Security analysis of MFAF configuration context transfer

This report in another language: 中文.

19.7.0 · ProVerif+ Agent report

Documents	5G System; Messaging Framework Adaptor Services; Stage 3 (protocol definition)
Properties examined	11: 7 hold, 4 fail (5 attacks)
Findings	3

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 Nmfaf_ContextManagement service of the Messaging Framework Adaptor Function moves an existing MFAF configuration from the MFAF that holds it to another MFAF, so that a consumer's data or analytics collection continues on the new MFAF. Three parties appear: the configurator, a DCCF or NWDAF, that created the Individual MFAF Configuration and holds its resource URI; the source MFAF that holds the configuration and its transaction reference; and the target MFAF that invokes the transfer and becomes the new holder. The NRF issues the access tokens both MFAFs use.

The target MFAF sends ContextTransferReq with the resource URIs of the configurations it wants, the source MFAF answers 200 OK with the configurations, the pre-transfer to post-transfer URI mapping and the target MFAF's notification information and removes its own copies, and the target MFAF creates each configuration under its new transaction reference id. When the exchange ends the target MFAF is sure that the response came from the source MFAF it addressed and carried exactly the configurations and notification bindings that source held; nothing in the procedure makes the source MFAF sure that the requester is entitled to the configuration.

The parties:

- **Configurator (DCCF)**: creates and later updates or removes the configuration. Holds: NF instance id, access token for nmfaf-3dadatamanagement.
- **Source MFAF**: holds the configuration and its transaction reference. Holds: configuration store, TLS credential.
- **Target MFAF**: invokes the transfer and becomes the holder. Holds: NF instance id, access token for nmfaf-contextmanagement, TLS credential.
- **NRF**: authorization server, issues access tokens. Holds: signing key.

Figure 1 shows the exchange as the documents specify it.

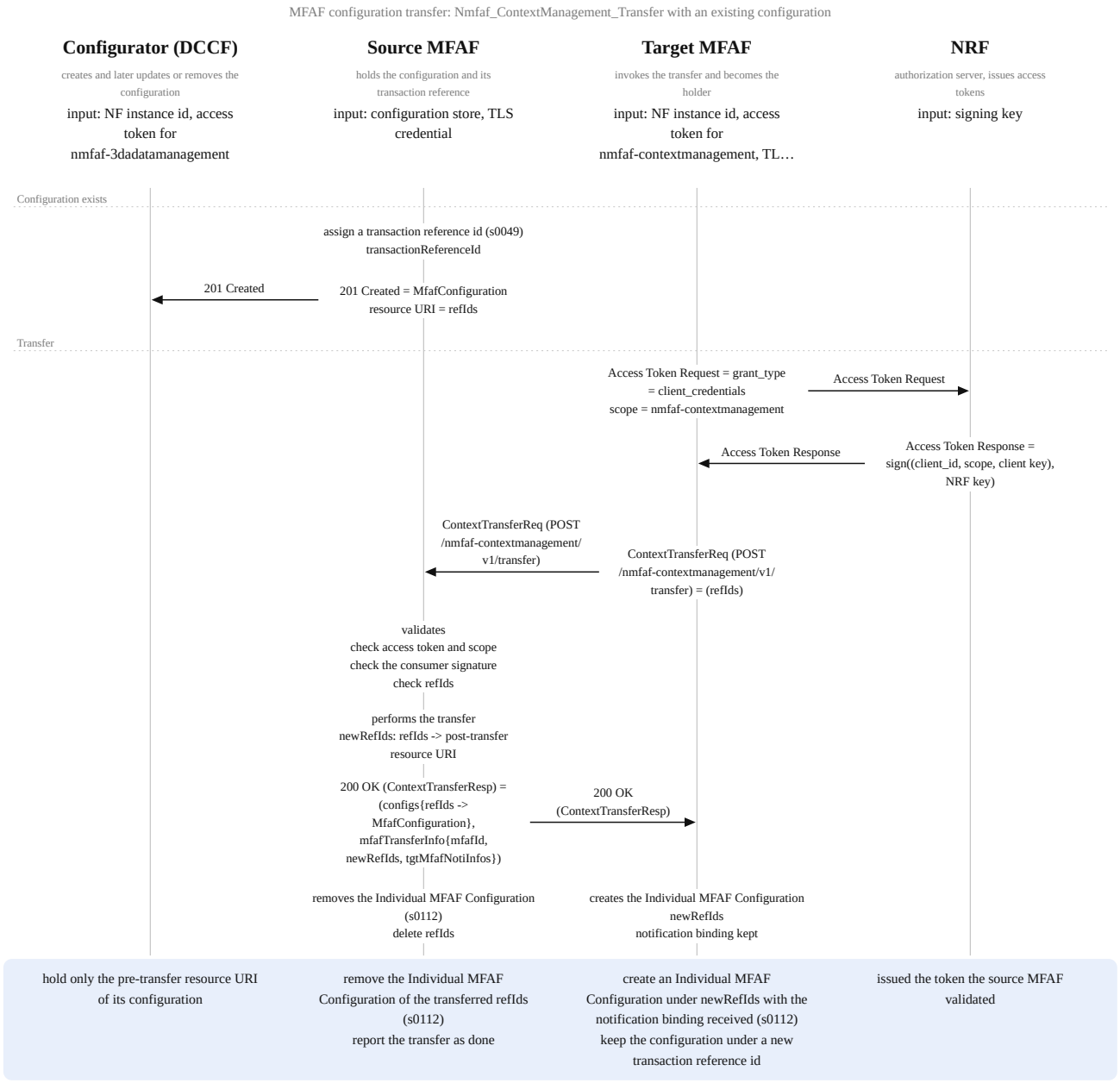


Figure 1. The protocol as specified.

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 3 findings: what follows is not stated in the documents. For 1 of them it is unclear whether the documents state it.

The procedure keeps what it promises about the exchange: a target MFAF accepts only a response that the source MFAF it addressed really sent, for the same refIds, and the migrated configuration keeps its notification binding. It does not keep the configuration where the configurator put it: any legitimate NF holding the service scope that knows a pre-transfer resource URI is handed that configuration, the source deletes its own copy in the same step in which it answers, and the configurator keeps a URI that no longer resolves after a migration. Three findings, none of them stated in the document; the rule that leaves authorization at the level of who can name a resource is the single

service scope the document does state. The answers rest on an idealised TLS and on the assumption that the transaction reference id is not guessable, and the replay and failure questions are answered only for the two steps the state model unrolls.

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 A legitimate NF with the service scope takes another consumer's configuration

Kind	Weakness of the protocol as specified
New	Unclear: the documents state it in part, or could not be searched
Value	high
Properties	Z01, S04 (section 3)
Source text	s0110, s0112, s0332, s0362, s0365
Shown by	ProVerif attack query `z01_attack`; Figures 2 and 3

What happens. A legitimate NF that invokes Nmfaf_ContextManagement_Transfer for another consumer's configuration is answered normally: the source MFAF validates the access token and the scope, finds the configuration the request names, returns it in configs and removes its own copy, and never checks that the requester is the consumer that created the configuration or the MFAF the configurator appointed. The only protection is that the pre-transfer resource URI must be named in refIds.

What the attacker needs. A legitimate NF instance with its own credentials and a valid NRF token for the nmfaf-contextmanagement scope, plus the pre-transfer resource URI of the configuration (the refIds value, which the model gives to the attacker). It needs no key, no compromised party and no protocol error.

What the attacker gains. It becomes the holder of that configuration: it reads the configuration content and the notification binding (notificationURI, correId, mfafNotiInfo) and the source MFAF deletes its own copy, so the configuration continues at a party the configurator never named.

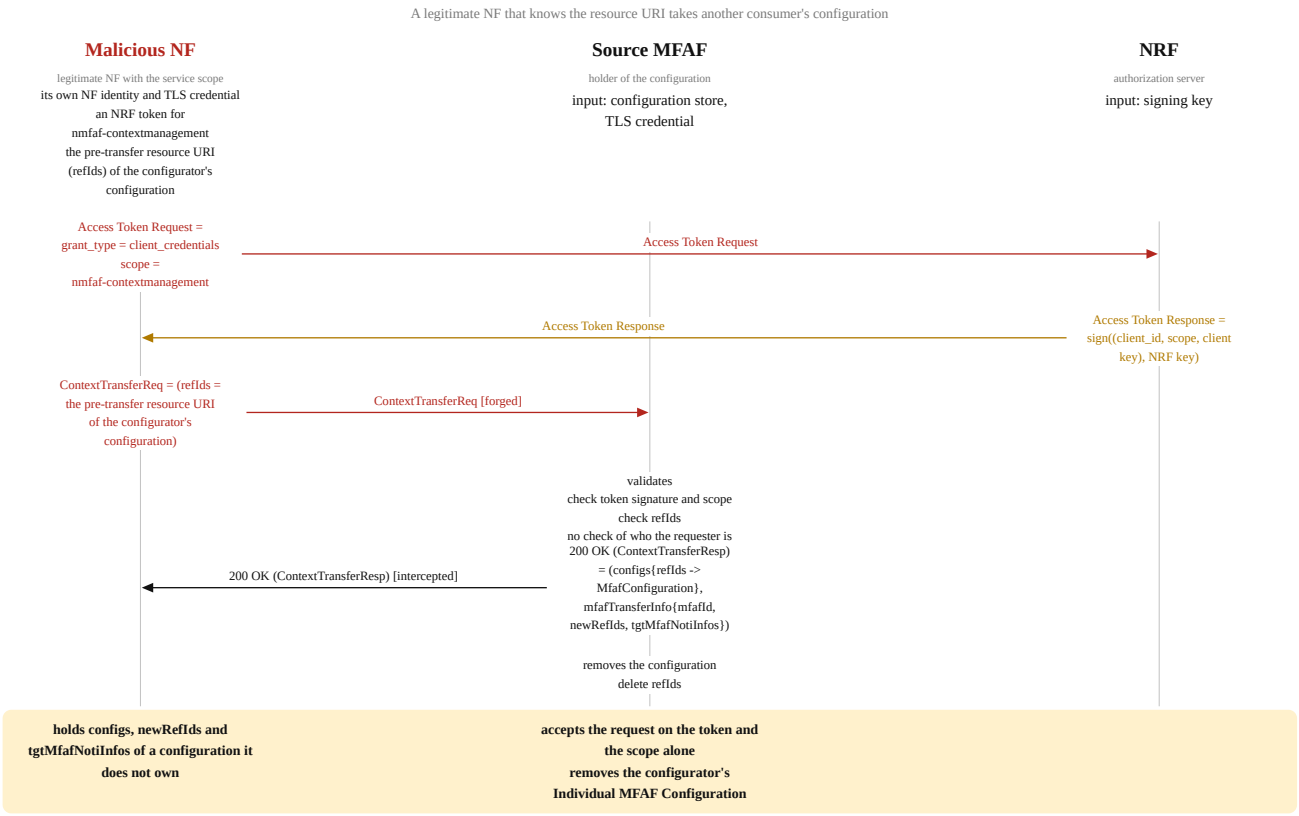


Figure 2. A legitimate NF that knows the resource URI takes another consumer's configuration. Property Z01; attack query z01_attack.

