PayPal Certificate Update Procedure

The Coastsidearc website employs a 'Let's Encrypt' Certificate. This certificate expires 90 days after it is issued. However, DreamHost renews the website certificate 30 days before the expiration date.

The PayPal application also uses a certificate which is generally the same as the website certificate. It is recommended that the PayPal certificate be updated shortly after the website certificate is updated in order to synchronize the certificate management process. There is a 30 day window within which this can be done.

Although the website certificate updating is automatic, the PayPal certificate must be manually updated. The process for updating the PayPal certificate is provided below.

1. Log into the website server shell
   ```
   ssh ai6bb@coastsidearc.org (get password from webmaster)
   ```
2. Change Directory to access the certificates
   ```
   cd ~/node-test-01/carc-paypal/.ca
   ```
3. Check the expiration date of the certificate
   ```
   openssl  x509 -enddate -noout -in server.crt
   ```
4. Log into DreamHost (www.dreamhost.com) under username pvatkins@pacbell.net. Navigate to Websites/Secure Certificates/Settings. Under the 'Certificate', select Copy to Clipboard. Then open an editor in the website shell.
   ```
   nano crt.txt
   ```
   Paste the certificate text into the file, write out, and exit.
5. In a similar manner, copy the 'RSA Private Key' to the clipboard and open another file in the website shell
   ```
   nano key.txt
   ```
   Paste the RSA Private Key text into the file, write out, and exit.
6. Overwrite the existing server.crt and server.key files with the newly created data.
   ```
   mv crt.txt server.crt
   mv key.txt server.key
   ```
7. In the DreamHost panel, select 'Manage Servers'. You should see our server, vps56613, listed. To the far right of that line, click on the vertical row of 3 dots. Then click on the option 'Restart Server'. Then click 'Restart' button that will appear when asked if you are sure. Note: Your ssh connection to the server will be closed when the server is restarted.
8. Reconnect to the server shell
   ```
   ssh ai6bb@coastsidearc.org
   ```
9. Change directory to access the PayPal application
   ```
   cd ~/node-test-01/carc-paypal/
   ```
10. Kill and then restart the PayPal process
    ```
    pm2 kill
    ```

```
pm2 -n CARC_PAYPAL -i 2 start "npm run prod"
```

11. Check the status of the PayPal process to see if everything has succeeded.

```
pm2 status
```

In the table below, some columns of lesser interest have not been duplicated. The tables represent the results of an example test case. PIDs may vary. In the tables below, some columns of lesser interest have not been duplicated.

Initial Status upon task creation

id	name	mode	pid	uptime	↺	status	mem	user
0	CARC_PAYPAL	fork	52857	**0s**	**15**	**online**	31.8mb	ai6bb
1	CARC_PAYPAL	fork	53856	**0s**	**0**	**online**	29.4mb	ai6bb

And after 20 seconds...17h...

id	name	mode	pid	uptime	↺	status	mem	user
0	CARC_PAYPAL	fork	0	**0**	**15**	**errored**	0b	ai6bb
1	CARC_PAYPAL	fork	52858	**17h**	**0**	**online**	41.6mb	ai6bb

Namespace[default], version[N/A], cpu[0%] and watching[disabled] are the fields not duplicated in the above tables.

Google Gemini AI Review of PayPal Certificate Update Procedure

This is a well-structured procedure for updating the PayPal application's certificate. It clearly outlines the necessity for manual updates to synchronize with the DreamHost-managed website certificate. However, there are several areas that could be refined for clarity, security, and operational robustness.

Here's a detailed review:

Strengths of the Document:

- **Clear Purpose:** The introduction effectively explains why and when the PayPal certificate needs manual updating.
- **Logical Flow:** The steps generally follow a logical sequence from identifying the current certificate to deploying the new one and restarting the application.
- **Specific Commands:** Provides explicit shell commands, which is helpful for the operator.
- **Verification Point:** Includes a step to check the certificate's expiration date (Step 3) and the application status (Step 11).

Areas for Improvement and Concern:

1. **Certificate Chaining (Step 4):**
    - The document assumes DreamHost's "Certificate" field provides the certificate in the exact format needed for server.crt. Let's Encrypt certificates usually require an intermediate chain. It should be clarified if server.crt is intended to be just the end-entity certificate or the full chain (server certificate + intermediate certificates). If DreamHost provides them separately, the procedure should include combining them.
2. **Private Key Security (Step 5 & 6):**
    - While using nano to paste a private key is functional, it carries minor risks (clipboard history, visibility).
    - **Crucially Missing:** After moving the new key to server.key (Step 6), permissions must be set to protect it. Add:

Bash

chmod 600 server.key

This restricts access to the owner only.

