

Encriptación Centralizada de Correo Electrónico con Anubis

ENCRYPTACIÓN EGIPCIA

El servicio de manipulación de correo Anubis permite la encriptación centralizada del correo saliente. **POR**

DANIEL S. HAISCHT

Los expertos conocen los peligros de transmitir planos, datos personales y acuerdos confidenciales por Internet sin encriptar, aunque los usuarios finales raras veces se percatan de ello. Estos usuarios normalmente no utilizan herramientas como PGP, GnuPG [2] y S/Mime.

Los CTOs pueden lamentar su destino o tomar acciones: Los denominados servidores PGP proporcionan una gestión centralizada de las claves de usuarios y se encargan de los procesos de encriptación y desencriptación. Estos servicios eliminan la necesidad de tener que instalar y configurar los clientes

PGP en cada estación de trabajo, ahorrando bastante tiempo.

Los administradores de Linux tienen la posibilidad de usar programas de encriptación gratuitos como GPG-Relay [3] o Kurvert [4], y hay una serie de aplicaciones comerciales (como [5] y [6]) disponibles para Windows. Pero si se prefiere evitar el uso de aplicaciones especializadas, la mejor opción es elegir el programa de gestión de correo universal, GNU Anubis [1].

Anubis, que recibe el nombre de un antiguo dios egipcio, es un servicio de preprocesamiento SMTP. El servicio Anubis recibe los mensajes de un MUA o Mail



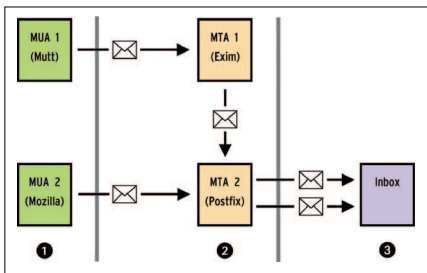


Figura 1: En una infraestructura normal de correo, cada cliente (MUA, Mail User Agent) pasa los mensajes al servidor SMTP (MTA, Mail Transfer Agent) que gestiona la entrega.

User Agent, (como el cliente Mutt mostrado en la Figura 2), y lo modifica antes de enviarlo al MTA o Mail Transfer Agent, (como el servidor Postfix también mostrado en la figura 2). Anubis puede procesar los mensajes de diversas formas, pero en este caso, una de sus características más útiles es la posibilidad de encriptar el correo usando GnuPG. Posee una herramienta de manipulación de correo experimentada, es una solución de Código Abierto que permite a los administradores instalar una infraestructura PGP centralizada. La descryptación automática de correo entrante o el uso del estándar PGP/Mime para procesar los ficheros adjuntos requieren el uso de programas como GPG-Relay o Kuvert, pero Anubis es la elección perfecta para encriptar y firmar el correo saliente.

La Funcionalidad SMTP Relay

Una de las técnicas más comunes para la encriptación o la firma de los mensajes de correo electrónico es usar en cada cliente un plugin PGP como Enigmail [9]. Una

herramienta como Anubis, por el contrario, proporciona una solución alternativa. Ofrece un procesamiento centralizado que a menudo es más fácil de gestionar si se tiene un gran número de usuarios. Con esta solución el cliente ya no tiene que enviar más el correo al servidor de correo, sino que lo dirige al software Anubis, el cual actúa como un proxy, modificando el contenido del mensaje, firmándolo o encriptándolo y por último enviándolo al servidor de correo.

Puede ejecutarse en un servidor independiente o en la misma máquina que el cliente o el MTA. Los usuarios tan sólo tienen que configurar sus clientes de correo electrónico de modo que Anubis sea su servidor de correo (véase el cuadro "Trucos SMTP").

