Web Content Management Systems: an analysis of forensic investigatory challenges

ABSTRACT

With an increase in the creation and maintenance of personal websites, web content management systems are now frequently utilised. Such systems offer a low cost and simple solution for those seeking to develop an online presence, and subsequently, a platform from which reported defamatory content, abuse and copyright infringement has been witnessed. This article provides an introductory forensic analysis of the 3 current most popular web content management systems available, Wordpress, Drupal and Joomla!. Test platforms have been created and their site structures have been examined to provide guidance for forensic practitioners facing investigations of this type. Results document available metadata for establishing site ownership, user interactions and stored content following analysis of artefacts including Wordpress's *wp_users, and wp_comments* tables, Drupal's 'watchdog' records, and Joomla!'s *_users, and _content* tables. Finally, investigatory limitations documenting the difficulties of investigating WCMS usage are noted, and analysis recommendations are offered.

KEYWORDS: Forensic science; digital forensics; Internet; web content management systems; crime; investigation.

Web Content Management Systems (WCMS) are systems designed to house and coordinate digital content online (1). Typically, these systems maintain two fundamental components, a front-end web interface and back-end content management facility, often database driven (of which types varying, depending on the platform). WCMS are an all-in-one, low cost and often non-technical service for an individual to tailor and build their own website or blog. As of 2016, statistics indicate that the three most popular WCMS available are Wordpress (2), Joomla! (3) and Drupal (4), with market shares of 60%, 6.5% and 4.6% respectively (and collectively 71.1%) (5). Wordpress (6; 7) reports that in October 2016, almost 24 billion pages hosted by their services had been viewed, with over 65 million posts made to Wordpress sites and over 11 million new pages created.

An increase in WCMS pages coincides with an emerging blogging culture, with around 30 million bloggers in the United States (US) alone (8). In the United Kingdom (UK), 1.25% of all Internet browsing equates to blog and personal website visits. With increasing popularity comes a potentially greater chance of abuse where self-hosted and personal website content may fall foul of many legal requirements, ranging from copyright infringement to offences against a person. Such instances include blogger arrests for exposing human rights violations, reported in 2008 (9). Since then, WCMS and personal sites have been linked to disputes around the world including the advertisement of professional assassins (10), the making of libellous comments (11), the making of grossly offensive (12) or political (13; 14) statements and extremism (15). In turn, WCMS are somewhat frequently exposed to targeted cyberat-tacks, where in 2014, a reported 12 million Drupal sites were potentially compromised (16). In 2016, it was reported that the Panamanian law, Mossack Fonseca data leak was attributed to an outdated version of Wordpress being utilised (17).

WCMS provide a platform for individuals to post and express views and content which may not always sit within the bounds of existing legislation. Those whose actions overstep into illegality may find their site subject to investigation, requiring an understanding of the underlying functions of WCMS, implementation methods and strategies for evidence acquisition and interpretation. This article provides an examination of the three most popular WCMS, namely Wordpress, Joomla! and Drupal, with a focus on establishing procedures for investigating WCMS which have been utilised for, or involved in content-based offences. This includes cases where a WCMS platform is both a victim (for example, subject to harassing comments) and an offender (to host and distribute illegal content). Available data for identifying suspect content and attributing this to an individual will be examined and technical issues are discussed.

Regulation

With over 3.5 billion internet users worldwide (18) and the majority of western civilization having access to an Internet connection (19), developing and maintaining a personal website is now a relatively common practice. Many WCMS now provide built-in 'turnkey' solutions, removing the difficulties traditionally involved in building a site from the ground up, increasing the accessibility of this task to those who do not maintain strong technical and coding skill sets. The creation of an accessible and indexable website can take minutes, with it's purpose only limited to the imagination of the creator. Although WCMS maintain usage policies (see for example, Wordpress (20), Joomla! (21)) denoting the acceptable use of their platforms, a user may still opt to act in breach. In such instances, a site may remain active until reported (although there are automated copyright detection procedures (22)), at which point content may have been viewed, copied and distributed (indexed by search engines etc), beyond the control of the user.

Regulation of online content on such platforms provides a continuous issue, highlighted by Google in regards to its Blogger platform where it is stated that content is not monitored to

promote freedom of speech, and that users should try to settle disputes before taking formal action (23). Since 2013, 793 Wordpress sites were the subject of 482 government issued takedown requests, with only 23% of requests resulting in content or the site itself removed (24). Harvard University's 'Lumen Project' provides an insight into the volume and type of takedown notices regularly submitted to regulate content online (25).