3. **Backup Existing Files (Before Step 6):**
    - It's highly recommended to back up the current server.crt and server.key before overwriting them. This allows for a quick rollback if the new certificate/key pair causes issues. Add a step like:

Bash

cp server.crt server.crt.bak_$(date +%Y%m%d%H%M%S)

cp server.key server.key.bak_$(date +%Y%m%d%H%M%S)

4. **VPS Restart (Step 7):**
    - Restarting the entire VPS (vps56613) to apply a certificate change for a single Node.js application is likely **overkill and unnecessarily disruptive**. This will cause downtime for all services running on that VPS.
    - If the Node.js application (carc-paypal) directly uses these certificate files for its HTTPS server, restarting only the Node.js application (as done in Step 10) should be sufficient for it to load the new certificate.
    - If a reverse proxy (like Nginx or Apache) on the VPS is terminating SSL using these certificates, then that specific proxy service should be reloaded (e.g., sudo systemctl reload nginx), not the entire VPS.
    - **Recommendation:** Remove the VPS restart unless there's a specific, unstated dependency requiring it. Rely on the application restart in Step 10.
5. **pm2 Command Precision (Step 10):**
    - pm2 kill: This command stops **all** applications managed by PM2 for the current user (ai6bb). If other PM2-managed applications are running, they will also be killed, which might be unintended.
    - **Recommendation:** Use a more targeted command for the CARC_PAYPAL application:
        - To restart: pm2 restart CARC_PAYPAL
        - For a clean stop and start if restart doesn't pick up changes (some Node.js HTTP servers might cache certs):

Bash

pm2 stop CARC_PAYPAL

pm2 start CARC_PAYPAL # Or the full: pm2 -n CARC_PAYPAL -i 2 start "npm run prod"

- If the intent of pm2 kill is a full refresh of only this app because delete is preferred:

Bash

pm2 delete CARC_PAYPAL

pm2 -n CARC_PAYPAL -i 2 start "npm run prod"

6. **Interpreting pm2 status Example (Step 11):**
    - The example output is concerning and confusing:
        - **Errored Instance:** The "after 20 seconds...17h..." table shows CARC_PAYPAL instance 0 as errored with 0b memory and pid 0. The document needs to address:
            - Why does this happen? Is this expected?
            - What should the user do if they encounter this? Troubleshooting steps are needed.
        - **Uptime Discrepancy:** Instance 1 shows 17h uptime "after 20 seconds" of a supposed restart. This suggests the pm2 kill and start sequence might not have fully reset this instance, or the example is not from an immediate post-restart scenario. This should be clarified, as it implies the restart process might not be behaving as expected or the example is misleading.
        - **Multiple Instances:** The -i 2 flag starts two instances. If one consistently errors, it might indicate an issue with the application running in cluster mode or a port conflict. Consider if running a single instance (-i 1 or default) would be more stable unless clustering is essential and proven functional.
7. **Post-Update Verification:**
    - The procedure should include a step to actively verify that the PayPal integration is working correctly with the new certificate. This could involve:

        - Checking application logs for successful PayPal communications.
        - Performing a test transaction or using a feature that interacts with PayPal.
        - Externally checking the served certificate (e.g., using openssl s_client -connect yourdomain.com:port if the application serves HTTPS directly on a known port).

8. **Minor Details:**
    - The phrase "In the tables below, some columns of lesser interest have not been duplicated" is repeated.
    - Consider creating temporary files like crt.txt and key.txt in a /tmp directory or ~/tmp to keep the .ca directory cleaner, then moving them.

Recommendations Summary:

- **Add:** Backup of old certificate files.
- **Add:** chmod 600 server.key command.
- **Clarify:** Certificate chain requirements for server.crt.
- **Revise/Remove:** The full VPS restart (Step 7) in favor of only restarting the necessary application or service.
- **Refine:** pm2 commands to be more targeted (e.g., pm2 restart CARC_PAYPAL).
- **Address:** The errored instance in the pm2 status example, provide troubleshooting, and clarify the uptime discrepancy.
- **Add:** A final verification step to confirm PayPal functionality.

By addressing these points, the procedure can become safer, less disruptive, and provide better guidance for the user, especially if encountering issues.