Gaps:

1) No specific tool for requirement gathering

1) No end to end traceability between tools

2) Jenkins in underutilized.

3) No build automation i.e; unit testing, code-coverage, static code analysis.etc.

4) No configuration management tool used.

5) Merge conflict faced during commit resolved manually.

6) Little long span being used for new feature deployment.

7) No automation for static code analysis.

8) No tracking mechanism is used for code check-in review.

9) No api documentation for Code that is written.

10) No gurantee of code stability during deployment.

11) Rollback process is manual.

12) There are 5 environments with different configurations.

13) Shell script is used to create environments

14) Shell scripts are maintained manually

Solution:

1. Use Confluence for requirement gathering.
2. Have a separate credential for developer in Jenkins .
3. Use build tool either Ant or Phing or through Jenkins which can do following

i) unit testing

ii) static code analysis

iii) UI Testing (Selenium)

iv) Code coverage

v) Code quality check

vi) Deploymnet

vii) Trigger build

viii) Scripts for configuration management can be triggered using Jenkins.

1. Create pull request for code review through Jenkins before merging.
2. Use Clover-Php plugin to generate code coverage report
3. Use Sonarqube for static code analysis
4. Build a delivery pipeline for deployment in different env.
5. Instead of having two different configurations, a stable version should be created. Ex: create a stable version of code in php 5.3 and later create for php 5.5.
6. Use single shell script for all the environments.
7. As shell scripts are itself difficult to maintain, use configuration management tool.
8. Jenkins can be used to trigger the scripts in remote machines for configuration management.