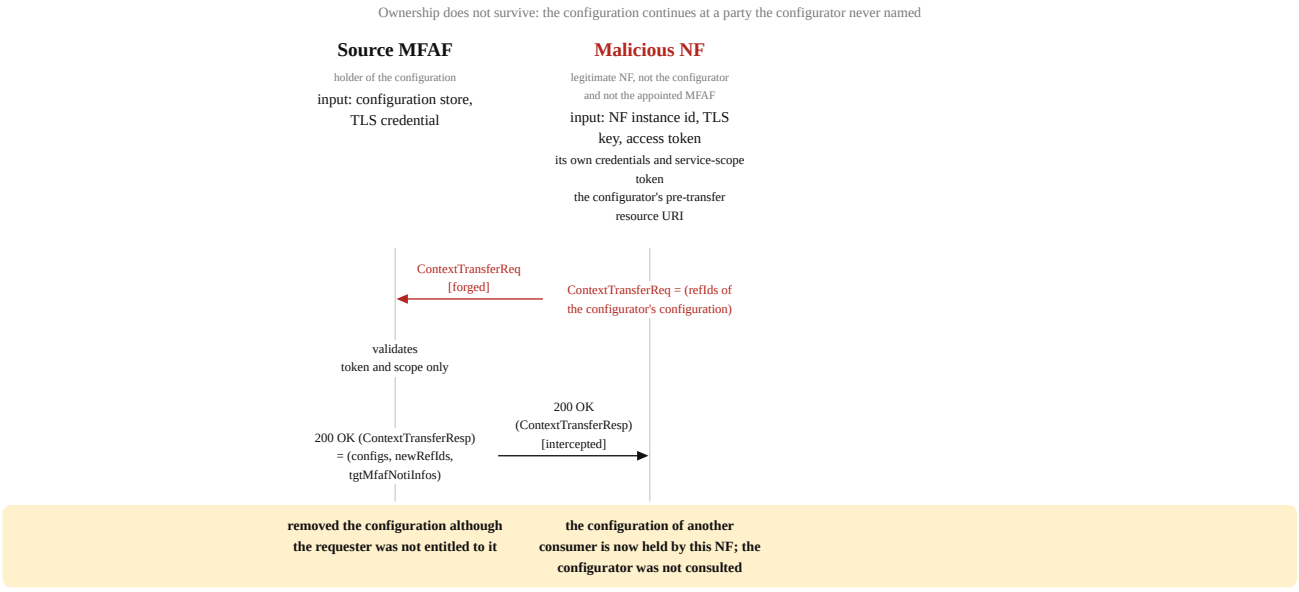


Figure 3. Ownership does not survive: the configuration continues at a party the configurator never named. Property S04; attack query z01_attack.

Proposed change. Bind the operation to the appointed MFAF instead of the service scope: let the configuration record the target MFAF the configurator appointed (the field mfafTransferInfo already names the holder MFAF and triggers the transfer, TS 29.572 s0037, s0051), and have the source MFAF compare the authenticated consumer with that value, answering 403 Forbidden otherwise, in addition to validating the token and its scope. Checked on a copy of the model: the attack query becomes verified, the normal run stays and no other recorded answer changes.

Check of the change with ProVerif. The change was put into a copy of the model: The configuration carries the target MFAF the configurator appointed for it (set when the configuration is created, as mfafTransferInfo names the holder and the transfer trigger, s0037, s0051). On a ContextTransferReq the source MFAF, in addition to validating the token and its scope, compares the authenticated consumer with that appointed MFAF and answers 403 Forbidden otherwise. On the changed model ProVerif proves `z01_attack`: the attack is no longer possible there. The normal run of the protocol is still found. The other 13 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/0045-fix-owner.pv`.

2.2 A dropped transfer response destroys the configuration at both MFAFs

Kind	Weakness of the protocol as specified
New	Yes: the documents do not state it
Value	high
Properties	U01 (section 3)
Source text	s0112, s0113
Shown by	ProVerif attack query `u01`; Figure 4

What happens. The source MFAF answers 200 OK with the configurations and removes its own copy in the same step (s0112), and the source text defines nothing for the case where that answer never reaches the target: no timer, no retry, no compensation, and s0113 covers only errors raised while processing the request. An attacker that drops the response therefore leaves the configuration deleted at the source and never created at the target.

What the attacker needs. Only the ability to drop one message on the network: the 200 OK from the source MFAF to the target MFAF. No credential, no key, no compromised party.

What the attacker gains. The configuration exists nowhere: the source has deleted it, the target never created it, the configurator's URI no longer resolves, and a retry by the target finds no configuration at the source. The data delivered under that configuration is lost until the configurator configures an MFAF again.

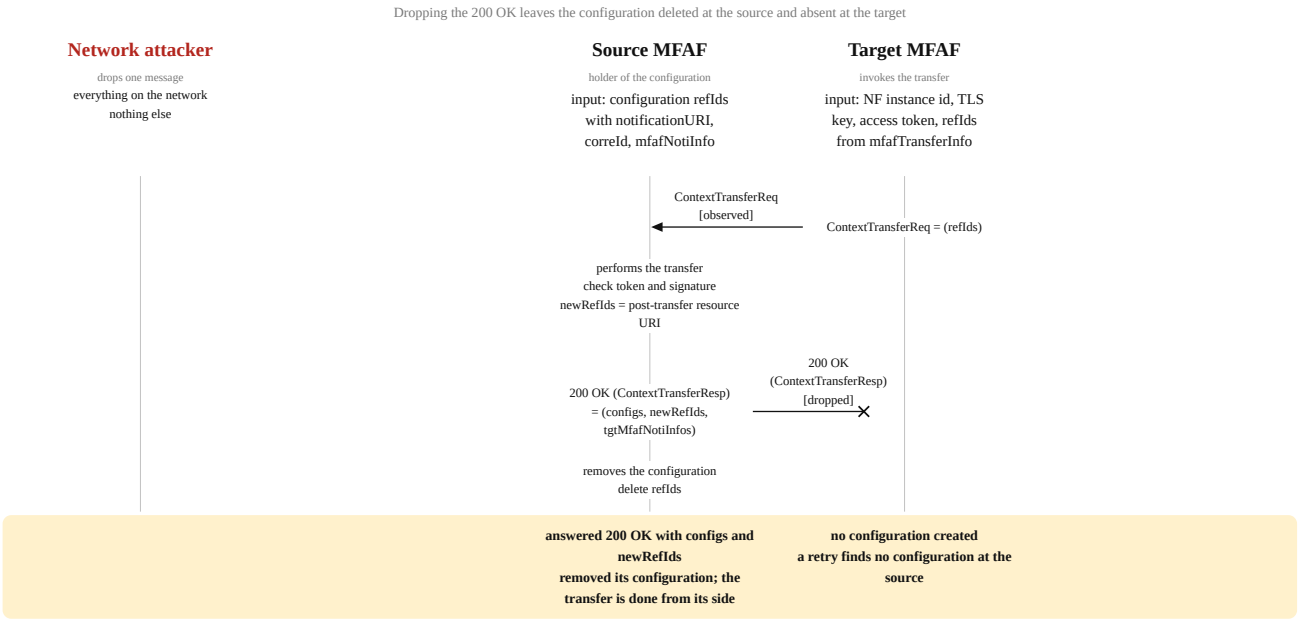


Figure 4. Dropping the 200 OK leaves the configuration deleted at the source and absent at the target. Property U01; attack query u01.

Proposed change. Keep the source-side configuration, marked as transferred, until the migrating consumer confirms the creation, and answer a repeated transfer request for the same refIds with the same newRefIds instead of an error, so that a lost response can be repaired; report the removal as complete only after that confirmation.

Check of the change with ProVerif. Not checked: Removing the weakness means changing the exchange, not only a check: either the source MFAF keeps the configuration until the consumer confirms the creation, which is a message the procedure does not have, or the target's retry must be answered from a copy the source keeps for a transfer identifier. Either way the fixed file would no longer be a one-change copy of the modelled flow, so it was not checked.

2.3 The configurator loses the handle to its configuration after a migration

Kind	Weakness of the protocol as specified
New	Yes: the documents do not state it
Value	medium
Properties	X02, S04 (section 3)
Source text	s0055, s0071, s0112, s0234
Shown by	ProVerif attack query `x02`; Figures 5 and 6

What happens. After the migration the source MFAF has deleted the configuration and the target MFAF holds it under a new transaction reference id, but newRefIds appears only in the response to the consumer that invoked the transfer (s0234). The configurator that created the configuration keeps the old resource URI, and its Update and Deconfigure requests are addressed to that URI (s0055, s0071), which no longer resolves anywhere.

