

Troubleshooting Checkout Timeouts & "Invalid Invoice ID"

Proxmox KVM module **WHMCS**

[Order now](#) | [Download](#) | [Community](#)

Symptoms

During customer checkout (especially when ordering multiple virtual machines or when account credit is applied) or when an admin clicks **Accept Order**:

1. **HTTP 504 Gateway Timeout:** The browser waits for 30–60 seconds, after which a white screen or a `504 Gateway Timeout` is displayed.
2. **"Invalid Invoice ID":** The user is redirected to `https://your-domain/cart.php?a=complete`, which renders a blank page with the plain text error message:

Invalid Invoice ID

3. **Order stuck in Pending with Paid Invoice:** In the WHMCS Admin area (`orders.php?action=view&id=...`), the order status is `Pending`, the Invoice is marked `Paid` / `Complete`, but all services remain in `Pending` status and were never provisioned.

Root Cause Analysis

What Happens During Checkout

When a client completes an order, WHMCS executes a chain of heavy operations sequentially inside the same HTTP POST request (`|cart.php?a=checkout|`):

1. **Order & Invoice Creation:** Creates the order record in `|tblorders|`, services in `|tblhosting|`, and invoice in `|tblinvoices|`.
2. **Payment Application & Hooks:** If the invoice is paid immediately (e.g. via account credit or instant payment gateway), WHMCS applies payment, triggers `|InvoicePaid|` hooks, and executes fraud/sanctions screening.
3. **PDF Generation & SMTP Delivery:** WHMCS generates a PDF invoice via TCPDF and connects synchronously to the mail server via SMTP to dispatch the "Customer Invoice" email (this typically takes **5-10 seconds**).
4. **Payment Confirmation Email:** WHMCS sends another email confirming payment (another **5-10 seconds**).
5. **Module Provisioning:** WHMCS loops through each order item and calls `|CreateAccount|`.
6. **Session Finalization:** WHMCS stores `|$_SESSION['orderdetails']|` containing `|InvoiceID|` and redirects the browser to `|cart.php?a=complete|`.

Why the Error Occurs

By default, many web servers (Nginx reverse proxies, cPanel, Plesk, HestiaCP) have a strict **30-second proxy timeout** (`|proxy_read_timeout 30s|` or `|fastcgi_read_timeout 30s|`).

If the combined duration of PDF generation, SMTP delivery, and payment processing reaches **30 seconds**, the web server forcefully drops the HTTP connection and returns an HTTP 504 Gateway Timeout.

Because the connection was aborted mid-flight:

- The PHP process is terminated by the server before reaching session finalization and module provisioning.
 - When the client's browser reloads or lands on `|cart.php?a=complete|`, `|$_SESSION['orderdetails']['InvoiceID']|` is empty or null.
 - WHMCS queries the database with an empty invoice ID, fails to locate the record, and halts with `|Invalid Invoice ID|`.
 - The services were never invoked by the module and remain in `|Pending|`.
-

Resolution

1. Enable Smart Hybrid Bulk Provisioning in Module Settings

In the WHMCS Admin area:

1. Navigate to **Addons > PUQ Proxmox KVM > Settings > General**.
2. Locate **Bulk Provisioning Queue Threshold** (`bulk_provision_threshold`).
3. Set the threshold to `2` (the default).

How this helps: When an order contains more than 2 VMs (i.e. 3 or more), the module bypasses heavy synchronous Proxmox API calls during checkout. Instead, it queues the VMs for background provisioning via cron in approximately **0.04 seconds**, marks the services `Active`, and defers the Welcome Email until the VM is actually running. This keeps the total provisioning time during checkout under 1 second.

2. Increase Web Server & PHP Timeouts (Recommended: 120s – 300s)

To accommodate PDF generation and external SMTP latency during multi-item checkouts, increase script and proxy timeouts to **at least 120 seconds** (recommended: **300 seconds**):

PHP Configuration (`php.ini` & `www.conf`)

In your active PHP configuration (e.g. `/etc/php/8.x/fpm/php.ini`):

```
max_execution_time = 120      ; Recommended: 120 to 300
max_input_time = 120          ; Recommended: 120 to 300
```

In your PHP-FPM pool configuration (e.g. `/etc/php/8.x/fpm/pool.d/www.conf`):

```
request_terminate_timeout = 300s
```

Nginx Configuration (Reverse Proxy with Apache)

If Nginx proxies requests to Apache:

```
# Add to your Nginx site configuration or /etc/nginx/conf.d/timeout.conf:
proxy_connect_timeout 300s;
proxy_send_timeout    300s;
proxy_read_timeout    300s;
```

Nginx Configuration (Direct FastCGI / PHP-FPM)

If Nginx connects directly to PHP-FPM:

```
# Add to your location ~ \.php$ block or /etc/nginx/conf.d/timeout.conf:
fastcgi_connect_timeout 300s;
fastcgi_send_timeout    300s;
fastcgi_read_timeout    300s;
```

Apache Web Server

In `/etc/apache2/apache2.conf` or `/etc/httpd/conf/httpd.conf`:

```
Timeout 300
```

After modifying the configuration, reload the web server and PHP-FPM:

```
systemctl reload nginx
systemctl reload php8.x-fpm
systemctl reload apache2
```

How to Fix an Existing Stuck Order

If an order was already interrupted and remains in `Pending` status:

1. Open the WHMCS Admin area and navigate to **Orders > List All Orders**.
2. Click on the affected Order ID (e.g. Order #1308).
3. Verify that the invoice is marked `Paid` / `Complete`.
4. At the bottom of the page, ensure **Run Module Create** and **Send Welcome Email** are checked.

5. Click **Accept Order**.

With **Smart Hybrid Provisioning** active, all services will be queued in less than a second, and the background cron (`|Process Virtual Machines|`) will deploy the virtual machines one by one.

Revision #2

Created 8 October 2026 19:20:45 by Ruslan

Updated 8 October 2026 19:21:16 by Ruslan