WHITE PAPER

THE FIVE NASTIEST SECURITY MISTAKES LURKING IN PUBLIC CLOUD INFRASTRUCTURE
Public cloud infrastructure has become a critical asset to enterprises that compete on innovation. Protecting that competitive edge means finding and eliminating serious infrastructure exposures—many of them caused by preventable, common mistakes.

Since software started eating the world, nearly every industry has become dependent on software to deliver market-facing products and services. Software is now a competitive battleground for all types of companies, ranging from accounting SaaS providers to car manufacturers to even apparel manufacturers. Even shoes come with software. Because software innovation is fast, it’s often a key source of competitive advantage.

The competitive battle for continuous software innovation means fast, iterative deployment is important to keep new features in front of customers. Cloud infrastructure and DevOps have proven revolutionary in accelerating software delivery, in turn accelerating innovation and competitive opportunity. Mistakes that expose IaaS environments to compromise directly impact the businesses that depend on those environments.

Cloud infrastructure is incredibly powerful and flexible, with the number of services offered by providers growing constantly. On-demand resources like elastic compute, object storage, and advanced database services all have many variations and configuration options available. The possible ways these assets can be configured and interact are almost limitless. This is the power that IaaS providers put into your hands, but that power comes with shared responsibility between the provider and the customer. As with any powerful technology, cloud infrastructure has to be used knowledgeably and responsibly to ensure its potential value doesn’t turn to chaos, or worse.

Provider responsibilities are typically very well defined by the providers themselves. More nuanced is the customer’s responsibility, which itself has two related facets:

- DevOps teams must architect, configure, and operate cloud-based application stacks in alignment with security and compliance standards. The scope of one DevOps team’s responsibility is limited to the application stack (or stacks) that they own.
- Security teams must set the standards, monitor for consistent compliance, and deliver data to many DevOps teams in a manner that can be easily consumed and acted upon by those teams. The scope of one security team typically encompasses many, if not all, DevOps teams and their application stacks.

In many ways the security teams have the tougher responsibility—they’re accountable for catching what many others may have missed. They’re the fail-safe for every application in the enterprise, and have to collaborate with many DevOps teams—and keep up with them consistently and effectively.
Missing just one exposure can spell disaster. The configuration permutations exposing the environment to attack are immeasurable and the bad guys only have to find one. On the other hand, the security stakeholders have to get an enormous number of points right, every time. This in an environment where change is constant because infrastructure is code, systems are ephemeral, and infrastructure deployment is continuous. So much is happening so fast and in so many places that it’s easy for something to get missed.

It’s important to know which issues are exposing your cloud infrastructure environments at any given time, which is a job for automation. For years, CloudPassage has worked with some of the best-known logos in the world on this exact issue. We’ll share some of our experiences about what works and what doesn’t a little later.

For now, let’s look at our list of the five nastiest mistakes that can lurk in IaaS environments. Reading about every possible issue would be exhaustive, but of course, some exposures are nastier than others.

Here are a few that we believe stand out as the nastiest.

The Five Nastiest Mistakes

Easily Hacked Administrative Credentials

The Issue
Administrative credentials are literally the keys to the kingdom. Weak controls over administrative credentials equates to a low-effort, high-value target for attackers. Methods for attacking cloud service provider accounts are simple and well known, from old-school dictionary attacks to AI-backed “smart guessing” tools that combine stolen credentials with publicly available information. The only thing the attackers need is a badly configured set of privileged credentials.

How It Happens
Exposures created from easily hacked credentials often stem from one of two places: cutting corners in the heat of battle, and the decay of initial diligence.

When a new cloud infrastructure stack is built, there are some best practices that are quite well defined and straightforward to implement. Even so, it’s all too common for there to be a mad rush to get things moving—after all, the expectations of cloud infrastructure border on magic. In this mad rush it’s easy to back-burner implementing things like multi-factor authentication, password construction rules, and minimized blast radius. With the best
intentions of getting back to these details, other needs pile on and before you know it those best practices are buried under a massive backlog of other issues.