La Instalación

Gracias a autoconf, el proceso de instalación y configuración de Anubis es realmente sencillo. Pero hay que asegurarse de especificar qué base de datos va a contener la información de los usuarios. En el modo Pixie, se puede además autenticar a los usuarios por medio de un servicio Ident ejecutándose localmente en las estaciones de trabajo. Esta variante no necesita de una base de datos relacional. Además, es posible compilar todos los módulos y decidir una instalación posterior:

- Regex (soporte para expresiones regulares)
- GSASL (autenticación de usuarios)
- El módulo para el servicio Ident se incluye siempre

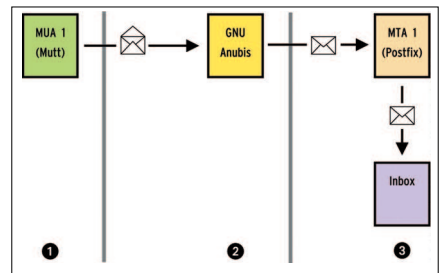


Figura 2: Anubis de GNU se sitúa entre el MUA y el MTA. Desde el punto de vista del cliente, Anubis es el servidor de correo. En esta posición, puede manipular el tráfico arbitrariamente.

- Guile (para los scripts)
- OpenSSL o GnuTLS (soporte SSL)
- GPG (GNU Privacy Guard) y GPGme
- MySQL (base de datos)
- PostgreSQL (base de datos)
- GDBM y ficheros de textos (estas variantes de bases de datos se incluyen siempre)

Otros componentes son opcionales y sólo tienen sentido en escenarios específicos:

- PAM (autenticación)
- Libwrap (TCP Wrapper)
- SOCKS (soporte SOCKS)
- NLS (Internacionalización)
- PCRE (expresiones regulares con la sintaxis de Perl).

Desafortunadamente, Anubis no tiene soporte para el servicio de directorio LDAP. Esta sería una característica útil, ya

Tabla 1: Abreviaciones utilizadas en el artículo

BLOB	Binary Large Object
GPG	GNU Privacy Guard
Guile	GNUs Ubiquitous Intelligent Language for Extensions
Mime	Multipurpose Internet Mail Extensions
MTA	Mail Transfer Agent (Mailserver)
MUA	Mail User Agent (Mailclient)
NLS	Native Language System
PAM	Pluggable Authentication Modules
PGP	Pretty Good Privacy
SASL	Simple Authentication and Security Layer
S/Mime	Secure Mime
SMTP	Simple Mail Transfer Protocol
SSL	Secure Sockets Layer
TLS	Transport Layer Security

Listado 1: Modo Pixie

```
01 #> Reading system file /usr/local/
    etc/anubis/anubisrc...
02 #> UID:0 (root), GID:0, EUID:0, EGID:0
03 #> GNU Anubis bound to<192.168.1.6:24
04 #> [68239] GNU Anubis is running...
05 #> [68239] Connection from 192.168.120.239:1310
06 #> [68244] IDENT: connected to
    192.168.120.239:113
07 #SERVER >>> 1310, 24 : USERID : UNIX :
    haischt(36)
08 #> [68244] IDENT: resolved remote user to
    haischt.
09 #> [68244] UID:65534 (nobody), GID:65534,
    EUID:65534, EGID:65534
10 #> [68244] Getting remote host information...
11 #> [68244] Connected to 192.168.1.6:25
12 #> [68244] Transferring message(s)...
13 #SERVER >>> 220 smtp. abyssworld.de ESMT
    Postfix (2.2.3)(46)
14 #CLIENT <<< 220 smtp. abyssworld.de (GNU Anubis
    v4.0) ESMT Postfix (2.2.3)(64)
```

Listado 2: Configuración Dixie

```

01 #---BEGIN CONTROL---
02 ## ...
03 #mode auth
04 ## ...
05 #---END---
06 #
07 #---BEGIN AUTH---
08 #smtp-greeting-message ESMTP
   Anubis (4.0.0)
09 #smtp-help-message help message
10 ### Simple text database:
11 ## sasl-password-db
   file:/usr/local/etc/anubisdb.txt
12 ### Relational MySQL Database
13 #sasl-password-db
                                mysql://mail:access4anubis@mysql
                                .abyssworld.de/mail;table=anubis
                                _user
14 #sasl-allowed-mech NTLM GSSAPI
                                DIGEST-MD5 CRAM-MD5
15 #---END---
16 #
17 #---BEGIN TRANSLATION---
18 ##translate [USER@]ADDRESS into
                                USERNAME
19 #translate
                                me@daniel.stefan.haischt.name
                                into haischt
20 #---END---

