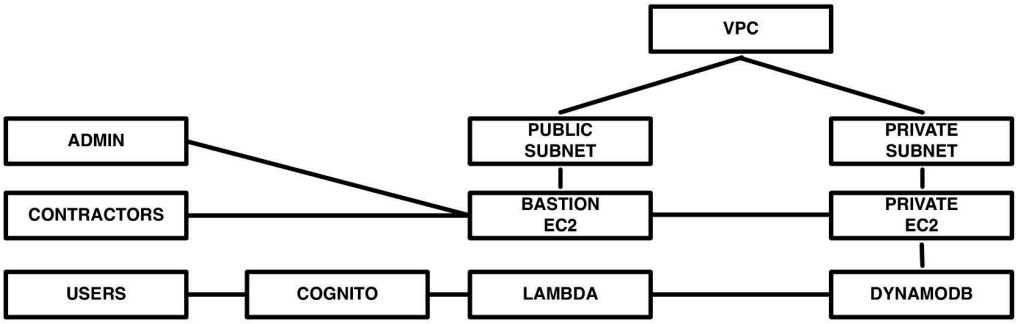
Topography


VPC
ADMIN
PUBLIC SUBNET
PRIVATE SUBNET
CONTRACTORS
BASTION EC2
PRIVATE EC2
USERS
COGNITO
LAMBDA
DYNAMODB

Abstract

This report analyzes how a modern enterprise authentication feature that follows current best practices can still be compromised when a legacy bastion host is present inside the same environment. The application tier uses Amazon Cognito Hosted UI with Authorization Code plus PKCE, a private S3 origin behind CloudFront with Origin Access Control, and AWS WAF at the edge. Despite this, the organization retained a shared bastion that allowed SSH agent forwarding for administrative workflows. In the controlled lab, a lower privileged user on the bastion discovered and interacted with an administrator's forwarded SSH agent, then used it to authenticate to private hosts as the administrator without ever possessing or exfiltrating private key material. The attack chain maps to multiple MITRE ATT&CK techniques, including Remote Service Session Hijacking, SSH, and Valid Accounts. This report describes the environment, threat model, high-level methods, findings, impact, detection opportunities, and prioritized mitigations. It concludes that strong web identity and edge security are necessary but are not sufficient when older trust patterns, such as permissive agent forwarding on shared bastions, remain in place. The remediation plan centers on disabling agent forwarding, replacing bastions with identity-bound access such as AWS Systems Manager Session Manager, shortening credential lifetime through SSH certificates or hardware bound keys, and improving host level telemetry that can reveal agent misuse.

Keywords

SSH agent hijacking, bastion host, Amazon Cognito, PKCE, CloudFront OAC, AWS WAF, MITRE ATT&CK, T1563.001, T1021.004, T1078, identity aware access

Introduction and Scope

Enterprises increasingly adopt cloud native patterns for public facing authentication because these patterns reduce the number of places where credentials, tokens, and user identity flows can be mishandled. In the lab environment for this report, the application tier leverages Amazon Cognito Hosted UI for sign in and sign out, Authorization Code plus PKCE to protect the authorization code exchange, a private S3 website origin accessible only through CloudFront with Origin Access Control, and WAF protections at the edge. The front end uses minimal static pages including a landing page, a privacy notice, an OAuth callback handler, and a simple profile page that displays only the user email after a successful login. Tokens are not stored persistently

in the browser and the design explicitly keeps client secrets out of the front end.

The modern application surface was intentionally designed to be resilient, but the broader environment included a legacy operational pattern in the form of a shared bastion host. The bastion offered shell access to administrators who routinely used SSH agent forwarding to reach private instances. The lab tested a common failure mode in mixed maturity environments. The question was whether a lower privileged user on the same bastion could observe or leverage a forwarded administrator SSH agent and then laterally move to private hosts with administrator authority. The goal was not to bypass the web authentication feature, but to evaluate whether older trust patterns could make the strong web identity posture irrelevant.

