Disk encryption is essential for physical computer security, but seldom used due to the trouble of remembering and typing a password at every restart. We describe Mandos, a program which solves this problem, its security model, and the underlying concepts of its design.

Any security system must have a clear view of its intended threat model – i.e. what threats it is actually intended to protect against; the specific choices and tradeoffs made for Mandos will be explained. Another danger of security system design is the risk of its non-use; i.e. that the system will not be used for some real or perceived drawbacks, such as complexity. The deliberate design choices of Mandos, involving low-interaction, “invisible” and automatic features, will be covered.
Mandos

http://www.recompile.se/mandos

TL;DL

aptitude install mandos
aptitude install mandos-client
Threat Model
Threat Model
Threat Model
Threat Model
Threat Model

No Server
Threat Model

No Server
Threat Model
The installer can guide you through partitioning a disk (using different standard schemes) or, if you prefer, you can do it manually. With guided partitioning you will still have a chance later to review and customise the results.

If you choose guided partitioning for an entire disk, you will next be asked which disk should be used.

Partitioning method:

- Guided - use entire disk
- Guided - use entire disk and set up encrypted LVM
- Manual

<Go Back>
Booting the kernel.

Loading, please wait...
  Volume group "glorfindel" not found
  Volume group "glorfindel" not found
Enter passphrase to unlock the disk /dev/hda2 (hda2_crypt):

Kernel alive
kernel direct mapping tables up to 100000000 @ 8000-d000
Threat Model

Argh it burns

Encrypted
New threat: non-use

Inconvenient
Burdensome

“I’ll do it some day”
New t

Password: 12345
Security needs to be transparent
Full Disk Encryption

/boot

(rest of disk)
Full Disk Encryption

/boot

(rest of disk)
Full Disk Encryption

/boot

(rest of disk)
Full Disk Encryption

/boot

(rest of disk)
Mandos

http://www.recompile.se/mandos

Servers provide passwords to each other
Mandos
http://www.recompile.se/mandos

Normal operation
Mandos

http://www.recompile.se/mandos

Normal operation
Mandos
http://www.recompile.se/mandos

Lockdown state
Administrator attention required
Mandos
http://www.recompile.se/mandos

/boot

(rest of disk)

Client

Server
Mandos

http://www.recompile.se/mandos

Client

Server

/boot

(rest of disk)
Mandos
http://www.recompile.se/mandos

/rest of disk

Client

Server
Mandos

http://www.recompile.se/mandos

/boot

(rest of disk)

Client

Server

Password

[Key symbols]
Mandos

http://www.recompile.se/mandos

//boot

(rest of disk)

Client

Server

<table>
<thead>
<tr>
<th>ID</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>
Mandos
http://www.recompile.se/mandos

```
<table>
<thead>
<tr>
<th>ID</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>
```
Mandos
http://www.recompile.se/mandos

/boot
3

(rest of disk)

Client

Server

<table>
<thead>
<tr>
<th>ID</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>
Mandos

http://www.recompile.se/mandos
Mandos

http://www.recompile.se/mandos

Client

/rest of disk

Server

<table>
<thead>
<tr>
<th>ID</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>
“GPG for data at rest. TLS for data in motion.”

If You're Typing The Letters A-E-S Into Your Code, You're Doing It Wrong

TLS has a “server” side and a “client” side, and the “server” side needs a key.
The TLS key can be a X.509 certificate
X.509:

“Someone tried to explain public-key-based authentication to aliens. Their universal translators were broken and they had to gesture a lot.”

— Peter Gutmann

Everything you Never Wanted to Know about PKI but were Forced to Find Out
Alternatively, the TLS key can be an OpenPGP key.
Mandos

http://www.recompile.se/mandos

TLS for data in motion

TLS “Server”
/boot
OpenPGP key 3
(rest of disk)

Client

TLS “Client”

Argh it burns

OpenPGP

<table>
<thead>
<tr>
<th>FP</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>🔐</td>
</tr>
<tr>
<td>5</td>
<td>🔐</td>
</tr>
<tr>
<td>7</td>
<td>🔐</td>
</tr>
</tbody>
</table>
“GPG for data at rest”? 
Mandos
http://www.recompile.se/mandos

TLS for data in motion
Mandos
http://www.recompile.se/mandos

TLS for data in motion

TLS “Server”
/boot
OpenPGP key 3
(rest of disk)

Client

TLS “Client”
OpenPGP

Server

Not encrypted!
Mandos

http://www.recompile.se/mandos

TLS for data in motion

Encrypt to *that* key!

