# 仿真器实验

## 物理层

# 实验内容

- 信号传递时间的计算
- 数据传输时延的观察
- 有线信号错误率
- 无线信号传递范围

# 仿真器实验
# 信号传递时间的计算

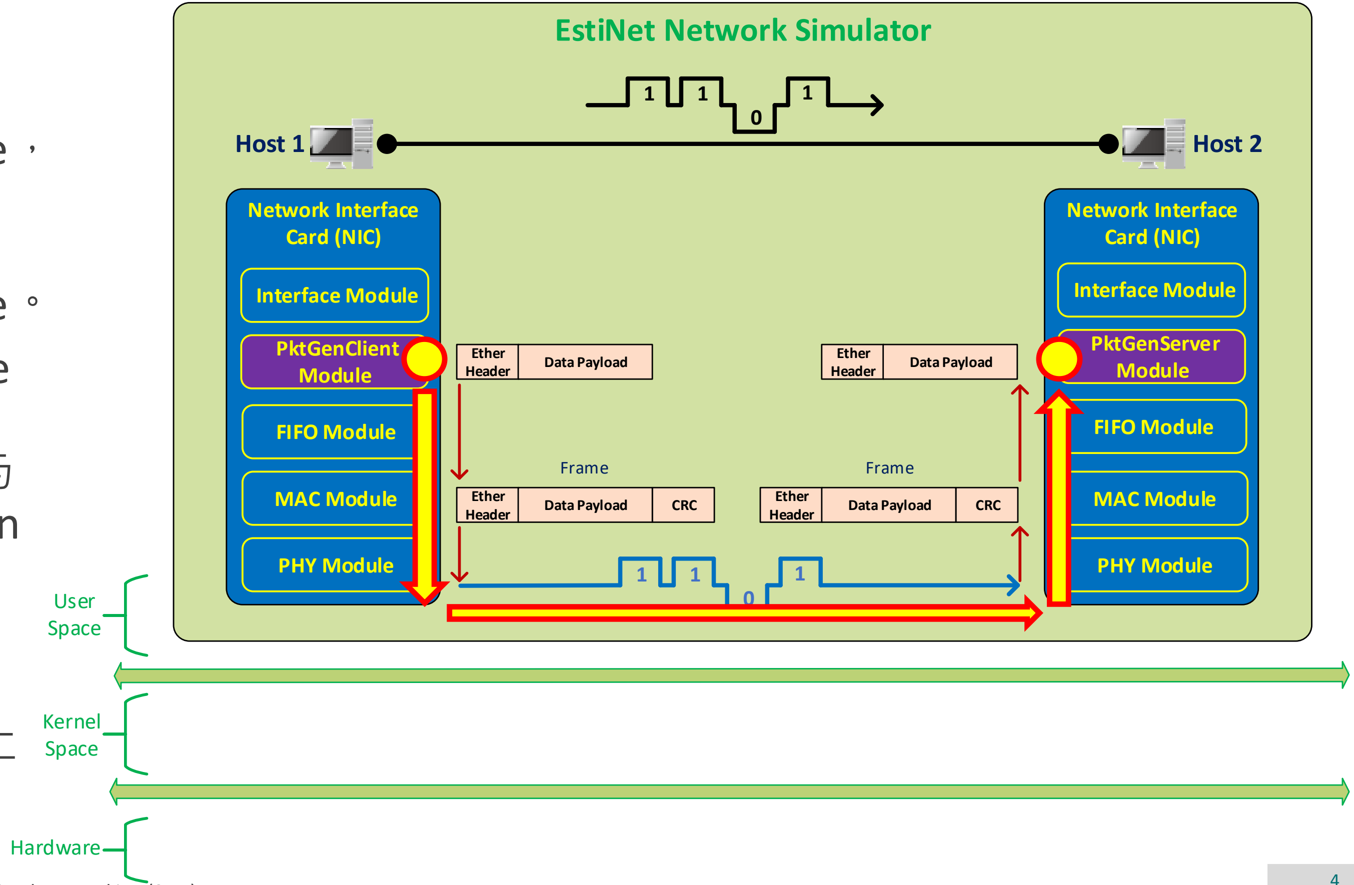
< Simulation Case >

trans_time_and_prop_delay.xtpl

trans_time_and_prop_delay_comparison.xtpl

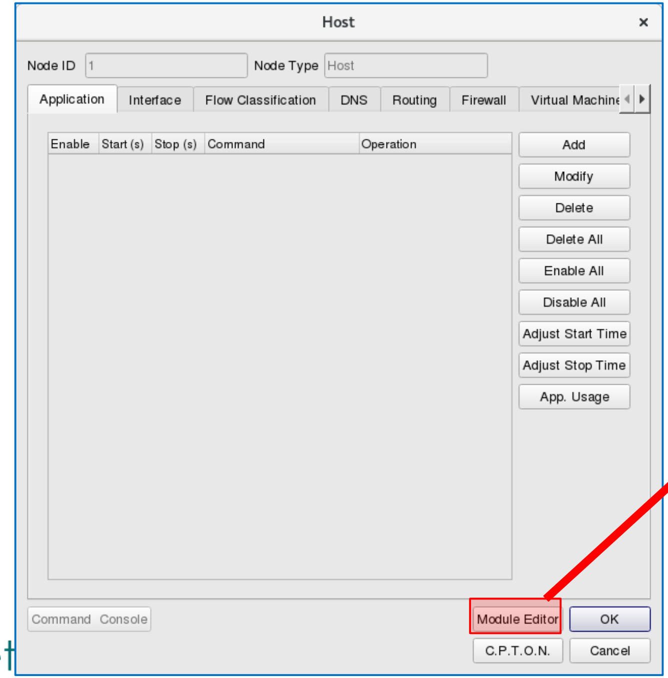
# 仿真环境架构

- 在信号传送端 Host 1 上使用 PktGenClient Module 来发送 Frame，在信号接收端 Host 2 上使用 PktGenServer Module 来接收 Frame。
- 在两端的 PHY Module 上会仿真传输时间 (Transmission Time) 与传播时延 (Propagation Delay) 的效果。
- 留意在 PktGenClient 与 PktGenServer 两个 Module 下方都要加上 FIFO Module。

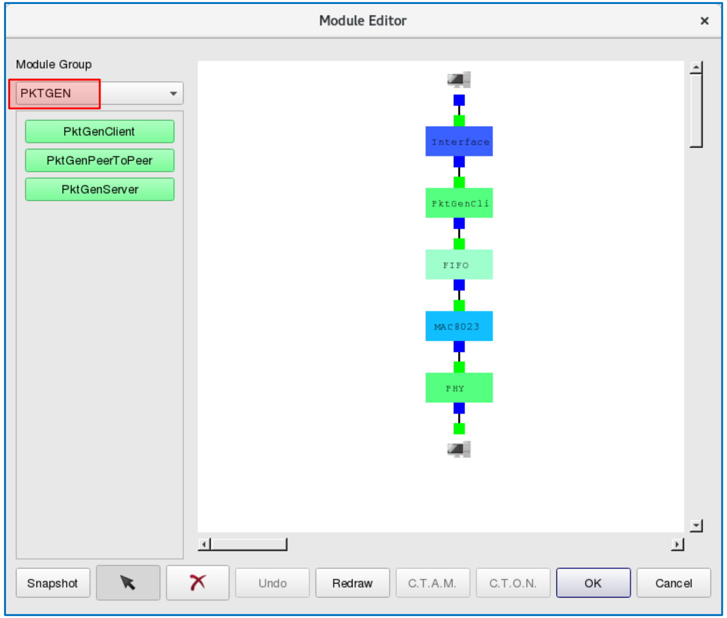


# 在 E Stage，于节点编辑器 (Node Editor) 操作视窗上开启模组编辑器 (Module Editor)

- 在 E Stage，于任一个节点 (Node) 上连击鼠标左键两下，可开启节点编辑器。按下 “Module Editor” 的按键，可开启模组编辑器。

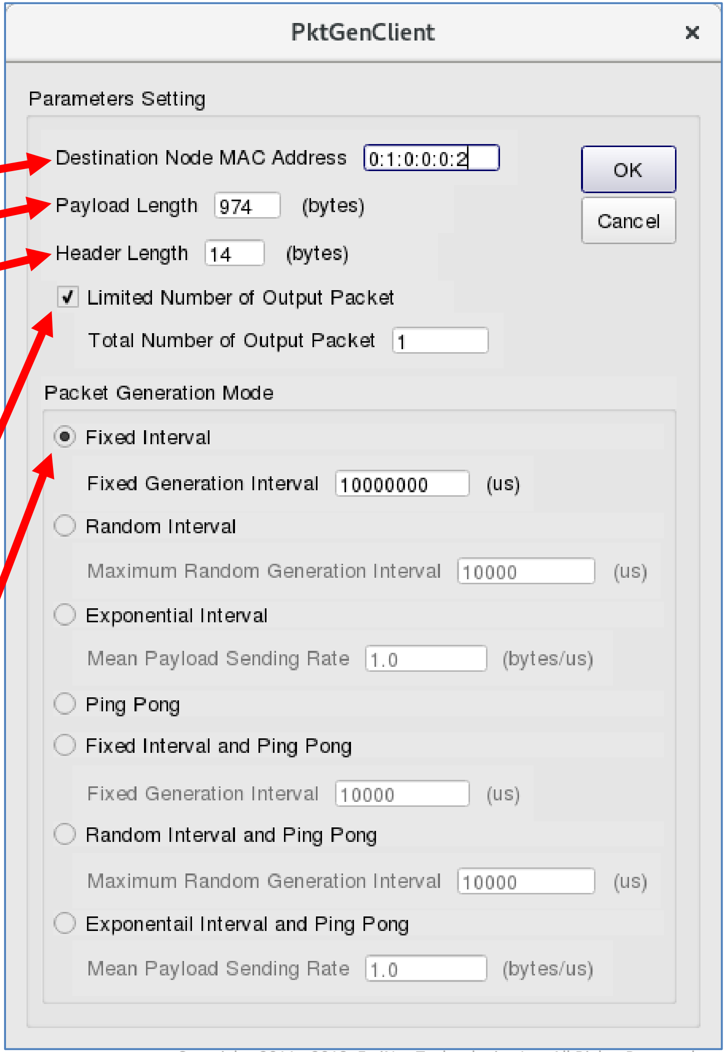



- 在模组编辑器左边可以根据群组分类来找到可以外加的模组，例如在 PKTGEN 群组中可以找到 PktGenServer 与 PktGenClient 模组，将所需模组挂到右边的协议栈 (Protocol Stack) 中。



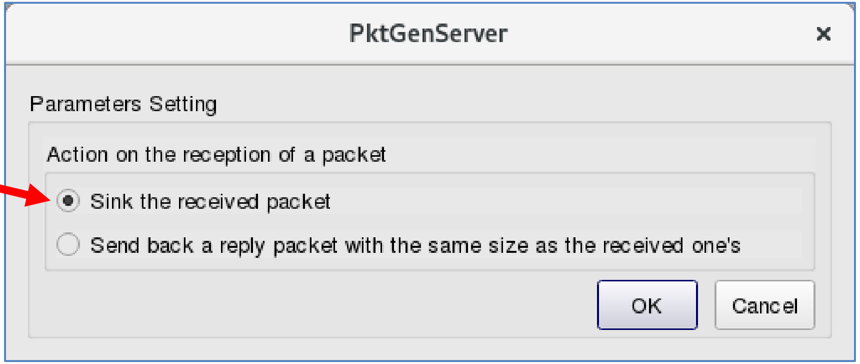
# PktGenClient Module 的设定

- 输入接收端 Host 2 的 MAC 地址。
- 输入数据长度 974 bytes。
- 输入表头长度 14 bytes。
- 数据长度 + 表头长度 = 988 bytes。
- 在 MAC8023 Module 中会再加上 7 bytes 的 Preamble、1 byte 的 Start Frame Delimiter 与 4 bytes 的 CRC Checksum，因此，整个 Frame Size 就是 1000 bytes。
- 输入在仿真过程中，限制只送出一个 Frame。
- 输入当仿真开始后，经过 10,000,000 us 才送出第一个 Frame。

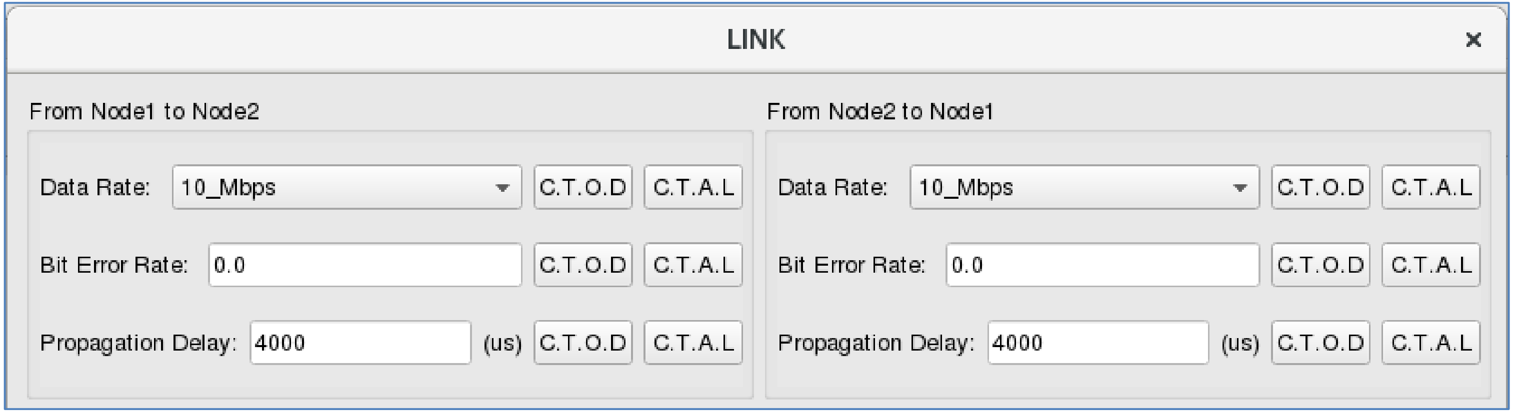


# PktGenServer Module 的设定

- 输入当接收端 Host 2 收到 Frame 后，直接将此 Frame 丢掉。



# 连接两端的 Link 上的设定



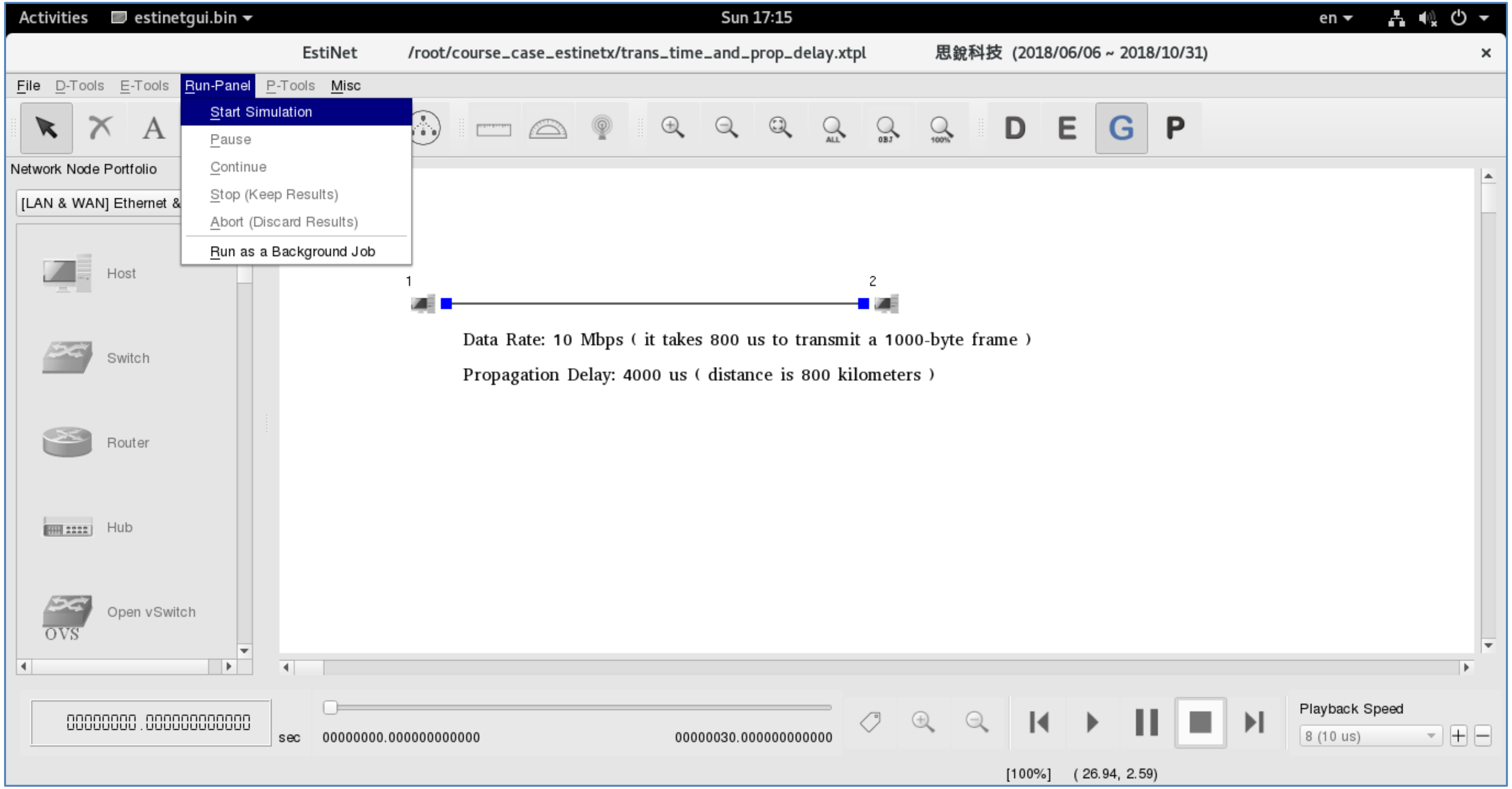

- Data Rate = 10 Mbps

  Transmission time of a 1000-byte frame = (8*1000)/(10 * 10^6) = 0.0008 sec

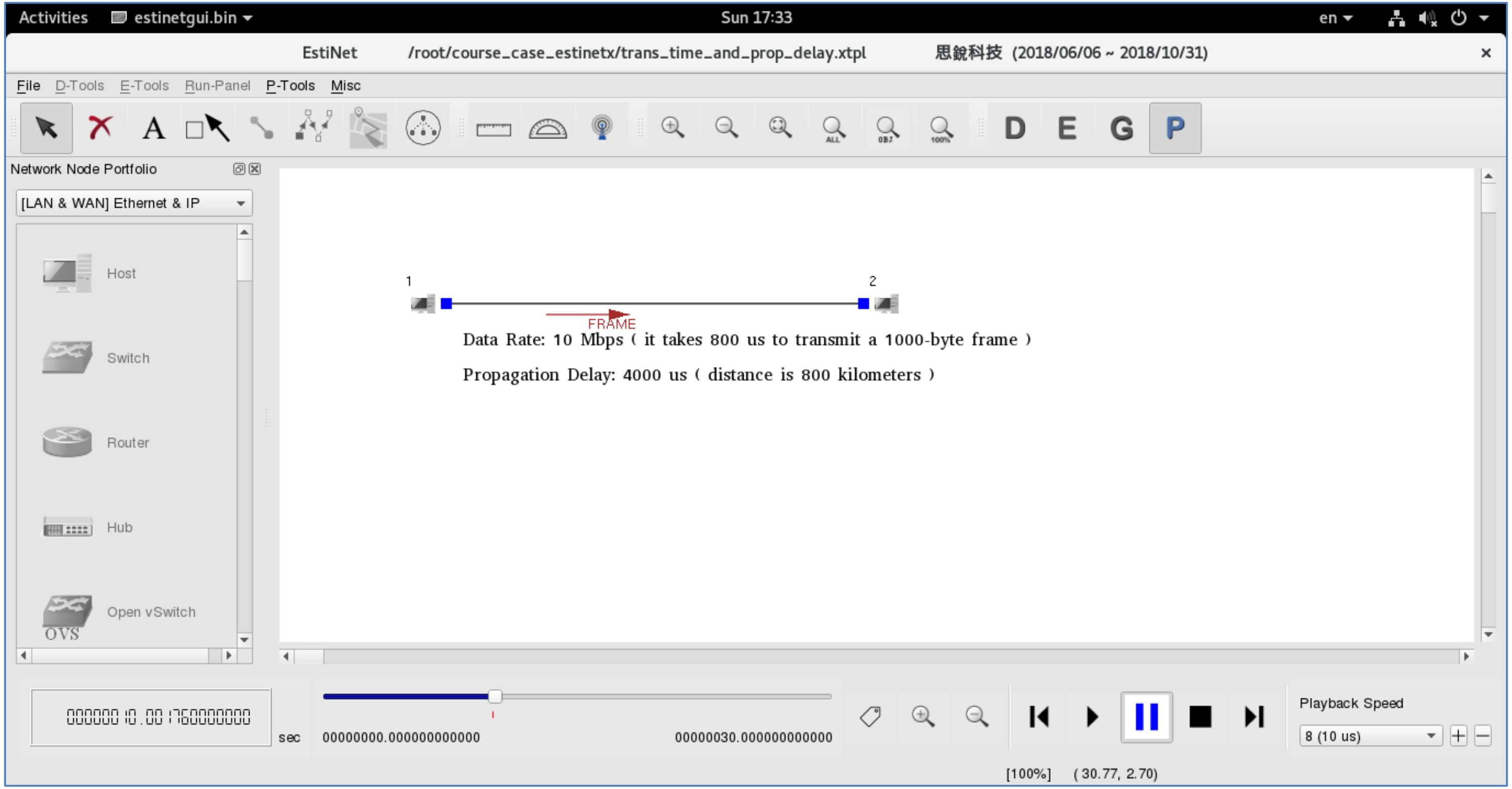
- Propagation Delay = 4000 us = 0.004 sec

  Distance = 0.004 * (2 * 10^8) = 800000 meters = 800 kilometers

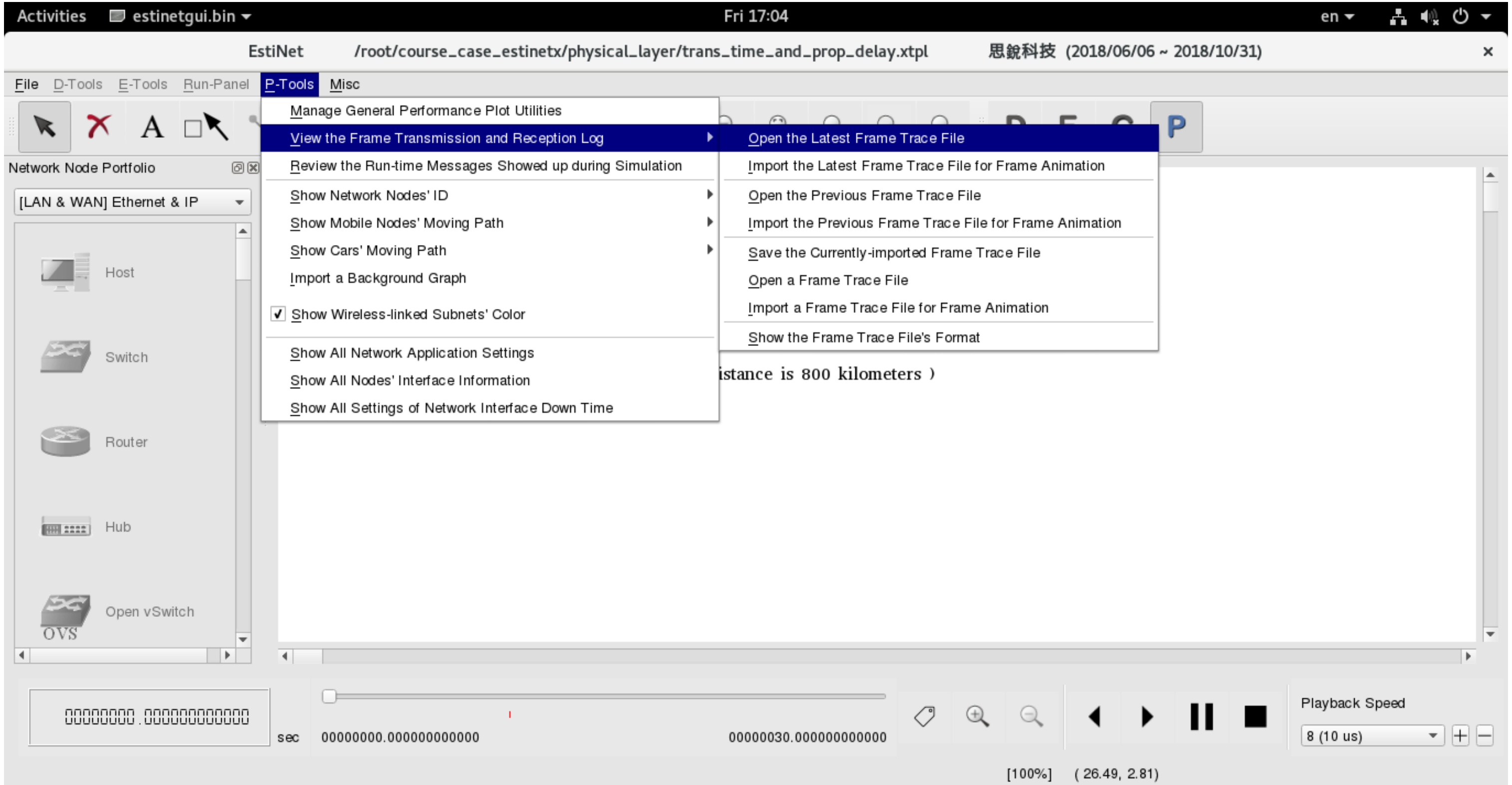
# 切换到 G Stage 产生仿真设定档，然后执行仿真



# 仿真结束后，在 P Stage 观看仿真结果



# 在 P Stage，打开 Frame Trace File 来观看 Frame 送收的记录



# 在 Frame Trace File 中观察传输时间与传播时延，两者相加就是信号传递时间

事件持续时间

事件起始时间

时间单位为 Picosecond，也就是 10^(-12) Second。

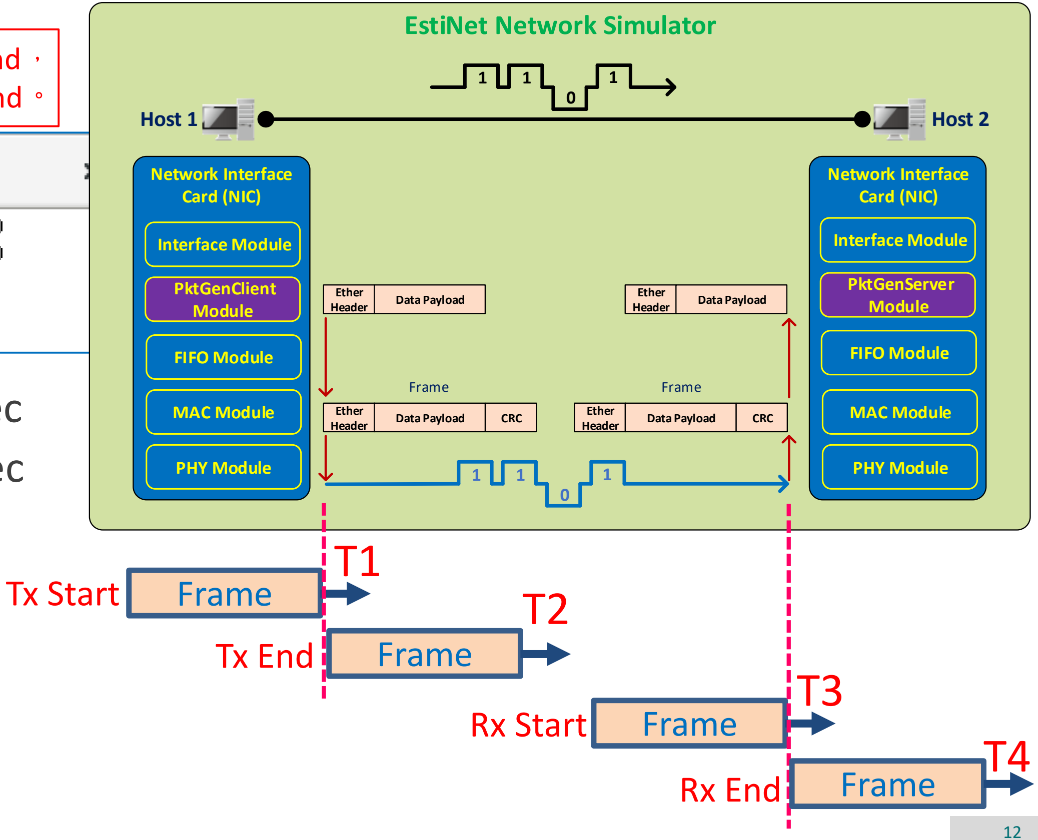
less

```
802.3 TX  10000000000000 800000000 DATA  <0 0> <1 2> 18 1000 0 NONE  0
802.3 RX  10004000000000 800000000 DATA  <0 0> <1 2> 18 1000 0 NONE  0
/tmp/frame_trace_file.log (END)
```