Where a formal complaint has been lodged regarding a WCMS site, a *post mortem* investigation can be required to establish interactions with a website in question and content which has been posted. Identifying users, user accounts and their actions can support in establishing culpability. However, acquiring access to the requisite data to establish this information may not be straightforward in some instances. WCMS can be utilized both locally (local-hosting) and within cloud storage hosting systems online. Both pose regulatory issues and challenges to a forensic practitioner and the setup behind each method is discussed in turn.

How do they work: local hosting

The first option considered is for users to host their WCMS site locally. Some WCMS providers (see for example, Wordpress, Drupal and Joomla!) offer users the option to download a local version of their WCMS software and configure their site offline through the use of a server stack application (see for example MAMP (Mac) / XAMPP (Windows)). When complete, users can then locally host their site (or find a host, discussed in section *Host Provider*). To avoid the problems posed by dynamic IP addresses, a user can either purchase a static IP address or implement dynamic-DNS (DDNS) to make this option viable. Locally hosted sites pose the following advantages and disadvantages to practitioners investigating an incident related to the site hosted in this way.

Advantages:

1. Configuration information for the WCMS is local and therefore potentially retrievable upon seizure and investigation of relevant equipment.

2. As the user is responsible for the maintenance of their site, arguably may be an increased chance of backup and version control files being stored. This means that not only is there the potential to see the current site log files in operation, but also historical records which may document activity which has since been deleted or overwritten. This is beneficial as deleted data may not be retrievable from WCMS backend databases.
3. There is no reliance placed on compliance from a hosting service to provide relevant information regarding a site for an investigation.

Disadvantages:

1. As all site data is local and controlled by a user, log / site data is vulnerable to deletion or manipulation at any time. If a site suspect believes that they are under investigation, the time taken for law enforcement to seek necessary legal authority to seize and investigate equipment may prohibit an investigation as content can be removed at any time in absence of any preservation request facilities.
2. Conversely, whilst the presence of back-ups is noted above as an potential advantage, such processes may not always take place (or partial, ineffective or incomplete back-ups may be taken) where a user carelessly maintains their site. In such cases, the automatic back-up processes which may be offered by a provider could offer a greater source of evidence in some cases. Similarly, poorly documented version control may also pose an investigatory concern in terms of interpreting site content and structure.
3. A locally hosted site also means dealing with a users personal hardware and software configuration during an investigation. This may be bespoke, historical, or lack sufficient documentation regarding the setup posing a

challenge to the practitioner as they attempt to maintain evidential integrity.

4. Access to data can in some cases be an issue where a user has implemented local security and data protection measures such as disk encryption. In such instances, decryption content or suspect password disclosure may pose an issue, where co-operation from a WCMS provider may prove more successful.
5. Data may be more volatile, where a suspect at any point can take down a hosted site or content and securely delete it. In such instances, WCMS sites hosted by a service may maintain records of usage for short periods beyond an initial deletion point.

Host provider

A WCMS hosted through a dedicated provider poses additional practitioner challenges. Some WCMS providers offer a built-in hosted solution (for example, Wordpress a hosted option via wordpress.com as well as the Wordpress software for local download and configuration from wordpress.org). Where a host is utilised, website information typically resides on the host's server and where access to it is needed as part of an investigation, a providers compliance to disclose information regarding a site under investigation must be sought. Compliance provides the main potential barrier to an effective investigation of this type where necessary legal procedures must be adhered to potentially facilitate access to content. Where site information is disclosed, exported formats may vary ranging from full back-end database disclosures to xml output describing a sites structure and content (see Wordpress's turnkey hosted site export option).

Providers

The following sections provide an examination of Wordpress, Drupal and Joomla!, analysing core database structures which maintain information about what is stored on a site, user accounts and user interactions.

Wordpress

Wordpress is the most popular WCMS system currently available and is the first to be analysed. Where a user has locally hosted or utilised a hosting service, the Wordpress site database is often a fundamental source of information for interrogation, and access to it is required. However, depending on the hosting method, access to it may not be straightforward, as noted above. Most hosting services allow a site database to be extracted from the hosting site for purpose of backup retention and therefore in some cases, even sites hosted using a hosting service may still have copies of the site data stored locally (see for example 'www.000webhost.com' which was utilised as a host during the test creation and examination of a Wordpress website). When a site is developed and hosted locally, the sites database is likely be resident on local storage media.

As a starting point for analysis, reference should be made to the Wordpress entity relationship diagram (ERD) (26). ERDs describe the relationships between entities, which in the case of relational databases are table fields, and objects, essentially additional tables within the overall database. The Wordpress ERD provides an overview of table metadata types and relationships, supporting an effective examination of this content. The main Wordpress (version 4.9.1) WCMS database contains 12 tables. In regards to site usage, the following entries have been highlighted as they provide potentially evidential content.

