TEC | Tecnológico de Costa Rica

Tarea 1

e-Mail

Marzo, 2025

Objetivo

Crear un Mail Server que permita la recepción, envío y consulta de correo electrónico utilizando la biblioteca Twisted, además de notificaciones a un usuario en NNTP

Datos Generales

- **Fecha de Entrega:** Lunes 17 de Marzo de 2025 antes de las 23:59:59 GMT-6.
- **Fecha de Revisión:** Sabado 22 de Marzo de 2025.
- **Lenguaje:** Python con Twisted
- **Recurso Humano:** Individual
- **Valor de la asignación:** 10 %

Profesor

Kevin Moraga
kmoragas@ic-itcr.ac.cr
Escuela de Computación

Introducción

Uno de los primeros protocolos en existir luego de la creación de ARPANET y posteriorente Internet es SMTP. Pero este protocolo, aún en la actualidad, es uno de los más conocidos y utilizados, después de HTTP. SMTP desde su creación intentó emular de una forma virtual el correo postal y es por ello que a la fecha posee problemas heredados desde su concepción. Sin embargo, este protocolo nos permite una comunicación en texto de forma distrubuida entre distintos dominios, sin la necesidad de una plataforma única y centralizada. Y es por ello, que es uno de los protocolos más utilizado en la industria para su trabajo diario.

Requerimientos

Requerimientos Funcionales

- **SMTP-Server:** Se debe crear un mail server el cual implemente el protocolo SMTP, este servidor se deberá de implementar con la ayuda de la biblioteca Twisted para python.
  - Este servidor permitirá el envío y recepción de correos electrónicos utilizando dicho protocolo.
  - Además este servidor deberá permitir la comunicación utilizando una capa segura como la de SSL (ahora TLS). Para ello puede utilizar la herramienta openssl.
  - El servidor deberá de verificar los dominios que acepta y recibir o rechazar correos electrónicos dirigidos a dichos dominios.
  - Además deberá permitir la recepción de archivos adjuntos utilizando el estándar MIME.
- **SMTP-client:** El cliente recibirá una lista de correos electrónicos, con el nombre del destinatario asociado, utilizando un archivo separado por comas (CSV) y además recibirá el servidor de correo electrónico que se utilizará para el envío de un mensaje.
  - El mensaje se le pasará como parámetro.
  - Dentro del mensaje se deberá permitir cargar el nombre (desde el CSV) del destinatario como variable, a través de un manejo de variables definido por el estudiante.
  - El objetivo de esto es que se puedan enviar correos masivos, y personalizados.
- **Servidor de IMAP:** Se deberá brindar el servidor de IMAP a un usuario, con el objetivo de que el usuario pueda acceder y leer su correo electrónico desde cualquier cliente que permita el uso de dicho protocolo.
  - Para ello se autenticará en el servidor, descargará los correos, los correos se eliminarán del servidor y se mostrarán en algún cliente de correo electrónico(ej. thunderbird).
  - El acceso a este servicio también deberá de ser configurado para que soporte una capa de cifrado como SSL o bien TLS.
- **NNTP-Notifier:** Este servicio se encargará de notificar a un usuario de NNTP que hay nuevo correo electrónico sin leer. La configuración será parte de la configuración de usuario.
- **Configuracion de Dominio:** El servicio de Correo Electronico debera de funcionar en un dominio adquirido por los estudiantes.

Requerimientos Técnicos

- El desarrollo se debe de realizar utilizando el lenguaje de programación Python para GNU/Linux.
- Es necesario utilizar la biblioteca Twisted para desarrollo del proyecto.
- El servidor debe de funcionar utilizando GNU/Linux

La sintaxis del smtp server será:

```
$ python smtpserver.py -d <domains> -s <mail-storage> -p <port>
```

La sintaxis del cliente de correo:

```
$ python smtpclient.py -h <mail-server> -c <csv-file> -m <message-file>
```

La sintaxis del IMAP server:

```
$ python IMAPserver.py -s <mail-storage> -p <port>
```

Nótese que el cliente debe de recibir los parámetros desde la terminal. El signo de dolar representa el shell del SO.

Aspectos Administrativos

Entregables

- Código fuente del programa que cumpla los requerimientos funcionales y técnicos.
- Binario del programa, compilado para una arquitectura x86.
- Fuente de la documentación en Latex o markdown.
- PDF con el Kick-off.
- PDF con la documentación.

Kick-off

Se deberá realizar un documento para la clase siguiente, con el siguiente contenido:

1. **Introducción**: Debe presentar el problema, utilizando una redacción propia. Incluyendo un pequeño esquema definiendo la estrategia para solucionar el problema asignado.
2. **Ambiente de desarrollo**: Indicar las herramientas que se utilizarán durante la elaboración de la tarea. Incluyendo entorno de desarrollo, forma de debugging, flujo de trabajo en un sistema de control de versiones.
3. **Control de Versiones**: Se debe de incluir y compartir el enlace al repositorio que se utilizará en el control de versiones seleccionado.