```

que normalmente tanto la información de los usuarios como las claves PGP están disponibles por medio de OpenLDAP o del servidor Active Directory. Durante mucho tiempo, la aplicación Anubis solamente hacía uso de bases de datos en formato texto o basadas en GDBM; el soporte para los sistemas relacionales es aún bastante nuevo para este programa. Así que esperemos que el soporte para LDAP esté disponible próximamente.

Se puede comprobar el fichero `config.log` después de completar la fase `./configure` para ver si los módulos se han configurado como se esperaba. Después de completar la instalación, tecleando `anubis-show-config-options` se puede consultar si los módulos requeridos están realmente disponibles.

Anubis espera encontrar sus datos de configuración en `/etc/anubisrc`. Esta ruta está codificada en `src/header.h`, pero se puede modificar la localización al vuelo usando `--altrc fichero`. La fase `make install` no crea realmente un fichero de

configuración; sin embargo, hay una plantilla en `examples/2anubisrc` en el paquete del código fuente de la aplicación.

Configuración Personal

Adicionalmente, los usuarios que necesiten enviar los correos por medio del servicio Anubis pueden crear un fichero `~/.anubisrc` en sus directorios home. El fichero `examples/1anubisrc` proporciona un ejemplo.

Para una prueba inicial, el servicio Anubis también dispone de un modo de depuración:

```

anubis --altrc
/usr/local/etc/anubis/anubisrc
--mode=transparent -v -D -f

```

Esto le indica a Anubis que se ejecute de forma transparente sin autenticación, proporcionando información detallada de salida (`-v`), produciendo datos de depuración (`-D`) y ejecután-

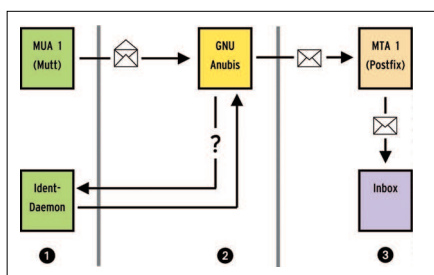


Figura 3: En el modo Pixie, Anubis solicita primero al servicio Ident de la estación de trabajo del emisor que identifique al usuario que ha abierto la conexión antes de aceptar y procesar el mensaje.

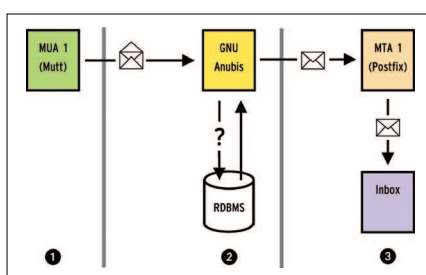


Figura 4: En el modo Dixie el servidor Anubis usa el método SMTP-AUTH para proporcionar una autenticación segura del usuario. El servidor necesita almacenar las credenciales del usuario en una base de datos.

dose en segundo plano (`-f`), en vez de evitar los canales estándar de entrada y salida, como normalmente hacen los servicios.

Autenticación Pixie

Anubis posee varias soluciones para identificar a los usuarios. El modo Pixie es una de las soluciones más simples para realizar la autenticación. Este modo se habilita con la entrada `mode transparent` en el bloque de control del fichero de configuración global. En esta configuración, un servicio Ident (protocolo AUTH) se ejecuta en la estación de trabajo del usuario. Anubis requiere que el usuario se autentique antes de manejar el correo (véase la Figura 3).