Not encrypted!
Mandos
http://www.recompile.se/mandos

TLS for data in motion

TLS “Server”
/boot
OpenPGP key 3
(rest of disk)

TLS “Client”
OpenPGP
FP | Password
---|---
3  |  
5  |  
7  |  

Client

Encrypted!
Mandos
http://www.recompile.se/mandos

TLS for data in motion

TLS “Server”
/boot
OpenPGP key 3
(rest of disk)

TLS “Client”
OpenPGP

Server

Client

GPG for data at rest
Mandos
http://www.recompile.se/mandos

Done!
Mandos server

mandos (process)

Mandos client

mandos-client (process)

Client data table:
OpenPGP fingerprint, Encrypted password
OpenPGP fingerprint, Encrypted password
...

Initial RAM disk image

OpenPGP key pair (public & secret key)

(Unlocked by decrypted password)

Encrypted root disk

Local network

TLS-encrypted communication

OpenPGP public key (as part of TLS handshake)

Encrypted password
FAQ

Grabbing the Mandos client key from the /boot partition’s initramfs image really quickly?
Mandos
http://www.recompile.se/mandos

• In Ubuntu “universe” since 2009
• In Debian since 2011

aptitude install mandos
aptitude install mandos-client
Mandos

http://www.recompile.se/mandos

Mandos is a system for allowing servers with encrypted root file systems to reboot unattended and/or remotely. See the manual for more information, including an FAQ list.

Mandos is Free Software, licensed using the GNU General Public License v3 or later. (The Halls of Mandos is, in the fictional world of J. R. R. Tolkien, where the spirits of dead elves would go to be judged and possibly reincarnated. Similarly, the Mandos system allows “dead” servers to request reincarnation, which can be either denied or granted by the Mandos server.)

Download

Documentation

- Intro & FAQ
- Diagrams
- Manual pages
- Support
Mandos

http://www.recompile.se/mandos

http://ftp.recompile.se/pub/mandos/misc
Mandos

http://www.recompile.se/mandos

Disk encryption is essential for physical computer security, but seldom used due to the trouble of remembering and typing a password at every restart. We describe Mandos, a program which solves this problem, its security model, and the underlying concepts of its design.

Any security system must have a clear view of its intended threat model – i.e. what threats it is actually intended to protect against; the specific choices and tradeoffs made for Mandos will be explained. Another danger of security system design is the risk of its non-use; i.e. that the system will not be used for some real or perceived drawbacks, such as complexity. The deliberate design choices of Mandos, involving low-interaction, “invisible” and automatic features, will be covered.

`pub` 4096R/CA34C2C4 2013-10-05
Key fingerprint = 153A 37F1 0BBA 0435 987F  2C4A 7223 2973 CA34 C2C4
uid Mandos Maintainer Team <mandos@recompile.se>

Introductory text for people to read before the talk starts
Mandos
http://www.recompile.se/mandos

TL;DL

aptitude install mandos
aptitude install mandos-client

If you don’t have time to listen to this talk, here is how you install Mandos.

To continue with configuration, read /usr/share/doc/mandos-client/README.Debian.gz
Threat Model

What is the threat model?
Threat Model

Your server
Threat Model

Monster comes
Threat Model

Monster takes your server
Threat Model

Now you have no server
Threat Model

You and your users are sad
Threat Model

Monster eats your server’s data
Obvious solution: Use full-disk encryption

Debian installer shown
Prompted for password at boot
Data is now encrypted – Monster can still steal your server, but is now burnt by encrypted data
New threat: non-use

Inconvenient Burdensome
“I’ll do it some day”

Mail servers especially common to not be encrypted. Co-location make passwords inconvenient.
New threat:

Password: 12345

Passwords are hard to remember, especially if you seldom use them, like when rebooting a server.
Security needs to be transparent

Model after IPsec.
User behavior can stay unchanged.
A brief overview of how full disk encryption works.

