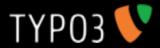


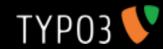
## Touchless Security with FLOW3

Andreas Förthner <andreas.foerthner@netlogix.de>

#### **Overview**

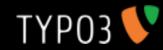


- About me
- What is a Security-Framework good for?
- Authorization with FLOW3
- Configuring security with ACLs
- Authentication
- Validation, Filtering and Application Firewall
- Current status and plans for the future



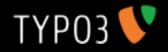
#### About me

- Andreas Förthner, born 04.12.1985
- Working with TYP03 in customer projects for netlogix in Nuremberg, Germany since 2003
- Studying computer science in Erlangen, Germany
- Member of the TYP03v5/FL0W3 Core Team since summer 2007



#### What is a Security-Framework good for?

- Support the developer in creating secure applications
- Security is handled at a central place
- Code is as secure as possible by default (?!)
- Provide a configurable and extensible architecture to secure any part of an application/scenario

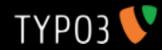


## Authorization



#### What should be protected?

- As we don't know what should be protected by the framework, we have to be able to protect anything
- The most general thing to protect are (PHP) functions
- Someone has to decide, if a functions is allowed to be executed in the current context
- If it's not allowed, the function call has to be aborted (e.g. by throwing a permission denied exception)



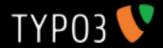
#### How to protect?

- We want to protect PHP functions
- So the developer has to call the security framework in every single PHP function ?!



#### Of course not... We have AOP!

- With Aspect Oriented Programming (AOP) we can intercept every method, if we need to, without touching the original function code
- That's why our security is "touchless" ;-)
- With AOP we can centralize security, although it is used almost everywhere in your application

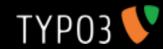


#### The security election – voting for access

- The access decision is based on the votes of so called access decision voters
- Access is only granted, if there is at least one "grant vote" and no "deny vote"
- You can implement your own voters, that may for example take function parameters into account
- Voters can abstain, if they are not responsible for the current method



## **Security Policy**



#### The security policy

- To tell the system who has access to which methods (the security policy), the standard way is to define Access Control Lists (ACLs)
- An ACL entry defines which <u>roles</u> (not users!) have which <u>privileges</u> on which <u>resources</u>
  - Privileges: ACCESS\_GRANT, ACCESS\_DENY, MYPRIVILEGE\_GRANT
  - Roles: ADMINISTRATOR, CUSTOMER
  - Resources: MyPackage::MyClass->delete.\*()



## Demo



### Authentication

#### **Authentication**



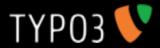
#### Identifying request partners

- Tell me your username and password and I'll tell you who you are!
- Really? Wouldn't it be better to identify him
  - by certificate?
  - or secure token?
  - or ask a LDAP directory?
  - or all together?
- Cool, but username and password are OK for the online shop...

**T3CON08** 

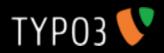
**Touchless Security with FLOW3** 

Inspiring people to **share** 

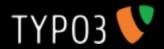


#### Managing authentication

- You can have more than one authentication provider in place (username/password, ldap, certificate, HTTP Basic, ...)
- A provider can be active for the whole application or just for a certain part (e.g. certificate authentication only in the extranet area)
- Configure, if it's enough to authentication one provider successfully or all
- Develop your own provider by implementing a simple interface and use it right away to authenticate your users

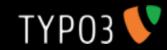


## Validation and Filtering



#### Never trust anyone

- ▼ In other words: Never trust any data!
- Especially GET/POST parameters, cli arguments, session Ids
- But that's no problem, you all check those things in your applications, right?



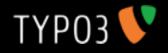
#### Accessing parameters in FLOW3

- You have to register them, otherwise they won't be available (No access to superglobals!)
- And you have to register a type!
- ▼ If the type is not correct you'll get an error instead of the parameter.
- You can register filters, to filter the parameter's value (remove HTML/JS code ...)



#### Which types are available?

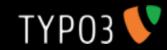
- ♣ Any you like ;-)
- A type is represented by a validator, that tells you if a given subject is of a certain type
- Implement your own validators and use them as a parameter type
- Of course there are many available in FLOW3 (Integer, Text, Email ...)



## Demo

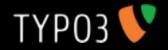


## Application Firewall



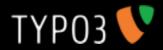
#### The first line of defense

- Block bad requests as soon as possible
- In the firewall you can identify request by so called request patterns (e.g. URL, IP address/range, ...)
- If a request matches a pattern you can define a security interceptor to be called (deny access, grant access, authentication required, ...)
- If no pattern matches, access is denied by default



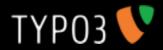
#### Guess what:

You can implement and configure your own patterns and interceptors, if you need something special



#### Summary

- The security framework does not solve all security related issues, but a lot of them
- It supports the developer to create secure applications, even if he's no security specialist
- ▼ It gives a strong basis to secure code right away, while leaving the flexibility to extend it to your special needs (access voters, authentication providers, request patterns, validators...)



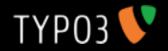
#### What's next?

- Implement the missing parts of the current architecture, especially add features like different authentication mechanisms...
- Add channel security (e.g. a password has to be transmitted over a SSL connection/channel)
- Implement a secure session handling
- Add logging to the whole thing
- **♥** Implement nice GUIs to configure policies
- Test in real world scenarios

**T3CON08** 

**Touchless Security with FLOW3** 

Inspiring people to **share** 



## Questions?

# TYPO3