Sin embargo, esta técnica sólo tiene sentido en un escenario muy específico. El servidor tiene que confiar en el servicio Ident, además, esta clase de comprobación de seguridad del lado cliente sólo funciona si la estación de trabajo es manejada por un administrador responsable que conoce a sus usuarios y que puede estar seguro de que nadie manipulará la configuración de la red. Las buenas prácticas sugieren evitar el uso de este modo de operación.

Práctico pero Inseguro

Aunque el nombre del protocolo, AUTH, podría suponer otra cosa, Ident no realiza realmente una autenticación. El servicio está diseñado para seguir la pista de los ataques: el administrador del sistema atacado puede utilizar Ident para comprobar la ID de los usuarios de las máquinas de origen de las conexiones TCP. Basándose en estos datos, el administrador puede entonces contactar con el administrador responsable de la máquina origen. Muchas de las implementaciones de Ident simplemente hacen uso de una ID por defecto ([7], [8]).

El Listado 1 muestra una sesión de Anubis en el modo Pixie. Las líneas 6 (intento de autenticación Ident) y 8, donde el usuario de correo se corresponde con la cuenta Unix local `haischt`, son de interés. Después de la fase de autenticación, el servicio GNU Anubis pasa los mensajes al servidor de correo (Líneas de la 11 a la 14).

Los sistemas clientes basados en Windows XP a menudo poseen un corta-

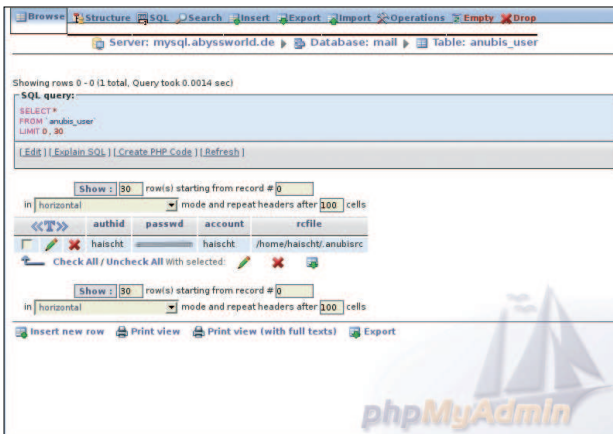


Figura 5a: La tabla MySQL contiene las credenciales del usuario **haischt**, que utiliza **Dixie** para autenticarse. La cuenta Unix para este usuario tiene el mismo nombre, y el fichero de configuración está localizado en **/home/haischt/anubisrc**.

uegos integrado que puede ocasionar problemas durante la fase de comprobación. En la configuración por defecto, el cortafuegos incomprensiblemente restringe todas las peticiones Ident al puerto 113. Esto tiene que modificarse (o bien desbloqueando el puerto 113 o deshabilitando el cortafuegos). Además, el programa de correo no puede hacer uso de los nombres de usuarios SMTP-AUTH en el modo Pixie.

Dixie, la Alternativa Preferida

El modo Dixie (Figura 4) es más nuevo y mejor que el modo Pixie. La alternativa Dixie está basada en el estándar

SMTP-AUTH. Anubis lee el nombre del usuario, la contraseña y otras credenciales de una base de datos (un simple fichero de texto o un sistema de base de datos relacional). La variante del fichero de texto incluye el ID del usuario (SMTP-AUTH-ID), la contraseña y, opcionalmente, la cuenta Unix y la ruta a la configuración específica del usuario (Tabla 2).

El fichero de texto dispone de una línea para cada usuario; los campos están separados por blancos. De acuerdo con la documentación oficial, los campos deberían separarse por el carácter “dos puntos”, pero esta sintaxis ya ni siquiera funciona en Anubis 4.0.

La variante del fichero de texto está bien al principio para realizar pruebas y para entornos pequeños. Si se necesita manejar un gran número de usuarios, probablemente se prefiera trabajar con una base de datos en vez de con un fichero de texto. La Figura 5a muestra una tabla con el esquema apropiado para esta base de datos.

