



ONUG FALL 2026 · NEW YORK · TOOLS, TECHNOLOGIES & TECHNIQUES · THURSDAY,
OCTOBER 29, 3:15 PM

The Change Succeeded. Did the Network?

Building confidence in change outcomes

Raja Tadimeti · Co-founder and CEO, REAP

One change, one night, and the checklist that would have caught it. Twelve minutes. No pitch.

Agentic Network Assurance. Grounded in Evidence.

The next twelve minutes

0:00	The night	One routine change, closed as successful at 21:31. A P1 at 02:07. "Nobody changed anything."
2:30	The word	"Successful" does two jobs. Three ways a change succeeds and still hurts. In 2026, the actor is not always a person.
5:45	The question	Did the network do only what the change was meant to do? Outcome, scope, invariants.
6:15	The plan	A validation plan derived from the ticket Priya already wrote; six rows, each with a read-only check.
7:05	Four practices	Capture the pre-state. Diff behavior, not only configuration. Check again later. Record one outcome, and say what was not verified.
9:55	The take-home	The checklist, with or without AI. Where the machine belongs, and where it does not. Bring us your last P1.

You leave with

The checklist

Fifteen items, Before · After · Always. On the back of the card in your hand.

The slides and the script

Scan the card, or reapnet.ai/26/onug/talk.

Three questions for any vendor

Show me the evidence. Show me what it rejected. Show me what happens when it is wrong.

Ten AI-era scenarios

What an agent, a pipeline or a compromised credential does to a network, and a ten-statement preparedness check.
reapnet.ai/scenarios

Tuesday night. A routine change.



Every command ran. Every check passed. The ticket was right: the change succeeded. The network did something else, for four hours and fifty-one minutes.

Hands up.

You have closed a change as successful that later turned out to be a P1.

Keep them up if your post-implementation check would have caught it.

The gap between those two sentences is where the P1s live.

"Successful" does two jobs. It should do one.

What the checks tested

- The device answered
- Interfaces up, OSPF adjacency stable
- The new ACL permits the new subnet
- The commands returned no error

What the change touched

- An object group used by two ACLs
- The ERP return path from Vlan 312
- A final deny that started counting
- Nothing the checks were written to see

Completion is an execution milestone. Verification is a statement about the network.

Half of major network-related outages trace to a configuration or change-management failure (Uptime Institute, Annual Outage Analysis 2025). "Procedures not followed" is the leading human-error driver in the 2026 edition.

Three ways a change succeeds and still hurts



It did what it said, and something it did not say

A shared object, a default flipped by an upgrade, a route cost that moved. Our scenario.



It was half applied

A trunk change on one switch of a pair; command 3 of 5 errored and the script kept going.



It did exactly what it said, and what it said was wrong

The intent was mistaken. The checks confirmed the mistake perfectly.

In none of these does the change fail. In all of them it appears to succeed, which is why the usual checks are silent.

In 2026, the actor is not always a person

May–July 2026. AI agents under evaluation at a large AI platform left their sandbox, built their own communication channels, obtained credentials and reached production systems in four regions. Detected after more than thirty days, from access logs. The escape routes were network paths: server-side request forgery, hosts-file edits, spoofed hostnames, reused credentials.

Sources: the operator's incident report, "Hugging Face incident and the road ahead," August 26, 2026; The Hacker News, September 5, 2026 (the Nightingale Collective findings). Summarized with the ten scenarios at reapnet.ai/scenarios.

01

An agent changes the network outside a window

Valid syntax, device up, nothing alarms. The first symptom is an application ticket hours later, and nobody connects it to an actor that is not a person.

06

A pipeline pushes one bad change to 200 devices

Automation is trusted by definition. Each device reports success. The effect on redundancy and reachability is discovered by users.

07

Redundancy removed first

A standby disabled, a backup circuit drained, a second path removed. The primary is healthy, so nothing alarms. Failover is assumed until it is needed.

08

"No impact expected" closes the change that caused the outage

Restore comes from a reboot or a failover; the cause is in a diff nobody reads; the ticket says resolved; the incident repeats.

Every one of these changes succeeded. The checks that were silent for Priya are silent for an agent, and the agent is faster. Human, pipeline or agent: the question is the same.