- T1 = 10,000,000,000,000 * 10^(-12) = 10 sec
- T2-T1 = 800,000,000 * 10^(-12) = 0.0008 sec

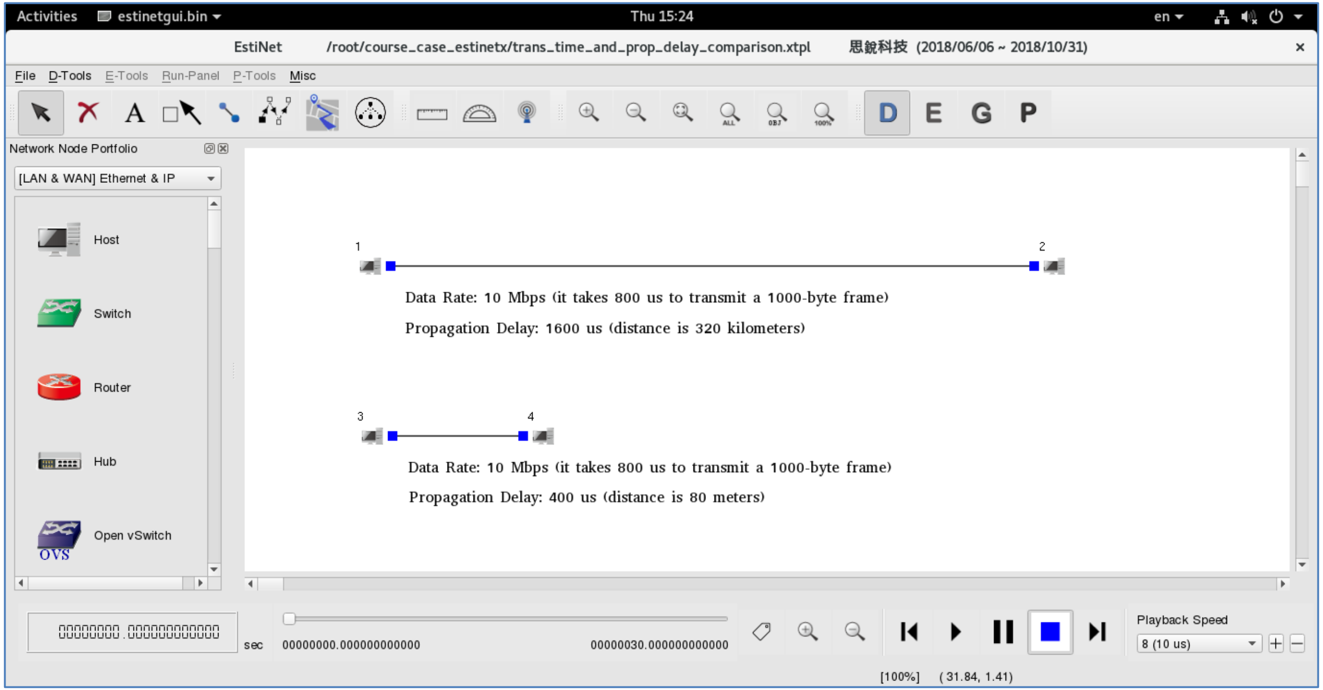
  Transmission Time = 0.0008 sec
- T3 = 10,004,000,000,000 * 10^(-12)

  = 10.004 sec
- T3-T1 = 0.004 sec
- Propagation Delay = 0.004 sec



# 变化题

- 探讨传输时间 (Transmission Time) 与传播时延 (Propagation Delay) 的大小关系。利用仿真器来改变传输时间或传播时延后，透过仿真结果来进行观察。
  - Transmission Time > Propagation Delay
  - Transmission Time < Propagation Delay



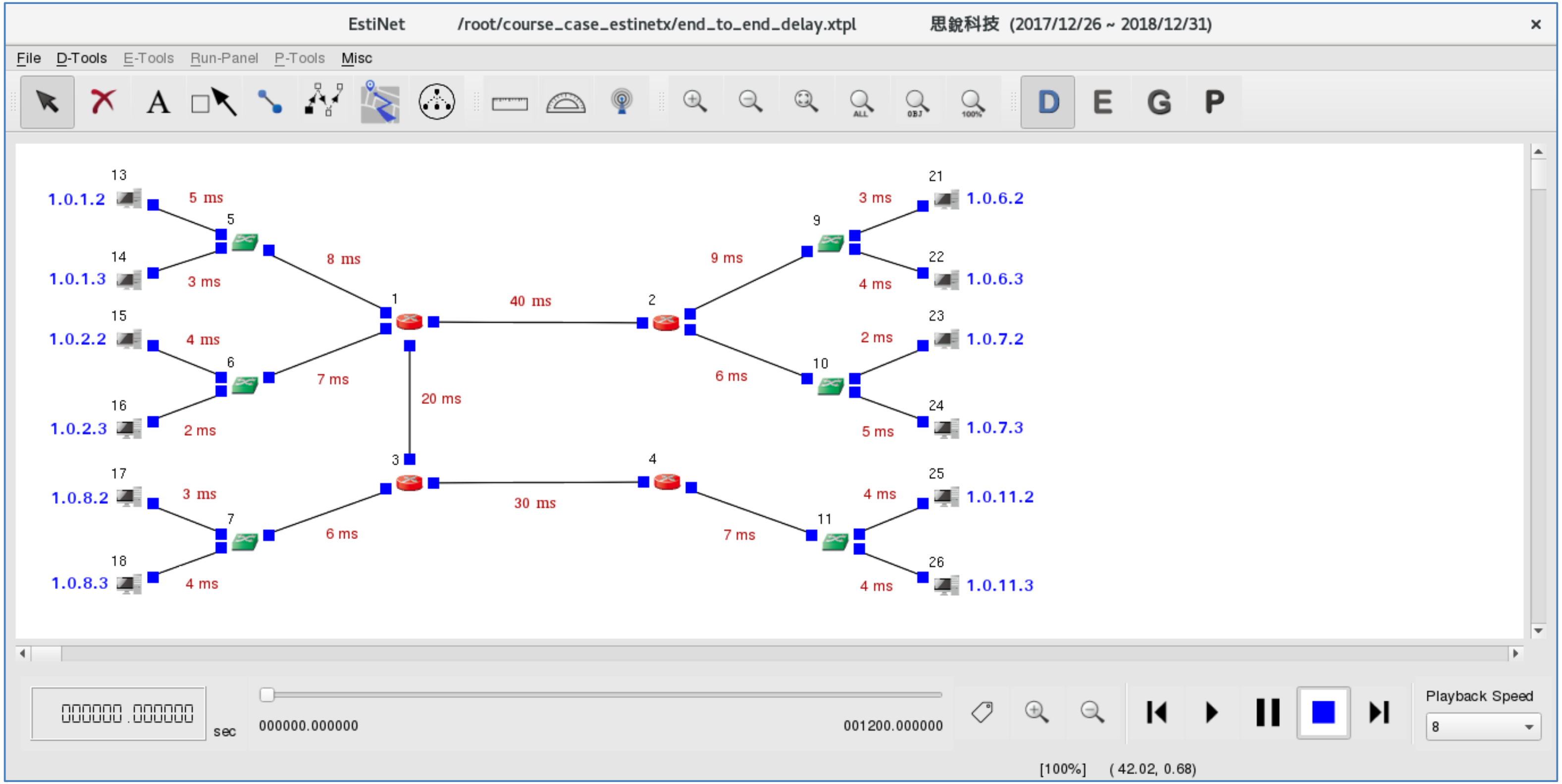
# 仿真器实验
# 数据传输时延的观察

< Simulation Case >

end_to_end_delay.xtpl

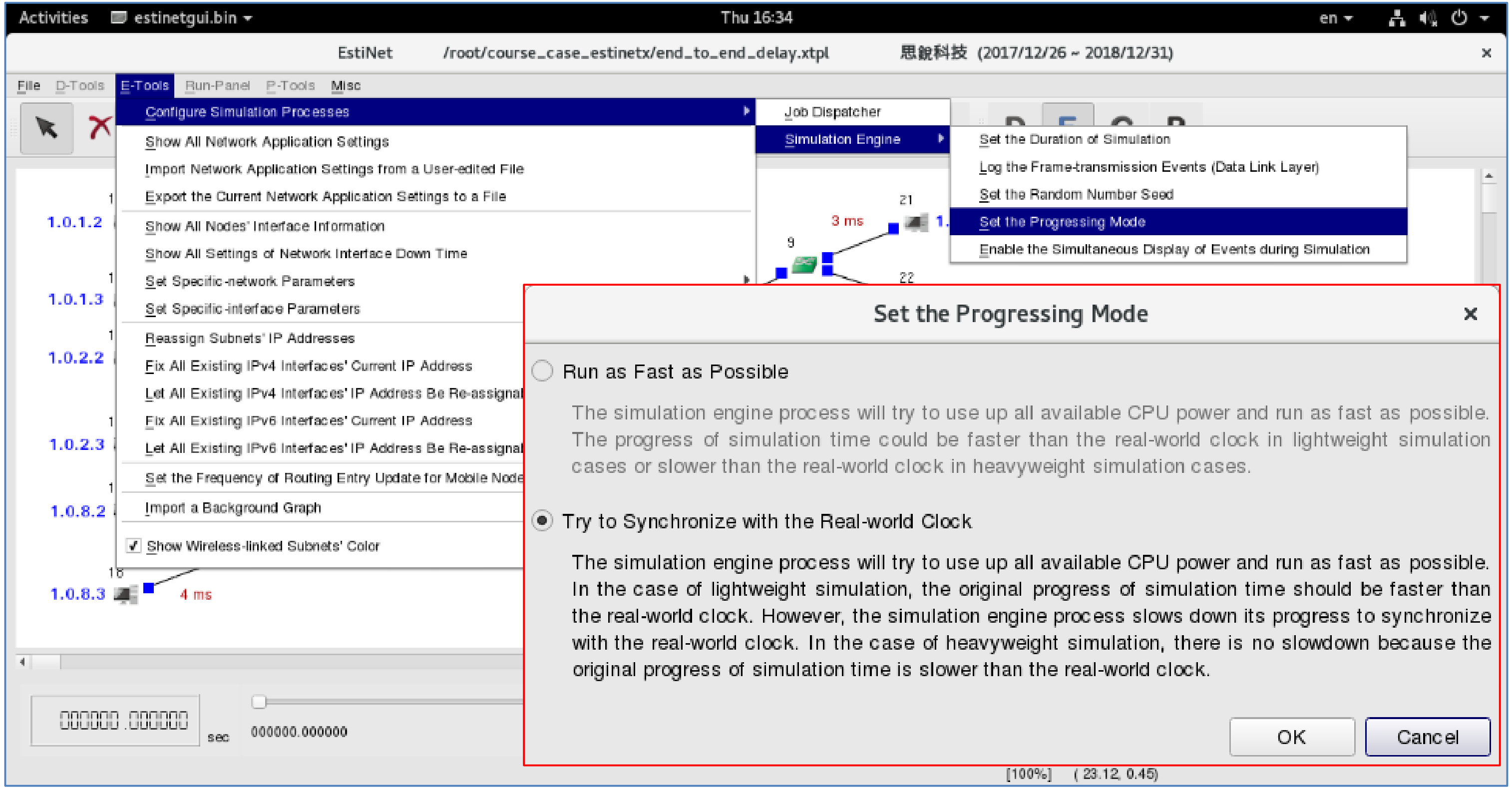
end_to_end_delay_trans_time_dominates.xtpl

# 在 D Stage 建立一个网络。在 E Stage 设定网络中各个 Link 的 Propagation Delay 。在 G Stage 进行仿真时，透过执行 ping 指令来观察端到端的延迟。

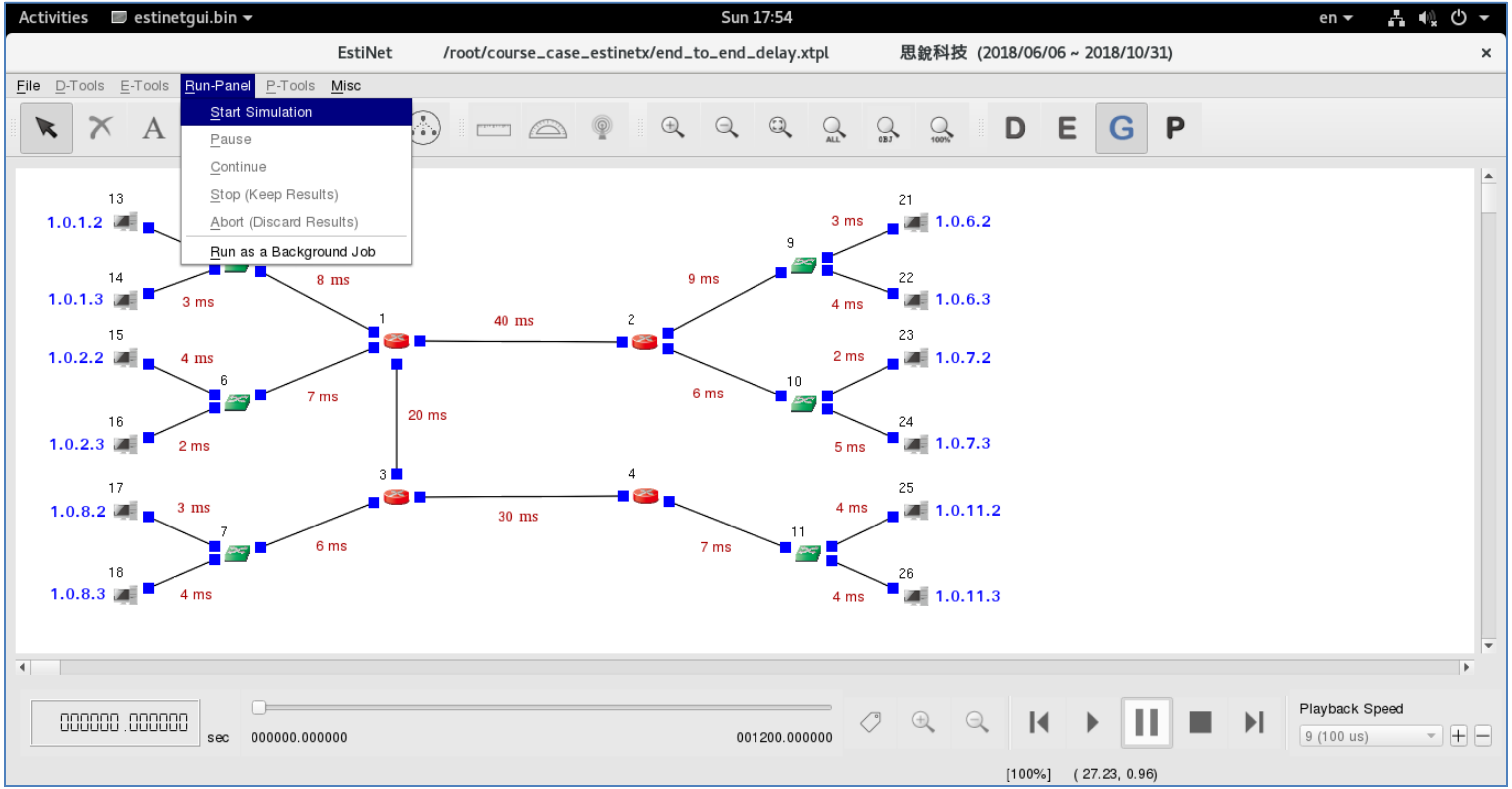


# 在 E Stage，设定 Progressing Mode 为 “Try to Synchronize with the Real-world Clock”

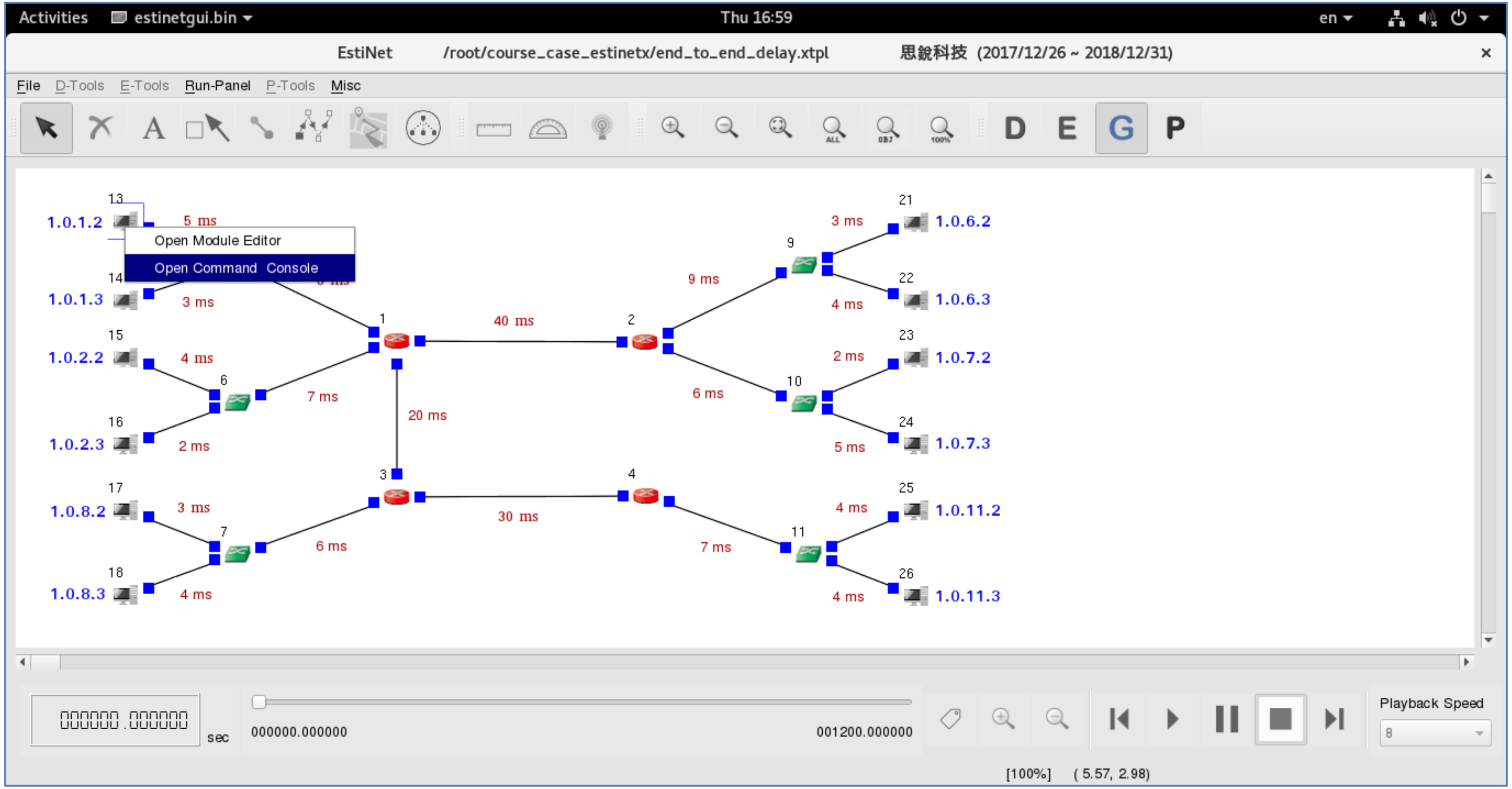
- 这个设定动作，是让稍后仿真进行时，虚拟时间的前进速度降低，如此用户才方便进行与仿真器的互动操作。