El *authid* está definido como la clave primaria; los campos son todos de

texto. Como se puede ver en la Figura 5b, el contenido de la tabla refleja el del fichero de texto. El siguiente comando importa el fichero de texto a la base de datos MySQL:

```
anubisadm --create >
'mysql://mail:access4anubis@mysql.abysworld.de/mail;
table=anubis_test' <
/usr/local/etc/
anubis/anubisdb.txt
```

El comando especifica la tabla destino como un parámetro URL, la notación es confusa, ya que los parámetros no están separados por el carácter &, como lo estaría en una dirección HTTP, sino por puntos y comas. La URL y los parámetros deben ir entrecorchetados para impedir que los interprete la shell.

Integrando la Base de Datos

Para indicarle a Anubis que utilice la tabla nueva, el administrador tiene que cambiar el modo de autenticación de *mode transparent* a *mode auth* en la sección de control de la configuración global en *anubisrc* (véase el Listado 2, de la línea 1 a la 5). También es necesario añadir la ruta a las credenciales del usuario en la sección *AUTH* (líneas 7 a 15). El Listado 2 muestra cómo realizar esto para la base de datos MySQL. Además, los usuarios tienen que decirles a sus programas que utilicen la autenticación SMTP-AUTH cuando envíen un correo electrónico.

El Listado 3 muestra una sesión Dixie en funcionamiento. La línea 7 y siguientes muestran a un programa de correo dialogando con el servicio Anubis. El cliente puede usar el mecanismo SASL DIGEST-MD5 y CRAM-MD5 para la autenticación. Ambas soluciones transmiten un hash de la contraseña por la red en vez de una contraseña en texto en claro.

En la línea 9, se puede observar al programa de correo tratando con el comando *STARTTLS* para establecer una conexión SSL segura con el servicio Anubis. Se produce un fallo porque Anubis aún no está configurado para SSL seguro. En la línea 16, el programa de correo y Anubis acuerdan usar el mecanismo SASL CRAM-MD5. El ser-

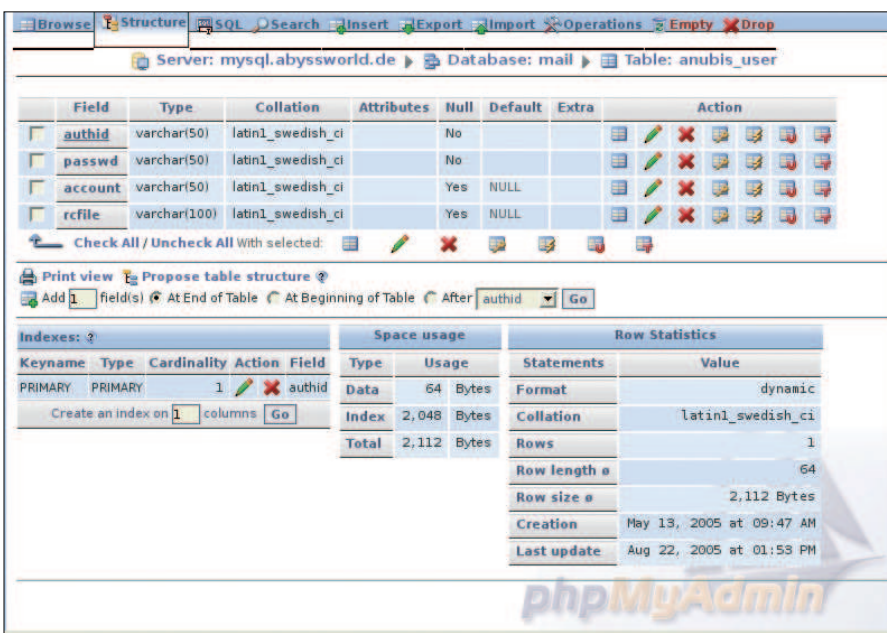


Figura 5b: PHPMyAdmin muestra la estructura de la tabla MySQL utilizada como el almacén de datos del modo Dixie de Anubis.

vicio busca entonces en la base de datos MySQL un registro que corresponda con las credenciales del usuario *haischt* y encuentra dicho registro (desde la línea 18 a la 20).