What the attacker needs. Nothing beyond a normal successful migration: the configuration must simply be migrated once.

What the attacker gains. The configurator cannot Update or Deconfigure the configuration any more, so it can neither change the delivery of the consumer's data nor remove the configuration; only the MFAF that holds it and its new reference remain valid.

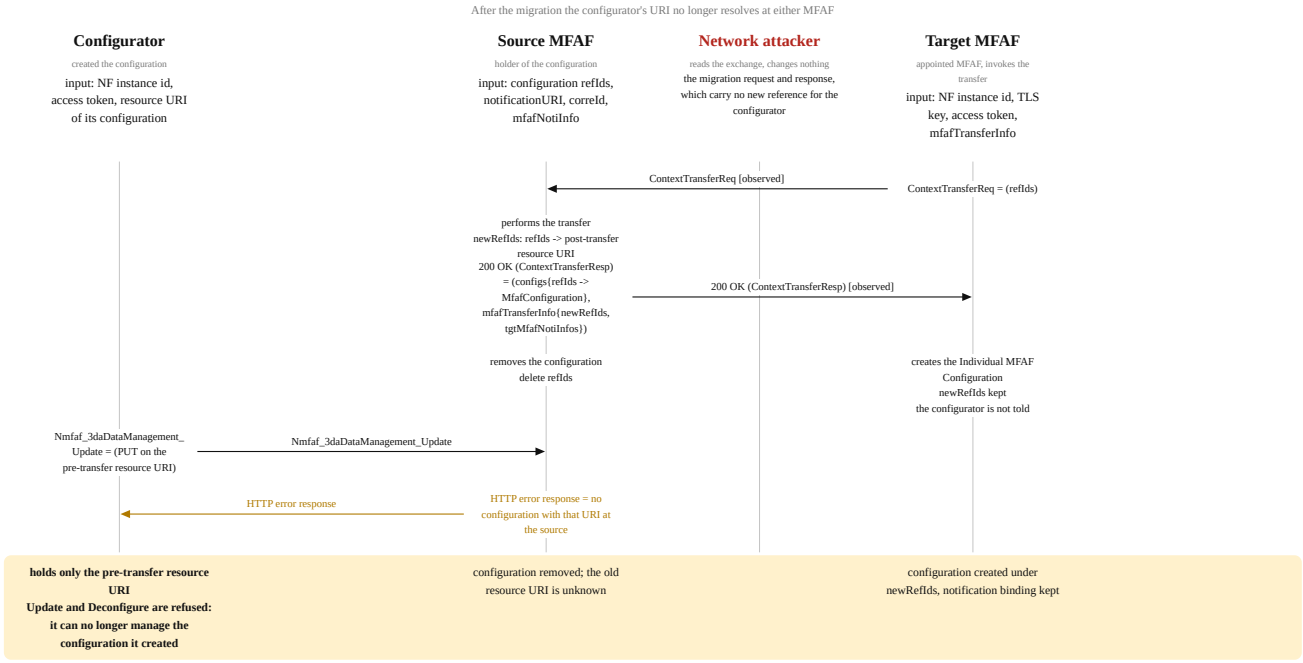


Figure 5. After the migration the configurator's URI no longer resolves at either MFAF. Property X02; attack query x02.

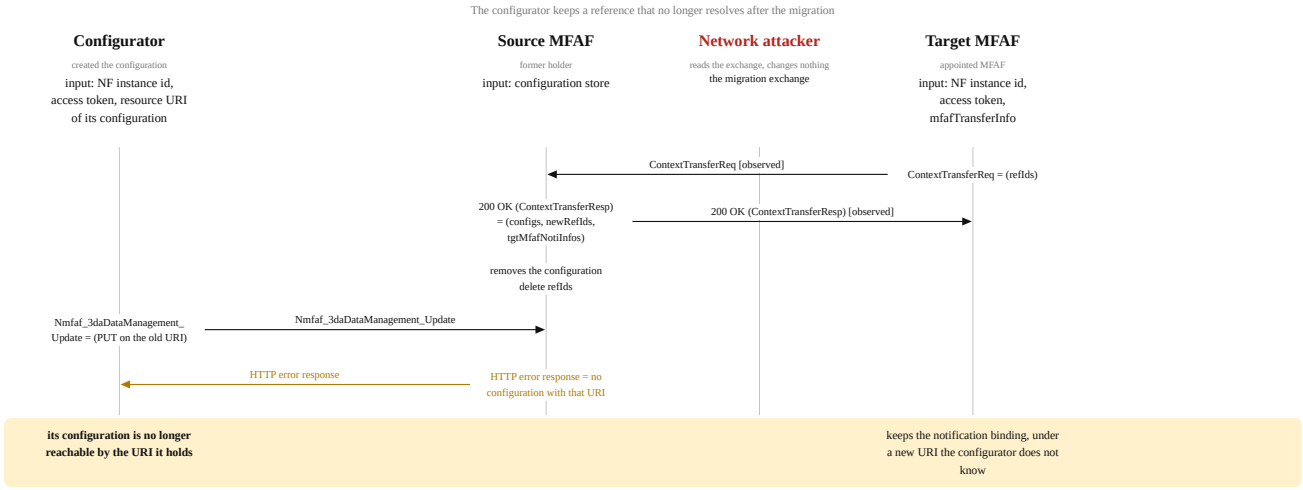


Figure 6. The configurator keeps a reference that no longer resolves after the migration. Property S04; attack query x02.

Proposed change. Return the newRefIds mapping of a transferred configuration to the consumer that created it, for example in the TargetMfafNotiInfo the target MFAF already receives, or let the source MFAF keep a forwarding record from the old resource URI to the new one for the duration of the migration.

Check of the change with ProVerif. Not checked: The remedy gives the new reference to a party the response does not address today (the configurator), which is a new message or a new field in a notification rather than a check on an existing message; the model would have to add that exchange, so it was not checked.

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	**Accepted response matches a real transfer.** Whenever the target MFAF accepts a migration response (200 OK containing configs), there is a real migration run of the source MFAF, and the two agree on the source MFAF identity, the target MFAF identity, the set of refIds and the corresponding configuration content.	no compromise	Holds	medium	no
A08	**Migration preserves reference and callback mapping.** Every newRefIds entry accepted by the target MFAF corresponds to the same refIds entry in the request, and the configuration's notificationURI, correId, notifEndpoints and mfafNotiInfo agree with the original configuration held by the source MFAF; the keys of tgtMfafNotiInfos are taken from refIds in the request and from newRefIds in the response, and the inner key is correId.	no compromise	Holds	medium	no
D01	**Requests and responses are authentic.** Every ContextTransferResp accepted by the target MFAF corresponds to a real send by the source MFAF, and refIds, configs and newRefIds agree with the configuration the source MFAF actually holds; every ContextTransferReq accepted by the source MFAF corresponds to a real send by the target MFAF.	no compromise	Holds	medium	no
R03	**Transfer side effects happen at most once.** For the same logical migration operation (same set of refIds, same source/target MFAF pair), the source MFAF's local deletion side effect and the target MFAF's resource creation side effect are each committed at most once; the request may be retransmitted, but must not lead to duplicate deletion or duplicate creation.	no compromise	Holds within a bound (the changing value unrolled to 2 steps)	low	no
S04	**Configuration ownership unchanged by migration.** For every configuration migrated on the target MFAF, its source configuration (the source-side resource that refIds points to), the identity of the NF service consumer that created the configuration, and the notificationURI/correId binding agree with the corresponding configuration on the source MFAF before the migration; the migration does not change the owning subject of the configuration.	no compromise	Fails (2 attacks)	high	no
Z01	**Transfer is authorized by a valid token.** Every migration performed by the source MFAF (deleting the local configuration and returning configs) carries a valid access token issued by the NRF and held by the target MFAF, and the token's scope covers nmfaf-contextmanagement; a legitimate NF holding only that scope cannot migrate a configuration it is not entitled to own.	no compromise	Fails (1 attack)	high	unclear
A03.7	**Identity and refIds binding of response.** The migration response accepted by the target MFAF agrees with the real run of the source MFAF on the source/target MFAF identities and on refIds.	no compromise	Holds	medium	no
A08.8	**Reference and endpoint mapping consistency.** Every newRefIds entry accepted by the target MFAF corresponds to the same refIds entry in the request, and notificationURI, correId, notifEndpoints and mfafNotiInfo agree with the original configuration on the source side.	no compromise	Holds	medium	no
U01	**No half-committed success.** The success determination of the migration and the side effects at both ends (source-side deletion, target-side creation) agree on the full-success and partial-success/failure paths; there is no case where one end has committed and the other has not, yet the outcome is treated as complete success.	no compromise	Fails (1 attack)	high	no

ID	Property	Compromise assumed	Result	Value	In the documents
X01	**Configuration content stays confidential in transit.** An external network attacker cannot learn the configs (notificationURI, correId, mfafNotiInfo) and bufferedNotifs in the migration response, nor the refIds in the request: no value driven by a network attacker depends on this content.	no compromise	Holds	medium	no
X02	**Configurator keeps its handle after migration.** After a successful migration, the NF service consumer that created the configuration can still address it in order to Update or Deconfigure: either the source text has the source/target MFAF still serve the configurator under the old URI after the migration, or it sends the new resource URI to the configurator.	no compromise	Fails (1 attack)	medium	no

Goals the documents state that the drafted list of properties did not have, added by the analyst (Appendix D): X01, X02.