Website Users

One of the first steps to investigating a hosted website lies with the identification of who has access to the site, usable accounts and the privileges which are assigned to these accounts. To start gathering this information, the *wp_users* table should be interrogated where those

who have created an account on the site and their registration metadata are provided (see Table 1).

Table 1: A breakdown of the *wp_users* table (relevant fields only).

Combine this information with the *wp_usermeta* table and privilege information can be gleaned for each account. Correlating the *ID* from table *wp_users* to *user_id* in the *wp_usermeta* table, levels of privilege given to each user can be established. The *meta_key* (*wp_capabilities* field) and associated *meta_value* entries denote the privilege status of each user.

Privileges include ‘subscriber’, ‘author’, ‘contributor’, ‘administrator’ and ‘editor’. The role attributed to each individual can impact the availability and integrity of poster information. Administrator roles have full access and control over the site, editors have the power to publish and manage all posts and authors can publish and manage their own posts. Contributors and subscribers can be considered minor roles and do not have the ability to publish their own content (27). Establishing the privileges associated to an account may narrow-down an investigation by helping to establish those who have the ability to post offending content, however it must be noted that signup credentials associated to an account may not be a reliable method of physically identifying an offender (the use of real names for sign up is unlikely where an individual intends to commit an offence). Identifying a physical user from account details assigned to a user of a WCMS site will arguably in most cases be limited to establishing the owner of an email account used as part of any WCMS signup processes by contacting the email addresses originating email service provider (Google, Microsoft etc.). Although validation of accounts is improving, such services are themselves not immune from the creation of fake, potentially untraceable accounts. In addition, where an offender registers a Wordpress account with an email account utilising fake details, identification may not be possible.

In addition to identifying accounts, where content has been posted in breach, establishing the user account which has posted the content is important.

Pages, Posts and Comments

A created and customised Wordpress site can take many forms, however, standard content containers are provided (pages / articles). Depending on the purpose of the site, it may contain few (synonymous with blogs) or many page artefacts. An account with appropriate privileges can create a new page. On each page, an account with appropriate privileges can post content. Subsequent viewers of content can make comments, where an account is not required, simply the submission of a name and email address along with the comment text. All of these actions maintain metadata within the *wp_posts* and *wp_postmeta* tables.

The *wp_posts* table describes posted content within the site (pages / articles, not comments), with Table 2 identifying the potentially evidential fields.

Table 2: *wp_posts* table (relevant fields only).

Comments

Those utilising a Wordpress site for the purpose of blogging or interaction with others online can provide a comments facility. Those logged into the site (with accounts) can comment on content and if the page/post if public facing, passive visitors can submit comments, providing they submit their name and email address (these are however not validated) as part of the comment submission process. Information surrounding comments is stored within the *wp_comments* table (see Table 3).

Table 3: *wp_comments* table.

Additional information regarding deleted comments is stored in the *wp_commentmeta* table, described below in Table 4.

Table 4: *wp_commentmeta* table.

Items which are removed from the trash are permanently deleted tests showed they were not recoverable from the site database using forensics methods. By default, Wordpress does not have any default version control systems (although plugins may offer this functionality, see for example 'VaultPress' (28)) therefore content subject to deletion may not be be stored elsewhere.

Evidential considerations

The Wordpress database maintains metadata documenting website interactions and construction of elements. One of the main challenges facing the forensic practitioner is acquiring access to this database, and this depends on the hosting strategy. If hosted locally, using Wordpress software, the database should also be stored on the local drive in the site directory of the server stack application in use (see for example, *Applications/MAMP/db*, the default installation for MAMP server stack on the Macintosh platform). The availability of this database for those who host Wordpress sites using hosting services with built-in Wordpress development suites may prove troublesome, with four areas to be considered.

1. Backups: Most hosting services provide a backup functionality whereby users can export their website database and store this content locally. Export formats may vary and it is not necessarily the case that exported site databases will remain in SQL format (exports include .xml, .csv etc.). In the case of 'www.000webhost.com' (a chosen test host with built-in Wordpress turnkey solution which offers a free site development and hosting option), a full sql export from phpMyAdmin is possible.
2. Identifying the host: Typically, hosting vendors offering a free service will provide a URL with the host provider's identifiable service included (for example, 000webhost provides a free service where site URLS are typically formatted as

'https://testsite.**000webhostapp**.com'). This can allow host identification in order to request the extraction of data regarding a site under investigation.