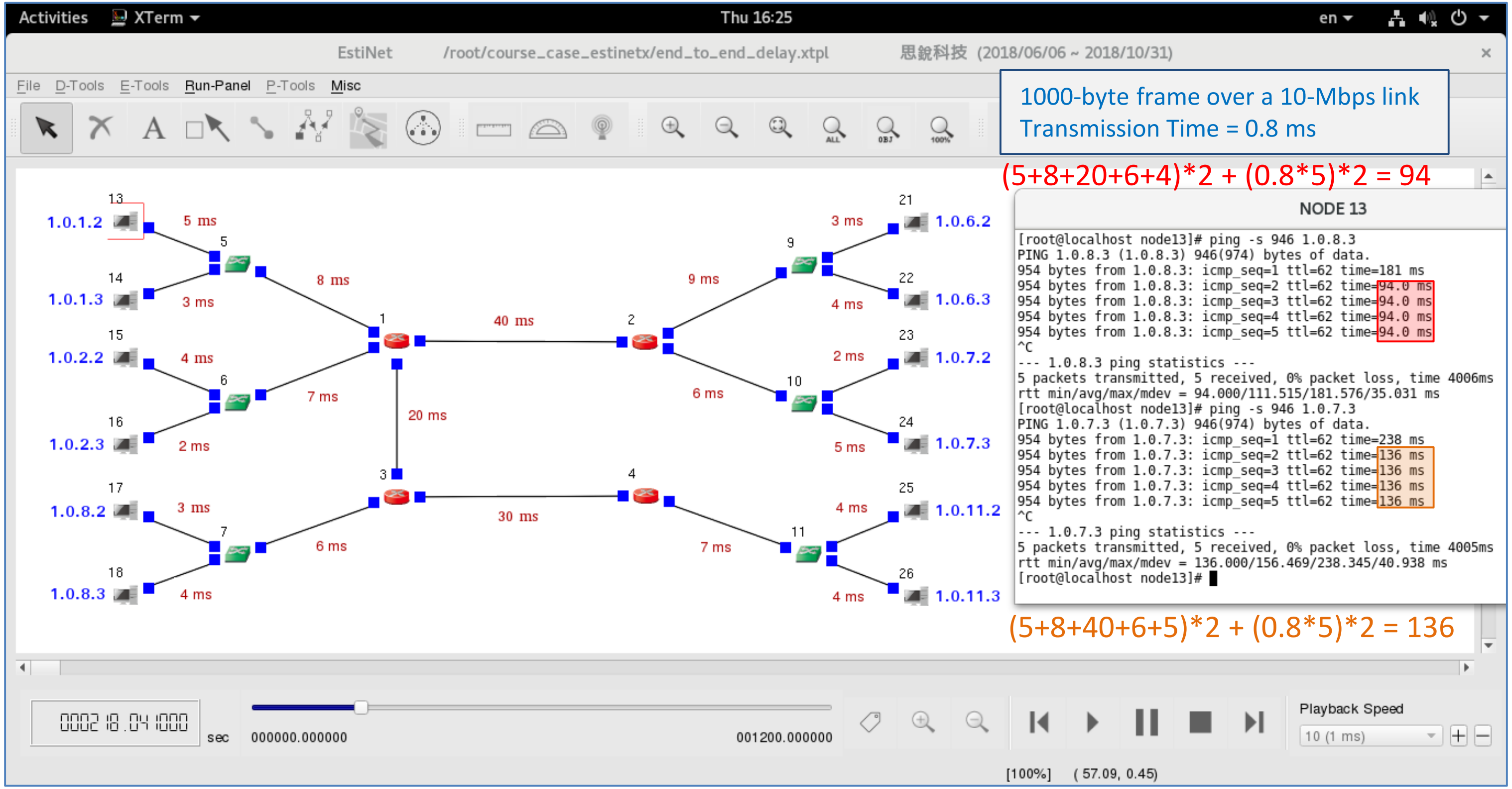
# 切换到 G Stage 产生仿真设定档，然后执行仿真



# 仿真进行期间，在任意 Host 上以鼠标右键来开启 Command Console

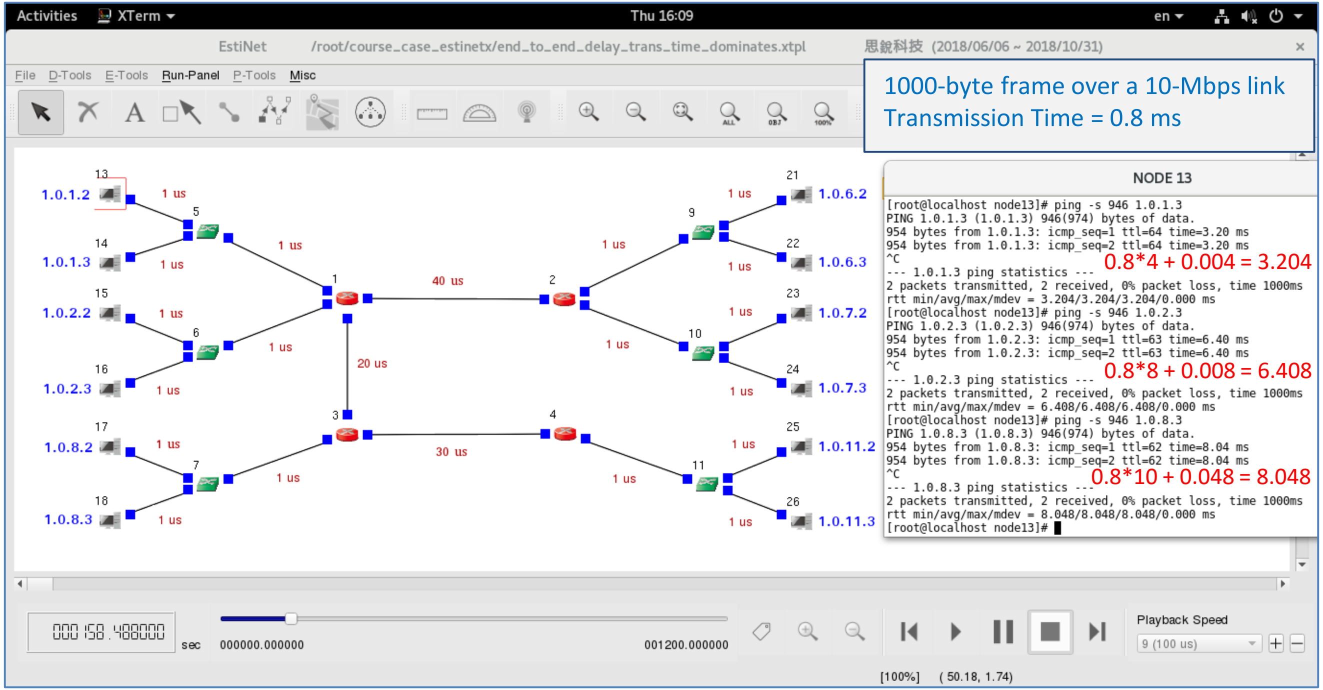


# 在 Command Console 视窗中，利用 ping 指令来观察任两个 Host 之间的 Round Trip Time



# 变化题

- 关于端到端时延 (End-to-end Delay)，若忽略在网络设备上的运算时延，则:
  - 探讨当传输时间 (Transmission Time) 远大于传播时延 (Propagation Delay) 的情况
  - 探讨当传输时间 (Transmission Time) 远小于传播时延 (Propagation Delay) 的情况

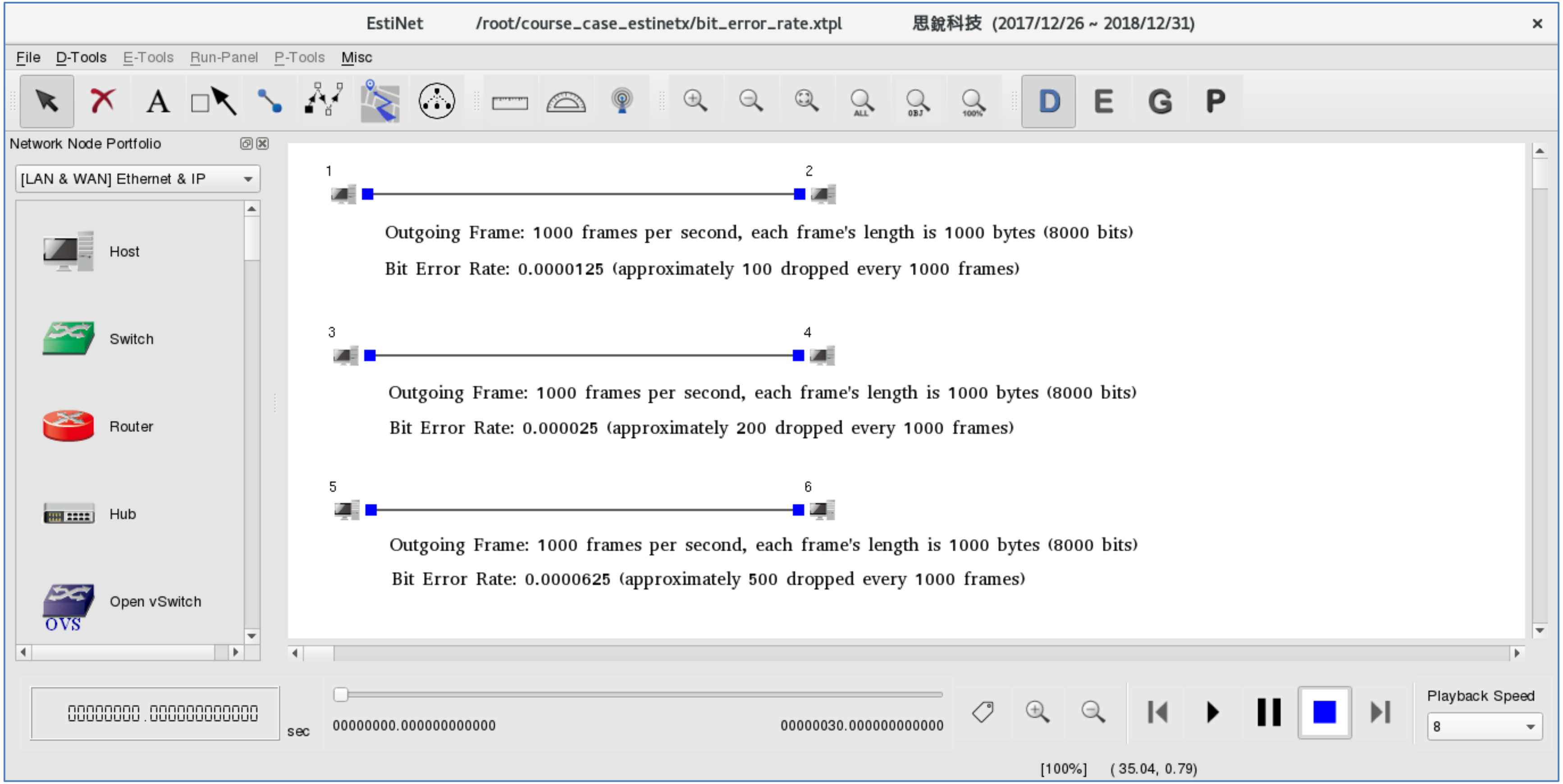


# 仿真器实验
# 有线信号错误率

< Simulation Case >

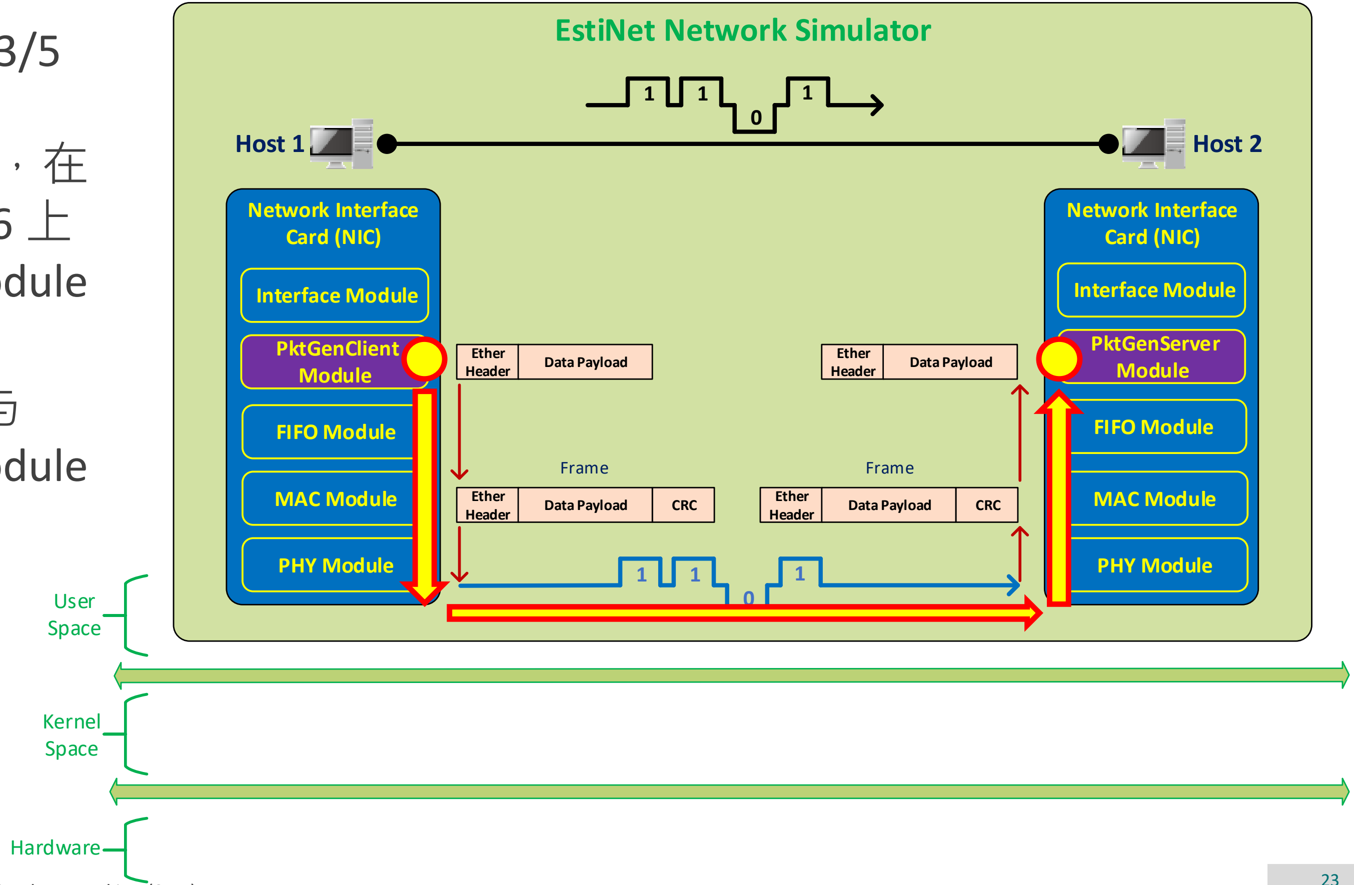
bit_error_rate.xtpl

# 在 D Stage，建构三组互相比较用的网络



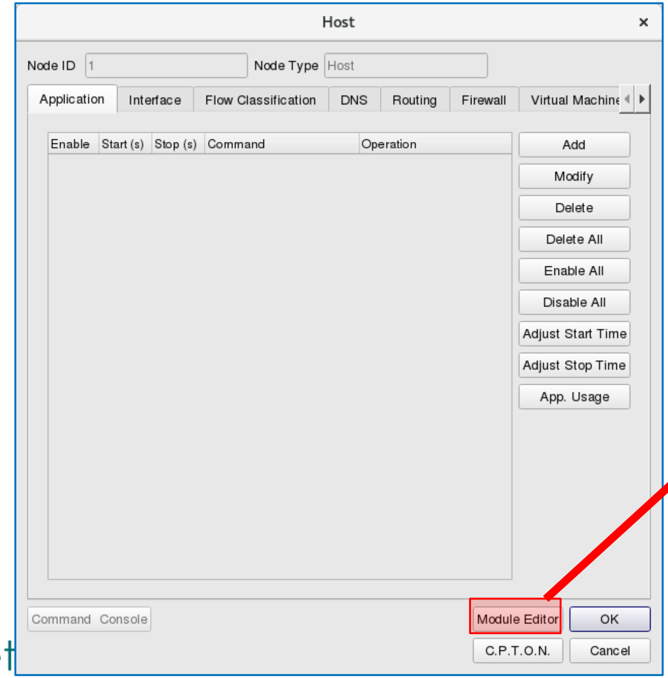
# 在 E Stage，利用 PktGen Module 在仿真期间送出 Frame

- 在信号传送端 Host 1/3/5 上使用 PktGenClient Module 来发送 Frame，在信号接收端 Host 2/4/6 上使用 PktGenServer Module 来接收 Frame。
- 留意在 PktGenClient 与 PktGenServer 两个 Module 下方都要加上 FIFO Module。

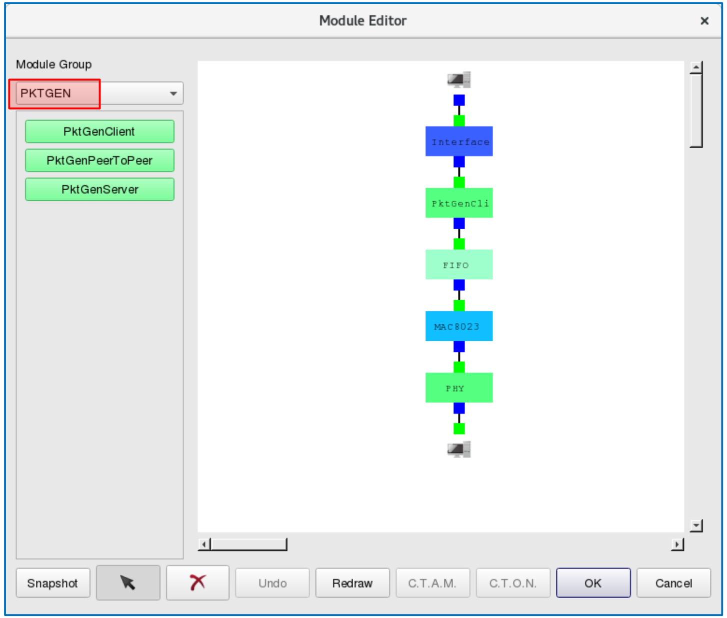


# 在节点编辑器 (Node Editor) 操作视窗上开启模组编辑器 (Module Editor)

- 在 E Stage，于任一个节点 (Node) 上连击鼠标左键两下，可开启节点编辑器。按下 “Module Editor” 的按键，可开启模组编辑器。



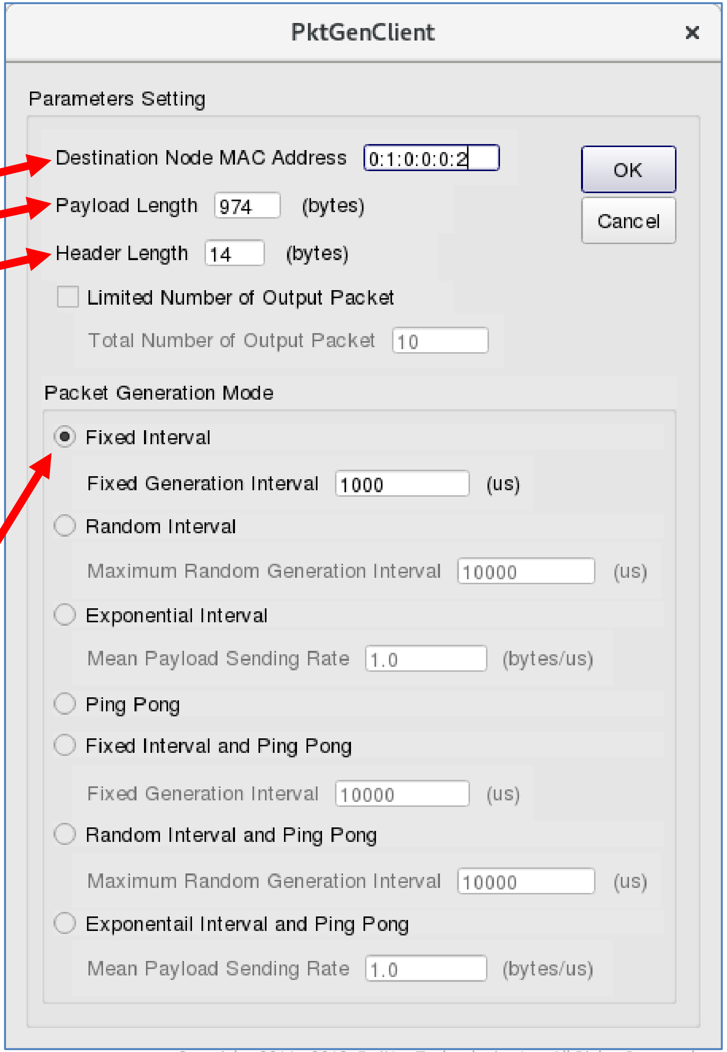

- 在模组编辑器左边可以根据群组分类来找到可以外加的模组，例如在 PKTGEN 群组中可以找到 PktGenServer 与 PktGenClient 模组，将所需模组挂到右边的协议栈 (Protocol Stack) 中。




EstiNet

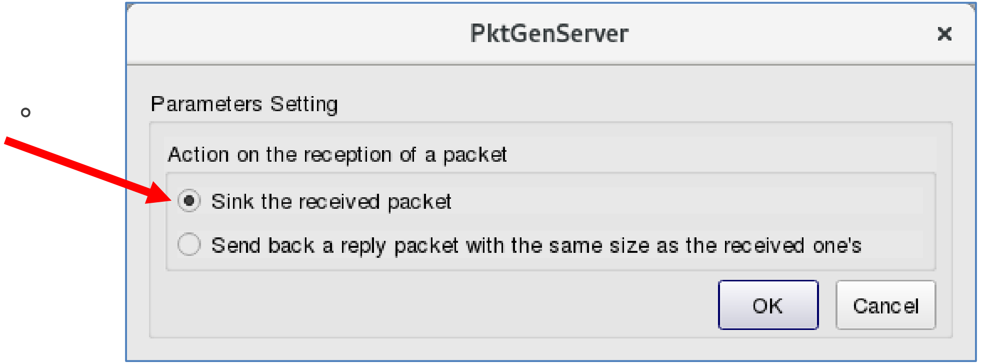
# PktGenClient Module 的设定

- 输入接收端 Host 2/4/6 的 MAC 地址。
- 输入数据长度 974 bytes。
- 输入表头长度 14 bytes。
- 数据长度 + 表头长度 = 988 bytes。
- 在 MAC Module 中会再加上 7 bytes 的 Preamble、1 byte 的 Start Frame Delimiter 与 4 bytes 的 CRC Checksum，因此，整个 Frame Size 就是 1000 bytes。
- 输入当仿真开始后，每隔 1000 us 就送出一个 Frame，换句话说，每秒会送出 1000 个 Frame。



# PktGenServer Module 的设定

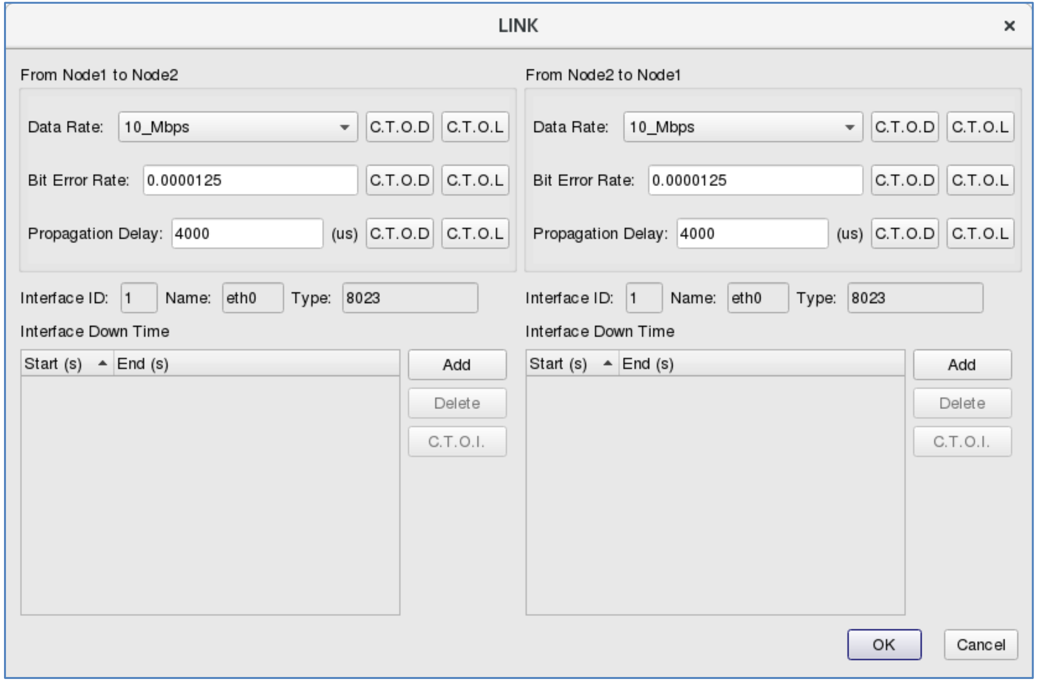
- 输入当接收端 Host 2/4/6 收到 Frame 后，直接将此 Frame 丢掉。



# 在 E Stage，分别在三条 Link 上设定不同的 Bit Error Rate

- 在连接 Host 1 与 Host 2 的 Link 上，设定 Bit Error Rate 为 0.0000125。
- 在连接 Host 3 与 Host 4 的 Link 上，设定 Bit Error Rate 为 0.000025。
- 在连接 Host 5 与 Host 6 的 Link 上，设定 Bit Error Rate 为 0.0000625。

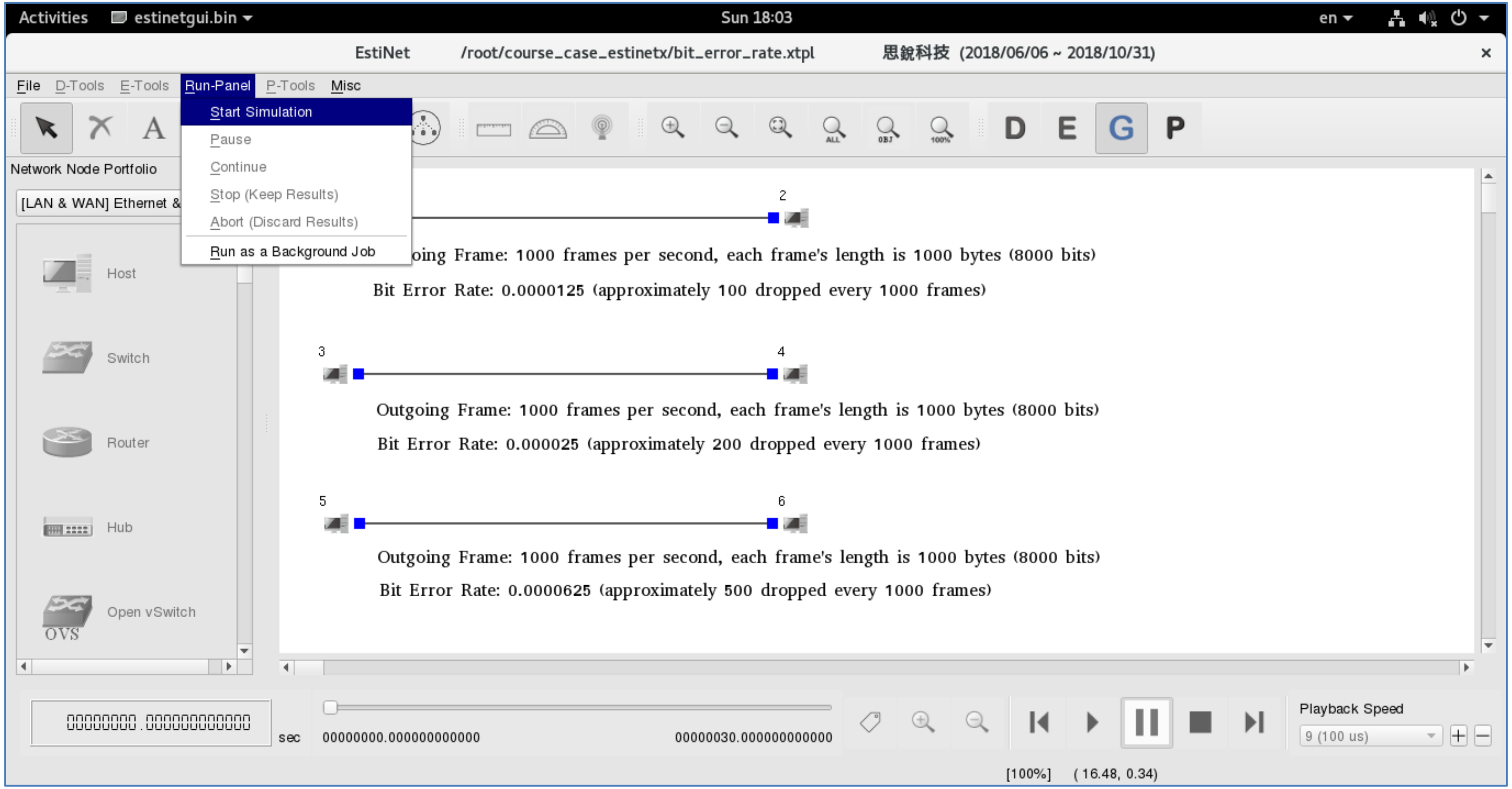
- 三条 Link 上的 Propagation Delay 都设为 4000 us。



