Prozessinnovationen für Sicherheit in Open Source Projekten

Martin Gruhn
Institut für Informatik
FU Berlin
Dezember 2008

Ziele dieses Vortrags

- Für mich:
  - Eigene Forschung **verständlich zusammenfassen**
- Für Studierende
  - **Forschung** der AG Software Engineering **vermitteln**
- Für beide Seiten
  - Fruchtbarer Boden für **Seminar**- und **Abschlussarbeiten**

Inhalt

Freie Universität Berlin

- Einführung E
  - Motivation, **Ziele**, Forschungsfragen, Begriffe
- Bisherige **Projekte**, Relevante **Literatur** P
  - Was war? -> Erkenntnisse
- **Methode** M
  - Empirische qualitative Forschung: Action Research
  - Herausforderungen und Ansätze
- Mögliche **Stoßrichtungen** X
  - Ausblick
  - X-Arbeiten

- Zwischenstand: Abschlussarbeit von **Florian Thiel**

Motivation für Forschung

Freie Universität Berlin

- Relevanz von (Web-) Sicherheit und OSS
  - Einsatz von **Freier Software** [FSF08] ist **verbreitet**!
  - **Viele Verletzlichkeiten** bei Webanwendungen
- Forschungsbereich: SWT
  - Gesucht: Praktiken im Entwicklungsprozess für Sicherheit
  - Open Source Projekte als **Testbett** [OP07]
    - Annahme: OSler sind fähige Programmierer
- Auch: Wirtschaftliches Einsatzszenario
  - Unternehmen bieten Dienstleistung bzgl. CMS an
  - Wie Sicherheit gewähleisten?

Ziel der Forschung

Freie Universität Berlin

- Ziel
  - **Untersuchung von Innovationen für Sicherheit**
  - In Open Source **Entwicklungsprozess**en
  - Anwendung/Produkt sicherer machen
- Annahmen
  - Technologien für Sicherheit sind vorhanden
  - Problem: Umsetzung (Anwendung, Konfiguration)
  - Prozessoptimierung erzeugt sichereres Produkt
  - Sicherheit bedeutet: Weniger Verletzlichkeiten
- Folgerungen
  - Gute und schlechte Ansätze identifizieren und verstehen
  - Beobachtungen und Experimente in realer Umgebung
  - Empirische, qualitative Forschung!

(Forschungs-) Fragen

- **Praktiken**
  - zur Vermeidung und Behandlung von Verletzlichkeiten?
  - Welche sind gut oder schlecht? Warum?
- **Werkzeuge**/Technologie
  - Welche Werkzeuge gibt es für Qualitätssicherung bzgl. S.?
  - Wie werden sie angewandt?
  - Inwiefern tragen sie zu Sicherheit bei?
- **Verletzlichkeiten**
  - Was ist die wichtigsten Ursachen für Schwachstellen?
  - Wie kann man sie vermeiden?
  - Warum werden immer wieder die gleichen Fehler gemacht?

Eingrenzung, Terminologie

Freie Universität Berlin

- **Zugang zu Sicherheit**
  - Nicht ganzheitlich: Wie mache ich Produkt sicher?
  - Sondern: Wie vermeide ich Verletzlichkeiten?
- **Schwachstellen** (Verletzlichkeiten)
  - Strukturelle Unzulänglichkeit eines Systems, die zu einem Unfall durch einen Angriff führen kann (nach [AWS08])
    - Fehler kann sein: Versäumnis, Falschtun
  - Häufige Verletzlichkeiten in **OWASP Top 10**
    - XSS, SQLIA, ...
  - Häufig: Schwachstellentyp <-> Angriffstyp
- **Webanwendungen**, CMS
  - Weisen häufig Schwachstellen auf, viele praxisrelevante OSS Projekte: Bsp: Wordpress, Drupal, Joomla, phpBB, TWiki, ...

Literaturrecherche

- **Entwicklungsprozess** für Sicherheit
  - Entwicklungsmethoden: MS SDL, OWASP CLASP
  - Wiss. Paper: Vergleich [GBW+07], emp. Studien [GMW06]
- **Evaluierung**
  - Standards: Common Criteria, SSE-CMM
- **Praktiken**: Ratgeber/Praxisbücher
  - Writing Secure Code [HL03], Building Secure Software [VG01], MSDN: Security Patterns & Practices (Web)

Literaturrecherche

Freie Universität Berlin

- **Schwachstellen**-Forschung:
  - Technologie und Praktiken gegen SQLIA und XSS
    z. B. [HVO06, BV08]
  - Analyse und Katalogisierung von Schwachstellen
    z. B. [LBM94, CWE, CAPEC]
- Bekannte **Konzepte aus Sozialforschung**
  - To do: Community of Practice, Cognitive biases...