Documentación

Las siguientes son las instrucciones para la documentación. NO LA IMPRIMA. Además la documentación se debe de realizar utilizando Latex o Markdown.

1. **Introducción**: Debe presentar el problema, utilizando una redacción propia, discutido y redactado en el kick-off de la asignación.
2. **Ambiente de desarrollo**: Indicar las herramientas usadas para implementar la tarea.
3. **Estructuras de datos usadas y funciones**: Se debe describir las principales funciones y estructuras utilizadas en la elaboración de esta asignación.
4. **Instrucciones para ejecutar el programa**: Presentar las consultas concretas usadas para correr el programa para el problema planteado en el enunciado de la tarea y para los casos planteados al final de esta documentación.
5. **Actividades realizadas por estudiante**: Este es un resumen de las bitácoras de cada estudiante (estilo timesheet) en términos del tiempo invertido para una actividad específica que impactó directamente el desarrollo del trabajo, de manera breve (no más de una línea) se describe lo que se realizó, la cantidad de horas invertidas y la fecha en la que se realizó. Se deben sumar las horas invertidas por cada estudiante, sean concientes a la hora de realizar esto el profesor determinará si los reportes están acordes al producto entregado.
6. **Autoevaluación**: Indicar el estado final en que quedó el programa, problemas encontrados y limitaciones adicionales. Adicionalmente debe de incluir el reporte de commits de git. Por otro lado, también debe incluir una calificación con la rúbrica de la sección "Evaluación" y "Autoevaluación" con cada ítem evaluado de 0 a 10.
7. **Lecciones Aprendidas** del proyecto: Orientados a un estudiante que curse el presente curso en un futuro.

8. **Bibliografía** utilizada en la elaboración de la presente asignación.
9. Es necesario documentar el código fuente.

Evaluación

- kick-off: 5% adicional o 25% negativo si no se presenta.
- smtp-server: 15%
- smtp-client: 15%
- IMAP-server: 15%
- NNTP-notifier: 15%
- Modo SSL en el IMAP y smtp-server: 10%
- smtp-server en dominio: 10%
- Documentación utilizando latex o markdown: 20%
- Opcional 1: 5%. Permitir el envío, recepción y revisión de correos cifrados, bajo el modo de PGP/MIME, utilizando el servidor creado.
- Opcional 2: 5%. La creación de interfaz gráfica para el cliente-smtp.

Autoevaluación

Realice una autoevaluación utilizando una escala [*1*] *Muy Malo,* [*2*] *Malo,* [*3*] *Regular,* [*4*] *Bueno y* [*5*] *Muy bueno*, de los siguientes rubros:

[1] [2] [3] [3] [5]	Aprendizaje del protocolo SMTP.
[1] [2] [3] [3] [5]	Aprendizaje del protocolo IMAP.
[1] [2] [3] [3] [5]	Aprendizaje del protocolo NNTP.
[1] [2] [3] [3] [5]	Aprendizaje de la capa SSL/TLS
[1] [2] [3] [3] [5]	Organización de Tiempo.

Aspectos Adicionales

Aún cuando el código y la documentación tienen sus notas por separado, se aplican las siguientes restricciones:

1. Sí no se entrega documentación, automáticamente se obtiene una nota de 0.
2. Sí el código no compila se obtendrá una nota de 0, por lo cuál se recomienda realizar la defensa con un código funcional.
3. El código debe ser desarrollado en el lenguaje especificado al inicio del enunciado, en caso contrario se obtendrá una nota de 0.
4. Sí no se siguen las reglas del formato del envío a través de Google Classroom se obtendrá una nota de 0.
5. La revisión de la documentación será realizada por parte del profesor, no durante la defensa del proyecto.
6. Cada grupo tendrá como máximo 20 minutos para exponer su trabajo al profesor y realizar la defensa de éste, es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo cuál se recomienda tener todo listo antes de ingresar a la defensa.
7. Cada excepción o error que salga durante la ejecución del proyecto y que se considere debió haber sido contemplada durante el desarrollo del proyecto, se castigará con 2 puntos de la nota final de la presente asignación.
8. Cada grupo es responsable de llevar los equipos requeridos para la revisión, si no cuentan con estos deberá avisar al menos 2 días antes de la revisión al profesor para coordinar el préstamo de estos.
9. Durante la revisión podrán participar asistentes, otros profesores y el coordinador del área.
10. Cualquier indicio de copia será calificado con una nota de 0 y será procesado de acuerdo al reglamento.