Securing Jenkins

Kohsuke Kawaguchi
Creator of the Hudson/Jenkins project

©2011 CloudBees, Inc. All Rights Reserved

About CloudBees

Our Mission	Become the leading Java™ Platform as a Service (PaaS)
Why We're Different	**CloudBees services the complete lifecycle of Cloud application development and deployment.** **No Servers. No Virtual Machines. No IT.**
Strategy	▪ **DEV@cloud** – Cloud Services for Developers ▪ **RUN@cloud** – Frictionless runtime PaaS for Java apps

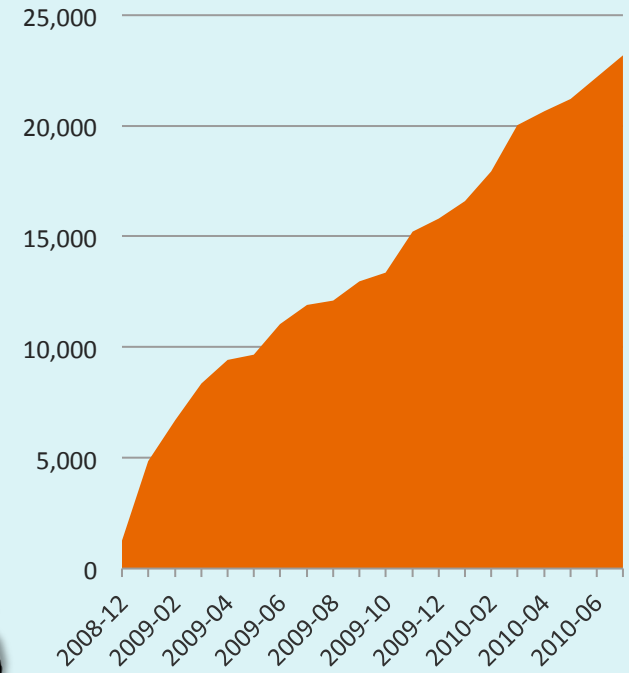
Continuous Integration - Jenkins

Nectar – On-Premise Enterprise Jenkins

- **Support** from the experts.
- **VMware** scale your Jenkins environment.
- **Enterprise Features** extend Jenkins for large environments.
- **Integrate with the Cloud** integration with DEV@Cloud and RUN@Cloud coming

Benefits of DEV@cloud Jenkins Service:

- **Scale** your Jenkins environment with the power of the Cloud
- **Ease** your Jenkins management overhead
- **Speed** your builds
- **Save money** with on-demand Jenkins Service. Starts from $0/month



Source: jenkins-ci.org

Idea Behind This Webinar

- Architecture & modeling of access control in Jenkins
- Walk-through of security related plugins/ core
- Practical tips in configuring security
- Security beyond access control

©2011 CloudBees, Inc. All Rights Reserved

Access Control Architecture

- Three extension points
  - Authentication: figuring out who you are
  - Permission: activity that may need protection
  - Authorization: are you allowed to do XYZ?

Authentication

- Figures out user ID and groups
  - For example, via username/password field
    - But not always. E.g., OpenID, SSO
  - Often additional information as well
    - e-mail address, full name, …
- HTTP handling carries this around
- Plugins can control this completely

System-defined Identities

- “anonymous” user
  - Automatically given to unauthenticated requests
- “SYSTEM” user
  - All background threads run under this identity. Supposed to have full access
- “authenticated” group
  - Every non-anonymous user automatically gets it

Permission

- Unit of activity to control access
  - “Build a job”, “Create a view”, “Read Jenkins”, etc.
- Organized in shallow tree structure
  - A permission can imply others
    - “Read job configuration” implies “Read job”
    - “Administer” implies everything else
- Plugins often define their permissions
  - “Promote a build”, “Make a Maven release”, etc.