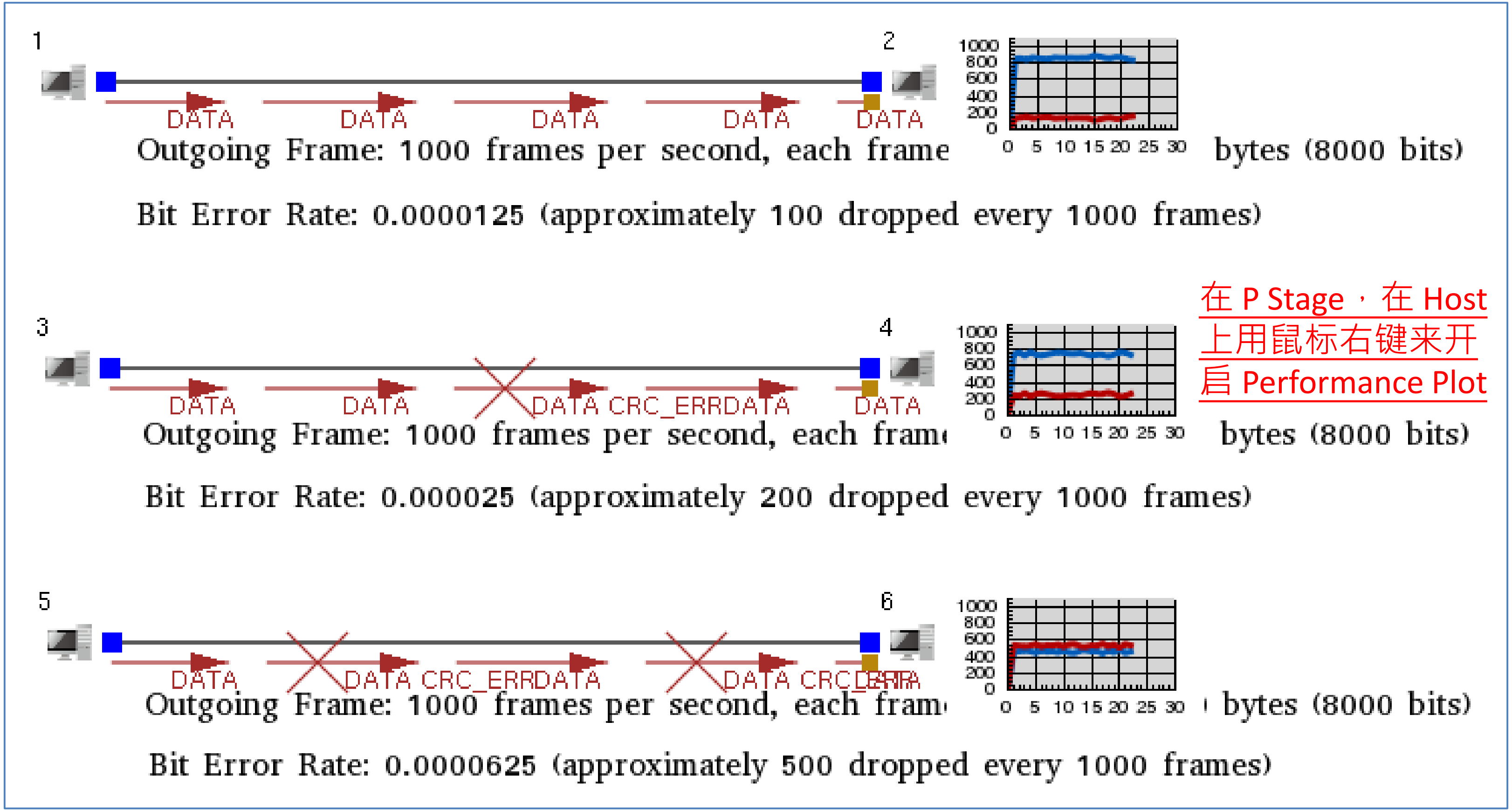
# 根据 Bit Error Rate 来预先估算 Frame Error Rate

- 若一个 Frame 为 1000 bytes (8000 bits)，而 Bit Error Rate 为 1/80000 (0.0000125)，也就是说，每 80000 个 bit 会有一个 bit 发生错误，换句话说，每 10 个 Frame 就会有一个 Frame 的内容会发生一个 bit 出错的情况，因此，Frame Error Rate 就是 1/10 (0.1)。
  - Bit Error Rate = 0.0000125 (1/80000) → Frame Error Rate = 0.1 (10%)
  - Bit Error Rate = 0.000025 (1/40000) → Frame Error Rate = 0.2 (20%)
  - Bit Error Rate = 0.0000625 (1/16000) → Frame Error Rate = 0.5 (50%)
- 以传送端每秒送出 1000 Frame 的流量来看:
  - Frame Error Rate = 10% → 在接收端，约有 100 Frame 出错，有 900 Frame 成功接收
  - Frame Error Rate = 20% → 在接收端，约有 200 Frame 出错，有 800 Frame 成功接收
  - Frame Error Rate = 50% → 在接收端，约有 500 Frame 出错，有 500 Frame 成功接收

# 切换到 G Stage 产生仿真设定档，然后执行仿真



# 仿真结束后，在 P Stage，根据仿真结果来观察 Frame Error Rate



# 仿真器实验
# 无线信号传递范围

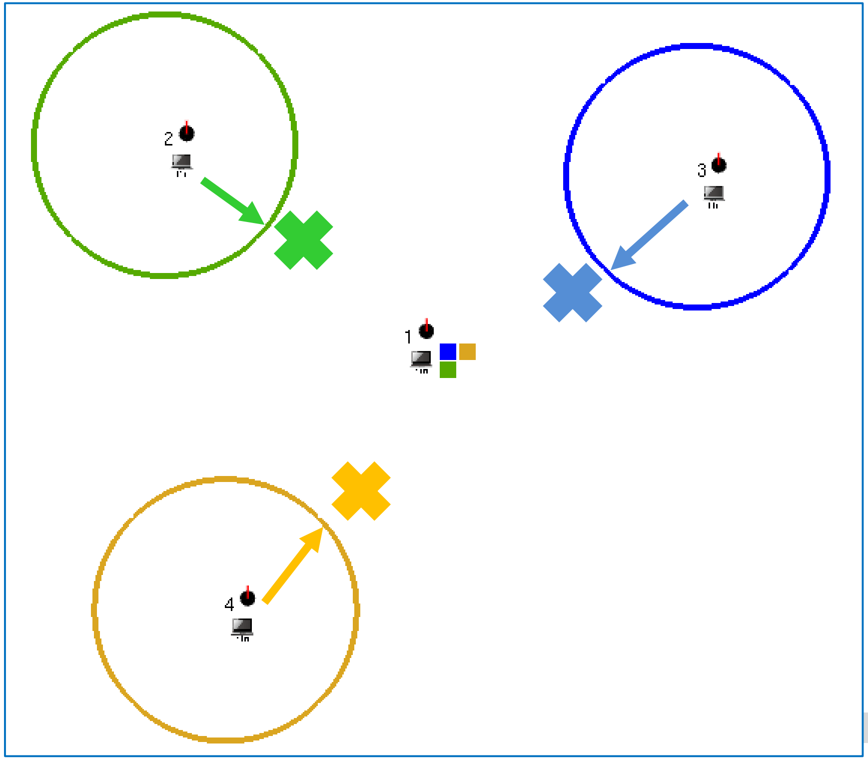
< Simulation Case >

antenna_gain_pattern_adjustment.xtpl

# 决定无线信号传递范围的因素

- 传送端与接收端的天线增益方向图（Antenna Gain Pattern）
- 传送端传递信号所使用的能量（Tx Power）
- 接收端的信号接收灵敏性（Rx Sensitivity）
- 信号频率与环境参数（地形、地物、天气、干扰等等）

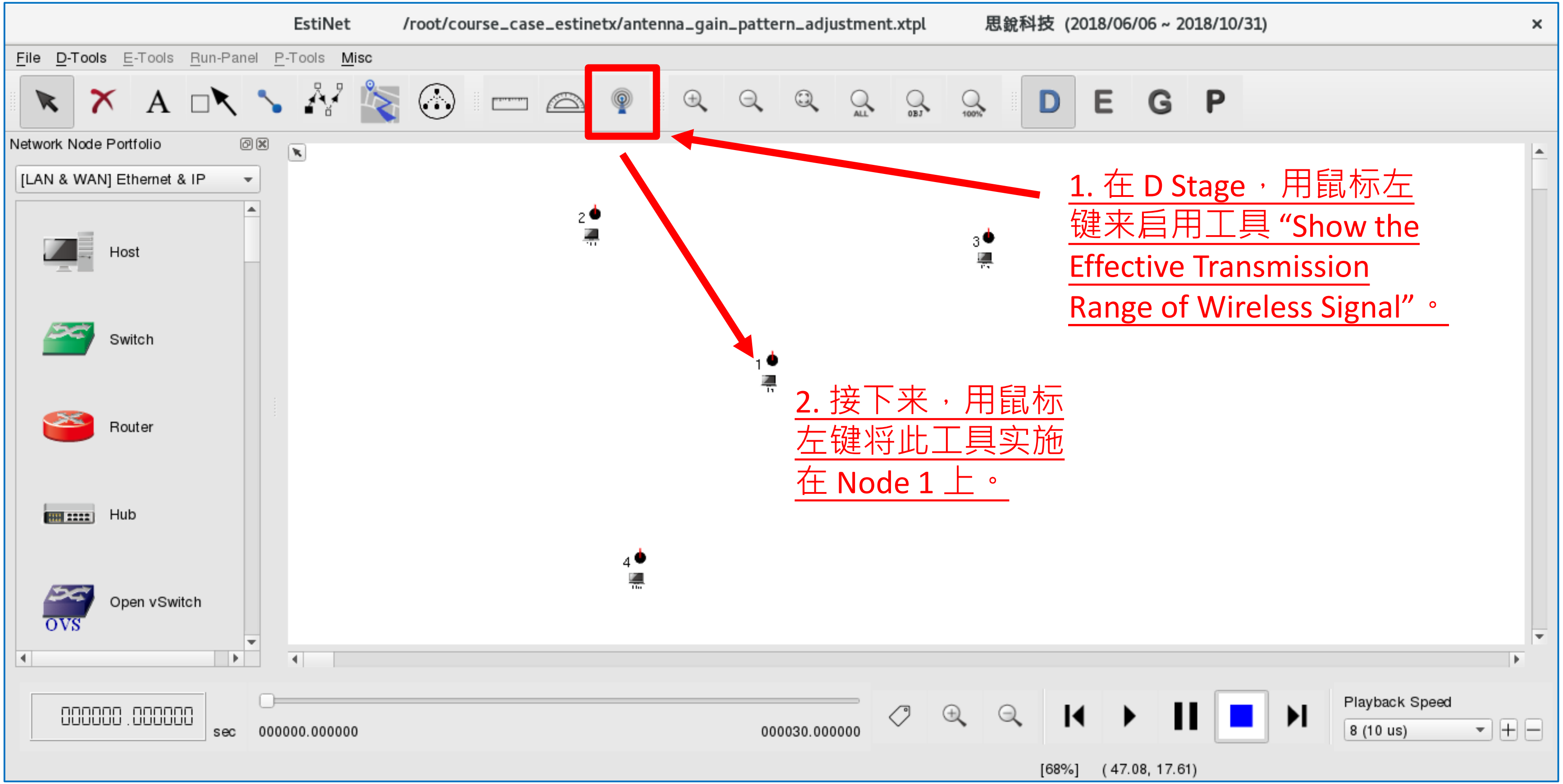
在右图所展示的网络中，位于中央的 1号节点是信号接收者，位于周围的 2 号、3 号与 4 号节点都是信号传送者，目前的情况是，周围三个节点的信号传递范围都不够大，因此都无法将信号成功传递给 1 号节点。