Even when diligence is addressed upfront, it’s not unusual to see the effects of that diligence decay. Changes over time can chip away at an otherwise well-configured administrative access structure. This is especially common when administrative restrictions stand in the way of key functionality or deployments. It’s all too easy to loosen up the control structure, which allows exposures to creep in. It’s not enough to get access credential controls right at the onset—they must be maintained on an ongoing basis.

Regardless of the source, capturing the keys to a cloud-infrastructure kingdom is the brass ring for attackers—so they’ll be diligent in looking for the weak spots. Only ongoing diligence can prevent their finding them.

**Why It’s One Of The Nastiest**

It’s obvious that compromised administrative credentials means gross compromise of anything within the scope of those credentials. Cloud infrastructure environments are often broken up into many smaller IaaS accounts for this very reason—the blast radius of compromise is minimized.

The most obvious implication of gross compromise is mass data exposure. That’s nasty enough, but it goes further. Attackers who gain access to administrative credentials often play it cool, because often the environment itself can be more valuable than even the data exposed to them. Installation of backdoors and rootkits, injecting bitcoin mining software into workloads, and even running full-blown botnets are possible with full administrative access. And make no mistake, with complete privileges at their disposal, attackers can make this kind of compromise extremely difficult to find.

These factors put easily hacked administrative credentials at the top of our list for nastiest mistakes that can expose your cloud infrastructure.

**Exposed Data Assets**

**The Issue**

Data-related services such as object storage, indexing and search engines, and database services are among the most valuable and heavily-used IaaS resources. If administrative credentials are the keys to the kingdom, the data stored in these services are the crown jewels. Poor configuration of data services and related security controls (e.g. authentication and network access mechanisms) can leave those jewels in a state of extreme exposure—an easy win for attackers.
How It Happens

Data exposure in the cloud typically results from insecure settings or insufficiently granular access controls. Whether this occurs in the initial configuration or the heat of a technical crisis, it can leave your data unintentionally discoverable and accessible to unauthorized parties.

Decentralized management of infrastructure resources is a key attribute of cloud infrastructure that enables enterprise agility. This flexibility has significant advantages and necessarily distributes control, but not all those with such control are versed in security practices. As a result, poor configurations in IaaS data resources can crop up as they are deployed with a focus on function over security, or with little attention to security at all. These issues can range from the location of data being discoverable to full public access—mistakes that can have broad-sweeping impact with the click of a button.

Configuration drift and access scope dilation increase the risk of data loss as demand for certain data grows and more systems need to interact with it. From urgent troubleshooting tactics to good old fashioned fat-finger mistakes, erroneous configuration drift can be introduced with shocking ease and can linger undetected in an environment for long periods of time.

Why It’s One of the Nastiest

The goal of many compromises is to export data, and compromise of these IaaS data services leads directly to large scale data loss. While other types of compromise can damage the business, loss of data carries the largest liability of issues with the courts, public opinion, and regulators.

While a bulk export of data is the most obvious way damage could happen, perhaps more nefarious is the classic “salami attack” in which the data is slowly siphoned off so as not to draw attention. In this scenario, there are no unusual data transfer volumes to alert system owners. In fact, the system may appear to be operating completely normally with misconfigurations allowing “permitted” access and the attack appearing part of regular operations.

If attackers find writable data stores, corruption of downstream processes by mangling data or injecting meaningfully incorrect data can have massive impacts and can be very hard to troubleshoot. Not every attack is focused on removing data. Some are intended as industrial sabotage and are especially nefarious as the system continues to operate in a seemingly normal way until corrupted data propagates.
With on-demand IaaS data services, it becomes easier than ever to deploy and manage data. It also makes it easy for data storage locations to proliferate and for misconfigurations to be introduced undetected, making exposed data services one of the nastiest security mistakes.