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: Migration context bound to the source/target MFAF identity	Yes, against a network attacker: the accepted response is bound to the source MFAF the target addressed, to the target itself, and to the refIds and content that source really held; it fails only for an attacker that already holds the pre-transfer resource URI, which is a name and not an identity binding.	A03, A03.7, D01
Q2: Consistency of the old and new references and the notification endpoint mapping	The refIds to newRefIds mapping and the notification binding are preserved on every migration, but the new URI reaches only the MFAF that invoked the transfer, so the configurator that created the configuration keeps a reference that no longer resolves.	A08, A08.8, X02
Q3: A failed or partially successful migration must not be treated as complete success	No: a transfer whose response is dropped is not reported as success anywhere, but the source has already deleted the configuration, so the outcome is the loss of the configuration rather than a recoverable failure; partial success inside one response is outside the modelled bound.	U01, X02

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 between two NF endpoints is idealised as authenticated encryption plus a signature by each endpoint's long-term key, with the identity binding carried by the NRF-signed token; key agreement, certificate handling and record-layer replay protection are not modelled.
- The OAuth2 token is a signed statement of client identity, scope and client key; audience, expiry, claim matching and token reuse across requests are not modelled, and the NRF is a single trusted token issuer.
- One configuration per transfer: the multi-entry case (configs, refIds, bufferedNotifs with 1..N entries) and partial success inside one response are not modelled.
- Rejection paths are modelled as refusal, not as HTTP status codes and ProblemDetails, so the failure-oracle question (whether an error code tells a requester that a resource URI exists) is recorded in the state review as a candidate and not proved.
- The 307/308 redirect responses of the transfer operation, with Location and the optional 3gpp-Sbi-Target-Nf-Id provided by an SCP, are not modelled; indirect communication has no mutual authentication per TS 33.501

5.9.2.1, so a redirected transfer request is outside what was analysed.

- The configuration content is an opaque value; MessageConfiguration, notifEndpoints and MfafNotiInfo appear only through the notificationURI/correId binding, and the notification delivery flow (MFAF to consumer) is outside the modelled subflow.
- The source MFAF is unrolled to two steps in bounded.pv (configuration present, then removed) and the target to one, so longer runs, a configuration re-created under the same URI, and several migrating consumers at once are outside that model.
- The analysis covers one PLMN and direct communication between two MFAFs; roaming, SEPP and trust-zone boundaries are not modelled.

What the host records:

- The answers of A08, R03, S04, A08.8, U01, X02 rest on an operation the model simplifies (Appendix D).
- Proved without a control query (Appendix C): A08: a08. 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.

A03: Accepted response matches a real transfer

Whenever the target MFAF accepts a migration response (200 OK with configs), there is a real migration run of the source MFAF, and the two agree on the source MFAF identity, the target MFAF identity, the set of refIds and the corresponding configuration content.

The target MFAF accepts a configuration only when the source MFAF it addressed really sent it, for the same refIds and with the same content, and this holds against an attacker on the network. The guarantee rests on the mutual TLS of the direct NF-to-NF exchange, which the source text leaves to TS 33.501 and TS 29.500.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It is the central assurance of the operation, proved without an increase in attacker power, but its strength comes from the baseline TLS that the model idealises.
- Stated in the documents: no. Source text it rests on: s0110, s0112, s0332, s0335
- Rests on: TLS is modelled as authenticated encryption plus a signature per endpoint, with the endpoint identity bound by the NRF-signed token.; The target knows the source MFAF's public key out of band (discovery or certificate), modelled as a parameter.; One configuration per transfer.

A08: Migration preserves reference and callback mapping

Every newRefIds entry the target MFAF accepts corresponds to the same refIds entry in the request, and the configuration's notificationURI, correId, notifEndpoints and mfafNotiInfo agree with the original configuration held by the source MFAF; the keys of tgtMfafNotiInfos are taken from refIds in the request and from newRefIds in the response, and the inner key is the correId.

The configuration the target MFAF creates keeps the notification binding and the content the source MFAF held, and the new URI corresponds to the refIds entry that was requested. What the protocol does not check is that the requester may have that configuration.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A proof of a stated consistency requirement (s0234, s0237), with the keying simplification named.
- Stated in the documents: no. Source text it rests on: s0231, s0234, s0237, s0335
- Rests on: The source MFAF is treated as the authority on the mapping it returns; it chooses the new transaction reference id, which the source text does not settle (s0112 speaks of the consumer assigning it, s0234 of the producer including it).; The target MFAF notification information is represented as one entry keyed by the new URI whose inner key is the correId; the map structure and notifEndpoints are not modelled separately.

D01: Requests and responses are authentic

Every ContextTransferResp the target MFAF accepts corresponds to a real send by the source MFAF, and refIds, configs, newRefIds agree with the configuration the source MFAF actually holds; every ContextTransferReq the source MFAF accepts corresponds to a real send by the target MFAF.

A request the source MFAF accepts really came from the consumer whose identity it carries, and a response the target MFAF accepts really came from the source it addressed. Authenticity is not authorization here: the source MFAF learns who called, not whether that caller may take the configuration.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It shows exactly where authenticity stops and authorization is missing, which is the root of the main finding.
- Stated in the documents: no. Source text it rests on: s0110, s0112, s0362
- Rests on: TLS endpoints idealised as above; the consumer identity is carried by the NRF-signed token and matched with the TLS client signature.; Refusals are modelled as the party stopping, not as HTTP codes.

R03: Transfer side effects happen at most once

For the same logical migration operation (the same set of refIds, the same source/target MFAF pair), the source MFAF's local deletion side effect and the target MFAF's resource creation side effect are each committed at most once; the request may be retransmitted, but must not cause a duplicate deletion or a duplicate creation.

Within the unrolled run the source MFAF removes a configuration at most once, however often the request is replayed; the second request is refused because the configuration is gone. A source that kept the configuration would remove it twice, which the control variant shows.

- Result: Holds within a bound (the changing value unrolled to 2 steps); answers what is asked: yes; usable with caveats
- Value low: At-most-once here follows from the removal itself rather than from a replay counter, and it is verified only over the unrolled steps.
- Stated in the documents: no. Source text it rests on: s0112
- Rests on: bounded.pv unrolls the source to two steps and the target to one; the response side of a replay is not unrolled.; The source MFAF is not replicated, so two requesters are served one after the other.
- Open: Whether a replayed 200 OK makes the target MFAF create the same resource twice is not answered: the target is unrolled to one response, and a replayed message names the same new URI.

S04: Configuration ownership unchanged by migration

For every configuration migrated to the target MFAF, its source configuration (the source-side resource pointed to by refIds), the identity of the NF service consumer that created it, and the notificationURI/correId binding agree with the corresponding configuration on the source MFAF before migration; migration does not change the owning subject of the configuration.

The binding of the migrated configuration to its notificationURI and correId is preserved, so the configuration stays the configurator's configuration. Ownership as control does not survive: a legitimate NF that knows the pre-transfer URI receives the configuration, and after any migration the configurator can no longer address it.

- Result: Fails (2 attacks); answers what is asked: partly; usable
- Value high: It separates what the protocol does preserve (the notification binding) from what it does not (control over who holds the configuration and by which reference).
- Stated in the documents: no. Source text it rests on: s0035, s0055, s0071, s0112, s0234
- Rests on: The two failures are answered on different models: the third-party takeover in model.pv with the refIds given to the attacker, the lost handle in bounded.pv.; Update and Deconfigure are modelled by one Update message to the old URI, sharing the token machinery of the transfer.

- Open: Whether a dishonest MFAF or a dishonest configurator can direct a configuration elsewhere: they can, by their own authorization, which the documents do not restrict.; Whether a second consumer sharing the callback endpoint changes the binding: one configuration per transfer in the model.

Z01: Transfer is authorized by a valid token

Every migration the source MFAF executes (deleting the local configuration and returning configs) is backed by a valid access token held by the target MFAF and issued by the NRF, and the token's scope covers nmfaf-contextmanagement; a legitimate NF holding only that scope cannot migrate a configuration it is not entitled to own.

Every transfer the source executes is backed by a valid token with the service scope, and a legitimate NF that does not know the pre-transfer URI is stopped. Stopped means only that it cannot name the configuration: the resource name is the authorization, and an NF that knows it obtains the configuration without any ownership check.

- Result: Fails (1 attack); answers what is asked: yes; usable with caveats
- Value high: The proof shows the protection is the unguessable resource name and not authorization, which is the fact an editor needs; the source states the scope rule but not this consequence.
- Stated in the documents: unclear. Source text it rests on: s0362, s0363, s0365
- Rests on: The refIds are modelled as unguessable names; the attacker that knows them is the separate attacker refid_known, an assumption and not something the source text states.; The token is checked for signature, subject and scope; audience, expiry and reuse are not modelled.
- Open: Whether a deployment enforces an ownership check outside the API, which the source text neither requires nor forbids.

A03.7: Identity and refIds binding of response

The migration response the target MFAF accepts agrees with a real run of the source MFAF on the source/target MFAF identities and on the refIds.

The accepted response agrees with a real source-side run on both MFAF identities and on the refIds. Nothing in the exchange lets a network attacker make the target accept a response of another transfer or another source.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The same proof as A03, stated for the identity and refIds binding the user asked about.
- Stated in the documents: no. Source text it rests on: s0110, s0112, s0332, s0335
- Rests on: Same TLS idealisation and identity binding as A03.; One configuration per transfer.

A08.8: Reference and endpoint mapping consistency

Every newRefIds entry the target MFAF accepts corresponds to the same refIds entry in the request, and notificationURI, correId, notifEndpoints, mfafNotiInfo agree with the original configuration on the source side.