This report focuses on enterprise architecture lessons. The screenshots are redacted and illustrate mechanics only at a high-level; the intent is not to teach exploitation, but to surface an oversight: assuming a limited privilege SSH keypair is harmless. Once tolerable on-premises, in cloud computing it is dangerous. We present high-level methods, outcomes, and MITRE ATT&CK mapping, close with mitigations per NIST CSF and SOC 2.

Environment and Architecture

Application tier: The web facing component uses Amazon Cognito Hosted UI so that credentials are never entered on the application's origin domain. The client follows Authorization Code plus PKCE. This removes the need for a browser embedded client secret and mitigates authorization code interception. Static assets are hosted in a private S3 bucket and delivered only through CloudFront using Origin Access Control. AWS WAF is attached at the CloudFront distribution scope to block common web threats and to add a modest rate limit. Response headers at the edge are strict, including HSTS and a conservative Content Security Policy suitable for a static site with an OAuth redirect. The minimal front end includes the root page, a privacy page, an OAuth callback page, and a profile page that displays only the user email claim after sign in. Logout returns the user to the landing page.

Operational tier: The organization had a legacy bastion host to reach private subnets. For the lab, this included a VPC with a public subnet and a private subnet, an EC2 bastion instance in the public subnet, and one or more private EC2 instances reachable only from the bastion. A DynamoDB gateway endpoint was present in the VPC to emulate private data plane access patterns that are common for automation and administrative tooling. Administrators used SSH from their workstations to the bastion and frequently enabled SSH agent forwarding for

convenience, then from the bastion they reached private hosts with their forwarded agent. The lower privileged user had a separate account on the same bastion for troubleshooting and monitoring tasks.

Security posture assumptions: The application tier is correctly configured and not misusing tokens or secrets. The bastion is shared, and at least one administrator is expected to connect with agent forwarding enabled. File system and process namespace controls on the bastion are not hardened against cross user observation of agent sockets. The private EC2 instances trust the administrator public key or SSH certificate for direct shell access and do not require jump host identity binding beyond ordinary SSH trust.

Threat Model

Actors: The primary victim is an administrator who uses SSH agent forwarding through the bastion. The simulated adversary is a lower privileged user with interactive access to the same bastion. The adversary does not have administrator credentials, does not possess private key material, and does not have console level permissions that would grant them direct access to private instances.

Assumptions: The administrator connects to the bastion with SSH agent forwarding enabled. The forwarded agent exposes a Unix domain socket on the bastion that is addressable from the administrator's session context. Due to weak segregation on the shared bastion, the lower privileged user can discover and interact with that agent socket or a proxied interface to it. Private targets accept the administrator's SSH key or short-lived certificate for access. The lower privileged user understands enough about SSH agent semantics to request signatures through the agent without extracting any secrets.

Adversary Objective and Hypothesis: The objective is to achieve downstream shell access on private hosts with administrator authority while never possessing the administrator's private key or stealing credentials. The hypothesis is that if the lower privileged user can send requests to the forwarded agent, the agent will perform cryptographic operations on their behalf, thus allowing them to authenticate to private hosts as the administrator. If true, the strong user authentication at the web tier becomes irrelevant to the integrity of administrative operations in the private estate.

Methods

This lab used a narrative runbook rather than exploit scripting. The steps are listed at a conceptual level.

Step 1 - baseline the web application: Verify Cognito Hosted UI flow with Authorization Code plus PKCE. Confirm that the profile page displays only the user email claim and that WAF and edge security headers are present. Confirm that tokens are not persistently stored client side.

Step 2 - set up two separate bastion users: One user represents the administrator who routinely uses SSH agent forwarding from their workstation to the bastion and onward to private hosts. The other user represents the lower privileged operator with basic shell access for troubleshooting.

Step 3 - simulate routine administrator operations: The admin logs in to the bastion with agent forwarding enabled and proceeds to connect to one or more private EC2 instances. The admin session remains active long enough to reflect realistic workflows.