Projekt: Werkzeuge für Sicherheit

Freie Universität Berlin

Werkzeuge für Sicherheitstests

- Bachelorarbeit von Michael Osipov
- Ziel:
  - Eingesetzte Werkzeuge klassifizieren, Bedeutung für Sicherheit
- Fallstudie:
  Web CMS OSS-Projekte
  - Joomla!, JAMWiki, Bugzilla, MediaWiki, Apache Tomcat, Gallery, phpBB
- Untersuchung von
  - Projektseiten, Repos, Kommunikationskanäle, Kontakt mit Entwicklern

Erkenntnisse

- Statische Codeanalyse
  - gut für Syntax- (und Struktur-) Prüfung und Musterabgleich
  - problematisch zum Finden von Sicherheitsschwachstellen
- Gut: Durchsichten und Penetrationstests
  - von erfahrenen Entwicklern, Aufstellen von Code-Regeln
- Fuzzer bringen's nicht?

Weitere Stoßrichtung

- Durchsichten mit **Werkzeugen** für **statische Analyse**
  - Aufsatz: Automated Code Review Tools for Security [McG08]
  - Hypothesen:
    - Tools are faster, can evaluate code more frequently
    - Tools encapsulate security knowledge
    - Tool operator is not required to have the same level of expertise as a human editor
  - Forderung:
    - Results must be understandable to normal developers who might not know much about security
  - Offen: Einschränkungen
    - Anwendungstyp/Progrmmiersprache
    - Programmiersprache
    - Schwachstellentypen
- X-Arbeit?

Werkzeuge für Sicherheitstests?

Freie Universität Berlin

Mark Thomas (PMC Mitgl. von Tomcat) nach Anfrage von Michael Osipov auf Tomcat Developers List (30.04.2008):

- We do occasionally receive reports to the security team that provide outputs from various security testing tools. In short, **the output is nearly always complete garbage.**
- For example, on one occasion a handful of XSS issues were reported **all of which were invalid** whilst valid XSS issues (later reported by others) were completely missed.
- I have yet to see an automated security test tool that offers **any useful** output against the Tomcat code base.
- [...] automated tools for finding general bugs can work. **I** haven't (and **wouldn't**) **use them to find security issues**.

Projekt: Schwachstellenbehebung

Freie Universität Berlin

Verhalten bei Schwach-stellenbehebung in OSSP

- Masterarbeit von Tobias Opel
- Ziel: Verstehen und Beschreiben des Behebungsprozesses
  - gute und schlechte Praktiken identifizieren
- Qualitative Inhaltsanalyse:
  - von Patch trackern, Schwachstellen DBs, Mailinglisten, Blogs
- Web CMS OSS-Projekte
  - Wordpress, Joomla, TWiki

Erkenntnisse

- Wichtig:
  - Sicherheitsteam sollte Risiko von S. selbst einstufen
  - Spezielle Kommunikationskanäle für Schwachstellen
- Unklar (aber interessant)
  - Full disclosure vs. nichtöffentliche Meldung von S.
  - Einfluss von: Externen Helfern und Drängeln

Umfragen zu Security Assurance

Freie Universität Berlin

Schwachstellenbehebung und -vermeidung

- auf FOSDEM im Frühjahr 2008
- **Erkenntnisse**
  - Bebehungsprozess
    - Report, identify & discuss, assign, develop, release & announce
  - Maßnahmen für S.-Vermeidung
    - Technology, conventions, code reviews, testing (automated, manual, tools), external security assessment

Maßnahmen für Security Assurance (Ü-Empirie 08)

- Andrej Haralevich, Alexander Kunze, Sascha Rasmussen
- Fragen
  - Maßnahmen angegeben vs. selbst durchgeführt
  - (Partitionierung bzgl.: Projektgröße, Erfahrung, Softwaretyp)
- **Erkenntnisse**
  - Vom Projekt Vorgegebene Maßnahmen werden von großem Anteil d. Entwickler nicht durchgeführt
  - z. B. Coding Guidelines

