



AWSWhitepaper

# Le migliori pratiche WordPress per AWS



# Le migliori pratiche WordPress per AWS: AWSWhitepaper

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

I marchi e l'immagine commerciale di Amazon non possono essere utilizzati in relazione a prodotti o servizi che non siano di Amazon, in una qualsiasi modalità che possa causare confusione tra i clienti o in una qualsiasi modalità che denigri o discrediti Amazon. Tutti gli altri marchi non di proprietà di Amazon sono di proprietà delle rispettive aziende, che possono o meno essere associate, collegate o sponsorizzate da Amazon.

# Table of Contents

Sintesi .....	1
Sei Well-Architected? .....	1
Introduzione .....	2
Implementazione semplice .....	3
Considerazioni .....	3
Approcci disponibili .....	3
Amazon Lightsail .....	4
Selezione di un piano tariffario Amazon Lightsail .....	4
Installazione WordPress .....	5
Recupero da un guasto .....	5
Miglioramento delle prestazioni e ottimizzazione dei costi .....	7
Accelerazione della distribuzione di contenuti .....	7
Offload di contenuti statici .....	8
Contenuti dinamici .....	8
Caching di database .....	9
Caching bytecode .....	10
Distribuzione elastica .....	11
Architettura di riferimento .....	11
Scalabilità del livello web .....	13
Livello stateless .....	14
Storage condiviso (Amazon S3 e Amazon) EFS .....	15
Livello dati (Amazon Aurora e Amazon) ElastiCache .....	16
Conclusione .....	18
Collaboratori .....	19
Revisioni del documento .....	20
Appendice A: configurazione CloudFront .....	21
Origini e comportamenti .....	21
CloudFront creazione di distribuzione .....	21
Appendice B: Dati di configurazione statici dei contenuti .....	25
Creazione di utenti .....	25
Creazione di bucket Amazon S3 .....	25
Creazione di origini statiche .....	27
Appendice C: Backup e ripristino .....	28
Appendice D: Implementazione di nuovi plug-in e temi .....	30

Note ..... 31

AWS Glossario ..... 32

..... xxxiii

# Le migliori pratiche per WordPress on AWS

Data di pubblicazione: 19 ottobre 2021 () [Revisioni del documento](#)

Questo white paper fornisce agli amministratori di sistema indicazioni specifiche su come iniziare a usare Amazon Web WordPress Services (AWS) e su come migliorare sia l'efficienza in termini di costi dell'implementazione che l'esperienza dell'utente finale. Delinea inoltre un'architettura di riferimento che soddisfa i requisiti comuni di scalabilità e alta disponibilità.

## Sei Well-Architected?

Il [AWS Well-Architected](#) Framework ti aiuta a comprendere i pro e i contro delle decisioni che prendi quando crei sistemi nel cloud. I sei pilastri del Framework consentono di apprendere le best practice architetturali per progettare e gestire sistemi affidabili, sicuri, efficienti, economici e sostenibili.

Utilizzando [AWS Well-Architected Tool](#), disponibile gratuitamente in [Console di gestione AWS](#), puoi esaminare i tuoi carichi di lavoro rispetto a queste best practice rispondendo a una serie di domande per ogni pilastro.

[Per ulteriori indicazioni e best practice da parte degli esperti per la tua architettura cloud \(implementazioni dell'architettura di riferimento, diagrammi e white paper\), consulta l'Architecture Center.AWS](#)

# Introduzione

WordPress è uno strumento open source di creazione di blog e un sistema di gestione dei contenuti (CMS) basato su PHP e MySQL adatto per creare sia blog personali sia siti Web a traffico elevato.

La prima versione di WordPress è stata rilasciata nel 2003, per cui non è stata creata secondo i criteri moderni di elasticità e scalabilità propri delle infrastrutture cloud. Grazie al lavoro della comunità WordPress e al rilascio di diversi moduli, le capacità di questa soluzione CMS sono in continua espansione. Oggi è possibile creare un'architettura WordPress in grado di sfruttare numerosi vantaggi di AWS Cloud.

# Implementazione semplice

Per blog o siti Web a basso traffico senza requisiti rigorosi di alta disponibilità, potrebbe essere adatta una semplice implementazione di un singolo server. Questa implementazione non è l'architettura più resiliente o scalabile, ma è il modo più rapido ed economico per far funzionare il tuo sito Web.

## Argomenti

- [Considerazioni](#)
- [Approcci disponibili](#)
- [Amazon Lightsail](#)

## Considerazioni

Questa discussione inizia con l'implementazione di un singolo server Web. Potrebbero esserci occasioni in cui lo diventi troppo, ad esempio:

- La macchina virtuale su cui è distribuito il tuo WordPress sito Web è un singolo punto di errore. Un problema con questa istanza causa una perdita di servizio per il tuo sito web.
- La scalabilità delle risorse per migliorare le prestazioni può essere ottenuta solo mediante la «scalabilità verticale», ovvero aumentando le dimensioni della macchina virtuale su cui è in esecuzione il WordPress sito Web.

## Approcci disponibili

AWS dispone di diverse opzioni per il provisioning delle macchine virtuali. Esistono tre modi principali per ospitare il proprio WordPress sito Web su AWS:

- Amazon Lightsail
- Amazon Elastic Cloud (AmazonEC2)
- Marketplace AWS

[Amazon Lightsail](#) è un servizio che consente di avviare rapidamente un server privato virtuale (un'istanza Lightsail) per ospitare un sito Web. WordPress Lightsail è il metodo più semplice per iniziare se non hai bisogno di tipi di istanze altamente configurabili o di accedere a funzionalità di rete avanzate.

[Amazon EC2](#) è un servizio web che fornisce una capacità di elaborazione ridimensionabile in modo da poter avviare un server virtuale in pochi minuti. Amazon EC2 offre più opzioni di configurazione e gestione rispetto a Lightsail, il che è preferibile nelle architetture più avanzate. Hai accesso amministrativo alle tue EC2 istanze e puoi installare qualsiasi pacchetto software tu scelga, inclusi. WordPress

[Marketplace AWS](#) è uno store online in cui è possibile trovare, acquistare e implementare rapidamente software eseguibile su AWS. Puoi utilizzare la distribuzione 1-Click per avviare WordPress immagini preconfigurate direttamente su Amazon EC2 nel tuo AWS account in pochi minuti. Esistono diversi fornitori di Marketplace che offrono ready-to-run WordPress istanze.

Questo white paper illustra l'opzione Lightsail come implementazione consigliata per un sito Web con server singolo. WordPress

## Amazon Lightsail

Lightsail è il modo più facile per iniziare AWS a utilizzare per gli sviluppatori, le piccole aziende, gli studenti e altri utenti che necessitano di una soluzione semplice di server VPS privato virtuale ().

Il servizio sottrae all'utente molti degli elementi più complessi della gestione dell'infrastruttura. È quindi un punto di partenza ideale se hai meno esperienza nell'infrastruttura o quando devi concentrarti sulla gestione del tuo sito Web e un prodotto semplificato è sufficiente per le tue esigenze.

Con Amazon Lightsail, puoi scegliere i sistemi operativi Windows o Linux/Unix e le applicazioni Web più diffuse, WordPress incluse, e distribuirle con un solo clic da modelli preconfigurati.

Man mano che le tue esigenze crescono, hai la possibilità di superare senza problemi i limiti iniziali e connetterti a servizi aggiuntivi di AWS database, storage di oggetti, caching e distribuzione di contenuti.

## Selezione di un piano tariffario Amazon Lightsail

Un piano [Lightsail definisce il costo](#) mensile delle risorse Lightsail che utilizzi per ospitare il tuo sito web. WordPress Sono disponibili diversi piani per coprire una varietà di casi d'uso, con diversi livelli di CPU risorse, memoria, archiviazione su unità a stato solido () SSD e trasferimento dati. Se il tuo sito web è complesso, potresti aver bisogno di un'istanza più grande con più risorse. Puoi raggiungere questo obiettivo migrando il tuo server a un piano più ampio [utilizzando la console Web](#) o come descritto nella documentazione di [Amazon Lightsail CLI](#).