Step 4 - perform lower privileged observation: The lower privileged user explores the bastion context to determine whether an SSH agent is present and reachable. The user confirms that a cryptographic signing request can be sent to the agent and that the agent responds, even though the user does not possess the private key.

Step 5 - attempt lateral movement: Using only the agent interface, the lower privileged user initiates SSH connections to a private EC2 instance that is configured to accept the admin key. If the agent responds with a valid signature, the private host treats the connection as an administrator session.

Step 6 - record outcomes: The lab notes the success or failure of the lateral step, any host logs that reflect which key or principal was used, and any bastion indicators that an agent was accessed by an unexpected process or user.

The methods align with widely recognized adversary behaviors. The attack chain uses the administrator agent as a live broker for authentication requests, which is a form of Remote Service Session Hijacking focused on SSH. The transport for lateral movement is SSH, which maps to Remote Services. The acceptance of a valid administrative identity on the target hosts reflects that the attack leverages valid accounts, even though credentials were never stolen in the traditional sense.

Results

Agent discovery and reachability: The lower privileged user confirmed that the bastion exposed a reachable agent for the administrator session. While the lab did not attempt to read private key files, the user was able to send a signing request to the agent and receive a valid response. This is the critical finding. The agent supplies cryptographic operations on demand without divulging raw secrets, which is exactly what makes agent hijacking attractive to adversaries.

Lateral movement to private hosts: With agent access confirmed, the lower privileged user initiated SSH connections from the bastion to a private EC2 instance that trusts the administrator. The target host accepted the connection based on the agent supplied authentication. The shell session on the target reflected administrator level access without the lower privileged user ever possessing the admin private key or feeding an admin password.

Collateral access potential: Because the VPC included a DynamoDB gateway endpoint, the lab noted that administrative privilege on the private instance could be a stepping stone to misuse private data paths or automation channels. The lab did not perform any data plane actions, but it is reasonable to assume that misuse could go beyond shell access if the target instance held credentials for automation or had roles attached with broader capabilities.

No effect on the web authentication surface: All web authentication flows continued to function correctly during the lab. The OAuth redirect, token handling in the browser, edge security headers, and WAF behavior were unchanged. The attack did not target the web tier. Instead, it bypassed it entirely by abusing the legacy operational pattern of a shared bastion with agent forwarding.

MITRE ATT&CK Mapping

T1563.001 Remote Service Session Hijacking, SSH: The crux of the scenario is the misuse of a forwarded agent that belongs to a privileged user. The adversary does not log in by supplying stolen credentials. Instead, the adversary co-opts an existing trust context. This is classic SSH agent hijacking and it sits under Remote Service Session Hijacking.

T1021.004 Remote Services, SSH: The lateral movement transport is SSH from the bastion to the private instance. The agent supplies authentication, which the target host accepts as if the administrator initiated the session.

T1078 Valid Accounts: Target hosts accept the request because the agent proves possession of a

valid administrative identity. This technique appears in many lateral movement chains because it allows an attacker to blend with normal activity. In this case no secret was stolen, but the agent effectively delivers a valid account in action.

T1098.004 Account Manipulation, SSH Authorized Keys: Although not executed in the lab, an obvious follow on for persistence is modification of authorized keys on private hosts to preserve access with a new key. The existence of an administrator session on a target often makes this practical if controls are not strict.

Impact

Integrity and least privilege: The immediate impact is the failure of least privilege in the private estate. A lower privileged user executed actions as an administrator on a private host. From a business perspective, this is equivalent to an insider or compromised account obtaining administrator privileges without any change in access approvals or console policy.

Separation between application identity and operational identity: The strong application identity posture did not prevent this outcome. The lab validates that excellent user authentication at the edge does not guarantee operational security if older workflows remain in production. This is a crucial governance lesson. Identity must be coherent across the environment, not just at the web boundary.