**Weak Network Access Controls**

**The Issue**
Just like data centers, cloud infrastructure environments have network perimeters—moats, the first line of defense. IaaS platform features (e.g. security groups), host-based firewalls, and third-party cloud gateways are all examples of logical network access controls that can protect cloud infrastructure. But these controls are only as good as their configurations. Poorly configured network access controls weaken the first line of defense for the cloud assets they’re intended to protect.

**How It Happens**
In a word, complexity.

Traditional perimeter firewall architectures are sophisticated, but the general approach of fewer policy enforcement points was relatively simple. Firewall choke points primarily enforcing north-south traffic at the network edge, although not trivial, results in a relatively straightforward set of policies to manage.

Distributed cloud firewalls can increase complexity significantly, especially in larger IaaS environments. Instead of fewer firewalls protecting entire data centers, a greater number of network access mechanisms protect many small groups of assets. This multiplies the number of policy enforcement points to be managed, in turn creating more interaction and overlap between policies. Both of these factors can lead to confusion about why network traffic is not flowing through an environment, which can, in turn, lead to loosening network access policies to an overly permissive state.

This situation isn’t uncommon in the midst of troubleshooting an outage or other production problem. In the middle of the night when things are in a tailspin, it’s easy to relax complex, layered webs of network access controls to address an actual or possible connectivity issue. It’s also easy to skip tightening things back down when the crisis is resolved for fear of recreating the problem—or just because someone just honestly forgot.
Why It's One of the Nastiest

Network access controls are aptly named “firewalls” because they’re indeed the barrier between safety and danger. This first line of defense is also the broadest—even though firewalls cannot stop every attack, they typically stop the vast majority of weaker, less-focused attacks. When network access controls are weak, it opens up the assets behind them to a larger universe of attacks—perhaps less sophisticated, but no less able to wreak havoc. In some cases, a barrage of seemingly unsophisticated attacks splashing against an environment is just the distraction needed to enable more sophisticated attacks to slip by undetected.

A well-architected network access configuration is also critical as the very last line of defense in many threat situations. If an attacker makes it through the inbound firewalls and finds other vectors enabling them to penetrate interior systems, the only thing stopping a data extrusion is often outbound network access controls.

The first line of defense is keeping attackers out. The last line of defense is keeping the crown jewels in. Misconfiguration of these critical “bookend” defenses warrants ranking as one of the nastiest cloud security mistakes to quickly detect and resolve.

Unconstrained Blast Radius

The Issue

Provider-delivered security combined with customer implementation of best practices go a long way to preventing attacks. But attacks will happen, and some will succeed. No control structure can eliminate all human error, social engineering, and threats from within. Controlling the scope of impact when an attack succeeds—constraining the blast radius—is critical to minimizing damage and secondary attacks. Failure to control blast radius can result in a successful attack having extremely far-reaching consequences, making analysis and clean-up very difficult.

How It Happens

Cloud infrastructure moves at a very fast pace, making it easy to overlook the steps needed to limit the blast radius of breaches. In some cases, security is not a factor in architectural or operational decisions, which results in large accounts with soft internal controls (similar to the old “hard on the outside, soft in the center” data center problems).

The technical reason for excessive blast radius is simple: failure to segment IaaS accounts, easily identifiable by the presence of a few large IaaS accounts (each contains many resources) versus many IaaS accounts with fewer resources each. The “more but smaller
accounts’ model is becoming very popular, with some enterprises assigning one or two IaaS accounts per application. With the failure to segment IaaS accounts, the breach of one account can expose much larger swaths of infrastructure assets to a successful intruder.

Failure to exercise the principle of least-needed privilege also increases compromise blast radius. The overuse of delegated roles or other cross-account access control mechanisms is a related and common mistake. Systems should ideally use credentials that provide only the access they need, but in rapidly evolving cloud environments it is common to provide broader general access for convenience. This expands the blast radius of an initial compromise, both in terms of the initial damage and opportunities for escalation, lateral movement, and establishing threat persistence.

