Security analysis of DCCF analytics coordination direct path

This report in another language: 中文.

19.7.0 · ProVerif+ Agent report

Documents	5G System; Data Collection Coordination Services; Stage 3 (protocol definition); 5G System; Network Data Analytics Services; Stage 3 (supporting)
Properties examined	14: 14 hold, 0 fail
Findings	None

Contents

1 Protocol and scope

2 Findings

3 Properties examined

4 Limits of this analysis

Appendix A: Reading of each property

Appendix C: Claims and prover answers

Appendix D: How the analysis was run

1 Protocol and scope

The DCCF coordinates analytics for an NF service consumer that does not address the NWDAF itself. The consumer subscribes for analytics notifications at the DCCF, the DCCF subscribes for the matching events at one NWDAF, and the DCCF passes the NWDAF's notifications on to the consumer under the correlation identifier of the consumer's subscription.

The consumer is assured that its subscription is served with the analytics it asked for and that notifications reach it under its own correlation identifier. The DCCF is assured that a request comes from the NF it names, and the NWDAF is assured that the upstream subscription is made by the DCCF under the scope granted by the NRF.

The parties:

- **NF service consumer**: subscribes for analytics, receives notifications. Holds: its OAuth2 token for the ndccf-datamanagement scope; anaNotifUri.
- **DCCF**: maps one consumer subscription to one NWDAF subscription. Holds: its OAuth2 token for the Nnwdaf_EventsSubscription scope; its own notification URI.
- **NWDAF**: produces the analytics and notifies. Holds: event subscription records.

Figure 1 shows the exchange as the documents specify it.

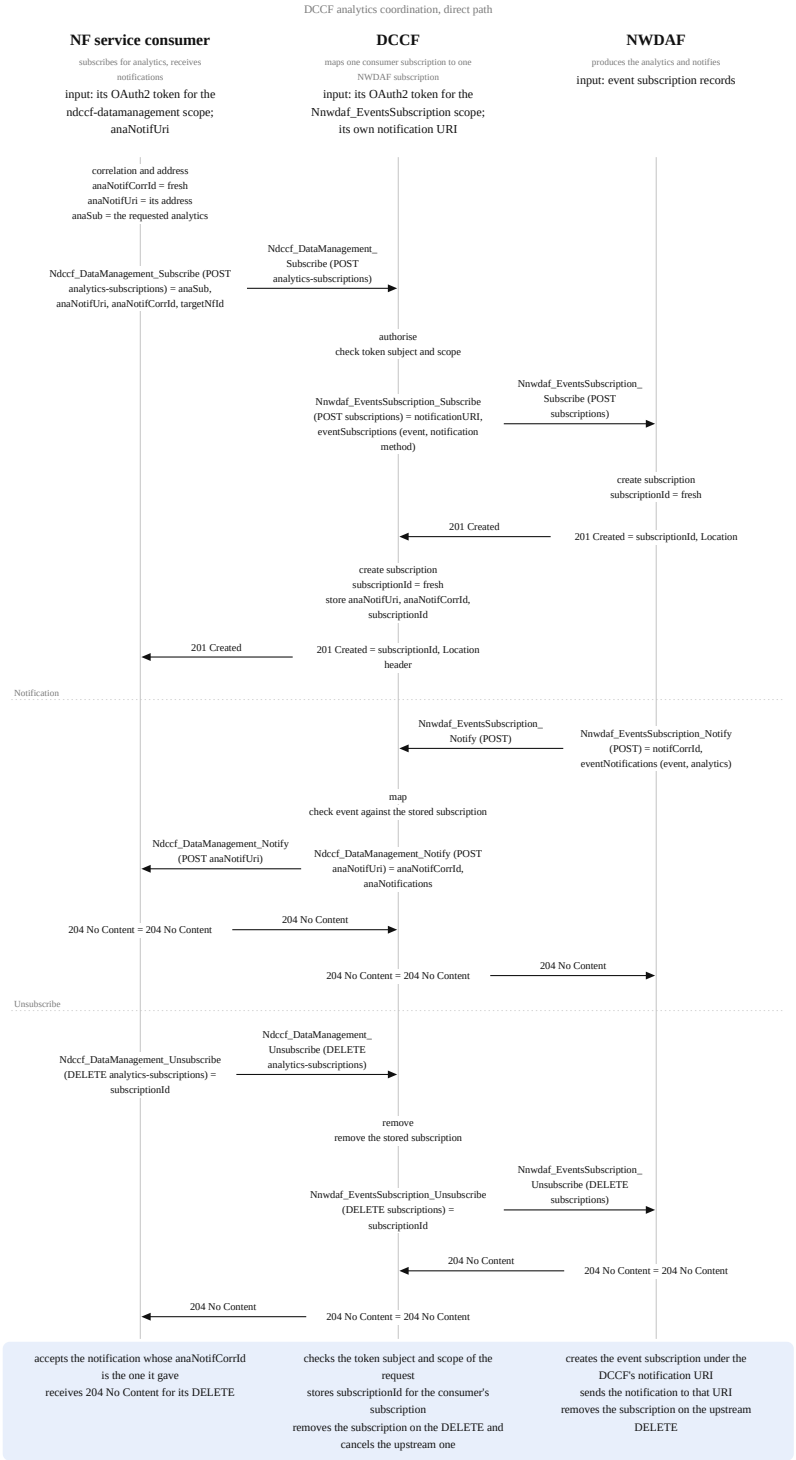


Figure 1. The protocol as specified.

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

No finding. The analysis found nothing that the documents do not already state or set aside. The table in section 3 has the result for every property.

No finding: every examined property holds in the model, and the analysis reproduced no weakness the documents state. All fourteen targets are answered, ten by proof, three also by proof on the unrolled subscription lifetime, and the cancellation targets by the reachability of the termination path. The proofs rest on two things the source text does not spell out: TLS as a private channel between two NFs, and the check of a token's subject and scope. Two state candidates remain unconfirmed: the DCCF removes its own subscription record before the upstream deletion is confirmed, and a failed upstream deletion desynchronises the two sides.

