8 errori poco noti in un sito WordPress

Nonostante il massiccio impiego di WordPress per la costruzione di siti web, molte volte questi non vengono curati nei dettagli, vuoi per il budget, vuoi perché è difficile stare al passo con tutte le tematiche e le insidie che rendono Internet quella fantastica onda anomala che vogliamo *surfare ad ogni costo* (cit.). Il mantra di plugin e buone pratiche riguardanti SEO, sicurezza e user experience viene ripetuto di articolo in articolo e avrete notato che dopo averne letti tre o quattro gli argomenti cominciano a ripetersi. Eppure esistono altri aspetti che vale la pena approfondire, attinenti e importanti ma meno noti. In questo articolo affronteremo quelli riguardanti alcune pratiche diffuse.

Mostrare gli avatar generati

Se il vostro blog consente la scrittura di commenti, è possibile che abbiate attivato un tipo di avatar “generato”: Gravatar, Identicon, Monsterid, Wavatar o Retro. Si tratta di avatar associati a o generati partendo da un indirizzo email, dimodoché un utente che commenti più volte (inserendo sempre lo stesso indirizzo email) possa essere riconoscibile a colpo d’occhio pur non avendo caricato alcun avatar e senza doverne pubblicare l’indirizzo email. Ebbene, partendo proprio da tali immagini, spammer e malintenzionati di ogni genere possono risalire agli indirizzi email ad esse associati! Meglio quindi disattivare questi avatar, sicuramente accattivanti ma decisamente poco sicuri, oppure selezionare il classico Mystery Man.

Usare link che si aprono in nuova finestra senza metterli in sicurezza

Ebbene sì: un link con target (es. target="_blank") è considerato un potenziale problema di sicurezza! Infatti, la pagina di destinazione avrà accesso alla pagina di origine tramite l’oggetto window.opener tanto quanto una qualsiasi popup aperta con window.open(), potendo quindi ad esempio sostituire la pagina di origine con una di phishing, oppure eseguire codice javascript malevolo. Le possibili soluzioni sono:

- rimuovere target dai link, lasciando che siano gli utenti a scegliere se aprire il link in una nuova finestra usando comandi come “CTRL+Click”, o come il “Click centrale” del mouse, o ancora come “Click destro > Apri in un’altra scheda”.
- aggiungere rel="noopener noreferrer" a tutti i link con target="_blank"
- dopo aver usato window.open() resettare la proprietà opener, così:script
- Se proprio non si riesce a gestire manualmente i link, vuoi perchè son troppi, vuoi perchè non si ha la possibilità di modificarli facilmente, è possibile installare il plugin WP External Links, che si occupa tra le altre cose di “correggere” in automatico i link (ma non i window.open). Ricordate comunque che è sempre bene installare il minor numero di plugin possibili per evitare di esporre il sito a troppe vulnerabilità e a caricarlo di troppe operazioni.

Non impostare gli header HTTP di sicurezza

Nel file .htaccess di Apache, tra i tag si possono aggiungere alcune istruzioni per aumentare il livello di sicurezza del sito, sostituendole ai puntini nel codice seguente:

Le modalità con le quali applicare queste istruzioni però sono talvolta controverse e vanno in ogni caso valutate una ad una. Qui di seguito troverete riassunte le implementazioni più comuni, ma è fortemente consigliata la lettura di questo approfondimento, che contiene tra le altre cose la sintassi per applicare gli header in webserver diversi da Apache.

X-XSS-Protection "1; mode=block"
In Chrome e Internet Explorer blocca quegli attacchi XSS che inviano codice malevolo come parte della richiesta http.

Content-Security-Policy: "..."
Versione avanzata della precedente, questa direttiva è molto utile per mitigare gli attacchi XSS, ma è davvero complicato gestire gli effetti dei parametri che verranno inseriti al posto dei puntini. Per questo motivo, purtroppo, molti scelgono di non usarla.

Strict-Transport-Security "max-age=31536000; includeSubdomains;”
Se nel vostro sito è raggiungibile sia da http che da https, la richiesta http potrebbe essere intercettata, anche in caso di immediato redirect verso https. Questo header dice al browser di utilizzare solamente https, evitando questa possibilità.

X-Frame-Options "deny"
Impedisce l’inclusione della pagina sia nei frame che negli iframe.

X-Content-Type-Options nosniff
Impedisce al browser di sovrascrivere a propria discrezione i mime-type degli elementi inclusi in pagina, evitando così possibili attacchi XSS.