Bisherige Projekte: Zusammenfassung

Freie Universität Berlin

- Offen: Welches sind die **Probleme?** Vermutungen:
  - Technisch
    - Technische Lösung noch nicht vorhanden
  - Psychosozial
    - Vermutung: Problem ist Faktor Mensch
    - z. B. bei Konfiguration, Anwendung
    - Mangel an Kompetenz vs. Wissen vs. Motivation/Disziplin?
- Bisheriger **Ansatz**:
  1. Projekte auf wichtige Schwachstellen hin erkunden
     - Häufige Schwachstellen: OWASP Top 10
  2. Ursachen herausfinden
  3. Action Research!
     - Evaluieren von Innovationen, indem wir sie ausprobieren

Action Research

- Nach Avison, Lau, Myers und Nielsen [ALM+99]:
  - „**Try out** a theory with practitioners in **real situations**,
  - gain **feedback** from this experience,
  - **modify the theory** as a result of this feedback,
  - and **try it again**.“
- Elemente :
  - Problemdiagnose
  - aktives Eingreifen
  - reflektierendes Lernen
- Qualitative wissenschaftliche Methode [Mye08]
  - da Analyse hauptsächlich qualitativ

Action Research

- **Wichtig**
  - Klarheit über Forschungsziel und (wiss.) Methode
  - Sorgfältige Dokumentation des Vorgehens
  - Anforderungserhebung zusammen mit Gruppe -> Offenheit
  - Ethische Schranken
- Weitere **Anforderungen** für Innovationseinführung [DGJ+08]:
  - Zeit in realer Umgebung
  - Ehrlichkeit über das, was funktioniert
  - Bereitschaft, einfache technische Lösungen zu akzeptieren

Problem? Problem!

- Problem und Lösung bei Action Research
  - Problemdiagnose erfolgt zusammen mit Entwicklern
  - Lösung wird gemeinsam erarbeitet
  - **Methodenwissen + Domänenwissen = Lösungswissen**
- Voraussetzung
  - Problem ist bekannt, Diagnose möglich
  - Offen bleibt Konkretheit der Lösung
- „Action Evaluation“
  - Forscher versucht, eine konkrete Lösung einzubringen
  - Praktiken sind vorgegeben
- Problem
  - **Was ist das Problem?**

Herausforderungen: Bsp. phpBB

Freie Universität Berlin

- Durchsicht des Codes von phpBB (Florian Thiel)
  - Bzgl. XSS und SQLIA (Eingabevalidierung)
  - Häufig SQL-Code durch String-Konkatenation
  - Furchtbares DB-Schema
  - Vgl. mit Code von professioneller Webanwendung:
    - Verwendung von Web-Frameworks, kaum SQL-Code
- **Vermutungen**
  - V. sind Relikt der Anfangsphase von Webanwendungen
    - wenig Bewusstsein für Sicherheit, gewachsene Anwendung
  - OSS Entwickler scheuen umfangreiches Refactoring
  - Eingabevalidierung verlagert sich von Projektcode in Frameworks!
  - Nutzung von Frameworks (Standardlösungen) beugt V. vor

