Drupal Security Checklist

1. Integrate your security team early on in the development process to assure that your needs can be met in an acceptable timeframe.
    - Applications should periodically be reviewed by a third-party, to assure security.
    - Develop an ongoing security testing plan
    - Re-review the application whenever major changes have been made.

2. Harden the application and server architecture.
    - Protect risky Drupal files from the internet: *install.php, cron.php, xmlrpc.php*.
    - Harden PHP: https://www.owasp.org/index.php/PHP_Security_Cheat_Sheet
    - Harden the server: http://www.sans.org/score/checklists/linuxchecklist.pdf

3. Disallow weak passwords for privileged users and enforce a strong password policy.
    - Utilize the Password Policy Drupal module (https://drupal.org/project/password_policy) to enforce a password policy that meets your company security guidelines.

4. Implement Server, Application, and Drupal logging. Assure that logs are being stored on a separate and trusted server and actively review/parse these logs for security events.

5. Make sure that Development modules are not installed on production applications. Examples:
    - Devel module (https://drupal.org/project/devel)
    - Masquerade module (https://drupal.org/project/masquerade)

6. Review and apply all available Drupal security updates.

7. Disallow untrusted user roles from creating content using HTML (filtered / unfiltered) to avoid JavaScript inclusion. Also explicitly disallow PHP code execution.
    - While limited HTML is recommended by the Drupal community, a skilled attacker may still bypass these restrictions and attack a site or its users via user-generated content.

8. Check file permissions; verify there are no unintentional world-writeable files.

9. Implement CAPTCHA or a similar mechanism in front of user-registration and login forms.
    - Assure that this is not configured to allow authentication/registration attempts following an initial successful CAPTCHA completion.
    - This will also help mitigate the creation of accounts by a botnet and deter subsequent comment spam.

10. Install and run the Security Review module (https://drupal.org/project/security_review).
    - Verify and resolve any uncovered issues.

11. Regularly check the site-status report page and resolve any open issues.

12. Assure that the HTTPOnly flag is set to protect user sessions from attacks such as XSS.
    - Whenever possible, implement the Secure Flag as well, so sessions are not inadvertently passed in plain text over HTTP.

13. Protect the application with network and application layer firewalls and/or IDS/IPS.

14. Assure there are no resident phpinfo files / phpmyadmin installations / etc. accessible to users.

4780 PEARL EAST CIRCLE, BOULDER CO 80301 O 720.881.5321 F 303.413.8791 LOGRHYTHM.COM

Greg.Foss@LogRhythm.com | Sr. Security Research Engineer | LogRhythm Labs