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
A02	**Peer identity agreement on operations.** Whenever the DCCF accepts an analytics subscription request (201 Created), or the consumer accepts an analytics notification (204/200), or the NWDAF accepts the DCCF's upstream subscription request, the claimed peer must actually have run the corresponding service operation under its own identity.	no compromise	Holds	medium	yes
A03	**Notification correlation and subscription binding.** Every notification the consumer accepts must have its anaNotifCorrId and anaNotifUri match the values the consumer previously provided for that subscription, and the corresponding DCCF must actually have created that subscription for the consumer; the upstream subscription the DCCF accepts must match the analytics events the consumer requested.	no compromise	Holds	medium	yes
A04	**No duplicate notification acceptance.** Every analytics notification the consumer accepts corresponds to a distinct notification run of the DCCF (distinguished by anaNotifCorrId and notification content); the same notification instance is not accepted twice as a new notification.	no compromise	Holds	low	unclear
K01	**Access tokens and TLS keys secret.** The attacker cannot obtain any legitimate NF's OAuth2 access token or TLS session key; each token is usable only by its legitimate holder.	no compromise	Holds	low	yes
D01	**Notification address bound to subscriber.** For every notification the consumer accepts, its anaNotifUri and anaNotifCorrId must come from that consumer's genuine subscription request, and the sender must be the DCCF; a malicious legitimate NF cannot cause a notification to be delivered to an address it controls.	no compromise	Holds	medium	unclear
A08	**Upstream subscription matches request.** The NWDAF and analytics events targeted by the upstream subscription the DCCF creates must match the targetNfId/targetNfSetId and anaSub in the consumer's request; the anaNotifCorrId of a notification the consumer accepts must match the subscription value.	no compromise	Holds	medium	yes

ID	Property	Compromise assumed	Result	Value	In the documents
Z01	**Authorisation of every protected operation.** Every protected operation submitted by the DCCF or the NWDAF (creating/deleting a subscription, sending a notification) must have a valid authorisation grant from the NRF, and the grant's scope must cover that operation.	no compromise	Holds	medium	yes
Z03	**Token subject audience scope binding.** The subject, audience, scope and usage context of every authorisation grant must match the NF submitting the operation and the operation itself; a token cannot be transplanted to another NF or another operation.	no compromise	Holds	medium	yes
Z04	**No notification after unsubscribe.** After a cancellation (successful DELETE) or termination (terminationReq=true) takes effect, the DCCF sends no new notifications for that subscription; in-flight notifications are handled under an explicitly defined finite state.	no compromise	Holds within a bound (the changing value unrolled to 2 steps)	medium	unclear
S05	**Terminated subscription identifier inert.** After the point at which a subscription termination (cancellation or terminationReq) takes effect, the old subscriptionId or old subscription state cannot trigger a new notification send.	no compromise	Holds within a bound (the changing value unrolled to 2 steps)	medium	unclear
U01	**Upstream and downstream subscriptions correspond.** The downstream subscription the DCCF creates for the consumer corresponds one-to-one with the upstream subscription it creates at the NWDAF, and the notification's anaNotifCorrId is not mismatched between the two layers.	no compromise	Holds	medium	yes
U02	**Notification address from the subscriber itself.** The anaNotifUri recorded by the DCCF must come from the consumer that initiated that subscription request itself; a malicious legitimate NF cannot replace another consumer's notification receiving address.	no compromise	Holds	medium	unclear
U03	**Unsubscribe propagates to the upstream.** After the consumer cancels its subscription, the DCCF cancels the corresponding upstream subscription; in-flight notifications are no longer accepted after the point at which the cancellation takes effect; the lifetime is bounded by a finite state.	no compromise	Holds within a bound (the changing value unrolled to 2 steps)	medium	unclear
X01	**Analytics payload kept from other NFs.** No party other than the subscribing consumer learns the analytics content of a notification for that subscription, neither the network attacker nor another legal NF holding its own token.	a legitimate NF with its own token for the ndccf-datamanagement scope; it can call the DCCF API and choose anaNotifUri, events and target NWDAF	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: Mapping of subscriptions and notification correlation IDs inside and outside the DCCF	The DCCF keeps one downstream subscription per upstream subscription in the same process: every notification it forwards comes from the upstream notification of that subscription and carries the consumer's anaNotifCorrId, and the consumer accepts nothing else.	U01, A03, A08
Q2: A legitimate NF cannot replace another's notification receiving address	A legal NF can only have its own identity and its own address recorded, because the DCCF binds the request to the token subject, so it cannot replace another consumer's address. The DCCF does not check that the address belongs to the requester, so the guarantee rests on that identity binding and on routing notifications by the requesting session.	U02, D01, Z03
Q3: Cancellation propagation and in-flight notifications are explicitly bounded by a finite state	The unsubscribe reaches the DCCF and is propagated to the NWDAF, and an in-flight notification that arrives after the termination point is not delivered in the unrolled model. The source does not fix the order of the two deletions, and the DCCF removes its own record first, so a failed upstream deletion desynchronises the two sides; that candidate is reported as not confirmed.	U03, Z04, S05

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:

- TLS is modelled as one private channel per endpoint pair, so the network attacker can neither read nor change an established session; the source text leaves TLS to the baseline.
- Token checks (subject and scope) are modelled as the baseline requires; the selected text only says that OAuth2 may be used and names the scopes.
- The DCCF and the NWDAF are not replicated: the model has one session per channel, so the answers speak of the written sessions, not of any number of them.
- The leak of an access token is not modelled; with a leaked bearer token impersonation follows by definition and the documents ask for token confidentiality instead.
- User consent through the UDM, the muting flag, feature negotiation and redirects are not modelled.
- Notifications are routed by the TLS session of the requester, so a registered address that belongs to no one has no delivery effect in the model.
- The subscription lifetime is unrolled to two notification steps in the second model; the direct query that a late notification is never accepted comes back inconclusive.
- No cryptographic primitive carries weight in this path: the token is a private constructor and the payload is a fresh value.

What the host records:

- The answers of Z04, S05, U03 rest on an operation the model simplifies (Appendix D).
- Proved without a control query (Appendix C): A02: a02a, A02: a02b, A02: a02c, A03: a03, A04: a04, K01: k01, D01: a03, D01: a02b, A08: a08, Z01: a02b, Z03: a02b, U01: u01, U02: a02b, X01: x01. A control is a variant of the property that must fail; without one, nothing shows that the property as written can fail in this 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.

A02: Peer identity agreement on operations

Whenever the DCCF accepts an analytics subscription request (201 Created), or the consumer accepts an analytics notification (204/200), or the NWDAF accepts an upstream subscription request from the DCCF, the claimed peer must actually have run the corresponding service operation under its own identity.

The DCCF accepts a subscription only for the identity named by the token it carries, the NWDAF accepts an upstream subscription only from the DCCF, and a legal NF with its own token cannot act in another NF's name.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It confirms the authorisation goal the documents state, under the assumed checks.
- Stated in the documents: yes. Where the documents say it: s0412, s0413, s0415
- Rests on: a private channel per endpoint pair stands for TLS; the subject and scope checks are assumed as the baseline requires; one session per channel
- Next step: none for this path.

A03: Notification correlation and subscription binding

Every notification a consumer accepts must carry an anaNotifCorrId and anaNotifUri that match the values the consumer previously gave for that subscription, and the corresponding DCCF must actually have created that subscription for that consumer; an upstream subscription the DCCF accepts must match the analytics event the consumer requested.