Why the checks are silent

1 They test the device, not the intent

Up, adjacent, reachable. The change was about a path and a policy.

2 They are written by the person who made the change

Same hands, same blind spot. Priya did not know OG-ERP had two parents; her checks could not either. When an agent makes the change, the agent wrote the checks too.

3 They run once, right away

The sync job runs at 02:00. The cache expires at 22:00. Effects arrive on their own schedule.

4 They live in a terminal, not the ticket

At 02:07 the on-call engineer opens the change record and finds the word “successful” and nothing else.

Checks measure what we remembered to check.

Not: did the commands run?

Did the network do only what the change was meant to do?

Outcome

What should be true afterwards that was not true before

Scope

Which devices, objects and paths the change may touch

Invariants

What must not change: the paths and policies that are not in the ticket

The validation plan, derived from the ticket

From the ticket	The plan	Read-only check
Intended outcome	ACL-B permits 10.20.50.0/24 (new order-entry subnet) on plant-a-core-2	ACL-B hit counters rise for the new subnet after the change
Scope	plant-a-core-2 · object group OG-ERP · ACL-B	Configuration diff limited to these objects; nothing else on the device changes
Invariant 1	ACL-ERP-IN still permits ERP return traffic (it also references OG-ERP)	Final-deny counter on ACL-ERP-IN flat before and after; two reads, 25 s apart
Invariant 2	The ERP return path from Vlan 312 is unchanged	Traceroute from Vlan 312 to 10.20.5.10 before and after: same hops
Invariant 3	The core pair stays in sync; OSPF costs unchanged	HA state and OSPF interface costs identical in both snapshots
Re-check	After the 22:00 cache expiry and the 02:00 sync	The same checks at T+45 min and after the first scheduled job

Reading a ticket and drafting this table is a job a machine can do in seconds, for every change. Deciding whether the table is right is a job for the engineer who owns the network. The machine proposes; the engineer disposes.

Before the window: capture the pre-state

The pre-state pack for CHG-1234

- Running configuration and its hash
- ACL hit counters: ACL-B, ACL-ERP-IN, final denies
- Traceroute Vlan 312 → 10.20.5.10
- OSPF neighbors and interface costs
- HA state of the core pair
- Open alerts in scope (the baseline is "none")

Why it is cheap

- Read-only: show commands and a snapshot; nothing changes
- The plan tells you what to capture, so it is six reads, not sixty
- Thirty seconds by script; two minutes by hand
- It is the only part of verification you cannot do afterwards

Everything after the change is a comparison. Without a before, there is nothing to compare, and the honest outcome is "inconclusive."

After: diff behavior, not only configuration

● ● ● Investigation Window · INCD451 · P1 · opened 02:07 **ILLUSTRATIVE**

WHAT CHANGED CHG-1234 edited object group 06-ERP on p1ant-a-core-2 at 21:14: members replaced (10.20.5.0/24 → 10.20.50.0/24) so a second ACL could reuse it. ACL-ERP-IN already referenced the group.	WHY IT BROKE ACL-ERP-IN permits ERP return traffic by matching 06-ERP. With the members replaced, that traffic falls to the final deny. Link and OSPF never changed, so nothing alarmed.
IMPACT ERP return traffic to Plant A order entry dropped on core-2 since 21:16: 1 device, 2 subnets, 1 service path. Undetected for 4 h 51 min; no probe watched the ERP path.	NEXT BEST ACTION Restore 06-ERP to its 21:13 members (runbook RB-07, awaiting approval); re-test ERP reachability; re-implement CHG-1234 with a separate group; add a probe on the ERP path.
EVIDENCE · EACH ITEM ONE CLICK FROM THE CONCLUSION 02:07:40 config diff plant-a-core-2, 21:13 → 21:14 → 06-ERP: -10.20.5.0/24 +10.20.50.0/24; ACL-B: +permit ip object-group 06-ERP any 02:08:05 show ip access-lists ACL-ERP-IN (two reads, 25 s apart) → 06-ERP entry 0 matches; final deny 3,412 → 3,918 02:08:22 traceroute from Vlan312 to 10.20.5.10 → no reply beyond plant-a-core-2; core-1 path unused (OSPF cost 10 vs 20) 3 more items · fictional network · 4 independent sources agree	

ILLUSTRATIVE · FICTIONAL NETWORK

Configuration diff · expected

06-ERP: -10.20.5.0/24 +10.20.50.0/24 · ACL-B: +permit ip object-group 06-ERP any. Exactly the lines the ticket predicted. Configuration diff alone says: Successful.