Operational blast radius: Once an attacker reaches a private host with administrator authority, the potential blast radius includes data exfiltration, manipulation of logs or telemetry, the introduction of backdoors for persistence, and indirect access to internal services. The presence of private data plane endpoints inside the VPC increases the risk that the attacker can leverage the compromised host to trigger actions in other services.

Detectability: The behavior can be detected, but it requires the right signals. On the bastion, logs can indicate agent usage from unexpected processes or users. On the target host, SSH logs reveal which principal authenticated. If the session runs commands whose TTY provenance or environment variables do not match the expected user, that is a strong indicator. Network telemetry that correlates login sources, command patterns, and user identities can further raise the signal to noise ratio.

Detection Opportunities

Bastion level signals: Collect detailed SSH server logs and link them to process accounting so

that agent forwarding events can be correlated with the user who owns the process. Alert when an agent socket associated with one user is accessed, directly or indirectly, by another user or process outside that user's session. Monitor for repeated short-lived authentication attempts that originate from the bastion but are inconsistent with the known administrator workflow.

Target host signals: Aggregate SSH authentication logs and include the public key fingerprint or certificate principal when possible. Correlate the key used with the process owner on the bastion. Alert when the administrator key is used in a pattern that does not match the administrator's normal behavior. Use command auditing on the host to tie each executed command to a TTY and session owner. Unexpected pairings are suspicious.

Control plane signals: If private hosts reach internal services or data plane endpoints, build detections that notice unusual call patterns that follow closely after an SSH login with an administrator key. It is especially useful to compare the source identity at the service layer with the shell user that initiated the action.

Mitigations and Controls

Immediate actions: Disable SSH agent forwarding for all bastion managed workflows. Enforce this in server configuration on the bastion and on all downstream hosts that might act as secondary pivots. Mandate that administrators connect without agent forwarding and provide alternative approved runbooks. Ensure strict file system permissions and process namespace isolation on the bastion so that no user can observe or interact with another user's agent artifacts.

Replace shared bastions with identity aware access: Adopt AWS Systems Manager Session Manager for administrative access to private instances. Session Manager binds access to IAM identities, supports MFA enforcement, and records detailed session logs. With Session Manager, inbound SSH exposure is unnecessary for most workflows. If there are exceptions, isolate them and document them, and still avoid agent forwarding.

Short-lived credentials and device binding: If SSH is required, move to SSH certificates with short time to live and constrained principals. Bind administrator operations to hardware backed keys such as FIDO2 where possible. The objective is to make each use of privileged identity short-lived and tightly controlled by context. Avoid shared keys. Avoid static long lived keys. Rotate keys on an aggressive schedule.

Network segmentation and east-west restrictions: Remove flat network trust relationships that allow the bastion to reach all private subnets by default. Use security groups that limit the

bastion to a narrow set of targets. Limit outbound access from private hosts where possible, so that even if a private host is compromised, it cannot freely reach internal services.

Policy, training, and governance: Update policies to prohibit agent forwarding in production and to favor identity aware access with centralized logging. Train administrators on new runbooks and the security rationale. Integrate configuration checks into CI and change management, for example by failing reviews that enable agent forwarding or that add new shared access patterns without explicit risk approval.

Detection engineering: Create analytics that raise alerts when the same administrator key appears in overlapping sessions from different users or from unusual origin hosts. Alert on any instance where a bastion user who is not an administrator appears to execute administrator commands on a private host. Perform periodic purple team exercises to validate that these analytics detect realistic hijacking behavior with low false positive rates.

Compliance and Governance

NIST Cybersecurity Framework: PR.AC Access control. The controls in this report reinforce unique identity, least privilege, and session restrictions not only at the web edge, but across administrative pathways. Disabling agent forwarding and replacing bastions with Session Manager improves the integrity of access control. DE.CM Monitoring. Centralized logging for SSH activity on both bastion and private hosts, combined with process accounting and session recording, supports the detection of agent hijacking. Specific analytics around agent socket access and mismatched command provenance address the behaviors seen in this lab. RS.MI Mitigation. The immediate response playbook for suspected hijacking should include disabling agent forwarding at configuration level, rotating any affected keys, and reviewing all administrator authorized keys on private hosts for integrity.