3. Cooperation: If the hosting service can be identified, access to the site configuration content is subject to their willingness to cooperate.
4. Comment edits: Unlike with posts, there is no record of edited comments. Tests showed it was not possible to detect which comments had been edited and by whom, using data contained within the site database.

Plugins

Wordpress maintains in excess of 47,000 plugins, downloaded almost 1.5 billion times, each offering the ability to extend the basic function of a Wordpress site (29). The problem posed by plugins lies with the variety of ways in which they store and maintain their data. Some plugins simply utilise the main site database to house information (see 'bbPress', a forum application for example) whereas others may function separately to this or not log information at all. Providing an analysis of the functionality of all available plugins is beyond the scope of this work, with each needing to be examined on an individual basis. Suggestions for analysis involve targeting plugins based on popularity and those likely to be involved with commonly occurring offences on these platforms.

Drupal

Unlike both Joomla! and Wordpress, Drupal does not offer a complete hosting solution (however hosting providers such as 'pantheon.io' do provide a turnkey solution). Typically, users download the Drupal software, configure and develop the site locally then either locally host it or identify a hosting service to upload their site to. As with Wordpress, the site database maintains a comprehensive record of content and interactions, and should be the starting point of an investigation into a Drupal site and forms the focus of the investigation presented in this article. The main Drupal (version 8) site database consists of 67 tables, with the following

analysis focusing on content relevant to denoting user interactions and posted content following a test examination of a local Drupal site.

Identifying users

In order to identify website account holders, the *users_field_data* table maintains a record of the logged in users (see Table 5).

Table 5: *users_field_data* table.

In addition to account information held in the *users_field_data* table, the *user_roles* table denotes the privileges attributed to each user account. Tying the *entity_id* in the *user_roles* table, to the *uid* in the *users_field_data* table, the name of the account can be identified (see FIG. 1).

FIG. 1- The *user_roles* table showing administrator privileges assigned to accounts 1, 3 and 5 (shown in *entity_id*, matched with *uid in users_field_data* table).

Drupal maintains 3 privilege levels, ‘anonymous user’, ‘authenticated user’ and ‘administrator’. By default, anonymous users can only view comments. Authenticated users have permission to post, and administrators have global access for complete configuration of the site. These permissions are flexible and therefore an administrator can expand the range of privileges that any user maintains.

Posted content

When analysing the content posted to a Drupal site, the *node_field_data* table provides a starting point for analysis (see Table 6).

Table 6: *node_field_data* table.

**Note:* The *node_field_revision* table (see FIG. 2) maintains a record of changes to an article. The *title* field demonstrates changes to the article title. Changes to an article content are recorded in the *node_revision_body* table. Matching the *vid* and *revision_id* values will identify the changed pages and additional content added / removed (see FIG. 3).

FIG 2- An edited article entry in the *node_field_revision* table.

FIG 3- An edited article entry in the *node_revision_body* table.

Tracking post edits: Where posts have been edited, it may be necessary to identify both the originating author and the editor, particularly where posted or edited content is in breach of law in order to attribute liability. Using the *revision_id*, matching this value to the corresponding *vid* entry in the *node_revision* table will allow the investigator to establish the *uid* of the account undertaking the revision and the timestamp the revision was made.

Comments

Similar to posted content, those with appropriate privileges can comment on website articles. Posted comment information is stored in the *comment_field_data* table (see Table 7).

Table 7: *comment_field_data* table.

Establishing what the comment was involves an interrogation of the *comment_comment_body* table (see Table 8).

Table 8: *comment_comment_body* table.

Note: Unlike posts, when comments are edited, the site database does not keep a strict log of changes. The watchdog table (see section *Watchdog Table*) does record events such as edited comments but there is limited information available to tie the revision to the event documenting the content of the comment before the edit took place.

Watchdog Table

The watchdog table provides an overview of actions carried out on the site itself, akin to a system log (see FIG. 4). These actions are chronologically ordered by the *wid* entry. The *uid* specifies the account triggering an event and the *message* denotes the event type. Of particular interest, the message field identifies when site content is deleted ('@type: deleted). The Drupal WCMS, unlike Wordpress and Joomla! does not (following testing) operate a trash functionality, therefore content which is deleted is not temporarily kept in this holding container. As a result, the watchdog table maintains the only record of acts of deletion (see FIG. 4). The binary large object (BLOB) data type allows binary data to be stored as a single entity. Through an analysis of the BLOB data, metadata surrounding the deletion can be ascertained. For example, the BLOB entry for a deleted 'article' with the title '6th page' is recorded as '*a:2:{s:5:"@type";s:7:"**article**";s:6:" %title";s:8:"**6th page**";}*'. The *hostname* field identifies the IP address of the account and *timestamp* identifies the time and date when the event took place (which in the case of an event denoting the deletion of content, the *timestamp* entry shows the time of deletion).