Non attivare le notifiche per i commenti

Chi abilita i commenti di WordPress vuole che i visitatori possano dare dei feedback ed esprimere la loro opinione. Per costruire un dialogo e mantenere alto il coinvolgimento non c’è niente di meglio che offrire ai visitatori la possibilità di farsi inviare una email di notifica quando qualcuno scrive un commento ad un determinato post. Incredibilmente, WordPress non offre nativamente questa funzionalità: possiamo però implementarla installando un plugin come Subscribe to Comments Reloaded.

Nascondere il form di ricerca

Uno dei problemi più immediati che si affrontano creando un layout mobile è come gestire il poco spazio a disposizione. Ad esempio, nella quasi totalità dei casi il menù viene nascosto in un layer che viene mostrato alla pressione del classico tasto “hamburger menu”. Quando poi ci si trova a dover disporre il form di ricerca interna al sito, si tende a ripetere la stessa scelta, sostituendo però l’icona del tasto con una “magnifying glass”. Talvolta, il tasto “hamburger menù” rivela addirittura menù e form di ricerca nello stesso layer.

Queste soluzioni però non sono il massimo della user experience: si sta infatti sacrificando l’immediatezza di una funzionalità primaria del vostro sito. Pensateci: qual è il motivo che più spesso vi porta ad estrarre lo smartphone dalla tasca? Voler cercare qualcosa. È importante quindi che il campo “cerca” sia sempre visibile e cliccabile, in modo che ci si possa immediatamente scrivere dentro. La scelta più comune è quella di metterlo a tutta larghezza nell’header. Se credete, potete visualizzarlo più stretto, per poi allargarlo quando viene cliccato. Potete pensare anche ad altre soluzioni, ma la buona norma rimane sempre quella: il form di ricerca dev’essere sempre visibile e cliccandoci sopra dev’essere subito possibile cominciare a scrivere.

Non usare PHP 7

Il solo fatto di passare da PHP 5 a PHP 7 porta grandi vantaggi ad ogni server. Infatti, PHP è stato ottimizzato sensibilmente con il risultato che usandolo WordPress sarà più sicuro e richiederà meno risorse per funzionare. Infatti, nonostante i requisiti minimi di WordPress siano PHP 5.2.4+ e MySQL 5.0+, quelli raccomandati sono PHP 7+ e MySQL 5.6+ (oppure MariaDB 10.0+) visto che rispettandoli otterrete probabilmente un sito più veloce ed il carico di lavoro del vostro server sarà alleviato.

Non sono tutte rose e fiori però: PHP 7 non è pienamente retrocompatibile e le seppur poche incompatibilità rischiano di bloccare completamente il funzionamento del sito. Ecco quindi che se possiamo tendenzialmente contare sulla compatibilità del core di WordPress non possiamo fare altrettanto per temi e plugin. Il plugin PHP Compatibility Checker può dare a priori un'idea di quali problemi potrebbero verificarsi passando alla nuova versione, ma purtroppo non è affidabile al 100%.

Come possiamo procedere, quindi? Innanzitutto, contattate l’assistenza per sapere quale versione di PHP 7 può essere attivata nel vostro hosting. Poi clonate il sito in un altro ambiente con installata la stessa versione di PHP indicata dall’assistenza (potrebbero comunque esserci delle differenze di configurazione, ma già è un buon punto di partenza). Se amate il rischio, potete invece eseguire le prossime istruzioni direttamente dal sito in produzione anziché dal sito clonato: c’è infatti un’alta probabilità che il processo sia reversibile senza danni, ma vi consiglio comunque di non fidarvi e di fare in ogni caso un backup completo prima di procedere.

Passate quindi a PHP 7. Per farlo, potete chiedere all'assistenza del vostro hosting di cambiare versione di PHP, oppure se usate Plesk, di darvi la possibilità di farlo voi stessi. Se il sito è incompatibile, mostrerà pagina bianca (errore 500), amministrazione compresa. Se otterrete pagina bianca, potete ritornare immediatamente alla versione di PHP precedente e decidere di restare su PHP 5. Oppure potete decidere di risolvere i problemi. A volte guardare i log del server potrebbe non essere molto utile: in quel caso vale la pena provare ad impostare temporaneamente l'opzione PHP "display_errors" su "On" per vedere gli errori direttamente in pagina (pratica poco sicura e sconsigliabile in ambiente di produzione: quando avete finito ricordate di reimpostarla su "Off"). Infine, se ancora non siete riusciti a capire dove sta il problema, potete provare a disattivare tema (selezionandone uno di default) e plugin, poi attivare PHP 7, e infine attivare tema e plugin uno alla volta: WP impedirà l'attivazione del tema o plugin incompatibile, consentendone la facile individuazione.

