PY: Skrypt Powiadomień Mailowych dla systemu Cacti

mail_cacti.py

```
import time
import smtplib
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
import mysql.connector

# Global variable to store the last processed data
last_processed_data = None

def download_data():
    # Replace these values with your database connection details
    db_config = {
        'host': '127.0.0.1',
        'user': 'cactiuser',
        'password': 'PASSWORD',
        'database': 'cactidb',
    }

    try:
        # Connect to the database
        connection = mysql.connector.connect(**db_config)

        # Create a cursor object
        cursor = connection.cursor()

        # SQL query
        query = "SELECT * FROM plugin_thold_log;"

        # Execute the query
        cursor.execute(query)

        # Fetch all rows
        rows = cursor.fetchall()

        return rows

    except mysql.connector.Error as e:
        print(f"Error: {e}")
        return None

    finally:
        # Close the cursor and connection
        if 'cursor' in locals() and cursor is not None:
            cursor.close()
        if 'connection' in locals() and connection.is_connected():
            connection.close()
```

```
def send_email(subject, message, to_emails):
    from_email = "python@ttcomm.net"  # Replace with your email address

    # Create the MIME object
    msg = MIMEMultipart()
    msg['From'] = from_email
    msg['To'] = ', '.join(to_emails)
    msg['Subject'] = subject

    # Attach the message to the MIME object
    msg.attach(MIMEText(message, 'plain'))

    # Connect to the SMTP server on 192.168.6.190, port 25
    try:
        server = smtplib.SMTP('192.168.6.190', 25)
    except Exception as e:
        print(f"Failed to connect to the SMTP server: {e}")
        return

    # Send the email without authentication
    try:
        server.sendmail(from_email, to_emails, msg.as_string())
        print("Email sent successfully!")
    except Exception as e:
        print(f"Failed to send email: {e}")
    finally:
        # Close the connection
        server.quit()

# Function to convert epoch time to human-readable format
def convert_epoch_to_time(epoch_time):
    return time.strftime('%Y-%m-%d %H:%M:%S',
time.localtime(epoch_time))

# Run the script in a loop with a delay of 1 minute between executions
while True:
    # Usage example
    data_from_db = download_data()

    # Check if data is retrieved and if it's different from the last
processed data
    if data_from_db:
        new_records = []

        if last_processed_data:
            for new_entry in data_from_db:
                if new_entry not in last_processed_data:
                    new_records.append(new_entry)
        else:
            new_records = data_from_db
```

```
        if new_records:
            subject = "Cacti Alert Script"
            messages = []

            for entry in new_records:
                timestamp = convert_epoch_to_time(entry[1])  # Assuming
timestamp is at index 1
                messages.append(f"{timestamp}: {entry[-1]}")

            message = "\n".join(messages)
            to_emails = ["MAIL@MAIL.MAIL","MAIL@MAIL.MAIL"]  # Replace
with the recipient's email addresses

            send_email(subject, message, to_emails)

        # Update the last processed data
        last_processed_data = data_from_db

    else:
        print("No data retrieved from the database.")

    # Wait for 1 minute before the next iteration
    time.sleep(60)
```