Insecure automation layers can lead to broad and insidious compromise. There are many valuable tools for automating management and deployment of cloud infrastructure, and these tools necessarily have the ability to make broad, deep changes very quickly. Malicious or erroneous use of these powerful tools can propagate compromises across cloud environments, unfettered and at blinding speed.

Why It’s One of the Nastiest

No matter how a compromise happens, a large blast radius can enable a breach to be truly expansive.

Modern attack automation can map an environment and effect lateral (a.k.a. “east-west”) movement with blinding speed. A manifest of all resources in an IaaS account is easily accessible to an attacker with compromised administrative access. That is basically a shopping list that the attacker can study to build compromise vectors for the environment and the assets within it.

It’s also critical to realize that when a successful attack does occur, the blast radius boundary is likely the only thing containing the compromise. Attacks are automated and happen quickly, and there won’t be time for a human to respond. Not having the containment effect of a small blast radius before the attack occurs is critical to minimizing its impact.

Compromises will happen and containment before the fact is critical. Failure to implement this simple best practice amplifies the worst of problems at the worst possible times. That makes it one of the nastiest mistakes lurking in IaaS environments.
Poor Event Logging

The Issue
Detecting events that indicate the potential for a threat or an active attack is a critical function. If protective controls fail for any reason (like honest mistakes, cutting corners, or malicious actions) detection is the very last line of defense. Although a roll of the dice, reacting quickly to indicators of threat activity can sometimes prevent the worst. When a threat does make it through your defenses, logged events are among the most important data to determine the impact, scope, and cause. Poorly configured logging mechanisms are a mistake guaranteed to enable attacks and seriously impede efforts to detect, contain and analyze resulting compromises.

How It Happens
Correct and comprehensive event logging is a more complicated task than it once was. Systems were once more centralized—going back to the early days of computing, mainframes were replaced by distributed server hosts, which were replaced by virtual machines, which are now being replaced by containers and even serverless compute environments. Each phase of this evolution has resulted in more distributed systems and more individual logging mechanisms that must be correctly configured. The breadth of available technologies can also challenge effective event logging given the countless types of systems with different logging mechanisms, event types, and data formats.

Maintaining consistent standards across large numbers of disparate logging mechanisms can lead to “best-effort” practices which are rarely good enough. Along with many other tasks that aren’t on the critical path to production, event logging is another corner that’s seductively easy to cut. This is particularly true in highly dynamic environments, where infrastructure and application components are being added, removed and changed constantly.

There’s also the “pendulum effect” that occurs when event configuration does not have strong standards backed by automated implementation. In a well-intended effort to get total visibility, logging mechanisms may be maxed out, sometimes even running in debug mode. This leads to operational problems like filesystems being overwhelmed and excessive cloud service fees for generating, moving, and storing events that are mostly noise. The “pendulum” swings in the other direction when logging is cut back to a more tenable volume, which fixes the operational issues but results in blind spots if not done carefully.

Perhaps the most disturbing cause of poor logging mechanisms is malicious action. Attackers subverting logging mechanisms is a well-known tactic, often one of the first steps towards bigger things once privileged access has been gained. The implications are obvious.
## Why It’s One of the Nastiest

Without event logging, you’re blind to what’s happening in your environment and all ability to perform after-the-fact forensics or other retrospective analysis is lost. During the attack, you won’t be able to see the attackers’ movements or assess the tactics they’re deploying. There will be no ability to evaluate the current and imminent impact of the attack. You’ll be responding to the incident blindfolded.

In the aftermath of an attack, 20/20 hindsight is the best you can hope for, but without effective event logging, you won’t even have that. Your ability to assess how the attacker initially penetrated your environment or the attack methods they deployed will be very limited. When you do find systems subverted or exposed, you won’t know if those conditions already existed or if they’re a result of the attacker. And most concerning, you may not be able to determine if attackers are still present—just because the obvious activities stop doesn’t mean the attackers are done.