## Installazione WordPress

Lightsail fornisce modelli per applicazioni di uso comune come WordPress. Questo modello è un ottimo punto di partenza per gestire il proprio WordPress sito Web in quanto viene preinstallato con la maggior parte del software necessario. È possibile installare software aggiuntivo o personalizzare la configurazione del software utilizzando il terminale interno al browser o il proprio SSH client o tramite l'interfaccia web di WordPress amministrazione.

Amazon Lightsail collabora GoDaddy con il prodotto Pro Sites per WordPress aiutare i clienti a gestire facilmente le proprie istanze gratuitamente. I server virtuali WordPress Lightsail sono preconfigurati e ottimizzati per prestazioni e sicurezza rapide, il che semplifica l'installazione e l'operatività del WordPress sito in pochissimo tempo. I clienti che utilizzano più WordPress istanze trovano difficile e dispendioso in termini di tempo aggiornare, mantenere e gestire tutti i loro siti. Grazie a questa integrazione, puoi gestire facilmente più WordPress istanze in pochi minuti con pochi clic.

Per ulteriori informazioni WordPress sulla gestione di Lightsail dopo l'installazione, consulta la sezione [Guida introduttiva all' WordPress utilizzo dall'istanza Amazon Lightsail](#). Una volta che hai finito di personalizzare il tuo WordPress sito Web, ti consigliamo di scattare un'istantanea della tua istanza.

Un'[istantanea](#) è un modo per creare un'immagine di backup dell'istanza Lightsail. È una copia del disco di sistema e memorizza anche la configurazione originale della macchina (ovvero memoria CPU, dimensione del disco e velocità di trasferimento dati). Le istantanee possono essere utilizzate per ripristinare una configurazione nota come valida dopo una distribuzione o un aggiornamento errati.

Questa istantanea consente di ripristinare il server, se necessario, ma anche di avviare nuove istanze con le stesse personalizzazioni.

## Ripristino dopo un errore

Un singolo server Web rappresenta un unico punto di errore, quindi è necessario assicurarsi che venga eseguito il backup dei dati del sito Web. Il meccanismo di istantanea descritto in precedenza può essere utilizzato anche per questo scopo. Per eseguire il ripristino in caso di errore, è possibile ripristinare una nuova istanza dallo snapshot più recente. Per ridurre la quantità di dati che potrebbero andare persi durante un ripristino, le istantanee devono essere le più recenti possibile.

Per ridurre al minimo il rischio di perdita di dati, assicurati che le istantanee vengano scattate regolarmente. È possibile pianificare istantanee automatiche delle istanze Linux/Unix di Lightsail.

Per i passaggi, consulta [Abilitazione o disabilitazione di snapshot automatici per istanze o dischi in Amazon Lightsail](#).

AWSconsiglia di utilizzare un IP statico: un indirizzo IP pubblico fisso dedicato al tuo account Lightsail. Se devi sostituire l'istanza con un'altra, puoi riassegnare l'IP statico alla nuova istanza. In questo modo, non hai bisogno di riconfigurare eventuali sistemi esterni (come DNS i record) per puntare a un nuovo indirizzo IP ogni volta che vuoi sostituire l'istanza.

# Miglioramento delle prestazioni e ottimizzazione dei costi

Qualora l'implementazione a server singolo non sia più sufficiente a soddisfare le esigenze di crescita, potrebbe essere necessario valutare opzioni per migliorare le prestazioni del sito Web. Prima di passare a un'implementazione scalabile multi-server (discussa più avanti in questo documento), è possibile applicare una serie di efficienze in termini di prestazioni e costi. Si tratta di pratiche che dovrebbero essere seguite comunque, anche se si passa a un'architettura multi-server.

Le sezioni seguenti illustrano diverse opzioni che possono migliorare aspetti delle prestazioni e della scalabilità di un sito Web WordPress. Alcune di esse possono essere applicate a una distribuzione a server singolo, mentre altre sfruttano la scalabilità di più server. Molte di queste modifiche richiedono l'utilizzo di uno o più plug-in di WordPress. Tra le varie opzioni disponibili, [W3 Total Cache](#) è una scelta popolare che combina numerose di queste modifiche in un unico plug-in.

## Argomenti

- [Accelerazione della distribuzione di contenuti](#)
- [Caching di database](#)
- [Caching bytecode](#)

## Accelerazione della distribuzione di contenuti

Qualsiasi sito Web WordPress deve fornire contenuti sia statici che dinamici. I contenuti statici includono immagini, file JavaScript o fogli di stile. I contenuti dinamici includono tutto ciò che viene generato sul lato server utilizzando il codice PHP di WordPress, ad esempio, elementi del proprio sito generati dal database o personalizzati per ciascun visualizzatore.

Un aspetto importante dell'esperienza dell'utente finale è la latenza di rete dovuta alla distribuzione dei contenuti precedenti agli utenti di tutto il mondo. L'accelerazione della distribuzione dei contenuti precedenti migliora l'esperienza degli utenti finali, in particolare di quelli dislocati nel mondo. Questo risultato può essere ottenuto con una rete per la distribuzione di contenuti (CDN, Content Delivery Network) come Amazon CloudFront.

[Amazon CloudFront](#) è un servizio Web che offre un modo semplice ed economico per distribuire contenuti con bassa latenza e velocità di trasferimento dati elevate in diverse posizioni edge nel mondo. Le richieste dei visualizzatori vengono instradate automaticamente in una [posizione edge](#) di CloudFront adatta per ridurre la latenza. Se il contenuto può essere memorizzato nella cache

(per pochi secondi, minuti o addirittura giorni) ed è già archiviato in una particolare posizione edge, CloudFront lo distribuisce immediatamente. Se il contenuto non deve essere memorizzato nella cache, è scaduto o non si trova in quella posizione edge, CloudFront lo recupera da una o più fonti di attendibilità, indicate come origini (in questo caso, l'istanza Lightsail) nella configurazione di CloudFront. Tale recupero avviene tramite connessioni di rete ottimizzate, che accelerano la distribuzione di contenuti sul sito Web. Oltre a migliorare l'esperienza dell'utente finale, il modello discusso riduce anche il carico sui server di origine ed è potenzialmente in grado di offrire notevoli risparmi sui costi.

## Offload di contenuti statici

Sono inclusi file CSS, JavaScript e immagini, sia che si tratti di temi WordPress o di file multimediali caricati dagli amministratori dei contenuti. Tutti questi file possono essere archiviati in Amazon Simple Storage Service (Amazon S3) utilizzando un plug-in come W3 Total Cache e possono essere forniti agli utenti in modo scalabile e con disponibilità elevata. [Amazon S3](#) offre un'infrastruttura di storage dei dati economica, altamente scalabile, affidabile e a bassa latenza, accessibile tramite API REST. Amazon S3 archivia in modo ridondante gli oggetti non solo su più dispositivi, ma anche su più strutture all'interno di una regione AWS, offrendo così livelli di durata eccezionalmente elevati.