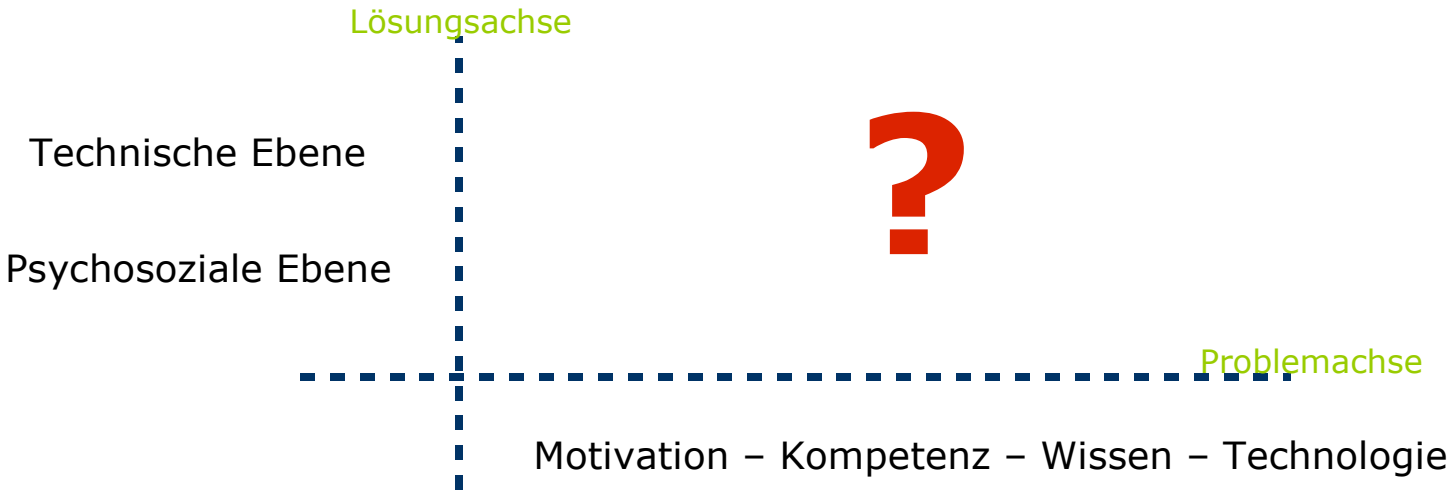
Herausforderungen: Bsp. phpBB

Freie Universität Berlin

- **Erkenntnisse**
  - Schlechtes Handwerk erzeugt schlechten Code
- Wissenschaftlich relevant?
  - Wollen, sollen und können wir an dieser Stelle eingreifen?
- Möglicher **Lösungsansatz** hier:
  - Gründe für bisherige Auslassung ermitteln
  - Disziplinproblem lösen
  - Refactoring forcieren

Ausblick

- **Forschungsansatz** (modifiziert)
  1. Projekte auf Schwachstellen hin erkunden
  2. Ursachen herausfinden
  3. Wenn wissenschaftlich relevant:
     - Lösungsmöglichkeiten ermitteln
     - Abhängig von Konkretheit: Action Research/Evaluation
- **Ausrichtung**



Vielen Dank!

Abschlussarbeiten

- Setzen voraus
  - Idee einer eigenständigen Leistung
  - Seminarvortrag (+ Seminar-Arbeit) in diesem Kontext
    - Literaturarbeit!
- Aus Seminar-Arbeit erwächst
  - Proposal/Exposé (1-2 Seiten) -> Aufgabenstellung

Verweise

[ALM+99] D. Avison, F. Lau, M. Myers, P. A. Nielsen. *Action Research*. Communications of the ACM, Vol. 42, No. 1, January 1999. (PDF online)

[AWS08] L. Prechelt. *Vorlesung Anwendungssysteme* im WiSe 2008/09.

[BV08] P. Bisht, V. N. Venkatakrishnan. *XSS-GUARD: Precise Dynamic Prevention of Cross-Site Scripting Attacks*. DIMVA 2008.

[CAPEC] Mitre Corp. *Common Attack Pattern Enumeration and Classification*. Online: capec.mitre.org.

[CWE] Mitre Corp. Common Weakness Enumeration. Online: cwe.mitre.org.

[DGJ+08] Donner, J.; Gandhi, R.; Javid, P.; Medhi, I.; Ratan, A.; Toyama, K. & Veeraraghavan, I. R. *Stages of Design in Technology for Global Development.* IEEE Computer, 2008.

[FSF08] Free Software Foundation Europe. *What is Free Software?* Online, 2008.

[GBW+07] Gregoire, J.; Buyens, K.; Win, B. D. & Riccardo Scandariato, W. J. *On the Secure Software Development Process: CLASP and SDL Compared.* Third International Workshop on Software Engineering for Secure Systems on ICSE 2007.

[GMW06] Glisson, W. B.; McDonald, A. & Welland, R. *Web engineering security: a practitioner's perspective.* Proceedings of the 6th international conference on Web engineering, ACM, 2006.

Verweise

[HL03] M. Howard, D. LeBlanc. *Writing Secure Code*. 2nd ed., Microsoft Press, 2003.

[HVO06] W. G. Halfond, J. Viegas, A. Orso. *A Classification of SQL Injection Attacks and Countermeasures*. IEEE Int'l Symposium on Secure Software Engineering, 2006.

[LBM94] Landwehr, C. E.; Bull, A. R.; McDermott, J. P. & Choi, W. S. *A taxonomy of computer program security flaws.* ACM Comput. Surv., ACM, 1994.

[Mye08] M. Myers. *Qualitative Research in Information Systems*. Online, 2008.

[OP07] C. Oezbek, L. Prechelt. *On Understanding How to Introduce an Innovation to an Open Source Project.* Proceedings of the 29th International Conference on Software Engineering Workshops (ICSEW '07), IEEE Computer Society, 2007. (PDF online)

[VG01] J. Viega, G. McGraw. *Building Secure Software*. Addison-Wesley, 2001.