实验目标：
根据上述所提到的决定信号传递范围的因素，调整对应的仿真参数，来让 1 号节点可以成功接收来自其它节点所发送的信号。

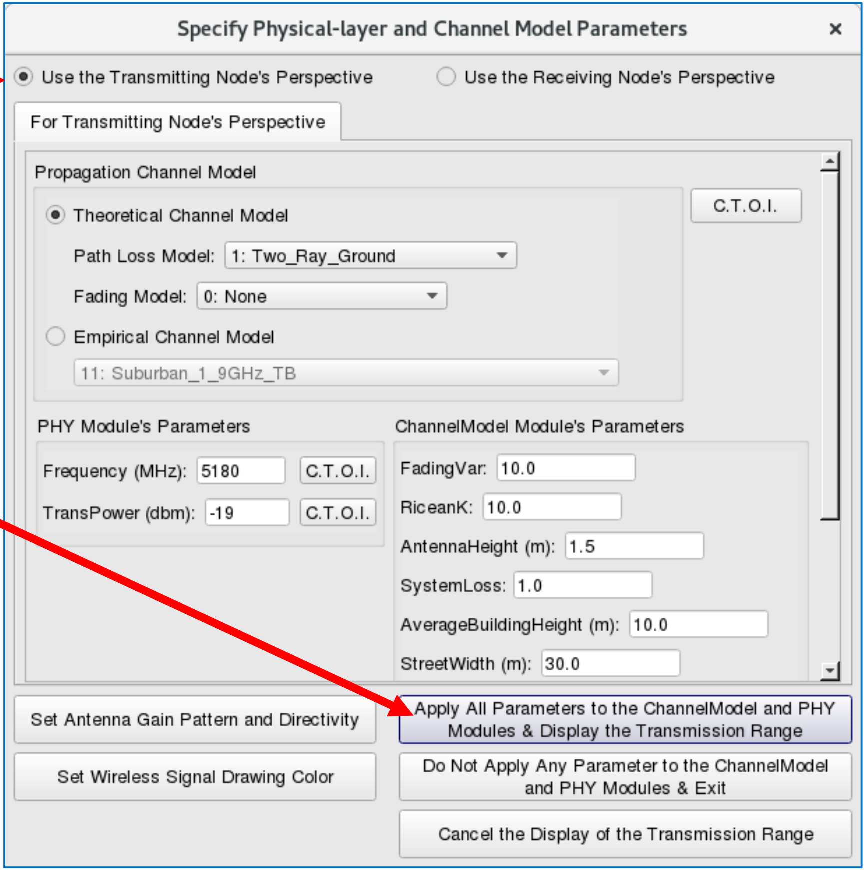
# 如何观看无线信号传递范围

# [步骤一] 启用"显示有效信号传递范围"的工具

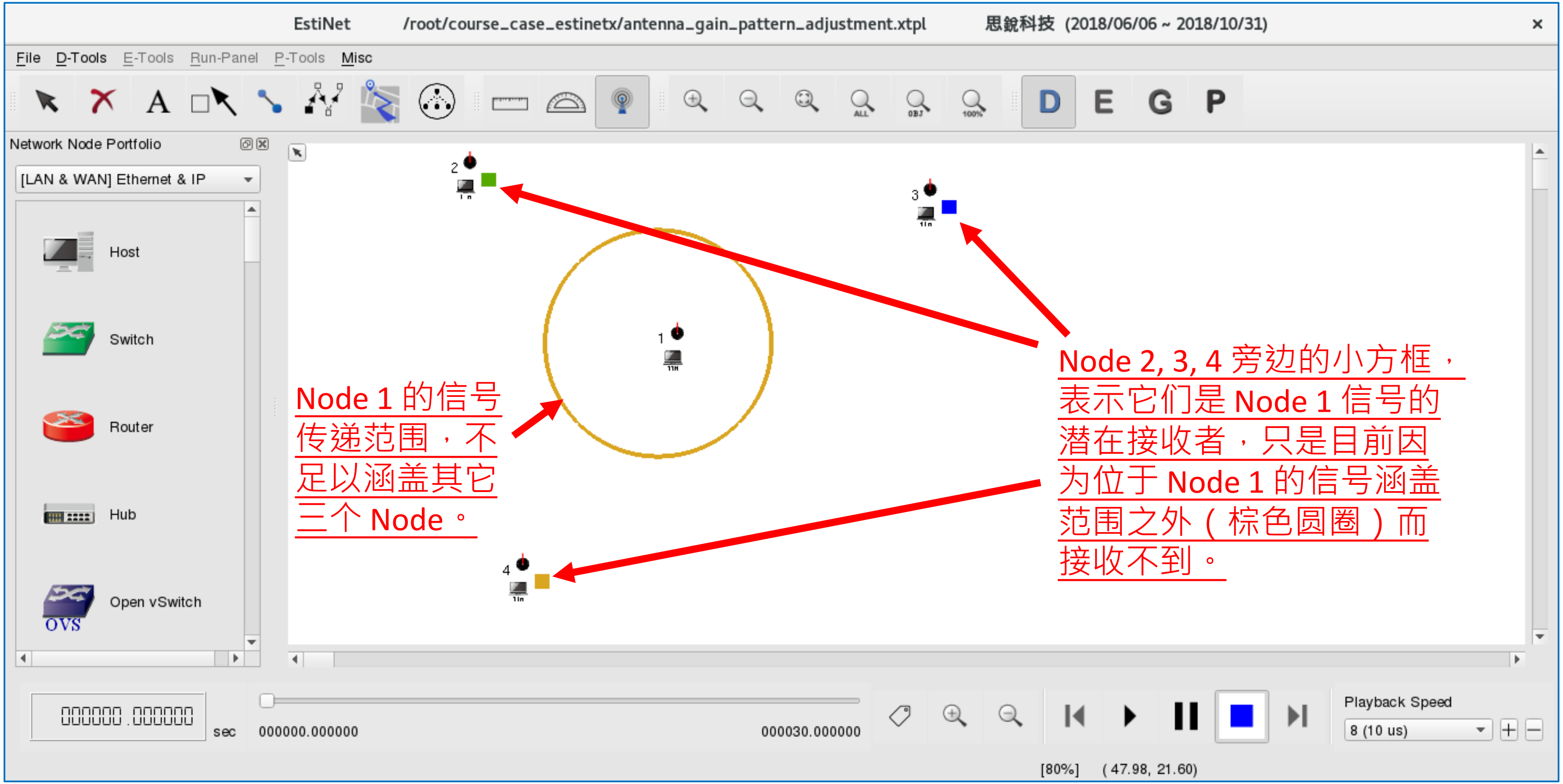


# [步骤二] 在设定视窗中，选择以信号传送者的观点来观看信号传递范围

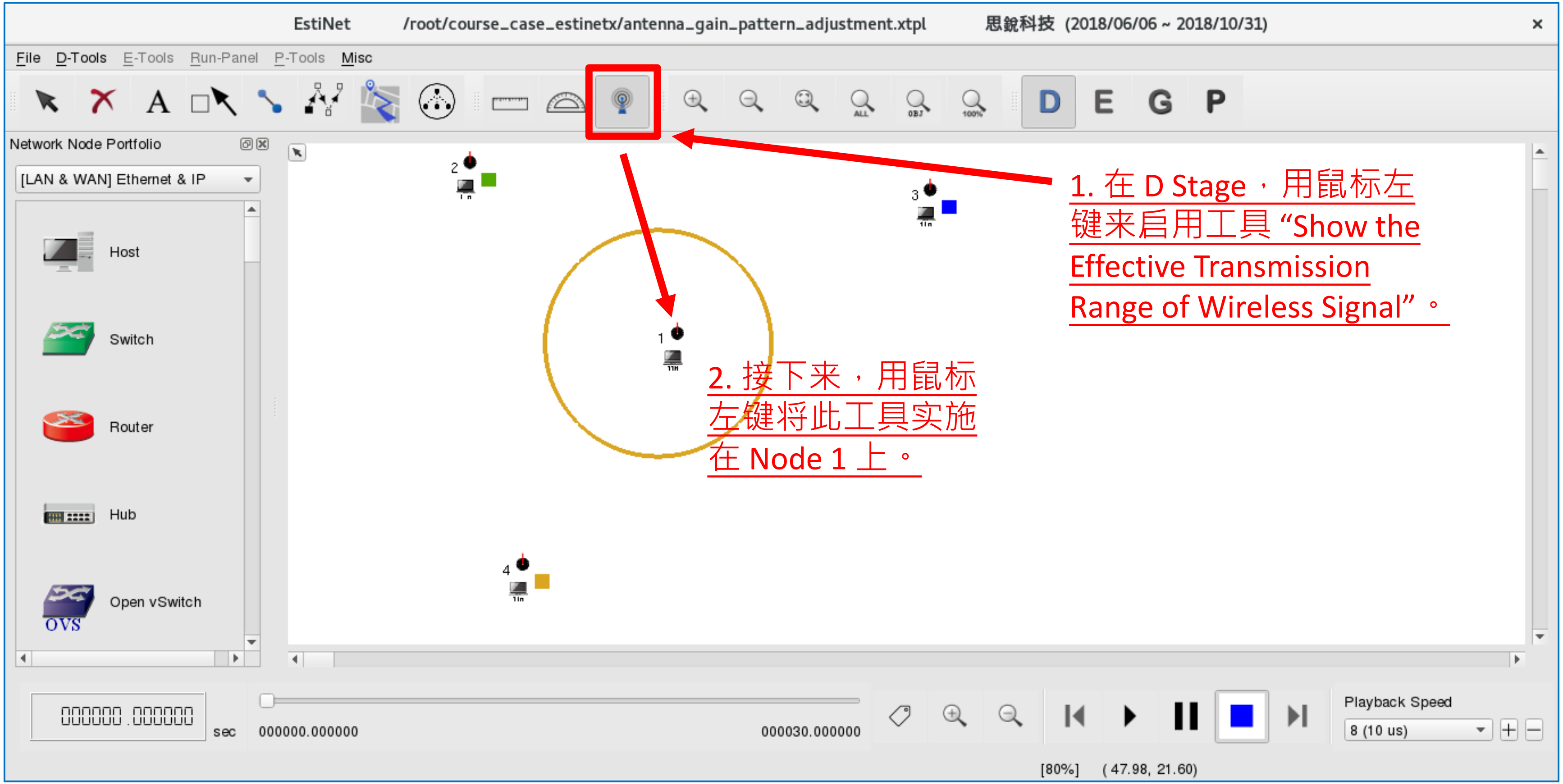
- 选择以信号传送者观点。
- 套用所设定参数并画出信号传递范围。



# [步骤三] 观看 Node 1 所送出的信号传递范围

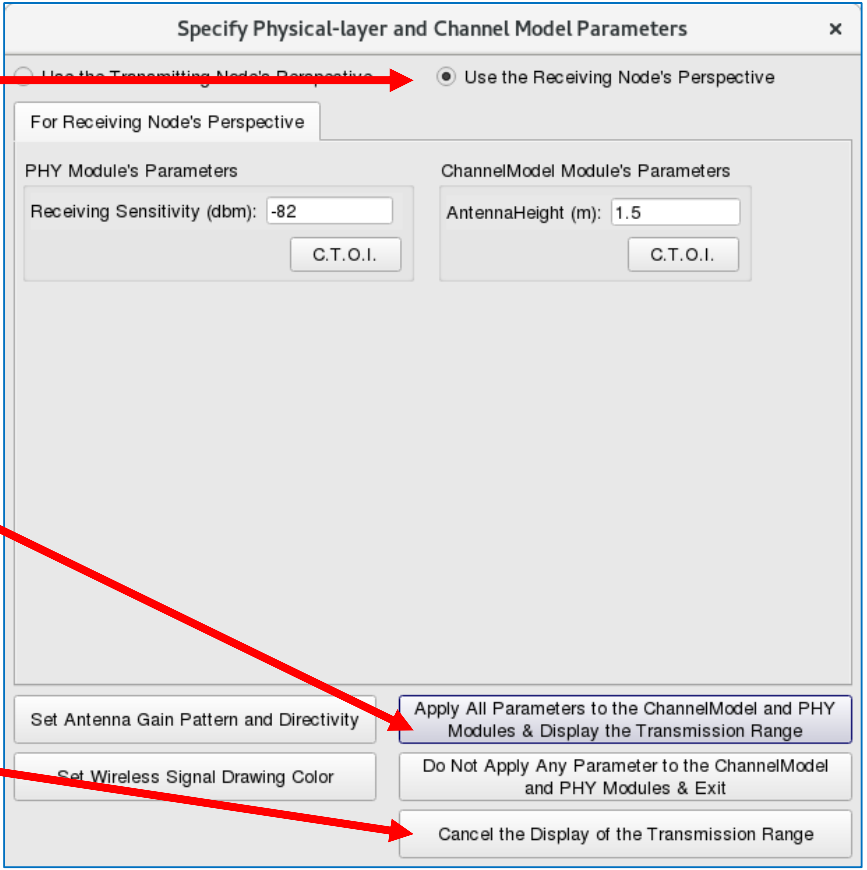


# [步骤四] 再次启用"显示有效信号传递范围"的工具



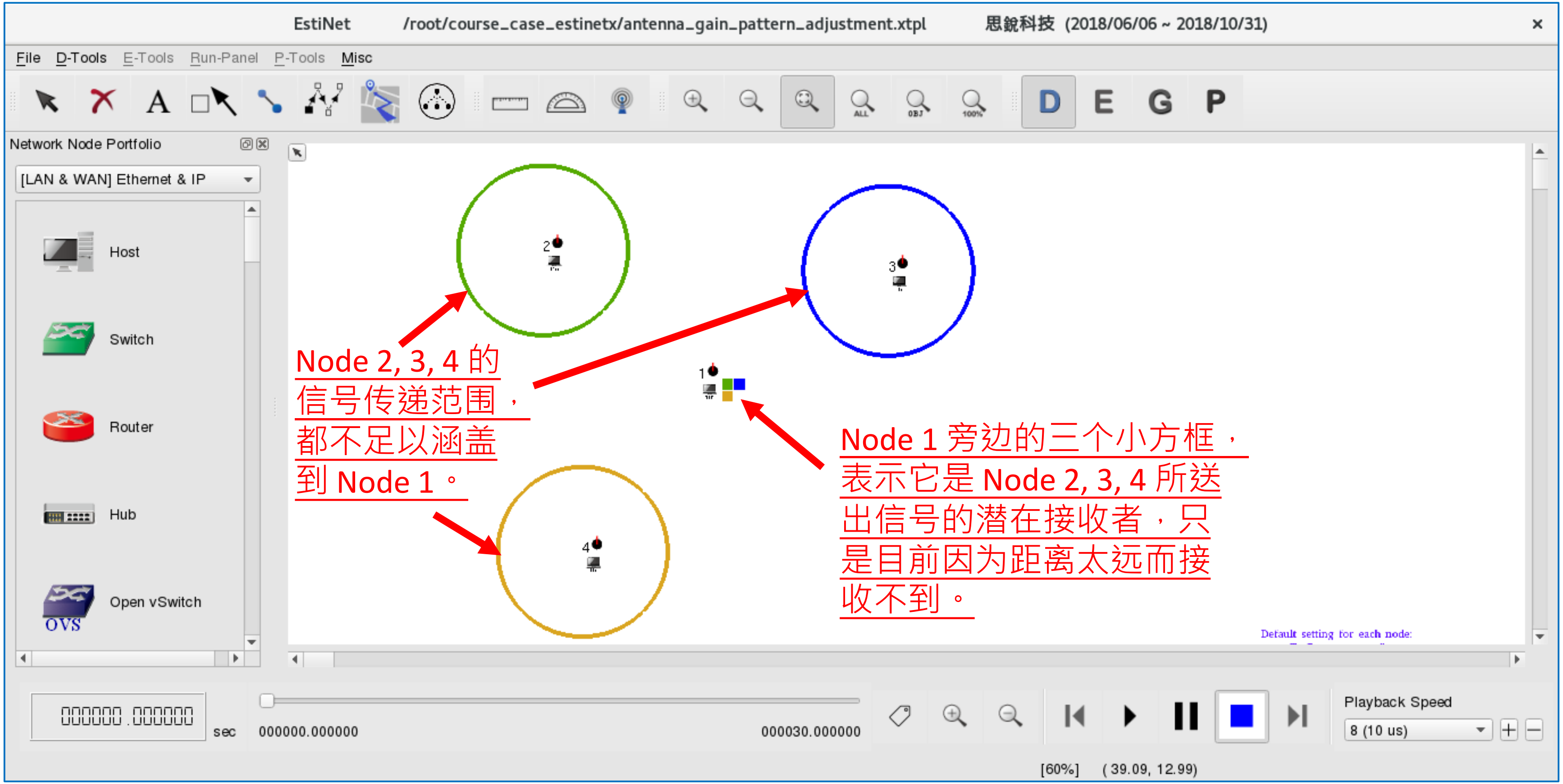
# [步骤五] 在设定视窗中，改选择以信号接收者的观点来观看信号传递范围

- 选择以信号接收者观点。
- 套用所设定参数并画出信号传递范围。



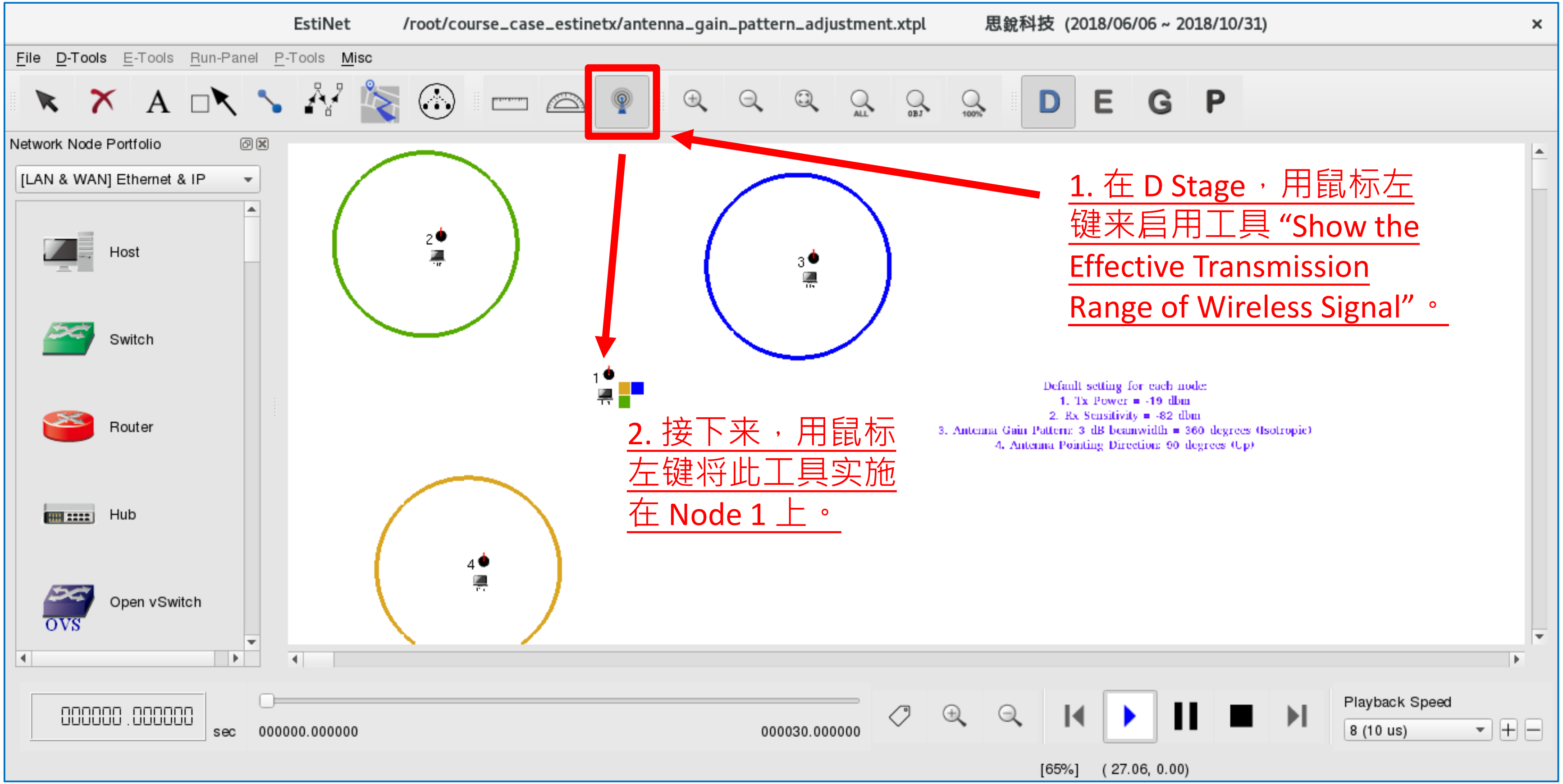

- 若想取消展示信号传递范围的功能，就可按此键。

# [步骤六] 观看 Node 1 所能接收的所有信号的传递范围



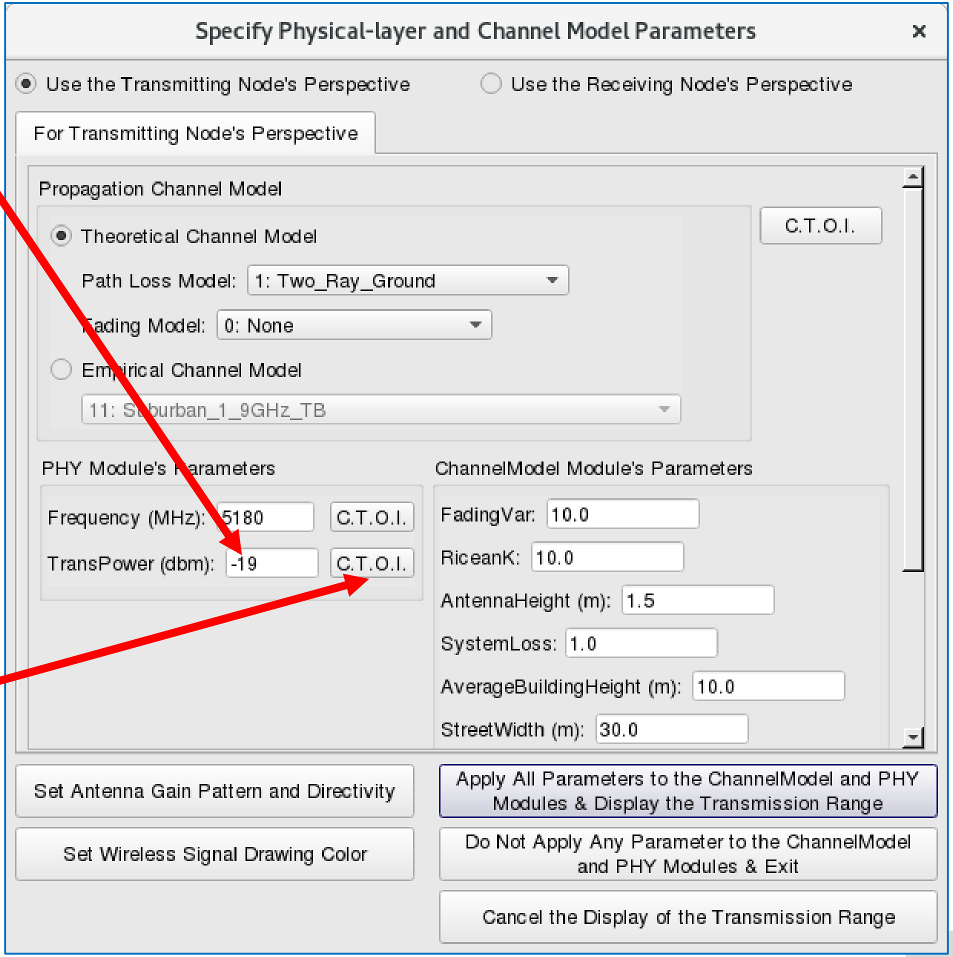
## 增加传送端传递信号所使用的能量（TX POWER）来增大信号传递范围

# [步骤一] 启用"显示有效信号传递范围"的工具



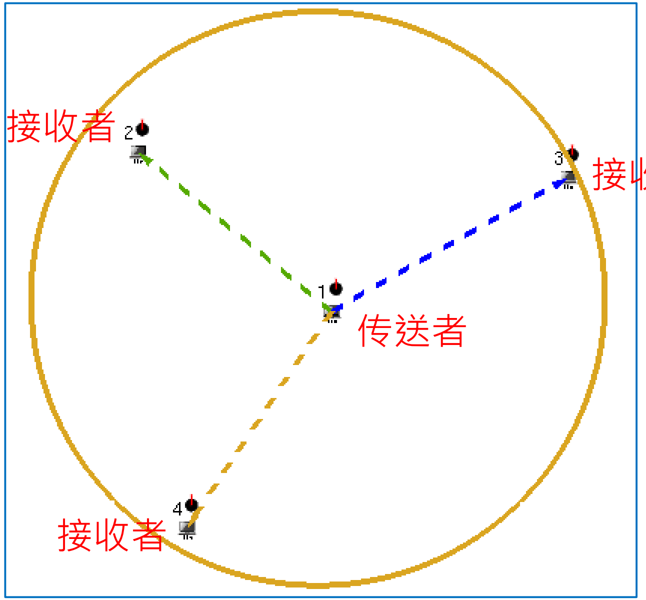
# [步骤二] 调整传递信号所使用的能量

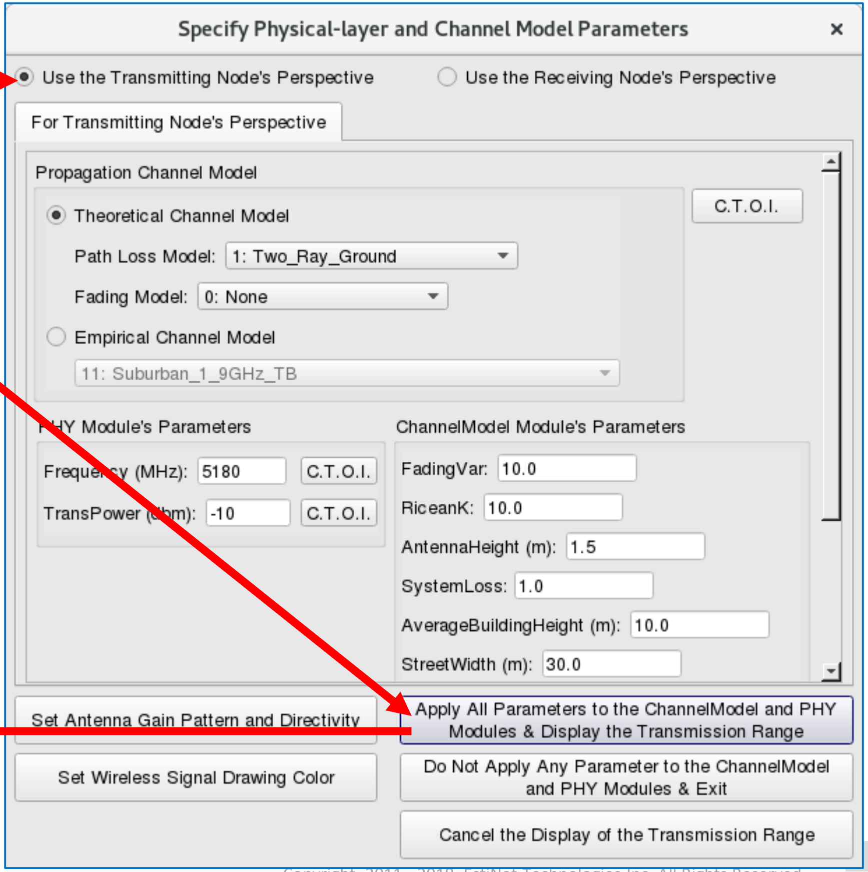
- 在 Node 1 的弹出视窗中，将 ”TransPower (dbm)” 的数值由 -19 改成 -10，藉此增加传送信号的能量。
  - 分贝毫瓦 (dbm) 单位是经由对数运算而来。
  - 若 Power_mW (milliwatt，毫瓦) 是原来的信号能量，则 Power_dbm = $10 * \log_{10}\left(Power\_mW / 1\ milliwatt\right)$。
  - 换句话说，分贝毫瓦是原传输能量相对于 1 毫瓦的比例再取对数后的结果。
  - 因此，-19 分贝毫瓦表示原传输能量是 0.01259 毫瓦，而 -10 分贝毫瓦表示原传输能量是 0.1 毫瓦，后者的能量增加了将近 7.94 倍。
- 按下按键 ”C.T.O.I” (Copy to Other Interfaces) 来将修改的数值复制到其它节点上 (Node 2, 3, and 4)。
  - 传送者与接收者两边的传输能量都加强，如此可达到双向通信。



# [步骤三] 观察由 Node 1 所发送的信号是否能被其它三个 Node 接收到

- 设定以信号传送者的观点来观察所送出的信号可以被哪些潜在的信号接收者所收到。
- 按下参数套用按键来观看结果。

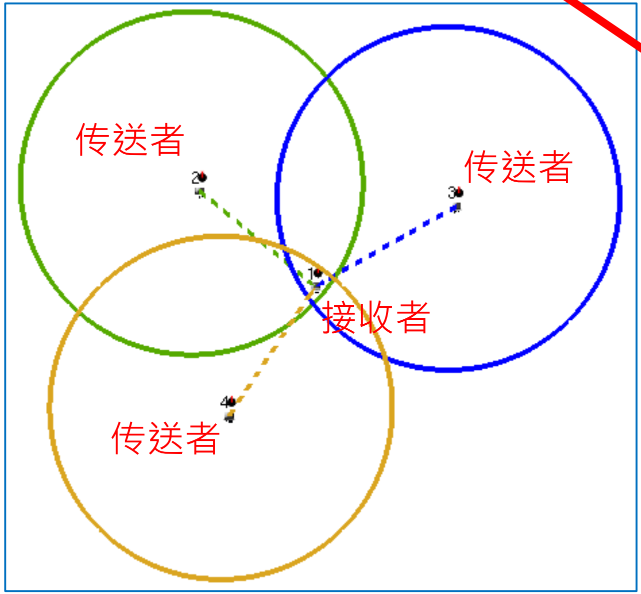


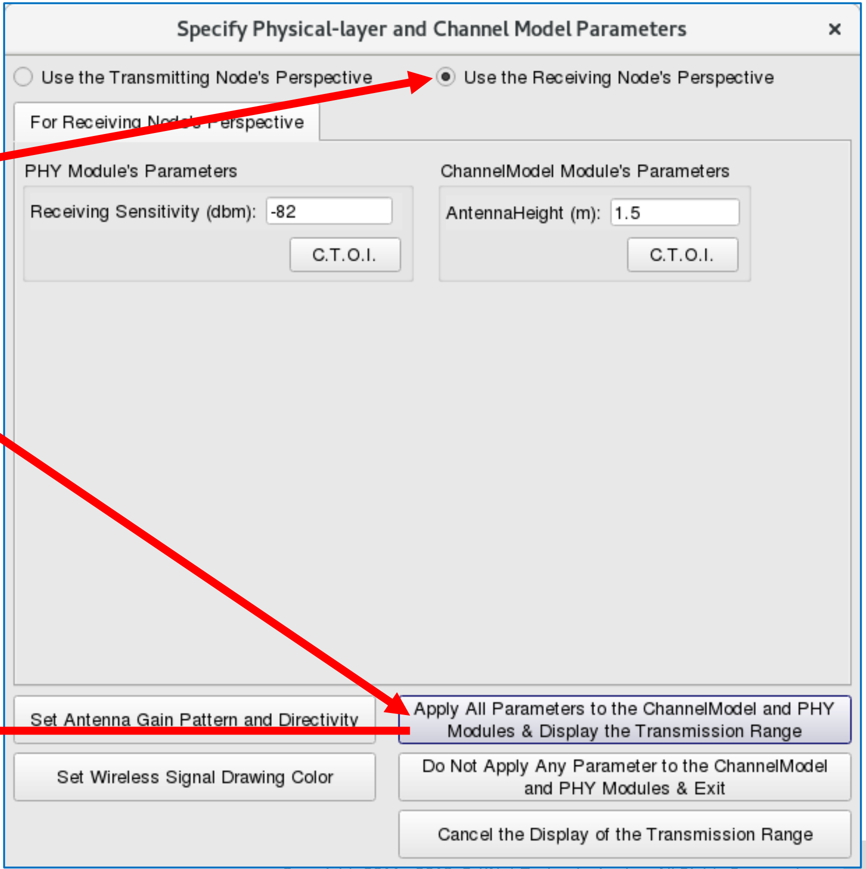




EstiNet

# [步骤四] 观察其它三个 Node 所发送的信号是否能被 Node 1 所接收

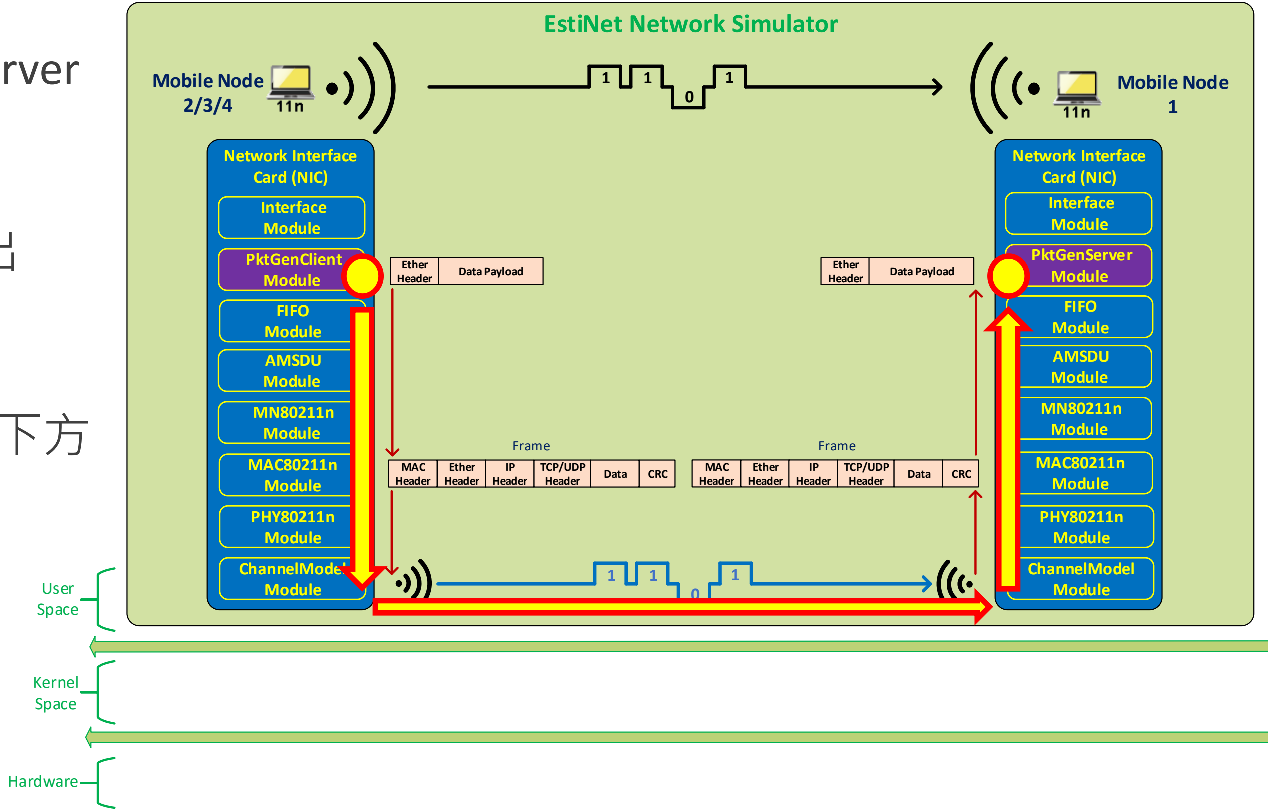
- 再一次将”显示有效信号传递范围”的工具套用到 Node 1 上面。
- 设定以信号接收者的观点来观察哪些信号传送者所送出的信号可以被接收者收到。
- 按下参数套用按键来观看结果。



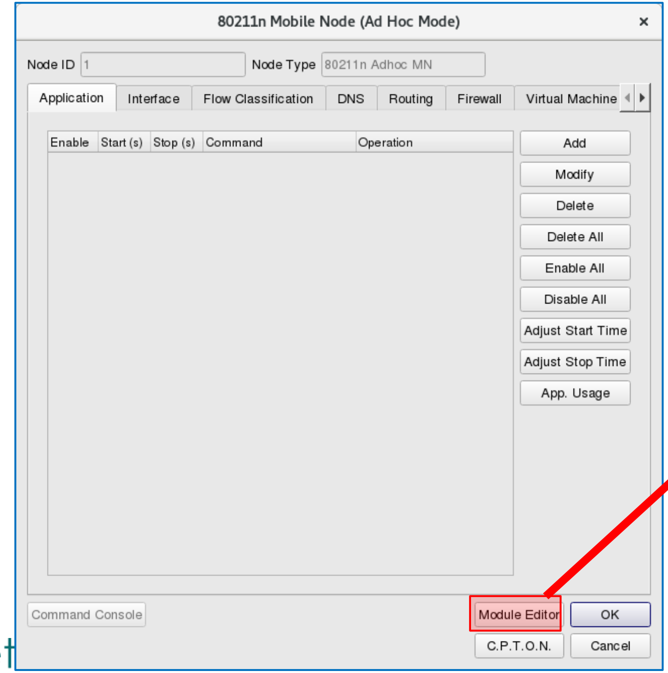
# [步骤五] 在 E Stage，设定让 PktGen Module 在仿真期间送出与接收 Frame

- 在 Node 1 上使用 PktGenServer Module 来接收 Frame。
- 在 Node 2, 3, 与 4 上使用 PktGenClient Module 来送出 Frame。
- 留意在 PktGenClient 与 PktGenServer 两个 Module 下方都要加上 FIFO Module。