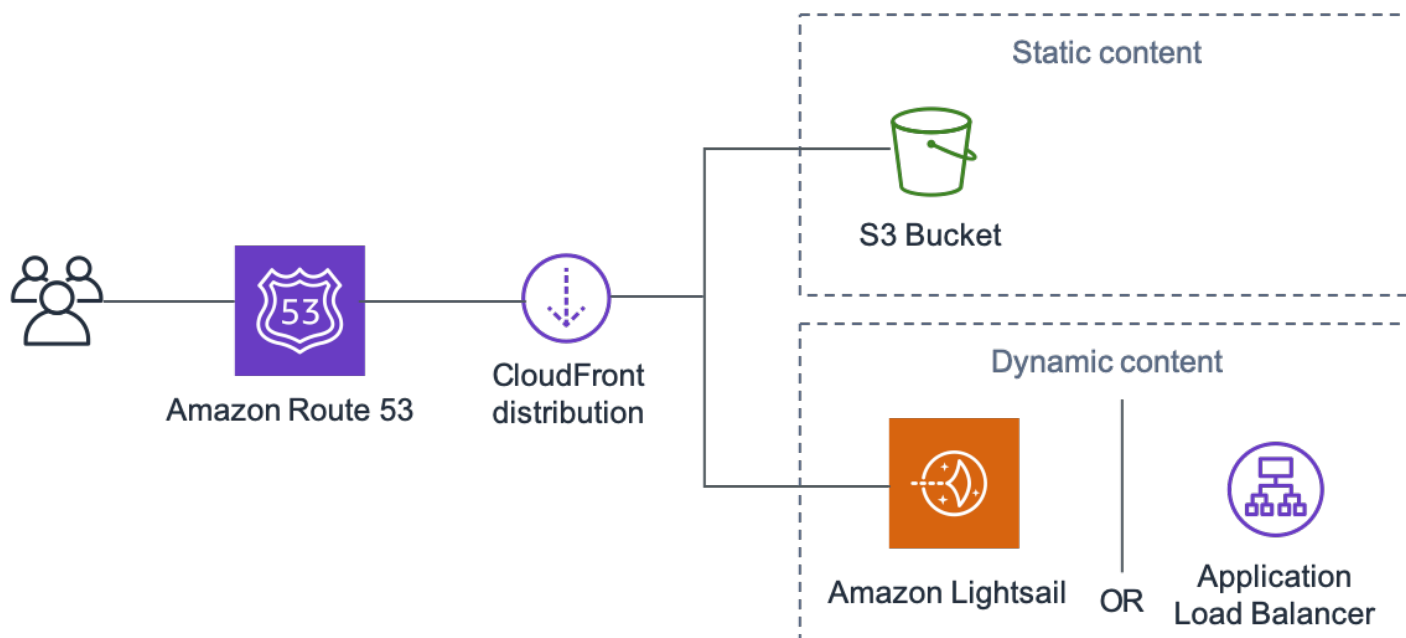
Ciò ha come effetto secondario positivo l'offloading del carico di lavoro dall'istanza Lightsail, che a sua volta consente di dedicare maggiore attenzione alla generazione dinamica di contenuti. Questo riduce il carico sul server e rappresenta un importante passo avanti verso la creazione di un'architettura senza stato (necessaria per l'implementazione del ridimensionamento automatico).

In seguito è possibile configurare Amazon S3 come origine per CloudFront per migliorare la distribuzione di tali risorse statiche agli utenti di tutto il mondo. Sebbene WordPress non sia integrato con Amazon S3 e CloudFront, sono disponibili diversi plug-in che aggiungono il supporto di questi servizi (ad esempio, W3 Total Cache).

## Contenuti dinamici

I contenuti dinamici includono l'output di script PHP WordPress lato server. I contenuti dinamici possono essere forniti anche tramite CloudFront configurando il sito Web WordPress come origine. Poiché i contenuti dinamici includono contenuti personalizzati, è necessario configurare CloudFront per l'inoltro di determinati cookie HTTP e specifiche intestazioni HTTP nell'ambito di una richiesta al server di origine personalizzato. CloudFront utilizza i valori dei cookie inoltrati come parte della chiave che identifica un oggetto univoco nella sua cache. Per ottimizzare l'efficienza del caching, è necessario configurare CloudFront in modo che inoltri solo i cookie HTTP e le intestazioni HTTP

che cambiano realmente il contenuto (non i cookie che vengono utilizzati solo sul lato client o da applicazioni di terze parti, ad esempio per l'analisi dei dati Web).



### Distribuzione dell'intero sito Web tramite Amazon CloudFront

La figura precedente include due origini: una per i contenuti statici e l'altra per i contenuti dinamici. Maggiori informazioni sull'implementazione sono disponibili nell'[Appendice A: Configurazione di CloudFront](#) e nell'[Appendice B: Installazione e configurazione di plug-in](#).

CloudFront utilizza intestazioni di controllo della cache standard per identificare se e per quanto tempo deve memorizzare nella cache risposte HTTP specifiche. Le stesse intestazioni di controllo della cache vengono utilizzate anche dai browser Web per decidere quando e per quanto tempo memorizzare i contenuti localmente nella cache, in modo da offrire all'utente finale un'esperienza ottimale (ad esempio, un file .css già scaricato non verrà scaricato nuovamente ogni volta che un visitatore non nuovo visualizza una pagina). È possibile configurare le intestazioni di controllo della cache a livello del server Web (ad esempio, tramite file .htaccess o modifiche del file httpd.conf) o installare un plug-in di WordPress (ad esempio, W3 Total Cache) per stabilire in che modo tali intestazioni vengono impostate per i contenuti statici e dinamici.

## Caching di database

Il caching di database può ridurre significativamente la latenza e aumentare il throughput per i carichi di lavoro di applicazioni con numerose operazioni in lettura come WordPress. Le prestazioni delle applicazioni vengono migliorate grazie all'accesso a bassa latenza consentito dall'archiviazione nella

memoria dei dati utilizzati di frequente (ad esempio, i risultati delle query del database che richiedono numerose operazioni di I/O). Quando una percentuale elevata delle query viene fornita dalla cache, il numero di query che devono raggiungere il database è inferiore, con conseguente riduzione dei costi associati all'esecuzione del database.

Sebbene WordPress abbia capacità di caching limitate, vari plug-in consentono di aggiungere il supporto dell'integrazione con [Memcached](#), un sistema di caching degli oggetti in memoria ampiamente diffuso. Il plug-in W3 Total Cache ne è un buon esempio.

Negli scenari più semplici, è sufficiente installare Memcached sul server Web e acquisire il risultato sotto forma di nuova snapshot. In questo caso, l'utente è responsabile delle attività amministrative associate all'esecuzione di una cache.

Un'altra opzione è sfruttare un servizio gestito come [Amazon ElastiCache](#) ed evitare questo onere operativo. ElastiCache rende semplice implementare, utilizzare e dimensionare una cache in memoria distribuita nel cloud. Informazioni su come connettersi ai nodi del cluster ElastiCache sono disponibili nella [documentazione di Amazon ElastiCache](#).

Se si utilizza Lightsail e si desidera accedere privatamente a un cluster ElastiCache nel proprio account AWS, è possibile farlo utilizzando il peering VPC. Le istruzioni su come abilitare il peering VPC sono disponibili in [Configura il peering di Amazon VPC in modo che funzioni con risorse AWS esterne ad Amazon Lightsail](#).

## Caching bytecode

Ogni volta che viene eseguito, uno script PHP viene analizzato e compilato. Usando una cache bytecode PHP, l'output della compilazione PHP viene memorizzato nella RAM in modo che lo stesso script non debba essere compilato più volte. Ciò riduce il sovraccarico relativo all'esecuzione di script PHP, con conseguente miglioramento delle prestazioni e riduzione dei requisiti della CPU.

Una cache bytecode può essere installata su qualsiasi istanza Lightsail che ospita WordPress e può ridurne notevolmente il carico. Per PHP 5.5 e versioni successive, AWS consiglia l'uso di [OPcache](#), un'estensione offerta in bundle con quella versione di PHP.

Nota: OPcache è abilitato per impostazione predefinita nel modello Bitnami WordPress Lightsail, per cui non è necessario intraprendere ulteriori azioni.

# Distribuzione elastica

Esistono molti scenari in cui un'implementazione a server singolo potrebbe non essere sufficiente per il tuo sito Web. In queste situazioni, è necessaria un'architettura scalabile multiserver.