Each accepted new URI corresponds to the refIds entry that was requested and carries the same notification data. The correspondence is guaranteed by the source MFAF, which is trusted for the mapping it returns.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A proof of the mapping requirement the user asked about, with the map structure simplified.
- Stated in the documents: no. Source text it rests on: s0234, s0237, s0335
- Rests on: The keying of the target MFAF notification information by newRefIds and of its inner map by correId is represented by passing the correId with the entry, not by a map structure.; notifEndpoints and

MessageConfiguration are not modelled separately.

U01: No half-committed success

The success determination of the migration and the side effects at both ends (deletion at the source, creation at the target) stay consistent across the full-success and partial-success/failure paths: one end never appears as a complete success while the other end has not committed.

The source MFAF removes its configuration in the same step in which it answers, so a dropped response leaves the configuration deleted at the source and never created at the target. There is no retry, timer or compensation, and the second attempt at the source finds nothing, so the failure cannot be recovered from within the procedure.

- Result: Fails (1 attack); answers what is asked: yes; usable
- Value high: A loss of configuration with a network-only attacker, on a path the source text simply does not describe.
- Stated in the documents: no. Source text it rests on: s0112, s0113, s0349
- Rests on: The failure is modelled by the attacker dropping the response; the source is not told, and the source text gives no confirmation step.; One configuration per transfer, so partial success reduces to this case.
- Open: Partial success within one response, that is some configurations of a multi-entry request transferred, is outside the modelled bound.

X01: Configuration content stays confidential in transit

An external network attacker cannot learn the configs (notificationURI, correId, mfafNotiInfo) and bufferedNotifs in the migration response, nor the refIds in the request: no value the network attacker drives depends on this content.

A network attacker that only relays, changes or drops messages learns neither the configuration content and its notification binding nor the pre-transfer resource URIs. This holds because every exchange is encrypted to the addressed endpoint and signed by the sender.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It states the transport protection the baselines require (TS 33.501 5.9.2.1) for the values this operation carries, and shows the protection is only as good as the resource name.
- Stated in the documents: no. Source text it rests on: s0335, s0231
- Rests on: TLS modelled as authenticated encryption to the endpoint key; the endpoint keys are obtained out of band.; Content values are modelled as fresh names; guessing is outside the symbolic model.
- Open: The strength of the protection is exactly the unguessability of the transaction reference id, which the source text does not require; if it were enumerable the content would be exposed.

X02: Configurator keeps its handle after migration

After a successful migration, the NF service consumer that created the configuration can still address it in order to Update or Deconfigure it: the source text either has the source/target MFAF continue to serve that configuration under the old URI after migration, or sends the new resource URI to that configurator.

After a migration the configurator's Update for the configuration it created is refused by the source MFAF, because the configuration is no longer there and nothing tells the configurator the new URI. Before a migration the same request is accepted, which is the control.

- Result: Fails (1 attack); answers what is asked: yes; usable

- Value medium: It names the exact consequence of returning newRefIds only to the migrating consumer, which the source text does not discuss.
- Stated in the documents: no. Source text it rests on: s0055, s0071, s0112, s0234
- Rests on: bounded.pv unrolls the source to two steps, so the refusal is reached only after the removal in the same run.; Update is modelled by one message with the same token machinery as the transfer; the data management API of clause 4.2 is otherwise outside the model.

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	checked		2 proved
A08	proved		a08	3 of 5		5 proved
D01	proved		d01	checked		3 proved
R03	bounded	the changing value unrolled to 2 steps	r03	checked		2 bounded
S04	proved		a08	checked	z01_attack (holds_in_source); x02 (holds_in_source)	2 proved, 2 attack
Z01	proved		z01_tok	checked	z01_attack (holds_in_source)	4 proved, 1 attack
A03.7	proved		a03	checked		2 proved
A08.8	proved		a08	checked		3 proved
U01	none				u01 (holds_in_source)	1 attack
X01	proved		x01_bind	checked		2 proved
X02	none				x02 (holds_in_source)	1 attack

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	Every 200 OK with configs that the target MFAF accepts corresponds to a real transfer run at that source MFAF, with the same source MFAF identity, the same target MFAF identity, the same refIds and the same configuration content.	`network`	proved	a03	checked
A03	Every 200 OK with configs that the target MFAF accepts corresponds to a real transfer run at that source MFAF, with the same identities, refIds and content, even when a legitimate NF with its own credentials and service-scope token is in the network.	`malicious_nf`	proved	a03	checked
A08	Every newRefIds entry the target MFAF accepts corresponds to the same refIds entry it requested, and the migrated configuration keeps the notificationURI, correId, notifEndpoints and mfafNotiInfo of the configuration the source MFAF held, also when a legitimate NF with its own credentials acts in the network.	`malicious_nf`	proved	a08	none
A08	The mapping and the notification binding are preserved on every migration the target MFAF performs, also for an attacker that holds the pre-transfer resource URI of another consumer's configuration.	`refid_known`	proved	a08	none

target	claim	attacker	answer	query	control
A08	Every newRefIds entry the target MFAF accepts corresponds to the same refIds entry it requested, and the migrated configuration keeps the notificationURI, correId, notifEndpoints and mfafNotiInfo of the configuration the source MFAF held.	`network`	proved	a08	checked
A08	Every newRefIds entry the target MFAF accepts corresponds to the same refIds entry it requested, and the migrated configuration keeps the notification binding of the configuration the source MFAF held, also when a legitimate NF with its own credentials acts in the network.	`malicious_nf`	proved	a08	checked
A08	The mapping and the notification binding are preserved on every migration, also for an attacker that holds the pre-transfer resource URI of another consumer's configuration.	`refid_known`	proved	a08	checked
D01	Every ContextTransferResp the target MFAF accepts corresponds to a real send by the source MFAF to that target.	`network`	proved	a03	checked
D01	Every ContextTransferReq the source MFAF accepts corresponds to a real send by that consumer MFAF to that source.	`network`	proved	d01	checked
D01	Every ContextTransferReq the source MFAF accepts corresponds to a real send by that consumer MFAF, also when a legitimate NF with its own credentials sends requests.	`malicious_nf`	proved	d01	checked
R03	For one logical migration the source MFAF's local removal happens at most once, however often the request is replayed, and the target MFAF creates at most one resource for it.	`network`	bounded (the changing value unrolled to 2 steps)	r03	checked
R03	For one logical migration the source MFAF's local removal happens at most once, also when a legitimate NF with its own credentials replays the transfer request.	`malicious_nf`	bounded (the changing value unrolled to 2 steps)	r03	checked
S04	The configuration does not land at a third party that holds its own credentials and the service scope but does not know the pre-transfer resource URI, and the binding preserved is the configurator's.	`malicious_nf`	proved	z01_noleak	checked
S04	The configuration is handed only to the MFAF the configurator designated: a third party that knows the pre-transfer resource URI does not obtain it, for an attacker that holds the pre-transfer resource URI beyond the network.	`refid_known`	attack (holds_in_source)	z01_attack (the control of z01_noleak)	
S04	After the migration the configurator can still address its configuration: its Update for that configuration is still accepted by the MFAF that holds it.	`network`	attack (holds_in_source)	x02	
S04	The migrated configuration keeps the notificationURI/correId binding and the content the source MFAF held, so after the migration it is still the configurator's configuration and not a new one.	`network`	proved	a08	checked
Z01	A legitimate NF that holds only the service scope cannot obtain a configuration it does not own while it does not know the pre-transfer resource URI: the unguessable resource name is what stops it, not an ownership check.	`network`	proved	z01_noleak	checked

target	claim	attacker	answer	query	control
Z01	A legitimate NF holding only the service scope does not obtain a configuration it does not own unless it knows the pre-transfer resource URI; its own token gives it no ownership-based restriction.	`malicious_nf`	proved	z01_noleak	checked
Z01	A legitimate NF that holds only the service scope but knows the pre-transfer resource URI of another consumer's configuration obtains that configuration, for an attacker that holds the pre-transfer resource URI (refIds) beyond the network; the source MFAF performs no check that the requester owns or is authorized for that configuration.	`refid_known`	attack (holds_in_source)	z01_attack (the control of z01_noleak)	
Z01	Every transfer the source MFAF executes is backed by a valid NRF-issued access token with the nmfaf-contextmanagement scope, held by that consumer.	`network`	proved	z01_tok	checked
Z01	Every transfer the source MFAF executes is backed by a valid NRF-issued access token with the nmfaf-contextmanagement scope held by that consumer, also for a legitimate NF that holds its own token.	`malicious_nf`	proved	z01_tok	checked
A03.7	The accepted migration response and a real source-side run agree on the source MFAF identity, the target MFAF identity and the refIds.	`network`	proved	a03	checked
A03.7	The accepted migration response and a real source-side run agree on both MFAF identities and the refIds, also when a legitimate NF with its own credentials acts in the network.	`malicious_nf`	proved	a03	checked
A08.8	Each accepted newRefIds entry corresponds to the same refIds entry of the request, and the notificationURI, correId, notifEndpoints and mfafNotiInfo match the source's original configuration.	`network`	proved	a08	checked
A08.8	Each accepted newRefIds entry corresponds to the same refIds entry of the request and the notification data matches the source's original configuration, also when a legitimate NF with its own credentials acts in the network.	`malicious_nf`	proved	a08	checked
A08.8	Each accepted newRefIds entry corresponds to the same refIds entry of the request and the notification data matches the source's original configuration, also when the attacker knows the pre-transfer resource URI.	`refid_known`	proved	a08	checked
U01	The source MFAF removes a configuration only after the target MFAF has created it: a lost response must not leave the configuration deleted at the source and absent at the target.	`network`	attack (holds_in_source)	u01	
X01	The configuration content and its notification binding (configs, notificationURI, correId, mfafNotiInfo) stay unknown to a network attacker on the migration response.	`network`	proved	x01_bind	checked
X01	The pre-transfer resource URIs (refIds) stay unknown to a network attacker on the migration request.	`network`	proved	x01_ref	checked
X02	After the migration the configurator can still address its configuration with the resource URI it holds: its Update is accepted, or the new URI is given to it.	`network`	attack (holds_in_source)	x02	

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-20261115T083807, 2026-11-15. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 63, prover 5 s, wall 19 min, cost 0.139 USD.