FIG. 4- Deleted entry in the watchdog table.

Note: The number of log records maintained can be adjusted within the settings of the site ('Configuration' -> 'Logging and Errors' settings). By default, 1000 are maintained, but this can be limited to 100 or expanded to all messages. Therefore the volume of information which can be obtained from the *watchdog* table may vary and is subject to user changes. When a log reaches its maximum size it operates on a first in first out basis, where initial records are

deleted (new events will continue to receive a wid value which increments past the maximum log value as earlier ones are deleted). The maintenance of log watchdog entries is initiated by 'cron', a Drupal automated task scheduler process. This can be set to run between every 1 hour to 1 week, never or can in cases be manually initiated by a user. Following testing, the watchdog log entries can exceed their maximum log size (where max 100 was set, over 100 records were initially stored), until a cron process is initiated. If cron is disabled, then watchdog records may be kept beyond log limits during this period, but further exhaustive testing over long-term usage of a site would be needed to determine this possibility.

Tracking users

Tracking interactions with the website can help to timeline specific events which have occurred. The *sessions* table provides information of the last account to be signed into the site. The IP address of the account is provided in the *hostname* field and *timestamp* provides the login time. Once the user is logged out, the session entry is removed from the *sessions* table, therefore the availability of this data depends on the point in which the database structure for the website was captured. In addition, the *watchdog* table maintains records for the opening (**Session opened for %name**.) and closing of (**Session closed for %name.**) account login sessions. An examination of the *uid* or BLOB entries for such events identify the name of the user (here, entry for account 'root5' login is shown:- **a:1:{s:5:"%name";s:5:"root5";}**). The associated timestamp documents the time of the event. As discussed above, there is no specific log entry for edited comments, but by establishing the user account logged in at the time of the edit (providing they have sufficient account privileges) can identify the editor.

Joomla!

As with Wordpress, Joomla! offers a hosted solution with a site URL containing a Joomla! domain suffix (for example, https://testsite1234.**joomla.com**/), or the Joomla! software can be downloaded locally for local configuration before hosting. Regardless of which option is chosen (Joomla's hosted option and local download were utilised in testing), access to the site's

underlying database offers the ability to identify site-usage events. The Joomla! (version 3.6.4) site database by default maintains 68 tables containing site usage information with the remainder of this section highlighting relevant entries.

Users

Identifying the users involved in the website's maintenance and interactions involves initial analysis of the *_users* table (see Table 9).

Table 9: *_users* table.

Once accounts have been established, privilege information requires interrogation of the *_user_usergroup_map* and the *_usergroups* tables. Mapping the account *id* to the *group_id* and then *group_id* to *parent_id*, the different roles of the account can be seen in the *_usergroup* table. Each user profile can maintain multiple roles as shown with account *id* '779' (in FIG. 5 and 6). As with the Wordpress and Drupal, roles dictate the actions which a user can carry out on the site.

FIG. 5- The *_user_usergroup_map* table showing user 779 with role id's 2, 3 and 4.

FIG 6- The *_usergroup* table showing user 779 (as shown in FIG. 5, with role id's 2, 3 and 4). Here, associating the group_id (from *_user_usergroup_map* table) to the parent_id (from *_usergroup* table) will reveal the roles which an account has, in this case 'Author', 'Editor' and 'Publisher'.

Site content

Establishing content on the site involves interrogating the *_content* table (see Table 10).

Table 10: *_content* table.

As with Drupal (discussed in section *Tracking users*), Joomla! maintains a record of active login sessions in the *_session* table. Here the *time* field denotes the login time, where the *userid* can be correlated with the *_users* table to establish the user accounts logged in. Following testing, *_session* table entries appear to be dynamic, and therefore if no one is logged into the site at the time of analysis (of export of the site database, this table will be empty). Unlike Drupal and Wordpress, Joomla! articles cannot be commented upon, and therefore those wanting to interact with the site must create an account and post an article.

Extensions

The Joomla! (30) extension directory currently maintains 7822 extensions (analogous to plugins in Wordpress) which can extend the functionality of site, with the overall number of extensions likely to be far greater when 3rd party vendor sites are considered. As with the issues noted above in section *Plugins*, extensions may utilise current back-end database structures to store configuration information, but each extensions must be examined on an individual basis.

