code and a cause that the requester is entitled to see.

The properties under each attacker

The attackers are the analyst's, set before a query was written; each answer was recorded under one of them. A cell is the prover's answer for the property under that attacker; "No answer" is an attacker set for the property under which the prover left a claim undecided or nothing was recorded; an empty cell is an attacker the analyst did not set for the property.

Attacker	Has	Basis	Where
`network`	the network only: it reads, drops, changes and sends messages, and holds no secret of a party		
`malicious_nf`	a legitimate NF instance id, its own TLS client certificate and private key, and its own access tokens obtained from the NRF; it may address any service operation of the API with any parameters	An NF service consumer is an authenticated principal that holds its own credentials and tokens (ts-33-501 7964, 7978); a malicious client can impersonate another only if credentials leak (RFC 6749 10.2), and the task asks to distinguish this attacker from the external one.	rfc6749:2827, ts-33-501:7964, s0455, s0456
`stolen_token`	the access token that NF service consumer A obtained and stored, read from A's storage; it does not have A's TLS client certificate or private key	An access token may be stored and reused during its validity time (ts-33-501 7978) and must be kept confidential in storage (RFC 6749 10.3); reading a stored token is the exposure of consumer A.	rfc6749:2859, ts-33-501:7978, s0455

ID	`network`	`malicious_nf`	`stolen_token`
A03	Holds	Holds	Holds
A08	Holds	Holds	Holds
D01	Holds	Holds	
K01	Holds	Holds	
Z01	Holds	Holds	Holds
Z03	Holds	Holds	Holds
Z05	Fails	Fails	
A08.8	Holds	Holds	
U01	Holds	Holds	Holds
U02	Holds	Holds	
X01	Holds	Holds	

Source and targets

Source: 460 units, 154966 bytes, frozen in source-delivery-01.json. Targets: 9 approved (A03, A08, D01, K01, Z01, Z03, Z05, U01, U02); the questions asked: 3. The list of targets was approved as drafted, beforehand, with --approve-targets.

Decisions the host took by its documented default (task `decisions: auto`), for you to check:

- approve_package: approved source package protocol-c88e8f733c4ff0010ba6dd330fee

This step as it reports itself: 2 requests (2 failed), 0.000 USD, 166551 tokens in and 7035 out, 259 s.

Notes on the prover and the model

What the analyst named for targets:

step	targets	features	reason
38	A03, A08, D01, U02, K01, Z01, Z03, Z05, A08.8, U01, X01	plain_primitives, many_sessions	All answers rest on hashing, digital signatures, public-key encryption and an abstract symmetric session key only; no exclusive-or and no Diffie-Hellman algebra is used. The TLS session is idealised as a fresh key shipped under the producer's public key and every later message encrypted and signed under it, so forward secrecy of a session key and the handshake itself are not modelled. Properties are asked over any number of NF service consumers and sessions, so the multi-session behaviour (token reuse, concurrent sessions) is part of the question.

The analyst's closing note

Model: one NF service consumer asks one NWDAF for NETWORK_PERFORMANCE analytics over an endpoint-authenticated, encrypted SBI transport, with OAuth2 Client-Credentials authorization from the NRF; one normal run is found. Principals: NRF, consumers A (honest) and B (legitimate NF driven by the attacker), and three producers — a conformant one, one that reads the API scope of Table 5.2.9-1 as covering every operation of the API, and a counterfactual one without client or token checks used only as a control.

All eleven targets are answered for every attacker set for them (network, malicious legitimate NF, holder of A's stored token), with controls: A03, A08, D01, U02, K01, Z01, Z03, Z05, A08.8, U01, X01. Ten are proved: the response a consumer accepts is the 200 OK produced for its own request; the requester identity bound by the NWDAF is the NF instance in the TLS client certificate; a 200 OK rests on an NRF-issued token whose audience is this NWDAF instance, whose scope covers the requested operation and whose subject is the TLS client; a 204 is never accepted as analytics; A's token never reaches the network or another NF; the NRF issues tokens only to the NF it names. The stolen-token case is shown concretely: B presents A's token at the conformant NWDAF after the leak and is refused, so the TLS subject binding (TS 33.501 13.4.1.1.2) closes the bearer-token gap RFC 6749 10.3 names.

One finding: the scope table. Read literally, nnwdaf-analyticsinfo is "Access to the Nnwdaf_AnalyticsInfo API", so a producer implementing that reading serves the ContextTransfer operation — and the context information of analytics subscriptions — to a token carrying only that scope; under the per-operation reading the same request is refused. This is submitted as depends_on_ambiguity, medium value. A recommendation (state that the API scope covers the analytics resource only and check the scope per operation) was checked on fix-scope.pv: the attack is gone, the normal run is found and all 16 other recorded answers still hold.

Blocks and limits: TLS is idealised as a fresh session key shipped under the producer's public key, so the handshake, forward secrecy and certificate-chain validation, HTTP/2 framing and TLS record replay protection are not modelled; the NRF grants the analytics scope to any authenticated NF (no per-slice policy); only one analytics event type and the 204 error branch are modelled, and SCP/SEPP, roaming, multi-NWDAF aggregation and UE-specific analytics are out of scope. These are listed as limits; they do not leave a target unanswered.