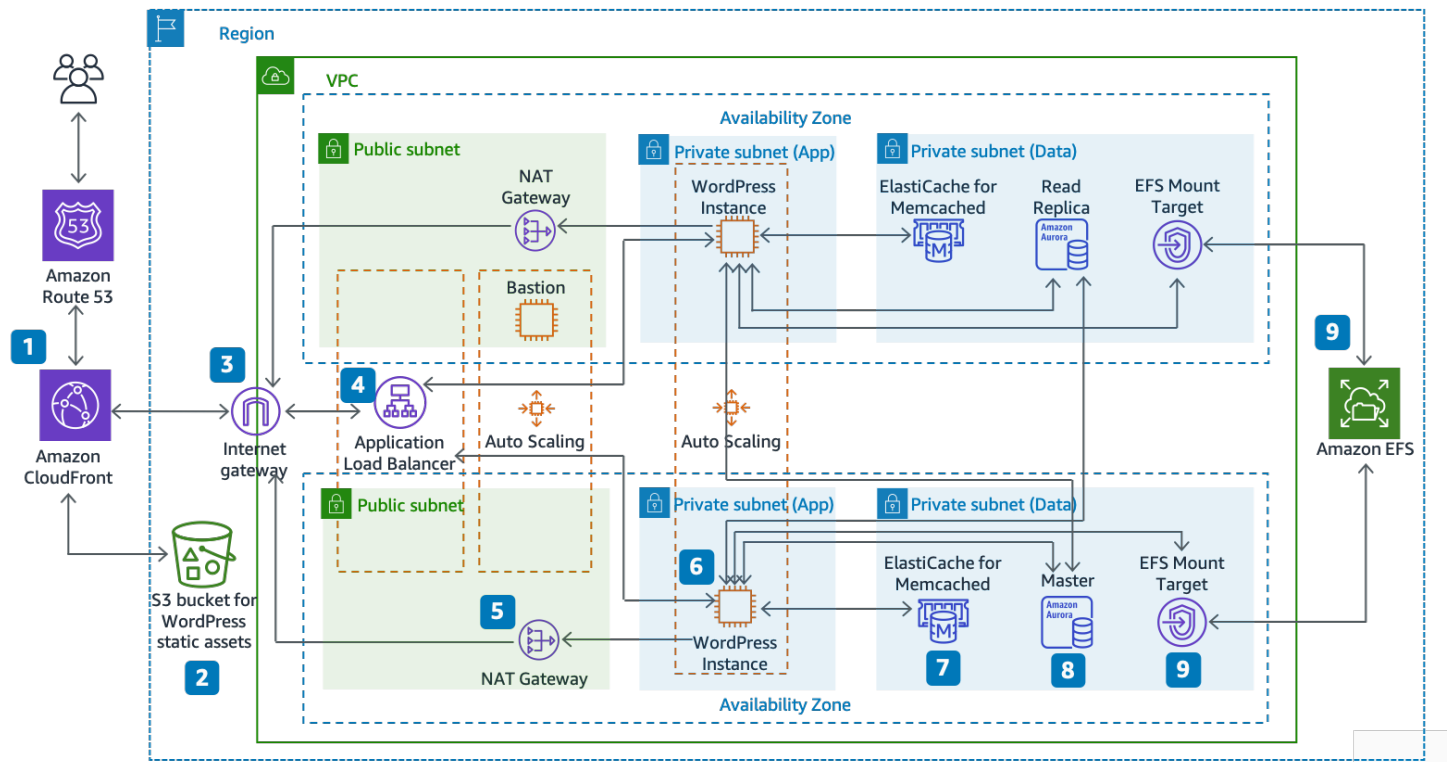
## Argomenti

- [Architettura di riferimento](#)
- [Scalabilità del livello web](#)
- [Livello stateless](#)

## Architettura di riferimento

L'[architettura di AWS riferimento Hosting WordPress on](#), disponibile su, GitHub delinea le migliori pratiche per l'implementazione WordPress AWS e include una serie di AWS CloudFormation modelli per renderti operativo rapidamente. La seguente architettura si basa su tale architettura di riferimento. Il resto di questa sezione esaminerà i motivi alla base delle scelte architettoniche.

Il based AML in the GitHub è stato modificato da Amazon Linux1 ad Amazon Linux2 nel luglio 2021. Tuttavia, i modelli di distribuzione in S3 non sono ancora stati modificati. Si consiglia di utilizzare i modelli in GitHub caso di problemi per distribuire l'architettura di riferimento con i modelli in S3.



## Architettura di riferimento per l'hosting WordPress su AWS

### Componenti dell'architettura

L'architettura di riferimento illustra un'implementazione completa delle migliori pratiche per un WordPress sito Web su AWS.

- Inizia con l'edge caching in Amazon CloudFront (1) per memorizzare i contenuti vicino agli utenti finali per una distribuzione più rapida.
- CloudFront estrae il contenuto statico da un bucket S3 (2) e il contenuto dinamico da un Application Load Balancer (4) davanti alle istanze web.
- Le istanze Web vengono eseguite in un gruppo Auto Scaling di istanze EC2 Amazon (6).
- Un ElastiCache cluster (7) memorizza nella cache i dati richiesti di frequente per velocizzare le risposte.

Un'istanza Amazon Aurora My (8) ospita il WordPress database.

- Le WordPress EC2 istanze accedono ai WordPress dati condivisi su un EFS file system Amazon tramite un EFS Mount Target (9) in ciascuna zona di disponibilità.
- Un Internet Gateway (3) consente la comunicazione tra le risorse dell'utente VPC e Internet.



- NATI gateway (5) in ciascuna zona di disponibilità consentono EC2 alle istanze in sottoreti private (app e dati) di accedere a Internet.

All'interno di Amazon VPC esistono due tipi di sottoreti: pubbliche (Public Subnet) e private (App Subnet e Data Subnet). Le risorse distribuite nelle sottoreti pubbliche riceveranno un indirizzo IP pubblico e saranno visibili pubblicamente su Internet. L'Application Load Balancer (4) e un host Bastion per l'amministrazione vengono implementati qui. Le risorse distribuite nelle sottoreti private ricevono solo un indirizzo IP privato e quindi non sono visibili pubblicamente su Internet, il che migliora la sicurezza di tali risorse. Le istanze del server WordPress Web (6), le istanze ElastiCache cluster (7), le istanze Aurora My SQL database (8) e EFSMount Targets (9) sono tutte distribuite in sottoreti private.

La parte restante di questa sezione tratta ognuna di queste considerazioni in modo più dettagliato.

## Scalabilità del livello web

Per far evolvere l'architettura a server singolo in un'architettura scalabile multiserver, è necessario utilizzare cinque componenti chiave:

- EC2Istanze Amazon
- Amazon Machine Image (AMIs)
- Sistemi di load balancer
- Scalabilità automatica
- Controlli dell'integrità

AWSfornisce un'ampia varietà di tipi di EC2 istanza, in modo da consentirti di scegliere la configurazione server migliore in termini di prestazioni e costo. In generale, il tipo di istanza ottimizzato per il calcolo (ad esempio, C4) può essere una buona scelta per un WordPress server Web. È possibile distribuire le istanze in più zone di disponibilità all'interno di una AWS regione per aumentare l'affidabilità dell'architettura complessiva.

Poiché hai il controllo completo dell'EC2istanza, puoi accedere con accesso root per installare e configurare tutti i componenti software necessari per la gestione di un WordPress sito Web. Al termine, puoi salvare la configurazione come fileAMI, che puoi utilizzare per avviare nuove istanze con tutte le personalizzazioni che hai apportato.

Per distribuire le richieste degli utenti finali su più nodi di server Web, è necessaria una soluzione di bilanciamento del carico. AWS fornisce questa funzionalità tramite [Elastic Load Balancing](#), un servizio ad alta disponibilità che distribuisce il traffico su più istanze. Poiché il tuo sito Web fornisce contenuti ai tuoi utenti tramite HTTP o HTTPS, ti consigliamo di utilizzare Application Load Balancer, un sistema di bilanciamento del carico a livello di applicazione con routing dei contenuti e la possibilità di eseguire più WordPress siti Web su domini diversi, se necessario.