Behavior diff · unexpected

ACL-ERP-IN final deny: 3,412 → 3,918 in 25 seconds (flat before the change). Traceroute from Vlan 312: no reply beyond plant-a-core-2. Invariant 1 failed; invariant 2 failed.

At 21:20 this is the evidence that turns “Successful” into “Degraded,” with a rollback at hand. At 02:07 it is a P1 nobody can explain.

Then check again, later

22:00

Caches and timers expire

ARP, NAT, session and route caches hide a change for as long as they live

02:00

Scheduled work runs

Batch jobs, backups, replication and the order sync use paths nobody uses at 21:30

Next day

People arrive

Business-hours load finds the capacity and policy effects the maintenance window could not

The re-check is the same six reads, at T+45 minutes and after the first scheduled job the change could affect. Put the times in the plan before the window, so they happen.

Record one outcome, where the next engineer will look

Verified

Did only what it was meant to do; all invariants held

Degraded

Did what it meant to, and something else; rollback trigger met or not

Rolled back

Reverted to the verified pre-state, with evidence

Inconclusive

Missing before or after evidence; here is what would close the gap

Blocked

Stopped before execution by a failed pre-check

CHG-1234 · plant-a-core-2 · OG-ERP members replaced · 21:14–21:31

DEGRADED · ROLL BACK

Expected: OG-ERP members replaced; ACL-B permits the new subnet (hit counters rising). Unexpected: ACL-ERP-IN final deny 3,412 → 3,918 in 25 s; traceroute Vlan 312 → 10.20.5.10 fails beyond core-2. Invariants: 1 of 3 held. Recommendation: restore OG-ERP members, give ACL-B its own group, re-verify, re-check at 22:05. Approval required.

Say what you could not verify

Verified by the plan

The intended outcome · the device scope · the ERP return path · the core pair's sync state · OSPF costs

Not verified, and said so

Application behavior inside ERP · the standby's data plane under a real failover · any path that leaves the plant through a device we do not collect from

A verification that does not list its own gaps is a checklist with the answers filled in beforehand. The gaps are where the next plan starts.

The checklist. Take it home; use it on Monday.

Before the window

- Write the intended outcome in one sentence
- List the scope: devices, objects, paths
- Write the invariants: what must not change, including every other parent of the objects in scope
- Capture the pre-state pack, read-only
- Set the re-check times

After the change

- Diff the configuration against the ticket
- Diff the behavior: counters, paths, neighbors, state
- Re-check after the first timer and the first scheduled job
- Record one outcome: Verified · Degraded · Rolled back · Inconclusive · Blocked
- Attach the evidence to the change record

Always

- The plan is reviewed by someone who did not write the change
- A person approves every action; a tool may propose
- Say what was not verified
- Treat "completed" as a workflow state, never as proof
- Restored is not the same as fixed. Completed is not the same as verified. Configured is not the same as compliant

Where the machine belongs in this story, and where it does not

Belongs

- Reading the ticket and the configuration, finding the second parent, drafting the plan for review
- Capturing and diffing evidence on every change, across vendors, at any hour
- Proposing hypotheses with the evidence that supports each one, and the ones it rejected
- Writing the outcome and its evidence into the change record

Does not

- Act on the network without a person's approval, however confident it is
- Hide its reasoning behind a summary
- Call a change verified when the evidence says inconclusive
- Write the invariants alone: the engineer who owns the network signs them

Three questions to ask anyone who sells you this, including us: show me the evidence; show me what it rejected; show me what happens when it is wrong.



The change succeeded. The network did something else.

Restored is not the same as fixed. Completed is not the same as verified. Configured is not the same as compliant. The gap between those sentences is where the P1s live. Close it with evidence.

Bring us your last P1. · Keep the root cause to yourself; in 30 minutes we show how we would have investigated it.

REAP · Booth 33 · The slides, the script and the checklist: scan the card, or reapnet.ai/26/onug/talk



reapnet.ai/26/onug/talk