SOC 2. CC6 Logical access controls: Policies must cover administrative pathways with the same rigor as application identity. Agent forwarding on shared bastions is inconsistent with strong logical access controls. identity-bound administrative access, short-lived credentials, and detailed session logging are aligned with SOC 2 expectations. CC7 Change management and monitoring. Moves to Session Manager, SSH certificates, and stricter segmentation should pass through formal change control with risk analysis. Monitoring and detection for agent misuse are ongoing operational controls that support this criterion.

Documentation and audit evidence: The architecture choices at the web tier already provide a

clean evidentiary record for auditors, since Hosted UI centralizes credential handling and the edge is hardened with WAF and strict headers. Extending that discipline to the operational tier by retiring bastions and adopting identity aware administrative access provides additional audit friendly artifacts, including IAM based authorization records and session transcripts.

Ethical Considerations and Limitations

The lab operated within an ethical framework and used a controlled environment with synthetic accounts and systems. No user data was accessed. The methods described avoided exploit details and focused instead on the architectural logic of the attack path. The lab did not execute persistence actions, such as modifying authorized keys on private targets, and did not attempt to escalate privileges on the private host after administrator access was obtained. Telemetry varied across hosts, which is typical of mixed maturity environments. Production deployments should standardize SSH logging configurations and centralize collection and analysis.

Conclusion

The lab demonstrates a clear lesson for enterprise security architecture. Strong authentication at the web edge is necessary, but it does not ensure end to end protection when older trust patterns remain active elsewhere in the environment. A lower privileged user on a shared bastion was able to leverage an administrator's forwarded SSH agent to obtain administrator authority on private hosts. No credential theft occurred, and the web authentication flow was never challenged. The attacker succeeded by using the administrator's live trust context indirectly. The attack chain maps to well documented adversary techniques in MITRE ATT&CK, including Remote Service Session Hijacking and Remote Services over SSH. The mitigation program is equally well established. Disable agent forwarding, retire shared bastions in favor of identity aware administrative access, adopt short-lived and device-bound credentials, and build detection around SSH session provenance that can reveal agent misuse. When these measures are adopted, the organization preserves the integrity benefits of its modern authentication layer and closes the back door that legacy bastion practices create.

References

CISA. Eviction Strategies Tool, T1563.001 SSH Hijacking. https://www.cisa.gov/eviction-strategies-tool/info-attack/T1563.001

MITRE ATT&CK. T1021.004 Remote Services: SSH. https://attack.mitre.org/techniques/T1021/004/

MITRE ATT&CK. T1078 Valid Accounts. https://attack.mitre.org/techniques/T1078/

MITRE ATT&CK. T1098.004 Account Manipulation: SSH Authorized Keys. https://attack.mitre.org/techniques/T1098/004/

MITRE ATT&CK. T1563 Remote Service Session Hijacking. https://attack.mitre.org/techniques/T1563/

MITRE ATT&CK. T1563.001 Remote Service Session Hijacking: SSH. https://attack.mitre.org/techniques/T1563/001/

Amazon Web Services. Amazon Cognito Hosted UI. https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-userpools-app-idp-settings.html