Concluding analysis

This article provides an introductory analysis of WCMS sites and available evidence for establishing standard site interactions and setup. In the case of all three of the WCMS analysed, the WCMS databases offer a primary source of information from which to commence an investigation involving a WCMS site. Acquiring access to it can help a forensic practitioner to establish which user accounts have access to, and have been involved in the creation of content on the site. Where a site is locally developed or hosted, the local WCMS configuration and setup must be identified an examined. In the cases of the WCMS examined in this article, site structural information and interaction data is provided by each sites underlying database and configuration files where their location on a system will be subject to the WCMS package in use and server stack or associated applications being ran. Where a hosted option is preferred, access to site data (exported site content and associated database) is subject to a

vendor's terms and conditions, and legal authority to request disclosure. From the analysis carried out, the following points must be considered.

1. The site databases for Wordpress, Drupal and Joomla! do not track site revisions in significant detail. The result of this means that where a post has been made and then subsequently edited, revision history beyond the time and date of a revision is limited (subject to the use of version control plugins). The problem this poses is where derogatory or malicious comments / posts have been made and edited, original posted content may not be viewable leading to difficulties in establishing a chain of amendments. Similarly, the speed at which content may be made available then removed places a greater emphasis on the 'live analysis' of a website to ensure all available records of an event are taken whilst this information is still available.
2. Similar to the issue noted above, if content is deleted from the site (and also from any built-in trash function), records of deleted content may no longer be available. Therefore where an administrator account removes any violating content before recording a backup of the site database and content, it may not be possible to identify the physical poster of any illicit content. If site artefacts are stored on a local machine, forensic analysis of file system artefacts (for example, volume shadow copies) may reveal some historic version information which has been passively recorded by the operating system, but this is not guaranteed.
3. The acquisition of an account holder's IP address may be the only method of identifying a suspect in cases of investigating illicit posts. However, by default, only Drupal and Wordpress recorded this data during testing, where Joomla! sites would require the use of an additional plugin.

Given the prominence of WCMS usage, such systems must consider the impact of abuses and look to build in methods of assigning accountability to those who misuse these platforms. As can be seen via testing, although there are methods for tracking user interactions with a

WCMS site, greater detail will support investigations where in-depth dedicated session management information would support any examination of specific illegal acts and their associated user account. Whilst the tested platforms provide metadata supporting the ability to link comments/added content to an account, it is the ability to then attribute the account to a physical user which is potentially the greatest challenge. IP address information (despite being vulnerable to spoofing / anonymising protocols) is arguably one of the most valuable sources of information and its potential adoption by all WCMS to attribute site actions would be of benefit to law enforcement.

Future work

To further this research, future work should include the examination of additional WCMS platforms. Despite this article targeting the three most popular WCMS, covering a combined 71% market share, remaining vendors need to be analysed. Further, the article provides an overview of standard WCMS functionality, and further research should involve a targeted investigation into relevant WCMS plugins.

References

1. Barker D. Web Content Management: Systems, Features, and Best Practices. United States of America, US: O'Reilly Media, 2016.

2. Wordpress 'Wordpress.org' Available at: https://wordpress.org/ (Accessed 29th January 2018)

3. Joomla! 'Joomla! ' Available at: https://www.joomla.org/ (Accessed 29th January 2018)

4. Drupal ‘Drupal’ Available at: https://www.drupal.org/home (Accessed 29th January 2018)

5. Statista ‘Ranking der 10 Content-Management-Systeme (CMS) weltweit nach Marktanteil im Januar 2018’ Available at: https://de.statista.com/statistik/daten/studie/320670/umfrage/marktanteile-der-content-management-systeme-cms-weltweit/ (Accessed 29th January 2018)

6. Wordpress ‘Traffic’ Available at: https://en.wordpress.com/activity/traffic/ (Accessed 29th January 2018)

7. Wordpress ‘Posting’ Available at: https://wordpress.com/activity/posting/ (Accessed 29th January 2018)

8. Statista ‘Number of bloggers in the United States from 2014 to 2020 (in millions)’ Available at: https://www.statista.com/statistics/187267/number-of-bloggers-in-usa/ (Accessed 29th January 2018)

9. BBC News ‘Blogger arrests hit record high’ Available at: http://news.bbc.co.uk/1/hi/technology/7456357.stm (Accessed 29th January 2018)

10. BBC News ‘Indonesian police arrest 'hitman advertiser'’ Available at: http://www.bbc.co.uk/news/world-asia-17317439 (Accessed 29th January 2018)

11. BBC News ‘‘Mr Monkey' blog: South Tyneside Council drops £200,000 action’ Available at: http://www.bbc.co.uk/news/uk-england-tyne-28143035 (Accessed 29th January 2018)