A notification a consumer accepts always carries the correlation identifier of a subscription the DCCF recorded for that consumer, and the address is the one that consumer gave. A consumer cannot be made to accept a notification of another subscription.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It is the core binding of the coordination function and it holds as specified.
- Stated in the documents: yes. Where the documents say it: s0060, s0061
- Rests on: the consumer checks the correlation identifier and the DCCF's token; the source only requires the identifier to be echoed; a private channel per endpoint pair stands for TLS
- Next step: none needed for this path.

A04: No duplicate notification acceptance

Every analytics notification a consumer accepts corresponds to a distinct notification run of the DCCF (distinguished by anaNotifCorrId and the notification content); the same notification instance is not accepted again as a new notification.

Each notification a consumer accepts corresponds to a distinct forwarding run of the DCCF, so a repeated delivery is not accepted as a new notification.

- Result: Holds; answers what is asked: yes; usable with caveats

- Value low: The source states no anti-replay rule, and the result follows from the fresh correlation identifier.
- Stated in the documents: unclear. Source text it rests on: s0061
- Rests on: the consumer accepts one notification per subscription session; no retransmission or failure path is modelled
- Next step: none.

K01: Access tokens and TLS keys secret

The attacker cannot obtain the OAuth2 access token or the TLS session key of any legal NF; each token is usable only by its legal holder.

No party learns another NF's access token: each token stays with its holder. The control shows that a token put on the open network is learned, so the query is not empty.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: It follows directly from the transport assumption the baseline gives.
- Stated in the documents: yes. Where the documents say it: s0412
- Rests on: a private channel per endpoint pair stands for TLS; the token is never sent on the open network
- Next step: none.

D01: Notification address bound to subscriber

Every notification a consumer accepts must have its anaNotifUri and anaNotifCorrId come from that consumer's real subscription request, and the sender must be the DCCF; a malicious legal NF cannot cause a notification to be delivered to an address it controls.

What reaches a consumer's notification address belongs to a subscription the DCCF stored for that consumer. A legal NF can only have its own identity and its own address recorded, so it cannot have another consumer's notifications delivered to itself.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It shows the binding holds because the identity is bound to the token, not because the address is checked.
- Stated in the documents: unclear. Source text it rests on: s0007, s0060
- Rests on: notifications are routed by the TLS session of the requesting NF; the DCCF does not check that an address belongs to the requester; the source states no such check
- Open: what a NF gains by registering an address that is not its own
- Next step: a model in which notification routing follows the registered address rather than the session.

A08: Upstream subscription matches request

The NWDAF and analytics event that the upstream subscription created by the DCCF points to must match the targetNfId/targetNfSetId and anaSub in the consumer's request; the anaNotifCorrId of a notification the consumer accepts must match the subscription value.

The event the consumer asked for is the event of the upstream subscription, and a notification for another event is not passed on to the consumer. A dishonest NWDAF cannot get an unrequested event through the DCCF.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It shows the check that keeps the two layers consistent.

- Stated in the documents: yes. Where the documents say it: s0021, s0061
- Rests on: the DCCF forwards the consumer's event unchanged; the NWDAF is modelled as able to choose the event it notifies, and the DCCF filters on it
- Next step: none.

Z01: Authorisation of every protected operation

Every protected operation submitted by the DCCF or the NWDAF (creating/deleting a subscription, sending a notification) must have a valid authorisation grant from the NRF, and the grant's scope must cover that operation.

The DCCF admits only requests whose token names the requesting NF and carries the service scope, and the NWDAF does the same for the DCCF's upstream request. The API defines a single scope for the whole service, so the scope does not distinguish one operation or one subscription from another.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It confirms the stated framework while showing that authorisation is per service, not per resource.
- Stated in the documents: yes. Where the documents say it: s0412, s0415
- Rests on: the token checks are assumed; the API-wide scope is modelled as one string covering all operations
- Next step: none.

Z03: Token subject audience scope binding

The subject, audience, scope and usage context of every authorisation grant must match the NF that submits the operation and the operation itself; a token cannot be transplanted to another NF or another operation.

A token is usable only by the NF it was issued to: the DCCF accepts a request only for the subject of its token. Audience and expiry are not modelled, so nothing is claimed about them.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It answers the part of the binding the model can express and marks the rest open.
- Stated in the documents: yes. Where the documents say it: s0415, s1227
- Rests on: only subject and scope are modelled; a private channel per endpoint pair stands for TLS
- Open: audience and expiry binding
- Next step: read the baseline for the audience and expiry rules and extend the token.

Z04: No notification after unsubscribe

After a cancellation (successful DELETE) or termination (terminationReq=true) takes effect, the DCCF sends no further notifications for that subscription; notifications in flight are handled according to a clearly defined, bounded state.

After an accepted DELETE the DCCF has no path that delivers a further notification, and in the unrolled model a notification that arrives after that point is dropped without reaching the consumer.

- Result: Holds within a bound (the changing value unrolled to 2 steps); answers what is asked: yes; usable with caveats
- Value medium: It shows the termination behaviour holds in the unrolled model, with the caveat that the check itself is not in the source.
- Stated in the documents: unclear. Source text it rests on: s0040, s0041

- Rests on: the lifetime is unrolled to two notification steps; the drop after the termination point is a modelling choice; the source states no such check; the direct query that a late notification is never accepted is inconclusive
- Open: the exact handling of a notification in flight at the termination point
- Next step: an editor's statement on in-flight notifications, then a model that follows it.

S05: Terminated subscription identifier inert

After the point at which a subscription termination (cancellation or terminationReq) takes effect, the old subscriptionId or the old subscription state cannot trigger a new notification delivery.

A message that carries the correlation identifier of a terminated subscription does not produce a new delivery to the consumer in the unrolled model.

- Result: Holds within a bound (the changing value unrolled to 2 steps); answers what is asked: yes; usable with caveats
- Value medium: It is the identifier-reuse question the target asks, answered on the unrolled model.
- Stated in the documents: unclear. Source text it rests on: s0041, s0069
- Rests on: the lifetime is unrolled to two steps; the terminated state is modelled by dropping further notifications
- Next step: none.

U01: Upstream and downstream subscriptions correspond

The downstream subscription the DCCF creates for a consumer corresponds one to one with the upstream subscription it creates to the NWDAF, and the anaNotifCorrId of a notification is not mismatched between the two layers.

Every notification the DCCF forwards for a subscription comes from an upstream notification of that same subscription and carries its content, so the consumer's subscription and the DCCF's upstream subscription correspond one to one.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It answers the mapping question the user asked, for the direct path.
- Stated in the documents: yes. Where the documents say it: s0021, s0026
- Rests on: the DCCF holds the mapping in one process, one downstream subscription per upstream subscription; no concurrent deletion or modification of the mapping is modelled
- Next step: none.