As opposed to strong prevention, trying to react to incidents puts you in a weak defensive posture to begin with. Poorly configured logging mechanisms undermine even this last foothold, making it one of the nastiest mistakes that can expose your cloud infrastructure to risk.

## See If You’re Exposed, Fast and Free

The five nastiest mistakes we’ve shared here are serious concerns, but they’re just the tip of the iceberg. There are many more issues like these with potential consequences just as severe.

Want to find out if your cloud environment is exposed? We can help.

CloudPassage offers a free trial of the Halo cloud security & compliance platform. Halo is a SaaS-based solution with a quick and easy setup. Most cloud infrastructure environments can be fully assessed within fifteen minutes.

With a Halo trial, you can assess the security and compliance issues in your AWS or Azure cloud infrastructure. You can also take a deep-dive into your server and container environments using the same trial account. Halo includes a number of policy templates for standards like CIS Benchmarks, PCI, HIPAA, and ISO 27001/2. And Halo doesn’t just show you problems - it also provides detailed technical remediation advice for identified issues.
If you want to try more advanced capabilities, check the Halo REST API to try some DevSecOps style automation. See what’s in the Halo Toolbox for inspiration and “starter” code, and plug into the Halo API SDK to make integrations go faster.

**Missing Links? No Problem.**

Register for a Halo free trial:
https://pages.cloudpassage.com/cloud-secure-free-trial.html

Check out the Halo REST API:
https://api-doc.cloudpassage.com

See what’s in the Halo Toolbox:
https://cloudpassage.github.io/halo-toolbox/

Browse the Python SDK for Halo:
https://github.com/cloudpassage/cloudpassage-halo-python-sdk

CloudPassage Cloud Secure Sample Screens

![CloudPassage Sample Screens](image)

Executive summary of all cloud assets and their security compliance status across multiple accounts
### IaaS Resource Issues