ProVerif: 89 proofs run, 89 decided, 0 without an answer, 3 s, 39 answers recorded.

What was analysed

Protocol	MFAF configuration context transfer
Version	19.7.0
Analysed	A bounded migration request and response for a fixed existing MFAF configuration from the old MFAF to the target MFAF; one legitimate configurator; retention of the transaction reference, the source/target NF identities and the callback endpoint mapping; modelled only where the specification's applicability conditions hold, not the full UE mobility procedure.
Not analysed	The symbolic model does not prove the computational security or post-quantum resistance of the algorithms; attack traces hold only under explicit assumptions and are not directly equivalent to specification vulnerabilities. Every property must have a reachability/positive control. Fix Rel-19. Distinguish the external network attacker from a malicious legitimate NF holding its own credentials; 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; the service scope does not automatically equal resource ownership. Monotonic tables cannot model real deletion; the finite-state lifecycle must be clearly delimited. The target is the specified API sub-flow, not complete 5G security.
Source text	406 numbered passages (160566 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	Nmfaf_3daDataManagement Service
main	4.2.1	Service Description
main	4.2.1.1	Overview
main	4.2.1.2	Service Architecture
main	4.2.1.3	Network Functions
main	4.2.1.3.1	Messaging Framework Adaptor Function (MFAF)
main	4.2.1.3.2	NF Service Consumers
main	4.2.2	Service Operations
main	4.2.2.1	Introduction
main	4.2.2.2	Nmfaf_3daDataManagement_Configure service operation
main	4.2.2.2.1	General
main	4.2.2.2.2	Initial configuration of the MFAF
main	4.2.2.2.3	Update the configuration of the MFAF

Document	Section	Title
main	4.2.2.3	Nmfaf_3daDataManagement_Deconfigure service operation
main	4.2.2.3.1	General
main	4.2.2.3.2	Remove MFAF configuration
main	4.4	Nmfaf_ContextManagement Service
main	4.4.1	Service Description
main	4.4.1.1	Overview
main	4.4.1.2	Service Architecture
main	4.4.1.3	Network Functions
main	4.4.1.3.1	Messaging Framework Adaptor Function (MFAF)
main	4.4.1.3.2	NF Service Consumers
main	4.4.2	Service Operations
main	4.4.2.1	Introduction
main	4.4.2.2	Nmfaf_ContextManagement_Transfer service operation
main	4.4.2.2.1	General
main	4.4.2.2.2	Transferring MFAF configurations
main	5.1	Nmfaf_3daDataManagement 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
main	5.1.3.2	Resource: MFAF Configurations
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 MFAF Configuration
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

Document	Section	Title
main	5.1.3.3.3.2	DELETE
main	5.1.4	Custom Operations without associated resources
main	5.1.5	Notifications
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: MfafConfiguration
main	5.1.6.2.3	Type: MessageConfiguration
main	5.1.6.2.4	Type: MfafNotiInfo
main	5.1.6.2.5	Type: MfafTransferInfo
main	5.1.6.2.6	Type: TargetMfafNotiInfo
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	5.1.6.4	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
main	5.3	Nmfaf_ContextManagement Service API
main	5.3.1	Introduction
main	5.3.2	Usage of HTTP
main	5.3.2.1	General
main	5.3.2.2	HTTP standard headers
main	5.3.2.2.1	General
main	5.3.2.2.2	Content type
main	5.3.2.3	HTTP custom headers
main	5.3.3	Resources
main	5.3.3.1	Overview
main	5.3.4	Custom Operations without associated resources
main	5.3.4.2	Operation: transfer
main	5.3.4.2.1	Description

Document	Section	Title
main	5.3.4.2.2	Operation Definition
main	5.3.5	Notifications
main	5.3.6	Data Model
main	5.3.6.1	General
main	5.3.6.2	Structured data types
main	5.3.6.2.1	Introduction
main	5.3.6.2.2	Type: ContextTransferReq
main	5.3.6.2.3	Type: ContextTransferResp
main	5.3.6.3	Simple data types and enumerations
main	5.3.6.3.1	Introduction
main	5.3.6.3.2	Simple data types
main	5.3.6.4	Data types describing alternative data types or combinations of data types
main	5.3.6.5	Binary data
main	5.3.7	Error Handling
main	5.3.7.1	General
main	5.3.7.2	Protocol Errors
main	5.3.7.3	Application Errors
main	5.3.8	Feature negotiation
main	5.3.9	Security

The purpose of the protocol, as the source step located it in the documents: pin down the bounded migration request and response that moves an existing MFAF configuration from the old MFAF to the target MFAF; a legitimate configurator retains the transaction reference, the source/target NF identities, and the callback endpoint mapping; model only the applicable conditions of the specification, excluding the full UE mobility flow.

The properties against the documents

Before writing any query, 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 do exist in the documents.

ID	Name	About	What the documents say	Where
A03	Accepted response matches a real transfer	The guarantee that the target MFAF (an NF service consumer, an MFAF, s0088/s0101) obtains when it accepts a 200 OK: the response comes from the source MFAF it addressed, corresponds to a real migration run at the source, and agrees on the source MFAF identity, the target MFAF identity, the refIds and the configuration content.	s0112 specifies that the source MFAF performs the migration and returns 200 OK with ContextTransferResp, and that the target MFAF establishes the configuration after receiving the response; s0335 specifies that the keys of configs must be taken from the refIds of the request. How authentication is achieved is given by the baseline: 33.501 5.9.2.1 requires mutual authentication for direct NF-to-NF communication, and 29.500 13.3.1 requires validation of the access token and scope. The source text itself does not specify authentication between MFAFs (s0362 only says OAuth2 may be used for authorization).	ts-33-501:2871, s0083, s0095, s0101, s0104, s0110, s0112, s0332, s0335
A08	Migration preserves reference and callback mapping	The consistency between the new Individual MFAF Configuration that the target MFAF establishes and its original configuration at the source: each newRefIds entry corresponds to the same refIds entry in the request, and notificationURI, correId, notifEndpoints and mfafNotiInfo (mfafNotifUri/mfafCorreId) are unchanged, with tgtMfafNotiInfos keyed by newRefIds in the response and, in the inner level, by correId.	s0234 specifies that newRefIds is placed in the response by the producer; s0237 specifies that the keys of tgtMfafNotiInfos are taken from refIds in the request and from newRefIds in the response, with correId as the inner key; s0231 defines mfafNotifUri/mfafCorreId; s0112 requires the target MFAF to establish a resource for each successfully migrated configuration.	s0110, s0112, s0231, s0234, s0237, s0332, s0335
D01	Requests and responses are authentic	Message authenticity for both parties: every ContextTransferReq the source MFAF accepts corresponds to a real send by the target MFAF, and every ContextTransferResp the target MFAF accepts corresponds to a real send by the source MFAF; an attacker on the external network cannot forge or tamper with them.	s0110/s0112 specify the request and the response; 33.501 5.9.2.1 requires mutual authentication of direct NF-to-NF service requests and responses and requires every NF to validate all received messages; TS 29.572 s0362 only says access may be authorized with OAuth2, and message integrity relies on the SBI security of 29.500.	ts-33-501:2871, ts-33-501:2875, s0110, s0112, s0302, s0309, s0311, s0362
R03	Transfer side effects happen at most once	For the same logical migration operation (same set of refIds, same source/target MFAF pair), the local deletion at the source (s0112) and the creation of the new resource at the target (s0112) are each committed at most once; the request may be retransmitted, and a retransmission must not lead to duplicate deletion or duplicate creation.	s0112 describes the two side effects on the success path (local deletion at the source, establishment of the resource at the target together with allocation of a new transaction reference id); the source text does not specify retries of the migration request, idempotency keys, or handling of duplicate responses.	s0112, s0173
S04	Configuration ownership unchanged by migration	Migration does not change the ownership of the configuration: after migration the configuration at the target is still bound to the NF service consumer (configurator) that created it before the migration and to the same notificationURI/correId, and the configuration lands on the target MFAF designated by the configurator, not with any third party able to name that resource.	s0035/s0048-s0053 specify that the configuration is created by the NF service consumer, that the MFAF allocates the transaction reference id and returns 201 Created; s0112 specifies that after migration the resource is established by the consumer that initiated the migration; s0234 specifies that newRefIds is returned only to the consumer that initiated the migration; s0105 states that the configurator (DCCF) triggers the migration through configuration in order to complete the MFAF relocation of the UE.	s0035, s0049, s0053, s0105, s0112, s0234, s0335