U02: Notification address from the subscriber itself

The anaNotifUri recorded by the DCCF must come from the consumer that initiated the subscription request itself; a malicious legal NF cannot replace another consumer's notification reception address.

The address recorded for a subscription is the one in the request of the NF that the token names, so a legal NF cannot have another consumer's address recorded under its own identity. The DCCF does not check that an address belongs to the requester, and the model routes notifications by the requester's session, so a foreign address has no delivery effect.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It answers the address question with an explicit statement of what the model cannot show.
- Stated in the documents: unclear. Source text it rests on: s0007, s0021

- Rests on: notification routing by the TLS session of the requester; no ownership check on the address in the source text
- Open: a DCCF that routes by the registered address
- Next step: a model with address-based routing, to test whether a foreign address can be abused.

U03: Unsubscribe propagates to the upstream

After the consumer unsubscribes, the DCCF cancels the corresponding upstream subscription; in-flight notifications after the point where the cancellation takes effect are no longer accepted; the lifecycle is delimited by a finite state.

The consumer's unsubscribe is propagated to the NWDAF and an in-flight notification that arrives after that point is not passed on to the consumer. The DCCF removes its own record before the upstream deletion is confirmed, which is the state candidate reported separately.

- Result: Holds within a bound (the changing value unrolled to 2 steps); answers what is asked: yes; usable with caveats
- Value medium: It answers the propagation question and marks the ordering gap that remains.
- Stated in the documents: unclear. Source text it rests on: s0039, s0041, s0981
- Rests on: two notification steps unrolled; no failure of the upstream deletion is modelled
- Open: what happens if the upstream deletion fails
- Next step: a source statement on retry or rollback, or a model of the failure path.

X01: Analytics payload kept from other NFs

No party other than the subscribing consumer learns the analytics content of a notification for that subscription, neither the network attacker nor another legal NF holding its own token.

The analytics content delivered to a consumer stays with that consumer: neither the network attacker nor another legal NF learns it. A legal NF does receive its own analytics, which is why the claim is about the content of another consumer's notification.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It is the confidentiality goal the notification path is meant to give, proved for the written sessions.
- Stated in the documents: yes. Where the documents say it: s0060, s0061
- Rests on: the content is a value derived from the upstream subscription; a private channel per endpoint pair stands for TLS
- Next step: none for this path.

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
A02	proved		a02a			4 proved
A03	proved		a03			2 proved
A04	proved		a04			1 proved
K01	proved		k01	1 of 2		2 proved
D01	proved		a03			2 proved
A08	proved		a08			2 proved
Z01	proved		a02b			2 proved
Z03	proved		a02b			2 proved
Z04	bounded	the changing value unrolled to 2 steps	once	checked		1 bounded, 1 reachable
S05	bounded	the changing value unrolled to 2 steps	once	checked		1 bounded, 1 reachable
U01	proved		u01			1 proved
U02	proved		a02b			2 proved
U03	bounded	the changing value unrolled to 2 steps	once	checked		1 bounded, 1 reachable
X01	proved		x01			2 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
A02	When the consumer accepts a 201 Created for its subscription, that creation carries the anaNotifCorrId and anaNotifUri of a subscription request the consumer itself ran	`network`	proved	a02a	none
A02	When the DCCF accepts a subscription request and stores the subscription, the request was made under the identity it names and with the address and correlation identifier of that request	`network`	proved	a02b	none
A02	When the NWDAF accepts an upstream subscription, the DCCF ran the corresponding upstream subscribe operation for that correlation identifier and event	`network`	proved	a02c	none
A02	A legal NF holding its own token cannot make the DCCF store a subscription under another consumer's identity (the token subject must equal the identity in the request)	`malicious_nf`	proved	a02b	none
A03	A notification accepted by a consumer carries the anaNotifCorrId of a subscription the DCCF stored for that consumer, so it is bound to that subscription	`network`	proved	a03	none
A03	Another legal NF cannot get a notification accepted by a consumer as one of that consumer's subscriptions: the correlation identifier is the one the DCCF stored for that consumer	`malicious_nf`	proved	a03	none

target	claim	attacker	answer	query	control
A04	Each notification accepted by a consumer corresponds to a distinct forwarding run of the DCCF; no notification is accepted twice	`network`	proved	a04	none
K01	No other party learns a legal NF's OAuth2 access token: tokens stay with their holder	`network`	proved	k01	checked
K01	A legal NF with its own token learns no token but its own	`malicious_nf`	proved	k01	none
D01	What reaches a consumer's anaNotifUri matches a subscription the DCCF stored for that consumer, with the correlation identifier of that subscription	`network`	proved	a03	none
D01	A legal NF cannot have the DCCF record another consumer's address for a subscription made in its own name, and cannot make the DCCF deliver another consumer's notification to it	`malicious_nf`	proved	a02b	none
A08	The upstream subscription that reaches the NWDAF carries the analytics event the consumer asked for, and the notification correlation is consistent across the two layers	`network`	proved	a08	none
A08	A dishonest NWDAF cannot get an event upstream that no consumer subscribed to: the DCCF passes on the subscribed event, and a notification for another event is not forwarded	`nwdaf_dishonest`	proved	a08	none
Z01	The DCCF stores a subscription only for a request that carries a valid token whose subject is the NF it names and whose scope covers ndccf-datamanagement	`network`	proved	a02b	none
Z01	A legal NF with its own token cannot have a subscription stored under another NF's identity, so the scope alone does not authorise acting for another NF	`malicious_nf`	proved	a02b	none
Z03	A token is only used for the NF it was issued to: the DCCF accepts a request only for the subject of the token it carries	`network`	proved	a02b	none
Z03	A legal NF cannot transplant its own token to act for another NF or into another identity in the request	`malicious_nf`	proved	a02b	none
Z04	The termination path exists: an accepted unsubscribe reaches the DCCF and the upstream NWDAF subscription is removed	`network`	reachable	z04r	
Z04	After the termination point a further notification of the subscription is not delivered: it is dropped by the DCCF, and the consumer accepts each notification at most once	`network`	bounded (the changing value unrolled to 2 steps)	once	checked
S05	The DCCF removes its own record of the subscription on an accepted DELETE	`network`	reachable	s05r	
S05	A message carrying the terminated subscription's correlation identifier after the termination cannot trigger a new delivery to the consumer	`network`	bounded (the changing value unrolled to 2 steps)	once	checked
U01	Every notification the DCCF forwards for a subscription comes from an upstream notification it received for that subscription, so the two layers correspond	`network`	proved	u01	none
U02	The DCCF records as anaNotifUri the address carried in the request of the NF it names, and every notification for that subscription carries that subscription's correlation identifier	`network`	proved	a02b	none
U02	A legal NF cannot replace another consumer's notification address: it can only have its own address recorded, under its own identity	`malicious_nf`	proved	a02b	none

target	claim	attacker	answer	query	control
U03	An accepted unsubscribe is propagated: the DCCF removes the upstream NWDAF subscription for the deleted consumer subscription	`network`	reachable	z04r	
U03	The unsubscribe is propagated to the upstream subscription and an in-flight notification arriving after that point is not passed on to the consumer	`network`	bounded (the changing value unrolled to 2 steps)	once	checked
X01	No party other than the subscribing consumer learns the analytics content of its notification	`network`	proved	x01	none
X01	A legal NF does not learn the analytics content delivered to another consumer, even though it can subscribe and receives its own analytics	`malicious_nf`	proved	x01	none

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-20261115T094237, 2026-11-15. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 47, prover 3 s, wall 13 min, cost 0.110 USD.