The disk is divided into two parts, an encrypted main part, and a small unencrypted part, where the code to run at boot resides.
The code which runs at boot prompts for the password on the display.
The password is typed on the keyboard.
The password is used to unlock the encrypted part, and the code there is started to run the main system.

This is the *normal* full disk encryption procedure, and as you can see, it requires some manual work at the keyboard at each boot.
Mandos
http://www.recompile.se/mandos

Servers provide passwords to each other

What does Mandos do to make this better?

It makes the servers provide the passwords to each other.
An unlocked and running server provides the password needed by a locked server when the latter boots.
If the *other* server is up, that server can in turn provide the password for the first one.
If both servers are down, like when the Monster has taken them, they are both locked – no server can provide the password for the other.

This deadlock / bootstrap problem is a security feature!

This is the usual state when you really need encryption.
Let’s look at the communication in detail. We call the left computer – the one needing a password – the “client”, and reduce the right server to an icon to make room.
The encrypted server’s unencrypted part runs code which locates a server on the network, and connects to it.

(Using either Zeroconf to find it automatically on the local network, or by pre-specified IP address and port number.)
The server contains a table of data.
The table is a table of passwords, since the server can serve passwords for many clients.
Since there are many clients, we need some sort of identifier to uniquely identify them to map them to their corresponding key.
The ID is looked up in the table and the correct password is sent to the client.
The password is used to unlock the client’s encrypted disk, and it can continue to boot up normally.
Let’s look at it in more detail.

The key is sent from the server to the client over the network.
But in the network there be monsters.
“GPG for data at rest. TLS for data in motion.”

If You're Typing The Letters A-E-S Into Your Code, You're Doing It Wrong

There is a good quote and principle to follow in this informative dialogue.

We should therefore use TLS for the client-server communication.
TLS has a “server” side and a “client” side, and the “server” side needs a key.

A fundamental property of TLS is the server side absolutely needs a key, and the client side key is merely optional.
The TLS key can be a X.509 certificate

The X.509 system is most commonly used.
X.509:

“Someone tried to explain public-key-based authentication to aliens. Their universal translators were broken and they had to gesture a lot.”
— Peter Gutmann

Everything you Never Wanted to Know about PKI but were Forced to Find Out

But we really don’t want to use X.509 if we don’t have to.

There have been multiple security bugs in various TLS implementations resulting from the sheer complexity of the X.509 specification alone.
Alternatively, the TLS key can be an OpenPGP key

A seldom-used feature of TLS comes to the rescue.

(RFC 6091: *Using OpenPGP Keys for Transport Layer Security (TLS) Authentication*)
There are several things going on here.

- The client has an OpenPGP key, and the fingerprint of that key is used as the ID into the table of passwords. ("FP" means fingerprint)
- The TLS handshake is done “backwards”, since the client has an OpenPGP key, but the Mandos server doesn’t. So the Mandos client is the TLS server, and vice versa.
- The fingerprint of the client is proven by the TLS handshake, i.e. the client does have access to the secret portion of the OpenPGP key, and this can not be faked by the client. This means clients cannot ask for any other password than its own.
“GPG for data at rest”? 

But what about the other part of that quote?
What data do we have at rest?
The passwords in the table are unencrypted, i.e. any admin on the server can read them.
Flash of insight: We have an OpenPGP right there in the client; we can encrypt the passwords with *that* key!
Now the passwords are encrypted.
And we can fulfill the other part of the quote.
This is actually the finished design; this is how the Mandos network protocol works.
Older diagram attempting to show the same thing
Client-side process model with plugins for various input methods, etc.
FAQ

Grabbing the Mandos client key from the /boot partition’s initramfs image really quickly?

Threat model: people grabbing servers fast. Sophisticated attackers can and will do cold-boot.

Mandos shrinks the window of opportunity to default 5 minutes, customizable
Mandos
http://www.recompile.se/mandos

- In Ubuntu “universe” since 2009
- In Debian since 2011

aptitude install mandos
aptitude install mandos-client

This is mature, well-used code, and has been available and in use for many years.
Mandos
http://www.recompile.se/mandos

The Mandos home page has more information.
Mandos
http://www.recompile.se/mandos

http://ftp.recompile.se/pub/mandos/misc

Link to these slides (and more)