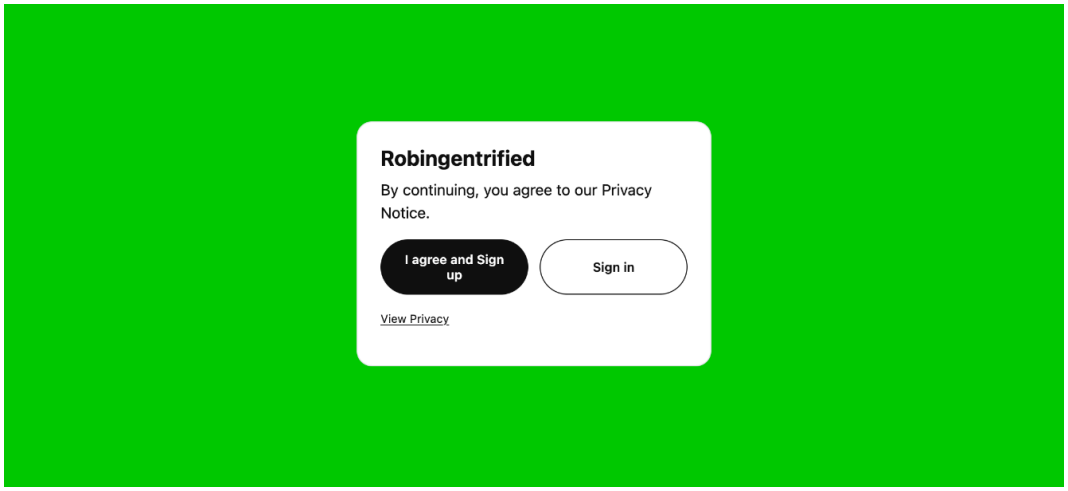
Amazon Web Services. Using PKCE with Authorization Code Grant in Amazon Cognito. https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-pool-app-integration.html

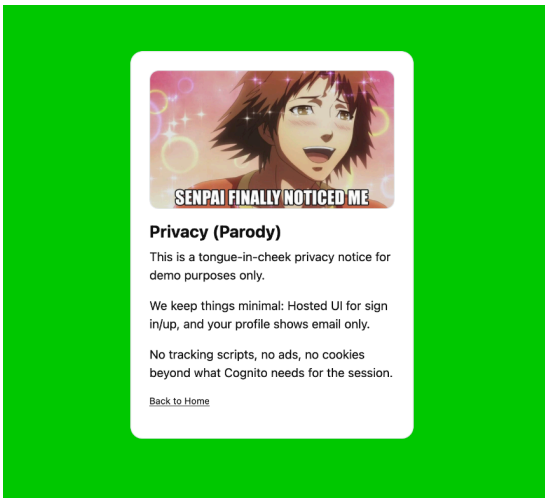
Amazon Web Services. CloudFront with Origin Access Control for private S3. https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html

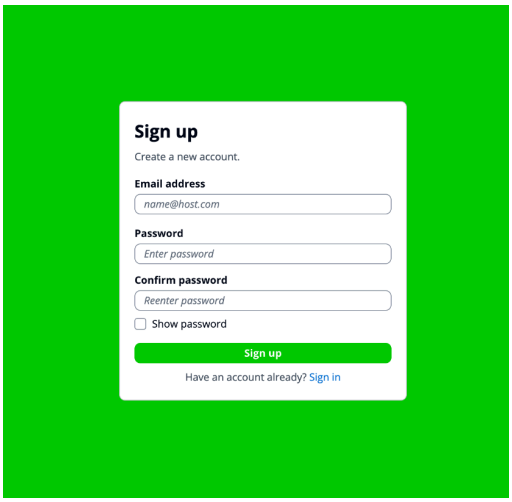
Amazon Web Services. AWS WAF on CloudFront distributions. https://docs.aws.amazon.com/waf/latest/developerguide/aws-waf-cloudfront.html

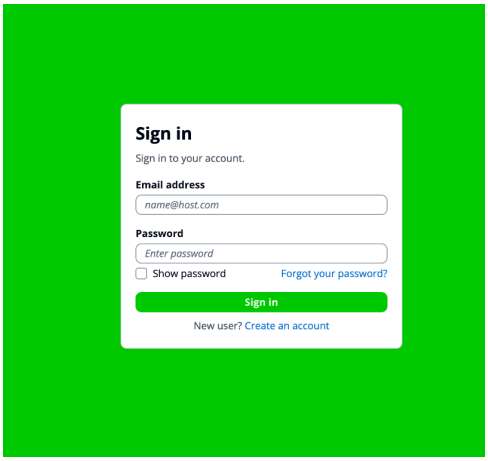
Amazon Web Services. AWS Systems Manager Session Manager. https://docs.aws.amazon.com/systems-manager/latest/userguide/session-manager.html