ProVerif: 48 proofs run, 45 decided, 3 without an answer, 1 s, 28 answers recorded.

What was analysed

Protocol	DCCF analytics coordination direct path
Version	19.7.0
Analysed	Single NF consumer → DCCF → single NWDAF, the direct notification path for non-UE aggregated analytics; maps one upstream subscription to one downstream subscription, notifications and cancellation; excludes MFAF, ADRF, cache reuse and multi-NWDAF aggregation. The upstream service must be modelled; DCCF data cannot be conjured out of thin air.
Not analysed	The symbolic model does not prove the computational security of the algorithms or their resistance to quantum attack; attack traces hold only under explicit assumptions and do not directly correspond to specification vulnerabilities. Each property must have a reachability/positive control. Fixed to Rel-19. The external network attacker and the malicious legal NF holding its own credentials are kept separate; TLS authenticated encryption is modelled per endpoint, and a legal 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, so the finite-state lifecycle must be explicitly delimited. The goal is the specified API sub-flow, not full 5G security.
Source text	1266 numbered passages (927204 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	4.2.2.2.2	Subscription for analytics notifications
main	4.2.2.3.2	Unsubscribe from analytics notifications
main	4.2.2.4.2	Notification about subscribed analytics
main	5.1	Ndccf_DataManagement Service API
main	5.1.1	Introduction
main	5.1.2	Usage of HTTP
main	5.1.2.1	General
main	5.1.2.2	HTTP standard headers
main	5.1.2.2.1	General
main	5.1.2.2.2	Content type
main	5.1.2.3	HTTP custom headers
main	5.1.3	Resources
main	5.1.3.1	Overview

Document	Section	Title
main	5.1.3.2	Resource: DCCF Analytics Subscriptions
main	5.1.3.2.1	Description
main	5.1.3.2.2	Resource Definition
main	5.1.3.2.3	Resource Standard Methods
main	5.1.3.2.3.1	POST
main	5.1.3.2.4	Resource Custom Operations
main	5.1.3.3	Resource: Individual DCCF Analytics Subscription
main	5.1.3.3.1	Description
main	5.1.3.3.2	Resource Definition
main	5.1.3.3.3	Resource Standard Methods
main	5.1.3.3.3.1	PUT
main	5.1.3.3.3.2	DELETE
main	5.1.3.3.4	Resource Custom Operations
main	5.1.3.4	Resource: DCCF Data Subscriptions
main	5.1.3.4.1	Description
main	5.1.3.4.2	Resource Definition
main	5.1.3.4.3	Resource Standard Methods
main	5.1.3.4.3.1	POST
main	5.1.3.4.4	Resource Custom Operations
main	5.1.3.5	Resource: Individual DCCF Data Subscription
main	5.1.3.5.1	Description
main	5.1.3.5.2	Resource Definition
main	5.1.3.5.3	Resource Standard Methods
main	5.1.3.5.3.1	PUT
main	5.1.3.5.3.2	DELETE
main	5.1.3.5.4	Resource Custom Operations
main	5.1.3.6	Void
main	5.1.3.7	Void
main	5.1.4	Custom Operations without associated resources
main	5.1.4.1	Overview
main	5.1.4.2	Operation: transfer-data-sub
main	5.1.4.2.1	Description
main	5.1.4.2.2	Operation Definition
main	5.1.5	Notifications
main	5.1.5.1	General
main	5.1.5.2	Analytics Notification

Document	Section	Title
main	5.1.5.2.1	Description
main	5.1.5.2.2	Target URI
main	5.1.5.2.3	Standard Methods
main	5.1.5.2.3.1	POST
main	5.1.5.3	Data Notification
main	5.1.5.3.1	Description
main	5.1.5.3.2	Target URI
main	5.1.5.3.3	Standard Methods
main	5.1.5.3.3.1	POST
main	5.1.5.4	Fetch Notification
main	5.1.5.4.1	Description
main	5.1.5.4.2	Target URI
main	5.1.5.4.3	Standard Methods
main	5.1.5.4.3.1	POST
main	5.1.6	Data Model
main	5.1.6.1	General
main	5.1.6.2	Structured data types
main	5.1.6.2.1	Introduction
main	5.1.6.2.2	Type NdccfAnalyticsSubscription
main	5.1.6.2.3	Type NdccfDataSubscription
main	5.1.6.2.4	Type NdccfAnalyticsSubscriptionNotification
main	5.1.6.2.5	Type NdccfDataSubscriptionNotification
main	5.1.6.2.6	Type FormattingInstruction
main	5.1.6.2.7	Type ProcessingInstruction
main	5.1.6.2.8	Type ParameterProcessingInstruction
main	5.1.6.2.9	Type NotifSummaryReport
main	5.1.6.2.10	Type EventParamReport
main	5.1.6.2.11	Type ReportingOptions
main	5.1.6.2.12	Void
main	5.1.6.2.13	Type DccfEvent
main	5.1.6.2.14	Type NotifyEndpoint
main	5.1.6.2.15	Type: StorageHandlingInformation
main	5.1.6.2.16	Type: DeletionAlert
main	5.1.6.2.17	Type: NotifResponse
main	5.1.6.2.18	Void
main	5.1.6.2.19	Type: DataTransferResp