Elastic Load Balancing supporta la distribuzione delle richieste su più zone di disponibilità all'interno di una AWS regione. È inoltre possibile configurare un controllo dello stato in modo che l'Application Load Balancer interrompa automaticamente l'invio di traffico a singole istanze che hanno avuto esito negativo (ad esempio, a causa di un problema hardware o di un crash del software). AWS consiglia di utilizzare la pagina di accesso dell' WordPress amministratore (`/wp-login.php`) per il controllo dello stato, poiché questa pagina conferma sia che il server Web è in esecuzione sia che il server Web è configurato per servire PHP i file correttamente.

Puoi scegliere di creare una pagina di controllo dello stato personalizzata che controlli altre risorse dipendenti, come le risorse del database e della cache. Per ulteriori informazioni, consultate la sezione [Health checks for your target group](#) nella Application Load Balancer Guide.

L'elasticità è una caratteristica chiave del AWS Cloud. Puoi avviare una maggiore capacità di elaborazione (ad esempio, server Web) quando ne hai bisogno ed eseguirla meno quando non ne hai bisogno. [Amazon EC2 Auto Scaling](#) è un AWS servizio che ti aiuta ad automatizzare questo provisioning per aumentare o ridurre la tua EC2 capacità Amazon in base a condizioni da te definite senza bisogno di interventi manuali. Puoi configurare Amazon EC2 Auto Scaling in modo che il numero di EC2 istanze che stai utilizzando aumenti senza interruzioni durante i picchi di domanda per mantenere le prestazioni e diminuisca automaticamente quando il traffico diminuisce, in modo da ridurre al minimo i costi.

Elastic Load Balancing supporta anche l'aggiunta e la rimozione dinamica EC2 degli host Amazon dalla rotazione di bilanciamento del carico. Inoltre, lo stesso Elastic Load Balancing aumenta e diminuisce dinamicamente la capacità di bilanciamento del carico per adattarsi alle richieste di traffico senza alcun intervento manuale.

## Livello stateless

Per sfruttare i vantaggi di più server Web in una configurazione di scalabilità automatica, il livello Web deve essere stateless. Un'applicazione stateless è un'applicazione che non richiede la conoscenza delle interazioni precedenti e non memorizza informazioni sulla sessione. Nel caso di WordPress,

ciò significa che tutti gli utenti finali ricevono la stessa risposta, indipendentemente dal server Web che ha elaborato la richiesta. Un'applicazione stateless può essere scalata orizzontalmente poiché qualsiasi richiesta può essere soddisfatta da una qualsiasi delle risorse di elaborazione disponibili (ovvero istanze di server Web). Quando tale capacità non è più necessaria, ogni singola risorsa può essere interrotta in modo sicuro (dopo che le attività in esecuzione sono state esaurite). Non è necessario che tali risorse siano consapevoli della presenza dei colleghi: è sufficiente un modo per distribuire loro il carico di lavoro.

Per quanto riguarda l'archiviazione dei dati della sessione utente, il WordPress core è completamente stateless perché si basa sui cookie memorizzati nel browser Web del client. L'archiviazione delle sessioni non è un problema a meno che non sia stato installato un codice personalizzato (ad esempio un WordPress plug-in) che si basa invece su sessioni native. PHP

Tuttavia, WordPress è stato originariamente progettato per funzionare su un singolo server. Di conseguenza, memorizza alcuni dati nel file system locale del server. Quando viene eseguito WordPress in una configurazione multiserver, ciò crea un problema perché c'è un'incoerenza tra i server Web. Ad esempio, se un utente carica una nuova immagine, questa viene memorizzata solo su uno dei server.

Questo dimostra perché è necessario migliorare la configurazione di WordPress esecuzione predefinita per spostare i dati importanti nell'archiviazione condivisa. L'architettura basata sulle best practice prevede un database come livello separato all'esterno del server Web e utilizza l'archiviazione condivisa per archiviare i caricamenti, i temi e i plug-in degli utenti.

## Storage condiviso (Amazon S3 e Amazon) EFS

Per impostazione predefinita, WordPress archivia i file caricati dagli utenti nel file system locale e quindi non è stateless. Pertanto, dobbiamo spostare l'WordPress installazione e tutte le personalizzazioni degli utenti (come configurazione, plug-in, temi e caricamenti generati dagli utenti) in una piattaforma di dati condivisa per ridurre il carico sui server Web e rendere il livello Web senza stato.

[Amazon Elastic File System](#) (AmazonEFS) fornisce file system di rete scalabili da utilizzare con le EC2 istanze. I EFS file system Amazon sono distribuiti su un numero illimitato di server di storage, consentendo ai file system di crescere in modo elastico e consentendo l'accesso massiccio in parallelo dalle istanze. EC2 Il design distribuito di Amazon EFS evita i colli di bottiglia e i vincoli inerenti ai file server tradizionali.

Spostando l'intera directory di WordPress installazione su un EFS file system e montandola in ciascuna EC2 istanza al momento dell'avvio, il WordPress sito e tutti i relativi dati vengono automaticamente archiviati su un file system distribuito che non dipende da alcuna EC2 istanza, rendendo il livello Web completamente privo di stato. Il vantaggio di questa architettura è che non è necessario installare plugin e temi all'avvio di ogni nuova istanza e che è possibile velocizzare notevolmente l'installazione e il ripristino delle istanze. WordPress Inoltre, è più semplice implementare le modifiche ai plugin e ai temi in WordPress, come illustrato nella sezione [Considerazioni sulla distribuzione](#) di questo documento.

Per garantire prestazioni ottimali del tuo sito Web quando viene eseguito da un EFS file system, controlla le impostazioni di configurazione consigliate per Amazon EFS e OPcache sull'[architettura AWS di riferimento per WordPress](#).

Hai anche la possibilità di scaricare tutte le risorse statiche, come immagini e JavaScript fileCSS, su un bucket S3 con CloudFront memorizzazione nella cache. Il meccanismo per eseguire questa operazione in un'architettura multiserver è esattamente lo stesso di un'architettura a server singolo, come illustrato nella sezione [Static Content](#) di questo white paper. I vantaggi sono gli stessi dell'architettura a server singolo: puoi trasferire il lavoro associato alla fornitura delle tue risorse statiche su Amazon S3 e, in tal modo CloudFront, consentire ai tuoi server Web di concentrarsi solo sulla generazione di contenuti dinamici e soddisfare più richieste utente per server Web.

## Livello dati (Amazon Aurora e Amazon) ElastiCache

Con l' WordPress installazione archiviata su un file system di rete distribuito, scalabile e condiviso e gli asset statici forniti da Amazon S3, puoi concentrare la tua attenzione sul componente di stato rimanente: il database. Come per il livello di storage, il database non deve dipendere da un singolo server, quindi non può essere ospitato su uno dei server Web. Invece, ospita il WordPress database su Amazon Aurora.

[Amazon Aurora](#) è un database relazionale SQL compatibile con My SQL e Postgre creato per il cloud che unisce le prestazioni e la disponibilità dei database commerciali di fascia alta alla semplicità e al costo ridotto dei database open source. Aurora My SQL aumenta SQL le prestazioni e la disponibilità di My grazie alla perfetta integrazione del motore database con un sistema di archiviazione distribuito creato appositamente, supportato da. SSD È tollerante ai guasti e si ripara automaticamente, replica sei copie dei dati in tre zone di disponibilità, è progettato per una disponibilità superiore al 99,99% ed esegue il backup continuo dei dati in Amazon S3. Amazon Aurora è progettato per rilevare gli arresti anomali del database e riavviarlo senza la necessità di un ripristino da arresto anomalo o di ricostruire la cache del database.