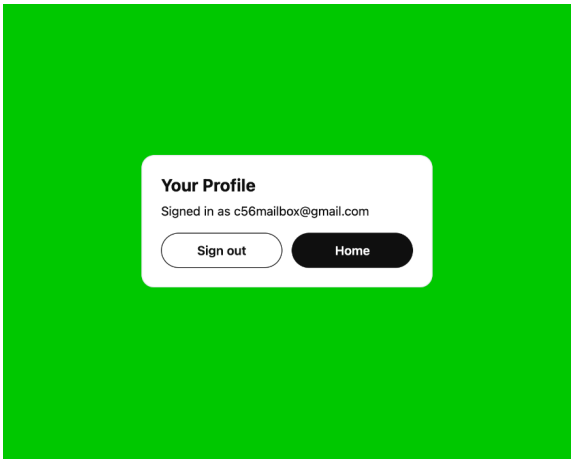
Screenshots











```
[redacted] downloads % eval "$(ssh-agent -s)"
Agent pid 2577
[redacted] downloads % ssh-add ./bastpriv.pem
Identity added: ./bastpriv.pem (./bastpriv.pem)
[redacted] downloads % ssh -A -i "bastpriv.pem" ec2-user@ec2-107-2[redacted].compute-1.amazonaws.com
Last login: Mon Sep  1 20:13:49 2025 from 143.2[redacted]
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-1-96 ~]$ echo $SSH_AUTH_SOCK
/tmp/ssh-[redacted]/agent.601
[ec2-user@ip-10-0-1-96 ~]$ ssh-add -L
error fetching identities for protocol 1: agent refused operation
ssh-rsa AAAAB3NzaC1yc2[redacted]

[ec2-user@ip-10-0-1-96 ~]$ ssh ec2-user@10.0.2.19
Last login: Mon Sep  1 20:13:57 2025 from ip-10-0-1-96.ec2.internal
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-2-19 ~]$
```

```
[redacted] downloads % eval "$(ssh-agent -s)"
Agent pid 2590
[redacted] downloads % ssh -A -i "bastonly.pem" ec2-user@ec2-107-2[redacted].compute-1.amazonaws.com
Last login: Mon Sep  1 20:15:54 2025 from 143.2[redacted]
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-1-96 ~]$ echo $SSH_AUTH_SOCK
/tmp/ssh-[redacted]/agent.690
[ec2-user@ip-10-0-1-96 ~]$ find /tmp -type s -name "agent*"
find: '/tmp/systemd-private-e93df9[redacted]chronyd.service-Smn[redacted]': Permission denied
/tmp/ssh-[redacted]/agent.601
/tmp/ssh-[redacted]/agent.690
[ec2-user@ip-10-0-1-96 ~]$ ssh-add -L
error fetching identities for protocol 1: agent refused operation
The agent has no identities.
[ec2-user@ip-10-0-1-96 ~]$ ssh ec2-user@10.0.2.19
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[ec2-user@ip-10-0-1-96 ~]$ export SSH_AUTH_SOCK=/tmp/ssh-[redacted]/agent.601
[ec2-user@ip-10-0-1-96 ~]$ ssh-add -L
error fetching identities for protocol 1: agent refused operation
ssh-rsa AAAAB3NzaC1yc2[redacted]

[ec2-user@ip-10-0-1-96 ~]$ ssh ec2-user@10.0.2.19
Last login: Mon Sep  1 20:21:19 2025 from ip-10-0-1-96.ec2.internal
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-2-19 ~]$
```