Document	Section	Title
main	5.1.6.3	Simple data types and enumerations
main	5.1.6.3.1	Introduction
main	5.1.6.3.2	Simple data types
main	5.1.6.3.3	Enumeration: SummarizationAttribute
main	5.1.6.3.4	Enumeration: AggregationLevel
main	5.1.6.3.5	Enumeration: DataCollectionPurpose
main	5.1.6.3.6	Enumeration: TermCause
main	5.1.6.4	Data types describing alternative data types or combinations of data types
main	5.1.6.5	Binary data
main	5.1.7	Error Handling
main	5.1.7.1	General
main	5.1.7.2	Protocol Errors
main	5.1.7.3	Application Errors
main	5.1.8	Feature negotiation
main	5.1.9	Security
ts-29-520	4.2.1	Service Description
ts-29-520	4.2.1.1	Overview
ts-29-520	4.2.1.2	Service Architecture
ts-29-520	4.2.1.3	Network Functions
ts-29-520	4.2.1.3.1	Network Data Analytics Function (NWDAF)
ts-29-520	4.2.1.3.2	NF Service Consumers
ts-29-520	4.2.2.2	Nnwdaf_EventsSubscription_Subscribe service operation
ts-29-520	4.2.2.2.1	General
ts-29-520	4.2.2.2.2	Subscription for event notifications
ts-29-520	4.2.2.2.3	Update subscription for event notifications
ts-29-520	4.2.2.3	Nnwdaf_EventsSubscription_Unsubscribe service operation
ts-29-520	4.2.2.3.1	General
ts-29-520	4.2.2.3.2	Unsubscribe from event notifications
ts-29-520	4.2.2.4	Nnwdaf_EventsSubscription_Notify service operation
ts-29-520	4.2.2.4.1	General
ts-29-520	4.2.2.4.2	Notification about subscribed event
ts-29-520	5.1.1	Introduction
ts-29-520	5.1.2	Usage of HTTP
ts-29-520	5.1.2.1	General
ts-29-520	5.1.2.2	HTTP standard headers
ts-29-520	5.1.2.2.1	General

Document	Section	Title
ts-29-520	5.1.2.2.2	Content type
ts-29-520	5.1.2.3	HTTP custom headers
ts-29-520	5.1.3.1	Resource Structure
ts-29-520	5.1.3.2	Resource: NWDAF Events Subscriptions
ts-29-520	5.1.3.2.1	Description
ts-29-520	5.1.3.2.2	Resource definition
ts-29-520	5.1.3.2.3	Resource Standard Methods
ts-29-520	5.1.3.2.3.1	POST
ts-29-520	5.1.3.2.4	Resource Custom Operations
ts-29-520	5.1.3.3	Resource: Individual NWDAF Event Subscription
ts-29-520	5.1.3.3.1	Description
ts-29-520	5.1.3.3.2	Resource definition
ts-29-520	5.1.3.3.3	Resource Standard Methods
ts-29-520	5.1.3.3.3.1	DELETE
ts-29-520	5.1.3.3.3.2	PUT
ts-29-520	5.1.3.3.4	Resource Custom Operations
ts-29-520	5.1.5	Notifications
ts-29-520	5.1.5.1	General
ts-29-520	5.1.5.2	Event Notification
ts-29-520	5.1.5.2.1	Description
ts-29-520	5.1.5.2.2	Operation Definition
ts-29-520	5.1.6.2.2	Type NnwdafEventsSubscription
ts-29-520	5.1.6.2.3	Type EventSubscription
ts-29-520	5.1.6.2.4	Type NnwdafEventsSubscriptionNotification
ts-29-520	5.1.6.2.5	Type EventNotification
ts-29-520	5.1.6.2.7	Type EventReportingRequirement
ts-29-520	5.1.6.2.22	Type NetworkPerfRequirement
ts-29-520	5.1.6.2.23	Type NetworkPerfInfo
ts-29-520	5.1.6.2.30	Type ThresholdLevel
ts-29-520	5.1.6.2.49	Type ConsumerNfInformation
ts-29-520	5.1.6.3.3	Enumeration: NotificationMethod
ts-29-520	5.1.6.3.4	Enumeration: NwdafEvent
ts-29-520	5.1.6.3.10	Enumeration: NetworkPerfType
ts-29-520	5.1.6.3.26	Enumeration: TermCause
ts-29-520	5.1.7	Error handling
ts-29-520	5.1.7.1	General

Document	Section	Title
ts-29-520	5.1.7.2	Protocol Errors
ts-29-520	5.1.7.3	Application Errors
ts-29-520	5.1.8	Feature negotiation
ts-29-520	5.1.9	Security

The purpose of the protocol as the source step located it in the documents: a single NF consumer → DCCF → single NWDAF, non-UE aggregated analytics direct notification path; one upstream subscription is created and mapped to one downstream subscription, with notification and cancellation; MFAF, ADRF, cache reuse and multi-NWDAF aggregation are excluded. The upstream service must be modelled; DCCF data must not be conjured out of nothing.

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
A02	Peer identity agreement on operations	the consumer is assured the DCCF, and the NWDAF accepting the upstream subscription, are the peers they claim; the DCCF is assured the request comes from the consumer it names	identity carried in the request and authorised by an OAuth2 token from the NRF (s0005, s0412, s0413)	s0005, s0412
A03	Notification correlation and subscription binding	the consumer is assured a notification carries the anaNotifCorrId and destination it gave and belongs to a subscription the DCCF created for it; the upstream carries the events it asked for	anaSub, anaNotifUri, anaNotifCorrId are required in the request (s0006, s0007, s0008) and returned in the notification (s0060, s0061)	s0061
A04	No duplicate notification acceptance	the consumer: every accepted notification is a distinct notification run of the DCCF for that subscription	the notification carries the anaNotifCorrId correlating it (s0061); no anti-replay rule is stated	s0061
K01	Access tokens and TLS keys secret	each NF: its OAuth2 token and TLS session keys stay with it	OAuth2 token from the NRF under the client credentials grant (s0412, s0413)	s0412
D01	Notification address bound to subscriber	the consumer: what reaches its anaNotifUri belongs to its own subscriptions and comes from the DCCF; a malicious NF cannot have notifications delivered to an address it does not own	the consumer supplies anaNotifUri (s0007) and the DCCF posts to it (s0060); no ownership check is stated	s0007, s0060
A08	Upstream subscription matches request	the upstream subscription names the NWDAF and the events the consumer asked for, and the correlation stays consistent across the two layers	targetNfId selects the NWDAF (s0014); the DCCF creates the Nnwdaf subscription to serve the request (s0021)	s0014, s0021
Z01	Authorisation of every protected operation	the DCCF and the NWDAF: each protected operation is submitted by a party holding a valid NRF authorisation whose scope covers it	OAuth2 with the NRF as authorisation server (s0412, s0413); one service-wide scope (s0415)	s0412, s0415
Z03	Token subject audience scope binding	the DCCF and the NWDAF: a token is usable only by the NF it was issued to, for the operation its scope covers	scope definition (s0415, s1227); binding left to the baseline	s0415, s1227
Z04	No notification after unsubscribe	the consumer: after a successful DELETE the DCCF sends no new notification for that subscription	the DCCF removes the subscription and answers 204 (s0040, s0041, s0042)	s0041