Amazon Aurora offre diversi [tipi di istanze per soddisfare diversi profili applicativi](#), tra cui istanze ottimizzate per la memoria e istanze espandibili. Per migliorare le prestazioni del database, puoi selezionare un tipo di istanza di grandi dimensioni per fornire maggiori risorse di memoria. CPU

Amazon Aurora gestisce automaticamente il failover tra l'istanza principale e [Aurora Replicas](#), in modo da consentire alle applicazioni di riprendere le operazioni database il più rapidamente possibile, senza alcun intervento amministrativo manuale. Il failover richiede in genere meno di 30 secondi.

Dopo aver creato almeno una replica Aurora, connettiti all'istanza principale utilizzando l'endpoint del cluster per consentire il failover automatico dell'applicazione in caso di guasto dell'istanza principale. È possibile creare fino a 15 repliche di lettura a bassa latenza in tre zone di disponibilità.

Man mano che il database si ridimensiona, anche la cache del database dovrà scalare. Come discusso in precedenza nella sezione [Database Caching](#), ElastiCache dispone di funzionalità per scalare la cache su più nodi di un ElastiCache cluster e su più zone di disponibilità in una regione per una maggiore disponibilità. Durante la scalabilità del ElastiCache cluster, assicurati di configurare il plug-in di caching per la connessione utilizzando l'endpoint di configurazione in modo che WordPress possa utilizzare i nuovi nodi del cluster man mano che vengono aggiunti e smettere di usare i vecchi nodi del cluster man mano che vengono rimossi. È inoltre necessario configurare i server Web per utilizzare il [ElastiCacheCluster Client PHP e aggiornarli per](#) AMI memorizzare questa modifica.

## Conclusione

AWS presenta numerose opzioni in termini di architettura per l'esecuzione di WordPress. L'opzione più semplice è l'installazione di un singolo server per siti Web con traffico ridotto. Per i siti Web più avanzati, gli amministratori del sito possono aggiungere diverse altre opzioni, ognuna delle quali rappresenta un miglioramento incrementale in termini di disponibilità e scalabilità. Gli amministratori possono selezionare le funzionalità più adatte alle loro esigenze e al loro budget.

# Collaboratori

I collaboratori di questo documento includono:

- Paul Lewis, Solutions Architect, Amazon Web Services
- Ronan Guilfoyle, Solutions Architect, Amazon Web Services
- Andreas Chatzakis, Solutions Architect Manager, Amazon Web Services
- Jibril Touzi, Technical Account Manager, Amazon Web Services
- Hakmin Kim, Migration Partner Solutions Architect, Amazon Web Services

# Revisioni del documento

Per ricevere una notifica sugli aggiornamenti del white paper, iscriversi al feed. RSS

Modifica	Descrizione	Data
<a href="#">Aggiornamento del whitepaper</a>	Aggiornato per modificare l'architettura di riferimento e AWS per il plugin. WordPress	19 ottobre 2021
<a href="#">Aggiornamento del whitepaper</a>	Aggiornato per includere nuovi approcci di implementazione e AWS per il WordPress plug-in.	30 ottobre 2019
<a href="#">Aggiornamento del whitepaper</a>	Aggiornato per chiarire la messaggistica dei prodotti Amazon Aurora.	1 febbraio 2018
<a href="#">Aggiornamento del whitepaper</a>	Aggiornato per includere AWS i servizi lanciati dalla prima pubblicazione.	1° dicembre 2017
<a href="#">Pubblicazione iniziale</a>	Pubblicato per la prima volta.	1 dicembre 2014



# Appendice A: configurazione CloudFront

Per ottenere prestazioni ed efficienza ottimali quando usi Amazon CloudFront con il tuo WordPress sito Web, è importante configurare correttamente il sito Web per i diversi tipi di contenuti offerti.

## Argomenti

- [Origini e comportamenti](#)
- [CloudFront creazione della distribuzione](#)

## Origini e comportamenti

Un'[origine](#) è una posizione in cui CloudFront invia le richieste di contenuti che distribuisce attraverso le edge location. A seconda dell'implementazione, è possibile avere una o due origini. Uno per i contenuti dinamici (l'istanza Lightsail nell'opzione di [distribuzione a server singolo o l'Application Load Balancer](#) nell'opzione di [distribuzione elastica](#)) utilizzando un'origine personalizzata. Potresti avere una seconda origine a cui CloudFront indirizzare i tuoi contenuti statici. Nell'[architettura di riferimento](#) precedente, si tratta di un bucket S3. Quando utilizzi Amazon S3 come origine per la tua distribuzione, devi utilizzare una [policy bucket](#) per rendere i contenuti accessibili al pubblico.

I [comportamenti](#) consentono di impostare regole che regolano il modo in cui memorizza i contenuti nella CloudFront cache e, a loro volta, determinano l'efficacia della cache. I comportamenti consentono di controllare il protocollo e i HTTP metodi con cui è accessibile il sito Web. Consentono inoltre di controllare se passare HTTP intestazioni, cookie o stringhe di query al backend (e, in caso affermativo, quali). I comportamenti si applicano a schemi di percorso specifici. URL

## CloudFront creazione della distribuzione

Crea una distribuzione CloudFront web seguendo la distribuzione, l'origine e il comportamento predefiniti creati automaticamente verranno utilizzati per i contenuti dinamici. Crea quattro comportamenti aggiuntivi per personalizzare ulteriormente il modo in cui vengono trattate le richieste statiche e dinamiche. La tabella seguente fornisce un riepilogo delle proprietà di configurazione per i cinque comportamenti.

Tabella 1: Riepilogo delle proprietà di configurazione per i comportamenti CloudFront

Proprietà	Statico	Dinamico (admin)	Dinamico (front-end)
Percorsi (comportamenti)	wp-content/* wp-includes/*	wp-admin/* wp-login.php	predefinito () *
Protocolli	HTTPe HTTPS	Reindirizza a HTTPS	HTTPe HTTPS
HTTPmetodi	GET, HEAD	ALL	ALL
HTTPintestazioni	NONE	ALL	Host  CloudFront-Forwarded-Proto  CloudFront-Is-Mobile-Viewer  CloudFront-Is-Tablet-Viewer  CloudFront-Is-Desktop-Viewer
Cookie	NONE	ALL	commento_*  wordpress_*  wp-impostazioni-*
Stringhe di query	YES(invalidazione)	YES	YES

Per il comportamento predefinito, AWS consiglia la seguente configurazione:

- Consenti alla politica del protocollo di origine di corrispondere a Viewer, in modo che se gli utenti si connettono a CloudFront UserHTTPS, CloudFront si connettano HTTPS anche all'origine utilizzando end-to-end la crittografia. Tieni presente che ciò richiede l'installazione di un SSL certificato affidabile sul sistema di bilanciamento del carico. Per i dettagli, consulta [Richiesta HTTPS di comunicazione tra CloudFront e l'origine personalizzata](#).

- Consenti tutti i HTTP metodi poiché le parti dinamiche del sito Web richiedono entrambe GET le POST richieste (ad esempio, POST per supportare i moduli di invio dei commenti).
- Inoltra solo i cookie che variano l' WordPress output, ad esempio `>wordpress_*wp-settings-*`, `ecomment_*`. È necessario estendere tale elenco se sono stati installati plug-in che dipendono da altri cookie non presenti nell'elenco.
- Inoltra solo le HTTP intestazioni che influiscono sull'output di WordPress, ad esempio, `Host`, `CloudFront-Forwarded-Proto`, `CloudFront-is-Desktop-Viewer`, `CloudFront-is-Mobile-Viewer`, e: `CloudFront-is-Tablet-Viewer`
  - `Host` consente di ospitare più WordPress siti Web sulla stessa origine.
  - `CloudFront-Forwarded-Proto` consente di memorizzare nella cache diverse versioni delle pagine a seconda che vi si acceda tramite HTTP o HTTPS.
  - `CloudFront-is-Desktop-Viewer`, `CloudFront-is-Mobile-Viewer`, `CloudFront-is-Tablet-Viewer` consentono di personalizzare l'output dei temi in base al tipo di dispositivo dell'utente finale.
- Inoltra tutte le stringhe di query alla cache in base ai loro valori perché, WordPress basandosi su questi, possono essere utilizzate anche per invalidare gli oggetti memorizzati nella cache.