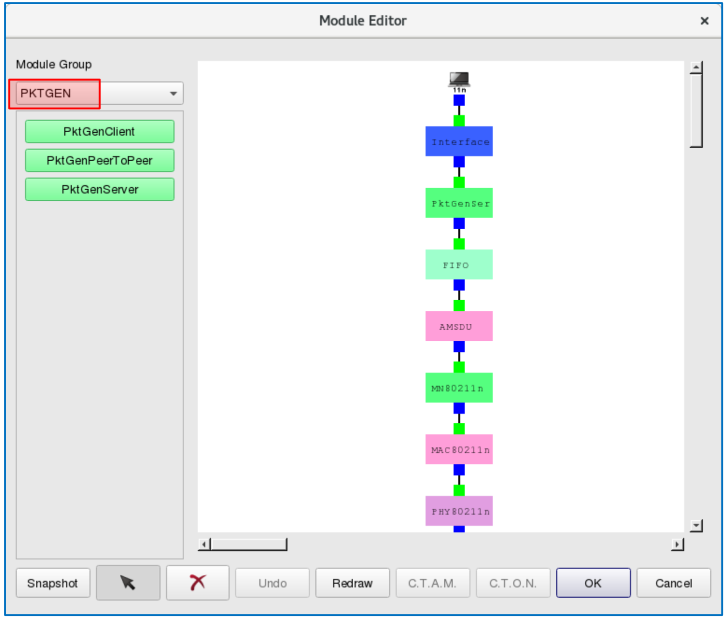


# 在节点编辑器 (Node Editor) 操作视窗上开启模组编辑器 (Module Editor)

- 在 E Stage，于任一个节点 (Node) 上连击鼠标左键两下，可开启节点编辑器。按下 “Module Editor” 的按键，可开启模组编辑器。

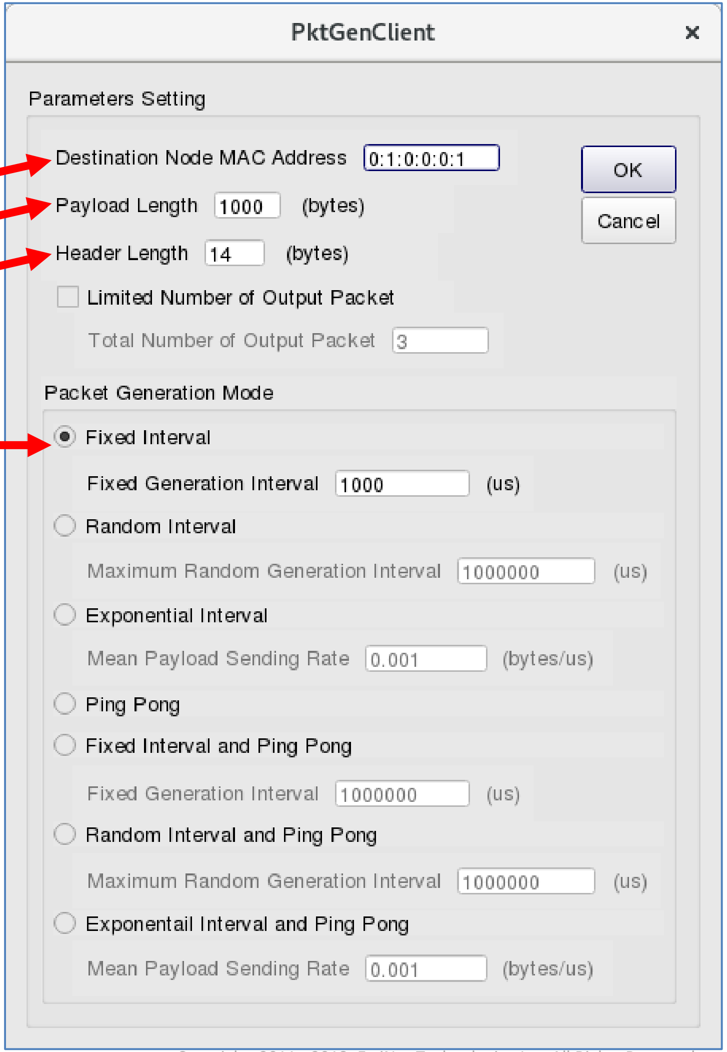



- 在模组编辑器左边可以根据群组分类来找到可以外加的模组，例如在 PKTGEN 群组中可以找到 PktGenServer 与 PktGenClient 模组，将所需模组挂到右边的协议栈 (Protocol Stack) 中。



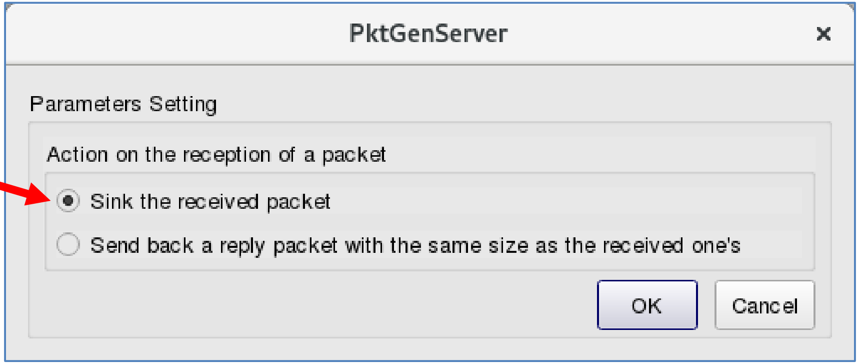
# PktGenClient Module 的设定

- 输入接收端 Node 1 的 MAC 地址。
- 输入数据长度 1000 bytes。
- 输入表头长度 14 bytes。
- 输入当仿真开始后，每隔 1000 us 就送出一个 Frame，换句话说，每秒会送出 1000 个 Frame。

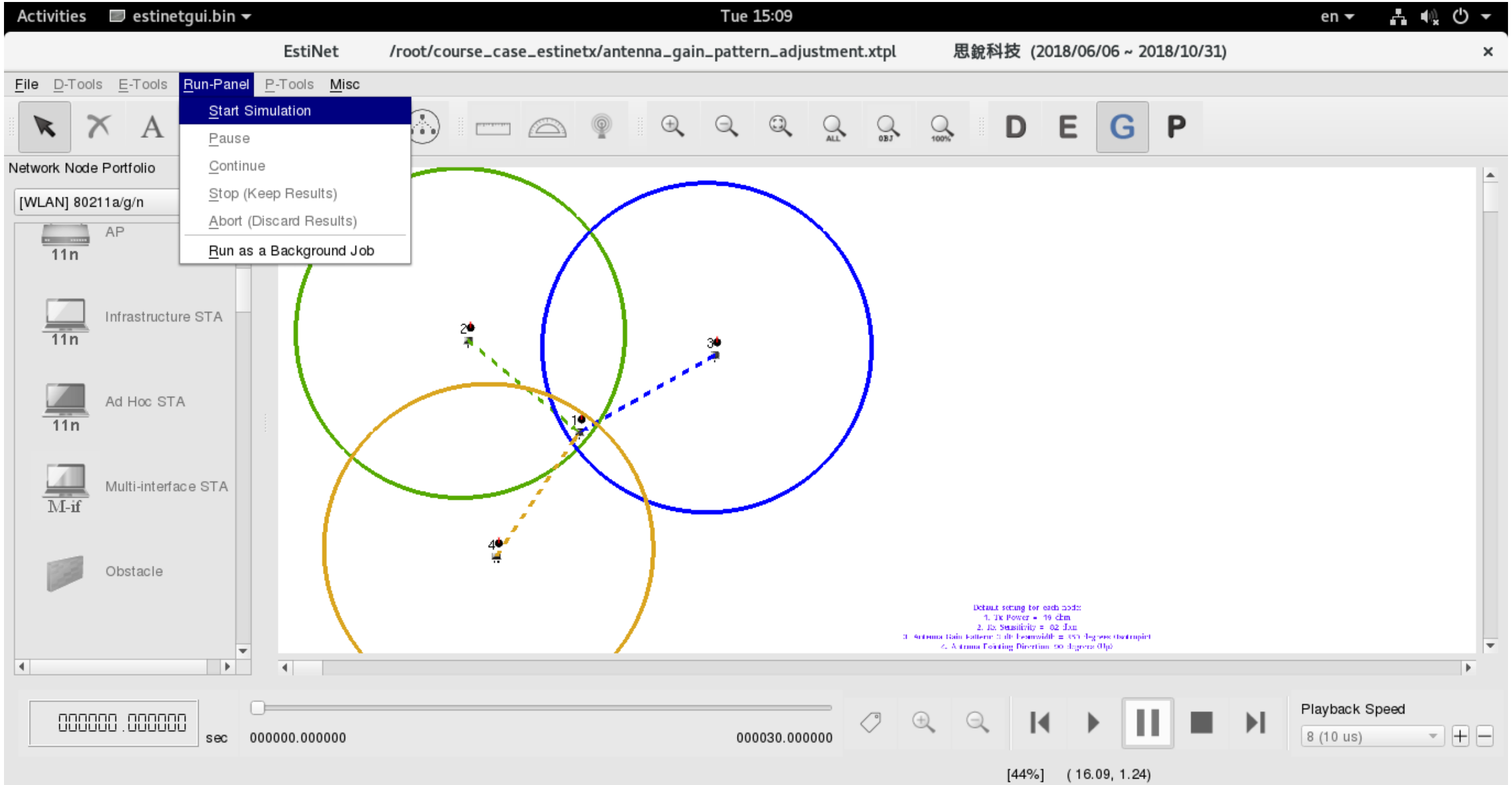


# PktGenServer Module 的设定

- 输入当接收端 Node 1 收到 Frame 后，直接将此 Frame 丢掉。

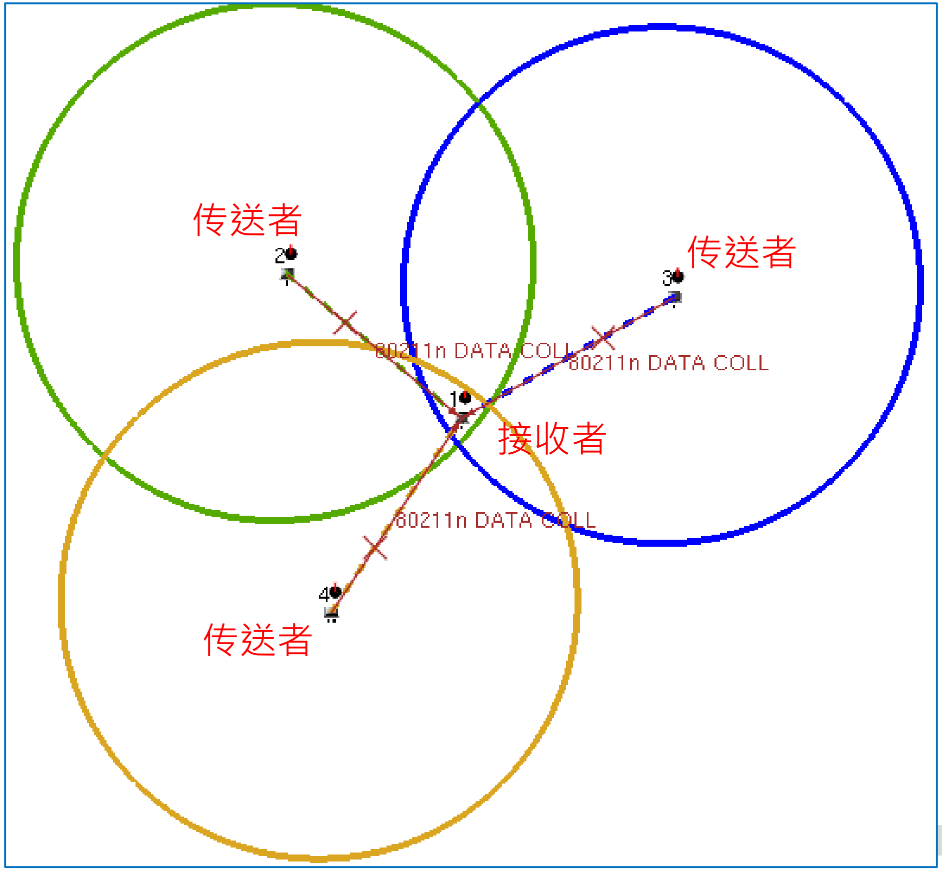


# [步骤六] 切换到 G Stage 产生仿真设定档，然后执行仿真



# [步骤七] 仿真完成后，在 P Stage 观看结果

- 在 P Stage，根据仿真结果来展示 Frame 传送动画，如右图所示，Node 2、Node 3 与 Node 4 所送出的信号可以到达 Node 1，但因为三个信号几乎同时间到达，所以造成信号碰撞的情况，导致 Node 1 没有成功收起任何 Frame。
- 信号碰撞的问题，会由网络第二层（数据链路层）的通信协议来解决，因此，在观看动画时，也可以看到双向 Frame 传送成功的情况。

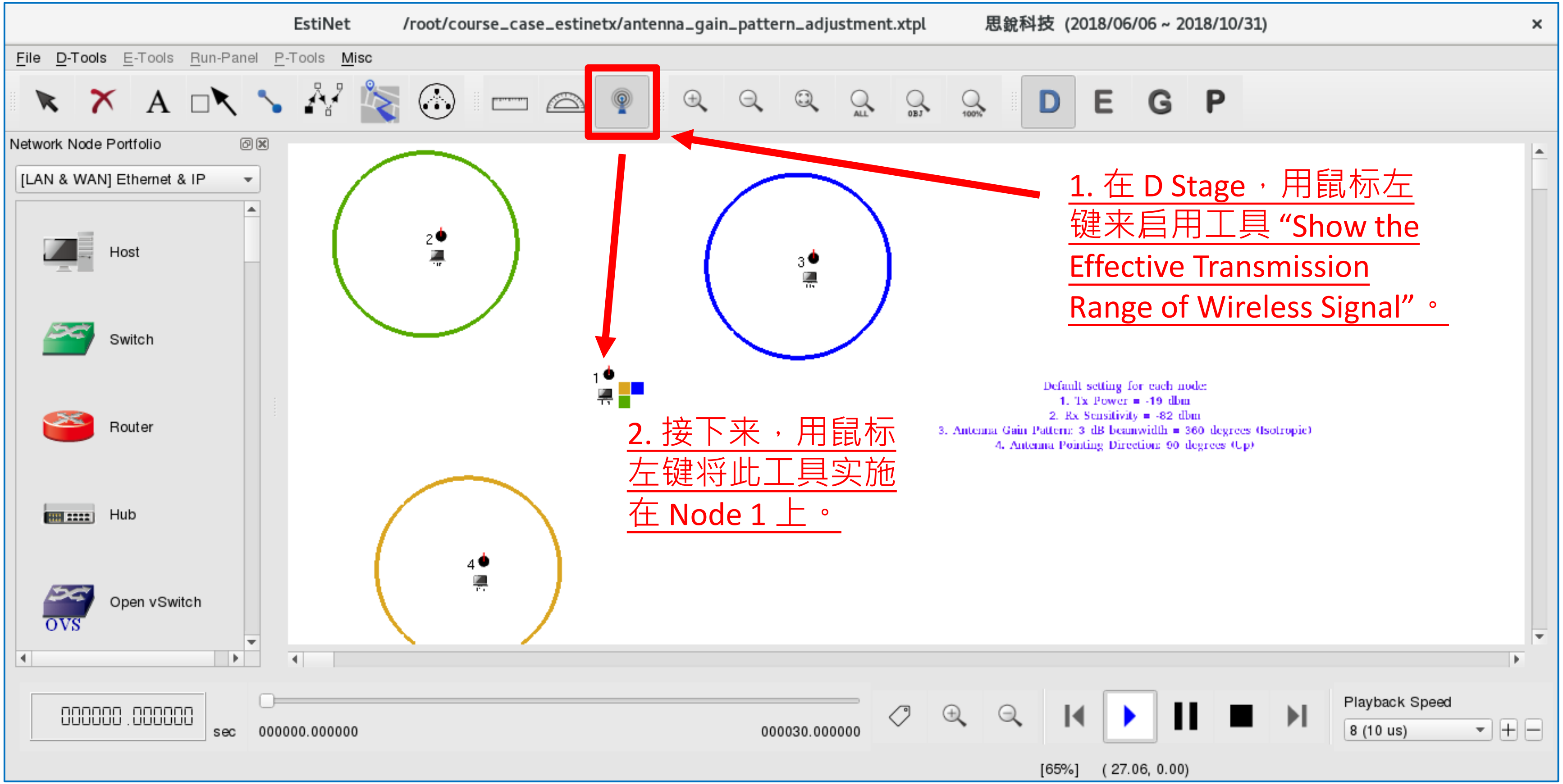


# 增加接收端的信号接收灵敏性（RX SENSITIVITY）来增大信号传递范围

# [步骤一] 将仿真参数设定成回预设值

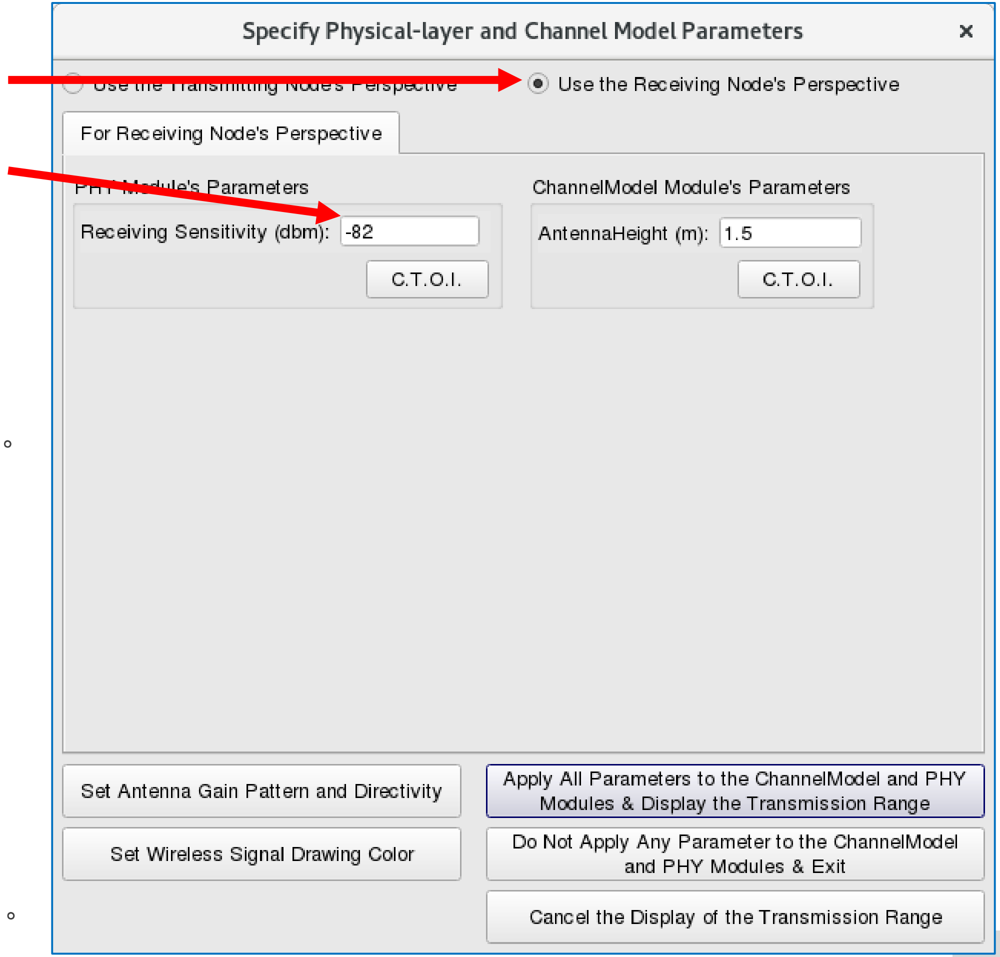
- 上一个实验将所有节点（Node 1, 2, 3, 4）的 ”TransPower (dbm)” 由 -19 改成 -10，使用同样的操作方式，先将此数值由 -10 改回原来的 -19，再继续进行这个实验。

# [步骤二] 启用"显示有效信号传递范围"的工具



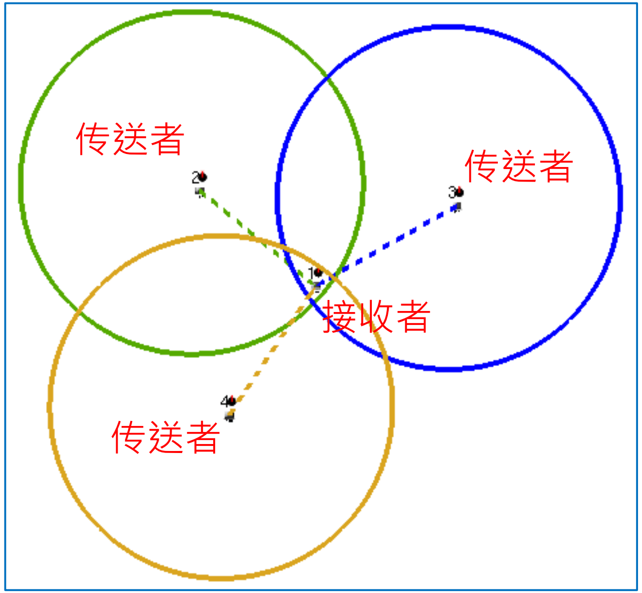
# [步骤三] 调整信号接收灵敏性

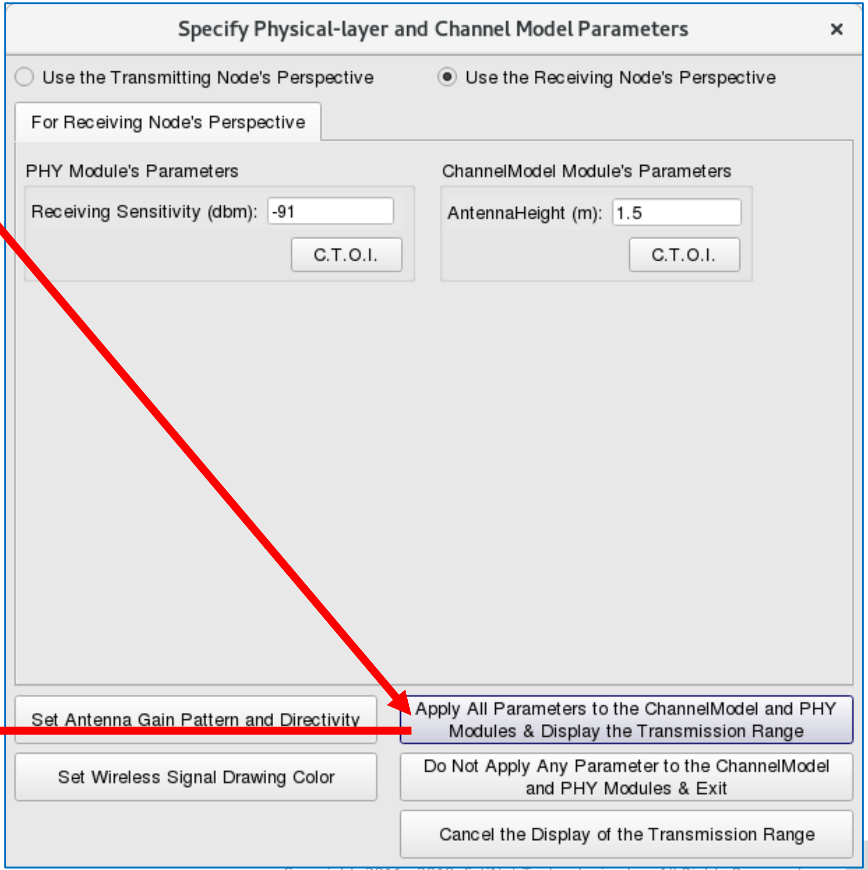
- 在 Node 1 的弹出视窗中，选择以信号接收者的观点，下方的参数设定页面的内容会自动转换成接收者可设定的参数。
- 将 ”Receiving Sensitivity (dbm)” 的数值由 -82 改成 -91，这个参数是表示信号可被接收的最低能量值。信号由传送端送出后，在空气中传递的过程中，会有能量衰减的现象，当信号到达接收端的时候，所剩馀的能量必须大于接收端可感应到的最低信号能量值，才能被接收端所接收到。
  - 分贝毫瓦 (dbm) 单位是经由对数运算而来。
  - 若 RxPower_mW (milliwatt，毫瓦) 是可接收信号的最低能量，则 RxPower_dbm = $10 * \log_{10}\left({}^{RxPower\_mW}/_{1\ milliwatt}\right)$。
  - 换句话说，分贝毫瓦是原可接收信号最低能量值相对于 1 毫瓦的比例再取对数后的结果。
  - 因此，-82 分贝毫瓦表示可接收信号的最低能量值是 63.1*10^(-10) 毫瓦，而 -91 分贝毫瓦表示可接收信号的最低能量值是 7.94*10^(-10) 毫瓦，后者的值低了将近 7.94 倍。也就是说，信号可以传递的更远（能量更低），仍可以被接收端给接收到。