12. BBC News ‘Blogger fined for 'menacing' rant’ Available at: http://news.bbc.co.uk/1/hi/wales/north_east/7373639.stm (Accessed 29th January 2018)

13. BBC News ‘Malaysia arrests second blogger’ Available at: http://news.bbc.co.uk/1/hi/world/asia-pacific/7622437.stm (Accessed 29th January 2018)

14. BBC News ‘Vietnam releases detained blogger’ Available at: http://news.bbc.co.uk/1/hi/world/asia-pacific/8253832.stm (Accessed 29th January 2018)

15. BBC News ‘Russian blogger Anton Nosik convicted of extremism’ Available at: http://www.bbc.co.uk/news/technology-37541039 (Accessed 29th January 2018)

16. BBC News ‘Millions of websites hit by Drupal hack attack’ Available at: http://www.bbc.co.uk/news/technology-29846539 (Accessed 29th January 2018)

17. McCarthy, Kieren ‘Panama Papers hack: Unpatched WordPress, Drupal bugs to blame?’ Available at: http://www.theregister.co.uk/2016/04/07/panama_papers_unpatched_wordpress_drupal/ (Accessed 29th January 2018)

18. Statista ‘Number of internet users worldwide from 2005 to 2017 (in millions)’ Available at: https://www.statista.com/statistics/273018/number-of-internet-users-worldwide/ (Accessed 29th January 2018)

19. Statista ‘Number of worldwide internet users as of January 2017, by region (in millions)’ Available at: https://www.statista.com/statistics/249562/number-of-worldwide-internet-users-by-region/ (Accessed 29th January 2018)

20. Wordpress ‘Terms of Service’ Available at: https://en.wordpress.com/tos/ (Accessed 29th January 2018)

21. Joomla! ‘Terms of Use for the Joomla.com Site’ Available at: https://www.joomla.com/tos (Accessed 29th January 2018)

22. Wordpress ‘Hall of Shame’ Available at: https://transparency.automattic.com/tag/hall-of-shame/ (Accessed 29th January 2018)

23. Google ‘Report inappropriate content ’ Available at: https://support.google.com/blogger/answer/76315?hl=en (Accessed 29th January 2018)

24. Wordpress ‘Takedown Demands’ Available at: https://transparency.automattic.com/takedown-demands/ (Accessed 29th January 2018)

25. Lumen ‘About’ Available at: https://lumendatabase.org/pages/about (Accessed 29th January 2018)

26. Wordpress ‘Database Description’ Available at: https://codex.wordpress.org/Database_Description (Accessed 29th January 2018)

27. Wordpress ‘Roles and Capabilities’ Available at: https://codex.wordpress.org/Roles_and_Capabilities (Accessed 29th January 2018)

28. Wordpress ‘Plugin Directory’ Available at: https://en-gb.wordpress.org/plugins/vaultpress/ (Accessed 29th January 2018)

29. Wordpress ‘Plugin Directory’ Available at: https://wordpress.org/plugins/ (Accessed 29th January 2018)

30. Joomla! ‘Joomla! Extension directory’ Available at: https://extensions.joomla.org/ (Accessed 29th January 2018)

Table 1: A breakdown of the *wp_users* table (relevant fields only).

Field	**Description**
ID	An integer value assigned to each user account. This value can be used to uniquely identify their actions on the site.
user_login	A string value representing the account holder’s chosen username.
user_registered	This is the date and time that the account was created.
user_pass	Encrypted account password using PHPass.
user_email	The email account used to register the account. The email is not verified.

Table 2: *wp_posts* table (relevant fields only).

Field	**Description**
ID	The *ID* column provides a chronological order of posted entries. An *ID* is given to each individual post. Where the sequence is out of order (for example, 1,2,4), an entry has been removed from the table (i.e. a post has been deleted from the site, in this example, '3').
post_author	The *post_author* field denotes the author of a post.

post_date & post_date_gmt	Time and date information that the post was made.
post_content	This field contains the content of the post. This will vary depending on the type of post made (see *post_type* field below).
post_title	Title of post. For example, if post is the creation of a new page, the title of that page is recorded here.
post _status & comment_status	*post _status* indicates the accessibility of the post. For example, whether it is 'private' or 'public'; accessible by viewers.
post_modified & post_modified_gmt	The time and date of post, if edited. If the post has not been edited, these values will match the *post_date* and *post_date_gmt.*
guid	URL of site page where content is contained.
post_type	This field indicates the type of post made. Entries include page (creation of a site page), post (post made to page), forum & topic (creation of forum structures on the site and posts made to the forum) and revision (edited content). In cases of post revisions, an associated *post_parent* field within the table denotes the *ID* field (noted above) of original post from which the edit was made. (for example, if post *ID* 10 is a revision of post *ID* 4, *post_parent* if *ID* 10 will have a value of 4). This allows the tracking of revisions on a post over a period of time.