ID	Name	About	What the documents say	Where
Z01	Transfer is authorized by a valid token	Protection of the resources of the source MFAF: every migration that is executed has a valid access token issued by the NRF covering the nmfaf-contextmanagement scope; and a legitimate NF holding only that service-level scope cannot migrate configurations it is not entitled to (configurations created by other consumers).	s0362/s0363: access may be authorized with OAuth2, and the consumer must first obtain a token from the NRF; s0365: this API defines only a single service-level scope and no resource-level or operation-level scope. 29.500 13.3.1 requires the server to validate the token, its validity period and the scope, and states that with only a service-level scope, authorization can only cover "operations that do not require dedicated authorization". 33.501 5.9.2.1 also requires the NRF to be able to decide registration/discovery/authorization requests.	ts-29-500:4994, ts-29-500:4996, s0264, s0265, s0267, s0362, s0363, s0365
A03.7	Identity and refIds binding of response	The migration response accepted by the target MFAF agrees with its real run on the source/target MFAF identities and the refIds (the wording of Q1).	Same as A03: s0110/s0112/s0332/s0335; the source text does not specify the authentication mechanism between MFAFs (s0362).	s0110, s0112, s0332, s0335
A08.8	Reference and endpoint mapping consistency	The newRefIds entries accepted by the target MFAF correspond to the same refIds entries in the request, and notificationURI, correId, notifEndpoints and mfafNotiInfo agree with the original configuration at the source (the wording of Q2).	Same as A08: s0112, s0234, s0237, s0335.	s0110, s0112, s0234, s0237, s0335
U01	No half-committed success	The side effects at both ends agree on the full-success and partial-success/failure paths: there is no state in which the source has deleted and the target has not established, yet the operation is advanced as a success (the wording of Q3).	s0112 describes only the success path (deletion at the source + establishment at the target); s0113 only says that on error an HTTP error response is returned and refers to 5.3.7 (s0349 refers to the generic error codes of 29.500), and does not specify compensation for partial success or a lost response.	s0112, s0113, s0335, s0349
X01	Configuration content stays confidential in transit	The configuration content and buffered notifications transferred between the source MFAF and the target MFAF; protected against an attacker with network capabilities only.	33.501 5.9.2.1 requires the registration/discovery of the SBA to support confidentiality, integrity and replay protection, and requires mutual authentication for direct NF-to-NF communication (P215-P220); 33.501 13.1 specifies that the SBI within a PLMN uses TLS; bufferedNotifs in TS 29.572 s0335 and the notification information in s0231/s0234 are part of that response body.	ts-33-501:2871, ts-33-501:2865, s0335, s0231
X02	Configurator keeps its handle after migration	The reachability of the management of its own configuration by the NF service consumer that created it, before and after migration.	Update in s0055 and Deconfigure in s0071 are both addressed by the resource URI of the Individual MFAF Configuration; s0112 specifies that after migration the source deletes that resource and the target establishes the resource with a new transaction reference id; s0234 specifies that newRefIds appears only in the response to the consumer that initiated the migration.	s0055, s0071, s0112, s0234, s0105

Added from the documents by the analyst: X01, X02.

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: 30. Not taken: 368. 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	The security considerations overview and threat model reference of the OAuth framework do not state properties that must hold or can be attacked for this API.
P003 to P006	rfc6749:2790-2823	Z01	
P007 to P010	rfc6749:2827-2850	Z01, S04, `malicious_nf`	
P011 to P015	rfc6749:2859-2881	Z01, A03, `malicious_nf`	
P016 to P033	rfc6749:2885-3003	Outside the scope	refresh token, authorization code, redirection URI and resource owner password grants; this API uses only the client credentials grant (s0362) and has none of these flows.
P034 to P035	rfc6749:3013-3019	X01	
P036	rfc6749:3023-3028	A03, D01, X01	
P037 to P039	rfc6749:3032-3041	The prover cannot be asked	The probability bounds on token guessability (2^-128 etc.) are computational claims; the symbolic model treats primitives as perfect and cannot answer them.
P040 to P062	rfc6749:3045-3210	Outside the scope	phishing, CSRF, clickjacking, code injection, open redirector, and token misuse in the implicit grant; this API is a direct NF-to-NF client credentials flow (s0362, 33.501:2871) and does not use user-agent flows.
P063	ts-33-501:2491-2491	No goal stated	Bidding down of 5G security features (between UE and network) is irrelevant to this API: feature negotiation in this API concerns only optional API features (s0358) and does not negotiate security algorithms.
P064 to P072	ts-33-501:2495-2511	Outside the scope	Subscription authentication between UE and network, serving network authentication, and emergency services; not part of this API sub-flow.
P073 to P213	ts-33-501:2515-2853	Outside the scope	5G key hierarchy, RAN/NAS/AS confidentiality and integrity, subscriber privacy, UDM/AUSF/UDR requirements; all unrelated to this SBI sub-flow.
P214	ts-33-501:2859-2859	X01, A03	
P215 to P220	ts-33-501:2865-2875	A03, D01, Z01, X01, `malicious_nf`	
P221 to P227	ts-33-501:2879-2891	Z01	
P228 to P234	ts-33-501:2895-2907	Outside the scope	NEF security requirements; NEF is not a party to this flow.
P235 to P241	ts-33-501:2911-2923	Outside the scope	SCP (indirect communication) security requirements; this flow is direct NF-to-NF. The migration response in TS 29.572 may carry SCP 307/308 redirects (s0311-s0315), a deployment option outside this task's scope, listed under limits.
P242 to P303	ts-33-501:2929-3055	Outside the scope	SEPP and roaming, roaming intermediaries; the two MFAFs are in the same PLMN (s0086).
P304 to P312	ts-33-501:3059-3075	Outside the scope	SEPP attribute protection (roaming).
P313 to P314	ts-33-501:3079-3081	Outside the scope	IPUPS function, not modeled.

Paragraphs	Where	Taken up as	Why not
P315 to P318	ts-33-501:3085-3091	Outside the scope	Requirements that NFs fill in PLMN/SNPN IDs in the 3gpp-Sbi-Originating-Network-Id header; this model has no PLMN boundary and so yields no goal for this flow.
P319 to P337	ts-33-501:3097-3141	Outside the scope	Configuration and streaming of signalling monitoring data, security visibility/configurability; these are operational flows, not properties of this protocol's messages.
P338 to P369	ts-33-501:3149-3215	The prover cannot be asked	Security of ciphering and integrity algorithm identification and algorithm selection; the symbolic model treats primitives as perfect and does not answer.
P370 to P398	ts-33-501:3219-11901	Outside the scope	Requirements on other functions: 5G-RG, NSSAAF, EAP-TLS, IAB, MBSF/MBSTF, NEF.

The parties and the protected values, one by one

The host listed the 28 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: 7. Without: 21. 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	Source MFAF	party	Everything lost by definition	Controlling the source MFAF means controlling the party that holds the configuration and sends the response; the properties of this flow are all stated as its behaviour and are therefore lost by definition, and the documents state no requirement under the assumption that the source MFAF is compromised.
S02	Target MFAF	party	Everything lost by definition	The target MFAF is the party being assured; controlling it makes all properties of this flow lost by definition.
S03	External network attacker	party	Nothing gained	This party is the network attacker itself (read, inject, drop, replay); it has no additional capability that could be named as another attacker.
S04	Malicious legitimate NF holding its own credentials	party	`malicious_nf`	
S05	Configurer	party	Everything lost by definition	The configurer triggers migration to a target it designates under its own authorization (s0037, s0051), so "where it can migrate its configuration" is by definition unconstrained; the other parties (the target MFAF's assurance toward the source MFAF) are answered under network and are unaffected.
S06	NRF	party	Everything lost by definition	The NRF is the trust anchor for tokens (s0362, s0363); controlling it allows issuing tokens with any scope to any NF, so Z01 is lost by definition, and the documents state no requirement under the assumption that the NRF is compromised.
S07	refIds	value	`refid_known`	
S08	configs	value	Everything lost by definition	Holding the configuration content means knowing the migrated configuration; the confidentiality goal is lost by definition, and the other goals do not ask about its confidentiality.
S09	NF instance identity of the source/target MFAF	value	Nothing gained	NF instance identities are public and discoverable via the NRF (s0086, s0087) and confer no additional capability.
S10	refIds / newRefIds mapping	value	`refid_known`	

ID	Party or value	Kind	Attacker	Why none
S11	notificationURI	value	Nothing gained	notificationURI, correId, notifEndpoints and mfafNotiInfo identify the consumer side and are not authentication bases in this sub-flow; forging notifications with them belongs to the notification flow of 4.2, which the goals of this sub-flow do not concern.
S12	correId	value	Nothing gained	notificationURI, correId, notifEndpoints and mfafNotiInfo identify the consumer side and are not authentication bases in this sub-flow; forging notifications with them belongs to the notification flow of 4.2, which the goals of this sub-flow do not concern.
S13	notifEndpoints	value	Nothing gained	notificationURI, correId, notifEndpoints and mfafNotiInfo identify the consumer side and are not authentication bases in this sub-flow; forging notifications with them belongs to the notification flow of 4.2, which the goals of this sub-flow do not concern.
S14	mfafNotiInfo	value	Nothing gained	notificationURI, correId, notifEndpoints and mfafNotiInfo identify the consumer side and are not authentication bases in this sub-flow; forging notifications with them belongs to the notification flow of 4.2, which the goals of this sub-flow do not concern.
S15	tgtMfafNotiInfos	value	Nothing gained	The key values of tgtMfafNotiInfos are only a re-association of notification information, and this sub-flow has no goal that relies on it for authentication.
S16	refIds in ContextTransferReq	value	`refid_known`	
S17	configs and newRefIds in ContextTransferResp	value	`refid_known`	
S18	Individual MFAF Configuration resource on the source MFAF	value	Everything lost by definition	Holding the source/target Individual MFAF Configuration resource means controlling the storage of the MFAF that holds it (S01/S02), and the related properties are lost by definition.
S19	Individual MFAF Configuration resource newly created on the target MFAF, and the new transaction reference id	value	Everything lost by definition	Holding the source/target Individual MFAF Configuration resource means controlling the storage of the MFAF that holds it (S01/S02), and the related properties are lost by definition.
S20	Individual MFAF Configuration resource and its transaction reference id	value	Everything lost by definition	Holding the source/target Individual MFAF Configuration resource means controlling the storage of the MFAF that holds it (S01/S02), and the related properties are lost by definition.
S21	Configurer identity	value	Nothing gained	The configurer identity is a public NF instance identity (the 201 Created body of s0053) and confers no additional capability.
S22	notificationURI and correId	value	Nothing gained	Same as S11-S14: notificationURI and correId are not authentication bases for this sub-flow.
S23	OAuth2 access token and its scope	value	`malicious_nf`	
S24	Source/target MFAF identity	value	Nothing gained	The NF instance identities of the source/target MFAF are discoverable via the NRF and are public information.
S25	refIds/newRefIds	value	`refid_known`	
S26	Source-side Individual MFAF Configuration resource	value	Everything lost by definition	Same as S18: holding the source-side resource means controlling the source MFAF's storage.
S27	Target-side newly created resource	value	Everything lost by definition	Same as S19: holding the target-side resource means controlling the target MFAF's storage.

