Kibana 9.4.0: Why Basic Authentication via Proxy Suddenly Returns 401 Unauthorized

Daniel Gleim · 2026-10-29



Kibana 9.4.0: Why Basic Authentication via Proxy Suddenly Returns 401 Unauthorized

After upgrading from Kibana 9.3.0 to 9.4.0, you might run into a frustrating situation: basic authentication that has worked flawlessly for years suddenly stops working. Your login page still works fine, your API endpoints respond correctly, but accessing Kibana app paths through an Apache reverse proxy returns a `401 Unauthorized` error with no obvious reason why.

This post walks through exactly that scenario, explains what changed in Kibana 9.4.0, and shows you how to fix it.

Problem Description

Imagine you have Apache in front of Kibana, injecting an `Authorization` header to handle authentication for your users:

```
RequestHeader set Authorization "Basic <base64-encoded-credentials>"
ProxyPass https://127.0.0.1:8100/
ProxyPassReverse https://127.0.0.1:8100/
```

This setup has worked since Elasticsearch and Kibana version 8.x. After upgrading to 9.4.0, however, any request to Kibana app paths (e.g., `/app/discover`) returns a `401 Unauthorized`. Kibana's logs show this error:

```
ERROR  plugins.security.user-profile
Failed to activate user profile: {"error":{"root_cause":[{"type":"security_exception",
"reason":"unable to authenticate user [myuser_account] for action
[cluster:admin/xpack/security/profile/activate]"}]}}
```

What makes it particularly confusing is that:

- Logging into Kibana through the login page works fine.
- Calling an authenticated Kibana API endpoint like `/api/spaces/space` with the same `Authorization` header returns `HTTP 200`. (Note that `/api/status` is not a useful test here, it requires no authentication at all, so a `200` from it tells you nothing about whether your credentials are accepted.)
- Direct `curl` requests to Elasticsearch with the same credentials work without issues.

Only accessing Kibana UI app paths through the `Authorization` header consistently fails.

Root Cause Analysis

After methodical debugging (stripping the setup down to a minimal fresh install and comparing 9.3.0 versus 9.4.0 side by side), one configuration detail that reproducibly correlates with the failure in the environments tested so far is a colon (`:`) in the user's password.

What appears to have changed between 9.3.0 and 9.4.0 is how Kibana handles requests that carry an `Authorization` header instead of a session cookie. Starting in 9.4.0, the `user-profile` plugin activates a user profile for such requests by calling the Elasticsearch profile activation endpoint (`cluster:admin/xpack/security/profile/activate`) using a password grant. App paths like `/app/discover` appear to go through this new code path, while some API paths do not, which is why the two behave differently. In the environments tested so far, when a password contains a colon, this profile-activation step consistently fails:

```
DEBUG  plugins.security.user-profile  Activating user profile via password grant.
ERROR  plugins.security.user-profile  Failed to activate user profile: {"error":{"root_cause":
[{"type":"security_exception","reason":"unable to authenticate user [myuser_account] for action
[cluster:admin/xpack/security/profile/activate]"}]}}
ERROR  http  401 Unauthorized
```

This looks like a regression. Per RFC 7617 (the "Basic" HTTP Authentication Scheme), only the first colon in the decoded string is the delimiter between username and password, colons in the password itself are explicitly permitted. The same Base64 token that returned `200` on 9.3.0 returns `401` on 9.4.0, and the only differing ingredient in the affected setups is a colon in the password. As of this writing, this has not been officially confirmed as a bug by Elastic, so treat it as the best available working theory, but it has been independently reproduced and is worth checking first.

Note: There is a separate, confirmed bug in 9.4.0 for setups where `xpack.security.enabled: false` is set in Elasticsearch. In that configuration, Kibana attempts to call `/_security/profile/_activate`, which does not exist when security is disabled, producing a `500 Server Error` on all `/app/*` paths regardless of the password used. This is tracked as elastic/kibana#268959. If you are running with security disabled, the workaround below will not help and you will need to either enable security or wait for an official fix from Elastic.

© 2026 floragunn GmbH · search-guard.com

Solution

If you are running with security enabled and hitting the `401` above, the practical workaround is straightforward: change the password of the affected user so it no longer contains a colon (`:`).

Step 1: Identify which user is affected

The user whose credentials are Base64-encoded in your Apache `RequestHeader` directive is the one that needs a new password:

```
RequestHeader set Authorization "Basic <base64-encoded-credentials>"
```

To decode the current credentials and check whether the password contains a `:`, run:

```
echo "<base64-encoded-credentials>" | base64 --decode
```

The output will be in the format `username:password`. If you see more than one `:`, the extra ones are part of the password.

Step 2: Generate a new password

Create a new password that does not contain a colon, then update the user in Elasticsearch:

```
curl -k -u elastic:<admin-password> -X POST \
  "https://<elasticsearch-host>:9200/_security/user/<username>/_password" \
  -H "Content-Type: application/json" \
  -d '{"password": "<new-password-without-colon>"}'
```

This uses the standard Change Password API.

Step 3: Re-encode the credentials and update Apache

Generate a new Base64-encoded string for the updated credentials:

```
echo -n 'username:new-password' | base64
```

Note the `-n` flag: it prevents `echo` from appending a newline character, which would otherwise be included in the Base64 encoding and produce an invalid token.

Update your Apache configuration with the new value:

```
RequestHeader set Authorization "Basic <new-base64-encoded-credentials>"
```

Then reload Apache:

```
apachectl graceful
```

Step 4: Verify

Test that Kibana app paths now respond correctly:

© 2026 floragunn GmbH · search-guard.com

```
curl -vk -H "Authorization: Basic <new-base64-encoded-credentials>" \
  "https://<kibana-host>:<port>/app/discover"
```

You should receive an `HTTP 200 OK` response with the Kibana HTML content.

Takeaways

- **A colon in a user's password has been observed to break Basic Authentication to Kibana app paths in version 9.4.0.** This looks like a regression, the same setup worked on 9.3.0, but it is a reproduced community finding rather than an officially confirmed bug. If you hit sudden `401` errors after upgrading, checking the password for colons is a quick and worthwhile first step.
- **The `-n` flag matters when generating Base64 credentials.** Running `echo 'user:pass' | base64` without `-n` encodes a trailing newline, producing an invalid token. Always use `echo -n 'user:pass' | base64`.
- **Kibana API endpoints and app paths can behave differently.** A path like `/api/spaces/space` may return `200 OK` even when `/app/discover` returns `401`. Remember that `/api/status` requires no authentication at all and is not a valid test. Always test the specific paths your use case relies on.
- **There is a separate, confirmed bug for setups with security disabled.** If `xpack.security.enabled: false` is set in Elasticsearch, Kibana 9.4.0 will call a security endpoint that does not exist in that mode, causing `500 Server Error` on all `/app/*` paths (tracked as elastic/kibana#268959). This is a distinct issue from the colon-in-password behavior.

Security is only as strong as the details — and details like password character handling can quietly break things across upgrades. If you want a security layer for your Elasticsearch and Kibana stack that gives you fine-grained control without the surprises, take a look at what Search Guard has to offer.

This article was published on the Search Guard blog: https://search-guard.com/blog/kibana-9-4-0-why-basic-authentication-via-proxy-suddenly-returns-401-unauthorized/
Search Guard is Security and Alerting for Elasticsearch and Kibana — search-guard.com · Start a free trial