Se vuoi servire il tuo sito web con un nome di dominio personalizzato (ovvero `no*.cloudfront.net`), inserisci quello appropriato in Nomi di dominio alternativi URIs nelle Impostazioni di distribuzione. In questo caso, è necessario anche un SSL certificato per il nome di dominio personalizzato. È possibile [richiedere](#) SSL certificati tramite AWS Certificate Manager e configurarli in base a una CloudFront distribuzione.

Ora, create altri due comportamenti di cache per i contenuti dinamici: uno per la pagina di accesso (path pattern: `wp-login.php`) e uno per la dashboard di amministrazione (path pattern: `wp-admin/*`). Questi due comportamenti hanno le stesse impostazioni, come segue:

- Applica una politica del protocollo Viewer di Only. HTTPS
- Consenti tutti i HTTP metodi.
- Cache basata su tutte le HTTP intestazioni.
- Inoltra tutti i cookie.
- Forward e cache basate su tutte le stringhe di query.

Il motivo alla base di questa configurazione è che questa sezione del sito Web è altamente personalizzata e in genere ha solo pochi utenti, quindi l'efficienza della memorizzazione nella cache

non è una preoccupazione primaria. L'obiettivo è mantenere la configurazione semplice per garantire la massima compatibilità con tutti i plugin installati passando tutti i cookie e le intestazioni all'origine.

Per impostazione predefinita, WordPress archivia tutto localmente sul server Web, che è lo storage a blocchi (AmazonEBS) per la [distribuzione a server singolo](#) e lo storage di file (AmazonEFS) per la [distribuzione elastica](#). Oltre a ridurre i costi di storage e trasferimento dei dati, lo spostamento di asset statici su Amazon S3 offre scalabilità, disponibilità dei dati, sicurezza e prestazioni. Esistono diversi plug-in che semplificano lo spostamento di contenuti statici su Amazon S3; uno di questi è [W3 Total Cache](#), anch'esso trattato [nell'Appendice B](#): Installazione e configurazione dei plugin.

## Appendice B: Dati di configurazione statici dei contenuti

Per impostazione predefinita, WordPress archivia tutto localmente sul server Web, che è lo storage a blocchi (AmazonEBS) per la [distribuzione a server singolo](#) e lo storage di file (AmazonEFS) per la [distribuzione elastica](#). Oltre a ridurre i costi di storage e trasferimento dei dati, lo spostamento di asset statici su Amazon S3 offre scalabilità, disponibilità dei dati, sicurezza e prestazioni.

In questo esempio, il plug-in W3 Total Cache (W3TC) viene utilizzato per archiviare risorse statiche su Amazon S3. Tuttavia, sono disponibili altri plugin con funzionalità simili. Se si desidera utilizzare un'alternativa, è possibile modificare di conseguenza i passaggi seguenti. I passaggi si riferiscono solo alle funzionalità o alle impostazioni pertinenti a questo esempio. Una descrizione dettagliata di tutte le impostazioni non viene riportata in questo documento. Per ulteriori informazioni, consultate la [pagina del plugin W3 Total Cache](#) su [wordpress.org](https://wordpress.org).

### Creazione di utenti

È necessario creare un utente per il WordPress plug-in per archiviare risorse statiche in Amazon S3. Per la procedura, consulta [Creazione di un utente nel tuo AWS account](#).

Nota: i ruoli forniscono un modo migliore per gestire l'accesso alle AWS risorse, ma al momento in cui scriviamo, il plugin W3 Total Cache non supporta [i ruoli](#).

Prendi nota delle credenziali di sicurezza dell'utente e conservale in modo sicuro: ti serviranno queste credenziali in un secondo momento.

### Creazione di bucket Amazon S3

1. Innanzitutto, crea un bucket Amazon S3 nella AWS Regione di tua scelta. Per i passaggi, consulta [Creazione di un bucket](#). Abilita l'hosting di siti Web statici per il bucket seguendo il [Tutorial: Configurazione di un sito Web statico su Amazon S3](#).
2. Crea una policy per fornire all'utente creato in precedenza l'accesso al bucket S3 specificato e allegala all'utente. Per i passaggi per creare la seguente politica, consulta [Gestione delle](#) politiche.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  

```