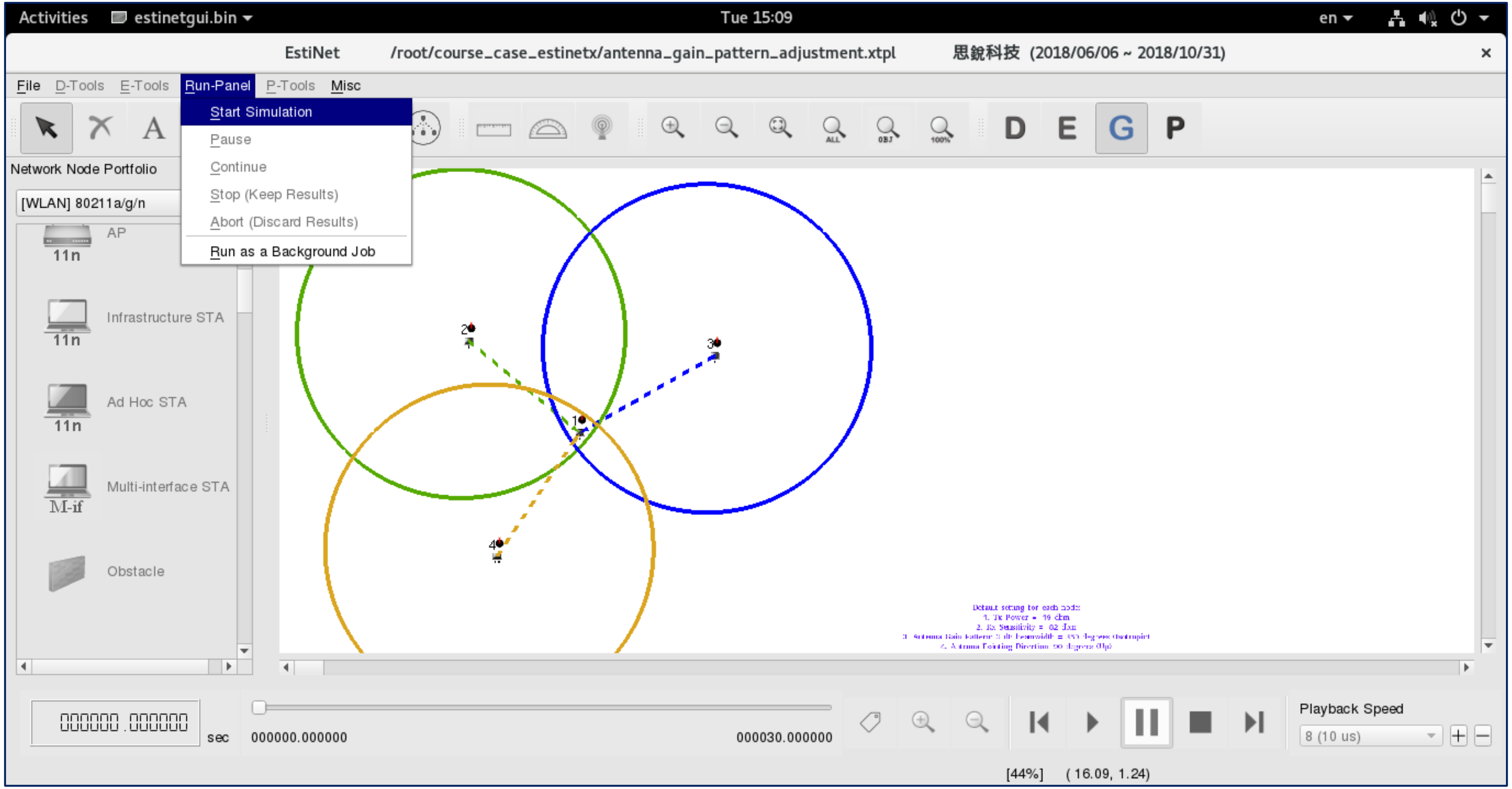
# [步骤四] 观察其它三个 Node 所发送的信号是否能被 Node 1 所接收

- 按下参数套用按键来观看结果。



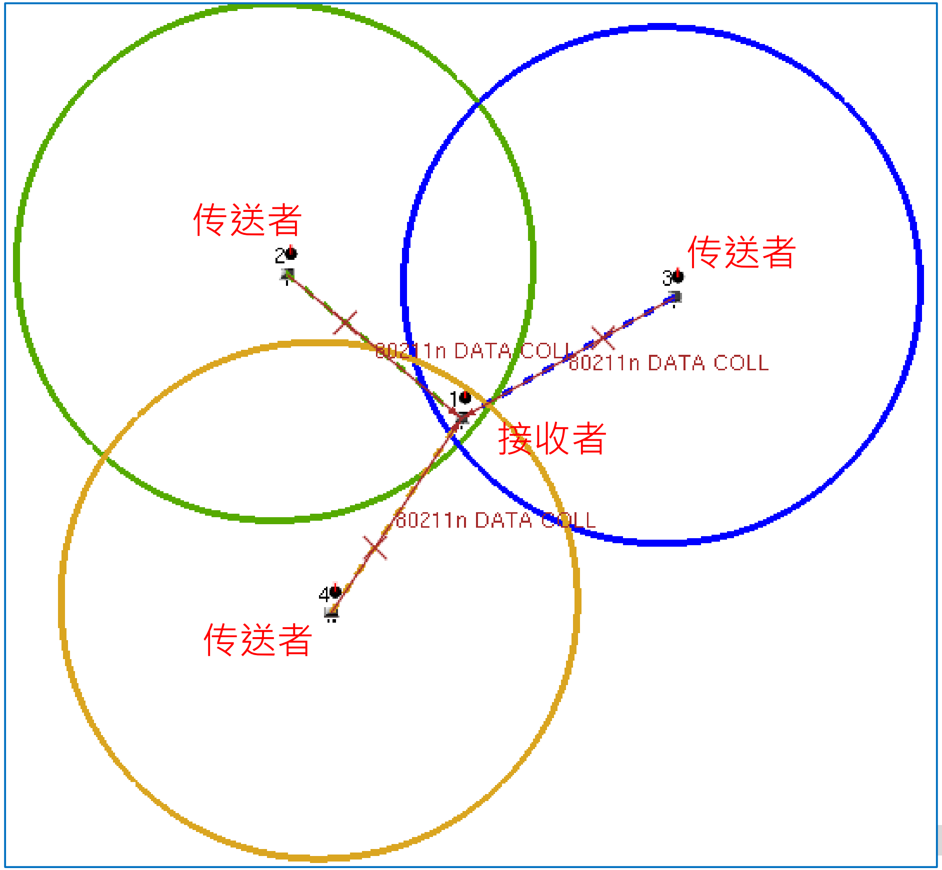



# [步骤五] 切换到 G Stage 产生仿真设定档，然后执行仿真



# [步骤六] 仿真完成后，在 P Stage 观看结果

- 在 P Stage，根据仿真结果来展示 Frame 传送动画，如右图所示，Node 2、Node 3 与 Node 4 所送出的信号可以到达 Node 1，但因为三个信号几乎同时间到达，所以造成信号碰撞的情况，导致 Node 1 没有成功收起任何 Frame。
- 信号碰撞的问题，会由网络第二层（数据链路层）的通信协议来解决，因此，在观看动画时，也可以看到有单向 Frame 传送成功的情况 (Node 2/3/4 to Node 1)。



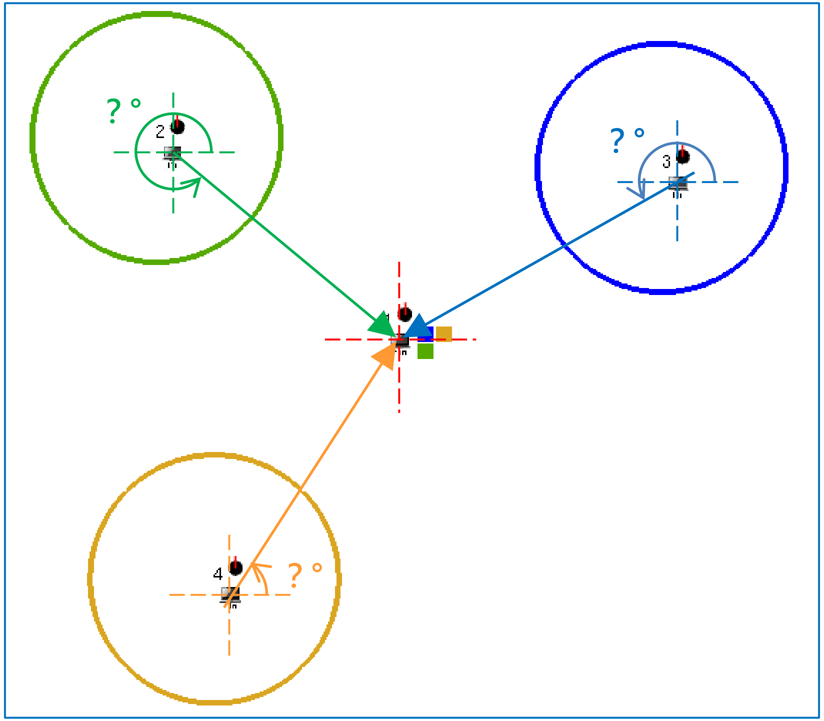
# 调整传送端的天线增益方向图（ANTENNA GAIN PATTERN）来增加信号传递范围

# [步骤一] 将仿真参数设定回预设值

- 上一个实验将 Node 1 的 ”Receiving Sensitivity (dbm)” 由 -82 改成 -91，使用同样的操作方式，先将此数值由 -91 改回原来的 -82，再继续进行这个实验。

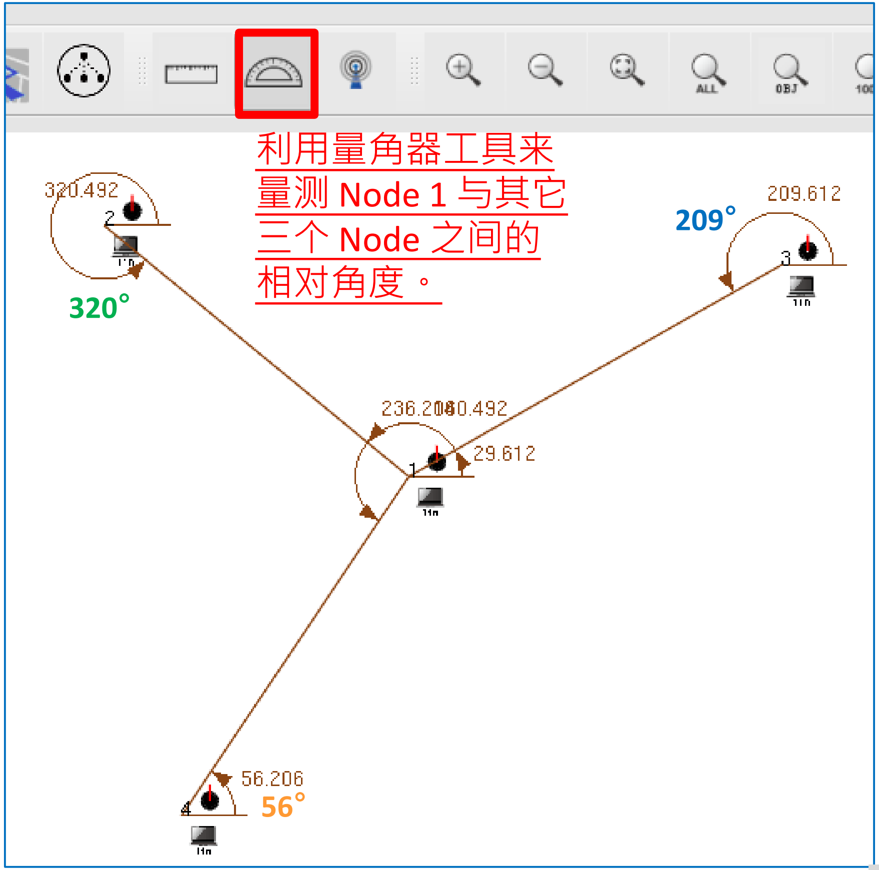
# [步骤二] 调整天线增益方向图来增加信号传递范围的原理

- 预设的天线增益方向图 (Antenna Gain Pattern) 是全向性 (Isotropic) 的，全向性的天线会将信号能量往四面八方送出 (360°)。
- 若是使用有向性 (Directional) 天线，信号能量会集中在特定方向送出，因此，若用来传递信号的能量没有改变，则有向性天线的信号传递距离在特定的方向上会比全向性天线来的更远。
- 因此，我们要将 Node 2, 3, 4 的天线参数改变成有向性天线，使它们送出的信号可以到达 Node 1。

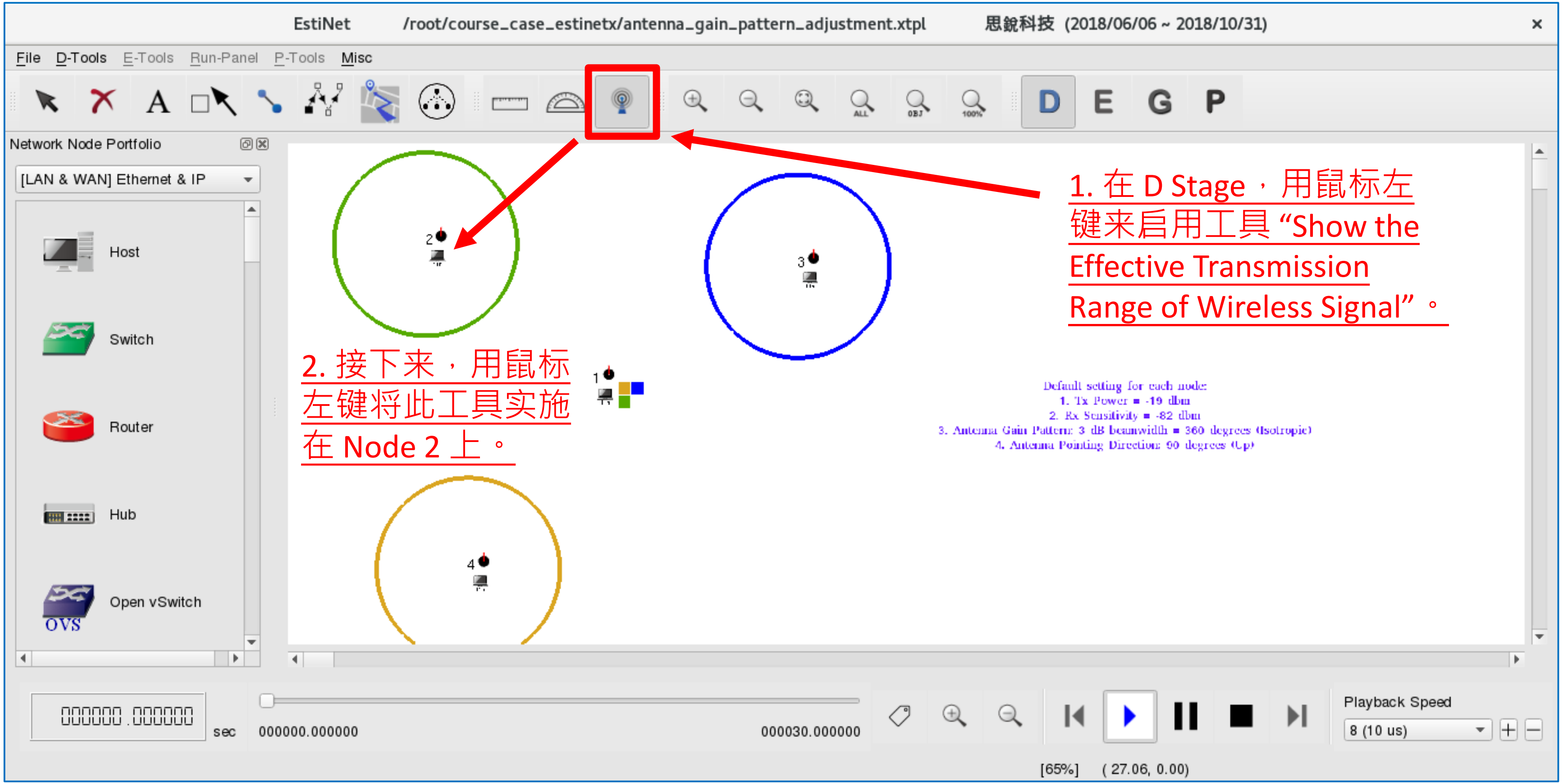


# [步骤三] 量测需要调整的天线的朝向角度

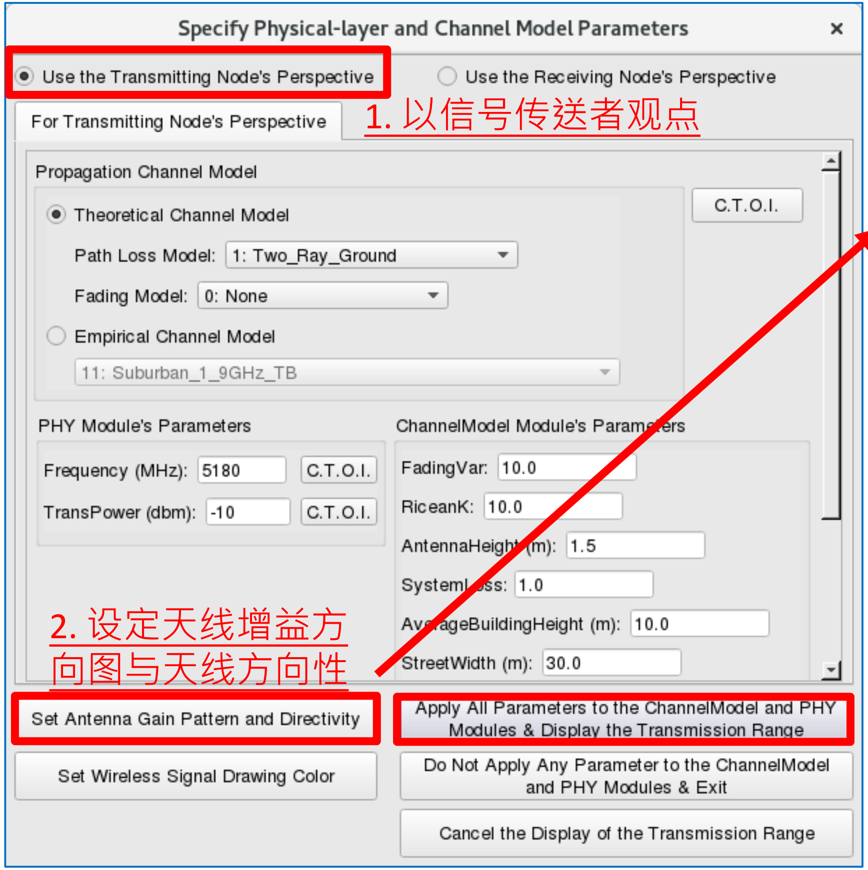
- 利用工具栏中的量角器工具，就可以量测到 Node 2, 3, 4 个别与 Node 1 的相对角度。
  - Node 2 对 Node 1 → 320°
  - Node 3 对 Node 1 → 209 °
  - Node 4 对 Node 1 → 56 °