ID	Name	About	What the documents say	Where
S05	Terminated subscription identifier inert	the consumer: after termination the old subscriptionId and old state cannot trigger a new notification	the DCCF removes the subscription on DELETE (s0041) and may request termination in a notification (s0069)	s0041, s0069
U01	Upstream and downstream subscriptions correspond	the DCCF: each consumer subscription maps to exactly one upstream NWDAF subscription and the correlation identifiers are not swapped	the DCCF creates the Nnwdaf subscription then creates, numbers and stores its own (s0021, s0026, s0028)	s0021
U02	Notification address from the subscriber itself	the DCCF records as anaNotifUri only the address given by the NF that made the request; no other NF can replace another consumer's notification address	the request shall include the notification target address (s0007) and the DCCF posts to it (s0060)	s0007, s0060
U03	Unsubscribe propagates to the upstream	the DCCF cancels the corresponding upstream subscription when the consumer unsubscribes, and in-flight notifications are not accepted after the effect point	the consumer deletes the subscription (s0039), the DCCF removes it and answers 204 (s0041)	s0039, s0041
X01	Analytics payload kept from other NFs	the notification content delivered to the consumer for its own subscription	the notification is posted to the consumer's anaNotifUri and carries the analytics content (s0060, s0061); the source states no confidentiality goal of its own for the payload	s0060

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: 53. Not taken: 345. 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	general introduction
P003 to P006	rfc6749:2790-2823	Z01, K01, `malicious_nf`	
P007 to P010	rfc6749:2827-2850	A02, Z03, `malicious_nf`	
P011 to P015	rfc6749:2859-2881	No goal stated	a leaked bearer token is usable by its holder; the documents require token confidentiality, which K01 states
P016 to P025	rfc6749:2885-2943	No goal stated	no refresh tokens or authorization codes in this grant
P026 to P029	rfc6749:2947-2980	D01, U02, `malicious_nf`	
P030 to P033	rfc6749:2984-3003	No goal stated	password grant not used
P034 to P035	rfc6749:3013-3019	K01, `network`, `malicious_nf`	
P036	rfc6749:3023-3028	A02, D01, `network`	

Paragraphs	Where	Taken up as	Why not
P037 to P039	rfc6749:3032-3041	K01, `network`	
P040 to P042	rfc6749:3045-3066	Outside the scope	no resource owner
P043 to P052	rfc6749:3070-3150	No goal stated	no browser or user agent takes part
P053 to P054	rfc6749:3154-3167	D01, U02, `malicious_nf`	
P055 to P062	rfc6749:3170-3210	No goal stated	implicit grant not used
P063	ts-33-501:2491-2491	Outside the scope	algorithm negotiation
P064 to P072	ts-33-501:2495-2511	Z01, Z03, A02, `network`, `malicious_nf`	
P073 to P213	ts-33-501:2515-2853	Outside the scope	RAN, UE and UDM requirements outside this API path
P214	ts-33-501:2859-2859	`network`, `malicious_nf`	
P215 to P220	ts-33-501:2865-2875	Z01, Z03, A02, `malicious_nf`	
P221 to P241	ts-33-501:2879-2923	Outside the scope	NRF, NEF and SCP internals
P242 to P254	ts-33-501:2929-2953	K01, A02, `network`	
P255 to P314	ts-33-501:2957-3081	Outside the scope	roaming, SEPP and IPUPS
P315 to P318	ts-33-501:3085-3091	Z01, A02, `network`, `malicious_nf`	
P319 to P327	ts-33-501:3097-3115	Outside the scope	signalling monitoring
P328 to P337	ts-33-501:3121-3141	No goal stated	operational visibility
P338 to P369	ts-33-501:3149-3215	The prover cannot be asked	algorithm strength and selection
P370 to P398	ts-33-501:3219-11901	Outside the scope	5G-RG, NSSAAF, N3, EAP-TLS, IAB, MBSF, NEF

The parties and the protected values, one by one

The host listed the 25 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: 13. Without: 12. 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	DCCF	party	Everything lost by definition	a dishonest DCCF mediates every operation of this path
S03	NWDAF	party	`nwdaf_dishonest`	
S04	NRF	party	Everything lost by definition	a dishonest NRF issues any token; the documents require NRF security instead
S05	Malicious legitimate NF	party	`malicious_nf`	
S06	consumer identity in the subscription request	value	`malicious_nf`	
S07	DCCF identity in notifications	value	Everything lost by definition	a forged DCCF identity needs the DCCF's own key, taken as secure
S08	DCCF identity in the upstream subscription	value	Everything lost by definition	same as the DCCF identity in notifications
S09	anaNotifUri	value	`malicious_nf`	
S10	anaNotifCorrId	value	`malicious_nf`	
S11	subscriptionId	value	`malicious_nf`	
S12	analytics events in anaSub	value	Nothing gained	event identifiers are public
S13	notification content	value	`malicious_nf`	
S14	OAuth2 access token	value	`malicious_nf`	
S15	TLS session keys	value	Nothing gained	TLS keys are internal to the private channels
S16	notification payload	value	`malicious_nf`	
S17	targetNfId/targetNfSetId	value	Nothing gained	NF identifiers are published by discovery
S18	anaSub	value	Nothing gained	the subscription content is the consumer's own data
S19	scope	value	Nothing gained	subject, audience and scope are public token fields
S20	subject	value	Nothing gained	subject, audience and scope are public token fields
S21	audience	value	Nothing gained	subject, audience and scope are public token fields
S22	subscription state	value	`malicious_nf`	
S23	upstream/downstream subscription mapping	value	`malicious_nf`	
S24	subscription request	value	Nothing gained	the request is what the consumer itself sent
S25	notification in transit	value	`malicious_nf`	

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		

Attacker	Has	Basis	Where
`malicious_nf`	a legitimate NF with its own token for the ndccf-datamanagement scope; it can call the DCCF API and choose anaNotifUri, events and target NWDAF	OAuth2 token from the NRF (s0412, s0413) and one service-wide scope (s0415): any NF holding the scope can call every operation.	s0412, s0415
`token_leak`	a valid access token issued to a legal NF, with its subject, audience and scope, obtained before the operation	RFC 6749 10.3: access tokens are bearer credentials; a leaked token is usable by its holder.	rfc6749:2859
`nwdaf_dishonest`	the NWDAF serving an upstream subscription is dishonest: it notifies events that were not subscribed with content of its choice	The NWDAF produces the analytics and posts Nnwdaf_EventsSubscription_Notify (s0997); nothing requires it to be honest.	s0997

ID	`network`	`malicious_nf`	`token_leak`	`nwdaf_dishonest`
A02	Holds	Holds		
A03	Holds	Holds		
A04	Holds			
K01	Holds	Holds		
D01	Holds	Holds		
A08	Holds			Holds
Z01	Holds	Holds		
Z03	Holds	Holds		
Z04	Holds			
S05	Holds			
U01	Holds			
U02	Holds	Holds		
U03	Holds			
X01	Holds	Holds		

Source and targets

Source: 1266 units, 927204 bytes, frozen in source-delivery-01.json. Targets: 13 approved (A02, A03, A04, K01, D01, A08, Z01, Z03, Z04, S05, U01, U02, U03); 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 the source package protocol-097945ce877ab933708785b53954

This step as it reports itself: 1 requests (1 failed), 0.000 USD, 388619 tokens in and 5952 out, 282 s.