La línea 20 tiene un detalle interesante. En esta línea es donde Anubis realiza la correspondencia entre el usuario y la dirección de correo electrónico *me@daniel.stefan.haischt.name* con el usuario local *haischt*. La sección *TRANSLATION* se encarga de realizar la correspondencia entre la dirección de correo y los nombres locales (véanse las cuatro últimas líneas del Listado 2). La sección *TRANSLATION* del fichero de configuración también puede efectuar la correspondencia entre todas las direcciones de un dominio con un nombre compartido (traducción del Dominio al Usuario).

El modo Dixie es útil, ya que hace uso de estándares actuales como SMTP-

Trucos SMTP

Algunos programas de correo, por ejemplo Mutt, llaman al programa *sendmail* directamente para enviar el correo. En este caso, es imposible cambiar el servidor SMTP y el puerto en la configuración del cliente de correo, ya que el comportamiento de *sendmail* está configurado por el administrador. Además, algunos clientes de correo no soportan la autenticación basada en el mecanismo SMTP-AUTH.

Programas como MSMTMP [10] o ESMTP [11] pueden ayudar actuando como proxies de correo. Los usuarios pueden configurar estas herramientas de línea de comandos individualmente, con el fichero de configuración *~/esmtprc* para el caso de ESMTP:

```
hostname = ↗
anubis.abysworld.de:24
username = "haischt"
password = "access4anubis"
starttls = disabled ↗
```

El servicio envía el correo al servidor Anubis (*anubis.abysworld.de*, puerto 24) y autentica usando las credenciales del usuario. La encriptación STARTTLS está deshabilitada. Mutts necesita saber que debería llamar a ESMTP en vez de llamar a *sendmail*:

```
set sendmail=↗
"/usr/local/bin/esmtprc" ↗
```

Esta línea en el fichero de configuración de Mutt le indica al cliente de correo qué servicio usar.

AUTH y proporciona a los usuarios de correo electrónico sin cuentas UNIX acceso al servicio. También es cierto que la implementación actual del modo Dixie de Anubis tiene unos cuantos inconvenientes molestos:

Las contraseñas se almacenan en claro en la base de datos. El campo de la base de datos para el fichero de configuración de los usuarios de Anubis apunta a un fichero existente. Sería más práctico almacenar los parámetros en un campo BLOB o en una tabla separada.

Como Anubis posee las credenciales del usuario, ahora es posible el procesamiento de los mensajes con destino; por ejemplo, se puede configurar Anubis

para que firme o encripte el correo. Para permitir esta posibilidad, los usuarios han de añadir sus entradas en la sección *RULE* de sus ficheros *~/anubisrc*. Es bastante sencillo añadir atributos a la cabecera del correo electrónico utilizando la notación *add header [nombre] valor*:

```
add header[X-Processed-By] ↗
"GNU Anubis"
```

Tabla 2: Campos de la base de datos Dixie

Campo	Descripción
Authid	Corresponde con el nombre establecido por el usuario en las preferencias del programa de correo. Este campo se requiere para la autenticación SMTP.
Passwd	El usuario, además, establece una contraseña en el programa de correo cliente, que de nuevo es necesario para la autenticación SMTP.
Account	Corresponde con la cuenta Unix. Este campo relaciona el nombre del usuario de correo con la cuenta Unix correspondiente, con el objeto de encontrar un fichero de configuración de Anubis en el directorio home del usuario, por ejemplo.
Config	Este campo contiene la ruta (absoluta o relativa) del fichero de configuración de Anubis de este usuario. Las rutas relativas empiezan en el directorio home del usuario.