Authorization

- Given three parameters, decide OK/NG
  - Object
    - A job, view, root Jenkins object, etc.
  - Permission
  - Subject (Identity)

- Plugin can completely control the logic

Architecture Key Points

- Authentication and authorization are orthogonal
  - Authentication establishes the identity (including membership)
  - Authorization uses that to decide OK/NG
- So you get to mix and match

PAM Authentication

- Fancy way of saying Unix user authentication
- It Just Works
  - Virtually zero configuration
  - Your ITops have already done the hard work
- Picks up Unix group memberships
- Gets local user/group support for free

Active Directory (plugin)

- Windows equivalent of PAM
  - Richer
- It Just Works, especially since 1.17
  - Zero conf on Windows, very little on Unix
  - AD forest, sites, DC fail over, …
- Picks up membership
  - Including indirect ones
- No WIA support yet

LDAP

- Supported well
  - Both binding modes, configurable group search, e-mail address retrieval
  - Default configuration and inference that goes beyond typical LDAP impl
- Caution: group name
  - Earlier version turned “group” into “ROLE_GROUP”. Fixed in 1.404
- But do you really need it?

OpenID (plugin)

- Login aid mode
  - Use OpenID instead of typing password
  - You've seen those on websites
- SSO mode
  - Clicking "login" auto-initiates OpenID session
  - With proper OpenID server configuration, it becomes password-less SSO
  - Better way of integrating with directory servers
- Extensibility to support group memberships

Script Realm (plugin)

- Gist of authentication is:

  f: (username,password) ⇒ (group*) or “invalid”

- Let people write a shell script to do that
  - Handy duct-tape solution for custom identity systems

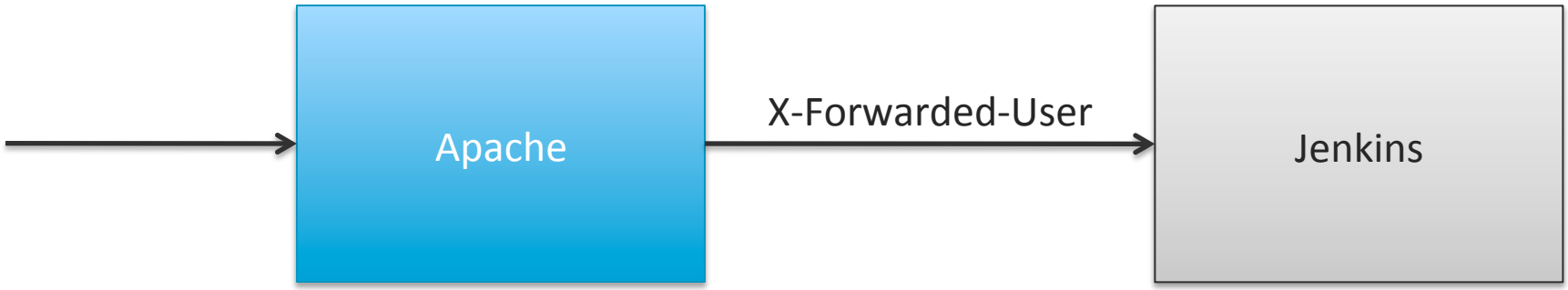
CloudBees

Delegates to servlet container

- Useful if…
  - You run Jenkins on an existing servlet container
  - Your admin has already set it up for authentication
  - You use directory servers that don't support OpenID
- Group membership support is clumsy

Delegate to reverse proxy (plugin)

- Let Apache does the authentication
  - For some people, this is easier and/or more powerful
- Jenkins get it via HTTP header



Jenkins' own user database

- Retain user/password info in Jenkins
  - No external identity system needed
  - Optionally let people sign up via UI
- No group support yet
- Very limited use case (or am I wrong?)

Other Authentication Implementations

- CAS
- Atlassian Crowd
- SourceForge Enterprise Edition
- CollabNet TeamForge
- ...

Authorization

- Several trivial implementations
- Really only two implementations
  - (Global) matrix security
  - Project-based matrix security

- Calling for more plugins!

Matrix security basics

- Recap of the concept
  - (subject,object,permission) → OK/NG
- Matrix Implementation
  - Define (subject,permission) as a checkbox matrix (aka ACL)
  - Honors all implied permissions
  - Honors all group memberships

User/group	Overall		Slave		Job							Run		View			VMWare Pools	SCM
	Administer	Read	Configure	Delete	Create	Delete	Configure	Read	Build	Workspace	Release	Delete	Update	Create	Delete	Configure	Configure	Tag
kohsuke	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Anonymous	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Global matrix security

- Just one matrix for the entire Jenkins
  - Object doesn't matter
- Adequate so long as you don't have black projects

Per-project security

- Global + separate matrix at each project
  - Optional
  - Individual matrix inherits global matrix
    - “OR” semantics. No “deny” entry
- Also note:
  - No mechanism to reuse matrix
  - Config job permission lets you edit project matrix

“Create job advanced” plugin

- Works well with per-project matrix
- Grant the creator full access when a new job is created
  - Can also grant anonymous read-access
  - From there, he can add others

Tip: what groups am I in?