Notes on the prover and the model

- Z04: answered in ProVerif, and the analyst names changing_state for this target
- S05: answered in ProVerif, and the analyst names changing_state for this target
- U03: answered in ProVerif, and the analyst names changing_state for this target

A ProVerif model that simplifies algebra (xor, dh_algebra, other_algebra) gives the attacker less than the real operation: a proof there is a proof against a weaker attacker, an attack there is an attack. A ProVerif model that leaves out a check on a value that changes (changing_state) lets parties accept more: a proof there holds, an attack there may be nothing but the missing check. Tamarin+ Agent models these as they are.

A bound in steps is an answer on bounded.pv: the analyst's second ProVerif model, in which the party that keeps a value that changes is unrolled to that many steps with its check in place. That model does less than the protocol does, so the property holds for those steps and no further, and an attack found there is an attack. The number of steps is the analyst's statement.

State review by the analyst

This part is the analyst's own reading of the source, not an answer of the prover. "yes" under shown by ProVerif means that ProVerif found a trace with these steps in the unrolled model, so the steps are possible; that they break a target is said only by an attack in the table of answers, and what else they cost is the analyst's reading. "nothing found" says which patterns were looked at and proves nothing.

value	kept by	changes when	checked when	refs
subscription record of the consumer (subscriptionId, anaNotifUri, anaNotifCorrId, the upstream subscriptionId)	DCCF	created when the DCCF accepts the analytics subscription request (s0026-s0029) and removed on an accepted DELETE (s0040, s0041)	when a notification for the subscription arrives from the NWDAF and is mapped to the consumer's anaNotifUri and anaNotifCorrId (s0060, s0061)	s0026, s0027, s0028, s0029, s0040, s0041, s0060, s0061
upstream NWDAF event subscription (subscriptionId issued by the NWDAF)	DCCF and NWDAF	created by the DCCF when serving the request (s0021) and removed by Nnwdaf_EventsSubscription_Unsubscribe (s0981-s0984)	no check of its state is specified before the NWDAF sends a notification (s0997)	s0021, s0981, s0982, s0983, s0984, s0985
phase of the subscription (open / terminated) as seen by the DCCF's notification path	DCCF	on an accepted DELETE (s0041)	the source does not state that the DCCF checks the subscription state when a notification arrives	s0040, s0041, s0060

pattern	result	target	shown by ProVerif	steps
replay	nothing found			In bounded.pv a notification of the terminated subscription arrives after the termination point and is dropped (event DroppedAfterTerm), and the direct query that the consumer ever accepts it again (late) is inconclusive: ProVerif's derivation reuses the earlier notification, which no trace can deliver twice. The injective query once is verified, so the consumer accepts each DCCF forward at most once.
repeat_one_time_action	nothing found			The DELETE is handled once per subscription in bounded.pv; a second DELETE has no path in the model.

pattern	result	target	shown by ProVerif	steps
update_before_check	candidate		no: not tried with ProVerif	The consumer subscribes for analytics notifications (s0005); the DCCF creates the upstream NWDAF subscription (s0021, s0026). / The consumer sends DELETE for its subscription; the DCCF removes its own subscription record and answers 204 (s0040, s0041, s0042). / Only after answering does the DCCF call Nnwdaf_EventsSubscription_Unsubscribe for the upstream subscription (s0981). / If that upstream DELETE fails or its 204 never comes, the source requires no retry and no rollback (s0985), so the NWDAF keeps its subscription while the DCCF has no record of it: the two sides keep different states.
desynchronisation	candidate		no: not tried with ProVerif	The DCCF treats the consumer's DELETE as the end of the subscription and answers 204 (s0041, s0042). / The upstream deletion is a separate request to the NWDAF (s0981); the source states no ordering or retry between the two. / A failure of the upstream deletion leaves the NWDAF subscribed and notifying, while the DCCF drops those notifications: the analytics keep being computed for nobody.
no_rollback_on_failure	nothing found			No failure path of the DCCF is modelled; the source specifies only that errors are answered as in clause 5.1.7 (s0043).
race_between_sessions	nothing found			model.pv gives each consumer its own TLS session and each subscription its own reply and notification channel, so two sessions do not read the same value; the source describes no shared subscription state between consumers.
distinguishable_failure	nothing found			Failure responses (400 SUBSCRIPTION_CANNOT_BE_SERVED, 403 USER_CONSENT_NOT_GRANTED) are specified but no failure path is modelled, so nothing can be distinguished here.
wrap_or_window	nothing found			The subscription identifier and the correlation identifier are opaque fresh values, with no counter and no acceptance window in this path.

What the analyst named for targets:

step	targets	features	reason
32	Z04, S05, U03	changing_state, plain_primitives	Z04, S05 and U03 rest on the subscription lifecycle: a subscription is created, notified on, and removed on DELETE, and the targets ask what happens at and after the termination point. The model.pv answers only show that the termination path is reachable, so these are unrolled in bounded.pv. The remaining targets use only hashing-free token construction, private channels and fresh names, i.e. plain primitives.

The analyst's closing note

Model built for the DCCF direct coordination path (consumer → DCCF → one NWDAF): subscribe at the DCCF, the DCCF's Nnwdaf_EventsSubscription subscription with its own notification URI, 201 Created with the subscriptionId, the upstream notification, the forwarded Ndccf_DataManagement_Notify under anaNotifCorrId, and the DELETE propagated upstream. TLS is a private channel per endpoint pair; each NF holds an OAuth2 token (private constructor with subject and scope) whose subject and scope the receiver checks; a malicious legal NF driven by the attacker asks for its own subscription with an address and event of its choosing.

All 14 targets have answers: A02, A03, A04, K01, D01, A08, Z01, Z03, U01, U02 and X01 by proof on model.pv; Z04, S05 and U03 by reachability of the termination path plus a proof on the unrolled lifetime in bounded.pv (two notification steps, control checked). K01 has a control that fails, so the secrecy query is not vacuous. No target failed, so there is no finding and the report says so. The proofs rest on the transport being TLS (a private channel per pair) and on the token subject/scope checks, neither of which the selected text spells out.

Unconfirmed state candidates: the DCCF removes its own subscription record before the upstream deletion is confirmed (s0041 against s0981), and a failed upstream deletion desynchronises the two sides; their steps are recorded in the state review, but no query shows them. Not modelled: user consent via the UDM, the muting flag, feature negotiation, redirects, and a leaked token, with which impersonation follows by definition while the documents ask for token confidentiality (K01).