Listado 3: Sesión Dixie

```
01 #> Reading system configuration file
    /usr/local/etc/anubis/anubisrc..
    .
02 #> UID:0 (root), GID:0,EUID:0, EGID:0
03 #> GNU Anubis bound to 192.168.1.6:24
04 #> [68643] GNU Anubis is running...
05 #> [68643] Connection from 192.168.120.10:40501
06 #CLIENT <<< 220
    abysone.abysworld.de GNU Anubis ESMTP; Identify yourself(64)
07 #CLIENT >>> EHLO [192.168.121.2](22)
08 #CLIENT <<< 250-Anubis is pleased to meet you.(36)
09 #CLIENT <<< 250-STARTTLS(14)
10 #CLIENT <<< 250-AUTH DIGEST-MD5 CRAM-MD5 (31)
11 #CLIENT <<< 250 HELP(10)
12 #CLIENT >>> STARTTLS(10)
13 #[68647] anubis.pem: No such file or directory
14 #CLIENT <<< 454 TLS not available due to temporary reason(47)
15 #CLIENT >>> AUTH CRAM-MD5(15)
16 #SASL mech=CRAM-MD5, inp=NULL
17 #CLIENT <<< 334
    PDEOMTUONTMyOTUzMDA2MzI0MTIzLjBAbG9jYWxob3N0Pg==(54)
18 #CLIENT >>>
    aGFpc2NodCA5ZmQ1MDhkYTZyZzQ3ODRiOGUwMzZTNhMmUyM2VjZQ==(58)
19 #> [68647] Found record for `haischt'.
20 #> [68647] Authentication passed. User name haischt, Local user haischt. Welcome!
21 #CLIENT <<< 235 Authentication successful.(32)
22 #> [68647] UID:1001 (haischt), GID:20, EUID:1001, EGID:20
23 #> [68647] Reading user configuration file /home/haischt/.anubisrc...
24 #> [68647] Getting remote host information...
25 #> [68647] Connected to 192.168.1.6:25
26 #> [68647] Starting SMTP session...
27 #SERVER >>> 220
    smtp.abysworld.de ESMTP Postfix (2.2.3)(46)
```



GNOME



**GNOME Users and Developers
European Conference**

Vilanova 2006

*June, 24-30
Catalonia
Spain*

Cornerstone Sponsor



Media Partner



Co-organisers



- Approach Weekend
- Core GUADEC
- After Hours



- Showcase
- Topaz (GNOME 3.0)
- Tangle



- User
- Client
- Developer

www.guadec.org

Anubis también posee ejecución condicional. Si un correo tiene una cabecera *with-signature* que contiene un valor arbitrario (expresión regular.*), véase el cuadro “Formatos de Expresiones Regulares”), las siguientes líneas son todo lo que se necesita para eliminar la cabecera y añadir una firma basada en texto al final del mensaje:

```
if header [with-signature] >
:re ".*"
  remove [with-signature]
  signature-file-append yes
```

signature-file-append yes añade un separador -, seguido por el contenido del fichero *~/signature*, *body-append* añade el contenido de un fichero arbitrario al final del mensaje mientras que *body-clear-append* elimina el texto del mensaje original antes de realizarlo.

Comandos en la Línea Asunto

Algunos clientes de correo hacen que le resulte complicado a los usuarios

Formatos de Expresiones Regulares

Anubis entiende unas cuantas variantes de expresiones regulares. La configuración utiliza las siguientes etiquetas para identificar las expresiones regulares:

- *:regex* o *:re*: Expresiones regulares simples (Posix Extendido por defecto)
- *:basic*: Conmuta a Posix básico
- *:extended*: Conmuta a Posix extendido (por defecto)
- *:perl* o *:perlre*: Expresiones regulares compatibles con Perl (sólo si el soporte PCRE está compilado en Anubis)
- *:exact* o *:ex*: Sin expresiones regulares, el patrón debe ser una coincidencia exacta
- *:scase*: Distingue entre mayúsculas y minúsculas
- *:icase*: No distingue entre mayúsculas y minúsculas

Una sentencia puede contener una secuencia de etiquetas: *:perl:scase* significa expresiones PCRE y con distinción entre mayúsculas y minúsculas. La sentencia *regex:perl:scase* establece esta variante como opción por defecto.