```
{
    "Sid": "Stmt1389783689000",
    "Effect": "Allow",
    "Principal": "*",
    "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:GetObjectAcl",
        "s3:ListBucket",
        "s3:PutObject",
        "s3:PutObjectAcl"
    ],
    "Resource": [
        "arn:aws:s3:::wp-demo",
        "arn:aws:s3:::wp-demo/*"
    ]
}
```

3. Installa e attiva il plugin W3TC dal pannello di amministrazione. WordPress
4. Vai alla sezione Impostazioni generali della configurazione del plugin e assicurati che sia Browser Cache che siano abilitati. CDN
5. Dall'elenco a discesa nella CDN configurazione, scegli Origin Push: Amazon CloudFront (questa opzione ha Amazon S3 come origine).
6. Vai alla sezione Browser Cache della configurazione del plugin e abilita le intestazioni expires, cache control e entity tag (). ETag
7. Attiva anche l'opzione Impedisci la memorizzazione nella cache degli oggetti dopo la modifica delle impostazioni in modo che una nuova stringa di query venga generata e aggiunta agli oggetti ogni volta che vengono modificate le impostazioni.
8. Vai alla CDN sezione della configurazione del plugin e inserisci le credenziali di sicurezza dell'utente che hai creato in precedenza, oltre al nome del bucket S3.
9. Se stai servendo il tuo sito Web tramite CloudFront URL, inserisci il nome del dominio di distribuzione nella casella pertinente. Altrimenti, inseriscine uno o più CNAMEs per i tuoi nomi di dominio personalizzati.
10. Infine, esporta la libreria multimediale e carica wp-includes, file del tema e file personalizzati su Amazon S3 utilizzando il plug-in W3TC. Queste funzioni di caricamento sono disponibili nella sezione Generale della pagina di configurazione. CDN

## Creazione di origini statiche

Ora che i file statici sono archiviati su Amazon S3, torna alla CloudFront configurazione nella CloudFront console e configura Amazon S3 come origine per i contenuti statici. A tale scopo, aggiungi una seconda origine che punti al bucket S3 che hai creato a tale scopo. Quindi crea altri due comportamenti di cache, uno per ciascuna delle due cartelle (`wp-content/wp-includes`) che dovrebbero utilizzare l'origine S3 anziché l'origine predefinita per i contenuti dinamici. Configura entrambi nello stesso modo:

- Serve solo HTTP GET le richieste.
- Amazon S3 non modifica l'output in base a cookie o HTTP intestazioni, quindi puoi migliorare l'efficienza della memorizzazione nella cache evitando di inoltrarli all'origine tramite CloudFront
- Nonostante questi comportamenti servano solo a contenuti statici (che non accettano parametri), inoltrerai le stringhe di query all'origine. In questo modo è possibile utilizzare le stringhe di query come identificatori di versione per invalidare istantaneamente, ad esempio, i file più vecchi CSS quando si distribuiscono nuove versioni. Per ulteriori informazioni, consulta l'[Amazon CloudFront Developer Guide](#).

### Note

Dopo aver aggiunto i comportamenti di origine statica alla tua CloudFront distribuzione, controlla l'ordine per garantire che i comportamenti `wp-admin/*` e `wp-login.php` abbiano una precedenza maggiore rispetto ai comportamenti per i contenuti statici. Altrimenti, potresti notare un comportamento strano quando accedi al pannello di amministrazione.

## Appendice C: Backup e ripristino

Il ripristino dopo un guasto in AWS è più rapido e semplice rispetto agli ambienti di hosting tradizionali. Ad esempio, è possibile avviare un'istanza sostitutiva in pochi minuti dopo un guasto hardware oppure utilizzare il failover automatico in molti dei nostri servizi gestiti per annullare l'impatto di un riavvio dovuto a manutenzione ordinaria.

Tuttavia, per il ripristino corretto dei dati è comunque necessario eseguirne il backup. Per ripristinare la disponibilità di un sito Web WordPress, è necessario avere a disposizione i seguenti componenti:

- Installazione e configurazione del sistema operativo (OS) e dei servizi (Apache, MySQL e così via)
- Codice e configurazione dell'applicazione WordPress
- Temi e plug-in di WordPress
- Caricamenti (ad esempio, file multimediali per i post)
- Contenuti del database (post, commenti e così via)

AWS offre vari metodi per il backup e il ripristino dei dati e delle risorse delle applicazioni Web.

In una sezione precedente di questo white paper si è discusso dell'uso delle snapshot di Lightsail per proteggere tutti i dati archiviati nello storage locale dell'istanza. Se il sito Web WordPress viene eseguito solo dall'istanza Lightsail, le comuni snapshot di Lightsail dovrebbero essere sufficienti per ripristinarlo nella sua interezza. Tuttavia, l'esecuzione del ripristino da una snapshot comporterà la perdita delle modifiche applicate al sito Web dall'ultima snapshot.

In una distribuzione multi-server, è necessario eseguire il backup di ciascuno dei componenti discussi in precedenza utilizzando meccanismi diversi. Ogni componente può avere requisiti diversi in termini di frequenza di backup; ad esempio, l'installazione e la configurazione del sistema operativo e di WordPress variano con una frequenza decisamente minore rispetto ai contenuti generati dall'utente, pertanto possono essere sottoposti a backup meno frequentemente senza comportare la perdita di dati in caso di ripristino.

Per eseguire il backup dell'installazione e della configurazione del sistema operativo e dei servizi, nonché del codice e della configurazione dell'applicazione WordPress, è possibile creare un'AMI di un'istanza EC2 configurata correttamente. Le AMI hanno una duplice funzione: possono essere utilizzate come backup dello stato dell'istanza e fungere da modello durante l'avvio nuove istanze.



Per eseguire il backup del codice e della configurazione dell'applicazione WordPress, è necessario utilizzare sia le AMI che i backup di Aurora.

Per eseguire il backup dei temi e dei plug-in di WordPress installati sul proprio sito Web, è necessario eseguire il backup del bucket Amazon S3 o del file system Amazon EFS in cui sono archiviati.

- Per i temi e i plug-in archiviati in un bucket S3 è possibile abilitare la [replica tra regioni](#), in modo che tutti gli oggetti caricati nel bucket primario vengano automaticamente replicati nel bucket di backup in un'altra regione AWS. La replica tra regioni richiede che il [controllo delle versioni](#) sia abilitato sui bucket di origine e destinazione, il che fornisce un ulteriore livello di protezione e consente di ripristinare una versione precedente di un determinato oggetto nel bucket.
- Per i temi e i plug-in archiviati in un file system EFS, è possibile creare un AWS Data Pipeline per copiare i dati dal file system EFS di produzione a un altro file system EFS, come descritto nella sezione [Eseguire il backup dei file system Amazon EFS](#) della documentazione. È inoltre possibile eseguire il backup di un file system EFS utilizzando l'applicazione di backup desiderata.
- Per eseguire il backup dei caricamenti degli utenti, è necessario seguire la procedura per il backup dei temi e dei plug-in di WordPress descritta in precedenza.
- Per eseguire il backup del contenuto del database è necessario effettuare il [backup tramite Aurora](#). Aurora esegue automaticamente il backup del volume del cluster e conserva i dati di ripristino per l'intero tempo di conservazione dei backup. I backup di Aurora sono continui e incrementali, pertanto è possibile eseguire rapidamente il ripristino a un momento qualsiasi del tempo di conservazione dei backup. Durante la scrittura dei dati di backup, non si verifica alcun impatto sulle prestazioni o interruzione del funzionamento del servizio del database. È possibile impostare il tempo di conservazione del backup da 1 a 35 giorni. È inoltre possibile creare [snapshot manuali del database](#), che vengono mantenute finché non vengono eliminate. Le snapshot manuali del database sono utili per i backup e l'archiviazione a lungo termine.

## Appendice D: Implementazione di nuovi plug-in e temi

Pochi siti Web rimangono statici. Nella maggior parte dei casi, verranno aggiunti periodicamente temi e plug-in di WordPress disponibili pubblicamente o verrà eseguito l'aggiornamento a una versione più recente di WordPress. In altri casi, verranno sviluppati da zero temi e plug-in personalizzati.

Ogni volta che si apporta una modifica strutturale alla propria installazione di WordPress si corre il rischio di introdurre problemi imprevisti. Effettuare almeno un backup del codice, della configurazione e del database dell'applicazione prima di apportare qualsiasi modifica significativa (come installare un nuovo plug-in). Per i siti Web di valore aziendale o di altro tipo, testare prima tali modifiche in un ambiente di gestione separato. AWS consente di replicare la configurazione dell'ambiente di produzione e di eseguire l'intero processo di implementazione in modo semplice e sicuro. Dopo aver terminato i test, è possibile semplicemente eliminare l'ambiente di test e smettere di pagare per tali risorse. In una sezione successiva di questo white paper verranno prese in esame alcune considerazioni specifiche di WordPress.

Alcuni plug-in scrivono informazioni di configurazione nella tabella del database `wp_options` (o introducono modifiche allo schema del database), mentre altri creano file di configurazione nella directory di installazione di WordPress. Dal momento che il database e lo storage sono stati trasferiti su piattaforme condivise, queste modifiche diventano immediatamente disponibili per tutte le istanze in esecuzione senza ulteriori interventi da parte dell'utente.

Se vengono implementati nuovi temi in WordPress, potrebbero essere necessarie ulteriori operazioni. Se si utilizza Amazon EFS solo per archiviare tutti i file di installazione di WordPress, i nuovi temi diventeranno immediatamente disponibili per tutte le istanze in esecuzione. Se invece si scaricano contenuti statici su Amazon S3, sarà necessario elaborarne una copia nella posizione del bucket corretta. Plug-in come W3 Total Cache consentono di avviare manualmente tale operazione. In alternativa, è possibile automatizzare questo passaggio all'interno di un processo di creazione.

Poiché le risorse dei temi possono essere memorizzate nella cache su CloudFront e sul browser, è necessario invalidare le versioni precedenti quando si implementano modifiche. A tale scopo, è possibile includere una sorta di identificatore di versione nell'oggetto, che può essere una stringa di query con un indicatore di data e ora o una stringa casuale. Se si utilizza il plug-in W3 Total Cache, è possibile aggiornare una stringa di query che viene aggiunta agli URL dei file multimediali.

## Note

I clienti sono responsabili della propria valutazione indipendente delle informazioni contenute nel presente documento. Questo documento: (a) è solo a scopo informativo, (b) rappresenta le offerte e le pratiche attuali di AWS prodotti, che sono soggette a modifiche senza preavviso, e (c) non crea alcun impegno o assicurazione da parte dei suoi affiliati, AWS fornitori o licenzianti. AWSi prodotti o i servizi sono forniti «così come sono» senza garanzie, dichiarazioni o condizioni di alcun tipo, esplicite o implicite. Le responsabilità di nei AWS confronti dei propri clienti sono definite dai AWS contratti e il presente documento non costituisce parte né modifica qualsivoglia contratto tra AWS e i suoi clienti.

© 2023, Amazon Web Services, Inc. o società affiliate. Tutti i diritti riservati.

# AWS Glossario

Per la AWS terminologia più recente, consultate il [AWS glossario](#) nella sezione Reference. Glossario AWS

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.