# Keeping data safe:
## how to implement a secure repository

**Jenny Evans** - University of Westminster
**Tom Renner** - Lead Repository Developer, Haplo
**Dr Peter Tribble** - System Administrator, Haplo

**Traditionally Repository Managers and Repositories have focused on supporting open access and making all data and outputs available without restriction. But there are increasing requirements to restrict access, especially when extending repositories to handle research data, and the GDPR adds significant financial penalties, in addition to reputational damage, if you get it wrong. The University of Westminster and Haplo worked together to ensure the security of their repository.**

| Risk | Threat | Probability | Impact | Risk level (Probability x Impact) | Actions |
|---|---|---|---|---|---|
| Accidental access by a user to edit another user's record | Malicious or accidental changes | 1 | 1 | 1 | Review permissions model annually. Train Repository Editors on security implications and using repository functionality correctly. |
| Denial of service on hosting | Users unable to access their information; loss of administrative access | 1 | 5 | 5 | Review security practices of repository hosting company annually. |
| Data corruption | Irretrievable loss of data | 1 | 5 | 5 | Review repository hosting provider backup procedures annually to ensure best practice is being maintained. |
| Accidental access to an embargoed file | Breach of publisher contract. Institutional reputation. | 2 | 3 | 6 | Review permissions model annually. Train Repository Editors on security implications and using repository functionality correctly. Test access to embargoed files. |
| Unauthorised access to confidential data files | Depends on nature of material in the data file. Could be GDPR issue if access to identifiable personal information. | 2 | 5 | 10 | Review permissions model annually. Train Repository Editors on security implications and using repository functionality correctly. Test access to restricted data files. |

*Example threat model, a full assessment would include more threats and risks specific to an institution.*

**A threat model helps clarify priorities and the actions you need to take to reduce risk.**

**Risks are a threat to:**

**Confidentiality**
Who has the right to access information, and when

**Integrity**
Protecting the data against modification or corruption

**Availability**
Ensuring the information is available when require

## Evaluating systems for security assurance

### Repository software

**Is the permission system sufficiently flexible?**
A flexible permission system enables all users to use the system as intended, rather than working around the limitations of a limited and rigid permission model and potentially creating security risks, such as sharing logins or implementing overly broad permissions. Haplo's flexible permissions express all roles required in different parts of the institution, along with finely-grained rules to control access down to the field level.

**Has security been implemented correctly?**
The permissions you define must be precisely enforced and implemented correctly by the repository software. "Security" cannot be retro-fitted successfully to legacy systems written for a different threat model. Historically, repositories have not managed sensitive information, so have not always been developed under a process which prioritises security.

**Indicators to evaluate whether a repository is appropriate for the information you're storing:**

**Historical and current applications:**
As well as public research outputs, Haplo's repository technology is used to store sensitive information internal to an institution such as ethics applications and research funding information, and has been used for security critical applications for over 10 years.

**Development team:**
Haplo has been developed with a security-first mindset from the beginning, and the development process (and hosting) uses an ISO27001 security management system.

**Logging and monitoring:**
Extensive logs allow detection and response to suspicious activity, along with a complete audit trail of all actions.

**Software architecture:**
Haplo is built as a security aware platform which provides a "safety first" development environment, with enforcement at the lowest level of the platform, which is why Haplo always passes penetration tests without any issues.

### Hosting

**Where is the data held?**
Data should be hosted in the correct jurisdiction, to ensure the right laws and protections apply to it. Haplo servers are in the UK, and no part of the service uses third parties, making it easy to comply with GDPR regulations.

**What redundancy is there in the hosting service?**
The hosting service needs to continue operating in the face of failure, and must not lose your data.

Haplo host their own servers in multiple datacentres, and within each datacentre everything is duplicated: Firewalls, networking, internet connections, power suppliers, air conditioning, disc drives, and so on.

The failure of any component will not affect the availability of the service; as the system is configured to seamlessly and automatically route around any failures. Even if a datacentre is lost, all the data is continuously replicated so another site can take over.

**Who has access to the physical servers that hold the data?**
Your repository supplier will need to have access to your data to be able to deliver and manage the service, but this access needs to be tightly controlled. We allow only a limited number of Haplo employees, and their access is heavily authenticated and audited.

**How is data protected?**
Haplo uses the redundancy of their storage to ensure the data is always available, exactly as it was uploaded. As well as the cryptographic checksums in the Haplo software, Haplo stores all data using the ZFS filesystem which stores cryptographic checksums of every bit of data on all the servers. Any corruption will be automatically detected and can be repaired, either by using the redundant copy of the data from a different drive, or alerting an operator to restore data from the replica in the other datacentre.

Built into ZFS is a process called a "scrub", which automatically reads and verifies every single piece of data stored. Haplo runs a frequent scrub of every bit of data to proactively detect and repair errors.