- Visit http://yourserver/jenkins/whoAmI
  - Useful for checking what the server is seeing

Who Am I?

Name:	kohsuke
IsAuthenticated?:	true
Authorities:	• "ROLE_ADMINS" • "authenticated" • "ROLE_ALL"
Details:	org.acegisecurity.ui.WebAuthenticationDetails@957e: RemoteIpAddress: 127.0.0.1; SessionId: null
toString:	org.acegisecurity.providers.rememberme.RememberMeAuthenticationToken@cb4a5a0b: Username: org.acegisecurity.userdetails.ldap.LdapUserDetailsImpl@6a64704b; Password: [PROTECTED]; Authenticated: true; Details: org.acegisecurity.ui.WebAuthenticationDetails@957e: RemoteIpAddress: 127.0.0.1; SessionId: null; Granted Authorities: ROLE_ADMINS, authenticated, ROLE_ALL

Tip: If you lock yourself out

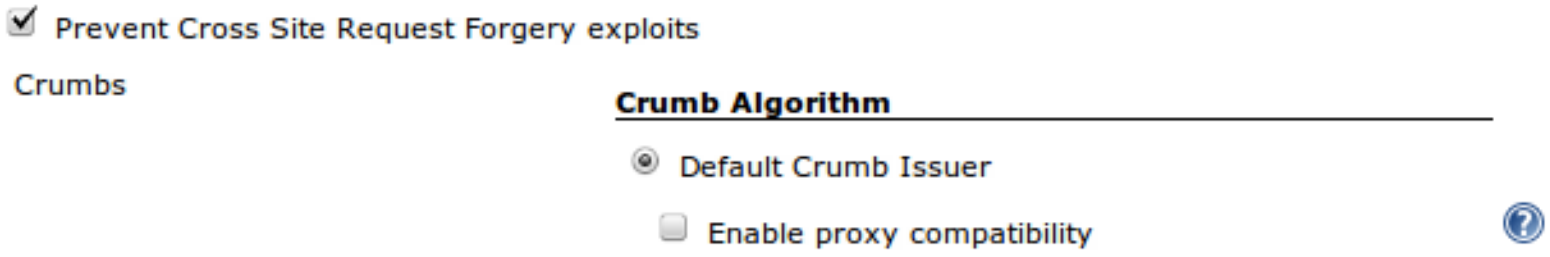
- Stop Jenkins
- vi $JENKINS_HOME/config.xml

```
<useSecurity>false</useSecurity>
```

- Start Jenkins

Cross-Site Request Forgery

- Malicious pages on the internet can forge requests to Jenkins
  - Even if your Jenkins is access controlled
  - Attacked needs to know your intranet host name and job name
- Not on by default for compatibility



Security implications of letting people build

- Build can be anything
  - Not only those who configure jobs, but those who write code
    - … which isn't any worse than "mvn install"
- Mitigation
  - Audit trail

Are your black projects really black?

- All builds run as the same user
  - They can interfere/interact with each other
  - Command line arguments, environment variables are all readable
  - Builds can see/modify the whole $JENKINS_HOME if run on master

- Mitigation
  - Isolate to different machines

CloudBees

Conclusions

- Securing Jenkins Web UI
  - Two orthogonal axes: authentication & authorization
  - CSRF
- Securing Jenkins from untrusted builds
  - Several mitigation techniques
  - Ultimately, you may have to split instances

Coming soon to Nectar

- Folder support
  - organize jobs into a hierarchical structure
  - Set ACL at folder
    - No need to individually set ACL at jobs
- Role-based access control support
  - Define roles, local groups
  - Control inheritance from ancestor ACLs

Q&A

Resources

CloudBees

http://www.cloudbees.com/

Nectar

http://nectar.cloudbees.com/

Try Dev@Cloud

https://grandcentral.cloudbees.com/account/signup

Register for news from CloudBees

http://www.cloudbees.com/company.cb

Upcoming training in London

http://cloudbees.com/training.cb

CloudBees