Non verificare le *disposable* email

Lo sapevate che esistono gli indirizzi email "usa e getta"? Migliaia di servizi come 10minutemail.com li forniscono gratuitamente "per combattere lo spam", in quanto dopo alcuni minuti si autodistruggono. Vengono spesso usate per ottenere omaggi come ebook, coupon ed audiolibri, o per registrarsi a qualche sito fugace dimenticato nella propria cronologia. L'unico a offrire un servizio valido sembra essere block-disposable-email.com, in grado di identificare circa 14.000 domini di email temporanee, a pagamento sopra i 200 utilizzi al mese. Gratuitamente, invece, si possono trovare altri plugin, ma le liste ai quali si appoggiano come questo o quest'altro ad oggi contengono soltanto 1300-1500 domini.

Le soluzioni viste finora proteggono la registrazione utenti e l'inserimento commenti dalle fake email: come fare invece per proteggere Contact Form 7, Gravity Form e più in generale le nostre form? Purtroppo non esiste una soluzione già pronta: sarà necessario quindi interpellare uno sviluppatore che crei un codice PHP ove possibile, JavaScript altrimenti, che vada a confrontare il dominio delle email raccolte con le liste di cui sopra. Ovviamente il controllo va fatto al momento dell'inserimento dell'email da parte dell'utente, per evitare di sporcare il database.

Nel caso di una soluzione sviluppata *ad hoc*, potreste cogliere l'occasione per verificare anche che l'email non sia un alias: Gmail ad esempio accetta automaticamente gli indirizzi col segno "+" come alias. Se cioè il vostro indirizzo email fosse mioindirizzo@gmail.com potreste invece usare mioindirizzo+alias@gmail.com oppure mioindirizzo+test@gmail.com e così via.

Non limitare l'accesso alle REST API e a xmlrpc.php

Ma come, le REST API sono state appena introdotte e già bisogna limitarne l'accesso? E cos'è questo file xmlrpc.php? Andiamo con ordine.

Le REST API sono una feature che prima di WordPress 4.7 era disponibile nella sua completezza

soltanto grazie a questo plugin. La funzionalità è stata integrata con molta prudenza e poco per volta ed è ormai considerata matura. Ciononostante, se non la state usando è inutile lasciarla attiva: la prudenza consiglierebbe di disattivarla con qualche riga di codice nel file config.php dato che è all'ordine del giorno trovare e correggere bug di WordPress sempre nuovi. Ma dato che un giorno costituiranno parte integrante di funzionalità future dell'area di amministrazione, WordPress consiglia invece di impedire l'utilizzo delle REST API agli utenti non loggati. Tuttavia, volendo essere paranoici, anche il controllo dei privilegi dell'utente potrebbe essere soggetto a bug, e le soluzioni proposte da alcuni dei più famosi plugin di sicurezza (di seguito quelle di iThemes Security e Wordfence) ne tengono probabilmente conto.

Il file xmlrpc.php invece consente ad app e applicazioni di gestire il sito da remoto, e consente quindi di loggarsi tanto quanto il più noto file wp-login.php usato da utenti umani. Mentre però quest'ultimo viene spesso gestito dai plugin di sicurezza, il primo viene talvolta ignorato, nonostante gli attacchi attuati verso l'uno vengano attuati in egual quantità anche verso l'altro, e nonostante un attacco a xmlrpc.php sia più performante. Se dunque il vostro plugin di sicurezza non avesse alcuna opzione per applicare una limitazione all'accesso di questo file, potete facilmente impostarla voi a mano aggiungendo le seguenti righe di codice al file .htaccess:

Con questo codice gli attacchi verranno bloccati, ma il server dovrà comunque gestire le chiamate. Per fare in modo che queste non arrivino proprio (alleggerendo il carico di lavoro del server) dovrete rivolgervi ad un professionista in sistemi informatici.

Conclusioni

Siamo portati a pensare che se una funzionalità è stata inserita in un software allora gli sviluppatori ne avranno già valutato tutti gli effetti. Ma Internet offre insidie e opportunità dove meno ce l'aspettiamo: le funzionalità fornite dai software non sono sempre innocue come sembrano, e non sempre le soluzioni pensate da altri sono quelle giuste. Il dibattito è aperto: è importante formarsi un'opinione e agire di conseguenza. Cercare di implementare le soluzioni col minor numero di plugin possibile, integrandole invece se possibile nei child theme, è probabilmente una buona strategia: ogni plugin infatti aumenta la vulnerabilità e la richiesta di risorse del vostro sito.