<table>
<thead>
<tr>
<th>Count</th>
<th>Critical</th>
<th>Issue Name</th>
<th>Policy Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>Critical</td>
<td>Ensure no security groups allow ingress from 0.0.0.0/0 to port 22 (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>27</td>
<td>Critical</td>
<td>AWS E2C Instance Naming Conventions</td>
<td>CloudPassage AWS E2C Best Practices v1.0</td>
</tr>
<tr>
<td>25</td>
<td>Critical</td>
<td>AWS E2C Egress AMI</td>
<td>CloudPassage AWS E2C Best Practices v1.0</td>
</tr>
<tr>
<td>18</td>
<td>Critical</td>
<td>Ensure the default security group of every VPC restricts all traffic (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>18</td>
<td>Critical</td>
<td>Ensure VPC flow logging is enabled in all VPCs (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>4</td>
<td>Critical</td>
<td>AWS E2C AMI Naming Conventions</td>
<td>CloudPassage AWS E2C Best Practices v1.0</td>
</tr>
<tr>
<td>4</td>
<td>Critical</td>
<td>Enable MFA for AWS IAM Users</td>
<td>CloudPassage AWS IAM Best Practices v1.1</td>
</tr>
<tr>
<td>3</td>
<td>Critical</td>
<td>Enable S3 Bucket Lifecycle Configuration</td>
<td>CloudPassage AWS S3 Best Practices v1.0</td>
</tr>
<tr>
<td>3</td>
<td>Critical</td>
<td>Ensure credentials unused for 90 days or greater are disabled (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>3</td>
<td>Critical</td>
<td>Ensure access keys are rotated every 90 days or less (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>3</td>
<td>Critical</td>
<td>Enable MFA Delete for AWS S3 Buckets</td>
<td>CloudPassage AWS S3 Best Practices v1.0</td>
</tr>
<tr>
<td>3</td>
<td>Critical</td>
<td>Enable Access Logging for AWS S3 Buckets</td>
<td>CloudPassage AWS S3 Best Practices v1.0</td>
</tr>
<tr>
<td>2</td>
<td>Critical</td>
<td>Unauthorized IAM cross-account roles trusting external accounts</td>
<td>CloudPassage AWS IAM Best Practices v1.1</td>
</tr>
<tr>
<td>2</td>
<td>Critical</td>
<td>Enable CloudFormation stack termination protection</td>
<td>CloudPassage AWS CloudFormation Best Practices v1.0</td>
</tr>
<tr>
<td>2</td>
<td>Critical</td>
<td>Server Side Encryption</td>
<td>CloudPassage AWS S3 Best Practices v1.0</td>
</tr>
<tr>
<td>2</td>
<td>Critical</td>
<td>Ensure multi-factor authentication (MFA) is enabled for all IAM users that have a console pass...</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Ensure IAM password policy prevents password reuse (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Enable Hardware MFA for AWS Root Account</td>
<td>CloudPassage AWS IAM Best Practices v1.1</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Ensure S3 bucket access logging is enabled on the CloudTrail S3 bucket (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Enable S3 bucket access logging</td>
<td>CloudPassage AWS IAM Best Practices v1.1</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>CloudTrail IAM Password Policy</td>
<td>CloudPassage AWS IAM Best Practices v1.1</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Enable MFA Delete for AWS CloudTrail bucket</td>
<td>CloudPassage AWS CloudTrail Best Practices v1.0</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Enable Route 53 domain transfer lock</td>
<td>CloudPassage AWS Route53 Best Practices v1.0</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Enable AWS RDS Event Notifications</td>
<td>CloudPassage AWS RDS Best Practices v1.0</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Ensure no security groups allow ingress from 0.0.0.0/0 to port 3389 (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>1</td>
<td>Critical</td>
<td>Ensure IAM policies that allow full &quot;*&quot; administrative privileges are not created (Scored)</td>
<td>CIS AWS Foundations Benchmark v1.1</td>
</tr>
<tr>
<td>64</td>
<td>Non-critical</td>
<td>Unrestricted Outbound Access on All Ports</td>
<td>CloudPassage AWS E2C Best Practices v1.0</td>
</tr>
<tr>
<td>64</td>
<td>Non-critical</td>
<td>Empty Description for Security Group Rules</td>
<td>CloudPassage AWS E2C Best Practices v1.0</td>
</tr>
<tr>
<td>64</td>
<td>Non-critical</td>
<td>Security Group Naming Conventions</td>
<td>CloudPassage AWS E2C Best Practices v1.0</td>
</tr>
</tbody>
</table>

**Figure 2.** Security Policy violations rolled up by issue name
Figure 3.
A list of assets violating a specific policy rule to not allow ingress to all traffic on port 22
Figure 4.
Detailed information about a violation of the policy rule to not allow ingress to all traffic on port 22 on a specific asset (in this case a Security Group)
Figure 5.
JSON details of the configuration that violated the rule (actual violation is in the unexpanded “SecurityGroupIpPermissions” section of the JSON)
ABOUT CLOUDPASSAGE

CloudPassage is the recognized leader in automated cloud security and compliance for dynamic application deployment environments like AWS and Azure. A true pioneer, the company’s groundbreaking innovations received the first-ever patents granted in the cloud security domain.

Today, CloudPassage safeguards cloud infrastructure for the world’s best-recognized brands in finance, e-commerce, gaming, B2B SaaS, and digital media with Halo, its flagship solution. Halo is an award-winning cloud security platform that automates continuous visibility for millions of serverless, server-based, and containerized assets across hundreds of public and hybrid cloud environments. Halo is software-as-a-service, deploying in minutes and scaling effortlessly. Halo integrates with configuration management and CI/CD tools such as Puppet, Chef, and Jenkins to align security functions with automated DevOps processes. CloudPassage is a proven solution for delivering automated security and compliance visibility, critical to protecting data and applications migrating to public IaaS environments.

Visit www.cloudpassage.com to learn how Halo can enable faster, more effective cloud infrastructure security for your enterprise.