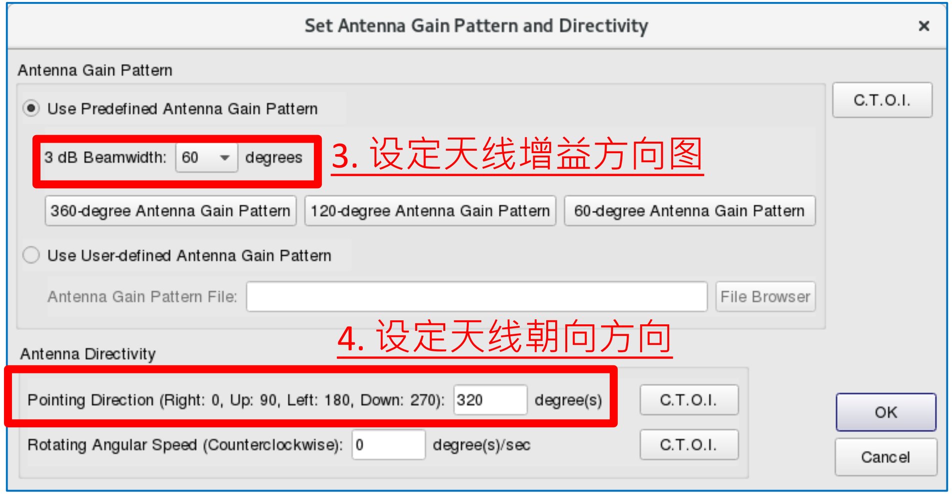


# [步骤四] 启用"显示有效信号传递范围"的工具

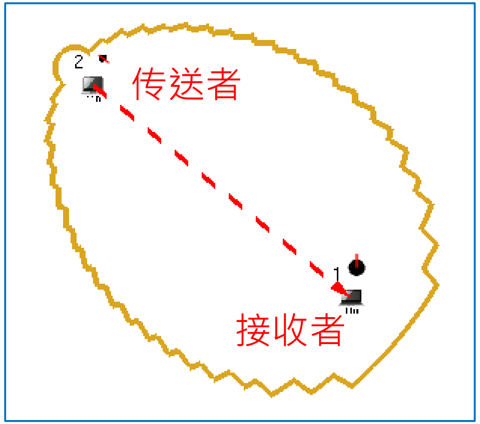


# [步骤五] 调整 Node 2 的天线增益方向图与天线朝向角度

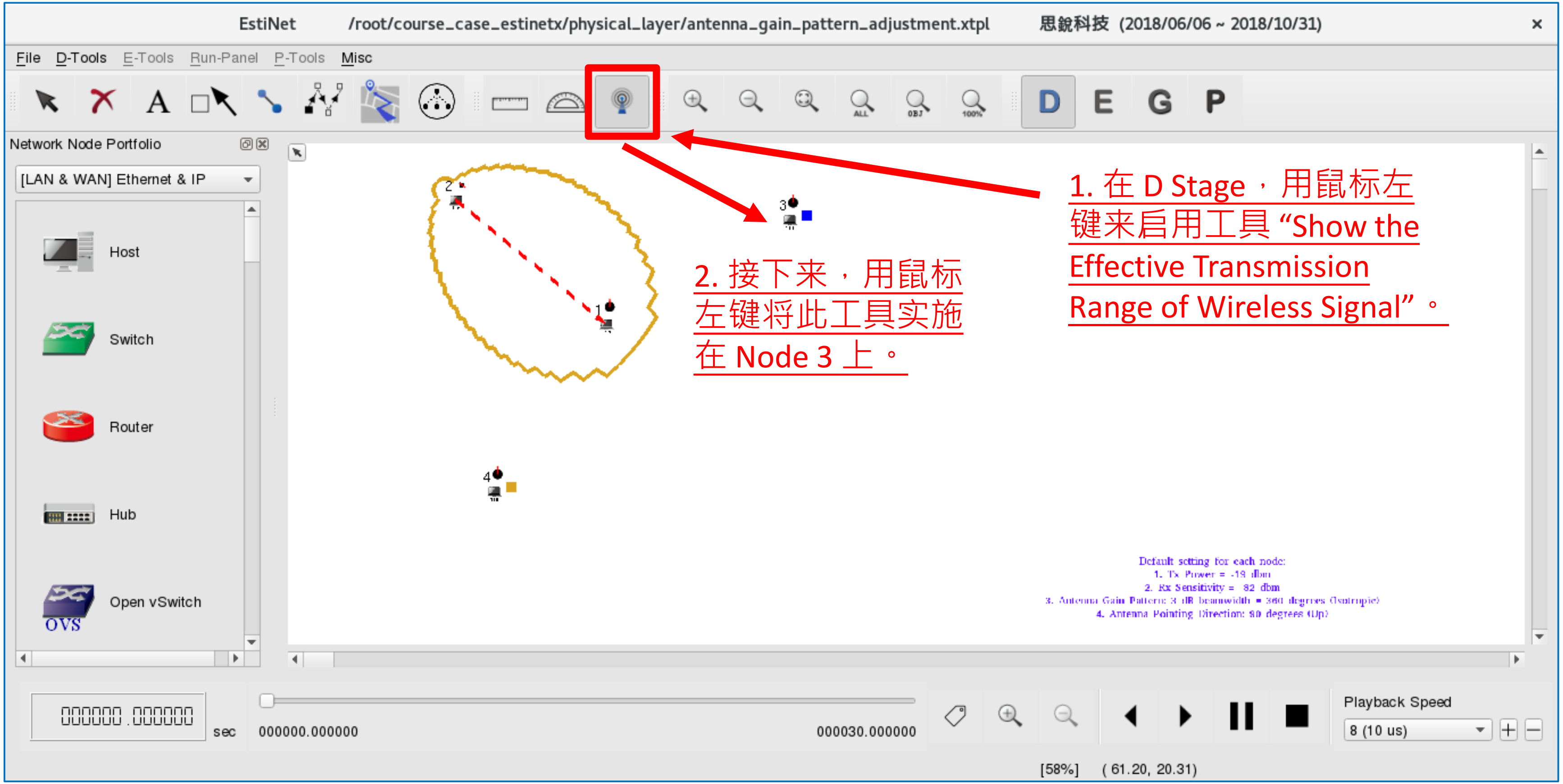


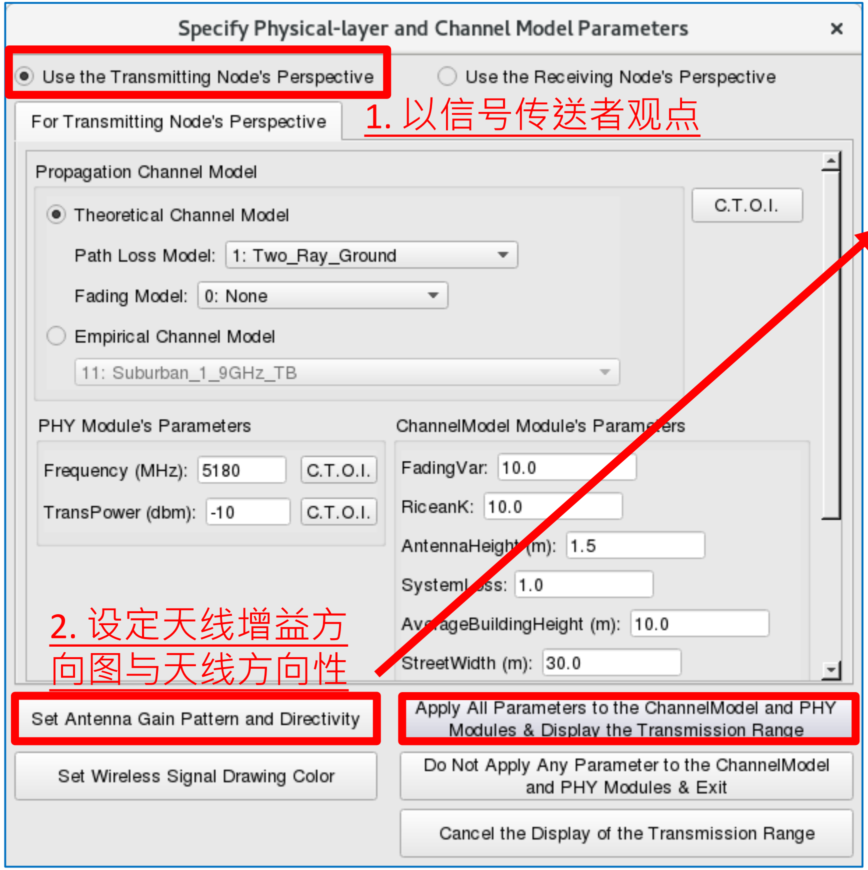

5. 按下参数套用按键来观看结果

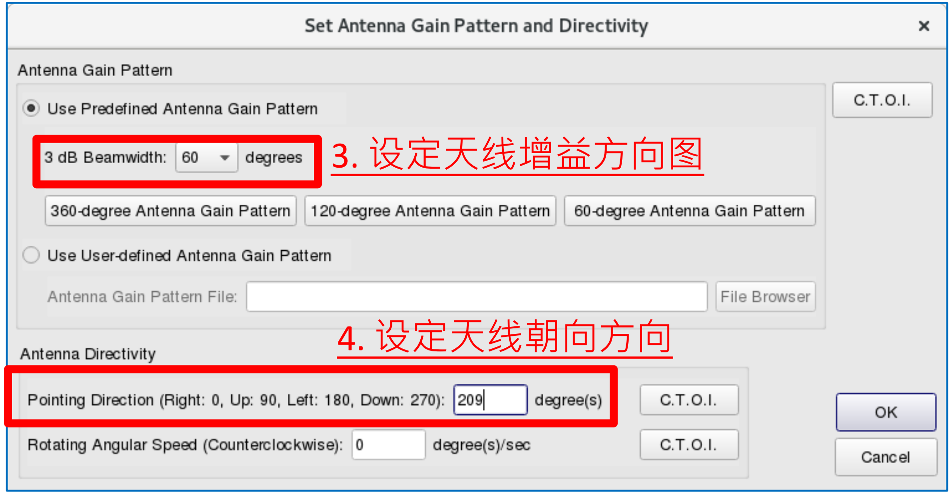


# [步骤六] 启用”显示有效信号传递范围”的工具

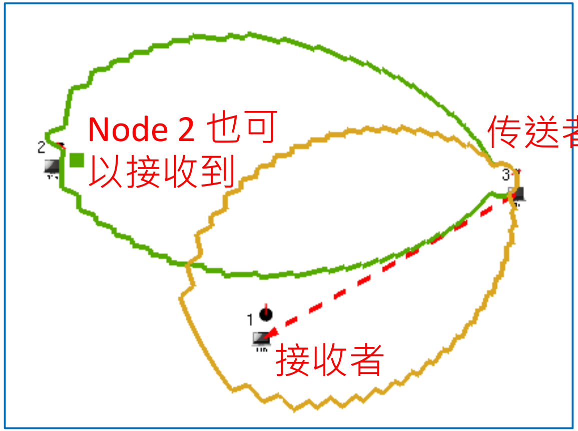


# [步骤七] 调整 Node 3 的天线增益方向图与天线朝向角度

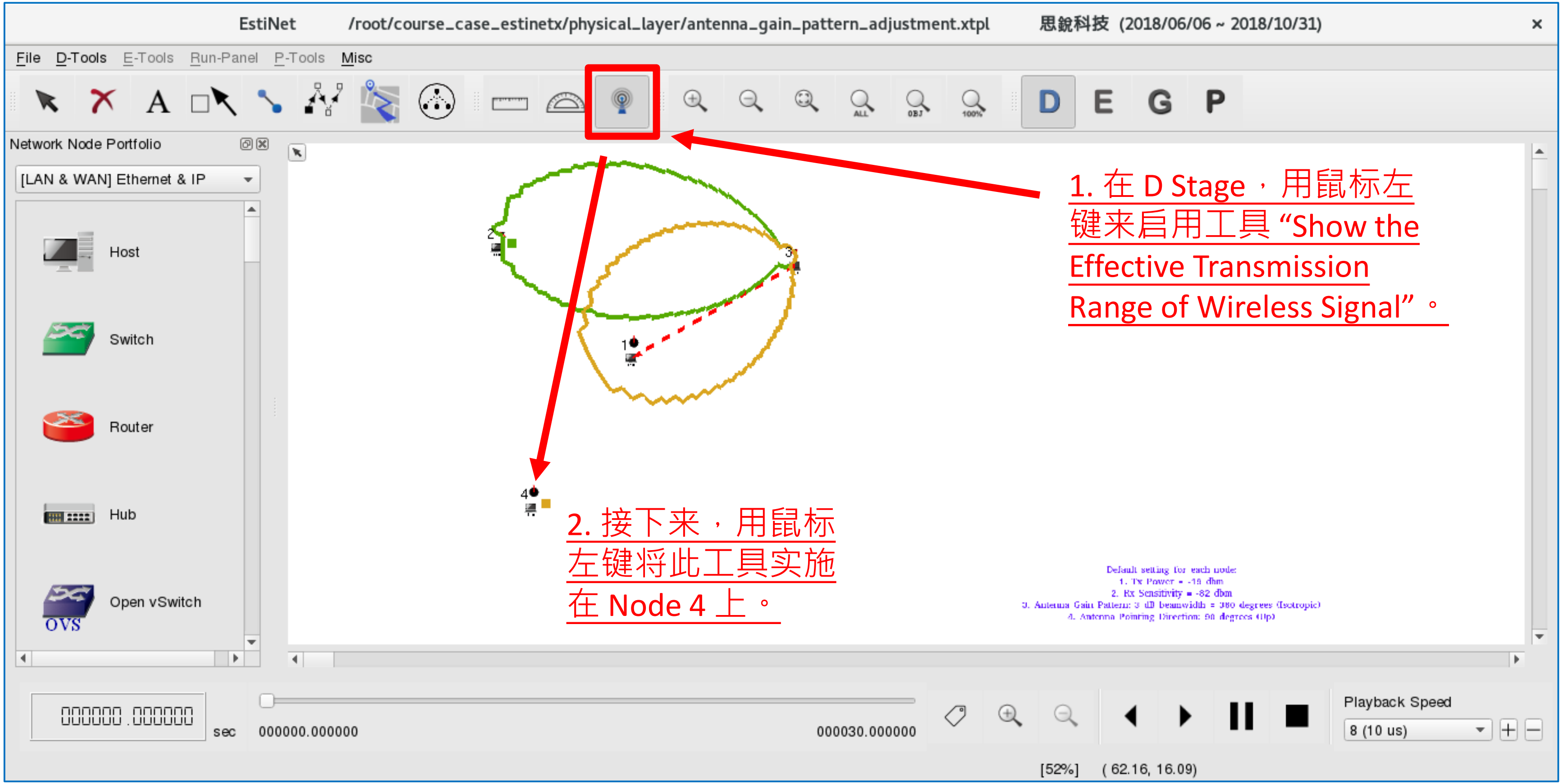


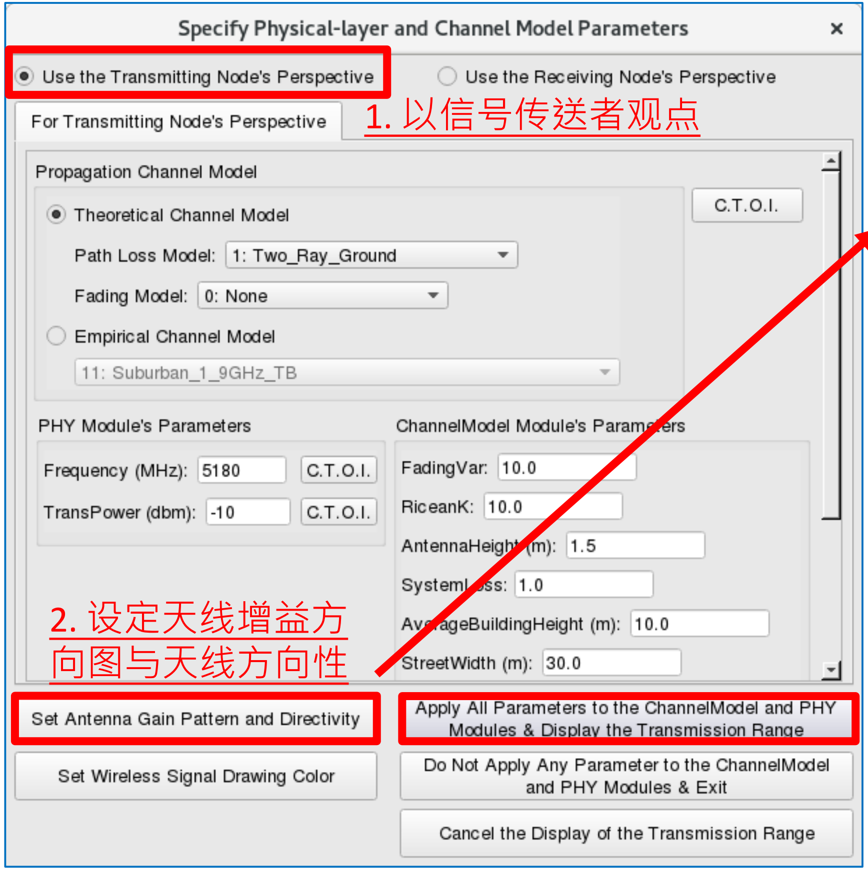

5. 按下参数套用按键来观看结果



# [步骤八] 启用"显示有效信号传递范围"的工具

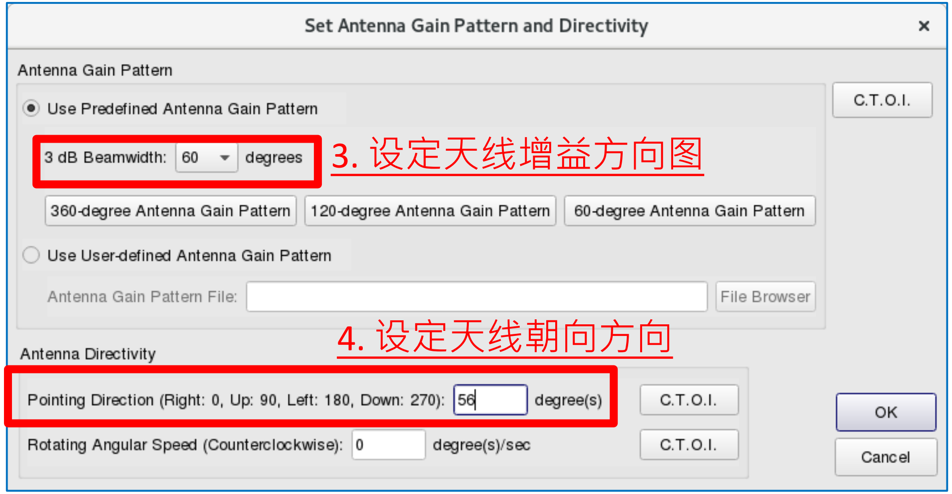


# [步骤九] 调整 Node 4 的天线增益方向图与天线朝向角度




1. 以信号传送者观点

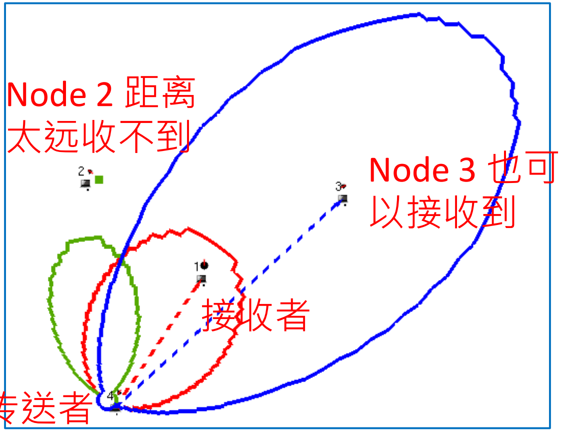
2. 设定天线增益方向图与天线方向性



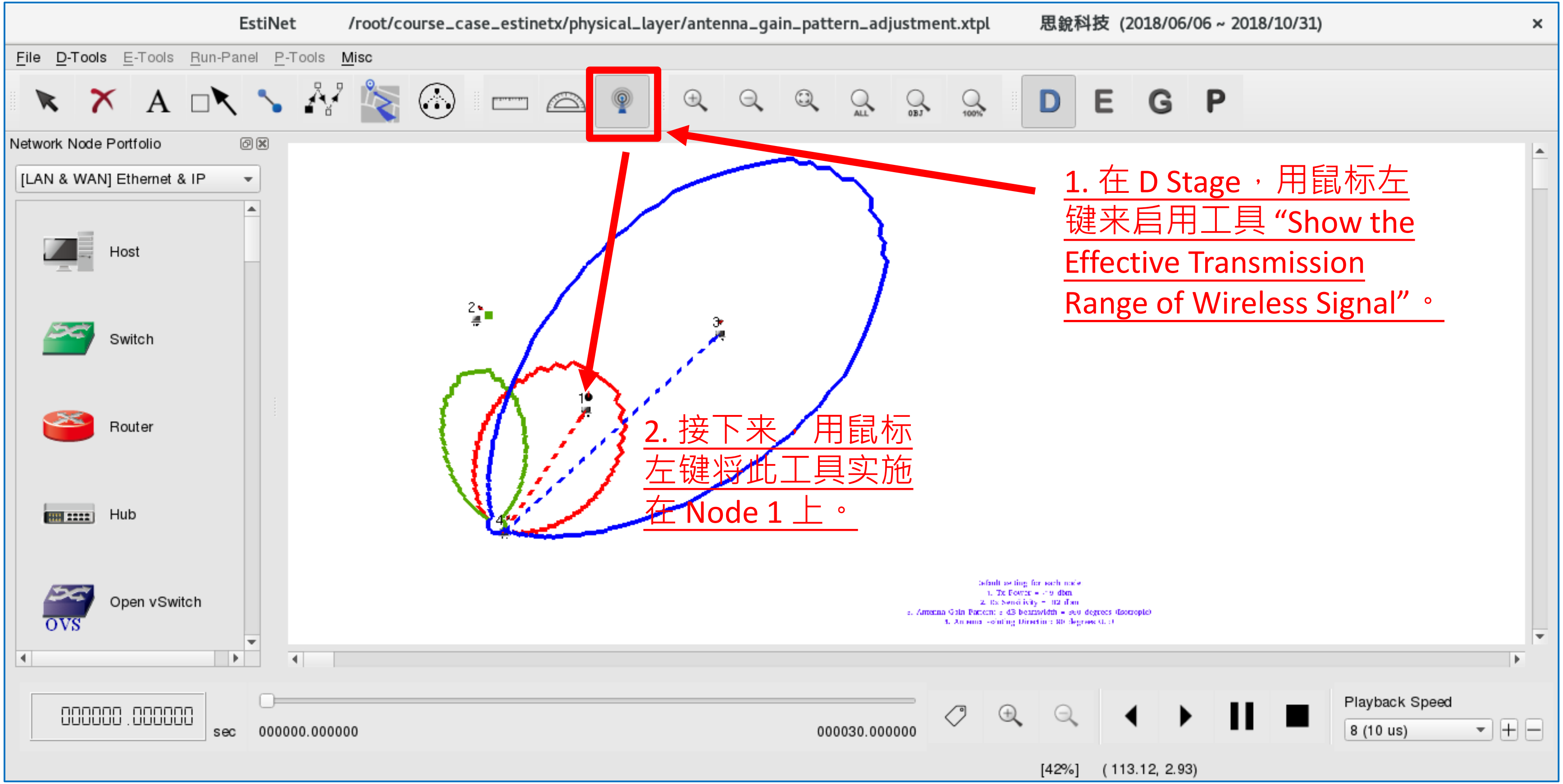

3. 设定天线增益方向图

4. 设定天线朝向方向

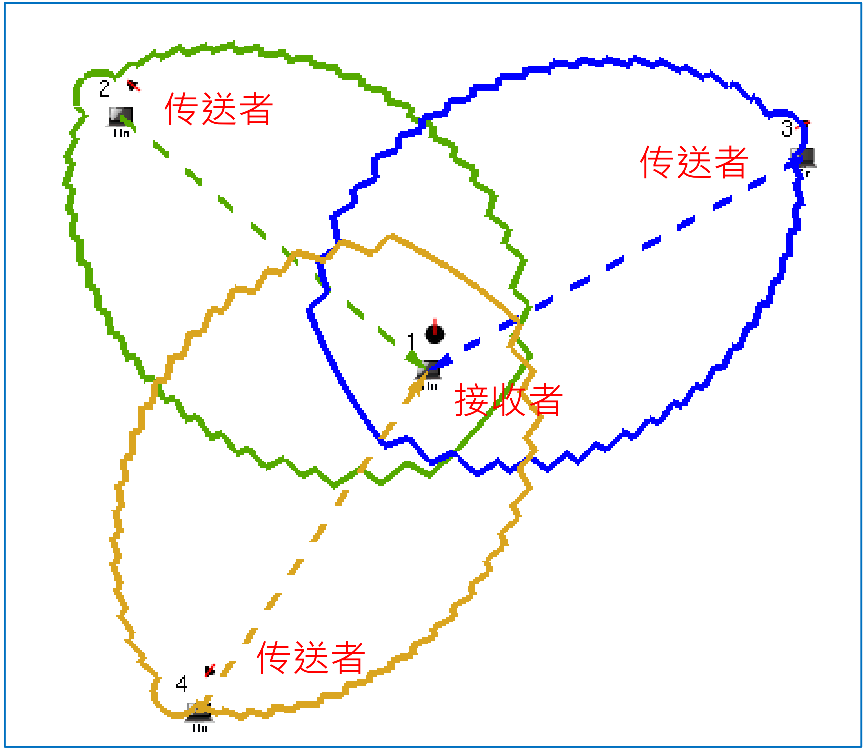
5. 按下参数套用按键来观看结果



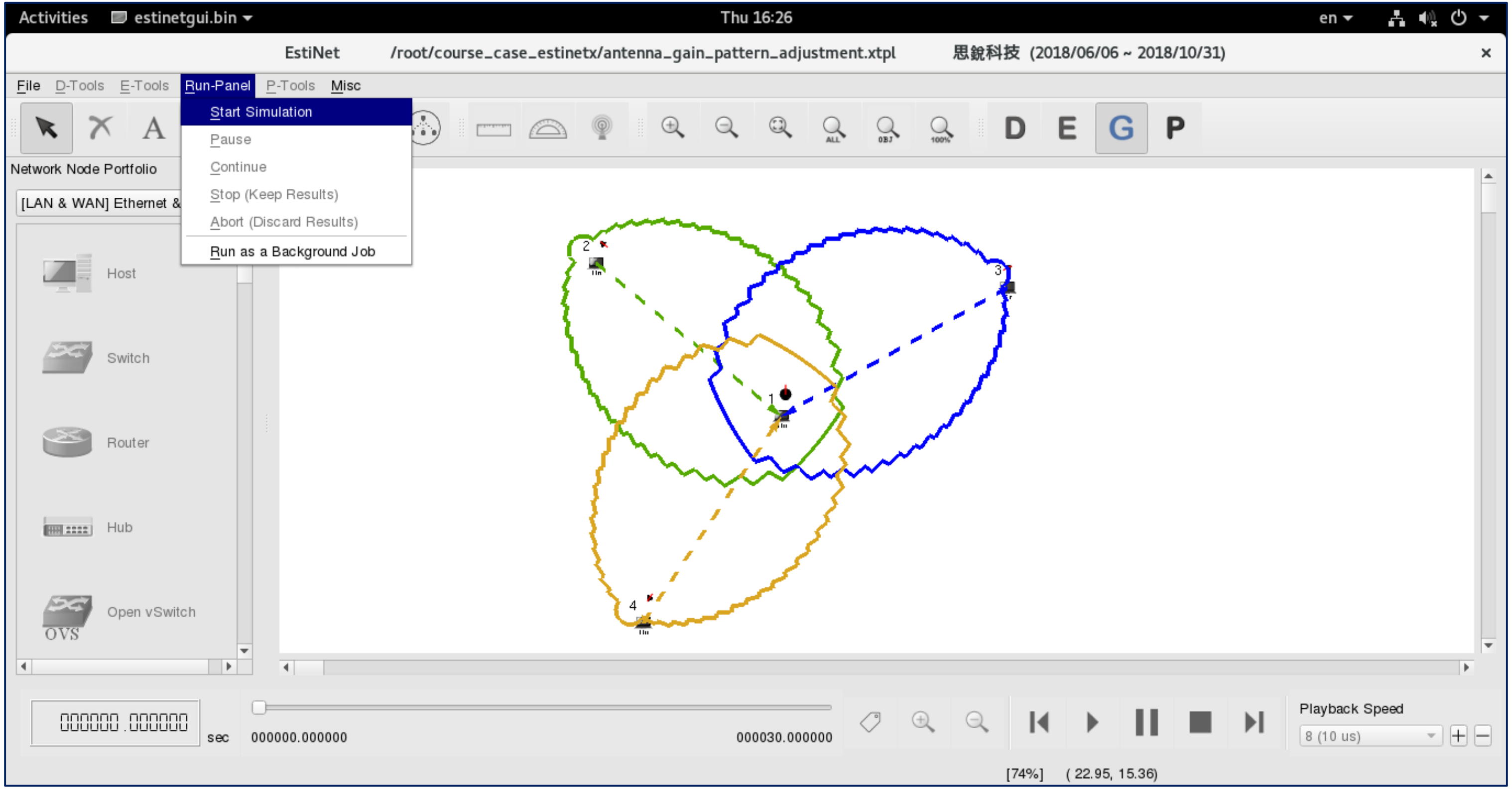
# [步骤十] 启用"显示有效信号传递范围"的工具



# [步骤十一] 在 Node 1 上观看其它三个 Node 经过调整后的信号传递范围

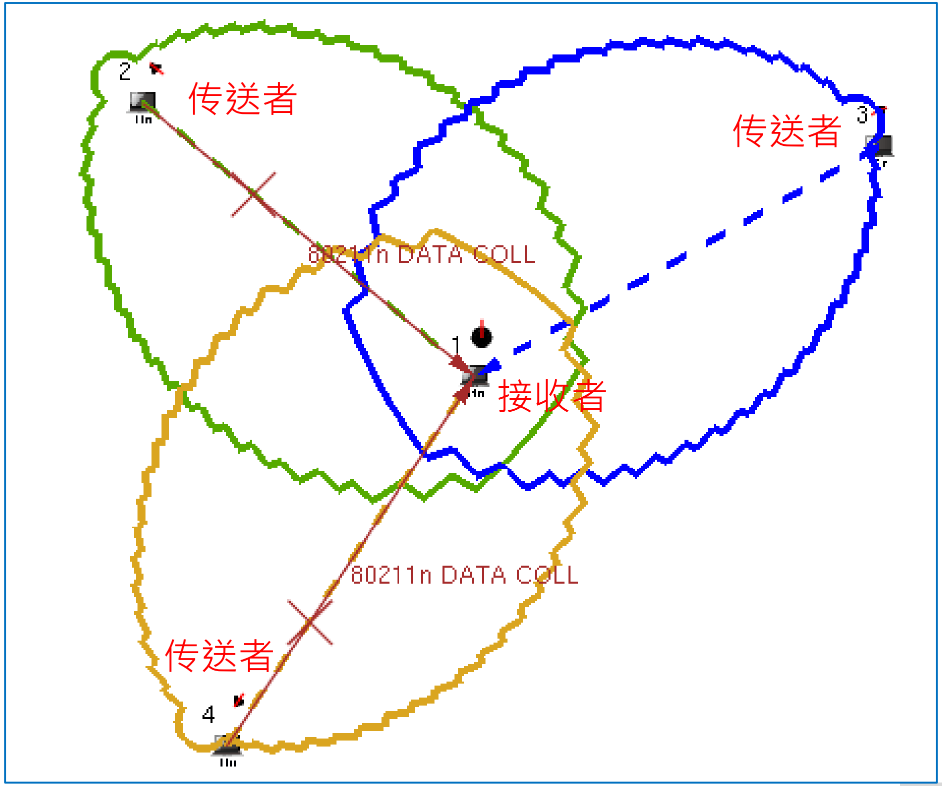



# [步骤十二] 切换到 G Stage 产生仿真设定档，然后执行仿真



# [步骤十三] 仿真完成后，在 P Stage 观看结果

- 在 P Stage，根据仿真结果来展示 Frame 传送动画，如右图所示，Node 2、Node 3 与 Node 4 所送出的信号可以到达 Node 1，若有一个以上的信号几乎同时间到达，就会造成信号碰撞的情况。
- 信号碰撞的问题，会由网络第二层（数据链路层）的通信协议来解决，因此，在观看动画时，也可以看到有双向 Frame 传送成功的情况。



# 调整信号频率与环境参数来观察信号传递范围

# 设定信号频率与环境参数（地形、地物、天气、干扰等等）

- 在制定通信规格（包含网络第一层物理层与第二层数据链路层）时，就会根据通信频道 (Channel) 所要实施的环境参数 (Environmental Parameters) 与所要提供的资料传输率 (Data Rate) 来决定信号频率 (Frequency) 与带宽 (Bandwidth)。
- EstiNet X 仿真器目前所提供的无线信号的仿真，在信号频率的选择上，是根据已经制定完成的 802.11a/g/p/n 等标准的通信规格，若随意调整信号频率到规格之外的频率，会让仿真结果失真，因此，目前暂时不支持可调整任意信号频率的实验。
- EstiNet X 仿真器目前有支持多种环境参数的设定，可配合十多种不同的频道模型 (Channel Model) 来使用，但这部分的操作需先具备进阶的专业知识，在此暂时不涉入进阶的领域。而仿真器预设的频道模型采用的是理论上的 Two Ray Ground，另可改选 Free Space 与 Free Space and Shadowing，目前也暂时不进行这方面的实验。