ID	Party or value	Kind	Attacker	Why none
S28	configs in ContextTransferResp	value	Everything lost by definition	Same as S08: holding the configs in the response means knowing the migrated configuration, and the confidentiality goal is lost by definition.

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`	The identity of one legitimate NF instance, its mTLS certificate and private key, and a valid access token issued to it by the NRF covering the nmfaf-contextmanagement service-level scope; it can call any operation of this API as an NF service consumer.	s0365 specifies that this API has a single service-level scope and defines no resource-level or operation-level scopes, so a legitimate NF holding that scope has the same authorization over this API's resources; 29.500 13.3.1 and 33.501 5.9.2.1 require only that the service-level scope be checked; RFC 6749 10.2 notes the malicious client problem, and 10.3 notes that the resource server cannot determine whether a token was issued to the client presenting it.	ts-29-500:4994, ts-29-500:4996, rfc6749:2827, s0365, s0362
`refid_known`	The pre-migration resource URI of another consumer's configuration (refIds, including the transaction reference id); apart from that, only network capabilities and its own legitimate identity.	Tests A08/S04/Z01 all list refIds as a protected asset; s0049/s0053 state that the MFAF allocates a transaction reference id when it establishes a configuration and returns 201 Created to the configurer, and s0273 states that the resource URI structure is defined by 29.501 4.4.1; the source text does not specify that this identifier is unguessable or cannot be enumerated through the management interface. This is a capability assumption, not a statement of the documents.	s0049, s0053, s0273, s0332

ID	`network`	`malicious_nf`	`refid_known`
A03	Holds	Holds	
A08	Holds	Holds	Holds
D01	Holds	Holds	
R03	Holds	Holds	
S04	Fails	Holds	Fails
Z01	Holds	Holds	Fails
A03.7	Holds	Holds	
A08.8	Holds	Holds	Holds
U01	Fails		
X01	Holds		
X02	Fails		

Source and targets

Source: 406 units, 160566 bytes, frozen in source-delivery-01.json. Targets: 7 approved (A03, A08, D01, R03, S04, Z01, U01); 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-0bef38e35135794d2771dc7b9c8b

This step as it reports itself: 2 requests (2 failed), 0.000 USD, 177564 tokens in and 9523 out, 224 s.

Notes on the prover and the model

- A08: answered in ProVerif, and the analyst names changing_state for this target
- R03: answered in ProVerif, and the analyst names changing_state for this target
- S04: answered in ProVerif, and the analyst names changing_state for this target
- A08.8: answered in ProVerif, and the analyst names changing_state for this target
- U01: answered in ProVerif, and the analyst names changing_state for this target
- X02: answered in ProVerif, and the analyst names changing_state for this target

A ProVerif model that simplifies the 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
The Individual MFAF Configuration of a consumer and its transaction reference id at the source MFAF (the pre-transfer resource URI, refIds)	the source MFAF (the MFAF currently holding the configuration, s0234)	When a ContextTransferReq naming it is accepted: the source MFAF answers 200 OK and locally removes the resource in the same step (s0112).	When a ContextTransferReq arrives, the requested refIds (ContextTransferReq.refIds, the resource URIs including the transaction reference ids, s0332) are compared with the resource URI it holds.	s0049, s0053, s0112, s0173, s0332
The new transaction reference id of the migrated configuration at the target MFAF (newRefIds, the post-transfer resource URI)	the target MFAF (the NF service consumer that invoked the transfer)	When the 200 OK is accepted: the target MFAF creates an Individual MFAF Configuration resource and is handed the new transaction reference id (newRefIds) by the response (s0112, s0234).	The response is accepted against the refId the target requested and the source identity it addressed; the refIds of the configs map must be one of the requested refIds (s0335).	s0112, s0234, s0335

value	kept by	changes when	checked when	refs
The resource URI the configurator holds for the configuration it created (the handle it uses to Update or Deconfigure it)	the configurator (the NF service consumer that created the configuration, s0035)	Only when the configurator itself changes it; the transfer procedure gives it no new value.	Whenever the configurator sends Update or Deconfigure, which are addressed by the Individual MFAF Configuration resource URI it holds (s0055, s0071).	s0035, s0055, s0071, s0112, s0234

pattern	result	target	shown by ProVerif	steps
replay	nothing found			In bounded.pv a second transfer request for the same configuration is refused (ReqRefused, Src_gone), and r03 is verified: no second removal. A replayed response makes the target create the same resource again under the same newRefId, which names one resource; the target side is not unrolled to two responses, so this is not confirmed by a query.
repeat_one_time_action	nothing found			The removal can happen once: the source forgets the configuration (Src_have goes to Src_gone) and r03 is verified; with the buggy variant that keeps it, r03_ctl is falsified and the same configuration is removed twice.
desynchronisation	candidate	X02	yes: ProVerif falsifies `x02` on bounded.pv	1. The configurator created the configuration at the source MFAF and holds its resource URI (refIds), s0035, s0049, s0053. / 2. The target MFAF invokes the transfer and the source answers 200 OK with newRefIds and removes its configuration, s0110, s0112. / 3. The target creates the configuration under the new transaction reference id, s0112. / 4. The source text gives newRefIds only to the migrating consumer (s0234) and defines no operation by which the configurator learns the new URI. / 5. The configurator's later Update or Deconfigure addressed to the URI it holds (s0055, s0071) finds no configuration at the source; the parties are left with different values and no way back within the transfer procedure.
update_before_check	nothing found			Src_have performs every check (token signature and scope, consumer signature, refIds match) before it answers and before ConfRemoved; no message changes the kept value before it is verified.
no_rollback_on_failure	candidate	U01	yes: ProVerif falsifies `u01` on bounded.pv	1. The target MFAF sends the transfer request (s0110). / 2. The source MFAF accepts it, answers 200 OK with configs and newRefIds, and removes its configuration in the same step (s0112). / 3. An attacker on the network drops the response. / 4. The target creates nothing; the source text defines no retry, no timer and no compensation (s0113 covers only errors raised while processing the request). / 5. The configuration is now held by neither MFAF, although the source MFAF treated the transfer as done.
race_between_sessions	nothing found			In bounded.pv one configuration is requested by the migrating target and, on the attacker's behalf, by a legitimate NF; the unrolled source serves the first and refuses the second, so no interleaving of two requesters produces two removals (r03 verified). A run in which the configuration is re-created between the two requests is outside the three steps unrolled.
distinguishable_failure	candidate		no: not tried with ProVerif	1. A legitimate NF holding the service scope asks for a guessed resource URI. / 2. The source MFAF answers 200 OK with configs if the configuration exists (s0112) and an error response otherwise (s0113). / 3. The two answers differ, so the requester learns whether the guess named a configuration and can enumerate the name space the migration relies on for protection.

pattern	result	target	shown by ProVerif	steps
wrap_or_window	nothing found			No counter, sequence number or acceptance window exists in the transfer; the transaction reference id is a resource name, not a monotonically increasing value.

What the analyst named for targets:

step	targets	features	reason
39	R03, S04, X02, U01, A08, A08.8	changing_state	These targets rest on the transaction reference id that the migration replaces (refIds to newRefIds, s0112, s0234) and on the source's removal of the configuration, a value that changes across the steps of one run. ProVerif answers them on the unrolled source in bounded.pv, not for any number of sessions.
39	A03, A03.7, D01, Z01, X01	plain_primitives	The model uses only hashing-free primitives: asymmetric encryption and signatures for the per-endpoint TLS of the SBI, plus the OAuth2 token as a signature by the NRF. The Diffie-Hellman key agreement assumed for TLS is idealised as a plain key agreement, and no algebraic law of a primitive is relied on by any answer.

The analyst's closing note

Eleven targets answered, each with a control. Proved: the target MFAF's accepted response matches a real source-side run on identities, refIds and content (A03, A03.7, D01), with controls on targets and sources that skip the signature and token checks; the migrated configuration keeps its notificationURI/correId binding and the refIds to newRefIds correspondence (A08, A08.8); transport confidentiality of the configuration and the refIds against a network attacker (X01); and that every transfer is backed by a token with the service scope (Z01, first claim). Verified over two unrolled steps: one removal per configuration however often the request is replayed (R03, control: a source that keeps the configuration removes it twice).

Attacked: a legitimate NF holding only the service scope that knows a pre-transfer resource URI is handed another consumer's configuration and the source deletes its copy, because the source checks the token and the scope but never whether the requester may have that configuration (Z01, S04; recommendation checked on a copy of the model, where the attack disappears, the normal run stays and no other answer changes); a dropped 200 OK leaves the configuration deleted at the source and never created at the target (U01); and after any migration the configurator's own URI no longer resolves, so it can neither update nor deconfigure the configuration it created (X02, S04). S04's two attacks are the same runs as Z01's and X02's and are drawn under them.

What blocks more: the proofs rest on idealised endpoint TLS and on the transaction reference id being unguessable — the attacker that knows it is the separate assumption refid_known; refusals are modelled as parties stopping rather than as HTTP codes, so the enumeration oracle seen in the state review stays a candidate and is not proved; the 307/308 redirect through an SCP and indirect communication are outside the scope; one configuration per transfer is modelled, so partial success inside a multi-entry response is not answered; and the state questions are answered only for the two steps bounded.pv unrolls. Two of the three recommendations change the exchange rather than a check and were therefore not checked.