añadir cabeceras. Anubis simplifica este proceso para los usuarios ya que analiza la línea asunto:

```
if header [Subject] >
"^ *\[sig\](.*)"
  remove [Subject]
  add [Subject] "\1"
  signature-file-append yes
```

La instrucción *if* comprueba si la línea asunto empieza con *[sig]* (Sintaxis Posix extendida, si no, no está precisamente especificada). Entonces elimina la línea asunto entera y añade una línea nueva que refleja parte de la línea original del asunto que sigue a *[sig]*. Para ello, *\1* referencia a la cadena entre corchetes pasada por la expresión regular.

La línea asunto es tan útil para comandos que Anubis posee su propia sintaxis para manipularla: Disparadores. Los usuarios pueden disparar eventos a través del correo añadiendo el comando al final de la línea de asunto seguido por dos caracteres “@”, como *cualquier cadena@@sign*. El siguiente disparador se encargará de todo:

```
trigger "sign"
  gpg-sign me>
@daniel.stefan.haischt.name
done
```

El comando *gpg-sign* firma el cuerpo del correo electrónico con la clave ID especificada. Para ello las claves GPG del usuario deben estar disponibles en el directorio *~/gnupg*, y los ficheros de configuración de los usuarios *~/anubis-rc* deben contener la contraseña GPG (*gpg-passphrase* “*micontraseña*”). GPG necesita las ID de las claves de los usuarios para realizar la encriptación. Un disparador extendido que analiza datos de la línea de asunto adicionales puede realizarlo mediante:

```
trigger : >
extended "^encrypt:(.*)"
  gpg-encrypt "\1"
  add [X-GPG-Comment] >
"Encrypted for \1"
done
```

La línea de asunto contiene el disparador *encrypt* seguido por la ID de la clave del receptor:*Hola Juan!@@encrypt:Receiver-Key*.

Anubis Astuto y Sutil

Tareas más complejas requieren de un lenguaje de programación más potente y Anubis proporciona esto por medio del lenguaje de script Guile (un dialecto de Scheme). Incluso se pueden utilizar programas externos para la manipulación del correo. En combinación con los comandos integrados, GNU Anubis soporta además técnicas extremadamente flexibles para procesar cabeceras de correo y contenidos. La fuerza real de este programa reside en la habilidad para realizar cualquier clase de manipulación sobre el correo que se pueda imaginar. Al mismo tiempo, el soporte integrado PGP/GnuPG ahorra bastante trabajo de configuración.

La autenticación de usuarios aún tiene bastantes cosas que mejorar. (El modo Pixie aplica la solución poco segura Ident y el modo Dixie almacena las contraseñas en la base de datos en claro). Las claves PGP y las configuraciones específicas de los usuarios se almacenan en el sistema de ficheros en vez de en la base de datos. Y además de esto, hay que añadir la carencia de soporte LDAP y PGP/Mime, junto con el hecho de que Anubis aún no es capaz de desencriptar los mensajes entrantes. Esperamos que en las próximas versiones del programa se mejoren todas estas características. ■

RECURSOS

- [1] GNU Anubis: <http://www.gnu.org/software/anubis/>
- [2] GNU Privacy Guard: <http://www.gnupg.org>
- [3] GPG Relay: <http://sites.inka.de/tesla/gpgrelay.html>
- [4] Kuvert: <http://www.snafu.priv.at/mystuff/kuvert/>
- [5] PGP Universal: <http://www.pgp.com/products/universal/>
- [6] GPG Shell (Windows): <http://www.jumaros.de/rsoft/>
- [7] Python Ident Daemon: <http://www.alcyone.com/software/fauxident/>
- [8] Windows Identd: <http://identd.dyndns.org/identd/>
- [9] Enigmail plugin: <http://enigmail.mozdev.org>
- [10] MSMTMP: <http://msmtp.sourceforge.net>
- [11] ESMTP: <http://esmtp.sourceforge.net>