Table 3: *wp_comments* table.

Field	**Description**

comment_ID	A chronological list of comments made on the site. Where the numeric sequence is out of order, a comment has been deleted from the site (same as the *ID* field discussed in Table 2).
comment_post_ID	This is a numeric identifier for the post that the comment has been made on. This entry should be matched against the *ID* field in the *wp_posts* table (shown above in Table 2) to identify the original post containing to the comment.
comment_author	Name of the commenter.
comment_author_email	Email account associated to the commenter.
comment_author_IP	IP address of the commenter. Given that no validation is carried out on other commenter details, this may be important when trying to attribute offending comments to a physical individual.
comment_date & comment_date_gmt	The date and time that the comment was made.
comment_content	The content of the comment.
comment_approved	Status of the comment. The value will be '0' if it has not received approval status from an account with the necessary privileges. If value is '1', the comment has been approved. Where the value is 'trash', this represents a deleted comment and further information is stored in the '*wp_commentmeta*' table (discussed below in Table 4).
comment_agent	Metadata surrounding the commenter's platform (i.e. browser type/version and device type/operating system).

Table 4: *wp_commentmeta* table.

Field	**Description**
comment_id	This is a numeric value identifying the deleted comment. Match this value to the *comment_ID* in the *wp_comments* table to establish further metadata about the deleted comment.
meta_value	Where this value equals '1', the comment is in the 'trash'. Each comment in the 'trash' has a corresponding *_wp_trash_meta_time* entry. This is a Unix timestamp, which when converted corresponds to the time which the comment was deleted.

Table 5: *users_field_data* table.

Field	**Description**
uid	This is a unique numeric value assigned to an account on the website. This value can be used to associate actions carried out on the site to a specific user account, for example, what content they have posted content.
name	This is the username assigned to the account.
mail	This is the email address used to sign up for the account. No validation is carried out on the account and this could be fake. In turn, it is not a requirement for account holders to input an email address, therefore this information can be omitted on signup.
created	This is the date and time that the account was created.
login	This is the last sign-in of the account.
changed	This is the time and date the user's profile settings were opened. The timestamp changes even where the settings have not been actually updated.

Table 6: *node_field_data* table.

Field	**Description**
uid	This is a unique numeric value assigned to an account on the website (the same as in the *users_field_data* table). This is used to attribute posted content (pages, articles) to a user.
type	For example, an ‘article’ or ‘page’.
title	The title of the article / page.
created	This is the created time and date of the posted content, stored as a Unix timestamp
changed	This is the time and date of the last time the posted content was edited,stored as a Unix timestamp
nid	A second identifier tied to a post. Match the *nid* value to the *nid* value in the *history* table to establish the last time a post was viewed (stored as a Unix timestamp).

Table 7: *comment_field_data* table.

Field	**Description**
uid	This is a unique numeric value assigned to an account on the website. This is used to attribute posted comments to a user.
subject	This is the name of comment (subject).
email	The email address of the account posting the comment.
created	The created time and date of the comment.
changed	The time and date of last edit on the comment.
cid	Unique ID of the comment. Link this ID with *entity_id* in *comment_comment_body* table (below, Table 8) to identify the actual content of the comment.
hostname	Hostname of the commenter.

Table 8: *comment_comment_body* table.

Field	**Description**
comment_body_value	Content of the comment.
entity _id	Unique identifier of the comment.

Table 9: *_users* table.

Field	**Description**
id	This is a unique numeric value assigned to an account on the web-site.
username	Username of the account.
email	The email address used to sign up for the account. No validation is carried out on the account and this could be fake.
registerDate	The date and time of the account creation.
lastvisitDate	The date and time of the last sign-in of the account.

Table 10: *_content* table.

Field	**Description**
id	This is a unique numeric value assigned to a created article. Where these values are out of sync (as demonstrated with Wordpress), an article has been deleted.
title	The title of the article.
introtext	The content in in the article.
created	The date and time of the article's creation.
created_by	The *id* (see *_users* table) of the account creating the article.

modified	The date and time of the article's last modification. If no modification has taken place since it was originally posted, this field will reflect the same date and time as in the *created* field.
modified_by	The *id* (see *_users* table) of the account who has modified the article. This field will be blank if it has never been modified.
version	This value indicates the number of edits that the article has undergone.
hits	This value shows the number of visits to the article's page (page views).
state	The state value identifies the current state of content. Where this value is '-2', the content is deleted and in the trash.